Secure Fine-grained Attribute-based Access Control with Hidden Policy for Electronic Health Record System

Sai Ji[1], Xin Jin[1], Jin-Feng Lai[3], Jian Shen[1,2]

[1] School of Computer & Software, Nanjing University of Information Science & Technology, China
[2] Cyberspace Security Research Center, Peng Cheng Laboratory, Shenzhen, China
[3] School of Information and Communication Engineering, University of Electronic Science and Technology of China, China
jisai@nuist.edu.cn, ndghtxx@163.com, lcf2018@uestc.edu.cn, s_shenjian@126.com

Abstract

Electronic health record system (EHRs) has become an important part of medical system, which has more meaningful benefits compared with paper-based records. However, how to conduct secure fine-grained access control remains challenging. Although ciphertext-policy attribute-based encryption (CP-ABE) is a promising candidate for solving the above challenges. It is still not suitable for EHRs when considering privacy preserving. The access policy is uploaded to cloud in plaintext form, which may leak sensitive personal privacy. In this paper, we present a secure fine-grained attribute-based access control with hidden policy for electronic health record system. In the proposed scheme, a novel attribute name randomization scheme is designed to randomize each entity's attribute names. Therefore, each entity's attribute name set is different and unreadable. In addition, we utilize garbled bloom filter (GBF) to hide necessary values which are used to help decrypt ciphertext. At the same time, only user has corresponding secret keys can he reveal the hidden values. Moreover, security and performance analysis demonstrate that our scheme is secure and privacy-preserving with low overhead.

Keywords: Electronic health record system, CP-ABE, Access control, Hidden policy

1 Introduction

To date, the development of information technology has raised health care system [1-4] to a new height. In response to this trend, progressively more medical treatment records are recorded digitally and stored in electronic equipment. And traditional paper-based records are replaced by electronic health record (EHR) [5] gradually. EHR is a series of collection of patient's digital medical information. The electronic health record system (EHRs) can be shared or researched between medical institutions or scientific research institution. EHRs can manage patient data more efficiently [6]. It also greatly improves the efficiency and quality of health care services. More importantly, EHRs can meaningfully improve patient safety, providing full data chart to help doctor diagnose patient condition.

In order to realize secure and efficient patient data sharing [7], it is necessary to set strict access control in EHRs [8]. According to the feature of patient data, it contains many various data form which can be classified by different attributes. In addition, the data owner does not know the identity information of decryptor in advance. In another word, data owner does not know exactly which entity he wants to share his sensitive health data with. He may be only known that the receiver should satisfy some certain conditions. Under such condition, ciphertext-policy attribute-based encryption (CP-ABE) [9-12] becomes a suitable solution for the above challenges. CP-ABE grants data owner the ability to encrypt data under specified access policy, so that ciphertext cloud only be decrypted by some user whose attributes satisfy access policy. CP-ABE greatly enriches the flexibility of encryption strategies [13-14]. It is also a fine-grained one-to-many data sharing pattern. Therefore, CP-ABE based access control scheme can achieve the purpose of protecting patient sensitive data in EHR system.

However, the existing CP-ABE based access control schemes are flawed in terms of privacy protection [15-20]. Access policy contains a series of descriptive information. And it is uploaded to cloud in plaintext form along with ciphertext. This issue is especially important in EHRs. It may leak patient sensitive information. For instance, patient A encrypts his data under access policy "(otitis media AND varicose veins) OR ENT doctors OR vascular surgery doctors". This access policy shows that patient A has two types of diseases. Anyone who can access EHRs can see this access policy and infer that patient A may suffer from some physical illness, which leaks sensitive privacy information. This behavior seriously violates the patient's privacy rights. Hereafter, although some access control schemes with privacy preserving have

*Corresponding Author: Jin-Feng Lai; E-mail: lcf2018@uestc.edu.cn
DOI: 10.3966/160792642020072104004

been researched [21-23], a secure and fine-grained scheme with hidden policy is still to be developed. Some policy hidden schemes [24-25] only hide attribute values, but the attribute names are still exposed to the public. Malicious entities can still analyze sensitive privacy information in non-negligible probability. What's more, most of the above schemes do not support complicated access structures.

Thus, this motivates us to design a secure fine-grained attribute-based access control with hidden policy for EHR system, which could solve above problems. Note that, the access policy is expressed by linear secret sharing scheme (LSSS) [26]. The contributions of the proposed scheme can be concluded as follows:

(1) We utilize chameleon hash function to design attribute name randomization scheme, which is used to hide the true attribute names and randomize them. According to this scheme, central authority (CA) can randomize initial attribute names to assign each data owner and user special attribute names. And the randomized attribute names which in processed from the same initial attribute names correspond to the same attribute value.

(2) A novel attribute garbled bloom filter (GBF) is designed to replace the mapping function ρin LSSS access structure (M,ρ). There are no attribute name nor value in GBF. This attribute GBF also support expressive access policy.

(3) Due to the use of one-way anonymous key agreement, the proposed scheme can realize efficient secret parameter reconstruction. We use one-way anonymous key agreement to generate intermediate keys which correspond to attribute names. The intermediate keys are the values which are inserted into the above designed GBF. Only the user has the corresponding secret keys, can he/she reconstruct the intermediate keys which could further help user reconstruct secret parameter.

We will now explain the layout for the remaining of this paper. In the next three sections (i.e. Sections 2 to 3), we will briefly describe the relevant background materials, and the problem statement. In Section 4, we will present our proposed scheme. The security and performance analysis of the proposed scheme are given in Sections 5 and 6. This paper is concluded in the last section.

2 Background Materials

2.1 Chameleon Hash Functions

We introduce chameleon hash functions as it is described in [27]. Not like common hash functions, chameleon hash functions are a trapdoor collision resistant hash function. It has a special key pair (pk_{CH}, sk_{CH}). Public key pk_{CH} is utilized to generate hash value for each input. Secret key sk_{CH} can be utilized to easily find collisions for given hash value. A chameleon hash function has the following polynomial time algorithms:.

(1) SPGen(k) → SP. It takes a security parameter k as input. Then, it outputs the system parameters SP.

(2) KeyGen(SP) → (pk_{CH}, sk_{CH}). It takes system parameters SP as input. Then, this algorithm generates a trapdoor key pair (pk_{CH}, sk_{CH}).

(3) $H_{CH}(pk_{CH}, m, r_{CH})$ → h. It takes public key pk_{CH}, a message m, and an auxiliary random integer $r_{CH} \in Z_q^*$ as input. Then, it generates hash value h.

(4) ComputeCollision (sk_{CH}, m, r_{CH}, m') → r'_{CH}. This algorithm takes secret key sk_{CH}, a message m, an auxiliary random integer r_{CH} as input. Then, it generates a new auxiliary random integer r'_{CH} which satisfy $H_{CH}(pk_{CH}, m, r_{CH}) = H_{CH}(pk_{CH}, m', r'_{CH})$.

A secure chameleon hash function has the properties of collision resistance and semantic security.

2.2 Garbled Bloom Filters

GBF is a variant of Bloom filters [28] and incorporates (k, k) secret sharing scheme. With the help of [29], the construction of GBF is described as follows:

(1) Initialization. Create an array of m bit strings, whose index numbers over the range [0, m-1]. Each element in GBF is λ bit, where λ is security parameter. And they firstly are set to NULL. Choose k independent uniform hash functions $H_i : \{0,1\}^* \rightarrow [0, m-1], \forall 1 \le i \le k$. The i-th element in GBF is represented as GBF[i].

(2) Insert. For element $x \in S$ which will be added to GBF, it is mapped to k positions which are computed as $H_i(x)_{i\in[1,k]}$ in turn. Write down the position idx whose λ bit is NULL after the first hash. For the hashed position after this, if the corresponding string is empty, we randomly set it to a string of length λ. Otherwise, it remains unchanged. We XOR the above strings other than the string labeled position idx. Then, we XOR the result with element x. The finally result is assigned to GBF [idx]. After inserting all elements, all the GBF [i] which are not assigned any string will be assigned random string.

The construction of attribute GBF in our scheme which is expressed by GBFBuild (M,ρ) algorithm is similar to above. The process of GBF can refer to Figure 1.

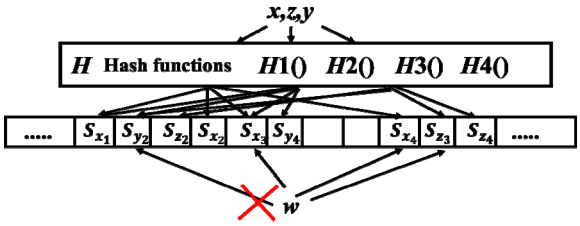

Figure 1. Garbled bloom filter

3 Problem Statement

3.1 System Model

Figure 2 describes our proposed scheme, which has four entities. Namely, there are cloud servers, central authority (CA), data owners (DOs) and users.

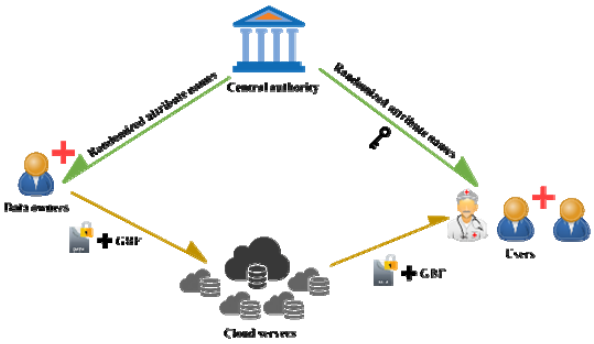

Figure 2. System model

Cloud servers provide electronic health record storage service. It is not fully trusted here. It is honest but curious.

Central authority initializes the whole system. And it is fully trusted here. In EHRs, it may be led by government or social public health agency.

Data owners are the data source in EHRs. They encrypt their data and upload them to cloud servers. They may play the role of patient or doctor. They are fully trusted in EHR system.

Users are data consumers in EHRs. They are not fully trusted in EHR system. They may collude together to decrypt ciphertext.

3.2 Security Assumption

Decisional q -Parallel Bilinear Diffie-Hellman Exponent Assumption (*Decisional q-BDHE* [11]): The description of q-BDHE is as follows. Let p be the prime order of group $\mathbb{G}$. And randomly select $a,s,b_1,b_2,...,b_q \in \mathbb{Z}_p$. If an adversary is given $\vec{y} =$

$$g, g^s, g^a, ..., g^{(a^q)}, g^{(a^{q+2})}, ..., g^{(a^{2q})}$$
$$\forall 1 \le j \le q, g^{sb_j}, g^{a/b_j}, ..., g^{(a^q/b_j)},$$
$$g^{(a^{q+2}/b_j)}, ..., g^{(a^{2q}/b_j)}$$
$$\forall 1 \le j,k \le q, k \ne j, g^{asb_j/b_j}, ..., g^{(a^q b_k/b_j)}$$

it must remain hard to distinguish $e(g,g)^{a^{q+1}s} \in \mathbb{G}_T$ from a random element in $\mathbb{G}_T$.

An algorithm B that outputs $z \in \{0,1\}$ has advantage $\in$ in dealing with decisional *q-BDHE* in $\mathbb{G}$ if

$$|P_r[B(\vec{y},T) = e(g,g)^{a^{q+1}s}) = 0]$$
$$-P_r[B(\vec{y},T = R) = 0 \ge \in$$

We say that the *Decisional q -BDHE* assumption holds if no adversary can work out the decisional q-BDHE problem in polynomial time with a non-negligible advantage.

4 The Proposed Scheme

4.1 High-level Overview

CA firstly initialize the whole system and deal with registration tasks. When DOs and users register with CA, CA executes the attribute name randomization scheme to randomize each entity's initial attribute names to assign each entity the corresponding unique attribute names. DOs adaptively choose a symmetric encryption algorithm to encrypt data. Then, DOs encrypt the symmetric key under CP-ABE. For each attribute in access policy (M,ρ), DOs utilize one-way anonymous key agreement to generate intermediate key for each attribute name. After that, DOs hash intermediate keys instead of attribute names to construct GBF. The access policy (M,ρ) is replaced as (M, GBF). Once user wants to decrypt ciphertext, he firstly utilizes his own secret key to compute intermediate key. Then, he searches GBF to check whether he can further decrypt ciphertext. In the whole process, the access structure is fully protected.

4.2 System Setup

CA chooses two cyclic multiplicative groups $\mathbb{G}_1$ and $\mathbb{G}_2$. All of them have the same prime order p. A generator $g \in \mathbb{G}_1$ is randomly selected. Let $e: \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ be a binary map. Then, CA randomly selects a and $\alpha \in \mathbb{Z}_p$. For each initial attribute name $ATName_i (i = 1,2,\ldots,n)$, CA randomly selects the corresponding attribute value $d_i \in \mathbb{G}_1$. The initial attribute name is the widely accepted official attribute naming rule, for example, the attribute "psychologist, heart disease, patient, depression and cough" and so on. CA selects k secure strong collusion-resistant hash functions $H_i(\ ): \mathbb{G}_1 \to [0, L_{GBF} - 1], \forall 1 \le i \le k$, where L_{GBF} is the maximum index number.

CA publishes public key as Eq. (1).

$$PK = \{ATName_1, ATName_2, ..., ATName_n \\ e(g,g)^{\alpha}, H_{CH}, hash(\ ), d_1, ..., d_n \\ g, g^{\alpha}, \mathbb{G}_1, \mathbb{G}_2, L_{GBF}, H(\ )_1, ..., H_k(\ )\} \quad (1)$$

CA sets master secret key $MSK = g^{\alpha}$.

4.3 Attribute Name Randomization Scheme

Another mission of CA is to deal with each entity's registration process and execute attribute name randomization scheme.

CA selects a secure chameleon hash function $H_{CH}\{0,1\}^* \rightarrow \mathbb{G}_1$. CA runs the SPGen(k) and KeyGen (SP) algorithm to generate chameleon hash key pair (pk_{CH}, sk_{CH}).

When user registers with CA, CA will assign a unique identity GID_j. According to the initial attribute names where the total number is n owned by user GID_j, CA randomize each one as follows. CA calculates $H_{CH}(pk_{CH}, ATName_i \| d_i, r_{Inti})$. Then, CA randomly selects auxiliary integer $r_i (1 \le i \le n)$ and utilizes chameleon hash secret key sk_{CH} to run ComputeCollision $(sk_{CH}, ATName_i \| d_i, r_{Inti}, r_i)$ to find the string $RT_{j,i} \in \{0,1\}^* (1 \le i \le n)$, so that a collision is successfully found. Finally, user's attribute names are randomized as $RT_{j,i}$. The corresponding auxiliary integers r_i are also sent to user. So, user has attribute value d_i, the corresponding randomized attribute names $RT_{j,i}$ and auxiliary integers r_i, where $(1 \le i \le n)$. For simplicity, we utilize $H_{CH}(RT_{j,i})$ to represent $H_{CH}(pk_{CH}, RT_{j,i}, r_i)$ which is equal to $(pk_{CH}, ATName_i \| d_i, r_{Inti})$. Here DO also needs to register with CA to acquire his identity GID_{doj}, his randomized attribute names $RT_{doj,i}$, the corresponding attribute values d_i, $H_{CH}(RT_{doj,i})$ and auxiliary integers r_{doi}, where $(1 \le i \le n)$. Each entity's attribute set is expressed as S_{GID_j} or $S_{GID_{doj}}$ according to their roles.

4.4 Encryption Phase

First, DO whose identity string is GID_{doj} randomly selects number $\mu \in \mathbb{G}_2$ in secret. Then, DO encrypts data *M* by using symmetric encryption algorithm under the selected symmetric key μ. After this, DO will encrypt symmetric key μ under CP-ABE algorithm and generate the initial ciphertext. DO formulates the access policy $\mathbb{A}$ according to his own interest. Then, DO turns access policy $\mathbb{A}$ into LSSS access structure (M, ρ). As we introduce above, *M* is an l ×n matrix. ρ is a injective function which maps each row of *M* to attributes. DO randomly selects $x, y_2, \ldots, y_n \in \mathbb{Z}_q$ in secret. Then, a vector $\vec{v} = (s, y_2, \ldots, y_n)$ is constructed by DO. At the same time, each $\lambda_i = M_i \vec{v}^{\top}$ is computed, where i=1, 2, …, l. Finally, DO randomly selects $r_1, r_2, \ldots, r_l \in \mathbb{Z}_p$ and utilizes PK to generate the following initial ciphertext as Eq. (2).

$$ES_{\delta}(M), C = \mu e(g,g)^{as}, C' = g^s \\ \forall i = 1 \, to \, l, C_i = (g^a)^{\lambda_i} \cdot d_{\rho(i)}^{-\gamma_i}, D_i = g^{\lambda_i} \quad (2)$$

Then, DO utilizes secret parameter s which is selected in first step to compute intermediate key $IK_{RTdoj,i} = e((g^a)^s, H_{CH}(RT_{doj,i}))$ for each attribute name.

Following is the GBF construction phase. GBFBuild $(M, \rho) \rightarrow$ GBF. This algorithm takes access policy (M, ρ) as input, then outputs attribute GBF. Following is the detailed process. DO sets a set of elements $S_{insert} = \{row_1 \| row_2 \| ... \| row_n\}$, where $row_n \in \mathbb{Z}_q^*$ represents the input of function ρ and the output of $\rho(row_n)$ is same, and n is the number of the input which has the same corresponding output. According to the algorithm defined in [29], DO can construct attribute GBF as follows. For each element $e \in S_{insert}$, it is firstly shared with (k,k) secret sharing scheme. Namely, DO randomly generates k-1 λ-bit strings s(i,e), where $1 \le i \le k-1$. And DO sets $s_{k,e} = s_{1,e} \oplus s_{2,e} \oplus ... \oplus s_{k-1,e} \oplus e$. Then, it utilizes hash functions $H_i(\ )$ and intermediate key $IK_{RTdoj,i}$ to compute each position. Namely, DO will get the following position for element e: $H_1(IK_{RTdoj,i}), H_2(IK_{RTdoj,i}), ..., H_k(IK_{RTdoj,i})$, where $IK_{RTdoj,i}$ corresponds to the attribute name associated with element e and $H_k(IK_{RTdoj,i})$ represents the position index. At the same time, DO assigns each position the corresponding string $s_{i,e}$.

Finally, the ciphertext CT is marked as Eq. (3).

$$C = \mu e(g,g)^{as}, C' = g^{as}, (M, GBF), \\ \forall i = 1 \, to \, l, C_i = (g^a)^{\lambda_i} \cdot d_{\rho(i)}^{-\lambda_i}, D_i = g^{r_i} \quad (3)$$

4.5 Key Generation Phase

When user whose identity string is GID_j wants to decrypt ciphertext, he needs to acquire his secret key. Firstly, user needs to interact with CA to verify his legality. If user is legal, CA will go on to generate secret key for user. Then CA will execute KeyGen (PK, MSK, S_{GIDj}) → SK algorithm to generate secret key. It takes public key PK, master secret key MSK and s set of user's unique attribute, S_{GIDj} as input. Then, CA randomly selects $t \in \mathbb{Z}_q^*$ and calculates Eq. (4).

$$K = g^a g^{at}, L = g^t,$$
$$\forall x \in S_{GIDj}, K_x d_x^t, T_x = H_{CH}(RT_{j,x})^a \quad (4)$$

4.6 Decryption Phase

The registered user can access ciphertext according to their interests. But they cannot decrypt ciphertext only if their attributes satisfy access policy $\mathbb{A}$. So user needs to reveal related parameter embedded in access structure (*M*, GBF). User could decrypt ciphertext like follows.

This phase is the process that user queries the attribute GBF to acquire necessary row number information, which can be marked as GBFQuery. Firstly, user with identity GID_j utilizes ciphertext and secret key to reconstruct the intermediate key $IK_{RTj,i} = e(g^s, H_{CH}(RT_{j,i})^a)$ for each randomized attribute name which belongs to him. If $H_{CH}(RT_{doj,i}) = H_{CH}(RT_{j,i})$, then $IK_{RTj,i} = IK_{RTdoj,i}$. For each intermediate key $IK_{RTj,i}$, user calculates the corresponding index numbers in attribute GBF with the help of k hash functions $H_n(\,)\forall 1 \le n \le k$. Namely, user computes the following positions: $H_1(IK_{RTj,i}), H_2(IK_{RTj,i}), ..., H_k(IK_{RTj,i})$. Then, user gets the corresponding strings from these positions. After that, user reconstructs the element e: $e = s_{1,e} = s_{2,e} \oplus ... \oplus s_{k-1,e} \oplus s_{k,e} = s_{1,e} \oplus s_{2,e} \oplus ... \oplus s_{k-1,e} \oplus s_{1,e} \oplus s_{2,e} \oplus ... \oplus s_{k-1,e} \oplus e$. Therefore, user can reveal the row number information embedded in e, where $e = \{row_1 \| row_2 \| ... \| row_n\}$. Finally, user can construct his attribute matrix M_U which is a sub-matrix of M according to the revealed row number information. The attributes in matrix M_U make up the set S'_{GIDj}. Note that, the whole mapping function ρ is still kept secret. If user has all attributes in access policy $\mathbb{A}$, he still cannot determine whether he has all attributes in specified attribute GBF due to our attribute name randomization scheme and intermediate key scheme.

If user's attribute matrix M_U satisfy the hidden access policy (M, ρ), then vector $\vec{e} = (1, 0, \ldots, 0)$ is in the span of matrix M_U. Then user can compute constant set $\{\omega_i\}_{i \in I}$ in polynomial time, where $\vec{e} = (\omega_1, \omega_2, ... \omega_{|I|}) M_U$. After that, user can reconstruct secret parameter s: s=(1, 0, …, 0) $\cdot (s, y_2, ..., y_n)^\top$. With the above parameters, user can reveal symmetric key μ with Eq. (5).

$$C_i = \frac{e(C', K)}{\prod i \in I(e(C_i, L) \cdot e(D_i, K_{\rho(i)}))^{\varpi_i}} = e(g,g)^{as} \quad (5)$$

Therefore, the symmetric key μ can be calculated as follows: $\mu = C / C_k = \mu e(g,g)^{as} / e(g,g)^{as}$. Once μ is restored, user can further decrypt plaintext M under the selected symmetric algorithm.

5 Security Analysis

5.1 Data Confidentiality

Theorem 1: If the *q-BDHE* assumption holds, no polynomial time adversary can selectively break our scheme with a challenge matrix of size $l^* \times n^*$, where $l^*, n^* \le q$.

Proof: This phase is similar to that in [11], we do not proof it here. Detailed proof can be found in [11].

5.2 Privacy Preservation

Theorem 2: The access policy is fully hidden in our scheme and our scheme is privacy-preserving.

Proof: In most existing CP-ABE based schemes, the access policy is expressed as LSSS structure (M, ρ) and is attached to ciphertext in plaintext form. The mapping function ρ fully reveals the readable attribute name information and access structure information, which leaks sensitive privacy information of DO and user. In our proposed scheme, such mapping function ρ is removed and is replaced by attribute GBF. Firstly, in our novel attribute name randomization scheme, CA computes $H_{CH}(pk_{CH}, ATName_i \| d_i, r_{Inti})$, where parameter r_{Inti} is kept secret. Then, CA utilizes chameleon hash secret key sk_{CH} run ComputeCollision $(sk_{CH}, ATName_i \| d_i, r_{Inti}, r_i)$ to find the string $RT_{j,i} \in \{0,1\}^* (1 \le i \le n)$, so that a collision is successfully found. Therefore, each entity's attribute names are randomized as $\{RT_{j,i}\}$ and there is no relation between each randomized attribute name. Each entity's attribute names are unreadable and different from each other.

After that, DO utilizes his own special $H_{CH}(RT_{doj,i})$ to compute intermediate key $IK_{RTdoj,i} = e((g^a)^s, H_{CH}(RT_{doj,i}))$. Then, DO utilize $IK_{RTdoj,i}$ to generate GBF. In GBF, each element is a random string in the eyes of others. Only user has the correct secret key, can he compute the corresponding unique intermediate key and further reveal the row mapping information. User can only reveal row mapping information by decrypting GBF. There is no sensitive privacy information in GBF. Therefore, the access policy is fully hidden and our scheme is privacy-preserving.

6 Performance Analysis

In this section, we firstly simulate the construction and query time of GBF with Python 2.7.15rc1 and MurmurHash3 2.5.1. Then, we simulate the encryption and decryption cryptographic operation with C Programming Language. Both the experiment environment is on a VMware Workstation machine with Intel Core i7-7700HQ processors running at 2.80 GHz with 4 GB memory, and Ubuntu 18.04.1 LTS. The cryptography library used is the PBC Library (pbc-0.5.14) and the Libfenc Library (fenc-0.2.0).

Figure 3 and Figure 4 show the simulation result of GBFBuild and GBFQuery. We set k=8, 16, 24 with the help of MurmurHash functions. m is set as 1024 and λ = 16. And n is the number of attributes. Form Figure 3, we can find that time increases with the increase of both number of hash functions and attributes. The insert time of 50 attributes with 24 hash functions is less than 2.4 ms. From Figure 4, we can find that the query time is much smaller than insert operation. And the query time of 50 attributes with 24 hash functions is less than 0.90 ms.

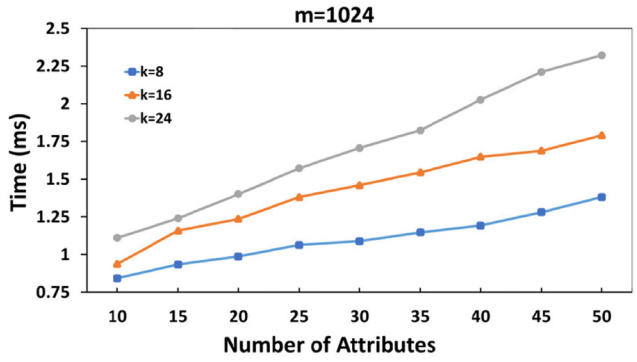

Figure 3. GBFBuild

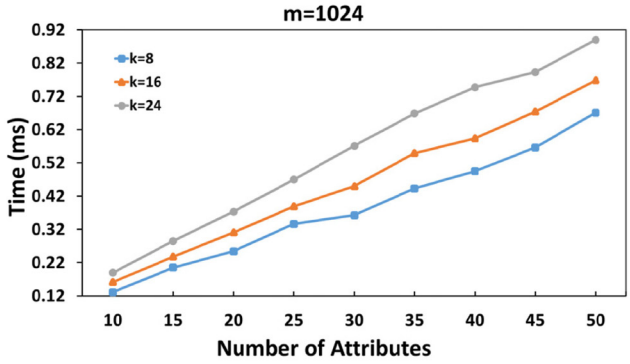

Figure 4. GBFQuery

Figure 5 shows the result of our encryption and decryption simulation. Figure 5 also includes the construction and query time of GBF, where k=16. The number of attributes is marked as x-axis. It represents the number of attributes in LSSS access structure, ciphertext and user's attribute set. To make sure that each attribute will appear in encryption and decryption phase, we set the access policy in the form of all AND-gates. Namely, no OR-gate appears in our simulation. The y-axis denotes the time cost which is measured in seconds. Note that, we do not take the symmetric encryption and decryption into consideration in our simulation. We simulate each one for 20 times and take the average value as the final result. From Figure 5, we can find that both time increases linearly with the increment of attribute number. We compare our scheme with [10] from encryption and decryption phase. The result is shown in Figure 6. Because our scheme includes GBFBuild and GBFQuery, the total time of ours is higher than [10]. However, the excess time is so small. Ours takes about 1.8 ms more than [10] in encryption phase and takes about 0.8 ms more than [10]. We can find that the simulation result is acceptable for real-world applications.

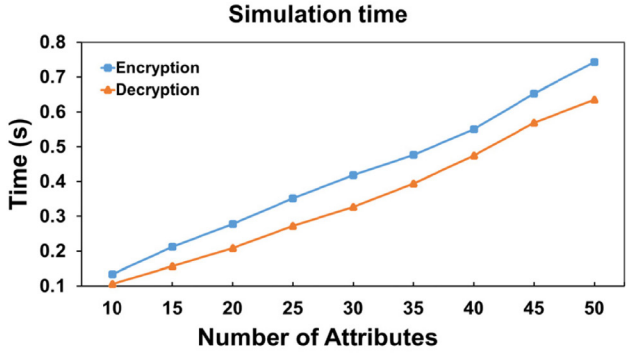

Figure 5. Simulation time

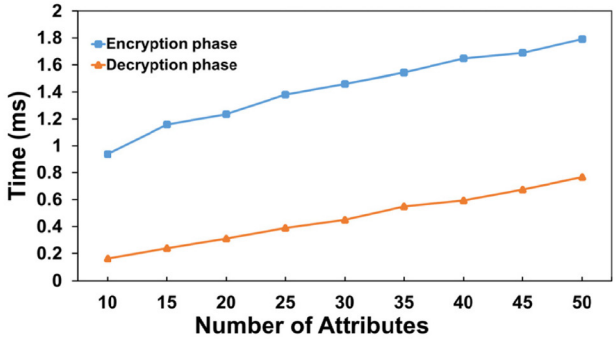

Figure 6. The comparison result between ours and [10]

7 Conclusion

The privacy protection is so important in EHRs. In this paper, a hidden policy access control which protects sensitive privacy in EHRs is proposed. Based on the designed attribute name randomization scheme, each entity's attribute set is randomized which makes it different from others' and unreadable. The randomized attribute name set is also utilized to generate intermediate key. Then, a novel attribute GBF is designed to replace the mapping function ρin LSSS structure (M, ρ). The hidden values in attribute GBF

are row number information. Moreover, we utilize one-way anonymous agreement to design an intermediate key scheme. The intermediate key is the inserted value in attribute value. Note that, the intermediate key is generated with the help of randomized attribute set. And only the user has the correct corresponding secret key, can he reconstruct the intermediate key and reveal the hidden information in attribute GBF. The security and performance analysis show that the proposed is secure and privacy-preserving with low overhead.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grants No. U1836115, No. 61672295, No. 61922045, No. 61672290, the Natural Science Foundation of Jiangsu Province under Grant No. BK20181408, the Foundation of State Key Laboratory of Cryptology under Grant No. MMKFKT201830, the Peng Cheng Laboratory Project of Guangdong Province PCL2018KP004, the CICAEET fund, and the PAPD fund.

References

[1] J. A. Linder, J. Ma, D. W. Bates, B. Middleton, R. S. Stafford, Electronic Health Record Use and the Quality of Ambulatory Care in the United States, *Archives of Internal Medicine*, Vol. 167, No. 13, pp. 1400-1405, July, 2007.

[2] P. C. Tang, J. S. Ash, D. W. Bates, J. M. Overhage, D. Z. Sands, Personal Health Records: Definitions, Benefits, and Strategies for Overcoming Barriers to Adoption, *Journal of the American Medical Informatics Association*, Vol. 13, No. 2, pp. 121-126, March, 2006.

[3] M. Li, S. Yu, Y. Zheng, K. Ren, W. Lou, Scalable and Secure Sharing of Personal Health Records in Cloud Computing Using Attribute-Based Encryption, *IEEE Transactions on Parallel & Distributed Systems*, Vol. 24, No. 1, pp. 131-143, January, 2013.

[4] A. Bahga, V. K. Madisetti, A Cloud-Based Approach for Interoperable Electronic Health Records (EHRs), *IEEE Journal of Biomedical & Health Informatics*, Vol. 17, No. 5, pp. 894-906, September, 2013.

[5] K. Häyrinen, K. Saranto, P. Nykänen, Definition, Structure, Content, Use and Impacts of Electronic Health Records: A Review of the Research Literature, *International Journal of Medical Informatics*, Vol. 77, No. 5, pp. 291-304, May, 2008.

[6] A. M. V. Ginneken, The Computerized Patient Record: Balancing Effort and Benefit, *International Journal of Medical Informatics*, Vol. 65, No. 2, pp. 97-119, June, 2002.

[7] C. Wang, J. Shen, Q. Liu, Y. Ren, T. Li, A Novel Security Scheme Based on Instant Encrypted Transmission for Internet of Things, *Security and Communication Networks*, Vol. 2018, pp. 1-7, May, 2018.

[8] H. Yan, J. Li, X. Li, G. Zhao, S.-Y. Lee, J. Shen, Secure Access Control of E-Health System with Attribute-Based Encryption, *Intelligent Automation & Soft Computing*, Vol. 22, No. 3, pp. 345-352, February, 2016.

[9] V. Goyal, O. Pandey, A. Sahai, B. Waters, Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data, *13th ACM Conference on Computer and Communications Security*, Alexandria Virginia, USA, 2006, pp. 89-98.

[10] J. Bethencourt, A. Sahai, B. Waters, Ciphertext-Policy Attribute-Based Encryption, *2007 IEEE Symposium on Security & Privacy (SP'07)*, Berkeley, CA, USA, 2007, pp. 321 - 334.

[11] B. Waters, Ciphertext-Policy Attribute-Based Encryption: An Expressive, Efficient, and Provably Secure Realization, *International Workshop on Public Key Cryptography*, Taormina, Italy, 2011, pp. 53-70.

[12] C.-J. Wang, Y. Liu, J.-T. Kim, An IND-CCA2 Secure Key-Policy Attribute-Based Key Encapsulation Scheme, *Journal of Internet Technology*, Vol. 11, No. 5, pp. 619-625, September, 2010.

[13] W. Li, K. Xue, Y. Xue, J. Hong, Tmacs: A Robust and Verifiable Threshold Multi-Authority Access Control System in Public Cloud Storage, *IEEE Transactions on Parallel and Distributed Systems*, Vol. 27, No. 5, pp. 1484-1496, May, 2016.

[14] S. Wang, J. Zhou, J. K. Liu, J. Yu, J. Chen, W. Xie, An Efficient File Hierarchy Attribute-Based Encryption Scheme in Cloud Computing, *IEEE Transactions on Information Forensics and Security*, Vol. 11, No. 6, pp. 1265-1277, June, 2016.

[15] K. Xue, W. Chen, W. Li, J. Hong, P. Hong, Combining Data Owner-Side and Cloud-Side Access Control for Encrypted Cloud Storage, *IEEE Transactions on Information Forensics and Security*, Vol. 13, No. 8, pp. 2062-2074, August, 2018.

[16] K. Yang, X. Jia, Expressive, Efficient, and Revocable Data Access Control for Multi-Authority Cloud Storage, *IEEE Transactions on Parallel and Distributed Systems*, Vol. 25, No. 7, pp. 1735-1744, July, 2014.

[17] F. Guo, Y. Mu, W. Susilo, D. S. Wong, V. Varadharajan, CP-ABE with Constant-Size Keys for Lightweight Devices, *IEEE transactions on Information Forensics and Security*, Vol. 9, No. 5, pp. 763-771, May, 2014.

[18] J. Li, W. Yao, J. Han, Y. Zhang, J. Shen, User Collusion Avoidance CP-ABE with Efficient Attribute Revocation for Cloud Storage, *IEEE Systems Journal*, Vol. 12, No. 2, pp. 1767-1777, June, 2018.

[19] J. Dong, Q. Zhao, Security Access Control Policy of Information System under Multi-domain Mode, *International Journal of Internet Protocol Technology*, Vol. 11, No. 1, pp. 44-50, May, 2018.

[20] Z. Qiu, Z. Zhang, S. Tan, J. Wang, X. Tao, Hierarchical Access Control with Scalable Data Sharing in Cloud Storage, *Journal of Internet Technology*, Vol. 20, No. 3, pp. 663-676, May, 2019.

[21] T. Nishide, K. Yoneyama, K. Ohta, Attribute-Based Encryption with Partially Hidden Encryptor-Specified Access Structures, *Applied Cryptography and Network Security*, New York, USA, 2008, pp. 111-129.

[22] J. Li, K. Ren, B. Zhu, Z. Wan, Privacy-Aware Attribute-Based Encryption with User Accountability, *International Conference on Information Security*, Pisa, Italy, 2009, pp. 347-362.

[23] D. Boneh, B. Waters, Conjunctive, Subset, and Range Queries on Encrypted Data, *Theory of Cryptography Conference*, Amsterdam, The Netherlands, 2007, pp. 535-554.

[24] J. Katz, A. Sahai, B. Waters, Predicate Encryption Supporting Disjunctions, Polynomial Equations, and Inner Products, *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Istanbul, Turkey, 2008, pp. 146-162.

[25] J. Lai, R. H. Deng, Y. Li, Fully Secure Cipertext-Policy Hiding CP-ABE, *International Conference on Information Security Practice and Experience*, Guangzhou, China, 2011, pp. 24-39.

[26] A. Beimel, *Secure Schemes for Secret Sharing and Key Distribution*, Ph.D. Thesis, Israel Institute of Technology, Technion, Haifa, 1996.

[27] X. Chen, F. Zhang, K. Kim, Chameleon Hashing Without Key Exposure, *International Conference on Information Security*, Palo Alto, CA, USA, 2004, pp. 87-98.

[28] B. H. Bloom, Space/Time Trade-Offs in Hash Coding with Allowable Errors, *Communications of the ACM*, Vol. 13, No. 7, pp. 422-426, July, 1970.

[29] C. Dong, L. Chen, Z. Wen, When Private Set Intersection Meets Big Data: An Efficient and Scalable Protocol, *2013 ACM SIGSAC Conference on Computer & Communications Security*, Berlin, Germany, 2013, pp. 789-800.

Biographies

Sai Ji received his M.S. degree from the Nanjing Aeronautics and Astronautics University, Nanjing, China, in 2006. He works as an Associate Professor at the NUIST. His research interests are in the areas of computer measurement and control, structural health monitoring, and WSNs.

Xin Jin received the B.E. degree in 2017 and is currently working toward the M.E. degree at Nanjing University of Information Science and Technology, Nanjing, China. He focuses on information security and access control. His research interests include information security, access control, and attribute-based encryption.

Jin-Feng Lai reis currently with the School of Information and Communication Engineering, University of Electronic Science and Technology of China. He has authored or coauthored over 100 refereed papers in journals, conferences, and workshop proceedings about his research areas within four years. His research interests include multimedia communications, sensor-based healthcare, and embedded systems. He is a member of the IEEE CIRCUITSAND SYSTEMS and the IEEE Communications Societies.

Jian Shen received the Ph.D. degrees in computer science from Chosun University, South Korea, in 2012. Since 2012, he has been a Professor with the Nanjing University of Information Science and Technology, Nanjing, China. His research interests include public cryptography, cloud computing, data auditing and sharing, and information security systems.