Hindawi
Security and Communication Networks
Volume 2022, Article ID 1926902, 15 pages
https://doi.org/10.1155/2022/1926902

Research Article

Analysis and Improvement of Blockchain-Based Multilevel Privacy-Preserving Location Sharing Scheme for Telecare Medical Information Systems

Zhenjie Huang[1,2] **Yafeng Guo**[3] **Hui Huang**[4] **Runlong Duan**[1,4] **and Xiaolong Zhao**[1,2]

[1]*Fujian Key Laboratory of Granular Computing and Application, Minnan Normal University, Zhangzhou 363000, China*
[2]*School of Mathematics and Statistics, Minnan Normal University, Zhangzhou 363000, China*
[3]*Department of Electronic and Informatics, Zhangzhou City College, Zhangzhou 363000, Fujian, China*
[4]*School of Computer Science, Minnan Normal University, Zhangzhou 363000, China*

Correspondence should be addressed to Zhenjie Huang; zjhuang@mnnu.edu.cn

Received 17 October 2021; Revised 17 November 2021; Accepted 29 November 2021; Published 18 January 2022

Guest Editor: Thippa Reddy G

Patient location sharing is an important part of modern smart healthcare and mobile medical services. Blockchain has many attractive properties and is suitable for managing patient locations in telecare medical information systems (TMIS). Recently, Ji et al. proposed a blockchain-based multilevel privacy-preserving location sharing (BMPLS) scheme for TMIS. In this paper, we show that Ji et al.'s BMPLS scheme does not achieve confidentiality and multilevel privacy-preserving. An adversary outside the system can use an ordinary personal computer to completely break the system within a dozen hours and obtain the location of any patient at any time. The adversary inside the system can use an ordinary personal computer to obtain the location of the designated patient within tens of seconds. Using salting technology, we propose an improved BMPLS scheme to fix our attacks. We also optimized the BMLS scheme to make it correct and executable. The security analysis shows that the improved BMPLS scheme achieves decentralization, untamperability, confidentiality, multilevel privacy-preserving, retrievability, and verifiability. The simulation shows that the improved BMPLS scheme is practical, the computational overhead of the location record phase is within 10 ms, and the computational overheads of the location sharing and location extraction phases are both within 30 ms.

1. Introduction

Blockchain is a new decentralized infrastructure and distributed computing paradigm, and it is one of the most revolutionary emerging technologies [1]. In a narrow sense, blockchain is a decentralized shared general ledger that combines data blocks in a chain into a specific data structure in chronological order and uses cryptography technology to ensure that data are untamperable and unforgeable. In a broad sense, blockchain technology is a new decentralized computing paradigm. It uses an encrypted chain block structure to store and verify data, consensus algorithms to update data, and smart contracts to manipulate data. Blockchain has the advantages of decentralization, trustless, anonymity, and untamperability. It can break through the limitations of traditional centralized systems and find important applications in a wide range of fields. More meaningful applications will appear with the further integration of blockchain with cloud computing, edge computing, and the Internet of Things [2–6].

With the rapid development of information technology, medical management has become more intelligent and real-time. Wireless mobile networks and wearable technology enable mobile medical services and telemedicine to be realized. For example, IBM has integrated a real-time asset locator (RTAL) for mobile medical and telecare medical to track the location of patients, equipment, and medical staff. Location management plays a significant role in remote patient service and monitoring, such as monitoring particular patients, handling emergencies, and analyzing

epidemic distribution. Blockchain has many attractive properties and is suitable for managing patient locations in telecare medical information systems (TMIS).

Based on blockchain technology, Zyskind et al. [7] proposed a data storage scheme to protect sensitive data such as user's location. In their scheme, user data needs to be encrypted before being stored on the blockchain to achieve confidentiality. Amoretti et al. [8] proposed a blockchain-based location proof scheme. Different from [7], in this scheme, the location information is signed before being put on the blockchain to achieve verifiability. Ji et al. [9] comprehensively analyzed the security requirements of blockchain-based location sharing for TMIS and proposed a blockchain-based multilevel privacy-preserving location sharing scheme (BMPLS). They claim that their scheme achieves decentralization, untamperability, confidentiality, multilevel privacy-preserving, retrievability, and verifiability. Unfortunately, we found that their scheme is insecure in practice. The adversary can recover any user's location effectively. Recently, Lee et al. [10] proposed a blockchain-based medical data preservation scheme for TMIS. Their scheme consists of a medical sensor area authentication protocol and a social network information transfer protocol.

1.1. Related Works. With the development of Internet technology, telemedicine has gradually replaced the traditional treatment model. Electronic medical records (EMR) and electronic health records (EHR) are generated in large quantities and frequently exchanged and shared among legitimate users. The protection of electronic medical information is related to patients' privacy and related to their life safety. Therefore, the security and privacy protection of electronic medical records are significant for the development of telemedicine.

Blockchain technology has the potential to improve the medical ecosystem. It provides a novel, efficient, and secure model for exchanging EMRs and EHRs and enhances medical data security, privacy, and interoperability [11, 12]. Using blockchain technology, Cao et al. [13] proposed a secure cloud-assisted electronic health system to protect outsourced electronic health records from illegal modification. Wang et al. [14] proposed a blockchain-based eHealthcare system interoperating with wireless body area networks. Using proxy reencryption, Huang et al. [15] proposed a blockchain-based decentralized medical data sharing scheme with privacy-preserving. Zhuang et al. [16] proposed a patient-centric health information exchange framework. Shamshad et al. [17] proposed a novel blockchain-based privacy and security preserving EHR sharing protocol. Huang et al. [18] proposed a blockchain-based eHealth system, in which the manipulation of EHRs can be audited. Zhu et al. [19] proposed an improved convolution Merkle tree-based blockchain electronic medical record secure storage scheme. Uddin et al. [20] proposed a blockchain leveraged decentralized eHealth architecture. Tanwar et al. [21] explored several solutions that use blockchain technology to improve the current limitations of medical systems, including frameworks and tools for measuring the performance of such systems. Using off-chain and on-chain blockchain system design, Miyachi et al. [22] proposed a modular hybrid privacy-preserving framework. Chen et al. [23] proposed a complete medical information system model based on blockchain technology to realize the goal of safe storage and sharing of medical data. Hossein et al. [24] proposed a novel blockchain-based privacy-preserving architecture for IoT healthcare applications. A summary of related works is shown in Table 1.

1.2. Motivation and Contributions. Among the previous blockchain-based location sharing schemes, Zyskind et al.'s scheme only provides decentralization, untamperability, and confidentiality [7], while Amoretti et al.'s scheme only provides decentralization, untamperability, and verifiability [8]. These are far from enough. Ji et al. [9] considered decentralization, untamperability, confidentiality, multilevel privacy protection, retrievability, and verifiability, but their scheme is insecure in practice. The adversary can recover any user's location effectively. Thus, it is of great significance to propose secure and practical blockchain-based multilevel privacy-preserving location sharing schemes. The comparison of previous blockchain-based location sharing schemes is shown in Table 2, where "$\surd$" means satisfied, "×" means dissatisfied, and "–" means uninvolved.

The main contributions of this paper are as follows:

(1) We analyze the security of Ji et al.'s BMPLS scheme [9] and show that it has fatal flaws in confidentiality and multilevel privacy-preserving. An adversary outside the system can use an ordinary personal computer to completely break the system within a dozen hours and obtain the location information of any patient at any time. The adversary inside the system can use an ordinary personal computer to obtain the location information of the designated patient within tens of seconds. In addition, in some cases, their scheme cannot be executed.

(2) Using salting technology, we propose an improved BMPLS scheme to fix our attacks. We add **Setup** and **Key generation** phases to the scheme to provide the foundation for other phases and replace the **Location verification** phase with the **Location extraction** phase. We also optimized the BMLS scheme to make it correct and executable. The security analysis shows that the improved BMPLS scheme achieves decentralization, untamperability, confidentiality, multilevel privacy-preserving, retrievability, and verifiability. The simulation shows that the improved BMPLS scheme is practical, the computational overhead of the location record phase is within 10 ms, and the computational overheads of the location sharing and location extraction phases are both within 30 ms.

1.3. Organization. The rest of this paper is organized as follows. Section 2 presents preliminaries. Section 3 presents the architecture of BMPLS and reviews Ji et al.'s BMPLS scheme. Section 4 analyzes the scheme of Ji et al. Section 5

Table 1: Summary of related works.

Ref.	Contribution	Technologies used	Key features
[11]	A blockchain-based data preservation system for medical data	Blockchain and Ethereum	Ensuring the primitiveness and verifiability of stored data while preserving privacy for users
[12]	An APP for health data sharing based on blockchain	Blockchain and secure multiparty computing	Patients own and control their healthcare data and use the indicator centric schema to organize personal healthcare data
[13]	A cloud-assisted secure eHealth systems	Blockchain and cloud storage	Every operation of the outsourced EHRs is recorded on the blockchain
[14]	An eHealthcare system interoperating with WBANs	Blockchain and wireless body area network	Providing a secure and low-power healthcare solution; utilizing the WBAN and blockchain technology
[15]	A blockchain-based privacy-preserving scheme	Blockchain, smart contract, zero-knowledge proof, and proxy reencryption	Achieving the data availability and consistency between patients and research institutions; using zero-knowledge proof to protect patient's privacy
[16]	A patient-centric health information exchange framework	Blockchain, data segmentation, and smart contract	Utilizing the smart contract feature to protect data security and patients privacy, ensure data provenance, and provide patients full control of their health records
[17]	A secure blockchain-based eHealth records storage and sharing scheme	Private and consortium blockchain and proxy reencryption	All EHRs are public-key encrypted and searchable, using private blockchain to store EHRs and consortium blockchain to store secure indexes
[18]	A blockchain-based eHealth system	Blockchain, cloud computing, and attributes-based proxy reencryption	Each legitimate manipulation will be written into the blockchain, and any threatening behavior will be discovered
[19]	Improved convolution Merkle tree	Blockchain and convolution operation	Using the convolutional layer structure to replace the original binary tree structure
[20]	An IoT eHealth framework based on blockchain	Blockchain, fog computing, edge computing, fuzzy inference, and task offloading	A patient agent (PA) software processes medical data, executes consensus mechanism, and utilizes a task-offloading algorithm to ensure patient's privacy
[21]	Blockchain-based electronic healthcare record system	Hyperledger fabric and Wireshark capture engine	Adopting chain code to ensure proper operation of blockchain ledger
[22]	Blockchain framework using on-chain and off-chain design	Blockchain and on-chain and off-chain design	Unpluggable components in the face of different data types to cope with different policy requirements
[23]	A blockchain-based preserving and sharing system	Proxy reencryption and hyperledger fabric	Real-time patient data collection and managing data with chain code
[24]	A blockchain-based architecture for IoT healthcare applications	Blockchain	Using a dual-chain architecture to develop access control policies that isolate data and policies

Table 2: Comparison of previous schemes.

Schemes	Zyskind et al.'s [7]	Amoretti et al.'s [8]	Ji et al.'s [9]	Our scheme
Decentralization	√	√	√	√
Untamperability	√	√	√	√
Confidentiality	√	×	×	√
Multilevel protection	×	×	×	√
Retrievability	—	—	√	√
Verifiability	—	√	√	√

proposes an improvement to fix our attacks with security and performance analysis. Section 6 concludes this paper.

2. Preliminaries

2.1. Order-Preserving Encryption. Order-preserving encryption (OPE) is deterministic encryption that can preserve numerical ordering on their plaintext space [25].

An order-preserving encryption scheme Π_{OPE} consists of the following algorithms OP-KeyGen and OP-Enc:

(1) OP-KeyGen is the key generation algorithm, takes as input the security parameter λ, and outputs a secret key opk.

(2) OP-Enc is the encryption algorithm, takes as input a secret key opk and a plaintext $x \in \{0,1\}^n$ interpreted as a numerical value $0 \le x \le 2^n - 1$, and outputs ciphertext $c \in \{0,1\}^m$ interpreted as a numerical value $0 \le c \le 2^m - 1$.

They satisfy that OP-$Enc(opk, i) <$ OP-$Enc(opk, j)$, for all $opk \leftarrow$OP-KeyGen$(1^\lambda), 0 \le i < j \le 2^n - 1$.

For $N, M \subseteq \mathbb{N}$ with $|N| \le |M|$, a function $f: N \longrightarrow M$ is order-preserving if for all $i, j \in N$, $f(i) > f(j)$ iff $i > j$. If an order-preserving encryption scheme Π_{OPE} is secure, then OP-Enc( ) is a pseudorandom order-preserving function [25].

2.2. Merkle Tree. Merkle tree [26] provides efficient data authentication. A Merkle tree is based on a binary tree and a one-way hash function. The value of its leaf node is the data, and the value of its nonleaf node is the hash of the values of its two child nodes. If the Merkle tree has n leaf nodes, it only needs at most $\lceil \log_{2n} \rceil$ data to authenticate a leaf node, not all n data.

In our improved scheme, we need to calculate the Merkle tree of $\text{node}_0^x, \text{node}_1^x, \text{node}_2^x, \ldots, \text{node}_{2^N}^x$, and Figure 1 is an illustrative example. In order to authenticate h_3, we only need to provide h_4 and h_6, then calculate $h_7' = \text{Hash}(h_3 \| h_4)$ and $h_8' = \text{Hash}(h_6 \| h_7')$ in sequence, and finally verify whether $h_8' = h_8$.

3. Review of Ji Et Al.'s BMPLS Scheme

This section reviews the architecture of BMLS and Ji et al.'s scheme.

3.1. Architecture of BMPLS. BMLS can be used as a module of TMIS to manage and share patient location information. The architecture of BMPLS is shown in Figure 2. There are two types of entities in the system: location data owner (LDO) and location data requestor (LDR). LDOs, such as infectious disease or chronic disease patients, record their location information in the blockchain. LDRs request LDOs' location information in different levels according to their trust levels and actual needs. For example, mobile clinics need to know the precise location of patients to provide them with on-site services; the medical center also requires an exact location to deliver the medicine. In contrast, infectious disease investigators only need to know the range of the patient, not the precise location.

3.2. Ji Et Al.'s BMPLS Scheme. Ji et al.'s BMPLS scheme consists of the following three phases [9].

3.2.1. Initialization. The location data owner (LDO) represents his visit region as a coordinate region $S = \{(x, y) | 0 \le x \le X, 0 \le y \le Y\}$, runs Algorithm 1 to generate the registration record regRec, and puts it into the blockchain.

In Algorithm 1, the partition function $\text{Parti}(S, N) = \{x_i = i \times X/2^N, \quad 1 \le i \le 2^N\} \cup \{y_i = i \times Y/2^N, \quad 1 \le i \le 2^N\}$. OP-Enc$(\cdot, \cdot)$ is the encryption algorithm of an order-preserving encryption scheme Π_{OPE} and Hash is a hash function. genMT$(\cdot)$ denotes the function of using leaf nodes to generate the complete Merkle tree, Translate^{-1} denotes the function of converting the location record into the actual geographic location, and $\text{Sig}_{LDO}(\cdot)$ denotes the LDO's signature.

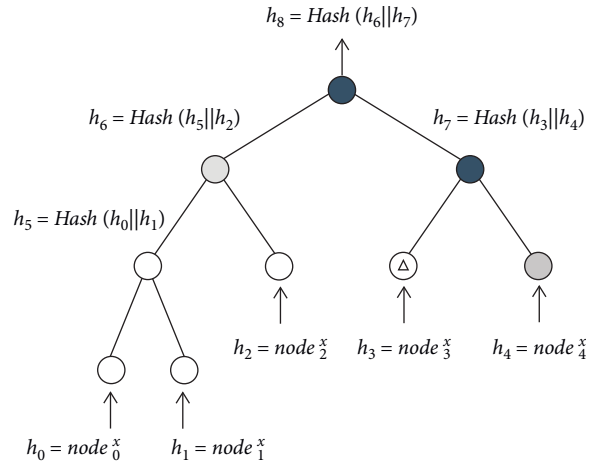

FIGURE 1: Merkle tree with leaf nodes $\text{node}_0^x, \text{node}_1^x, \text{node}_2^x, \text{node}_3^x$, and node_4^x.

3.2.2. Location Record. The LDO runs Algorithm 2 to generate a location record record_j^{LDO} and puts it into the blockchain.

In Algorithm 2, Enc$(\cdot)$ is the encryption algorithm of the symmetric encryption scheme.

3.2.3. Location Sharing. The location sharing phase consists of the location sharing stage and location verification stage.

(1) Location Sharing. When the location data requestor (LDR) wants to obtain the location corresponding to record_k^{LDO}, he generates a request request$\leftarrow \text{pub}_{LDR} \| \text{recoId}_k^{LDO} \| \text{n} \| \text{signature}_{LDR}$ and sends it to LDO.

The LDO returns location with corresponding granularity according to the trust level of the LDR. (1) If the LDR is fully trusted (level $n = \infty$), the LDO returns an accurate location. (2) If the LDR is semitrusted with level n, the LDO returns a rectangular border with side length 2^n. See Algorithm 3 for details.

In Algorithm 3, PK-Enc$(\cdot, \cdot)$ is the encryption algorithm of a public-key encryption scheme. id_1, id_2, id_3, and id_4 are the subscripts of $x_{\min}, x_{\max}, y_{\min}$, and $y_{\max}$, respectively. nodes^x and nodes^y are the Merkle tree data subsets required to authenticate $\text{node}_{id_1}^x, \text{node}_{id_2}^x$ and $\text{node}_{id_3}^y, \text{node}_{id_4}^y$, respectively.

(2) Location Verification. After receiving the response response, LDR runs Algorithm 4 to verify it.

In Algorithm 4, Dec$(\cdot, \cdot)$ is the decryption algorithm corresponding to Enc$(\cdot, \cdot)$. $A.c$ denotes c in $A = b \| c$ and MerkleHash denotes the function that uses a Merkle tree data subset to calculate its root value.

4. Cryptanalysis of Ji Et Al.'s BMPLS Scheme

Ji et al. [9] claim that their BMPLS scheme achieves decentralization, untamperability, confidentiality, multilevel privacy-preserving, retrievability, and verifiability. Unfortunately, we find that their scheme has fatal flaws in

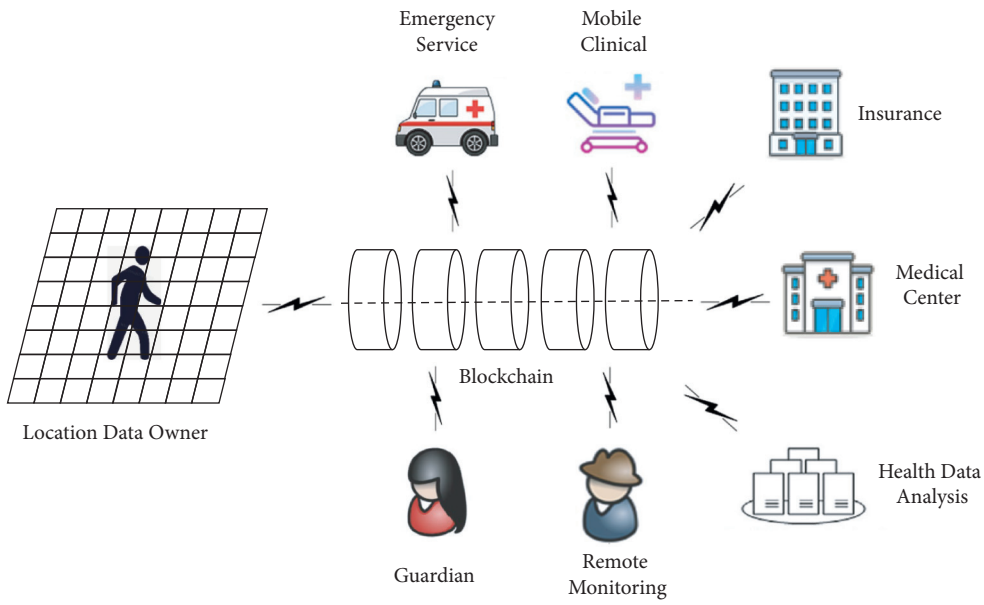

Figure 2: Architecture of BMPLS.

Input:
Region $S = \{(x, y)|0 \le x \le X, 0 \le y \le Y\}$; Maximum level of location partition N; LDO's secret key $k_{LDO} = k_{LDO}^{x} \| k_{LDO}^{y}$
Output:
Registration record regRec
(1) $\{x_1, x_2, \ldots, x_{2^N}\} \cup \{y_1, y_2, \ldots, y_{2^N}\} \longleftarrow \text{Parti}(S, N)$;
(2) **for** $i = 1; i \le 2^N; i++$ **do**
(3) $\text{ciph}_i^x = \text{OP-Enc}(k_{LDO}^x, x_i)$;
(4) $\text{ciph}_i^y = \text{OP-Enc}(k_{LDO}^y, y_i)$;
(5) $\text{node}_i^x = \text{Hash}(i \| x_i \| \text{ciph}_i^x)$;
(6) $\text{node}_i^y = \text{Hash}(i \| y_i \| \text{ciph}_i^y)$;
(7) **end for**
(8) $\text{horTree} \longleftarrow \text{genMT}(\text{node}_1^x, \text{node}_2^x, \ldots, \text{node}_{2^N}^x)$;
(9) $\text{verTree} \longleftarrow \text{genMT}(\text{node}_1^y, \text{node}_2^y, \ldots, \text{node}_{2^N}^y)$;
(10) $\text{regRec} \longleftarrow \text{Sig}_{LDO}(\text{Translate}^{-1} \| \text{horTree}_{\text{root}} \| \text{verTree}_{\text{root}})$;
(11) **return** *regRec*.

Algorithm 1: Generation of registration record.

Input:
LDO's j-th location (x_j, y_j); LDO's secret keys $k_{LDO} = k_{LDO}^{x} \| k_{LDO}^{y}, k_{sym}$; LDO's public-key pub_{LDO}
Output:
Location record record_j^{LDO}
(1) **LDO executes:**
(2) $\text{ciph}_j^x = \text{OP-Enc}(k_{LDO}^x, x_j)$;
(3) $\text{ciph}_j^y = \text{OP-Enc}(k_{LDO}^y, y_j)$;
(4) $\text{ciph}_j \longleftarrow \text{ciph}_j^x \| \text{ciph}_j^y$;
(5) $\text{OpeHash}_j \longleftarrow \text{Hash}(\text{ciph}_j)$;
(6) $\text{LocHash}_j \longleftarrow \text{Hash}(x_j \| y_j)$;
(7) $\text{SymCih}_j \longleftarrow \text{Enc}(k_{\text{sym}}, x_j \| y_j)$;
(8) $\text{LocInfo}_j \longleftarrow \text{OpeHash}_j \| \text{LocHash}_j \| \text{SymCih}_j \| \text{timetamp}_j$;
(9) $\text{record}_j^{LDO} \longleftarrow \text{pub}_{LDO} \| \text{LocInfo}_j \| \text{recId}_{j-1}^{LDO} \| \text{signature}_{LDO}$;
(10) **return** record_j^{LDO};

Algorithm 2: Generation of location record.

Input:
Location record ID recoId_k^{LDO}; Session key between LDO and LDR k_{ses}; Privacy protection level n; LDR's public-key pub_{LDR}
Output:
Shared location information response
(1) **LDR executes:**
(2) request $\longleftarrow \text{pub}_{LDR}\|\text{recoId}_k^{LDO}\|\text{n}\|\text{signature}_{LDR}$;
(3) LDR sends request to LDO;
(4) **LDO executes:**
(5) **if** $n = \infty$ **then**
(6) response $\longleftarrow \text{Enc}(k_{ses}, x_k\|y_k)\|\text{PK-Enc}(pub_{LDR}, k_{ses})$
(7) **else if** $0 \le n \le N$ **then**
(8) find the border $\{x_{\min}, x_{\max}, y_{\min}, y_{\max}\}$ in level n
(9) $\text{borInfo}_{id_1} \longleftarrow id_1\|x_{min}\|\text{ciph}_{id_1}^x$;
(10) $\text{borInfo}_{id_2} \longleftarrow id_2\|x_{\max}\|\text{ciph}_{id_2}^x$;
(11) $\text{borInfo}_{id_3} \longleftarrow id_3\|y_{\min}\|\text{ciph}_{id_3}^y$;
(12) $\text{borInfo}_{id_4} \longleftarrow id_4\|x_{\max}\|\text{ciph}_{id_4}^y$;
(13) $\text{borInfo} \longleftarrow \text{borInfo}_{id_1}\|\text{borInfo}_{id_2}\|\text{borInfo}_{id_3}\|\text{borInfo}_{id_4}$;
(14) $\text{nodes}^x \longleftarrow \{\text{node}_{x_1}^x, \text{node}_{x_2}^x, \ldots\}$;
(15) $\text{nodes}^y \longleftarrow \{\text{node}_{y_1}^y, \text{node}_{y_2}^y, \ldots\}$;
(16) response $\longleftarrow \text{Enc}(k_{ses}, \text{ciph}_k\|\text{borInfo}\|\text{nodes}^x\|\text{nodes}^y)\|\text{PK-}Enc(\text{pub}_{LDR}, k_{ses})$;
(17) **end if**
(18) **return** response.

Algorithm 3: Location sharing.

confidentiality and multilevel privacy-preserving. In addition, in some cases, their scheme cannot be executed.

4.1. On Confidentiality. The adversary can recover any LDO's location effectively.

Ji et al.'s BMPLS scheme uses the one-way property of the hash function to protect location. It is feasible and secure in the case of infinite (enough) locations. However, it is insecure in current practical applications because the amount of user locations is not enough to resist brute force attacks. The attack is as follows:

(1) Calculate the coordinate hash table T of all possible locations of the LDO's visit region.
(2) Obtain a record record_j^{LDO} from the blockchain, and extract the hash value LocHash_j .
(3) Find out the location $x_j\|y_j$ corresponding to LocHash_j in table T.

We take the commonly used Global Positioning System (GPS) as an example to evaluate the feasibility of the above attack. In the GPS, the longitude and latitude output formats are *dddmm.mmmm* and *ddmm.mmmm*, respectively. So there are only 3.60×10^{11} positions in a square area of 1° in longitude and latitude. If estimated by 40°' north latitude where New York City is located, 1 degree of longitude and 1 degree of latitude are equivalent to 85 and 111 kilometers, respectively, and a square area with 1 degree of longitude and latitude is 9,435 square kilometers. The land area of New York City is 789 square kilometers, so the number of available GPS locations is about 3.01×10^{10}.

We experiment on a personal computer using Python-3.9.7 and PyCharm Community Edition 2021.2.2 (64 bits). The configuration is CPU: Intel(R) Core(TM) i9-10900 CPU @ 2.80 GHz~2.81 GHz, RAM: 64 G, and OS: Windows 10 Home (Chinese) 64 bits (10.0.19041). As in [9], we also choose SHA-256 as the hash function. The simulation result shows that the time cost to calculate the hash values of 10^9 position coordinates is 27.01 minutes, and the size of the coordinate hash table is 77.20 GB. Therefore, using such a personal computer can calculate the hash value of all locations of New York City in about 13 hours and 33 minutes.

The above analysis and experiments show that if Ji et al.'s BMPLS scheme is used for New York City, the adversary can completely break the system in about 13.55 hours using a personal computer.

If we use supercomputers or adopt distributed computing or cloud computing technology, we can break the system with very little time overhead.

4.2. On Multilevel Privacy-Preserving. Semitrusted LDR Can Obtain Accurate Location. Ji et al. claim that, in their scheme [9], semitrusted LDR is impossible to reduce the privacy protection region. Unfortunately, we found that semitrusted LDR can recover the LDO's accurate location effectively. This is a more severe attack than reducing the privacy protection region. The attack is as follows:

(1) Obtain a record record_j^{LDO} from the blockchain, and extract the hash value LocHash_j.
(2) Run Algorithm 3 interactively with LDO to obtain the response response.
(3) Run Algorithm 4 to verify the response response, and extract the rectangular border $\{x_{\min}, x_{\max}, y_{\min}, y_{\max}\}$.

Input:
Response from LDO response; Record in the blockchain $\text{recold}_k^{L\ DO}$; Session key with LDO k_{ses}
Output:
Boolean variable b
(1) **initialize** $b \leftarrow$ False;
(2) **if** $x_k' \| y_k' \leftarrow \text{Dec}(k_{ses}, \text{response})$ **then**
(3) **if** $\text{Hash}(x_k' \| y_k')$ $\text{recold}_k^{L\ DO}.\text{LocInfo}_k$ LocHash_k **then**
(4) $b \leftarrow$ True
(5) **end if**
(6) **else if** $ciph_k \| \text{borInfo} \| \text{nodes}^x \| \text{nodes}^y \| \leftarrow \text{Dec}(k_{ses}, \text{response})$ **then**
(7) $borInfo_{id_1} \| borInfo_{id_2} \| borInfo_{id_3} \| borInfo_{id_4} \| \leftarrow borInfo$
(8) $\text{node}_{id_1}^x \leftarrow \text{Hash}(borInfo_{id_1})$;
(9) $\text{node}_{id_2}^x \leftarrow \text{Hash}(borInfo_{id_2})$;
(10) $\text{node}_{id_3}^x \leftarrow \text{Hash}(borInfo_{id_3})$;
(11) $\text{node}_{id_4}^x \leftarrow \text{Hash}(borInfo_{id_4})$;
(12) $horTree_{root}' \leftarrow \text{MerkleHash}\{\text{nodes}^x, \text{node}_{id_1}^x, \text{node}_{id_2}^x\}$;
(13) $vorTree_{root}' \leftarrow \text{MerkleHash}\{\text{nodes}^y, \text{node}_{id_3}^y, \text{node}_{id_4}^y\}$
(14) **if** $horTree_{toot}' = horTree_{root}$ and $vorTree_{toot}' = verTree_{root}$ **then**
(15) **if** $\text{Hash}(ciph_k) = recoId_k^{\text{LDO}}.LocInfo_k.OpeHash_k$ **and** $borInfo_{id_1}.ciph_{id_1}^x < ciph_k.ciph_k^x < borInfo_{id_2}.ciph_{id_2}^x$ **and** $borInfo_{id_3}.ciph_{id_3}^x < ciph_k.ciph_k^y < borInfo_{id_4}.ciph_{id_4}^x$ **then**
(16) $b \leftarrow$ True
(17) **end if**
(18) **end if**
(19) **end if**
(20) **return** b;

Algorithm 4: Location verification.

Input:
Location hash value LocHash_j; Rectangular border $\{x_{\min}, x_{\max}, y_{\min}, y_{\max}\}$,
Output:
Location information $x_j \| y_j$
(1) **for** x *in* $(x_{\min}, x_{\max})$ **do**
(2) **for** y *in* $(y_{\min}, y_{\max})$ **do**
(3) $\text{Hash} \longleftarrow \text{Hash}(x \| y)$;
(4) **if** $\text{Hash} = \text{LocHash}_j$ **then**
(5) **return** $x \| y$;
(6) **end if**
(7) **end for**
(8) **end for**

Algorithm 5: Location information extraction.

(4) Search for the location where the hash value is equal to $\text{record}_j^{LDO}.\text{LocInfo}_j.\text{LocHash}_j$ in the rectangular border $\{x_{\min}, x_{\max}, y_{\min}, y_{\max}\}$; that is, run Algorithm 5 to obtain the accurate location information $x_j \| y_j$.

Take $x_{\min} = 00587654$, $x_{\max} = 01012345$ as an example to illustrate how to make x traverse $(x_{\min}, x_{\max})$. Because order-preserving encryption requires the plaintext to be a positive integer, the format of the location needs to be changed from *ddmm.mmmm* to *ddmmmmmm*. Note that there are no location coordinates like *dd*60*mmmm*, *dd*61*mmmm*, ... *dd*99*mmmm*, and (00587654, 01012345) must be divided into (00587654, 00599999) and (01000000, 01012345). Then, the former is expressed as "$x = 00587655; x \le 0059999; x++$" and the latter as "$x = 01000000; x \le 01012344; x++$."

We evaluated the feasibility of this attack in the above environment. Assume that the rectangular border $\{x_{\min}, x_{\max}, y_{\min}, y_{\max}\}$ is a square area with a longitude and latitude of 0.5 minutes each. This is a rectangular area with a length of 928 meters and a width of 710 meters. The simulation result shows that the average cost to calculate the accurate location $x_j \| y_j$ in such an area is 20.75 seconds.

If a supercomputer is used or distributed computing or cloud computing technology is adopted, the time for LDR to find the accurate location will be milliseconds.

4.3. Other Weaknesses

(1) In [9], the coordinates $x_i = i \times X/2^N$ and $y_i = i \times Y/2^N$ may not be integers. But in Definition 1 of [9],

the plaintext space is clearly defined as the integer set $[m] = \{i | 1 \le i \le m\}$. Therefore, *it will not be feasible to encrypt x_i and y_i in* Algorithm 1 *(lines 3 and 4).*

(2) The LDO's location coordinate x_k or y_k may happen to be a position grid coordinate x_i or y_i . If it happens, *multilevel privacy protection will not be achieved.* For example, if $x_k = x_{2^{N-1}}$, then we have $x_k = x_{\min}$ or $x_k = x_{\max}$ for any trust level *n.* Then, the inequality in line 15 of Algorithm 4 will not hold, so LDR will not accept any rectangular border provided by LDO. Further, because $\text{borInfo}_{id_1}.\ \text{ciph}^x_{id_1} = \text{ciph}_k.\text{ciph}^x_k$ or $\text{ciph}_k.\text{ciph}^x_k = \text{borInfo}_{id_2}.\ \text{ciph}^x_{id_2}$, LDR will determine that the x-coordinate of LDO is $x_{\min}$ or $x_{\max}$.

(3) The n-level rectangular border $\{x_{\min}, x_{\max}, y_{\min}, y_{\max}\}$ must be in the form of $\{x_{i\times 2^n}, x_{(i+1)\times 2^n}, y_{j\times 2^n}, y_{(j+1)\times 2^n}\}$, to ensure that the privacy protection region will not be reduced. If the LDO's location coordinate x_k or y_k satisfies $x_k < x_1$ or $y_k < y_1$, then have $x_{\min} = x_0 = 0$ or $y_{\min} = y_0 = 0$. In this case, both Algorithms 3 and 4 *cannot be executed.*

(a) Because 0 is not in the plaintext space of order-preserving encryption, Algorithm 3 cannot generate the ciphertext $ciph^x_{id_1}$ (or $\text{ciph}^y_{id_2}$) of $x_{\min} = 0$ (or $y_{\min} = 0$) (see Definition 1 of [9]).

(b) node^x_0 and node^y_0 are used when calculating $\text{horTree}'_{\text{root}}$ and $\text{vorTree}'_{\text{root}}$, but they are not used when calculating horTree and verTree, which will make neither $\text{horTree}'_{\text{toot}} = \text{horTree}_{\text{root}}$ nor $\text{vorTree}'_{\text{toot}} = \text{verTree}_{\text{root}}$ holds.

5. Improvement

We present an improvement to fix our attacks in this section and optimize the BMLS scheme to make it correct and executable.

5.1. Ideas for Improvement. The main improvement ideas are as follows:

(1) Add **Setup** and **Key generation** phases to the scheme to provide the foundation for other phases.

(2) Replace the **Location verification** phase with the **Location extraction** phase. The original location verification algorithm only returns the verification result but not the location, while our extraction algorithm returns the location with the verification result.

(3) Use "Salt" to prevent brute force attacks. When calculating the hash value LocHash_j of the location $x_j \| y_j$, select a random number (salt) r_j and let $\text{LocHash}_j = \text{Hash}(x_j \| y_j \| r_j)$. As long as the length of the random number is large enough, the brute force attacks in Sections 4.1 and 4.2 cannot be implemented, even with supercomputers or cloud computing.

(4) Use integer steps to generate grid coordinates to ensure that the coordinate values are positive integers.

(5) When the LDO's location coordinate x_k or y_k is equal to the grid coordinate x_i or y_i, we make a minimum positive disturbance to the location coordinates to ensure they are not equal. In GPS, the minimum positive disturbance is 0.0001 minutes. This disturbance is only 18.5 cm and does not affect the actual performance of the system.

5.2. Improved Scheme. The improved BMPLS scheme consists of the following five phases.

5.2.1. Setup

(1) Choose an order-preserving encryption scheme Π_{OPE} = (OP-KeyGen, OP-Enc) and set up public parameters [25].

(2) Choose a public-key encryption scheme Π_{PKE} = (PK-KeyGen, PK-Enc, PK-Dec), such as RSA1024 and RSA2048, and set up public parameters.

(3) Choose a signature scheme Π_{Sig} = (Sig-KeyGen, Sig, Ver), such as DSA1024 and DSA2048, and set up public parameters.

(4) Choose a symmetric encryption scheme Π_{SE} = (S-Enc, S-Dec), such as AES128 and AES256, and set up public parameters.

(5) Choose a one-way collision resistance hash function Hash, such as SHA256 and SHA512.

(6) Set up and deploy a membership service provider (MSP) to maintain the identity of all users in the system and issue certificates for authentication and signature verification [27].

(7) Set up and deploy a trust level service provider (TLSP) to maintain the trust level of all users in the system and provide trust level query services [28].

5.2.2. Key Generation

(1) LDO runs Π_{OPE}'s key generation algorithm OP-KeyGen twice and obtains his x-coordinate encryption secret key opk^x_{LDO} and y-coordinate encryption secret key opk^y_{LDO}, respectively.

(2) LDO runs Π_{Sig}'s key generation algorithm Sig-KeyGen and obtains his public-key pk_{LDO} and signing key sk_{LDO}.

(3) LDO chooses his secret key k_{LDO} for Π_{SE}.

(4) LDR runs Π_{PKE}'s key generation algorithm PK-KeyGen and obtains his public-key pk_{LDR} and private key sk_{LDR}.

5.2.3. Initialization. LDO selects the origin $(\hat{x}_0, \hat{y}_0)$, step lengths x_{sl}, y_{sl}, and partition level N such that $\{(x, y) | \hat{x}_0 \le x \le \hat{x}_0 + 2^N \times x_{sl}, \hat{y}_0 \le y \le \hat{y}_0 + 2^N \times y_{sl}\}$ can cover his visit region properly. Assume that $\hat{x}_0, \hat{y}_0, x_{sl}$, and y_{sl} have all been converted into positive integers. For example, convert *dddmm.mmmm* and *ddmm.mmmm* into

Input:
Region region = $\{\hat{x}_0, \hat{y}_0, x_{sl}, y_{sl}, N\}$; LDO's secret keys opk^x_{LDO} and opk^y_{LDO}; LDO′ signing key sk_{LDO}
Output:
Registration record regRec

(1) **for** $i = 0; i \leq 2^N; i++$ **do**;
(2) $x_i = i \times x_{sl}$;
(3) $y_i = i \times y_{sl}$;
(4) $\text{ciph}^x_i = \text{OP-Enc}(opk^x_{LDO}, x_i)$;
(5) $\text{ciph}^y_i = \text{OP-Enc}(opk^y_{LDO}, y_i)$;
(6) $\text{node}^x_i = \text{Hash}(i \| x_i \| \text{ciph}^x_i)$;
(7) $\text{node}^y_i = \text{Hash}(i \| y_i \| \text{ciph}^y_i)$;
(8) **end for**
(9) $\text{horTree} \longleftarrow \text{genMT}(\text{node}^x_0, \text{node}^x_1, \text{node}^x_2, \ldots, \text{node}^x_{2^N})$;
(10) $\text{verTree} \longleftarrow \text{genMT}(\text{node}^y_0, \text{node}^y_1, \text{node}^y_2, \ldots, \text{node}^y_{2^N})$;
(11) $\text{regRec} \longleftarrow \text{region} \| \text{horTree}_{\text{root}} \| \text{verTree}_{\text{root}} \| \text{Sig}(sk_{LDO}, \text{region} \| \text{horTree}_{\text{root}} \| \text{verTree}_{\text{root}})$;
(12) **return** regRec.

Algorithm 6: Registration record generation.

dddmmmmmm and *ddmmmmmm*, respectively. We choose the order-preserving encryption scheme Π_{OPE} that allows the plaintext to be 0.

Denote the visit region as region = $\{\hat{x}_0, \hat{y}_0, x_{sl}, y_{sl}, N\}$. LDO executes Algorithm 6 to generate the registration record regRec and puts it into the blockchain.

5.2.4. Location Record. Suppose that the j-th location of the LDO is $(\hat{x}_0 + x_j, \hat{y}_0 + y_j)$. LDO runs Algorithm 7 to generate a location record record^{LDO}_j and puts it into the blockchain.

5.2.5. Location Sharing. The location sharing phase consists of the location sharing stage and location extraction stage.

(1) Location Sharing. When LDR wants to obtain the location corresponding to record^{LDO}_k, he generates a request request = $\text{recoId}^{LDO}_k \| n \| \text{Sig}(sk_{LDR}, \text{recoId}^{LDO}_k \| n)$ and sends it to LDO.

LDO generates a response response and returns it to LDR as follows:

(1) It requests the LDR's certificate from the membership service provider (MSP) and then verifies the request. If it is invalid, it returns response = $\perp$ and aborts.
(2) It requests the LDR's privacy protection level n_{LDR} from the trust level service provider (TLSP). If $n_{LDR} \neq \infty$ and $n < n_{LDR}$, it returns response = $\perp$ and aborts.
(3) It gets the location record record^{LDO}_k from the blockchain.
(4) It runs Algorithm 8 and returns response to LDR.

(2) Location Extraction. Receiving the response response, LDR gets the record record^{LDO}_k from the blockchain and then runs Algorithm 9 to extract the location information.

5.3. Security and Performance Analysis

5.3.1. Security Analysis. We follow the definitions of [9] to analyze the security of the improved BMPLS scheme as follows:

(1) Decentralization. One of the main advantages of blockchain is decentralization. The BMLS scheme uses blockchain to manage and store location information, so it inherits decentralization.

(2) Untamperability. The unforgeability mentioned in [9] is just untamperability, which is another main property of blockchain. For the same reason as above, the BMLS scheme also inherits untamperability.

(3) Confidentiality. In the improved BMPLS scheme, the only data that needs confidentiality protection is LDO's location coordinate (x_j, y_j). The scheme uses four different cryptographic techniques to protect it: hash function, symmetric encryption, order-preserving encryption, and hybrid encryption. Because the hash function cannot achieve indistinguishable security, the improved BMPLS scheme can only achieve confidentiality in the "all-or-nothing" sense, that is, *prevent the adversary from recovering the location coordinates*. We analyze the probability of a successful attack in different situations as follows:

(1) *Get* (x_j, y_j) *from* $\text{LocHash}_j = \text{Hash}(x_j \| y_j \| r_j)$. Let k be the binary bit length of r_j. According to the one-way property of the hash function, the probability that the adversary obtains $x_j \| y_j$ from LocHash_j is less than 2^{-k}, even if he knows a small rectangular border. When k is large enough, such as $k = 160$, the probability is negligible.
(2) *Get* (x_j, y_j) *from* $\text{SymCih}_j = \text{S-Enc}(k_{LDO}, x_j \| y_j \| r_j)$. Under the assumption that the symmetric encryption scheme Π_{SE} is secure, the probability that

Input:
LDO's j-th location (x_j, y_j); Region region = $\{\widehat{x}_0, \widehat{y}_0, x_{sl}, y_{sl}, N\}$; LDO's secret keys $(opk^x_{LDO}, opk^y_{LDO})$ and k_{LDO}; LDO's signing key sk_{LDO}
Output:
Location record record^{LDO}_j

(1) **for** $i = 0; i \le 2^N; i++$ **do**;
(2) **if** $x_j = i \times x_{sl}$ **then**
(3) $x_j \longleftarrow x_j + 1$;
(4) **end if**
(5) **if** $y_j = i \times y_{sl}$ **then**
(6) $y_j \longleftarrow y_j + 1$;
(7) **end if**
(8) **end for**
(9) $\text{ciph}^x_j = \text{OP-Enc}(opk^x_{LDO}, x_j)$;
(10) $\text{ciph}^y_j = \text{OP-Enc}(opk^y_{LDO}, y_j)$;
(11) $\text{ciph}_j \longleftarrow \text{ciph}^x_j \| \text{ciph}^y_j$;
(12) $\text{OpeHash}_j \longleftarrow \text{Hash}(\text{ciph}_j)$;
(13) Generate a random number r_j;
(14) $\text{LocHash}_j \longleftarrow \text{Hash}(x_j \| y_j \| r_j)$;
(15) $\text{SymCih}_j \longleftarrow S - \text{Enc}(k_{LDO}, x_j \| y_j \| r_j)$;
(16) $\text{LocInfo}_j \longleftarrow \text{OpeHash}_j \| \text{LocHash}_j \| \text{SymCih}_j \| \text{timetamp}_j$;
(17) $\text{record}^{LDO}_j \longleftarrow \text{LocInfo}_j \| \text{recId}^{LDO}_{j-1} \| \text{Sig}(sk_{LDO}, \text{LocInfo}_j \| \text{recId}^{LDO}_{j-1})$;
(18) **return** record^{LDO}_j;

Algorithm 7: Location record generation.

Input:
Location record record^{LDO}_k; LDO's secret keys k_{LDO} and $(opk^x_{LDO}, opk^y_{LDO})$; Privacy protection level n; LDR's public-key pk_{LDR}
Output:
Shared location information *respons*

(1) $x_k \| y_k \| r_k \longleftarrow S - \text{Dec}(k_{LDO}, \text{record}^{LDO}_k.\text{LocInfo}_k.\text{SymCih}_k)$;
(2) choose a session key k_{ses}
(3) **if** $n = \infty$ **then**
(4) response $\longleftarrow$ S-Enc$(k_{ses}, x_k \| y_k \| r_k)$PK-Enc$(pk_{LDR}, k_{ses})$
(5) **else if** $0 \le n \le N$ **then**
(6) find i such that $i \times 2^n \times x_{sl} < x_k < (i+1) \times 2^n \times x_{sl}$
(7) $id_1 \longleftarrow i \times 2^n$; $id_2 \longleftarrow (i+1) \times 2^n$;
(8) $x_{\min} \longleftarrow i \times 2^n \times x_{sl}$; $x_{\max} \longleftarrow (i+1) \times 2^n \times x_{sl}$;
(9) find j such that $j \times 2^n \times y_{sl} < y_k < (j+1) \times 2^n \times y_{sl}$
(10) $id_3 \longleftarrow j \times 2^n$; $id_4 \longleftarrow (j+1) \times 2^n$;
(11) $y_{\min} \longleftarrow j \times 2^n \times y_{sl}$; $y_{\max} \longleftarrow (j+1) \times 2^n \times y_{sl}$;
(12) $\text{ciph}^x_{id_1} = \text{OP-Enc}(opk^x_{LDO}, x_{\min})$; $\text{ciph}^x_{id_2} = \text{OP-Enc}(opk^x_{LDO}, x_{\max})$;
(13) $\text{ciph}^y_{id_3} = \text{OP-Enc}(opk^y_{LDO}, y_{\min})$; $\text{ciph}^y_{id_4} = \text{OP-Enc}(opk^y_{LDO}, y_{\max})$;
(14) $\text{borInfo}_{id_1} \longleftarrow id_1 \| x_{\min} \| \text{ciph}^x_{id_1}$; $\text{borInfo}_{id_2} \longleftarrow id_2 \| x_{\max} \| \text{ciph}^x_{id_2}$;
(15) $\text{borInfo}_{id_3} \longleftarrow id_3 \| y_{\min} \| \text{ciph}^y_{id_3}$; $\text{borInfo}_{id_4} \longleftarrow id_4 \| x_{\max} \| \text{ciph}^y_{id_4}$;
(16) $\text{borInfo} \longleftarrow \text{borInfo}_{id_1} \| \text{borInfo}_{id_2} \| \text{borInfo}_{id_3} \| \text{borInfo}_{id_4}$;
(17) $\text{nodes}^x \longleftarrow \{\text{node}^x_{x_1}, \text{node}^x_{x_2}, \ldots\}$; $\text{nodes}^y \longleftarrow \{\text{node}^y_{y_1}, \text{node}^y_{y_2}, \ldots\}$;
(18) $\text{ciph}^x_k = \text{OP-Enc}(opk^x_{LDO}, x_k)$; $\text{ciph}^y_k = \text{OP-Enc}(opk^y_{LDO}, y_k)$;
(19) $\text{ciph}_k \longleftarrow \text{ciph}^x_k \| \text{ciph}^y_k$;
(20) response $\longleftarrow$ S-Enc$(k_{ses}, \text{ciph}_k \| \text{borInfo} \| \text{nodes}^x \| \text{nodes}^y) \| \text{PK-Enc}(pk_{LDR}, k_{ses})$;
(21) **end if**
(22) **return** response.

Algorithm 8: Location sharing.

Input:
Response from LDO response; Record in the blockchain record_k^{LDO}; LDR's private key sk_{LDR}
Output:
Location $x_k \| y_k$ or rectangular border $\{x_{\min}, x_{\max}, y_{\min}, y_{\max}\}$, or False

(1) $k_{ses} \longleftarrow$ PK-Dec $(sk_{LDR}, \text{response.PK-Enc}(pk_{LDR}, k_{ses}))$;
(2) **if** $x_k' \| y_k' \| r_k' \longleftarrow$ S-Dec $(k_{ses}, \text{response})$ **then**
(3) **if** Hash $(x_k' \| y_k' \| r_k') = \text{record}_k^{LDO}.\ \text{LocInfo}_k.\text{LocHash}_k$ **then**
(4) return $x_k' \| y_k'$;
(5) **end if**
(6) **else if** $\text{ciph}_k \| \text{borInfo} \| \text{nodes}^x \| \text{nodes}^y \longleftarrow$ Dec $(k_{ses}, \text{response})$ **then**
(7) $\text{borInfo}_{id_1} \| \text{borInfo}_{id_2} \| \text{borInfo}_{id_3} \| \text{borInfo}_{id_4} \longleftarrow \text{borInfo}$
(8) $\text{node}_{id_1}^x \longleftarrow \text{Hash}(\text{borInfo}_{id_1})$; $\text{node}_{id_2}^x \longleftarrow \text{Hash}(\text{borInfo}_{id_2})$;
(9) $\text{node}_{id_3}^y \longleftarrow \text{Hash}(\text{borInfo}_{id_3})$; $\text{node}_{id_4}^y \longleftarrow \text{Hash}(\text{borInfo}_{id_4})$;
(10) $\text{horTree}'_{\text{root}} \longleftarrow \text{MerkleHash}\{\text{nodes}^x, \text{node}_{id_1}^x, \text{node}_{id_2}^x\}$;
(11) $\text{vorTree}'_{\text{root}} \longleftarrow \text{MerkleHash}\{\text{nodes}^y, \text{node}_{id_3}^y, \text{node}_{id_4}^y\}$;
(12) **if** $\text{horTree}'_{\text{toot}} = \text{horTree}_{\text{root}}$ and $\text{vorTree}'_{\text{toot}} = \text{verTree}_{\text{root}}$ **then**
(13) **if** Hash $(\text{ciph}_k) = \text{recoId}_k^{LDO}.\text{LocInfo}_k.\text{OpeHash}_k$ and
$\text{borInfo}_{id_1}.\text{ciph}_{id_1}^x < \text{ciph}_k.\text{ciph}_k^x < \text{borInfo}_{id_2}.\text{ciph}_{id_2}^x$ and
$\text{borInfo}_{id_3}.\text{ciph}_{id_3}^x < \text{ciph}_k.\text{ciph}_k^y < \text{borInfo}_{id_4}.\text{ciph}_{id_4}^x$ **then**
(14) $x_{\min} \longleftarrow \text{borInfo}_{id_1}.x_{\min}$; $x_{\max} \longleftarrow \text{borInfo}_{id_2}.x_{\max}$;
(15) $y_{\min} \longleftarrow \text{borInfo}_{id_3}.y_{\min}$; $y_{\max} \longleftarrow \text{borInfo}_{id_4}.y_{\max}$;
(16) return$\{x_{\min}, x_{\max}, y_{\min}, y_{\max}\}$;
(17) **end if**
(18) **end if**
(19) **end if**
(20) **return** False.

Algorithm 9: Location extraction.

the adversary obtains $x_j \| y_j$ from SymCih_j is negligible.

(3) *Get* (x_k, y_k) *from* response = S-Enc $(k_{ses}, x_j \| y_j \| r_j) \| \text{PK-Enc}(pk_{LDR}, k_{ses})$. If the selected symmetric encryption scheme Π_{SE} and public-key encryption scheme Π_{PKE} are secure, then the probability that the adversary obtains (x_k, y_k) from S-Enc $(k_{ses}, x_k \| y_k \| r_k) \| \text{PK-Enc}(pk_{LDR}, k_{ses})$ is also negligible.

(4) *Get* (x_k, y_k) *from* $ciph_k^x = \text{OP-Enc}\ (opk_{LDO}, x_k)$; $\text{ciph}_k^y = \text{OP-Enc}(opk_{LDO}, y_k)$. Similarly, if the selected order-preserving encryption scheme Π_{OPE} is secure, the probability of recovering the plaintext from the ciphertext is also negligible.

Therefore, the probability that the adversary obtains the LDO's location coordinate (x_j, y_j) is negligible, and the improved BMPLS scheme achieves confidentiality.

(4) Multilevel Location Privacy Protection. Multilevel location privacy protection aims to ensure that the semitrusted LDR cannot reduce the rectangular border of the LDO's location.

Firstly, confidentiality guarantees that semitrusted LDR cannot obtain the exact location coordinates of LDO.

Secondly, the n-level rectangular border obtained by LDR is $\{x_{\min}, x_{\max}, y_{\min}, y_{\max}\}$, where $x_{\min} = i \times 2^n \times x_{sl}$, $x_{\max} = (i+1) \times 2^n \times x_{sl}$, $y_{\min} = j \times 2^n \times y_{sl}$, and $y_{\max} = (j+1) \times 2^n \times y_{sl}$. Requesting n-level rectangular borders at different times will get the same results, which will not help reduce the rectangular border. In addition, when $n_1 < n_2$, the n_1-level rectangular border must be included in the n_2-level rectangular border. It is impossible to reduce the rectangular border by finding the intersection of the rectangular borders of different levels.

Thirdly, because the order-preserving encryption algorithm OP-Enc is a pseudorandom order-preserving function controlled by a key, as long as the ciphertext space is large enough, such as 2^{160}, the probability of reversing ciph_i^x (or ciph_i^y) from $\text{node}_i^x = \text{Hash}(i \| x_i \| \text{ciph}_i^x)$ (or $\text{node}_i^y = \text{Hash}(i \| y_i \| \text{ciph}_i^y)$) will be negligible. Therefore, the adversary's probability of reducing the rectangular border by finding the coordinate ciphertext is negligible.

Finally, the x-coordinate and y-coordinate are encrypted with different keys, so there is no order-preserving relationship between the ciphertext of the x-coordinate and the ciphertext of the y-coordinate. $\text{ciph}_k^x < \text{ciph}_i^y$ (or $\text{ciph}_k^y > \text{ciph}_i^y$) does not imply $x_k < y_i$ (or $x_k > y_i$). It is impossible for the adversary to use $y_{\min}$ and $y_{\max}$ to narrow the range of x_k. The same is true for y_k.

(5) Retrievability. Retrievability means that the LDO's location coordinates can be retrieved effectively [9]. In the improved BMPLS scheme, the location coordinate (x_j, y_j) is stored in the blockchain via ciphertext SymCih_j = S-Enc $(k_{LDO}, x_j \| y_j \| r_j)$. LDO knows the decryption key k_{LDO} and can retrieve the location coordinate efficiently. On the other hand, in the location sharing phase, the LDO sends the

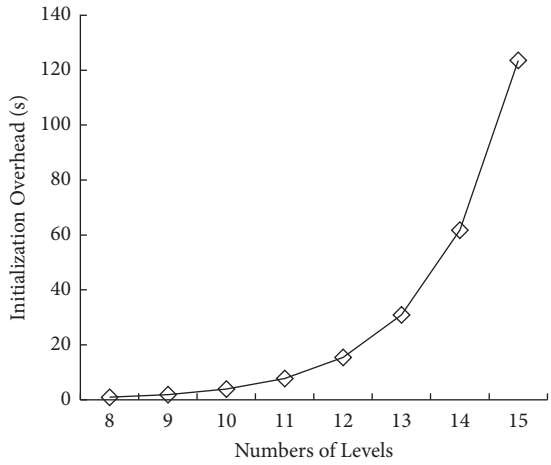

Figure 3: Initialization overhead.

location coordinate (x_k, y_k) to the LDR in the form of response = S-Enc($k_{ses}, x_k \| y_k \| r_k$)$\|$PK-Enc$(pk_{LDR}, k_{ses})$. LDR can use its decryption key sk_{LDR} to decrypt the location coordinate (x_k, y_k). So, the improved BMPLS scheme achieves retrievability.

(6) Verifiability. Verifiability means that LDR can use blockchain to verify whether the location information he gets is correct. The improved BMPLS scheme provides different verification methods in two cases:

(1) The trusted LDR firstly decrypts the location coordinates and random number $x'_k \| y'_k \| r'_k$ and calculates the hash value Hash$(x'_k \| y'_k \| r'_k)$. Then, he extracts the hash value LocHash_k corresponding to the coordinate from the blockchain and checks whether Hash$(x'_k \| y'_k \| r'_k) = \text{LocHash}_k$. The untamperability of the blockchain and the signature of the LDO ensure that LocHash_k will not be tampered with, and the collision resistance of the hash function ensures that $x'_k \| y'_k \| r'_k$ will not be replaced, so (x'_k, y'_k) is correct.

(2) The semitrusted LDR gets the coordinates of a rectangular border $\{x_{\min}, x_{\max}, y_{\min}, y_{\max}\}$, with their order-preserving ciphertext $\{\text{ciph}^x_{id_1}, \text{ciph}^x_{id_2}, \text{ciph}^y_{id_3}, \text{ciph}^y_{id_4}\}$, and then verifies it by the Merkle tree and comparing the order of the ciphertext. Similarly, the untamperability and the signature ensure that $\text{horTree}_{\text{root}}$ and $\text{verTree}_{\text{root}}$ will not be tampered with, and the collision resistance of the Merkle tree ensures that $\text{ciph}^x_{id_1}, \text{ciph}^x_{id_2}, \text{ciph}^y_{id_3}$, and $\text{ciph}^y_{id_4}$ are correct. Finally, the property of order-preserving encryption ensures that LDO's location (x_k, y_k) is in the rectangular border $\{x_{\min}, x_{\max}, y_{\min}, y_{\max}\}$.

The security comparison of related schemes is shown in Table 2. It shows that our improved scheme has apparent advantages in security.

5.3.2. Performance Analysis. We perform performance simulations under the same environment in Section 4. To ensure security, we set the ciphertext space of order-preserving encryption to be 2^{160}, denoted as OPE (2^{160}). Choose SHA, AES, RSA, and DSA as hash, symmetric encryption, public-key encryption, and digital signature algorithms. We simulate on two data scales, AES128 + RSA1024 + DSA1024 and AES256 + RSA2048 + DSA2048. We also use two hash functions, SHA256 and SHA512. Since the time cost of the hash function is very small, it has no significant impact on the simulation. The simulation results of each phase are as follows:

(1) Initialization. For initialization, we simulated the partition level N from 8 to 15. The result is shown in Figure 3. When $N = 10$, the LDO's initialization time is only 3.87 seconds. Estimated with a step length of 10 meters, the visit region region is a rectangle with a side length of 10.24 kilometers in this case. When $N = 13$, the LDO's initialization time is 30.89 seconds. In this case, the visit region's side length is 81.92 kilometers, which can meet the needs of most scenarios. When $N = 15$, the initialization of the LDO takes 123.53 seconds. At this time, the side length of the visit region has been as long as 327.68 kilometers.

The simulation shows that the initialization time is generally tens of seconds. Because each LDO only needs to be initialized once, this overhead is acceptable.

(2) Location Record. We conducted eight simulations of location record generation, and the results are shown in Figure 4. In the case of OPE(2^{160}), AES128, and DSA1024, the minimum overhead is 7.81 ms, the maximum is 8.17 ms, and the average is 8.00 ms. When the case of OPE(2^{160}), AES256, and DSA2048, the minimum is 8.53 ms, the maximum is 9.05 ms, and the average is 8.78 ms. In both cases, the overheads of location record generation are very small.

(3) Location Sharing. There are two cases of location sharing: $n = \infty$ (trusted LDR) and $0 \le n \le N$ (semitrusted LDR). The simulation results are shown in Figures 5(a) and 5(b).

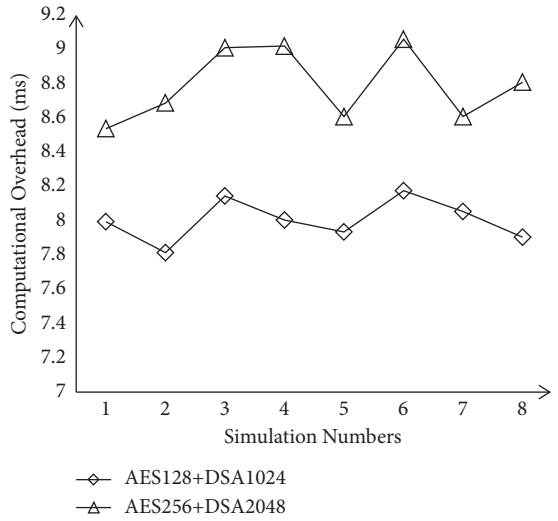

Figure 4: Record generation overhead.

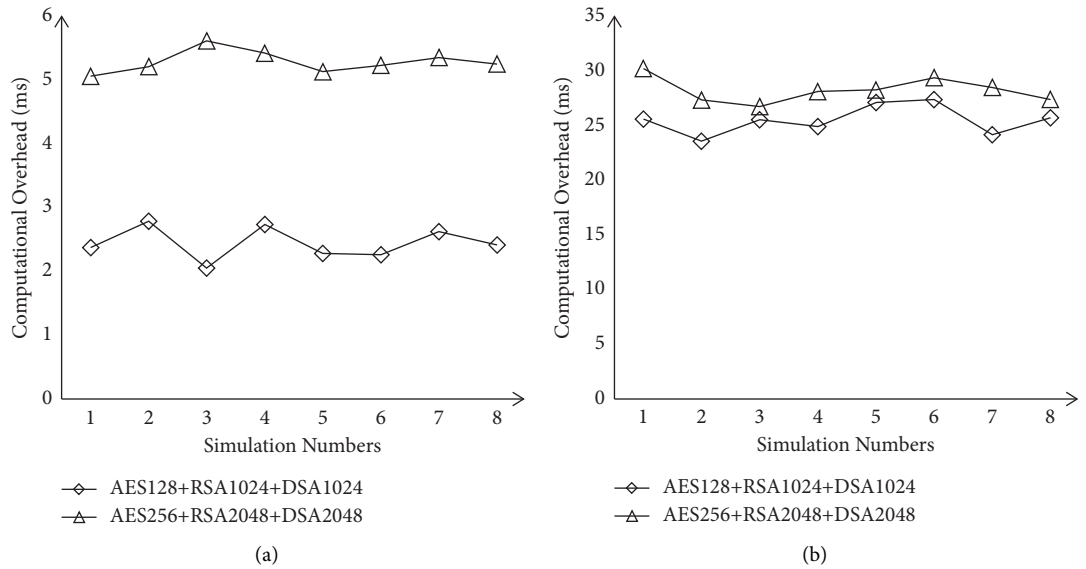

Figure 5: Location sharing overhead, (a) $n = \infty$, (b) $n = 3$.

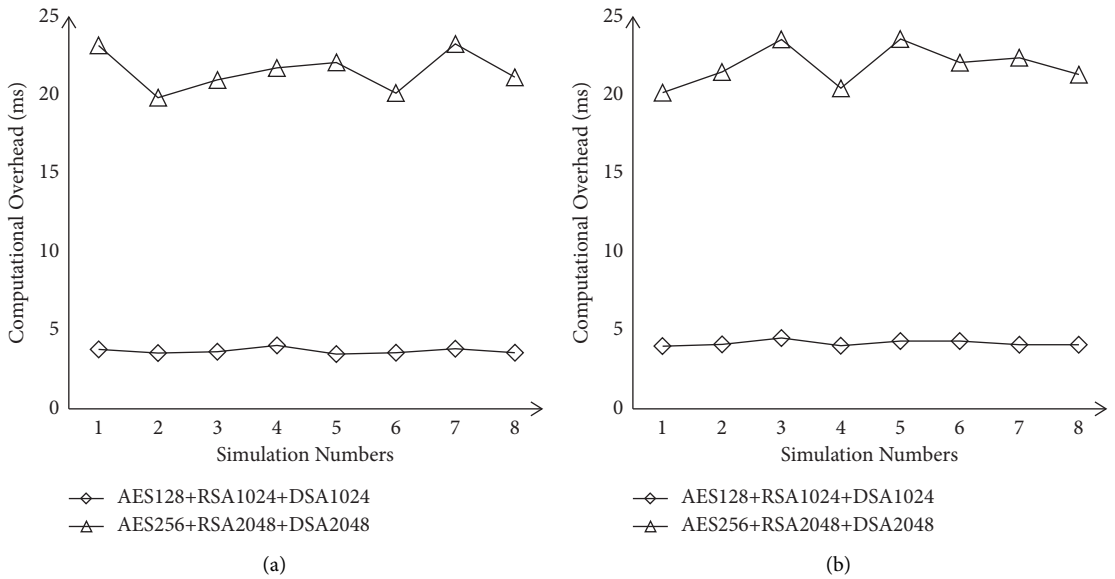

Figure 6: Location extraction overhead, (a) $n = \infty$, (b) $n = 3$.

Case of $n = \infty$. When using AES128, RSA1024, and DSA1024, the average overhead is 2.43 ms. While using AES256, RSA2048, and DSA2048, it is 5.25 ms.

Case of $0 \le n \le N$. The computational overheads increase to 25.35 ms and 28.07 ms, respectively. Most of the overhead is used to calculate six order-preserving encryption, which is about 22.70 ms. In this case, the computational overhead of our scheme is greater than that of [9] since we use order-preserving encryption with large ciphertext space, and [9] uses small ciphertext space. This is the time cost to improve security.

(4) Location Extraction. There are also two cases for location extraction, and the simulation results are shown in Figures 6(a) and 6(b). Unlike location sharing, there is little difference in overheads between the two cases. When using AES128, RSA1024, and DSA1024, the average overheads are 3.67 ms and 4.16 ms, respectively. While using AES256, RSA2048, and DSA2048, they are 21.41 ms and 21.741 ms, respectively.

The above simulation shows that the computational overhead of the location record phase is within 10 ms, and the computational overheads of the location sharing and location extraction phases are both within 30 ms. Therefore, our improved BMPLS scheme is practical.

6. Conclusion

In the telecare medical information systems, patient location information is sensitive data that needs to be protected. Blockchain technology provides a new method for patient location privacy protection and secure sharing. Recently, Ji et al. [9] proposed a blockchain-based multilevel privacy-preserving location sharing (BMPLS) scheme for telecare medical information systems. In this paper, we show that Ji et al.'s BMPLS scheme does not achieve confidentiality and multilevel privacy-preserving. Using salting technology, we propose an improved BMPLS scheme to fix our attacks. We also optimized the BMLS scheme to make it correct and executable. Analysis shows that the improved BMPLS scheme achieves all security while maintaining its high performance. Our improved BMPLS scheme is currently based on the general blockchain, and we may further study the use of edge-of-thing blockchain to enhance system performance [2, 3]. For different scenarios, designing an applicable trust level management scheme is another future research direction.

Data Availability

No data were used to support this study.

Conflicts of Interest

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

Acknowledgments

The authors would also like to thank Ms. Hongying Chen for her help in performance simulation. This work was supported by the National Social Science Fund of China (no. 21XTQ015), the Natural Science Foundation of Fujian Province of China (nos. 2019J01750, 2019J01752, and 2020J01814), the Education and Scientific Research Project for Young and Middle-Aged Teachers of Fujian Province (no. JAT201387), and the Natural Science Foundation of Zhangzhou (no. ZZ2019J27).

References

[1] S. Nakamoto, "Bitcoin: a peer-to-peer electronic cash system," pp. 1–9, 2009, https://bitcoin.org/bitcoin.pdf.

[2] P. B, N. Deepa, Q.-V. Pham et al., "Toward blockchain for edge-of-things: a new paradigm, opportunities, and future directions," *IEEE Internet of Things Magazine*, vol. 4, no. 2, pp. 102–108, 2021.

[3] T. R. Gadekallu, Q.-V. Pham, D. C. Nguyen et al., "Blockchain for edge of things: applications, opportunities, and challenges," *IEEE Internet of Things Journal*, p. 1, 2021.

[4] W. Wang, C. Qiu, Z. Yin et al., "Blockchain and PUF-based lightweight Authentication protocol for wireless medical sensor networks," *IEEE Internet of Things Journal*, p. 1, 2021.

[5] W. Wang, H. Huang, L. Zhang, and C. Su, "Secure and efficient mutual authentication protocol for smart grid under blockchain," *Peer-to-Peer Networking and Applications*, vol. 14, no. 5, pp. 2681–2693, 2021.

[6] W. Wang, H. Xu, M. Alazab, T. R. Gadekallu, Z. Han, and C. Su, "Blockchain-based reliable and efficient certificateless signature for IIoT devices," *IEEE Transactions on Industrial Informatics*, p. 1, 2021.

[7] G. Zyskind, O. Nathan, and A. S. Pentland, "Decentralizing privacy: using blockchain to protect personal data," in *2015 Proceedings of the 2015 IEEE Security and Privacy Workshops*, pp. 180–184, San Jose, CA, USA, May 2015.

[8] M. Amoretti, G. Brambilla, F. Medioli, and F. Zanichelli, "Blockchain-based proof of location," in *Proceedings of the 2018 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C)*, pp. 146–153, Lisbon, Portugal, June 2018.

[9] Y. Ji, J. Zhang, J. Ma, C. Yang, and X. Yao, "BMPLS: blockchain-based multi-level privacy-preserving location sharing scheme for telecare medical information systems," *Journal of Medical Systems*, vol. 42, no. 8, 2018.

[10] T.-F. Lee, H.-Z. Li, and Y.-P. Hsieh, "A blockchain-based medical data preservation scheme for telecare medical information systems," *International Journal of Information Security*, vol. 20, no. 4, pp. 589–601, 2021.

[11] H. Li, L. Zhu, M. Shen, F. Gao, X. Tao, and S. Liu, "Blockchain-based data preservation system for medical data," *Journal of Medical Systems*, vol. 42, no. 8, 2018.

[12] X. Yue, H. Wang, D. Jin, M. Li, and W. Jiang, "Healthcare data gateways: found healthcare intelligence on blockchain with novel privacy risk control," *Journal of Medical Systems*, vol. 40, no. 10, 218 pages, 2016.

[13] S. Cao, G. Zhang, P. Liu, X. Zhang, and F. Neri, "Cloud-assisted secure eHealth systems for tamper-proofing EHR via blockchain," *Information Sciences*, vol. 485, pp. 427–440, 2019.

[14] J. Wang, K. Han, A. Alexandridis et al., "A blockchain-based eHealthcare system interoperating with WBANs," *Future Generation Computer Systems*, vol. 110, pp. 675–685, 2020.

[15] H. Huang, P. Zhu, F. Xiao, X. Sun, and Q. Huang, "A blockchain-based scheme for privacy-preserving and secure sharing of medical data," *Computers & Security*, vol. 99, Article ID 102010, 2020.

[16] Y. Zhuang, L. R. Sheets, Y.-W. Chen, Z.-Y. Shae, J. J. P. Tsai, and C.-R. Shyu, "A patient-centric health information exchange framework using blockchain technology," *IEEE Journal of Biomedical and Health Informatics*, vol. 24, no. 8, pp. 2169–2176, 2020.

[17] S. Shamshad, K. Minahil, K. Mahmood, S. Kumari, and C.-M. Chen, "A secure blockchain-based e-health records storage and sharing scheme," *Journal of Information Security and Applications*, vol. 55, Article ID 102590, 2020.

[18] H. Huang, X. Sun, F. Xiao, P. Zhu, and W. Wang, "Blockchain-based eHealth system for auditable EHRs manipulation in cloud environments," *Journal of Parallel and Distributed Computing*, vol. 148, pp. 46–57, 2021.

[19] H. Zhu, Y. Guo, and L. Zhang, "An improved convolution Merkle tree-based blockchain electronic medical record secure storage scheme," *Journal of Information Security and Applications*, vol. 61, Article ID 102952, 2021.

[20] M. A. Uddin, A. Stranieri, I. Gondal, and V. Balasubramanian, "Blockchain leveraged decentralized IoT eHealth framework," *Internet of Things*, vol. 9, Article ID 100159, 2020.

[21] S. Tanwar, K. Parekh, and R. Evans, "Blockchain-based electronic healthcare record system for healthcare 4.0 applications," *Journal of Information Security and Applications*, vol. 50, Article ID 102407, 2020.

[22] K. Miyachi and T. K. Mackey, "hOCBS: a privacy-preserving blockchain framework for healthcare data leveraging an on-chain and off-chain system design," *Information Processing & Management*, vol. 58, no. 3, Article ID 102535, 2021.

[23] Z. Chen, W. Xu, B. Wang, and H. Yu, "A blockchain-based preserving and sharing system for medical data privacy," *Future Generation Computer Systems*, vol. 124, pp. 338–350, 2021.

[24] K. Mohammad Hossein, M. E. Esmaeili, T. Dargahi, A. Khonsari, and M. Conti, "BCHealth: a novel blockchain-based privacy-preserving architecture for IoT healthcare applications," *Computer Communications*, vol. 180, pp. 31–47, 2021.

[25] A. Boldyreva, N. Chenette, Y. Lee, and A. O'Neill, "Order-preserving symmetric encryption," in *Advances in Cryptology - EUROCRYPT 2009*, A. Joux, Ed., vol. 5479, pp. 224–241, Springer, Cham, 2009.

[26] R. C. Merkle, "A certified digital signature," in *Proceedings of the 9th Annual International Cryptology Conference on Advances in Cryptology*, pp. 218–238, Santa Barbara, CA, USA, July 1989.

[27] E. Androulaki, A. Barger, C. Cachin et al., "Hyperledger fabric," *Proceedings of the Thirteenth EuroSys Conference*, vol. 30, pp. 1–15, 2018.

[28] Z. Yang, K. Yang, L. Lei, K. Zheng, and V. C. M. Leung, "Blockchain-based decentralized trust management in vehicular networks," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1495–1505, 2019.