Protecting Vaccine Safety: An Improved, Blockchain-Based, Storage-Efficient Scheme

Laizhong Cui, *Senior Member, IEEE*, Zhe Xiao, Fei Chen, *Member, IEEE*, Hua Dai, *Member, IEEE*, and Jianqiang Li

***Abstract*—In recent years, vaccine safety incidents have occurred frequently. To protect vaccine safety, researchers have proposed to use blockchain to secure the vaccine circulation process. Technically, blockchain has some limitations in solving vaccine and other supply chain problems, such as large on-chain storage consumption and low throughput. To better alleviate these restrictions, we propose an improved, blockchain-based, storage-efficient vaccine safety protection scheme in this work. Specifically, we first model the vaccine circulation process. We then design a system to protect vaccine circulation using blockchain, cloud, and cryptographic mechanisms. The proposed system leverages the cloud to implement the vaccine circulation model. Correspondingly, it uses the blockchain to store circulating data certificates and signatures. We evaluated the proposed conceptual model using a consortium blockchain. The experimental results show that the proposed system is efficient.**

***Index Terms*—Blockchain, cloud computing, digital signature, smart contract, vaccine safety.**

I. Introduction

IN RECENT years, vaccine safety incidents have attracted people's attention. The reported fraud and data tampering in the vaccine supply chain is causing worries. The vaccine supply chain has several potential risks [1]. First, manufacturers could produce unqualified vaccines or even modify vaccine data. Second, unqualified intermediate suppliers could collude with vaccination institutions (VIs) to sell and use fake/unqualified vaccines. Third, vaccine transportation fails to comply with strict cold chain requirements.

Compared with other product supply chain management issues, vaccine supply chain management is more urgent and worrying. The current vaccine supervision system uses the traditional centralized information management system. Traditional information management systems face potential security problems, such as easy tampering and a single point of failure. Each entity in the vaccine circulation process also needs to have its own database to store the data, which causes data islands. Thus, it is not easy to form an effective closed loop of trust in the entire supply chain. Because the blockchain is able to build a global and distributive trust, using blockchain to solve the trust issue becomes a promising approach. Blockchain provides distributed data storage. The entire distributed network jointly maintains a globally unique chain. The data on the chain can only be added and modified through consensus mechanisms [2]–[6]. The blockchain also maintains the system's consistency without relying on the central node. This feature makes the blockchain tamper proof. The decentralized architecture of the blockchain has also become a new form of distributed software system design.

The state-of-the-art research in blockchain enabling vaccine safety includes [7]–[10]. These works model the vaccine circulation process using different blockchain infrastructures (such as Ethereum, Hyperledger, FISCO BCOS, etc.) and different blockchain architectures (single chain, double chain, etc.) to integrate vaccine circulation data on the chain to protect the safety of the vaccine supply chain. Due to the blockchain infrastructure's performance limitation, these systems face large data expansion and system delay problems. The blockchain network requires each node to keep the same copy of the data. The amount of data in the vaccine supply chain is huge, which will eventually lead to excessive storage pressure on each node. Simultaneously, due to the throughput limitation of the blockchain infrastructure, transactions in a certain period of time cannot be uploaded to the chain in time, resulting in system delays.

Aiming at better performance, we have made a new research attempt. We design a blockchain-cloud-based vaccine traceability system to protect vaccine circulation safety. At a high level, we model the blockchain as a globally unique database. We use the blockchain to store the digital digest of vaccine circulation data and different entities' digital signatures in the vaccine circulation process. They

Manuscript received 12 April 2021; revised 28 August 2021 and 5 December 2021; accepted 25 March 2022. Date of publication 13 April 2022; date of current version 17 May 2023. This work was supported in part by the National Natural Science Foundation of China under Grant 61872243, Grant 61872197, Grant 61772345, and Grant U1713212: in part by the National Key Research and Development Plan of China under Grant 2018YFB1800302 and Grant 2018YFB1800805; in part by the Shenzhen Science and Technology Program under Grant RCYX20200714114645048, Grant JCYJ20190808142207420, and Grant GJHZ20190822095416463; in part by the Pearl River Young Scholars funding of Shenzhen University; in part by the Natural Science Foundation of Guangdong Province-Outstanding Youth Program under Grant 2019B151502018; in part by the Guangdong Basic and Applied Basic Research Foundation under Grant 2020A1515011489; and in part by the Project from Webank and SZU-Webank Fintech Research Institute. This article was recommended by Associate Editor L. Tang. *(Corresponding author: Fei Chen.)*

Laizhong Cui, Zhe Xiao, Fei Chen, and Jianqiang Li are with the College of Computer Science and Software Engineering, the National Engineering Laboratory for Big Data System Computing Technology, and the Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ), Shenzhen University, Shenzhen 518060, China (e-mail: cuilz@szu.edu.cn; 1810272052@email.szu.edu.cn; fchen@szu.edu.cn; lijq@szu.edu.cn).

Hua Dai is with the School of Computer Science & Technology, Nanjing University of Posts and Telecommunications, Nanjing 210049, China (e-mail: daihua@njupt.edu.cn).

Digital Object Identifier 10.1109/TCYB.2022.3163743

serve as evidence for tracing vaccine safety incidents. We employ the cloud to provide vaccine data storage, circulation management, and vaccine tracing. The cloud serves to alleviate blockchain storage requirements and accelerates system processing.

The proposed scheme covers the four vaccine circulation processes, that is: 1) production; 2) sales; 3) transportation; and 4) distribution. From the beginning of vaccine production, each vaccine is assigned a digital identity (i.e., vaccine trace code) to uniquely identify a vaccine. Vaccine trace code represents the endorsement of regulatory agencies and manufacturers. It is essential to vaccine safety throughout the vaccine circulation process. The involved entities will query the blockchain at each circulation step to verify the correctness of the vaccine data before entering the next circulation step. Once a vaccine incident happens, the system supervisor (i.e., the national authority) will query the blockchain to track the vaccine's source and then conduct timely prevention and accountability.

Technically, there exist two key challenges in the proposed scheme. The first challenge is how to ensure the reliability of vaccine circulation data. The second challenge is how to relieve the pressure of blockchain storage and communication.

For the first challenge, the blockchain can only guarantee that the chain's data cannot be tampered with, but cannot verify the reliability of the data. Our basic solution is as follows. By adding a unique digital identity to each vaccine, we bind each vaccine to the vaccine circulation process's digital identity. The system supervisor is responsible for issuing digital identities. In each circulation step, the involved entity needs to verify each other and sign the transactions. Due to vaccine safety regulations, each entity in the circulation process is responsible for validating the received vaccine data before processing the received vaccine. Only all the data are correct, the vaccine circulation proceeds. Because we use digital identities to track vaccines and each entity's signature serves as an endorsement of the vaccine circulation process, this solves the data reliability challenge.

The second challenge requires us to decouple the blockchain infrastructure's performance limitations from the overall system efficiency. To solve this, our idea is to tradeoff computation for storage on the blockchain. As mentioned earlier, the cloud provides management and traceability services for vaccine circulation; the blockchain only stores the data digest and involved entities' signatures of the circulation data. We structure and store the vaccine circulation data in the cloud into a Merkle tree. We store the root of the tree on the chain as evidence. The involved entity signs the root of the tree after verifying the vaccine data. Meanwhile, we use an aggregate signature scheme to compress multiple users' signatures of the same tree into one signature. This realizes the decoupling of the blockchain performance limitation from the overall system efficiency. We compress multiple vaccine circulation data into a much smaller piece of data that is later stored on the blockchain. The bottleneck of the system is thus no longer the throughput of the blockchain but the system's computing power.

The remainder of the article proceeds as follows. Section II discusses related work on blockchain applications in supply chain management and blockchain scalability research. Section III introduces the vaccine circulation process model and system framework of the proposed scheme. Section IV presents the detailed system design. Section V evaluates and analyzes the experimental performance. Section VI discusses further issues. Section VII summarizes this article.

II. Related Works

We review the recent related research in this section. This work aims to provide a high-performance blockchain solution in the vaccine supply chain management scenario. Thus, we organize the review into two aspects, that is: 1) blockchain for supply chain management and 2) blockchain scalability enhancement.

A. Blockchain for Supply Chain Management

Specific Scenarios: In the milk supply chain scenario, Zhang *et al.* [11] proposed a blockchain-based method to control milk production quality. Milk production includes six steps: 1) harvesting; 2) transportation; 3) storage; 4) testing; 5) processing; and 6) packaging. Blockchain is mainly used to store the source information of milk raw materials. Based on these data, milk production, transportation, and sales processes are tracked throughout the milk supply chain. Once problems with milk raw materials are discovered, the problematic milk's sales channels can be traced back in time. Simultaneously, as a public distributed database, the blockchain increases the data transparency of the supply chain and makes the identification of responsibility easier.

In the steel production supply chain scenario, Cao *et al.* [12] proposed a blockchain-based steel supply chain traceability system. They combined blockchain, sensors, RFID, and GPS technologies to manage and track the production, distribution, consumption, and steel supervision. Each product has an RFID tag, which is the virtual identification of the product in the system. The product traceability system can track product information through the electronic product code system and product RFID tags at distribution nodes. The system can trace the origin of the product from down to top. Data are transparent to consumers; consumers can also confirm product quality. At the same time, products can be tracked from top to down. Once a problem is found, it can be located and dealt with in time. They prototyped and verified the system on the Hyperledger platform.

In the vaccine supply chain scenario, Peng *et al.* [9] proposed a two-layer blockchain architecture to protect the security of the vaccine supply chain. They focused on the safety of vaccine production. The first layer is the private data of vaccine manufacturers; the second layer is public data, including production record hashes and vaccine information. Through the two-tier architecture, manufacturers can submit production records in time and avoid exposing some sensitive information. At the same time, they also proposed a cutting mechanism for vaccine data. To save the blockchain space, the

vaccine data is cut according to the timestamp and the vaccine validity period.

General Design Methods: Westerkamp *et al.* [13] proposed a distributed supply chain management system based on blockchain smart contracts. In their proposed system, commodities are represented by tokens. The token can be transferred, split, merged, and authenticated. Tokens represent the circulation of commodities in various roles in the network, reflecting how commodities are circulated in reality. They implemented the system in Ethereum and evaluated it. The system has strong scalability and flexibility. The tokens' transaction records are stored on the blockchain, which is difficult to tamper with and promotes goods' traceability. One potential concern is the reliability of the mapping relationship between tokens and commodities. It is thus critical to guarantee the accuracy of the input data.

Waltonchain [14] was created by combining the blockchain technology with RFID to achieve effective IoT integration. The system tracks the circulation process of products in the supply chain by implanting RFID tags and reader control chips into products. Then, the system stores the information about the product status to a blockchain for analysis.

To summarize, the general method of applying blockchain to the supply chain management problem is similar. Apart from the difference of the selected underlying blockchain technology, blockchain application's main problems can be divided into two aspects, that is: 1) what data are stored in the blockchain and 2) how to store it. In supply chain management, key information related to supply chain products' quality and traceability should be stored. It is possible to store the data directly on the blockchain as in the existing research. However, system performance should be considered. It is worth noting that in addition to supply chain management, blockchain is also helpful in secure data storage for industrial IoT [15], enhancing reliable authentication [16], [17], and securing payment systems [18].

B. Blockchain Scalability

How to enhance blockchain scalability has attracted many research efforts. Existing research for boosting blockchain scalability could be divided into on-chain scalability and off-chain scalability research, respectively. The on-chain scalability can be enhanced from the data layer, network layer, and consensus layer.

The simplest data layer scalability solution is to increase the block size and reduce the block generation interval directly. The representative algorithms are Bitcoin-NG [19], Hybrid Consensus [20], etc. This method comes at the expense of safety. Another idea is to change the structure of the blockchain. For example, a round of consensus in GHOST [21] allows two forks conducive to rapid mergers after forks. Although the GHOST protocol in Ethereum is still the longest chain consensus, it continues to expand in this way and eventually becomes a directed acyclic graph (DAG). In the long run, it is no longer the longest chain consensus, but allows multiple chains. The representative DAG algorithms includes [22] and [23].

The main method of network layer expansion is network sharding. Sharding is actually a traditional database technology that divides large databases into smaller, faster, and easier-to-manage parts. These parts are called data shards. In the blockchain, the network can be divided into many smaller parts, or "sharded" processing, such that each small network only needs to run a smaller range of consensus protocols. Transactions on the network will be divided into different shards composed of different nodes on the network. Therefore, each node only needs to process a small part of the incoming transactions while a large amount of verification work can be completed by processing in parallel with other nodes in the network. Splitting the network into fragments allows more transactions to be processed and verified at the same time. The typical work of sharding includes [24] and [25].

The consensus layer's scalability mainly includes BFT consensus [26], [27], PoS consensus [28], hybrid consensus [29], [30], etc. Among them, the BFT consensus algorithm comes from the Byzantine fault tolerance problem of traditional distributed systems. The main idea of consensus layer scaling is to employ some selected nodes in the network to achieve consensus. Different from equal peers in proof-or-work mining, those selected nodes have a larger influence on consensus.

The above blockchain scalability methods are all on-chain methods, which expand the blockchain under strict nontampered and traceable conditions. Off-chain solution is a different research direction. It mainly includes side-chain and off-chain calculations. Off-chain scalability is also called second-layer scalability, which is an application-layer expansion scheme and does not change the public chain's basic protocol. It does not change the blockchain's rules (e.g., block size, consensus mechanism, etc.). Some typical work of off-chain schemes include [31]–[34].

III. Proposed Scheme

This section presents the proposed scheme. We first model the vaccine circulation process and establish a vaccine circulation protection model. We then present an architectural overview of the proposed scheme. Finally, we discuss the assumptions for the proposed system.

A. Vaccine Circulation Process

The vaccine circulation process can be roughly divided into four stages: 1) production; 2) sales; 3) distribution; and 4) transportation. We use the business process model to represent the vaccine circulation process. Fig. 1 shows the traceability process of a batch of vaccines. The vaccine circulation route is regulatory agency/supervisor, manufacturers, provincial center of disease control (CDC), local CDC (e.g., city or village), and vaccination units. Professional transportation companies undertake the transportation of vaccines.

The process begins when the manufacturer submits a batch of vaccine production application to the regulatory agency. The regulatory agency reviews the manufacturer's relevant paperwork (such as production qualifications, trade contracts, orders, etc.). After the review is passed, the relevant

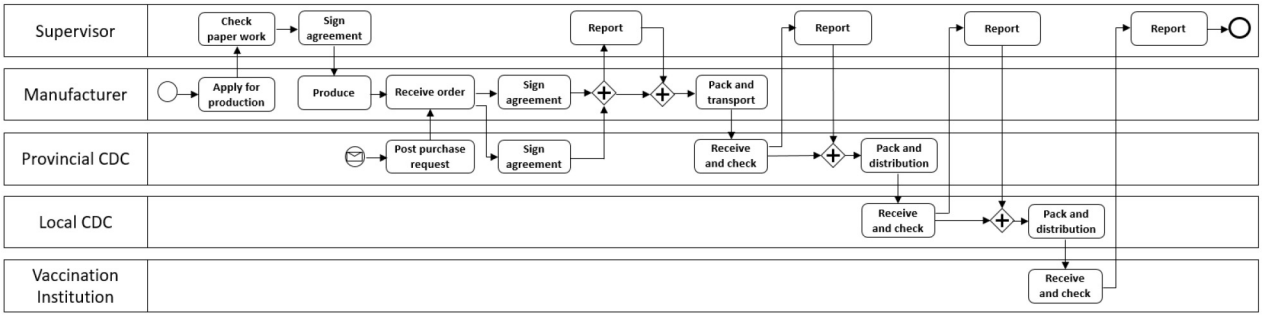


Fig. 1. Vaccine traceability process.

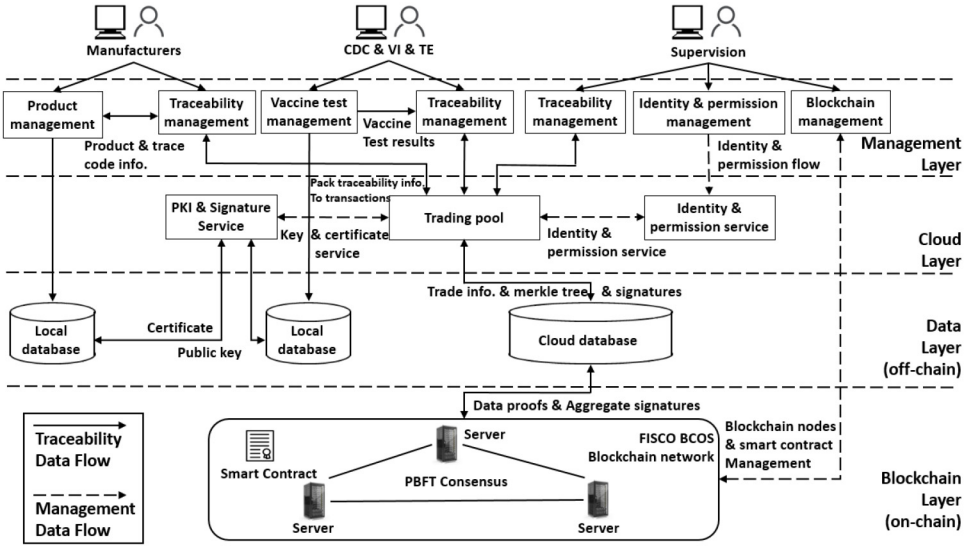


Fig. 2. Vaccine traceability system architecture.

1documents are signed and the trace code is assigned to the batch of vaccines. The trace code uniquely corresponds to the vaccines in one batch, representing the digital identity of each vaccine. Then, the manufacturer starts vaccine production and prepares for receiving purchase orders.

Later, a provincial CDC issues a purchase order request. The manufacturer takes the order. After the provincial CDC and the manufacturer have signed a purchase contract, they need to report to the supervisor. A vaccine traceability system will provide information about the vaccine to both parties. The manufacturer employs a transportation enterprise to send the vaccine. After the provincial CDC receives the vaccine, it needs to check the vaccine's trace code. After completing the trace code verification, the CDC compares the received vaccines with the data that has been entered in the vaccine traceability system to verify whether the vaccine's circulation path is consistent. The same process also occurs when assigning the vaccine from the provincial CDC to the next-level local CDC, and from the local CDC to the final user-oriented VI. These two processes are basically the same as the process from the manufacturer to the provincial CDC. The involved entity receiving the vaccine needs to verify the trace code's correctness and report it to the supervisor.

Throughout the circulation process, the sequence of events and certain processes are dynamic. There is no absolute chronological relationship between signing a purchase contract and conducting production activities. There may be multiple manufacturers, local CDCs, and vaccination units. In Fig. 1, we simplify these details to make the description of the process concise. Furthermore, in each negotiation step, the negotiation's failure will result in the entire process's termination.

B. System Architecture

Fig. 2 presents the architecture of the proposed blockchain-cloud-based vaccine traceability system. It consists of four parts: 1) management layer; 2) cloud layer; 3) data layer; and 4) blockchain layer. There are five types of users in the vaccine traceability system, that is: 1) supervisor; 2) manufacturer (MAU); 3) CDC; 4) VI; and 5) transport enterprise (TE). The traceability system coordinates the involved entities and records the circulation data reliably using blockchain and signature techniques. As in the circulation model in Fig. 1, the involved entities during vaccine circulation need to check whether the received vaccine is correct using the traceability system. They also need to commit their checking results to the traceability system. These checks and commitments serve as digital proof of vaccine circulation; in case of a vaccine safety incident, they are also used to track the involved entities' responsibilities.

Management Layer: The management layer is mainly the interface of services provided to system users. The

manufacturer's product management module is used to manage the production of vaccines, apply for production licenses, and enter trace codes. The traceability service is to negotiate, publicize, and verify the trace code. It also confirms the information of the involved entities in each circulation step. Because each vaccine's trace code is unique, it is necessary to negotiate the vaccine trace code's scope during vaccine transactions and submit it to the supervisor for publicity. When the vaccine is handed over, the vaccine consignee needs to check the vaccine, compare and verify it with the published traceability data, and then generate a signature as a commitment. The entire system incorporates a strict access control mechanism. The supervisor has the authority to manage the system's identity and access control.

Cloud Layer: The traceability system's main services are deployed in the cloud. It has three major components, that is: 1) public-key infrastructure (PKI); 2) trading pool; and 3) identity/authority management services. The vaccine circulation data are structured into transactions and sent to the transaction pool in the cloud. The transaction pool has a threshold, which is the maximum number of transactions accommodated in the transaction pool. When the transactions in the transaction pool exceed the threshold, or a period of time has passed, the transaction pool packages all the current transactions into a Merkle tree. The Merkle tree is stored in a cloud database along with transaction data. The PKI is used to manage public keys and certificates of involved users. The user's private key is stored securely in its local storage. The signature service is used to sign vaccine circulation data and generate aggregate signatures. The signature serves as a commitment of vaccine data checked by an entity.

Data Layer: The data layer in the cloud serves as off-chain data storage of vaccine circulation. The cloud stores all the vaccine circulation data. Besides, in each round of trading pool submission, the cloud automatically packages the transactions in the trading pool into a Merkle tree. Each user signs the root of the Merkle tree to commit the transaction it involves in. The cloud generates an aggregate signature combining all the individual signatures. The cloud then calls the smart contract to upload the Merkle tree root and the aggregate signature to the blockchain. The signature is an endorsement of proof for an involved user.

Blockchain Layer: The blockchain layer is used to record concise yet critical information about vaccine circulation. It stores the Merkle tree root value and the aggregate signature. These data are stored in the smart contract. Each user also runs a blockchain node. Global consensus is achieved by the nodes. In the vaccine traceability scenario, we use a consortium blockchain other than a public chain for better control and performance. As the blockchain administrator, the supervisor is responsible for the authorization and management of blockchain, and management of contracts.

C. Assumption

We assume the supervisor as trusted and the trace code is unique. However, MAUs, CDCs, VIs, and TEs may cheat. Cheating may be due to technical errors or financial gains. Manufacturers, CDCs, VIs, and TEs may dishonestly upload vaccine circulation data or sell a trace code multiple times. They all have interest-dependent relationships. If the upstream of the circulation cheats, the downstream data will be incorrect. Nevertheless, there is also the possibility of collusion. Vaccine circulation data mainly refers to the trace code and the corresponding circulation path. We assume that vaccinated end users (i.e., people) are honest. This assumption is also reasonable, which involves the health and safety of the vaccinated users.

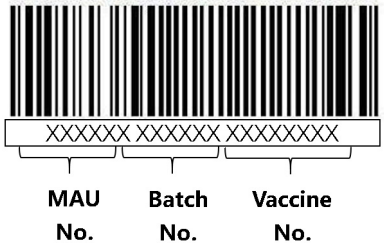


Fig. 3. Trace code design.

IV. Detailed System Design

This section shows the design details of the proposed system. We divide the system into six components, namely: 1) trace code; 2) transactions; 3) trading pool; 4) data verification; 5) signature generation; and 6) smart contract. The details are as follows.

A. Trace Code

The trace code (Fig. 3) is a string of 20 digits, which has three parts. The first six digits are the manufacturer's identify number; the middle six digits are the vaccine batch number; the last eight digits are the unique identify number of the vaccine in one batch. Users can query all the information corresponding to the vaccine according to the trace code, including the manufacturer, batch number, production date, affiliated CDC, and VI.

B. Transactions

The proposed vaccine traceability system contains five types of transactions among the involved entities, that is: 1) production transaction; 2) purchase transaction; 3) distribution transaction (CDC to CDC); 4) distribution transaction (CDC to VI); and 5) transportation transaction. We model one transaction as a key-value table, which is convenient to query using key-value storage. We note that a key-value storage is supported by existing consortium blockchain providers, for example, WeBank FISCO BCOS consortium blockchain [35]. The traceability system implements a strict access control mechanism. Thus, all enterprises and institutions have an internal identity number to represent one entity.

Production Transaction: It works as follows. MAUs submit production plans and production license applications. The supervisor processes the production license application after checking the production plan. After the supervisor approves the application, this batch of vaccines will be assigned a label number according to the production plan. The structure of the transaction is shown in Table I.

TABLE I
PRODUCTION TRANSACTION TABLE

Key	Value					
Producer_no	Vac_batch	Vac_name	Vac_amount	Time	State	Trace_code
010001	463214	HBV	200000	2020-6-1 16:42	1	00000000,00200000
010002	131649	OPV	150000	2020-6-1 17:23	0	Null
010003	164897	DPT	250000	2020-6-1 17:31	0	Null
...	...	...	...	...	...	...

TABLE II
PURCHASE TRANSACTION TABLE

Key	Value						
Producer_no	Vac_batch	Vac_name	Vac_amount	Time	State	CDC_no	Order_no
010001	463214	HBV	10000	2020-7-1 14:05	1	020001	00000572
010001	463214	HBV	20000	2020-7-1 17:02	1	020002	00000573
010003	164897	DPT	10000	2020-7-1 17:31	0	020001	00000574
...	...	...	...	...	...	...	...

TABLE III
DISTRIBUTION TRANSACTION (CDC TO CDC) TABLE

Key	Value					
Producer_no	Batch	Sender_no	Receiver_no	Trace_code	Time	State
010001	463214	020001	026001	...000001,...010000	2020-9-2 13:20	1
010001	463214	020001	026002	...010001,...020000	2020-9-2 13:20	1
...	...	...	...	...	...	...

The meanings of the fields in each row of the table are as follows.

1) *Producer_no:* It is the MAU identity number, which is assigned by the supervisor when the MAU registers the identity using an identity management contract. It is used as the key value in the table for easy management and is also the first part of the trace code.
2) *Vac_batch:* It is the batch number of the vaccine, which is also the second part of the trace code. The same batch of vaccine has the same production conditions. If a problem is found in one batch of vaccines, the entire batch of vaccines will be traced and recalled.
3) *Vac_name:* It is the name of a vaccine.
4) *Time:* It is the time when submitting production application.
5) *State:* It is a status indicator. It is initially set to 0. It is set to 1 if approved by the supervisor; it is set to -1 if rejected.
6) *Trace_code:* After the supervisor approves the application, the trace code will be allocated for the batch of vaccines.

Purchase Transactions: The function of this module is to manage the order negotiation between MAU and CDC. CDC initiates a purchase application to MAU, and submits the vaccine type/name, quantity, time, etc. Then, MAU processes the order. The structure of the purchase transaction is shown in Table II.

The meanings of the fields in each row of the table are as follows.

1) `Producer_no`: It is the MAU number, which is the same as in Table I.
2) `Vac_batch`: It is the batch number of the vaccine, which is also the same as in Table I.
3) `Vac_name`: It is the name of one vaccine.
4) `Time`: It is the time when CDC submits the purchase order.
5) `State`: It is the status value, which is initially set to 0. It is set to 1 if approved by the Supervisor and -1 if rejected, which is similar to Table I.
6) `CDC_no`: It is the identity of the CDC.
7) `Order_no`: It is the order identify number, which is used to distinguish each order.

Distribution Transactions (CDC to CDC): The function of this module is to manage the order negotiation between the upper level CDC and the lower level CDC. The transaction structure is shown in Table III. The `Sender_no` field is the identity number of the upper level CDC. The `Receiver_no` is the identity number of the lower level CDC. The meaning of the other fields is consistent with Tables I and II.

Distribution Transactions (CDC to VI): The function of this module is to manage the order negotiation between CDC and VI. The transaction structure is shown in Table IV, where `VI_no` is the identity number of VI, and the remaining fields are the same as before.

Transport Transactions: This module's function is to record the sender and receiver information of the transportation order and the information of the transportation enterprise. The transportation transaction data are shown in Table V, where the `TE_no` field is the identification number of the transportation enterprise, and the remaining fields are the same as before.

C. Trading Pool

The trading pool lies in the cloud part. It stores all the transactions between MAUs, CDCs, VIs, and TEs. When the number of transactions reaches the transaction pool threshold

TABLE IV
DISTRIBUTION TRANSACTION (CDC TO VI) TABLE

Key			Value			
CDC_no	MAU_no	Batch	VI_no	Time	Trace_code	State
020001	010001	463214	020001	2020-9-2 13:20	...000001,...010000	1
020002	010002	463214	020001	2020-9-2 13:35	...010001,...020000	1
...	...	...	...	...	...	...

TABLE V
TRANSPORT TRANSACTION TABLE

Key			Value		
TE_no	Sender_no	Receiver_no	Time	State	Order_no
040001	010001	020001	2020-9-2 13:20	1	00001746
040001	010002	020001	2020-9-2 13:35	1	00001747
040002	020001	020003	2020-9-2 14:15	0	00001748
...	...	...	...	...	...

or exceeds the time period of one round of transactions, the transaction pool will package the transactions in the current transaction pool.

The packaging is divided into three steps. The first step is to save the current transaction pool data in the form of log files in the cloud database. The transaction pool stops accepting transactions; new transactions will be stored in the blocking queue. The second step is to construct the transactions in the transaction pool into a Merkle tree, which is also stored in the cloud database. The third step is to call the smart contract and save the Merkle tree root as proof of the packed transactions on the chain. After the Merkle root is uploaded to the chain, this round of transactions is handled. The transaction pool is then cleared and the next round of transactions continues to be processed.

D. Data Verification

When the Merkle root as a proof is uploaded to the chain, it does not mean that the vaccine circulation data are reliable. The data has not been confirmed and endorsed by involved entities. The data need to be verified and signed for potential future tracking.

The verification includes two aspects, including: 1) the Merkle tree construction data verification and 2) the verification of the vaccine trace code. The verification of the structured data means that the user verifies whether the local data are consistent with the data stored in the cloud, whether the Merkle tree structure is correct, and whether the proof stored on the chain is correct. The verification of the vaccine trace code means that the recipient verifies whether the actual traceability source code is consistent with the trace code uploaded in the system. Only after all verification is passed, the involved entities will sign the Merkle root.

E. Aggregate Signature Scheme

To enable tracking and nonrepudiation, the proposed system uses a special signature scheme to aggregate signatures from different entities into one signature. This further saves on-chain storage cost. In a certain round of transactions, the involved entities sign the Merkle root. The system aggregates the signatures into one signature. Then, the smart contract is called to upload the aggregated signature to the chain. Thus, although the proof on the chain corresponds to one signature, it corresponds to multiple transactions. After the aggregate signature is verified, the vaccine circulation data in this round of transactions are considered to be endorsed.

- InitParameters $(\mathbb{G}, p, g)$
  - p is a k-bit integer
  - $\mathbb{G}$ is a cyclic group of order p
  - g is a generator of G
  - $H : \{0,1\}^* \rightarrow \{0,1\}^l$
  - group members $P = \{P_i \mid i \in (1, n)\}$
- KeyGen (x_i, X_i)
  - P_i generates a random secret key $x_i \leftarrow \mathbb{Z}_p$
  - P_i computes the public key $X_i : X_i = g^{x_i}$
  - P_i broadcast X_i
  - P_i computes group public key $\tilde{X} = \prod_{i=1}^{n} X_i$
- AggregateSign $(X_i, x_i, root) \rightarrow \sigma$
  - P_i generates a random $r_i \leftarrow \mathbb{Z}_p$
  - P_i computes $R_i = g^{r_i}$
  - P_i broadcast R_i, computes $R = \prod_{i=1}^{n} R_i$
  - P_i computes $c = H(\tilde{X}, R, root)$, $s_i = r_i + cx_i$
  - P_i broadcast s_i, computes $s = \sum_{i=1}^{n} s_i$
  - $\sigma = (R, s)$
- Verify$(root, \sigma) \rightarrow \{0,1\}$
  - $g^s \stackrel{?}{=} R \prod_{i=1}^{n} X_i^c \stackrel{?}{=} R\tilde{X}^c$

Fig. 4. Aggregate signature scheme.

Specifically, we use the Schnorr signature scheme [36]. The signing process is divided into four parts, that is: 1) initialization; 2) key generation; 3) signature; and 4) verification, which is shown in Fig. 4.

Initialization runs when the system is first deployed. It sets up the signature key, a cyclic group, the cyclic group generator, and a hash function. The key generation stage is divided into the user's personal key and the group public key. Each user generates the user's private key x_i, calculates the corresponding public key X_i, and broadcasts the public key. The group key is the combination of all the involved users' personal keys. The group key depends on the involved entities in each round of transactions. Because the entities involved in each round of transactions are different, the corresponding group keys are also different.

In the signing phase, each user first generates a random number. At the same time, each user also broadcasts a masked version of the random number $R^i = g^{r_i}$. They are used to

Algorithm 1 Submit Merkle Tree Root

Require: the packing time *date*; the Merkle tree root *root*;
Ensure: execution status `ret_code` $\in \{0, 1\}$;
1: get the smart contract table object T;
2: create entry object e;
3: assign values to e, $e.Date \leftarrow date$, $e.Proof \leftarrow root$;
4: set the remaining attributes as empty;
5: `ret_code` $\leftarrow T$.insert($date$, e);
6: **return** `ret_code`;

Algorithm 2 Submit Signatures and Parameters

Require: the table key *date*; aggregate signature (R, s); large prime number p, q; the generator g of the cyclic group $\mathbb{G}$; the group public key $\tilde{X}$;
Ensure: execution status `ret_code` $\in \{0, 1\}$;
1: get table object T;
2: create entry object e;
3: assign signature values to e;
4: set the *condition* as $entry.Key == date$;
5: select entry from T where $entry.Key$ equals $date$;
6: `ret_code` $\leftarrow T.update(date, e, condition)$;
7: **return** `ret_code`;

generate a global unified parameter R in the group as part of the signature. Then, each user signs the root of the Merkle tree of the current transaction group with the private keys. The users also send the signatures to the cloud. The cloud aggregates the signatures and uploads the result to the chain. In the verification phase, any node can verify the signature on the chain using the group public key.

F. Smart Contract

The proposed system employs two smart contracts to support secure vaccine tracing. The first is the *Agency Contract*, which is used to upload the Merkle tree root and aggregate signature. All users can verify the correctness of the aggregate signature. When the signature verification fails, it means that the traceability data does not match or is incorrect. The second is the *Arbitration Contract*, which is used to initiate arbitration when there is a mismatch in vaccine traceability data. The supervisor will intervene to investigate the problematic product's circulation path and publish the investigation results on the chain.

Agency Contract: This contract maintains a key-value table and a submission function to accept invocations from the cloud. The data table contains the Merkle tree root, signatures, and related parameters for each packaging round. The parameters are as follows.

1) `Date:` It is the packing time. Because the packaging time is unique, this field is used as the key.
2) `Proof:` It is the Merkle tree root.
3) `UsersSign.R & UsersSign.S:` It is the aggregate signature as in Fig. 4.
4) `Para.g & Para.p & Para.q:` They are the signature initialization parameters.
5) `GroupPK:` It is the group public key $\tilde{X}$ of the current signature group.

Algorithm 1 shows the proof submission function in the agency contract. When the contract is deployed, it initializes its data table. When calling the submission function, the table object must be obtained first. Then, create an entry object and assign values. The date are the key value of the table and must be nonempty. The remaining fields are initialized to be empty. Finally, the entry object is inserted into the table. The execution result is then returned.

Algorithm 2 presents the pseudocode of the contract's signature submission function. First, obtain the table object, create the entry object, and assign values. Then, set the condition for later table updates. Because the date are unique, the condition is set to whether the key equal to the date. Finally, use the created object to update the table data.

Arbitration Contract: The Arbitration contract also maintains a key-value table data. It has the following parameters.

1) `Date:` It is the packing time. Because the packaging time is unique, the date is used as the key.
2) `ID_num:` It is the identity number to register system participants.
3) `Batch:` It is the batch identity of a problematic vaccine.
4) `Inv_result:` It is the result of a problematic vaccine handling and tracking. The supervisor intervenes the problematic vaccine, initiates an investigation, and releases the result of the investigation.

The table in this contract is used to save arbitration related data when a vaccine data mismatch problem occurs. The key is still the packaging time; the value includes the arbitration initiator's identity code, vaccine batch, and arbitration result. To save storage cost, the arbitration result could be chosen according to practical needs, for example, it could be a supervisor's Web address. The proposed system thus demonstrates a proof-of-concept idea regarding the arbitration.

V. Evaluation

A. Setup

We built a proof-of-concept prototype of the proposed system using a virtual machine on a personal computer with 8 GB of memory and a 512-GB hard drive. We simulated the cloud using function calls. We employed a consortium blockchain as the underlying infrastructure, that is, the FISCO BCOS blockchain platform [35]. We note that it is also possible to use other consortium blockchain platforms. We chose the FISCO BCOS blockchain for prototype mainly because it has a large influence around China. Correspondingly, it is also easier to improve and transfer the proof-of-concept prototype under FISCO BCOS to a workable, influential system in the real world. Our system setup and smart contract code could be found at [37].

We have configured six MAU nodes, ten CDC nodes, six TE nodes, and 40 VI nodes, with only one supervisor. All nodes participate in the consensus process of the blockchain. The blockchain configuration is shown in Table VI. The *Listen_IP* is the node's peer-to-peer listening IP address. We used the default value of 127.0.0.1. The listening port is the channel port, which uses the port configured by *Web3SDK* and has the default port number 20200. The port range we assigned

TABLE VI
BLOCKCHAIN CONFIGURATION PARAMETERS

Group Number	Number of Nodes				Number of nodes	Channel Listen Port	Node Port	Gas Limit (Wei)	Block Generate Time (s)
	MAU	CDC	TE	VI					
1	6	10	6	40	127.0.0.1	20200	30300-30330	3×10^7	1

to each node is 30 300 to 30 330. The *Gas_Limit* for each block is set to the default 3×10^7 Wei. We set up multiple sets of experiments for the threshold of the transaction pool. The thresholds are set to 1000, 2000, 5000, and 10 000. The maximal time period to pack the transaction pool is the same as the block generation time of the blockchain, which is 1 s.

B. Performance Metrics

We mainly consider four performance indicators: 1) computation; 2) storage; 3) communication; and 4) contract gas consumption. The computation includes the time to pack the transactions into a Merkle tree, the time to generate the key, the time to sign, and the time to verify the signature. Storage includes the size of off-chain transaction data and the size of on-chain smart contract transactions. Communication includes the size of the communication data between a user and the cloud, and the smart contract's input data size. Gas consumption includes the gas for contract deployment and the gas for calling the contract.

The principle of saving space is that the original data are stored in the cloud. The blockchain only stores the root of the Merkle tree constructed from multiple transactions and the aggregate signature. The original multiple transactions are combined into one piece of data on the chain (i.e., Merkle tree root and aggregate signature). In essence, this is the use of cloud computing storage capabilities to exchange space on the chain. How much data can be processed in the cloud, in theory, determines how many times the space on the chain can be saved. The upper limit of cloud computing power determines how much space can be saved. It is worth noting that only transaction packaging is synchronized with blockchain while signatures can be performed asynchronously. Therefore, under the premise of not exceeding the consortium blockchain's TPS, the only operation that affects system performance is transaction pool packaging.

C. Results

Storage: We measured the size of data storage corresponding to different numbers of transactions. For different scenarios, the constructed transactions are different; the transaction data size is also different. Thus, we average different transactions. Table VII shows the storage cost. Each transaction is only about 120 bytes. A pool of 20 000 transactions consumes 2325 kilobytes of cloud storage. The proof and signature stored on the chain are only 156 bytes. The results show that on-chain storage is significantly smaller than storing all data on the blockchain.

Communication: The communication cost includes the communication when submitting transactions, generating group public keys, communicating with other entities when signing and invoking smart contracts. The measured communication overhead results are shown in Table VIII. It costs around 100 bytes when calling smart contract functions. In practice, the blockchain is deployed over the public network infrastructure. Nowadays, the network bandwidth normally achieves more than 20 Mb/s (i.e., 2.5×10^6 bytes per second) for end users. Thus, the communication cost is practically small considering existing network infrastructure.

TABLE VII
STORAGE CONSUMPTION

Threshold	Cloud (KB)	Blockchain (Byte)
1000	112	156
2000	220	156
5000	548	156
10000	1060	156
20000	2325	156

TABLE VIII
COMMUNICATION CONSUMPTION

Cloud (byte)			Blockchain (byte)			
Submit Transaction	Key Generation	Group Sign	Submit Proof	Submit Sign	Apply for Arbitration	Publish Results
111	32	103	91	125	56	112

TABLE IX
COMPUTATION CONSUMPTION

Threshold	Packing Time (ms)	Key Generation Time (ms)	Sign Time (ms)
1000	129	380	134
2000	189	701	161
5000	254	1100	346
10000	491	2974	988
20000	852	5103	1731

Gas Consumption: We also measured the gas consumption for the smart contracts. The gas consumption for contract deployment is about 1041654 Wei; uploading proof costs about 50 900 Wei; uploading aggregate signature consumes 51 913 Wei. The gas consumption for arbitration is approximately 45 129 Wei. The consumption is small. We note that there is no need for economic expenses in exchange for gas in a consortium blockchain. However, the block capacity has a gas limit. Thus, we use gas consumption as a parameter to measure system performance.

Computation: We measured the time cost to pack transactions under different thresholds, generate the key, generate and aggregate the signature, and verify the signature. The key length is set to 256 bits. Table IX shows the result. When packing 20 000 transactions, it costs less than 1 s. The corresponding key generation time and signature generation time are around 5.1 and 1.7 s. Because users only need to verify one signature, verification time is constant, which is about 2 ms and not included in the table.

From the table, we could find that the computation time increases linearly with the increase of transactions. The packing and signing of transactions are decoupled. According to the measurement, within 1 s of block generation time, the highest number of possible packed transactions is about 25 000. It should be noted that the transaction packing time has a relationship with the computing power of the machine and the computation method of the algorithm. The 25 000 transactions per second are not the upper limit of the proposed system. It is also possible to improve system performance by optimizing machine performance, optimizing the signature operations, etc.

VI. Discussion

A. Security

The discussion on security is divided into two parts. One is on the possible security risks in the threat model. The other is whether putting the original data of vaccine circulation off-chain will increase the risk of data tampering. We model the supervisor as trusted in the threat model while other agencies and companies may cheat and collude. We argue for security by discussing the possibility of cheating by each entity.

If the MAU uploads incorrect vaccine distribution data, the downstream CDC will compare the actual source data with the manufacturer's data. If the CDC finds a mismatch, it can apply to the supervisor for arbitration. There is also a possibility of collusion between the manufacturer and the CDC. Then, the CDC's vaccine data that is sent to the lower level CDC and VI will not match. Initiating arbitration at any step can effectively track the circulation process.

The number of CDC, VI, and TE involved in a batch of vaccines is huge. This implies that all relevant institutions and companies could collude only with a very low possibility. Even if they collude, it will cause the final vaccinated users' vaccine data to be different from the actual data. End users then could report the incident. Simultaneously, if one vaccine code has multiple uses, there must be a vaccine fraud. Because the traceable source code is globally unique, if multiple vaccinated users have inquired about the same vaccine, the system will detect the occurrence of vaccine fraud.

The original transaction data are stored in the cloud. They are public within the involved MAUs, CDCs, VIs, and TEs that have obtained the chain access qualification from the supervisor. Vaccine circulation data are constructed as a Merkle tree. Suppose some wrong data are uploaded before construction. The downstream entities will detect this, which has been discussed above. If the data are tampered with after constructing the Merkle tree, the Merkle tree root will also change. In this case, other entities will fail to verify the transactions. If the verification fails, the entities will not sign the Merkle root, resulting in the aggregate signature verification failure. Therefore, the proposed system enhances the security of the vaccine circulation process.

B. Limitation

Compared with existing vaccine safety protection systems, the proposed blockchain-based system builds a much better trust. No single entity is able to cheat throughout the vaccine circulation process. However, the proposed system imposes larger computation and storage burden. This is because the involving parties need to achieve a consensus and store the transactions. However, considering the better trust, we believe the performance tradeoff is worthy.

Currently, the proposed vaccine traceability system is only a proof of concept. It could be improved further. The vaccine circulation data is structured as a transaction, which is later submitted to the transaction pool and packed into a Merkle tree. The Merkle tree root is uploaded on the chain. But the efficiency of the entire system is limited by the processing speed of the packing step. To improve efficiency, one could use the sharding idea to divide the transactions into multiple shards. Each shard could be packed and verified independently. In this way, the block generation and transaction packing steps are decoupled; system efficiency thus could be improved by trading off more system complexity.

VII. Conclusion

Ensuring vaccine safety is critical. To improve the safety of vaccine circulation, we proposed an improved blockchain-based system to make the vaccine circulation process traceable and verifiable. The improved system alleviates the storage pressure of the blockchain. We also implemented and verified the proposed scheme. The experimental results showed promising performance and potentiality. We also discussed the limitation of the proposed scheme and possible further improvement.

We hope that the proposed scheme provides new insights for ensuring vaccine safety. In the future, one important work is to fine-tune the proposed system to achieve better performance by incorporating different entities' interests. Another important work is to conduct incentive mechanism research that could encourage the stakeholders along the vaccine supply chain to migrate existing systems to a blockchain-based one.

References

[1] S. Jarrett *et al.*, "The role of manufacturers in the implementation of global traceability standards in the supply chain to combat vaccine counterfeiting and enhance safety monitoring," *Vaccine*, vol. 38, no. 52, pp. 8318–8325, 2020.

[2] B. M. G. Rosa, S. Anastasova, and G. Z. Yang, "NFC-powered implantable device for on-body parameters monitoring with secure data exchange link to a medical blockchain type of network," *IEEE Trans. Cybern.*, early access, Jul. 1, 2021, doi: 10.1109/TCYB.2021.3088711.

[3] W. Zheng, K. Wang, and F.-Y. Wang, "Gan-based key secret-sharing scheme in blockchain," *IEEE Trans. Cybern.*, vol. 51, no. 1, pp. 393–404, Jan. 2021.

[4] C. Tang, C. Li, X. Yu, Z. Zheng, and Z. Chen, "Cooperative mining in blockchain networks with zero-determinant strategies," *IEEE Trans. Cybern.*, vol. 50, no. 10, pp. 4544–4549, Oct. 2020.

[5] Y. Li, H. Li, X. Ding, and G. Zhao, "Leader–follower consensus of multiagent systems with time delays over finite fields," *IEEE Trans. Cybern.*, vol. 49, no. 8, pp. 3203–3208, Aug. 2019.

[6] B. Tian, H. Lu, Z. Zuo, and W. Yang, "Fixed-time leader–follower output feedback consensus for second-order multiagent systems," *IEEE Trans. Cybern.*, vol. 49, no. 4, pp. 1545–1550, Apr. 2019.

[7] B. Yong, J. Shen, X. Liu, F. Li, H. Chen, and Q. Zhou, "An intelligent blockchain-based system for safe vaccine supply and supervision," *Int. J. Inf. Manage.*, vol. 52, Jun. 2020, Art. no. 102024.

[8] G. Falco, C. Li, P. Fedorov, C. Caldera, R. Arora, and K. Jackson, "NeuroMesh: IoT security enabled by a blockchain powered botnet vaccine," in *Proc. Int. Conf. Omni-Layer Intell. Syst.*, 2019, pp. 1–6.

[9] S. Peng *et al.*, "An efficient double-layer blockchain method for vaccine production supervision," *IEEE Trans. NanoBiosci.*, vol. 19, no. 3, pp. 579–587, Jul. 2020.

[10] L. Cui *et al.*, "Improving vaccine safety using blockchain," *ACM Trans. Internet Technol.*, vol. 21, no. 2, pp. 1–24, 2021.

[11] Y. Zhang, X. Xu, A. Liu, Q. Lu, L. Xu, and F. Tao, "Blockchain-based trust mechanism for iot-based smart manufacturing system," *IEEE Trans. Comput. Social Syst.*, vol. 6, no. 6, pp. 1386–1394, Dec. 2019.

[12] Y. Cao, F. Jia, and G. Manogaran, "Efficient traceability systems of steel products using blockchain-based Industrial Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 16, no. 9, pp. 6004–6012, Sep. 2020.

[13] M. Westerkamp, F. Victor, and A. Küpper, "Blockchain-based supply chain traceability: Token recipes model manufacturing processes," in *Proc. IEEE Int. Conf. Internet Things (iThings)*, 2018, pp. 1595–1602.

[14] "Waltonchain." [Online]. Available: https://www.waltonchain.org/#/en (Accessed: Apr. 6, 2022).

[15] J. Lu, J. Shen, P. Vijayakumar, and B. B. Gupta, "Blockchain-based secure data storage protocol for sensors in the Industrial Internet of Things," *IEEE Trans. Ind. Informat.*, early access, Sep. 14, 2021, doi: 10.1109/TII.2021.3112601.

[16] C. Lin, D. He, N. Kumar, X. Huang, P. Vijayakumar, and K.-K. R. Choo, "Homechain: A blockchain-based secure mutual authentication system for smart homes," *IEEE Internet Things J.*, vol. 7, no. 2, pp. 818–829, Feb. 2020.

[17] P. Bagga, A. K. Sutrala, A. K. Das, and P. Vijayakumar, "Blockchain-based batch authentication protocol for Internet of Vehicles," *J. Syst. Archit.*, vol. 113, Feb. 2021, Art. no. 101877.

[18] M. R. Ahmed, K. Meenakshi, M. S. Obaidat, R. Amin, and P. Vijayakumar, "Blockchain based architecture and solution for secure digital payment system," in *Proc. IEEE Int. Conf. Commun.*, 2021, pp. 1–6.

[19] I. Eyal, A. E. Gencer, E. G. Sirer, and R. Van Renesse, "Bitcoin-NG: A scalable blockchain protocol," in *Proc. 13th USENIX Symp. Netw. Syst. Design Implement. (NSDI)*, 2016, pp. 45–59.

[20] R. Pass and E. Shi, "Hybrid consensus: Efficient consensus in the permissionless model," in *Proc. 31st Int. Symp. Distrib. Comput.*, 2017, pp. 1–16.

[21] Y. Sompolinsky and A. Zohar, "Secure high-rate transaction processing in bitcoin," in *Proc. Int. Conf. Financ. Cryptogr. Data Security*, 2015, pp. 507–527.

[22] Y. Sompolinsky and A. Zohar, "Phantom: A scalable blockdag protocol," IACR, Lyon, France, Rep. 2018/104, 2018.

[23] C. Li, P. Li, D. Zhou, W. Xu, F. Long, and A. Yao, "Scaling nakamoto consensus to thousands of transactions per second," 2018, *arXiv:1805.03870*.

[24] L. Luu, V. Narayanan, C. Zheng, K. Baweja, S. Gilbert, and P. Saxena, "A secure sharding protocol for open blockchains," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, 2016, pp. 17–30.

[25] E. Kokoris-Kogias, P. Jovanovic, L. Gasser, N. Gailly, E. Syta, and B. Ford, "OmniLedger: A secure, scale-out, decentralized ledger via sharding," in *Proc. IEEE Symp. Security Privacy*, 2018, pp. 583–598.

[26] A. Miller, Y. Xia, K. Croman, E. Shi, and D. Song, "The honey badger of BFT protocols," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, 2016, pp. 31–42.

[27] M. Yin, D. Malkhi, M. K. Reiter, G. G. Gueta, and I. Abraham, "HotStuff: BFT consensus with linearity and responsiveness," in *Proc. ACM Symp. Principles Distrib. Comput.*, 2019, pp. 347–356.

[28] "Proof of Stake Consensus." 2021. [Online]. Available: https://en.bitcoin.it/wiki/Proof_of_Stake

[29] S. Bouraga, "A taxonomy of blockchain consensus protocols: A survey and classification framework," *Expert Syst. Appl.*, vol. 168, Apr. 2021, Art. no. 114384.

[30] J. Liu, T. Yin, D. Yue, H. R. Karimi, and J. Cao, "Event-based secure leader-following consensus control for multiagent systems with multiple cyber attacks," *IEEE Trans. Cybern.*, vol. 51, no. 1, pp. 162–173, Jan. 2021.

[31] J. Poon and T. Dryja. "The Bitcoin Lightning Network: Scalable Off-Chain Instant Payments." 2016. [Online]. Available: http://lightning.network/lightning-network-paper.pdf

[32] R. Khalil and A. Gervais, "Revive: Rebalancing off-blockchain payment networks," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, 2017, pp. 439–453.

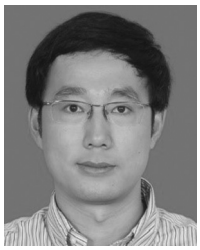
[33] C. Decker and R. Wattenhofer, "A fast and scalable payment network with bitcoin duplex micropayment channels," in *Proc. Symp. Self-Stabilizing Syst.*, 2015, pp. 3–18.

[34] A. Back *et al.* "Enabling Blockchain Innovations with Pegged Sidechains." 2014. [Online]. Available: http://www.opensciencereview.com/papers/123/enablingblockchain-innovations-with-pegged-sidechains

[35] "FISCO BCOS: The Building Block of Open Consortium Chain." 2021. [Online]. Available: http://fisco-bcos.org/

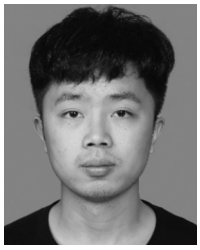
[36] C.-P. Schnorr, "Efficient signature generation by smart cards," *J. Cryptol.*, vol. 4, no. 3, pp. 161–174, 1991.

[37] Z. Xiao. "Source Code for Performance Evaluation." 2021. [Online]. Available: https://github.com/xiaozhe-szu/BBVSPS



Laizhong Cui (Senior Member, IEEE) received the B.S. degree from Jilin University, Changchun, China, in 2007, and the Ph.D. degree in computer science and technology from Tsinghua University, Beijing, China, in 2012.

He is currently a Professor with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China. He led a project of the National Natural Science Foundation, and several projects of Guangdong Province and Shenzhen City. His research interests include future Internet architecture, edge computing, IoT, computational intelligence, software-defined network, and machine learning.



Zhe Xiao is currently pursuing the master's degree with the College of Computer Science and Engineering, Shenzhen University, Shenzhen, China.

His research interests include data privacy and protection.

Fei Chen (Member, IEEE) received the Ph.D. degree in computer science and engineering from The Chinese University of Hong Kong, Hong Kong, in 2014.

He is currently an Associate Professor with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China. His research interests include data protection and privacy, and distributed systems and applications.

Hua Dai (Member, IEEE) received the Ph.D. degree in computer science and technology from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2011.

He is currently an Associate Professor with the Nanjing University of Posts and Telecommunications, Nanjing. His research interests include data management and security and database security.

Dr. Dai is a member of CCF.

Jianqiang Li received the Ph.D. degree in automation engineering from the South China University of Technology, Guangzhou, China, in 2008.

He is a Professor with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China. He led three projects of the National Natural Science Foundation and three projects of the Natural Science Foundation of Guangdong Province, China. His current research interests include data analysis, embedded systems, and the Internet of Things.