Hindawi
Security and Communication Networks
Volume 2021, Article ID 5781354, 12 pages
https://doi.org/10.1155/2021/5781354

Research Article

A Privacy-Preserving Data Transmission Scheme Based on Oblivious Transfer and Blockchain Technology in the Smart Healthcare

Huijie Yang,[1] **Jian Shen**,[1,2] **Junqing Lu**,[1] **Tianqi Zhou**,[1] **Xueya Xia**,[1] **and Sai Ji**[1,3]

[1]*School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing, Jiangsu, China*
[2]*Cyberspace Security Research Center, Peng Cheng Laboratory, Shenzhen, China*
[3]*Suqian University, Suqian, Jiangsu, China*

Correspondence should be addressed to Jian Shen; s_shenjian@126.com

Received 16 June 2021; Accepted 17 August 2021; Published 3 September 2021

Academic Editor: Debiao He

With the development of the Internet of Things and the demand for telemedicine, the smart healthcare system has attracted much attention in recent years. As a platform for medical data interaction, the smart healthcare system is demanded to ensure the privacy of both the receiver and the sender, as well as the security of data transmission. In this paper, we propose a privacy-preserving data transmission scheme where both secure ciphertext conversion and malicious users identification are supported. In particular, the $(\mathrm{OT})_m^n$ protocol is introduced to guarantee the two-way privacy of communication parties. Meanwhile, we adopt proxy reencryption algorithm to support secure ciphertext conversion so as to ensure the confidentiality of data in many-to-many communication pattern. In addition, by taking advantage of the concept of blockchain technology, a novel $(\mathrm{OT})_m^n$ protocol is proposed to prevent data from being tampered with and effectively identify malicious users. Theoretical and experimental analyses indicate that the proposed scheme is practical for smart healthcare with high security and efficiency.

1. Introduction

With the extension of average life expectancy and people's increasing demand for health, the demand for smart healthcare systems such as telemedicine and e-health system is more and more urgent [1–3]. The smart healthcare system is an IoT health system composed of cloud computing, smart wearable devices, an expert system based on artificial intelligence, and so on [4–6]. The deep learning technology and data mining technology also promote the development of smart healthcare system [7, 8], which is convenient for doctors to quickly diagnose diseases and formulate medical plans and to ensure that everyone can get adequate medical resources [9]. In addition, some scholars introduce blockchain technology into the smart healthcare environment [10–12]. They utilize the characteristics of blockchain such as decentralization and antitampering to design the smart healthcare schemes. Those schemes can realize data sharing and ensure the confidentiality and correctness of the data. During the research, scholars discovered that there were two security challenges in the process of medical data transmission [13, 14]: The first is how to ensure the confidentiality of medical data during the interaction; that is, malicious users cannot obtain or tamper with the data. The second is how to realize the two-way privacy protection between the server and the client side. Therefore, we need to discuss and solve the above two security challenges in this paper.

Consider the following situation: the patient who suffers from many diseases goes to different specialist hospitals for treatment, and so many medical records are stored by different servers of hospitals that are not connected in the same network. That is, it is difficult for doctors to obtain the data across the different networks. To settle the mentioned problem, we suppose that all the data are stored in the same server. However, all data in this server are returned to the doctor when he employs the oblivious transfer (OT)

protocol to request the data, which leads to the high communication overhead. Thus, we assume that all data are stored in the distributed server, and the users, hospitals, and also servers are all in the smart healthcare system, where the patients' records can be accessed by the departments' doctors from the different hospitals. In fact, the stored data are susceptible to collusion or tampering because of the semitrusted server. The confidentiality of stored data cannot be ensured when users employ those data. Additionally, a user employs the amount of data to request it from the server, which can try to understand the corresponding relationship between the sequence number and the stored data. Some researchers utilize the oblivious random access memory (ORAM) protocol [15] to hide that relationship [16, 17]. Meanwhile, it is only applied to some simple systems due to its complex ORAM structure and the great increase in cost overheads. Thus, how to ensure that the server makes users know the data without knowing the serial number is one of the main contents of our scheme. Moreover, the public keys of the distributed servers are different for users owing to those servers which belonged to different hospitals. In view of the above, the data in such servers can be comprehended by users, which exposes the privacy of stored data. Then, some data are attacked by malicious users to focus on, in accordance with that private information. What is more, the authorities of the user can be faked or tampered with by the revoked or malicious users who can collude the data.

Motivation of This Paper. As is mentioned above, the existing data transmission schemes are not suitable for the smart healthcare environment. Therefore, our goals are to protect user privacy and guarantee the security of data transmission based on OT and blockchain technology under the smart healthcare environment. To accomplish this goal, the three following crucial issues should be considered for us. First, the confidentiality of data should be assured during the process of data transmission. The medical data are related to the life safety of patients; once they are tampered with or faked, this will endanger the lives of patients and put the hospital in financial compensation. Second, while guaranteeing that a piece of accessed data is not known by the server, the other data in it cannot be learned by the user. In addition, the stored data in such servers cannot be figured out. In case of reveal, the privacy of stored data may be leaked out. Finally, the revoked or malicious users who try to collude should be discerned by the group manager. They will go beyond their authority to access data or modify medical data, leading to medical accidents in the hospital.

1.1. Main Contributions. We design a privacy-preserving scheme for data transmission based on oblivious transfer and blockchain technology in the smart healthcare environment which is to resolve the above issues. The main contributions are as follows:

(1) A novel $(\mathrm{OT})_m^n$ protocol supporting two-way privacy-preserving and distributed servers is proposed. Suppose that u_1 data is stored in multiple servers. Once a doctor requires u_1 data, he needs to employ many private keys of servers to decrypt those ciphertexts. In that way, the privacy of servers where the data is stored will be exposed. By applying this novel $(\mathrm{OT})_m^n$ protocol, a doctor can decrypt all the ciphertexts with only his key. In other words, this protocol not only queries data quickly but also protects the privacy of servers and doctors. In addition, the proposed $(\mathrm{OT})_m^n$ protocol can efficiently support the access control of users and many-to-many data transmission pattern.

(2) A secure data transmission scheme supporting collusion resistance and to prevent data from being tampered with is proposed. Our scheme is a data secure transmission protocol based on blockchain technology and OT technology. We utilize the characteristics of blockchain structure to store the user's identity in blocks and then form three lists, namely, patient identity list, doctor identity list, and revocation user list. Therefore, our protocol can effectively verify revocation or malicious users and resist their collusion attacks. Meanwhile, in terms of the hash value in blockchain, malicious users cannot modify the data.

(3) Data confidentiality is guaranteed and the computation of our scheme is effectively reduced. We analyze and prove the security of the proposed scheme. We provide a performance comparison between $(\mathrm{OT})_m^n$ protocol and other $(\mathrm{OT})_n^k$ protocols through a theoretical performance analysis and an experimental analysis.

1.2. Related Work. Oblivious transfer (OT) has gradually become an important research direction in the field of multiparty computation (MPC). At present, according to the total amount of data and the number of choices, the research of OT protocol is mainly divided into four categories: classical oblivious transfer protocol [18], 1-out-of-2 oblivious transfer $(\mathrm{OT})_2^1$ protocol [19], 1-out-of-n oblivious transfer $(\mathrm{OT})_n^1$ protocol [20], and k-out-of-n oblivious transfer $(\mathrm{OT})_n^k$ protocol [21].

$(\mathrm{OT})_n^1$ protocol was proposed by Brassard et al. [20] firstly in 1986; they invoked $(\mathrm{OT})_2^1$ protocol n times to implement $(\mathrm{OT})_n^1$ protocol. On the basis of the above, Gertner et al. [22] firstly achieved a distributed version of $(\mathrm{OT})_n^1$ protocol with information-theoretic security and sublinear communication complexity. In 2001, Naor et al. [23] described a novel $(\mathrm{OT})_n^1$ protocol, which improved the efficiency of multiple invocations of OT applications. In 2004, Tzeng [24] designed a secure and efficient $(\mathrm{OT})_n^1$ protocol under the assumption of the decisional Diffie-Hellman problem. After that, based on the above, an adaptive k-out-of-n oblivious transfer scheme was proposed by Chu and Tzeng [25], which allowed the receiver to choose the messages one by one adaptively. In 2015, the simplest and most efficient protocol for $(\mathrm{OT})_n^1$ protocol was presented by Chou et al. [26] and it could resist some active attacks. Then, in 2007, Hauck et al. [27] proposed an $(\mathrm{OT})_n^1$

protocol under the CDH assumption, which was built on ideas from the CO protocol. In 2020, Wang et al. [28] presented an $(\mathrm{OT})_n^1$ protocol and the Private Set Intersection (PSI) protocol to protect user privacy in the case of VANET feature matching.

$(\mathrm{OT})_n^k$ protocol was proposed by Bellare et al. [21] firstly in 1989, where a receiver could select and receive multiple ciphertexts at one time. Naor et al. [29] described a novel construction for $(\mathrm{OT})_n^k$ protocol which is more efficient than k repetitions of $(\mathrm{OT})_n^1$ protocol. Then, the classical and universal $(\mathrm{OT})_n^k$ protocol was designed by Naor et al. [23]. In 2005, an $(\mathrm{OT})_n^k$ protocol with adaptive queries was proposed by Naro et al. [30], and it was considerably more efficient than k repetitions of $(\mathrm{OT})_n^1$ protocol. After that, Chu et al. [31] proposed several two-round $(\mathrm{OT})_n^k$ protocols under the decisional Diffie-Hellman problem, in which a receiver sent $O(k)$ data to a sender and he returned $O(n)$ data. In 2010, a secure and low-bandwidth-consumption $(\mathrm{OT})_n^k$ scheme based on bilinear pairings was proposed by Chen et al. [32]. A novel $(\mathrm{OT})_n^k$ protocol for private information retrieval which was more suitable for smart cities was presented by Lou et al. [33]. In 2018, Lai el al. [34] proposed an $(\mathrm{OT})_n^k$ scheme with the least communication cost, which preserved a sender's security and the privacy of a receiver's choice.

What is more, some researchers have integrated OT technology into blockchain scheme in order to solve the problem of easy exposure of private data in the blockchain. In 2017, Hsiao et al. [35] combined the advantages and properties of blockchain and secret sharing scheme, Paillier's homomorphic encryption, and oblivious transfer to construct a decentralized e-voting system. This scheme could protect the anonymity of voter's identity, the privacy of data transmission, and verifiability of ballots during the billing phase. In 2019, Tso et al. [36] proposed the decentralized electronic voting and bidding systems based on a blockchain and smart contract, which uses cryptographic techniques such as oblivious transfer and homomorphic encryptions to improve privacy protection. Then, in 2021, Li et al. [37] presented a fair scheme for big data exchanging that allows buyers and sellers to autonomously and fairly complete transactions, without involving any third-party middle person. This scheme employed OT technology to preserve the privacy of transactions.

1.3. Organization. The structure of the paper is organized as follows. Some preliminaries in cryptographic are presented in Section 2. The system model, design goals, and threat model are described in Section 3. The proposed scheme is introduced in detail in Section 4. The security and performance analyses are provided in Sections 5 and 6, respectively. Section 7 concludes this paper and our work.

2. Preliminaries

2.1. Proxy Reencryption Technology. We adopt the key-private proxy reencryption scheme which was proposed by Ateniese et al. [38]. This algorithm applies proxy reencryption technology to achieve ciphertext conversion, which converts a ciphertext m_a of u_a to a ciphertext m_b of u_b. The specific design of this scheme is as follows.

(i) Step 1. $\mathrm{Setup}(1^k) \longrightarrow \mathrm{par}$: This is the initialization phase for generating parameters. Input security parameter 1^k and then the public parameters par are output by this algorithm.

(ii) Step 2. $\mathrm{KeyGen}(\mathrm{par}) \longrightarrow (pk, sk)$: This algorithm is applied to generate the public-private key pair for users. Public parameter par is input, and then the key pair (pk, sk) is produced for users.

(iii) Step 3. $\mathrm{Enc}(\mathrm{par}, pk_a, m) \longrightarrow m_a$: This algorithm is employed to encrypt the message via a public key pk_a from u_a. pk_a and message m are input, and then an original ciphertext m_a is produced by this algorithm.

(iv) Step 4. $\mathrm{Re-KeyGen}(\mathrm{par}, sk_a, pk_b) \longrightarrow rk_{a\longrightarrow b}$: This algorithm generates the conversion key, which realizes the transformation from m_a to m_b. A private key sk_a of u_a and a public key pk_b of u_b $(a \neq b)$ are provided, and the conversion key $rk_{a\longrightarrow b}$ is output. This phase is crucial for reencryption data.

(v) Step 5. $\mathrm{Re-Enc}(\mathrm{par}, rk_{a\longrightarrow b}, m_a) \longrightarrow m_b$: To gain the transfer message m_b, this algorithm utilizes the conversion key $rk_{a\longrightarrow b}$ to reencrypt a message m_a. Input A reencryption key and an original ciphertext are input, and then a reencryption ciphertext m_b from a to b is output.

(vi) Step 6. $\mathrm{Dec}(\mathrm{par}, sk_a, m_a) \longrightarrow m$: Finally, a private key sk_a and a ciphertext m_a are provided; this algorithm can compute a plaintext m.

2.2. Oblivious Transfer Protocols. The concept of OT was first proposed by Rabin [18] in 1981. In Rabin's protocol, the sender only wanted the receiver to get the message he chooses, and the receiver did not want the sender to know about other messages, which guaranteed the privacy of both parties. Then, the 1-out-of-2 data transmission protocol under the semihonest model through three public key cryptography operations was implemented by Naro and Pinkas [23]. The steps of (OT_1^2) protocol are as follows:

(i) Setup: The system generates two prime orders q and p, where $q|p-1$ holds. $\mathbb{G}_p$ is a p-order subgroup of $\mathbb{Z}_p^*$; and the system sets g as the generator of $\mathbb{Z}_p^*$.

(ii) Input: The sender inputs (X_0, X_1), and receiver inputs r.

(iii) Output: The receiver outputs X_r.

(a) Step 1. The sender generates a random number C and a, computes g^a and C^a, and broadcasts C.

(b) Step 2. The receiver generates a random number $k\,(1 \leq k \leq q)$; and two public keys pk_r and pk_{1-r} are generated, where $pk_r = g^k$ and $pk_{1-r} = C/g^k$ hold. Then, the number pk_0 is sent to the sender.

(c) Step 3. The sender calculates (pk_0^a) and $(pk_1^a) = C^a/pk_0^a$. At the same time, he encrypts the

data (X_0, X_1), respectively. The equations are as follows:

$$E_0 = (g^a, \text{hash}((pk_0)^a) \oplus x_0),$$
$$E_1 = (g^a, \text{hash}((pk_1)^a) \oplus x_1). \quad (1)$$

(d) Step 4. The receiver computes $\text{hash}(pk_r^a) = \text{hash}((g^a)^k)$ and x_r; that is,

$$E_r = (\text{hash}((pk_r)^a, r) \oplus x_r) \oplus \text{hash}(pk_r^a, r). \quad (2)$$

The concept of (OT_k^n) protocol is presented as follows. The sender encrypts the n secret messages $M_0, M_1, \ldots, M_{n-1}$ and sends them to the receiver; and the receiver can only recover k of them:

$$M_{\alpha_1}, M_{\alpha_1}, \ldots, M_{\alpha_k}, \quad (3)$$

where $\alpha_1, \ldots, \alpha_k \in \mathbb{Z}_p^*$ holds. However, the receiver cannot determine which $M_{\alpha_1}, M_{\alpha_1}, \ldots, M_{\alpha_k}$ from $M_0, M_1, \ldots, M_{n-1}$ are required.

2.3. Blockchain Technology. Blockchain is a kind of ledger technology that is jointly maintained by multiple parties, can achieve consistent data storage, is difficult to tamper with, and prevents denial [39, 40]. It has also become a distributed ledger technology. The blockchain is classified into the permissioned blockchain and the unlicensed blockchain according to whether the system has the node access mechanism. The fabric is employed in our paper, which belongs to the consortium Blockchain and is also the first distributed system of blockchain with an access mechanism [41]. Fabric is a modular, extensible, general-purpose blockchain with an access mechanism that supports the execution of distributed applications written in standard programming languages. The key components of fabric are as follows [42].

(i) Peers: There are four kinds of peers in Fabric.

(a) Committing peer: Each peer in the channel is the committing peer. It receives the generated transaction block, obtains the block structure, and verifies the legitimacy of the block structure.
(b) Endorsing peer: The client application must use its smart contract to complete the verification of the transaction, simulate the operation of the transaction, and generate a transaction response containing a digital signature.
(c) Leader peer: When the channel has multiple peers, the leader peer is responsible for distributing transactions from the ordering peer to other committing peers.
(d) Anchor peer: It helps to communicate with peers in other organizations.

(ii) Channel: The channel includes many authorized users, and each user can belong to different channels.

(iii) Consensus mechanism: It is defined as the comprehensive verification of the correctness of the blockchain transaction. It includes the SOLO, Kafka, PBFT, and SBFT.

3. Problem Statement

3.1. System Model. Our proposed scheme can be utilized to securely transfer data and also realize the privacy-preserving of the clients and servers. On the one hand, the private information of client side is protected. That is, a user has permission in virtue of the data's serial number to access data, yet he does not know which server the data is stored in. On the other hand, the private information of servers is protected. That is, a user only can obtain the requested data, and the other data are cannot be learned. This scheme is mainly designed in accordance with the actual situation of the smart healthcare environment. Both doctors and other healthcare workers look forward to acquiring treatment records about a patient in all hospitals as soon as possible. Moreover, the confidentiality of data can be ensured in our scheme, in which a user employs his private key to decrypt the stored data in servers rather than private keys of servers. In addition, this scheme also resists collusion attacks by revoked or malicious users. The system model contains three entities, doctors/ patients (client side), a proxy (blockchain), and servers. Figure 1 shows a system model of the proposed scheme.

A patient cures his diseases in different hospitals or in the same hospitals. In general, the data is stored in the nearest server, which is a server of the current hospital. This means that if a patient has seen a disease in different hospitals, multiple servers (different hospitals) store the patient's data. Our scheme implements a many-to-many model with users and servers. Firstly, doctors and patients register their identities with the blockchain. The blockchain generates a list of user identities and a list of revoked users so that it can verify their identities. Secondly, a doctor uses the private key of user to encrypt the medical records and then uploads them to a server of his hospital. When a patient goes to a hospital to treat his heart disease, his doctor of this department can gain his past medical records in servers. Thirdly, a doctor sends a request to blockchain for obtaining a patient's records. The blockchain verifies and checks his identity. If yes, the ciphertext encrypted with the patient's key needs to be converted into ciphertext which can be decrypted by doctors. The patient and blockchain run the encryption phase to complete the transformation of ciphertext. Fourthly, the blockchain transmits a request which includes some serial numbers of data to servers. Only servers that store the corresponding data respond to that request. Finally, the OT protocol is implemented between the doctor and the server to transmit and decrypt the request data.

3.2. Threat Model. In this section, the security goals and the security models for OT_m^n are provided.

Definition 1. A secure and privacy-preserving $(\text{OT})_m^n$ protocol should satisfy the following requirements:

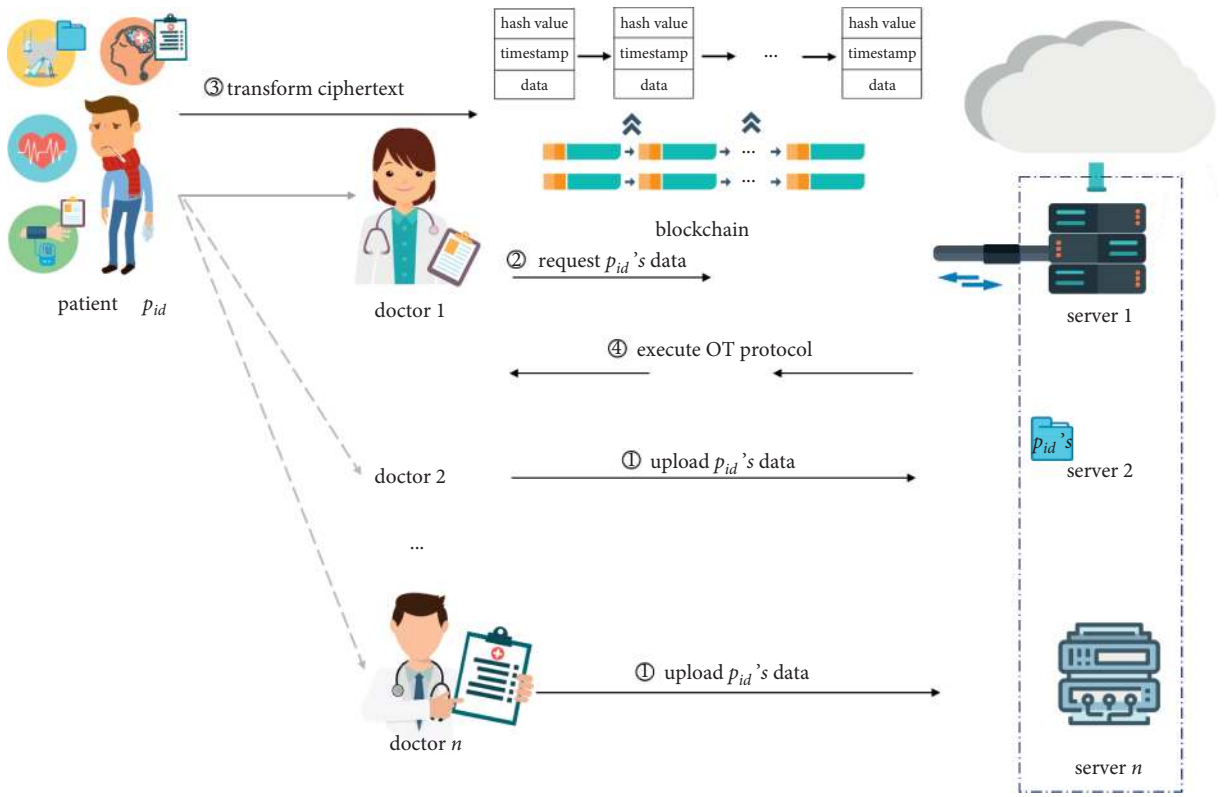

Figure 1: The system model of our scheme.

(1) The $(\mathrm{OT})_m^n$ protocol should protect the privacy of servers; namely, the users cannot obtain data from the server other than what they requested.

(2) The $(\mathrm{OT})_m^n$ protocol should protect the privacy of users; namely, the servers cannot figure out what data the users access.

The security model for server privacy of the OT_m^n protocol is described as follows. In this model, adversary $\mathcal{A}$ plays the role of users and challenger $\mathcal{C}$ plays the role of servers (the servers are trusted). The advantage of $\mathcal{A}$ to break the server privacy is defined as follows:

$$\mathrm{Adv}_{\mathcal{A}} = Pr[\mathcal{A} \quad \text{wins}]. \tag{4}$$

(i) Setup: The system generates system parameters and sends the private keys to the blockchain. Then the blockchain generates several necessary parameters for servers. Adversary $\mathcal{A}$ chooses j data that it can access and chooses the corresponding a_j from $\mathbb{Z}_p^*$. Adversary $\mathcal{A}$ outputs its target $t\,(t \notin \{j\})$. Then, the blockchain sends corresponding D_j to adversary $\mathcal{A}$ and the servers send all ciphertexts c_i.

(ii) Hash Query: Adversary $\mathcal{A}$ can query the hash value via this oracle. It takes as input information and outputs hash value.

(iii) Decrypt Query: Adversary $\mathcal{A}$ can query plaintext m_i of ciphertext c_i but not ciphertext c_t.

(iv) Decrypt: Adversary $\mathcal{A}$ outputs plaintext m_t. If m_t is right, adversary $\mathcal{A}$ breaks the server privacy of the OT_m^n protocol.

The security model for user privacy of the OT_m^n protocol is described as follows. In this model, adversary $\mathcal{A}$ plays the role of servers and the challenger plays the role of users (the users are trusted). The advantage of $\mathcal{A}$ to break the server privacy is defined as follows:

$$\mathrm{Adv}_{\mathcal{A}} = Pr[\mathcal{A} \quad \text{wins}] - \frac{1}{2}. \tag{5}$$

(i) Setup: The system generates system parameters and sends the private keys to the blockchain. Then the blockchain generates several necessary parameters for adversary $\mathcal{A}$. The users choose j data that they can access and choose corresponding a_j from $\mathbb{Z}_p^*$. Then, the blockchain sends corresponding D_j to the users and the servers sends all ciphertexts c_i. Adversary $\mathcal{A}$ outputs its target $t_0, t_1\,(t_0, t_1 \in \{j\})$.

(ii) Hash Query: Adversary $\mathcal{A}$ can query the hash value via this oracle. It takes as input information and outputs hash value.

(iii) Challenge: The user selects $b \xleftarrow{R} \{0, 1\}$ and outputs A_{t_b} and D_{t_b}.

(iv) Guess: Adversary $\mathcal{A}$ outputs b^*. If $b = b^*$, $\mathcal{A}$ wins.

4. The Proposed Scheme

The proposed scheme is presented in detail in this section. Our scheme can be divided into four parts, in which the initialization phase is introduced in Section 4.1, the user registration phase is described in Section 4.2, the encryption phase is stated in Section 4.3, and the data access phase and $(\mathrm{OT})_m^n$ protocol phase among three roles are illustrated in Sections 4.4 and 4.5, respectively.

In the smart healthcare system, in the face of complex diseases, the attending doctor will conduct multidepartment consultations or cross-hospital consultations, which are more common. In addition, a patient can treat diseases in different hospitals, a hospital has its own servers, and the users who have access permission can request the data of servers in the smart healthcare system. In our scheme, to protect the two-way privacy of the server and user, the data are allocated to the nearest server randomly, which obeys the principle of proximity; that is, the user with permission can store the data in which the server is near. We show the main idea of the system by giving an example. A patient suffers from high blood pressure, heart disease, and toothache. When he goes to the dental clinic, the doctor not only needs to diagnose his teeth but also prescribes medicine or prepares for surgery based on his other medical history. At this moment, an attending doctor verifies his permission to request all the data about that patient. The requests can be sent to the determined server which has stored the data of that patient. Then, to protect the privacy information, only the determined server delivers all its data to the requester by using the designed $(\mathrm{OT})_m^n$ protocol. This protocol guarantees that the server does not have idea about the accessed data, and the user cannot obtain the extra data and figure out the source of data. For instance, if the sequence number 5 is requested from the user, he sends the requirement to all the servers in the smart healthcare system. Only sever S_y that has the data about that patient responds to the request. More comprehensively, assume that the sequence number d is accessed; the user sends the requests to servers $S_1, S_2, \ldots, S_y$ (the needed data are in servers S_a and S_b). At last, servers S_a and S_b have the opportunity to communicate with the user. In the meantime, blockchain technology is merged into our scheme, which maintains the attributes list and stops user attributes from being tampered with. Correspondence between symbols and definitions is shown in Table 1.

4.1. Initialization Phase. We hypothesize that there are y distributed servers, and the users with permission to manipulate data (e.g., the doctor, nurse, and healthcare worker) have n data, and each piece of data has the same bits.

Input the security parameter s, and then the system randomly selects the number $\{k_1, k_2, \ldots, k_y, k_1', k_2', \ldots, k_y'\} \in \mathbb{Z}_p^*$, where the formal $k_1 + k_1' = \kappa$ $(\kappa \in \mathbb{Z}_p^*)$ holds, server S_i possesses k_i, and $\{\kappa, k_1', \ldots k_y'\}$ is stored in the blockchain. Set $\mathbb{G} = <g>, \mathbb{G}_T$ as the multiplicative cyclic groups, with bilinear mapping $e\colon \mathbb{G} \times \mathbb{G} \longrightarrow \mathbb{G}_T$, randomly choose generator $g_1 \in \mathbb{G}$, and compute $\alpha = e(g, g_1)$. Set the hash functions $h_1\colon \{0,1\}^n \longrightarrow \mathbb{G}$ and $h_2\colon \{0,1\}^n \longrightarrow \{0,1\}^l$, where l is the fixed value.

Table 1: Correspondence between symbols and definitions.

Symbols	Definition
n	The serial number of data
$p_{i\,d}, d_{i\,d}$	The identity of the patients and doctors
δ_m	Data with serial number m requested by the user
k_y	The employed key of server in OT protocol
k_y'	The stored server subkey of proxy in OT protocol
w_n	The hash value of the serial number
(sk_p, pk_p)	The public-private key pairs of patients
(sk_d, pk_d)	The public-private key pairs of doctors
$rk_{p \longrightarrow d}$	The transform key from patient to doctor
β	The attribute sets of all the users
$u_n = \{a_n, b_n, \ldots, z_n\}$	The set of each attribute of the user
tag_{m_i}	The tag for each piece of data

We initialize the device of proxy based on the blockchain. The security parameter s is input; the blockchain computes the formulas $f = g^\kappa$ and $w_i = h_1(i)$, where i represents the label of the patient's medical data. Then, the blockchain computes $R_{1i} = w_i^{k_1}, \ldots, R_{yi} = w_i^{k_y}$ and the following finite sets are satisfied, where $0 < i < n + 1$ and $i \in \mathbb{Z}_p^*$ hold. f is sent to the user. Then $R_1, R_2, \ldots, R_y$ are sent to servers $S_1, S_2, \ldots, S_y$ orderly.

$$\begin{aligned} S_1\colon R_1 &= \left\{w_1^{k_1'}, w_2^{k_1'}, \ldots, w_n^{k_1'}\right\}, \\ &\vdots \\ S_y\colon R_y &= \left\{w_1^{k_y'}, w_2^{k_y'}, \ldots, w_n^{k_y'}\right\}. \end{aligned} \tag{6}$$

The symbols and the corresponding meanings are shown in Table 1.

4.2. User Registration Phase. We integrate blockchain technology into user registration phase to maintain the identity lists about users. The data is requested via proxy, while the blockchain inquires and verifies the user's identity in accordance with his tag. Only through the verification can the $(\mathrm{OT})_m^n$ protocol be executed to transmit the data. There are five functions of blockchain in our scheme; they are described briefly as follows:

(i) The committing peer generates and maintains blocks for users.

(ii) The identity of required users is verified.

(iii) The endorsing peer verifies the legality of updated identities; if the transaction is legal, the peer simulates to perform the smart contract. Then, it sends the updated lists to users.

(iv) The attributes are prevented from being faked through utilizing the structural characteristics of the block.

(v) The revoked and malicious users are distinguished to preclude them from colluding the data.

The user registration phase contains the four following steps to accomplish the registration:

(i) Step 1. The key generation center (KGC) chooses $x_{p,1}, x_{p,2} \xleftarrow{R} \mathbb{Z}_p^*$ as the patient's private key $sk_p = (x_{p,1}, x_{p,2})$ and computes the patient's public key $pk_p = (pk_{p,1} = \alpha^{x_{p,1}}, pk_{p,2} = g^{x_{p,2}})$. Then, it sends (sk_p, pk_p) to the patient through the secure channel.

(ii) Step 2. The key generation center (KGC) chooses $x_{d,1}, x_{d,2} \xleftarrow{R} \mathbb{Z}_p^*$ as the doctor's private key $sk_d = (x_{d,1}, x_{d,2})$ and computes the doctor's public key $pk_d = (pk_{d,1} = \alpha^{x_{d,1}}, pk_{d,2} = g^{x_{d,2}})$. Then, it sends (sk_d, pk_d) to the doctor through the secure channel.

(iii) Step 3. KGC inserts (pk_p, p_{id}) into the patient list and inserts (pk_d, d_{id}) into the doctor list. Then KGC sends the above lists to the blockchain via the smart contract.

4.3. Encryption Phase. After the doctor diagnoses the patient, he records the medical data on the computer. Subsequently, the encryption algorithm will be executed on the data.

(i) Step 1. $\mathrm{Verify}(d_{\mathrm{id}}, pk_d, sk_d)$: The doctor chooses $r_1, r_2 \xleftarrow{R} \mathbb{Z}_p^*$, computes $\mathrm{Committ}_1 = \alpha^{r_1}$ and $\mathrm{Committ}_2 = g^{r_2}$, $c = h_0(d_{\mathrm{id}} \| pk_d \| \mathrm{Committ}_1 \| \mathrm{Committ}_2)$, and computes responses $R_1 = r_1 + c \times x_{d,1}$ and $R_2 = r_2 + c \times x_{d,2}$. Then, the doctor sends $\Pi_{\mathrm{proof}} = (\mathrm{Committ}_1, \mathrm{Committ}_2, c, R_1, R_2)$ and (d_{id}, pk_d) to the blockchain. Once the blockchain obtains $(\Pi_{\mathrm{proof}}, d_{\mathrm{id}}, pk_d)$, it computes $c = h_0(d_{\mathrm{id}} \| pk_d \| \mathrm{Committ}_1 \| \mathrm{Committ}_2)$ and checks whether $\alpha^{R_1} \stackrel{?}{=} \mathrm{Committ}_1 \cdot pk_{d,1}^c$ and $g^{R_2} \stackrel{?}{=} \mathrm{Committ}_2 \cdot pk_{d,2}^c$. If the above equation holds, the blockchain checks whether the tuple (d_{id}, pk_d) belongs to the doctors list. If yes, the verification process is completed. Then, the doctor can upload data.

(ii) Step 2. $\mathrm{GenTag}(m_{kw}, p_{\mathrm{id}}, \mathrm{dep})$: In order to facilitate users to accurately access data, the data needs to be classified in the light of departments and patients. Generate a tag $\mathrm{tag}_{m_i} = \mathrm{GenTag}(m_{kw}, p_{\mathrm{id}}, \mathrm{dep})$ corresponding to the departments dep and patients p_{id}, and add it to m. The advantage of this is that the doctor can accurately acquire all the data about a certain department of this patient, and some invalid data are automatically removed, where the communication overhead of gained data by the OT protocol is reduced.

(iii) Step 3. $\mathrm{Encrypt}(m, pk_p, \mathrm{tag}_{m_i}) = ct_p$: The doctor encrypts data m of patient, which employs the patent's public key pk_p and encryption (Enc) algorithm from KP – PRE scheme [38]. Then, upload the ciphertext to the server.

4.4. Data Access Phase

(i) Step 1. When doctor d_{id} sees a patient p_{id}, he sends p_{id} and pk_d to the blockchain, along with the proof $\Pi_{d_{\mathrm{id}}} = (\mathrm{Committ}_1, \mathrm{Committ}_2, c, R_1, R_2)$, where $\mathrm{Committ}_1 = \alpha^{r_1}$, $\mathrm{Committ}_2 = g^{r_2}$, $r_1, r_2 \xleftarrow{R} \mathbb{Z}_p^*$, $c = h_0(d_{\mathrm{id}} \| pk_d \| \mathrm{Committ}_1 \| \mathrm{Committ}_2)$, $R_1 = r_1 + c \times x_{d,1}$, and $R_2 = r_2 + c \times x_{d,2}$.

(ii) Step 2. The blockchain computes $c = h_0(d_{\mathrm{id}} \| pk_d \| \mathrm{Committ}_1 \| \mathrm{Committ}_2)$ and checks whether $\alpha^{R_1} \stackrel{?}{=} \mathrm{Committ}_1 \cdot pk_{d,1}^c$ and $g^{R_2} \stackrel{?}{=} \mathrm{Committ}_2 \cdot pk_{d,2}^c$. If the above equation holds, the blockchain checks whether the tuple (d_{id}, pk_d) belongs to the doctors list. If yes, the blockchain executes the next step.

(iii) Step 3. The blockchain sends pk_d to patient p_{id}. The patient computes transform key $tk_{p \longrightarrow d}$ via $\mathrm{Re-KeyGen}(sk_p, pk_d)$ in the KP – PRE scheme and sends it to the blockchain.

(iv) Step 4. The blockchain sends $tk_{p \longrightarrow d}$ and p_{id} to all servers. The servers transform ciphertext m of patient p_{id}, namely, compute $ct_d \leftarrow \mathrm{KP-PRE.ReEnc}(tk_{p \longrightarrow d}, ct_p)$.

4.5. $(OT)_m^n$ Protocol Phase. Assume that doctor d_{id} treats p_{id}'s heart disease; he sends a data request which includes all the serial numbers of data from different hospitals about this patient's heart treatment records to servers. Suppose that the doctor needs to require the data with the serial number $s_n = \{\delta_1, \delta_2, \ldots, \delta_m\}$, and those data are stored in servers $S_1, \ldots, S_i, \ldots, S_n$. The steps of the $(\mathrm{OT})_m^n$ algorithm are shown in Figure 2.

(i) Step 1. The doctor (client side) transmits request s_n to the blockchain side. Then, server S_i responds and executes the following steps.

(ii) Step 2. The client side selects parameters $a_j \in \mathbb{Z}_p^*$ randomly and computes all parameters A_j of serial number of data; that is, $A_j = w_j g^{a_j}$, where $0 < j < m + 1$ $(j \in \mathbb{Z}_p^*)$ holds and parameters $w_1, \ldots, w_m$ are calculated previously. Then, (A_j, g^a) is sent to the blockchain.

(iii) Step 3. The blockchain side computes $D_j = (A_j)^{\kappa}$ and $\epsilon' = \{g^{ak_1'}, g^{ak_2'}, \ldots, g^{ak_y'}\}$ and sends ϵ_i' to server S_i. Then, it delivers D_j and f to the client side.

(iv) Step 4. The server side computes all the ciphertexts of its $c_i = ct_{d,i} \oplus h_2(w_i^{k_y} R_{yy_i'} \| g^{ak_y} \epsilon_y')$ and transmits c_i to the client side.

(v) Step 5. Only if $\delta_i \in s_n$ meets, the formulas $Y_j = (D_j / f^{a_j}) \| f^a$ and $ct_{d,j} = c_j \oplus h_2(Y_j)$ can be computed. After that, the ciphertexts of requested data s_n are calculated.

(vi) Step 6. $\mathrm{Decrypt}(ct_{d,j}, sk_d) = m_j$. The doctor employs his key sk_d to decrypt the above ciphertext.

Finally, the doctor obtains all the heart disease records about that patient.

5. Security Analysis

Theorem 1. *The proposed OT_m^n is server privacy, if the CDH assumption holds. Assume that any probability polynomial time adversary $\mathscr{A}$ can break the server privacy of the scheme; it can be utilized to solve CDH problem. The definition of CDH problem is that, given a tuple $(\mathbb{G}, g, g^\alpha, g^\beta)$ where $\alpha, \beta \xleftarrow{R} \mathbb{Z}_p^*$, the adversary should compute $g^{\alpha\beta}$.*

(i) Setup: In this phase, after challenger $\mathscr{C}$ obtains the CDH tuple $(\mathbb{G}, g, g^\alpha, g^\beta)$ where $\alpha, \beta \xleftarrow{R} \mathbb{Z}_p^*$, it sets $k = \alpha$ and $f = g^\alpha$. Then, it generates other system parameters to complete the setup of the whole scheme.

(ii) Query: In the hash query phase, adversary $\mathscr{A}$ queries the value of w_i. Challenger $\mathscr{C}$ sets

$$w_i = \begin{cases} g^{\beta_i}\left(\beta_i \xleftarrow{R} \mathbb{Z}_p^*\right), & i \neq t, \\ g^\beta, & i = t. \end{cases} \tag{7}$$

and outputs w_i to adversary $\mathscr{A}$. In the Decrypt query phase, adversary $\mathscr{A}$ cannot query the plaintext of ciphertext c_t.

(iii) Decrypt: Adversary $\mathscr{A}$ outputs plaintext m_t'. If $m_t' = m_t$, $\mathscr{A}$ wins this game.

Proof:. If m_t is right, it means that $\mathscr{A}$ can compute $Y_t = w_t^k \| f^a$, where $w_t^k = g^{\alpha\beta}$. Therefore, challenger $\mathscr{C}$ can utilize the adversary to output the solution $w_t^k = g^{\alpha\beta}$ of the CDH problem. In conclusion, if $\mathscr{A}$ wins, challenger $\mathscr{C}$ can output $g^{\alpha\beta}$ to solve the CDH problem. We can obtain that $\mathrm{Adv}_{\mathscr{A}}^{\text{server privacy}} \leq \mathrm{Adv}_{\mathscr{A}}^{\mathrm{CDH}}$. □

Theorem 2. *The proposed OT_m^n is user privacy, if the DDH assumption holds. Assume that any probability polynomial time adversary $\mathscr{A}$ can break the user privacy of the scheme; it can be utilized to solve DDH problem. The definition of DDH problem is that, given a tuple $(\mathbb{G}, g, g^\alpha, g^\beta, Z)$ where $\alpha, \beta \xleftarrow{R} \mathbb{Z}_p^*$, $Z = g^{\alpha\beta}$, or $Z \xleftarrow{R} \mathbb{G}$, the adversary should decide whether $Z = g^{\alpha\beta}$.*

(i) Setup: In this phase, after challenger $\mathscr{C}$ obtains the DDH tuple $(\mathbb{G}, g, g^\alpha, g^\beta, Z)$ where $\alpha, \beta \xleftarrow{R} \mathbb{Z}_p^*$ and $Z \xleftarrow{R} \mathbb{G}$ or $Z = g^{\alpha\beta}$, it sets $k = \beta$ and $f = g^\beta$. Then, it generates other system parameters to complete the setup of the whole scheme.

(ii) Query: In this phase, adversary $\mathscr{A}$ queries the value of w_i, and challenger $\mathscr{C}$ guesses the targets of adversary A, t_0 and t_1. Then, it sets

$$w_i = \begin{cases} g^{-a_{t_b}+\alpha} = g^{-a_{t_b}} \cdot g^\alpha, & \left(a_{t_b} \xleftarrow{R} \mathbb{Z}_p^*\right), \quad i = t_0 \vee i = t_1, \\ g^{a_i}, & \left(a_i \xleftarrow{R} \mathbb{Z}_p^*\right), \quad i \neq t_0 \wedge i \neq t_1, \end{cases} \tag{8}$$

and outputs w_i to adversary $\mathscr{A}$.

(iii) Challenge: Challenger $\mathscr{C}$ chooses $b \xleftarrow{R} \{0, 1\}$ and sets $A_{t_b} = g^\alpha$ and $D_{t_b} = Z$.

(iv) Guess: Adversary $\mathscr{A}$ outputs b^*. If $b^* = b$, challenger $\mathscr{C}$ outputs $p = 1$ (i.e., $Z = g^{\alpha\beta}$). Otherwise, $\mathscr{C}$ outputs $p = 0$ (i.e., $Z \xleftarrow{R} \mathbb{G}$).

Proof:. We assume that the probability of challenger $\mathscr{C}$ obtaining $Z = g^{\alpha\beta}$ is 1/2 and the probability of obtaining $Z \xleftarrow{R} \mathbb{G}$ is also 1/2. We assume that the advantage of $\mathscr{A}$ winning is ϵ and denote by E challenger $\mathscr{C}$ solving the DDH problem. It is easily deduced that $Pr[p = 1 | Z = g^{\alpha\beta}] = 1/2 + \epsilon$ and $Pr[p = 0 | Z \xleftarrow{R} \mathbb{G}] = 1/2Pr$. Therefore, we can get $\mathrm{Adv}_{D\,D\,H} = Pr[\mathscr{C} \text{ wins}] - 1/2 = Pr[p = 1 | Z = g^{\alpha\beta}] \times 1/2 + Pr[p = 0 | Z \xleftarrow{R} \mathbb{G}] \times 1/2 - 1/2 = (1/2 + \epsilon) \times 1/2 + 1/2 \times 1/2 - 1/2 = \epsilon/2$. So, $\epsilon = 2 \times \mathrm{Adv}_{D\,D\,H}$. Since the DDH assumption holds, it is difficult for $\mathscr{A}$ to decide whether $Z \stackrel{?}{=} g^{\alpha\beta}$. Therefore, adversary $\mathscr{A}$'s advantage to break the user privacy is negligible. □

Theorem 3. *Any revoked user cannot tamper with the data or his identity. Malicious or revoked users cannot get through the verification at our scheme. Meanwhile, if they attempt to request data, then they would be identified via the scheme. Therefore, the malicious or revoked users do not obtain the permission to request the data.*

Proof:. Firstly, the user needs to get through the identity authentication at the data access phase. Whether this formula $\mathrm{Committ}_{u_i} \cdot pk_{d,u_i}^c$, $g^R \stackrel{?}{=} \mathrm{Committ}_{u_r} \cdot pk_{d,u_r}^c$ is satisfied is checked. In general, a malicious user's commitment value cannot meet the calculation formula, and he would be judged as an invalid user by scheme. Secondly, even if a malicious user tries to modify his identity, the list of user identities is stored in the blockchain. That is, the modified identity cannot satisfy the formula $\mathrm{Hash}_{256}(d_{u_i}) = \mathrm{Hash}_{256}(d_{u_r})$. Then, that malicious user would be judged as an invalid user.

$(\mathrm{OT})_m^m$ ensures the two-way privacy of communication parties, and the proxy reencryption algorithm is secure. Therefore, the confidentiality of data can be protected by our proposed scheme. □

6. Performance

In this section, we first analyze the proposed scheme and provide a simplified comparison in Table 2. Then, an experimental evaluation of the proposed scheme is presented.

6.1. Performance Analysis. In our scheme, most of computation cost comes from the XOR operation, hash operation, Weil operation, power operation in $\mathbb{G}_1$, and power operation in $\mathbb{G}_T$, which are denoted as T_x, T_h, T_e, T_{E_1}, and T_{E_T}. In Table 2, n_d presents the number of doctors registered, n_p describes the number of patients registered, n_c is the number of patient ciphertexts, n_y illustrates the number of servers, and n_j states the number of j.

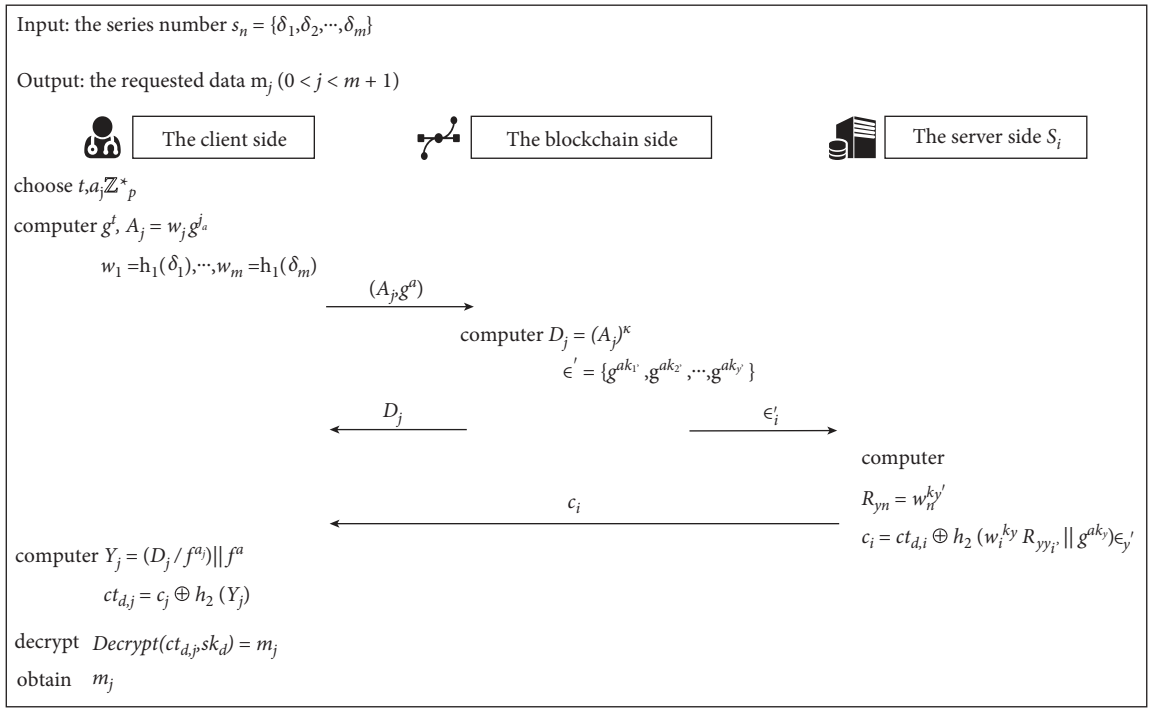

Figure 2: The steps of the $(\mathrm{OT})^n_m$ algorithm.

Table 2: Computational cost comparison.

Entities	Registration phase	Encryption phase	Data access phase	$(\mathrm{OT})^n_m$ phase
KGC	$(n_d + n_p)(T_{E_1} + T_{E_T})$	—	—	—
Client	—	$2T_{E_1} + 3T_{E_T} + T_h$	$4T_{E_1} + 3T_{E_T} + T_h + T_p$	$n_j(3T_{E_1} + T_{E_T} + T_h)$
Blockchain	—	$2T_{E_1} + 2T_{E_T} + T_h$	$2T_{E_1} + 2T_{E_T} + T_h$	$T_{E_1}(n_j + n_y)$
Servers	—	—	$n_c(2T_p + 2T_{E_T})$	$n_c(2T_{E_1} + T_h)$

T_x: XOR operation; T_e: Weil operation; T_h: hash operation; T_{E_1}: power operation in $\mathbb{G}_1$; and T_{E_T}: power operation in $\mathbb{G}_T$. n_d: the number of doctors registered; n_p: the number of patients registered; n_c: the number of patient ciphertexts; n_y: the number of servers; n_j: the number of j.

In registration phase, KGC generates the public-private key pairs for doctors and patients, and it costs computation overhead $(n_d + n_p)(T_{E_1} + T_{E_T})$. In encryption phase, blockchain verifies and checks the identities of client via lists to discern malicious or revoked users, which costs $4T_{E_1} + 2T_{E_T} + T_h$ computation overhead. Also, this phase is applied to encrypt the plaintext by using private key of patients, which defends the data confidentiality and costs $2T_{E_1} + T_{E_T}$ computation overhead. In data access phase, the ciphertext encrypted with the user's key should be converted into the ciphertext encrypted with the doctor's key, which costs $3T_{E_1} + T_{E_T} + T_p$ computation overhead. Moreover, this phase is also not involved in the general OT protocol, mainly to hide the access path of the server. In the $(\mathrm{OT})^n_m$ phase, it realizes the privacy-preserving of clients and servers.

6.2. Performance Evaluation. We simulate our proposed scheme employing the C language with PBC library (pbc-0.5.14) and GMP library (GMP-6.1.2) to evaluate the $(\mathrm{OT})^n_m$ protocol. All simulations are implemented on a desktop computer with the following features: (1) CPU: Intel(R) Core(TM) i5-9500 CPU @ 3.00 GHz 3.00 GHz; (2) random access memory: 8.0 GB; (3) OS: Ubuntu 14.04 over VMware workstation full 12.5.2; (4) system type: 64-bit.

We provide the computation comparison between doctors and patients in the $(\mathrm{OT})^n_m$ protocol in Figure 3. The X-axis describes the number of j requested by doctors. The Y-axis represents the time cost to perform the $(\mathrm{OT})^n_m$ protocol in doctor side and patient side. As shown in Figure 3, the time cost of doctors is higher than that of patients. The patient only needs to assist the blockchain to complete the transformation of the ciphertext. However, doctors need to participate in all the $(\mathrm{OT})^n_m$ protocols and calculate the transmission ciphertext of j data. Meanwhile, if the length of the ciphertext is fixed, the cost of transforming the ciphertext is roughly the same. Therefore, the patient's expenditure at this phase is approximately straight.

We provide the computation comparison of client side, blockchain, and servers in Figure 4. In order to make the comparison more obvious, the three entities are put in Figures 4 and 5. The computational overhead of the client side and blockchain is described in Figure 4; the X-axis represents the number of j and assumes that the number of

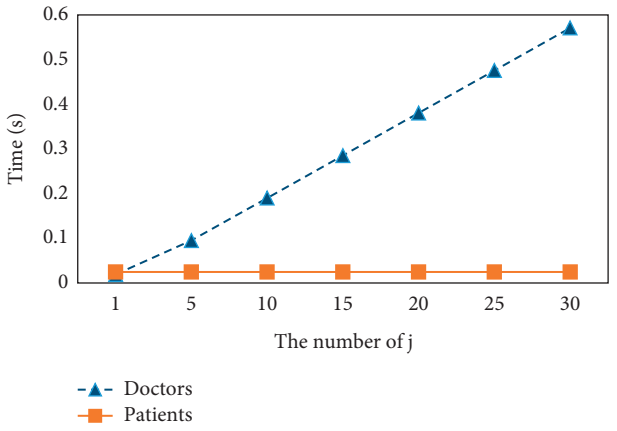

Figure 3: The computation comparison between doctors and patients in $(OT)_m^n$.

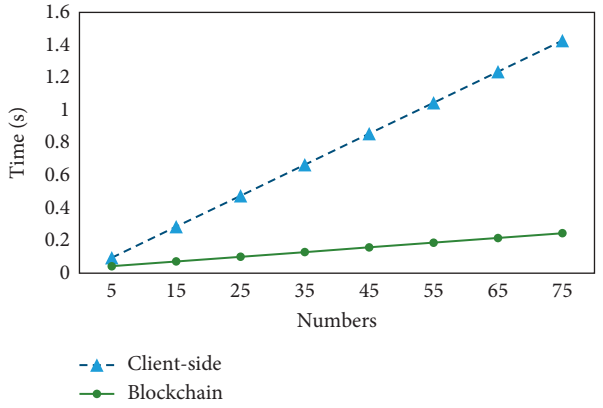

Figure 4: The computation comparison of the client side and blockchain in $(OT)_m^n$.

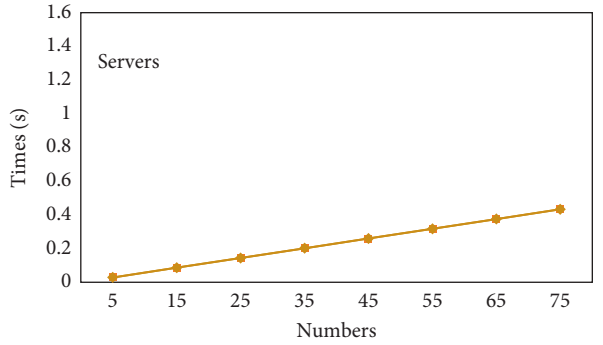

Figure 5: The computation comparison of the servers in $(OT)_m^n$.

servers is 10. For the server side, the X-axis represents the number of patient ciphertexts in Figure 5. As shown in the figures, we find that the overhead of the client is much higher than that of other entities. The proposed $(OT)_m^n$ protocol is an interactive protocol, which requires interaction between client side and servers to complete data transmission. At the meantime, this protocol uses the many-to-many data transmission pattern.

7. Conclusion

In this paper, a privacy-preserving data transmission scheme based on the oblivious transfer and blockchain technology in the smart healthcare system is proposed. Based on the proxy reencryption technology, the proposed $(OT)_m^n$ protocol can implement the ciphertext conversion to ensure the privacy of servers. Meantime, the two-way privacy between the client side and servers is guaranteed via the proposed $(OT)_m^n$ protocol, which also ensures the security and efficiency of data transmission. By taking advantage of blockchain technology, the proposed scheme can prevent data from being tampered with and effectively identify malicious users. After analyzing the protocol security, the confidentiality of data and security of our scheme are proved. Finally, the results of performance evaluation and experimental comparison can be considered as a validation of our protocol, making it substantially more convincing.

Data Availability

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

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grant nos. U1836115, 61672295, 61922045, and 61672290; the Peng Cheng Laboratory Project of Guangdong Province PCL2018KP004; the Postgraduate Research and Practice Innovation Program of Jiangsu Province (KYCX21_1003, KYCX21_0998, and KYCX21_1002); the CICAEET fund; and the PAPD fund.

References

[1] G. Manogaran, R. Varatharajan, D. Lopez, P. M. Kumar, R. Sundarasekar, and C. Thota, "A new architecture of Internet of Things and big data ecosystem for secured smart healthcare monitoring and alerting system," *Future Generation Computer Systems*, vol. 82, pp. 375–387, 2018.

[2] M. Mettler, "Blockchain technology in healthcare: the revolution starts here," in *Proceedings of the 2016 IEEE 18th International Conference on E-Health Networking, Applications and Services (Healthcom)*, pp. 1–3, IEEE, Munich, Germany, September 2016.

[3] J. Liang, Z. Qin, S. Xiao, L. Ou, and X. Lin, "Efficient and secure decision tree classification for cloud-assisted online diagnosis services," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 4, pp. 1632–1644, 2021.

[4] L. Catarinucci, D. De Donno, L. Mainetti et al., "An IoT-aware architecture for smart healthcare systems," *IEEE Internet of Things Journal*, vol. 2, no. 6, pp. 515–526, 2015.

[5] M. M. Baig and H. Gholamhosseini, "Smart health monitoring systems: an overview of design and modeling," *Journal of Medical Systems*, vol. 37, pp. 9898–9914, 2013.

[6] A. Solanas, C. Patsakis, M. Conti et al., "Smart health: a context-aware health paradigm within smart cities," *IEEE Communications Magazine*, vol. 52, no. 8, pp. 74–81, 2014.

[7] Y. S. Su, T. J. Ding, and M. Y. Chen, "Deep learning methods in Internet of medical things for valvular heart disease screening system," *IEEE Internet of Things Journal*, p. 1, 2021.

[8] Y. S. Su and S. Y. Wu, "Applying data mining techniques to explore user behaviors and watching video patterns in converged it environments," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–8, 2021.

[9] M. Hartmann, U. S. Hashmi, and A. Imran, "Edge computing in smart health care systems: review, challenges, and research directions," *Transactions on Emerging Telecommunications Technologies*, Article ID e3710, 2019.

[10] J. Qiu, X. Liang, S. Shetty, and D. Bowden, "Towards secure and smart healthcare in smart cities using blockchain," in *Proceedings of the 2018 IEEE International Smart Cities Conference (ISC2)*, pp. 1–4, Kansas City, MO, USA, September 2018.

[11] N. Tariq, A. Qamar, M. Asim, and F. A. Khan, "Blockchain and smart healthcare security: a survey," *Procedia Computer Science*, vol. 175, pp. 615–620, 2020.

[12] J. Alghazo, G. Rathee, S. Gupta et al., "A secure multimedia processing through blockchain in smart healthcare systems," *ACM Transactions on Multimedia Computing, Communications, and Applications*, 2020.

[13] J. Shen, Z. Gui, X. Chen, J. Zhang, and Y. Xiang, "Lightweight and certificateless multi-receiver secure data transmission protocol for wireless body area networks," *IEEE Transactions on Dependable and Secure Computing*, p. 1, 2020.

[14] J. Shen, T. Zhou, X. Chen, J. Li, and W. Susilo, "Anonymous and traceable group data sharing in cloud computing," *IEEE Transactions on Information Forensics and Security*, vol. 13, pp. 912–925, 2017.

[15] E. Stefanov, M. V. Dijk, E. Shi et al., "Path ORAM: an extremely simple oblivious ram protocol," in *Proceedings of the 2013 ACM SIGSAC Conference on Computer and Communications Security*, pp. 299–310, Berlin, Germany, November 2013.

[16] J. Shen, H. Yang, P. Vijayakumar, and N. Kumar, "A privacy-preserving and untraceable group data sharing scheme in cloud computing," *IEEE Transactions on Dependable and Secure Computing*, p. 1, 2021.

[17] B. Pinkas and T. Reinman, "Oblivious RAM revisited," in *Annual Cryptology Conference*, pp. 502–519, Springer, New York, NY, USA, 2010.

[18] M. O. Rabin, "How to exchange secrets with oblivious transfer," *IACR Cryptology ePrint Archive*, vol. 2005, no. 187, pp. 22–25, 2005.

[19] S. Even, O. Goldreich, and A. Lempel, "A randomized protocol for signing contracts," *Communications of the ACM*, vol. 28, no. 6, pp. 637–647, 1985.

[20] G. Brassard, C. Crépeau, and J. M. Robert, "All-or-nothing disclosure of secrets," in *Proceedings of the Conference on the Theory and Application of Cryptographic Techniques*, pp. 234–238, Springer, Kyoto, Japan, December 2000.

[21] M. Bellare and S. Micali, "Non-interactive oblivious transfer and applications," in *Proceedings of the Conference on the Theory and Application of Cryptology*, pp. 547–557, Springer, Houthalen, Belgium, April 1989.

[22] Y. Gertner, Y. Ishai, E. Kushilevitz, and T. Malkin, "Protecting data privacy in private information retrieval schemes," *Journal of Computer and System Sciences*, vol. 60, no. 3, pp. 592–629, 2000.

[23] M. Naor and B. Pinkas, "Efficient oblivious transfer protocols," in *Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms SODA*, vol. 1, pp. 448–457, Washington, DC, USA, January 2001.

[24] W.G. Tzeng, "Efficient 1-out-of-n oblivious transfer schemes with universally usable parameters," *IEEE Transactions on Computers*, vol. 53, no. 2, pp. 232–240, 2004.

[25] C. K. Chu and W. G. Tzeng, "Efficient k-out-of-n oblivious transfer schemes with adaptive and non-adaptive queries," in *Proceedings of the International Workshop on Public Key Cryptography*, pp. 172–183, Springer, Les Diablerets, Switzerland, January 2005.

[26] T. Chou and C. Orlandi, "The simplest protocol for oblivious transfer," in *Proceedings of the International Conference on Cryptology and Information Security in Latin America*, pp. 40–58, Springer, Guadalajara, Mexico, August 2015.

[27] E. Hauck and J. Loss, "Efficient and universally composable protocols for oblivious transfer from the CDH assumption," *IACR Cryptology ePrint Archive*, Report 2017/1011, 2017.

[28] X. Wang, X. Kuang, J. Li, J. Li, X. Chen, and Z. Liu, "Oblivious transfer for privacy-preserving in VANET's feature matching," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 7, pp. 4359–4366, 2021.

[29] M. Naor and B. Pinkas, "Oblivious transfer and polynomial evaluation," in *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing*, pp. 245–254, Atlanta, Georgia, May 1999.

[30] M. Naor and B. Pinkas, "Computationally secure oblivious transfer," *Journal of Cryptology*, vol. 18, no. 1, pp. 1–35, 2005.

[31] C. K. Chu and W. G. Tzeng, "Efficient k-out-of-n oblivious transfer schemes," *Jouranal of Universal Computer Science*, vol. 14, pp. 397–415, 2008.

[32] Y. Chen, J. S. Chou, and X. W. Hou, "A novel k-out-of-n oblivious transfer protocols based on bilinear pairings," *IACR Cryptology ePrint Archive*, vol. 2010, 2010.

[33] D. C. Lou and H. F. Huang, "An efficient-out-of-noblivious transfer for information security and privacy protection," *International Journal of Communication Systems*, vol. 27, no. 12, pp. 3759–3767, 2014.

[34] J. Lai, Y. Mu, F. Guo, R. Chen, and S. Ma, "Efficient k-out-of-n oblivious transfer scheme with the ideal communication cost," *Theoretical Computer Science*, vol. 714, pp. 15–26, 2018.

[35] J. H. Hsiao, R. Tso, C. M. Chen, and M. E. Wu, "Decentralized e-voting systems based on the blockchain technology," in *Advances in Computer Science and Ubiquitous Computing*, pp. 305–309, Springer, New York, NY, USA, 2017.

[36] R. Tso, Z.-Y. Liu, and J.-H. Hsiao, "Distributed e-voting and e-bidding systems based on smart contract," *Electronics*, vol. 8, no. 4, p. 422, 2019.

[37] T. Li, W. Ren, Y. Xiang et al., "FAPS: a fair, autonomous and privacy-preserving scheme for big data exchange based on oblivious transfer, ether cheque and smart contracts," *Information Sciences*, vol. 544, pp. 469–484, 2021.

[38] G. Ateniese, K. Benson, and S. Hohenberger, "Key-private proxy re-encryption," in *Proceedings of the Cryptographers' Track at the RSA Conference*, pp. 279–294, Springer, San Francisco, CA, USA, April 2009.

[39] D. Yaga, P. Mell, N. Roby, and K. Scarfone, "Blockchain technology overview," 2019, https://arxiv.org/abs/1906.11078.

[40] M. Pilkington, "Blockchain technology: principles and applications," in *Research Handbook on Digital Transformations*Edward Elgar Publishing, Cheltenham, UK, 2016.

[41] C. Cachin, "Architecture of the hyperledger blockchain fabric," in *Proceedings of the Workshop on Distributed Cryptocurrencies and Consensus Ledgers*,, Chicago, IL, July 2016.

[42] M. Vukolić, "The quest for scalable blockchain fabric: proof-of-work vs. BFT replication," in *Proceedings of the International Workshop on Open Problems in Network Security*, pp. 112–125, Springer, Zurich, Switzerland, October 2015.