This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2018.2790943, IEEE Transactions on Services Computing

Inference Attack-Resistant E-Healthcare Cloud System with Fine-Grained Access Control

Wei Zhang, Yaping Lin, *Member, IEEE,* Jie Wu, *Fellow, IEEE,* and Ting Zhou

Abstract—The e-healthcare cloud system has shown its potential to improve the quality of healthcare and individuals' quality of life. Unfortunately, security and privacy impede its widespread deployment and application. There are several research works focusing on preserving the privacy of the electronic healthcare record (EHR) data. However, these works have two main limitations. First, they only support the 'black or white' access control policy. Second, they suffer from the inference attack. In this paper, for the first time, we design an inference attack-resistant e-healthcare cloud system with fine-grained access control. We first propose a two-layer encryption scheme. To ensure an efficient and fine-grained access control over the EHR data, we design the first-layer encryption, where we devise a specialized access policy for each data attribute in the EHR, and encrypt them individually with high efficiency. To preserve the privacy of role attributes and access policies used in the first-layer encryption, we systematically construct the second-layer encryption. To take full advantage of the cloud server, we propose to let the cloud execute computationally intensive works on behalf of the data user without knowing any sensitive information. To preserve the access pattern of data attributes in the EHR, we further construct a blind data retrieving protocol. We also demonstrate that our scheme can be easily extended to support search functionality. Finally, we conduct extensive security analyses and performance evaluations, which confirm the efficacy and efficiency of our schemes.

Index Terms—E-healthcare cloud, electronic healthcare record (EHR), inference attack, fine-grained access control, two-layer encryption.

1 Introduction

1.1 Motivation

The electronic healthcare, providing timely, accurate, and low-cost healthcare services, has shown its potential to improve the quality of healthcare and individuals' lives. Many companies all over the world have developed their healthcare services, e.g., Google Fit [1], Apple HealthKit [2], etc. Meanwhile, with the increasing maturity and benefits brought by cloud computing, the e-healthcare cloud system has attracted many interests from both the academic and the industry. The IBM company has already established its e-healthcare cloud center, i.e., IBM Watson Health Cloud [3]).

Unfortunately, security and privacy will impede the widespread deployment and application of the e-healthcare cloud system. The fundamental reason is that, once the sensitive EHR data are outsourced to the cloud, data owners would lose their control [4], [5], [6], [7]. Although the cloud service providers promise they will preserve these data by installing anti-virus softwares, firewalls, and intrusion detection and prevention systems [8], they cannot stop their employees from accessing these data. For example, an employee in the department of veterans affairs once takes away 26.5 million sensitive data without authorization, which includes the social security numbers and sensitive health data [9]. When these sensitive data are abused, more serious problems will occur. For example, insurance companies would refuse to provide insurance to those who have serious health problems. Therefore, it is vital to preserve the security and privacy of EHR data stored in the e-healthcare cloud system.

1.2 Limitations of Prior Art

To preserve the security and privacy of the EHR data, some research works have been done in [10], [11], [12], [13]. However, they suffer from three main limitations.

First, they only support the 'black or white' access control policy. Specifically, once a data user is authorized, he can access all the data attributes in the EHR. For example, if a dentist is authorized to access a patient's EHR, then he can even access the patient's social security number, or health data regarding that patient's liver or kidneys. We argue that, if this problem is not solved, serious privacy leakage will occur.

- *W. Zhang, Y.p. Lin, and T. Zhou are with College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China, and with Key Laboratory of Trusted Computing and Networks in Hunan Province, Changsha, 410082, China. Yaping Lin is the Corresponding Author. E-mail: {zhangweidoc, yplin, zhouting}@hnu.edu.cn.*
- *J. Wu is with Department of Computer and Information Sciences, Temple University, 1805 N. Broad St., Philadelphia, PA 19122. E-mail: jiewu@temple.edu.*

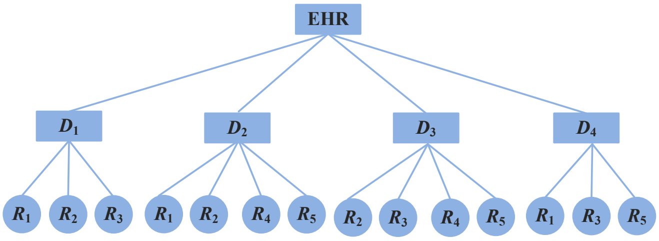

Fig. 1: Example of data attributes and role attributes contained in the EHR.

Second, they suffer from the inference attack. The inference attack includes the frequency analysis attack, sorting attack, and cumulative attack. Among them, the most well known attack is the frequency analysis attack, which breaks the classical encryption algorithms [14]. In the EHR scenario, the inference attack is rooted in two aspects. 1) By observing the access frequency of the EHRs, the cloud can deduce the content of the EHRs with some background information even if they are encrypted. 2) Existing schemes adopt the conventional ciphertext policy attribute-based encryption to encrypt the EHR, which inevitably expose the access policy to the cloud. Therefore, the cloud can deduce sensitive data from the EHR with some background information. For example, if the cloud observes that a patient's EHR can be accessed by the doctors from the department of radiologist and chemotherapy, then the cloud can deduce that patient to have cancer with high probability.

Third, they have to spend a large amount of time on secret generation for the repeated items. As shown in Fig. 1, the EHR has four data attributes, i.e., D_1, D_2, D_3, D_4. Each data attribute has its own role attributes. As we can see, there are a lot of repeated role attributes in the EHR. In conventional schemes, instead of generating ciphertext for the 5 distinct role attributes, they have to generate secrets for all the 14 role attributes, which means that the efficiency can be improved for nearly three times in this example. Since the data attributes in the EHR often have a lot of repeated role attributes, we need to propose schemes to save the computation cost spent on the repeated role attributes.

1.3 Technical Challenges

To design an efficient and inference attack-resistant e-healthcare cloud system with fine-grained access control, there are three key challenges.

1) To achieve the fine-grained access control, we need to define a specialized access policy for each data attribute in the EHR. Since different data attributes in the EHR usually share many role attributes in their access policies, for security concerns, we need to conceal the frequency of role attributes occurring in the EHR. Therefore, how to ensure an efficient and correct encryption on the data attributes while preserving the statistical data of the role attributes is a challenging problem.
2) To improve the efficiency of the whole system, the cloud is expected to execute computationally intensive works on behalf of the data users. Thus, how to prevent the cloud from deducing sensitive data, while achieving the above functionality is very important.
3) Since the cloud possesses all the EHR data and is responsible for returning accessed data, how to ensure the cloud correctly and efficiently returns the data attributes without knowing which data attributes are actually returned is also a challenging problem.

1.4 Our Approach and Key Contributions

In this paper, for the first time, we design an inference attack-resistant e-healthcare cloud system with fine-grained access control. We first propose a two-layer encryption scheme. In the first-layer encryption, we propose to define a specialized access policy for each data attribute in the EHR, generate a secret share for every distinct role attribute, and reconstruct the secret to encrypt each data attribute, which ensures a fine-grained access control, saves much encryption time, and conceals the frequency of role attributes occurring in the EHR. In the second-layer encryption, we propose to preserve the privacy of role attributes and access policies used in the first-layer encryption. Specifically, we merge the first-layer access policies, add noise to the merged access policy, and encrypt the first-layer access policies under the noisy and merged access policy. Additionally, to take full advantage of the cloud server, we propose to let the cloud execute computationally intensive works on behalf of the data user without knowing any sensitive information. To preserve the access pattern(access frequency) of the data attributes in the EHR, we construct a blind data retrieving protocol. Furthermore, we show that our scheme can be easily extended to support search functionality. Finally, we conduct extensive security analyses and performance evaluations, which confirm the efficacy and efficiency of our schemes.

Our main contributions are summarized as follows:

- To the best of our knowledge, this is the first attempt to address the inference attack problem in the e-healthcare cloud system with fine-grained access control. Compared with the existing solutions, our scheme not only ensures novel functionalities, but also achieves higher efficiency on encryption, decryption, and role attribute revocation.
- We systematically construct a two-layer encryption scheme. The first-layer encryption ensures the fine-grained access control, saves much encryption time, and conceals the frequency of role

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2018.2790943, IEEE Transactions on Services Computing

attributes occurring in the EHR. The second-layer encryption enables the cloud to execute computationally intensive works on behalf of the data user, while preserving the privacy of access policies used in the first-layer encryption.

- We design a blind data retrieving protocol, which preserves the access pattern of data attributes in the EHR, and achieves high efficiency.
- We provide rigorous security analyses and conduct extensive experiments to confirm the efficacy and efficiency of our proposed schemes.

The rest of this paper is organized as follows. Section 2 presents the preliminaries. Section 3 formulates the problem. Section 4 demonstrates the secure constructions. Section 5 presents the security and privacy analysis. Section 6 demonstrates the efficiency of our proposed scheme. Section 7 reviews the related works. In Section 8, we conclude the paper.

2 Preliminaries

In this section, we briefly introduce the preliminaries, which will be used in this paper.

2.1 Data Attributes and Role Attributes

The data attributes refer to the data in the EHR data that require to be encrypted. The role attributes refer to the roles that the users should have in order to access and decrypt the encrypted data attributes. For example, if the EHR data contains the social security number, the liver health data, and the dental health data, then there are three data attributes in the EHR data, i.e., the social security number, the liver health data, and the dental health data. Before we encrypt these data attributes, we need to specify the role attributes that the accessors should have. For example, if we specify the social security number can be accessed by the hospital financial staff, the dental health data can be accessed by the dentist or the radiologist, and the liver health data can only be accessed by the internist, then there are four role attributes here, i.e., the hospital financial staff, the dentist, the radiologist, and the internist.

2.2 Access Policy

The access policy is defined over role attributes. For example, to ensure fine-grained access control over data D, the data owner would encrypt D with the following access policy-the accessors should have at least three role attributes from (a, b, c, d), where (a, b, c, d) denotes four distinct role attribute. Therefore, the accessors with less than three role attributes from (a, b, c, d) cannot decrypt the cipher-text of D.

Linear Secret Sharing Scheme Matrix The Linear Secret Sharing Scheme matrix (LSSS) [15] is commonly used to implement the access policy. Specifically, the LSSS defines its access policy with (M, ρ), where M is the secret share generating matrix, ρ is a function, and $\rho(i)$ maps the row M_i to an authorized role attribute. Assume M has m rows and n columns, to distribute a secret s among m role attributes, the LSSS randomly chooses a column vector $v = (s, r_2, \cdots, r_n)$, and generates the secret share $\lambda_i = (Mv)_i$ for each role attribute. Meanwhile, the LSSS has the linear reconstruction property. Specifically, for any authorized role attribute set γ that satisfies the access policy (M, ρ), there exists a constant vector w, where $\sum\limits_{\rho(i)\in\gamma} w_i \cdot M_i = (1, 0, \cdots 0)$. We can reconstruct the secret s with $s = \sum\limits_{\rho(i)\in\gamma} w_i \cdot \lambda_i$.

The LSSS matrix is appropriate to implement the access policy when encrypting data. However, it is not intuitive to use. To bridge the gap between the usability and implementation techniques of the access policy, an alternative way is to denote the access policy with some intuitive methods(e.g., access tree, threshold-tree-string) and then transfer the method to the LSSS matrix when encrypting data.

Threshold-gate Access Tree We denote $\mathcal{T}$ as an access tree with root r, and $\mathcal{T}_x$ as $\mathcal{T}$'s subtree rooted at node x. Therefore, $\mathcal{T}$ can be also denoted with $\mathcal{T}_r$. If a set of role attributes γ satisfy the access tree $\mathcal{T}_x$, then we define $\mathcal{T}_x(\gamma) = 1$. The $\mathcal{T}_x$ is computed recursively as follows, if x is not a leaf node, evaluate all x's child node x', and set $\mathcal{T}_x(\gamma) = 1$ if and only if at least t_x children of x return 1, where t_x is the threshold value for node x. If x is a leaf node, then $\mathcal{T}_x(\gamma) = 1$ if and only if $x \in \gamma$.

Threshold-tree-string The threshold-gate access tree can represent the access policy expressively. However, if not appropriately designed, the access tree will require abundant storage cost. To meet this challenge, the threshold-tree-string [16], which denotes the threshold-gate access tree with a single string, is proposed.

The threshold-tree-string is composed of many sub-strings, each sub-string is composed of some role attributes and a number, where the number means the threshold number of role attributes an authenticated accessor should have. Each sub-string corresponds to a sub-tree in the threshold-gate access tree.

For example, once the access policy is represented with the threshold-tree-string $\mathbb{L} = (a, b, c, d, e, 4)$, it means that the accessors should have at least 4 role attributes out of the 5-element role attribute set (a, b, c, d, e). Additionally, the threshold-tree-string $\mathbb{L}$ can also be easily transformed to the LSSS matrix (M, ρ) [16]. For easy description, we will use these two conceptions alternatively when regarding to the access policies.

2.3 Bilinear Map

Let G and G_1 denote two cyclic groups with a prime order p. Let g be the generator of G, and e be the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2018.2790943, IEEE Transactions on Services Computing

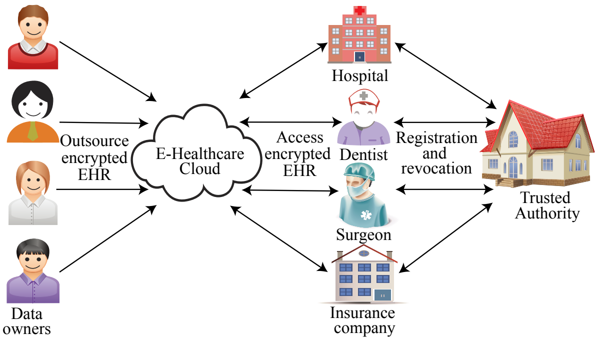

Fig. 2: Architecture of the e-healthcare cloud system.

bilinear map $e : G \times G \to G_1$. The bilinear map e will have the following three properties: 1) Bilinear: $\forall a, b \in \mathbb{Z}_p^*$, $e(g^a, g^b) = e(g, g)^{ab}$. 2) Non-degenerate: $e(g, g) \neq 1$. 3) Computable: bilinear map $e : G \times G \to G_1$ can be efficiently computed.

2.4 Paillier Encryption

Paillier encryption [17] is a public key cryptosystem with additive homomorphic properties. Let $E(a)$ denote the ciphertext after the Paillier encryption on a, and $D(E(a))$ denote the Paillier decryption on $E(a)$. $\forall a, b \in \mathbb{Z}_n$, we have the following properties:

1) $D(E(a) \cdot E(b) \bmod n^2) = a + b \bmod n$.
2) $D(E(a)^b \bmod n^2) = a \cdot b \bmod n$.

3 Problem Formulation

3.1 System Model

In our system model, four entities are involved, as shown in Fig. 2: they are the trusted authority, the data owners, the users, and the cloud. The trusted authority is responsible for user registration and revocation. The data owners are those who will outsource their EHR data to the cloud. To guarantee a fine-grained access control while preserving data privacy, the data owners encrypt their EHR data before outsourcing. To access this encrypted EHR data, the data user submits his role attributes to the cloud. Upon receiving the role attributes, the cloud retrieves the encrypted data and returns them to the data user. The data user further decrypts the ciphertexts, and obtains the authorized data attributes in the EHR with his role attributes.

3.2 Threat Model

We assume that the trusted authority and data owners are trusted. However, the cloud is not trusted, we treat it as 'curious but honest' [18], [19], [20], [21], [22]. Specifically, the cloud will follow our protocol, but it is very curious to deduce sensitive data from the EHR stored on it. Particularly, the cloud will try to collect the frequency of role attributes contained in the EHR, and the access frequency of data attributes in the EHR data. The cloud will also try to collect other useful background information to launch the inference attack, so that, he can deduce useful private data from the EHR data attributes even if they are encrypted. In this paper, we aim to defend the cloud from launching such inference attacks. Additionally, the data user can only access his authorized data attributes in the EHR, i.e., the data user's role attributes should satisfy the access policies of the accessed data attributes.

3.3 Design goals

Fine-gained access control: Data owners should specify the access policy for each data attribute in the EHR, so that the data user can only access and decrypt his authorized data attribute.
Efficiency: The data attributes encryption, decryption, and role attributes revocation should be executed efficiently.
Security: The encryption scheme should be secure under the security model formulated as follows:
Setup: The challenger generates the public keys and private keys, and sends the public keys to the adversary.
Phase 1: The adversary $\mathcal{A}$ queries the private keys for sets of role attributes $S_1, S_2, \cdots, S_{q1}$.
Challenge: $\mathcal{A}$ submits two equal length EHR data D_0 and D_1 to the challenger. Additionally, $\mathcal{A}$ submits a challenge access policy A^*, such that the queried $S_1, S_2, \cdots, S_{q1}$ do not satisfy A^*. The challenger flips a coin γ, encrypts D_γ under A^*, and returns the ciphertext CT^* to the adversary $\mathcal{A}$.
Phase 2: Phase 1 is repeated. The only restriction is the queried sets of attributes $S_{q1+1}, \cdots, S_q$ do not satisfy A^*.
Guess: $\mathcal{A}$ outputs a guess γ' for γ.

Definition 1. *Our scheme is secure if all probabilistic polynomial time adversaries have at most a negligible advantage $Pr[\gamma = \gamma'] - 1/2$ in the above game.*

Privacy: Our proposed scheme should control the privacy protection to a specific level. We measure the privacy disclosure of our scheme by the attacker's confidence in the success of an attack [23], [24].

1) ϵ-access-policy-privacy: Given there are n role attributes contained in the access policy of the EHR data, and we add n' noisy role attributes to the access policy, then the attacker's confidence in the success of an attack translates into the probability of finding out the true positive.

Definition 2. *our scheme achieves ϵ-access-policy-privacy, if and only if for any EHR data D_j with pre-defined privacy degree ϵ_j, the following inequality holds: $P(n'_j|(n_j + n'_j), \hat{A}) \geq \epsilon_j$, where $\hat{A}$ denotes the attacker's auxiliary information.*

2) ϵ'-access-pattern-privacy: Given the data user wants to retrieve n data attributes from the cloud, and he requests n' additional data attributes from the

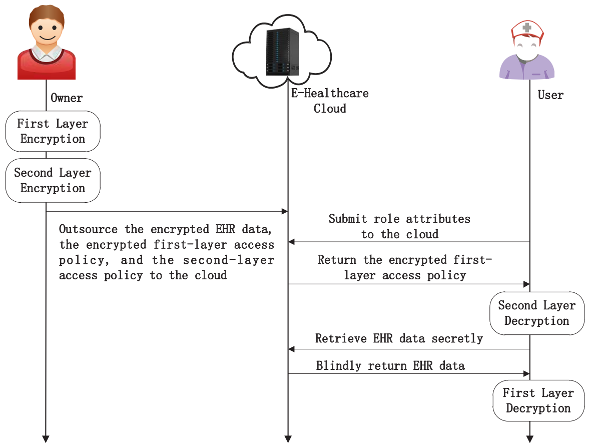
Owner
E-Healthcare Cloud
User
First Layer Encryption
Second Layer Encryption
Outsource the encrypted EHR data, the encrypted first-layer access policy, and the second-layer access policy to the cloud
Submit role attributes to the cloud
Return the encrypted first-layer access policy
Second Layer Decryption
Retrieve EHR data secretly
Blindly return EHR data
First Layer Decryption

Fig. 3: A brief overview of the secure constructions

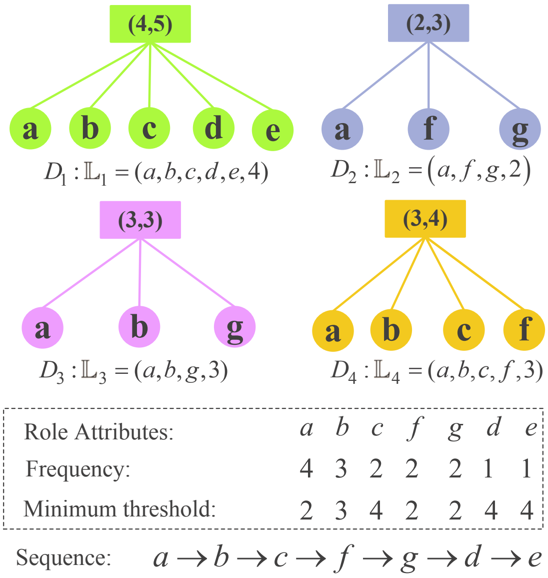
(4,5)
a b c d e
$D_1 : \mathbb{L}_1 = (a,b,c,d,e,4)$
(2,3)
a f g
$D_2 : \mathbb{L}_2 = (a,f,g,2)$
(3,3)
a b g
$D_3 : \mathbb{L}_3 = (a,b,g,3)$
(3,4)
a b c f
$D_4 : \mathbb{L}_4 = (a,b,c,f,3)$
Role Attributes: a b c f g d e
Frequency: 4 3 2 2 2 1 1
Minimum threshold: 2 3 4 2 2 4 4
Sequence: $a \rightarrow b \rightarrow c \rightarrow f \rightarrow g \rightarrow d \rightarrow e$

Fig. 4: Example of generating the secret share.

cloud, then the attacker's confidence in the success of an attack translates into the probability of finding out the true positive.

Definition 3. *our scheme achieves ϵ'-access-pattern-privacy, if and only if for any jth request, issued by the data user with pre-defined privacy degree ϵ'_j, the following inequality holds: $P(n'_j|(n_j + n'_j), \hat{A}') \geq \epsilon'_j$, where $\hat{A}'$ denotes the attacker's auxiliary information.*

4 Secure Constructions

In this section, we elaborate on how to achieve the efficient and inference attack-resistant e-healthcare cloud system with fine-grained access control. Fig. 3 demonstrates a brief overview of the secure constructions. As we can see, at the beginning, the data owner conducts the first-layer encryption on each data attribute in the EHR with the attribute based encryption algorithms. Then, to prevent the attacker from knowing the access policies used in the first-layer encryption, the data owner conceals these access policy, and conducts the second-layer encryption. After that, the data owner outsources the encrypted EHR data, the encrypted first-layer access policy, and the second-layer access policy to the cloud. Once the data user wants to retrieve data stored on the cloud, he submits his role attributes to the cloud, and the cloud will return the encrypted first-layer access policy. Upon receiving the ciphertext of the first-layer access policy, the data user performs the second-layer decryption, and retrieves his authorized data attributes from the cloud. With our design, the data retrieving process preserves the access pattern privacy. Finally, the data user conducts the first-layer decryption and obtains the authorized data attributes in the EHR with his role attributes. In what follows, we will demonstrate the secure constructions step by step.

For easy description hereafter, we introduce the following definition.

Definition 4. *We define the role attribute set of the EHR data D as $A = A_1 \cup A_2 \cdots \cup A_d$, T_j as the threshold of the access policy (threshold-tree-string) $\mathbb{L}_j$, and T_{jk} as the threshold of the kth sub-access policy of $\mathbb{L}_j$, i.e., $\mathbb{L}_{jk}$, where $T_j = 1 + \sum_{\mathbb{L}_{jk} \in \mathbb{L}_j} (T_{jk} - 1)$. If f data attributes' access policy contains the role attribute x, then we define f as the frequency of x. We further define the minimum threshold of the access policies involving x as the minimum threshold of x, and the corresponding access policy as x's minimum access policy.*

As shown in Fig. 4, the EHR data has four data attributes, i.e., (D_1, D_2, D_3, D_4), the corresponding access policies of these data attributes are $\{\mathbb{L}_1 = (a,b,c,d,e,4), \mathbb{L}_2 = (a,f,g,2), \mathbb{L}_3 = (a,b,g,3), \mathbb{L}_4 = (a,b,c,f,3)\}$. The tree shown above is the threshold-gate access tree, and the following string(e.g., $\mathbb{L}_1 = (a,b,c,d,e,4)$) is the equivalent threshold-tree-string. Both the threshold-gate access tree and threshold-tree-string denotes the access policies of the data attributes. For example, for the data attribute D_1's access policy $\mathbb{L}_1 = (a,b,c,d,e,4)$, the accessors should have at least 4 role attributes out of the 5-element role attribute set (a,b,c,d,e).

Obviously, the role attribute sets corresponding to (D_1, D_2, D_3, D_4) are $A_1 = (a,b,c,d,e), A_2 = (a,f,g), A_3 = (a,b,g), A_4 = (a,b,c,f)$, respectively. Since the role attribute a occurs in $(\mathbb{L}_1, \mathbb{L}_2, \mathbb{L}_3, \mathbb{L}_4)$, we call a's frequency is 4. Additionally, the thresholds of $(\mathbb{L}_1, \mathbb{L}_2, \mathbb{L}_3, \mathbb{L}_4)$ are $(T_1 = 4, T_2 = 2, T_3 = 3, T_4 = 3)$, respectively. Therefore, the minimum threshold for a is 2, and T_2 is a's minimum access policy.

4.1 Distributing keys and parameters

In our system, the trusted authority (TA) is responsible for distributing keys and parameters. Initially, TA randomly generates the primary key $mk = (\alpha, \beta \in \mathbb{Z}_p)$. Then he sets the public key as $pk =$

$(g^{\alpha}, e(g,g)^{\alpha\beta})$. Meanwhile, TA is also responsible for distributing public parameters for the system. Assume that the size of the whole role attribute set $\mathbb{A}$ is n, TA randomly chooses n distinct group members: $(h_{a1}, h_{a2}, \cdots, h_{an} \in G)$, and sets them as public parameters. Once a user U_i registers with TA, TA will authorize a role attribute set $\tilde{\mathbb{A}}_i$ to U_i, and generate the secret key $\hat{C}_{i1} = g^{\alpha r_i} g^{\alpha\beta}$, and public parameters $\{\hat{C}_{i2} = g^{r_i}, \{H_x = h_x^{r_i}\}_{\forall x \in \tilde{\mathbb{A}}_i}\}$ for U_i, where r_i is randomly chosen from $\mathbb{Z}_p$.

4.2 First-layer Encryption

To achieve the fine-grained access control over the EHR data D, before outsoursing D to the cloud, the data owner first specifies a specialized role attribute set A_j, and a corresponding access policy $\mathbb{L}_j$ for each data attribute D_j in D, where $j \in [1, d]$. Then he encrypts D with our first-layer encryption, which is formulated in the following three steps.

1. Generate a secret share λ_x for each role attribute x in A. Note that, though the attribute x would occur in the access policies of many data attributes, we only generate one secret share for x, which not only conceals the frequency of x, but also saves much computation cost. The secret share generation is achieved as follows:

1) The data owner adjusts the access policies of data attributes with the following principle. For any $\mathbb{L}_j$ and $\mathbb{L}_{j'}$, either T_j or $T_{j'}$ is greater than $|A_j \cap A_{j'}|$, where $|A_j \cap A_{j'}|$ denotes the size of $A_j \cap A_{j'}$.
2) Rank role attributes in descending order based on their frequencies, and mark their minimum thresholds.
3) Iteratively generate or compute the secret share for each role attribute. The principle is described as follows, for the role attribute x, if the number of role attributes that have set a secret share in x's minimum access policy is less than x's minimum threshold, then choose a random value as x's secret share. Otherwise, with the role attributes value in in x's minimum access policy, use Lagrange interpolation theorem to deduce the secret sharing formula, and compute the secret share for x and other role attributes in x's minimum access policy with that formula.

For the example demonstrated in Fig. 4, the process of generating the secret share is illustrated as follows. 1) we check these access policies with step 1, and find that no access policy needs to be adjusted. 2) we rank these role attributes according to their frequencies: $(a, 4; b, 3; c, 2; f, 2; g, 2; d, 1; e, 1)$, and mark their minimum threshold: $(a, 2; b, 3; c, 4; f, 2; g, 2; d, 4; e, 4)$. 3) we set the value of a, b, c, d by choosing random numbers, and use Lagrange interpolation theorem to deduce the secret sharing formula for D_1, D_2, D_3, D_4, and compute the value of f, g, e with that formula. Therefore, the sequence of generating the secret share is $(a \to b \to c \to f \to g \to d \to e)$.

2. For all the distinct role attribute x in the role attribute set A, generate a random r_x for x, and compute the ciphertext $C'_x = g^{\alpha\lambda_x} h_x^{r_x}, C''_x = g^{r_x}$.

3. Transform $\mathbb{L}_j$ to (M_j, ρ_j) [16], compute the ciphertext for each data attribute D_j with the following three steps.

1) Compute a weight vector w, such that: $\sum_{1 \le k \le |M|} w_k \cdot M_k = (1, 0, \cdots 0)$.
2) Compute the secret s_j for each data attribute D_j: $s_j = \sum_{\rho_j(k) \in A_j} w_k \cdot \lambda_{\rho_j(k)}$.
3) Generate the ciphertext for D_j: $C_{j0} = D_j \cdot e(g,g)^{\alpha\beta s_j}$, $C_{j1} = g^{s_j}$.

Therefore, the ciphertexts of D after the first-layer encryption are $\mathcal{C} = \Big\{\{C'_x, C''_x\}_{\forall x \in A}, \{C_{j0}, C_{j1}\}_{j \in [1,d]}\Big\}$.

4.3 Second-layer Encryption

With the first-layer encryption, we can achieve the efficient and fine-grained access control. However, the cloud can still launch the inference attack. Specifically, with some background knowledge, the cloud can deduce the content of the EHR by observing the role attributes, access policies, and the access frequency of data attributes in the EHR. To defend the cloud against knowing the actual role attributes and access policies, we propose to merge all the role attributes and access policies of data attributes in the EHR, add noise to the merged attributes and access policy, and encrypt the access policies used in the first-layer encryption under the noisy and merged access policy.

4.3.1 *Merging access policies*

To prevent the cloud from knowing the access policy of each data attribute, we propose loosely merging all these access policies, so that, if a set of role attributes satisfy any of the access policies used in the first-layer encryption, then these attributes also satisfy the merged access policy.

Given the role attribute set A, and the access policies (denoted by threshold-tree-strings, the LSSS access policy can be easily transformed to threshold-tree-string, and vise verse [16]) of all data attributes, i.e., $\{\mathbb{L}_j\}_{j \in [1,d]}$, the merging process is achieved in two phases. In the first phase, we initialize the threshold-tree-string set $\mathbb{L}$, by adding all these threshold-tree-strings, i.e., $\mathbb{L} = \{\mathbb{L}_1, \mathbb{L}_2, \cdots, \mathbb{L}_d\}$. In the second phase, we merge all these threshold-tree-strings. The key idea of merging is illustrated as follows: if two threshold-tree-strings have common attributes, then we merge their role attributes, and set their minimum threshold as the threshold of the merged threshold-tree-string. The merge process ends when all the role attributes in $\mathbb{L}$ are distinct.

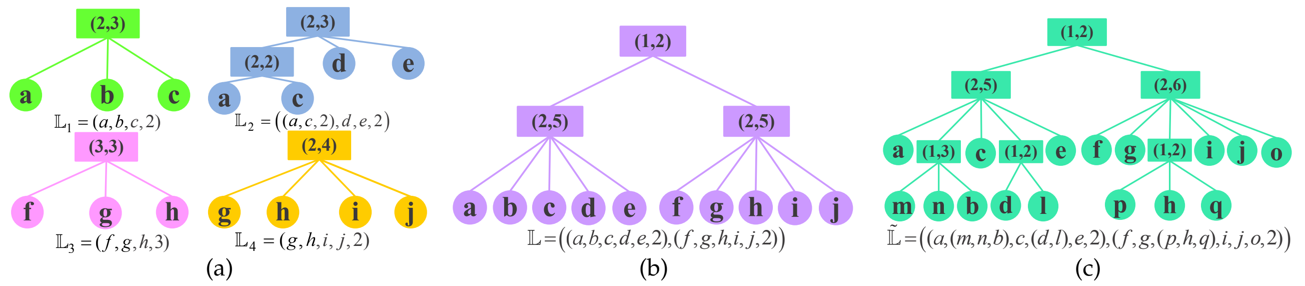
(2,3)
a
b
c
$\mathbb{L}_1 = (a,b,c,2)$
(2,3)
(2,2)
d
e
a
c
$\mathbb{L}_2 = ((a,c,2),d,e,2)$
(3,3)
f
g
h
$\mathbb{L}_3 = (f,g,h,3)$
(2,4)
g
h
i
j
$\mathbb{L}_4 = (g,h,i,j,2)$
(a)
(1,2)
(2,5)
(2,5)
a b c d e f g h i j
$\mathbb{L}=((a,b,c,d,e,2),(f,g,h,i,j,2))$
(b)
(1,2)
(2,5)
(2,6)
a (1,3) c (1,2) e f g (1,2) i j o
m n b d l p h q
$\tilde{\mathbb{L}}=((a,(m,n,b),c,(d,l),e,2),(f,g,(p,h,q),i,j,o,2))$
(c)

Fig. 5: Example of merging on a threshold-gate access tree and its equivalent threshold-tree-strings. (a) Access policies for $\{D_1, D_2, D_3, D_4\}$; (b) Merged access policy for D; (c) Noisy access policy for D.

Fig. 5 shows an example of merging. For the data attributes $\{D_1, D_2, D_3, D_4\}$, their corresponding access policies $\{\mathbb{L}_1, \mathbb{L}_2, \mathbb{L}_3, \mathbb{L}_4\}$ are illustrated in Fig. 5(a). The merged access policy $\mathbb{L}$ is shown in Fig. 5(b). As we can see, the EHR data D is composed of 4 data attributes, each attribute has its own access policy, if they are exposed to the cloud separately, the cloud would easily deduce their context from the role attributes of the accessors. By merging them together, the cloud does not know the relationship between the role attributes and the data attributes.

4.3.2 Adding noise to the merged access policy

By merging the role attributes and access policies of all data attributes in an EHR, the cloud does not know the accurate role attributes or access policy of a specific data attribute, but he can deduce that these role attributes are involved in the access policy of the EHR, therefore, the cloud can still deduce sensitive information about the data attributes in the EHR. To solve this problem, we propose adding noise to the merged role attributes and access policy. In this way, the cloud does not know whether a specific role attribute is actually involved in the access policy of the EHR, or just a noisy attribute.

We can add two types of noise here. The first type of noise can be any attributes that do not occur in the attribute set A, therefore we can choose any noise attributes according to the system requirement, and randomly distribute them among the threshold-tree-strings. The second noise can be any threshold-tree-strings, where role attributes in the threshold-tree-strings should have no intersection with those in A. Denote $\tilde{\mathbb{L}}$ as the threshold-tree-strings with noisy role attributes, and $(\tilde{M}, \tilde{\rho})$ is transformed from $\tilde{\mathbb{L}}$ with the method proposed in [16], we should ensure that, the number of columns in $\tilde{M}$ is greater than the number of role attributes in the original role attribute set A.

Fig. 5(c) shows an example of adding noise to the merged access policy.

Note that, to achieve the pre-defined ϵ-access-policy-privacy, for any EHR data D_j with pre-defined privacy degree ϵ_j, let n_j be the distinct role attributes used in the first-layer encryption, then the data owner chooses n'_j noisy role attributes that satisfies $n'_j/n_j \geq \epsilon'_j$.

4.3.3 Encryption

The key idea of the second-layer encryption is to encrypt the access policies used in the first-layer encryption under the noisy and merged access policy, so that only the authorized data users can decrypt them. Here, since the threshold-tree-string $\mathbb{L}$ is more expressive and saves storage cost, instead of encrypting (M, ρ), we choose to encrypt $\mathbb{L}$. Given the first-layer access policies: $\{\mathbb{L}_j\}_{j\in[1,d]}$ of D, the second-layer encryption is formulated with the following four steps.

1) $\forall x \in \hat{A}$, where $\hat{A}$ denotes the noisy attribute set, generate a secret share λ_x for the attribute x, choose a random r_x, and compute $C'_x = g^{\alpha\lambda_x} h_x^{r_x}, C''_x = g^{r_x}$.
2) Compute w: $\sum\limits_{1\leq k\leq|\tilde{M}|} w_k \cdot \tilde{M}_k = (1, 0, \cdots 0)$.
3) Compute the secret $\tilde{s} = \sum\limits_{\tilde{\rho}(k)\in A\cup\hat{A}} w_k \cdot \lambda_{\tilde{\rho}(k)}$.
4) Concatenate all the first-layer access policies: $\mathcal{L} = \mathbb{L}_1||\mathbb{L}_2||\cdots||\mathbb{L}_d$, and encrypt $\mathcal{L}$ as $\hat{\mathcal{L}} = \mathcal{L} \cdot e(g,g)^{\alpha\beta\tilde{s}}$.

After the two-layer encryption, the ciphertexts outsourced to the cloud are: $\left\{\{C'_x, C''_x\}_{\forall x\in A\cup\hat{A}}, \{C_{j0}, C_{j1}\}_{j\in[1,d]}, C, \hat{\mathcal{L}}\right\}$, where $C = g^{\tilde{s}}$. Note that, during the whole process described above, we do not need to regenerate ciphertext for the attributes in the original role attribute set A. Instead, we only need to generate the ciphertext for the attributes in the noisy role attribute set $\hat{A}$, which also saves much computation overhead.

4.4 Returning Encrypted Access Policies

Upon receiving a data user's access request, instead of returning all the data stored on the cloud and letting the data user decrypt a huge amount of EHR data, the cloud needs to filter out the EHR data that do not match the user's role attributes, and only return the encrypted access policies $\hat{\mathcal{L}}$ that match the user's role attributes. Assume the user U_i submits his role attributes $\tilde{\mathbb{A}}_i$, and parameters $\{g^{r_i}, \{H_x = h_x^{r_i}\}_{\forall x\in\tilde{\mathbb{A}}_i}\}$

to the cloud. If U_i's role attributes satisfy an EHR's second-layer access policy, the cloud will return the encrypted access policy $\hat{\mathcal{L}}$ of that EHR to U_i. Otherwise, U_i is regarded as an unauthorized user to all the data attributes in that EHR, and therefore, the cloud will refuse to return that EHR data to U_i. Note that, to save computation cost for U_i, we let the cloud compute $e(g,g)^{\alpha r_i \tilde{s}}$ on behalf of U_i with the following equation:

$$\begin{aligned}
&\prod_{\tilde{\rho}(k)\in\tilde{\mathbb{A}}_i} \left(e({C'}_{\tilde{\rho}(k)}, \hat{C}_{i2})/e({C''}_{\tilde{\rho}(k)}, H_{\tilde{\rho}(k)})\right)^{w_k} \\
&= \prod_{\tilde{\rho}(k)\in\tilde{\mathbb{A}}_i} \left(e(g^{\alpha\lambda_{\tilde{\rho}(k)}} h_{\tilde{\rho}(k)}^{r_{\tilde{\rho}(k)}}, g^{r_i})/e(g^{r_{\tilde{\rho}(k)}}, h_{\tilde{\rho}(k)}^{r_i})\right)^{w_k} \\
&= \prod_{\tilde{\rho}(k)\in\tilde{\mathbb{A}}_i} e(g,g)^{\alpha r_i \lambda_{\tilde{\rho}(k)} w_k} \\
&= e(g,g)^{\alpha r_i \tilde{s}}
\end{aligned} \tag{1}$$

Therefore, if U_i is an authorized user for the EHR, he will receive $\hat{\mathcal{L}}$, C, and $e(g,g)^{\alpha r_i \tilde{s}}$ from the cloud.

4.5 Second-layer Decryption

When the user U_i receives $\hat{\mathcal{L}}$, C, and $e(g,g)^{\alpha r_i \tilde{s}}$, U_i computes the decryption key for $\hat{\mathcal{L}}$ with the following equation:

$$\begin{aligned}
&e(C, \hat{C}_{i1}) \Big/ e(g,g)^{\alpha r_i \tilde{s}} \\
&= e(g^{\tilde{s}}, g^{\alpha r_i} g^{\alpha\beta}) \Big/ e(g,g)^{\alpha r_i \tilde{s}} \\
&= e(g,g)^{\alpha\beta\tilde{s}}
\end{aligned} \tag{2}$$

Then U_i decrypts and obtains $\mathcal{L} = \mathbb{L}_1||\mathbb{L}_2||\cdots||\mathbb{L}_d$. According to $\mathcal{L}$, U_i easily knows the data attributes that he is authorized to access.

4.6 Retrieving Data with Access Pattern Privacy Preserved

To preserve the access frequency of a specific data attribute, we propose to design a blind data retrieving protocol.

Assume the data user U_i has the authorization to t' data attributes of an EHR, we denote them as $D' = \{D_1, D_2, \cdots, D_{t'}\}$. The blind retrieving method is achieved in the following three steps.

First, U_i chooses a random public key pk_i, and prepares $\hat{t}$ ciphertexts, where t' ciphertexts are $E(pk_i, 1)$, and the other $(\hat{t} - t')$ ciphertexts are $E(pk_i, 0)$. Here $E(\cdot)$ denotes the homomorphic Paillier encryption. Due to the its randomness, the encryption of 1 and 0 would be different each time. Note that, to achieve the pre-defined ϵ'-access-pattern-privacy, for any jth data retrieving, the data user requests $\hat{t}_j - t'_j$ additional data attributes, that satisfies $(\hat{t}_j - t'_j)/\hat{t}_j \geq \epsilon'_j$.

Second, assume the ciphertext of D_j after the first-layer encryption is $C_j = C_{j0}||C_{j1}$, U_i specifies the cloud to compute $E'_j = E(pk_i, 1)^{C_j}$ for the data attributes in D', and compute $E''_j = E(pk_i, 0)^{C_j}$ for $(\hat{t} - t')$ data attributes not in D'. U_i would also specify the cloud to compute $E'_j \cdot E''_j$.

Third, the cloud executes the above method and returns the results.

Since U_i also knows which role attributes are required for the first-layer decryption after obtaining the access policies $\mathcal{L}$, U_i would also use the above method to access the corresponding role attributes data, i.e., $\{C'_x, C''_x\}_{\forall x \in A_j}$.

Now, we give an example to illustrate the above process. Assume U_i has the privilege of decrypting D_1. He first prepares three ciphertexts $E(pk_i, 1)$, $E(pk_i, 0)$, $E(pk_i, 0)$. Then U_i specifies the cloud to compute $E'_1 = E(pk_i, 1)^{C_1} = E(pk_i, C_1)$, $E''_2 = E(pk_i, 0)^{C_2} = E(pk_i, 0)$, $E''_3 = E(pk_i, 0)^{C_3} = E(pk_i, 0)$. Additionally, U_i specifies the cloud to multiply the three computation results together, i.e., $E(pk_i, C_1) \cdot E(pk_i, 0) \cdot E(pk_i, 0) = E(pk_i, C_1)$. After decryption, U_i obtains C_1. As we can see, during the whole process, the cloud only conducts computation on random ciphertexts, and therefore, he does not know which data attributes are actually returned. Assume D_1 needs $\{C'_{j1}, C''_{j1}, C'_{j2}, C''_{j2}\}$ for decryption, U_i will use the similar method to obtain $\{C'_{j1}, C''_{j1}, C'_{j2}, C''_{j2}\}$.

4.7 First-layer Decryption

After obtaining the ciphertext $C_{j0}, C_{j1}, \{{C'}_x, {C''}_x\}_{\forall x \in A_j}$, the user U_i first converts $\mathcal{L}_j$ to (M_j, ρ_j), then U_i uses his secret key $\hat{C}_{i1}$ and parameters $\{\hat{C}_{i2}, \{H_x\}_{\forall x \in \tilde{\mathbb{A}}_i}\}$ to decrypt the returned data attributes in the EHR. The process is formulated with the following two steps:

First, U_i computes

$$\begin{aligned}
&\prod_{\rho_i(k)\in\tilde{\mathbb{A}}_i} \left(e({C'}_{\rho_i(k)}, \hat{C}_{i2})/e({C''}_{\rho_i(k)}, H_{\rho_i(k)})\right)^{w_k} \\
&= \prod_{\rho_i(k)\in\tilde{\mathbb{A}}_i} \left(e(g^{\alpha\lambda_{\rho_i(k)}} h_{\rho_i(k)}^{r_{\rho_i(k)}}, g^{r_i})/e(g^{r_{\rho_i(k)}}, h_{\rho_i(k)}^{r_i})\right)^{w_k} \\
&= \prod_{\rho_i(k)\in\tilde{\mathbb{A}}_i} e(g,g)^{\alpha r_i \lambda_{\rho_i(k)} w_k} \\
&= e(g,g)^{\alpha r_i s_j}
\end{aligned} \tag{3}$$

Second, U_i decrypts and obtains D_j with the following equation.

$$\begin{aligned}
&C_{j0} \Big/ \left(e(C_{j1}, \hat{C}_{i1})/e(g,g)^{\alpha r_i s_j}\right) \\
&= D_j \cdot e(g,g)^{\alpha\beta s_j} \Big/ \left(e(g,g)^{\alpha r_i s_j + \alpha\beta s_j}/e(g,g)^{\alpha r_i s_j}\right) \\
&= D_j
\end{aligned} \tag{4}$$

4.8 Revoking Role Attributes

In conventional schemes, when a data owner wants to revoke several role attributes, say A', the data owner needs to update the secret for role attributes in A', and re-generate secret shares and ciphertexts for all the role attributes involved in the affected data attributes. Different from these schemes, we only need to update very few secret shares in the distinct role attribute set $A - A'$. Specifically, we need to update the ciphertext $\{C'_x, C''_x\}$ for the affected role attributes, $\{s_j, C_{j0}, C_{j1}\}$ for the affected data attributes in the first-layer encryption, and $\hat{\mathcal{L}}$ in the second-layer encryption.

We observe that, when the data attributes share very few repeated role attributes in an EHR data, then we only need to update secret shares for very few role attributes. When the data attributes share many repeated role attributes in an EHR data, though we need to update the secret shares for some role attributes, we can still outperform the conventional schemes. The fundamental reason is that, even if only one role attribute needs to be updated, conventional schemes have to update the secret share for all the role attributes of all the affected data attributes.

4.9 Extension and Discussion

4.9.1 *Achieving Search Functionality*

To ensure the search functionality on the EHR data, we can modify our scheme as follows, assume the data owner provides the keyword $\tilde{w}$ for an EHR data D, he encrypts $\tilde{w}$ as $C_{\tilde{w}} = g^{\tilde{s}\tilde{w}}$. Once a user wants to search $\tilde{w}$, he generates the trapdoor $T_{\tilde{w}} = g^{r_i \tilde{w}}$, then the cloud can determine whether D contains the searched keyword by checking the equality of the following equation:

$$e(C_{\tilde{w}}, \hat{C}_{i2}) = e(g,g)^{\tilde{s}\tilde{w}r_i} = e(T_{\tilde{w}}, C)$$

4.9.2 *Impact of the Frequency of Repeated Role Attributes*

Conventional attribute-based encryption schemes would randomly choose a secret for each data attribute, and generate the secret share for the role attributes in each data attribute. Different from these schemes, we reverse this process. Specifically, we first generate the secret share for each distinct role attribute, and then reconstruct the secret for each data attribute. This design will benefit our scheme from two aspects. First, for security concerns, since we conceal the frequency of these role attributes in the access policies of data attributes in the EHR, we can defend the cloud from doing the inference attack. Second, for efficiency concerns, since different data attributes will share many role attributes in their access policies (the frequency of repeated role attributes is high), we can save much computation cost for the expensive exponential operation. Note that, the higher frequency of the role attributes occur in the EHR, the more computation cost can be saved. Additionally, we observe that, these frequently repeated role attributes would contribute to the reconstruction of secrets for many data attributes, to enhance the security of our system, we propose to select a relatively longer secret key for the more frequently repeated role attributes.

5 Security and Privacy Analysis

In this section, we analyze the security and privacy of our proposed scheme, and show that the security and privacy goals have been achieved. We first prove that the two-layer encryption scheme is secure. For brief presentation, we reduce our security to the previous work [25]. Then we analyze the privacy of our proposed scheme.

5.1 Security Analysis

Theorem 1. *If the expressive ciphertext-policy attribute-based encryption is secure in the security game of [25], then our two-layer encryption scheme is secure in the security game defined in Section 3.*

Proof: Assume a probabilistic polynomial-time adversary $\mathcal{A}$ has a non-negligible advantage ϵ against our scheme in the security game defined in Section 3. Then we can build a simulator $\mathcal{B}$ that plays the security game in [25] with advantage $\epsilon/2$. The goal of $\mathcal{B}$ is to win the game by interacting with $\mathcal{A}$.

Setup: The challenger $\mathcal{C}$ sends the public key PK : $\{g, e(g,g)^\alpha, g^a, h_1, \cdots, h_U\}$ to the simulator $\mathcal{B}$, where U is the size of the whole role attribute set in the system. Then $\mathcal{B}$ transfers the public key PK to the adversary $\mathcal{A}$.

Phase 1: $\mathcal{A}$ queries the private keys for sets of role attributes S, $\mathcal{B}$ transfers S to $\mathcal{C}$. $\mathcal{C}$ returns $\{K = g^\alpha g^{at}, L = g^t, \{K_x = h_x^t\}_{\forall x \in S}\}$ to $\mathcal{B}$. $\mathcal{B}$ transfers them to $\mathcal{A}$. Note that, for easy description, we use h_x to denote $h_{id(x)}$, where $id(x)$ denotes the ID of x in the whole role attribute set.

Challenge: $\mathcal{A}$ submits two equal length EHR data D_0 and D_1, and an access policy A^* to $\mathcal{B}$. The restriction here is that all the previously queried S do not satisfy A^*. Then $\mathcal{B}$ transfers these data to the challenger $\mathcal{C}$. $\mathcal{C}$ flips a coin μ, encrypts D_μ as: $CT^*_\mu = \{C^* = D_\mu e(g,g)^{\alpha s}, C^{**} = g^s, \{C^*_x = g^{a\lambda_x} h_x^{-r^*_x}, D^*_x = g^{r^*_x}\}_{\forall x \in A^*}\}$, and sends CT^*_μ to $\mathcal{B}$. Upon receiving CT^*_μ, $\mathcal{B}$ also flips a coin γ, computes CT_γ based on CT^*_μ, i.e., $CT_\gamma = \{C_\gamma = C^{**}, \hat{\mathcal{L}}_\gamma = \mathcal{L}_\gamma C^*/D_\gamma, \{C_{\gamma j0} = D_{\gamma j}(C^*/D_\gamma)^{r_j}, C_{\gamma j1} = (C^{**})^{r_j}\}_{j \in [1,d]}, \{C'_x = C^*_x, C''_x = 1/D^*_x\}_{\forall x \in A^*}\}$, where r_j is randomly generated, and sends CT_γ to $\mathcal{A}$.

Phase 2: Phase 1 is repeated. The only restriction is that the queried sets of attributes S do not satisfy A^*.

Guess: $\mathcal{A}$ outputs a guess γ' for γ. If $\gamma' = \gamma$, $\mathcal{B}$ outputs $\mu' = \gamma'$, otherwise, $\mathcal{B}$ outputs $\mu' = 1 \oplus \gamma'$.

In the case that $\mu \neq \gamma$, the adversary $\mathcal{A}$ obtains no information about D_γ, thus, $\Pr[\mu' = \mu | \mu \neq \gamma] = \frac{1}{2}$.

In the case that $\mu = \gamma$, $\mathcal{A}$ sees an encryption of D_γ, since the adversary advantage is ϵ, therefore, $\Pr[\mu' = \mu | \mu = \gamma] = \frac{1}{2} + \varepsilon$.

Therefore, the advantage of $\mathcal{B}$ in the security game of [25] is $\frac{1}{2}\Pr[\mu' = \mu | \mu \neq \gamma] + \frac{1}{2}\Pr[\mu' = \mu | \mu = \gamma] = \frac{\varepsilon}{2}$. □

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2018.2790943, IEEE Transactions on Services Computing

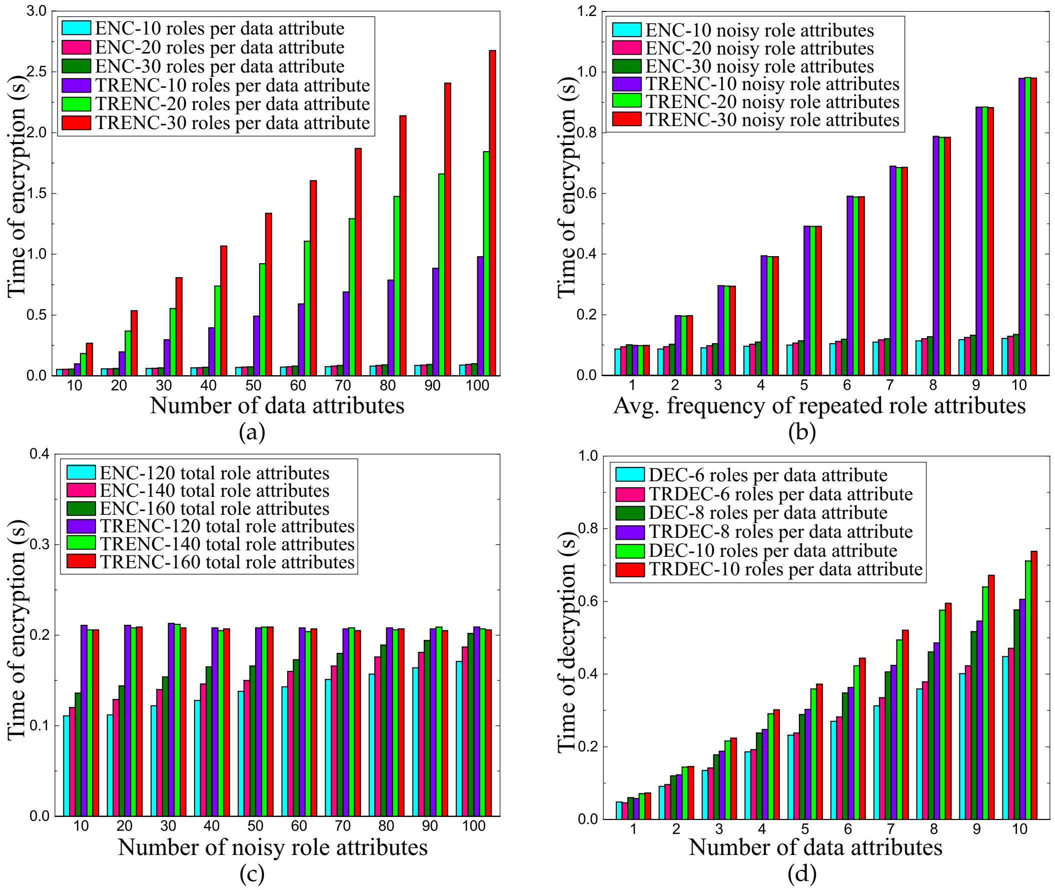

Fig. 6: Time cost of encryption and decryption.

5.2 Privacy Analysis

Access policy privacy: In our scheme, with the first-layer encryption, we conceal the data attributes in the EHR, while with the second-layer encryption, we conceal the role attributes and access policies used in the first-layer encryption. Therefore, what exposed to the attack is the $n + n'$ role attributes used in the second-layer encryption. Recall that, for any EHR data D_j with pre-defined privacy degree ϵ_j, let n_j be the distinct role attributes used in the first-layer encryption, the data owner chooses n'_j noisy role attributes that satisfies $n'_j/(n_j+n'_j) \geq \epsilon_j$. Therefore, for any EHR data D_j with pre-defined privacy degree ϵ_j, the attacker cannot have a confidence in the success of attack higher than ϵ_j. Therefore, our scheme achieves the ϵ-access-policy-privacy.

Access Pattern privacy: In our scheme, to preserve the access pattern(access frequency) of data attributes, the data user adds n' additional data attributes in his request. Additionally, we adopt the Paillier encryption scheme to encrypt the retrieved data. Due to the homomorphic property of Paillier encryption, during the whole process of data retrieval, the cloud only conducts homomorphic computation on random Paillier ciphertexts. Because the Paillier encryption scheme is semantically secure, the cloud cannot know anything from the ciphertext of the data attributes during the retrieving process. Therefore, what exposed to the attack is the $n + n'$ data attributes. Recall that, to achieve the pre-defined ϵ'-access-pattern-privacy, for any jth data retrieving, let $\hat{t}_j$ denote the number of total requested data attributes, the data user requests $\hat{t}_j - t'_j$ additional data attributes, that satisfies $(\hat{t}_j - t'_j)/\hat{t}_j \geq \epsilon'_j$. Therefore, for any jth request, issued by the data user with pre-defined privacy degree ϵ'_j, the attacker cannot have a confidence in the success of attack higher than ϵ'_j. Therefore, our scheme achieves the ϵ'-access-pattern-privacy.

6 Performance Evaluation

6.1 Evaluation Settings

The experiment programs are coded using Python programming language on a PC with 3.0GHZ Pentium Dual Core CPU and 2GB memory. We adopt the cryptographic framework and settings proposed in [26], and implement all necessary routines. We also make a thorough comparison with the state-of-the-art TR-MABE [13].

6.2 Evaluation Results

Fig. 6 shows the time cost of encrypting and decrypting data attributes in the EHR. We use ENC

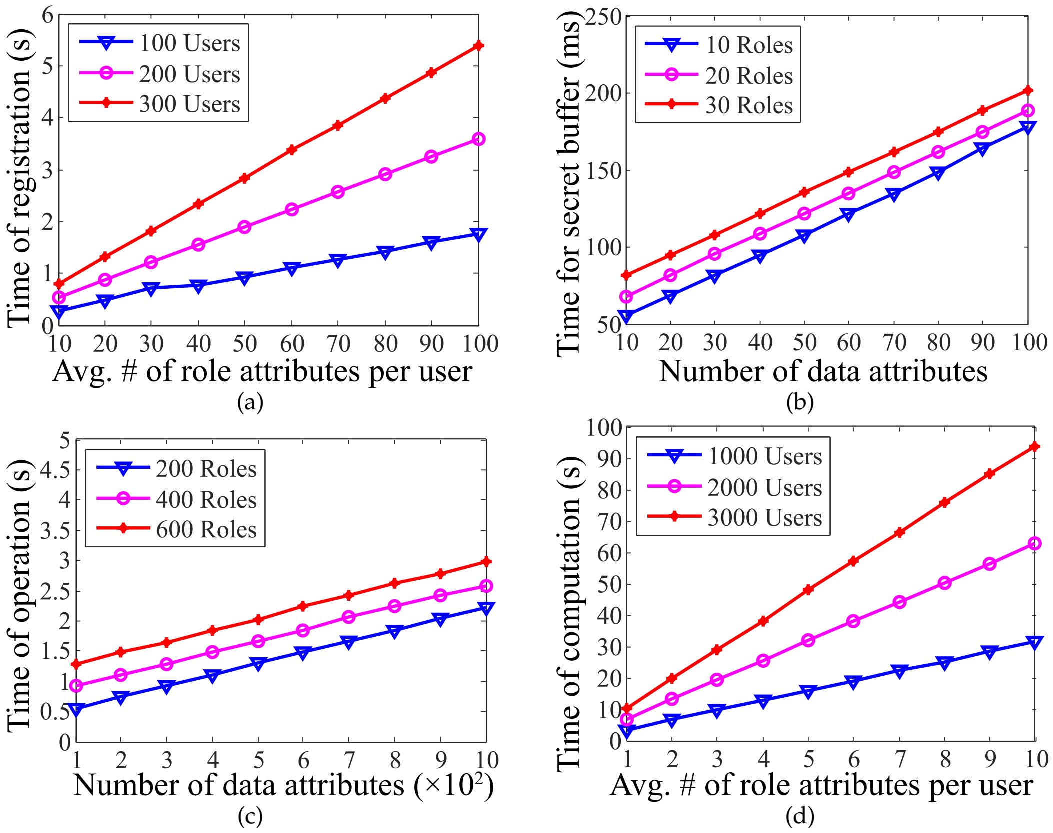

Fig. 7: Time cost of operation.

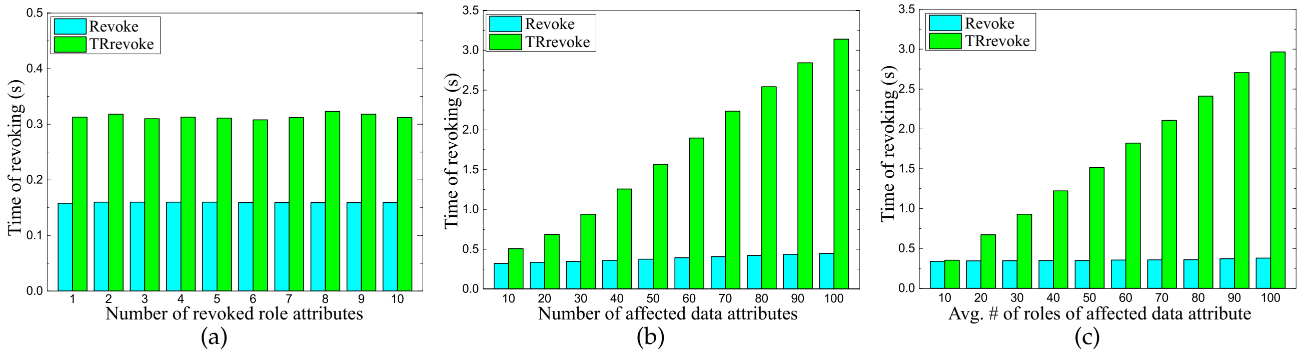

Fig. 8: Time cost of revoking role attributes.

and DEC, to denote the encryption and decryption scheme in our scheme, TRENC and TRDEC, to denote the encryption and decryption scheme in [13], respectively. In Fig. 6(a), we set the EHR involves 60 distinct role attributes, and we add 10 noisy role attributes in the second-layer encryption. We observe that, with the number of data attributes increasing, TRENC increases linearly, while ENC remains constant. Additionally, the more role attributes per data attribute, the more time is required by TRENC. The fundamental reason is that, TRENC has to conduct the exponential operation for all role attributes of all data attributes, while in ENC, we only need to conduct the exponential operation for every distinct role attribute. In Fig. 6(b), we set the EHR involves 100 distinct role attributes, each data attribute is accompanied with 10 role attributes. We observe that, as the frequency of repeated role attributes increases, TRENC increases linearly, while ENC also remains constant. Therefore, the higher frequency of the repeated role attributes, the more time is saved by our scheme. In Fig. 6(c), we set the EHR contains 20 data attributes, and each data attribute is accompanied with 10 role attributes. The figure illustrates that, as the number of noisy role attributes increases, ENC increases slowly, while TRENC remains the same. Meanwhile, the larger the total number of role attributes in the EHR, the more time is required by ENC. The reason is that, ENC is mainly affected by the number of noisy attributes and the number of distinct role attributes in the EHR,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2018.2790943, IEEE Transactions on Services Computing

while TRENC is mainly affected by the number of role attributes of each data attribute, and the number of data attributes in the EHR. Fig. 6(d) illustrates the time cost of decryption. As we can see, as the number of data attributes increases, both schemes spend more time on decryption. Additionally, the more role attributes are involved in the access policy of a data attribute, the more time is required for decryption.

Fig. 7(a) describes the time cost of user registration. As we can see, the more role attributes and more users are involved, the more time is required for registration. Fig. 7(b) shows the time cost of constructing the secret buffer, which is used to preserve the access pattern of data attributes and role attributes of the EHR. We observe that, the more data attributes and role attributes we submit to the cloud, the more time is spent. Fig. 7(c) illustrates the time cost of operation on the secret buffers. As we can see, the more data are provided, the more time is required by the cloud. Fig. 7(d) shows the time cost of the cloud spent on computing the secret data for data users. We observe that, as the number of role attributes increases, the time increases linearly. When 3000 users concurrently submit requests, and each user submits 10 role attributes, 93.721s are needed. This relatively long time further confirms that, instead of computing these data on the user-side, they should be executed on the cloud.

Fig. 8 illustrates the time cost for role attributes revoking. For a better comparison, we use Revoke and TRrevoke to denote the revoke operation in our scheme and in [13], respectively. In Fig. 8(a), we set 10 data attributes contain the revoked role attributes, each data attribute has 10 role attributes, and the EHR involves 200 role attributes. Fig. 8(a) illustrates that, as the number of revoked role attributes increases, the time cost of revoking remains nearly constant for both schemes. The reason is that, Revoke mainly depends on the number of distinct role attributes in the whole role attribute set. When we set that constant, the time cost would change very slowly. TRrevoke only depends on the number of affected data attributes, and the number of role attributes of the affected data attributes. This is also confirmed in Figs. 8(b) and (c). As we can see, with the number of the affected data attribute, and the number of role attributes of the affected data attributes increase, TRrevoke increases linearly, while Revoke increases very slowly.

7 Related Work

7.1 Privacy Preserving Electronic Healthcare Systems

The security and privacy problems in e-healthcare systems have attracted much interest. Benaloh et al. [10] proposed an efficient system that enables data owners to perform searches over their EHR data, and share partial access rights with other users. To achieve a data owner-centric access control over EHR in the multi-owner cloud system, Li et al. [11] proposed to adopt the multi-authority attribute-based encryption to encrypt each owner's EHR. In [12], Sun et al. designed a secure electronic health record system based on anonymous credentials, a pseudorandom number generator, and the proof of knowledge. Based on the noninteractive proof system, Guo et al. proposed a privacy preserving attribute-based authentication system in mobile health networks [27], and a verifiable and privacy-preserving monitoring scheme for the e-healthcare cloud system [28]. Zhou et al. [13] further proposed a white-box traceable and revocable multi-authority attribute-based encryption (TR-MABE) to achieve a multilevel privacy preservation for EHR data.

These works suffer from two main limitations. First, they only support the 'black or white' access control policy. Second, they suffer from the inference attack. Different from these works, we seek to design an inference attack-resistant e-healthcare cloud system with fine-grained access control.

7.2 Attribute-based Encryption

The Attribute-based Encryption (ABE) was first introduced by Sahai and Waters [29]. In the ABE, a user is authorized to decrypt a cipher-text only if his role attributes satisfy the corresponding access policy. Goyal et al. [30] first designed the Key-Policy Attribute-Based Encryption (KP-ABE), where a ciphertext is labeled with a set of role attributes, and the corresponding private key is associated with an access policy. Later, Bethencourt et al. [31] introduced the Ciphertext-Policy Attribute-Based Encryption (CP-ABE), where the private key is associated with role attributes and the cipher-text is associated with an access policy. In [25], Waters presented the efficient, expressive, and secure CP-ABE systems, where they embed a LSSS matrix into the public parameters.

Since the conventional ABE-based schemes will inevitably expose the role attributes and access policies to the public, they suffer from the inference attack. We aim to systematically construct a secure and privacy preserving e-health cloud system, so that it is immune to the inference attack and runs efficiently.

7.3 Inference Attack

The recent papers [14], [32] focus on the inference attack against encrypted databases. They demonstrate that by adopting techniques including frequency analysis and sorting attack, the inference attack can break most of existing encrypted databases. In these two papers, the data is assumed to be numerical, and encrypted with the property-preserving encryption schemes(the order preserving encryption, the deterministic encryption, etc.).

Different from these researches, we aim to protect the E-Healthcare data with fine-grained access control, the data can be either numerical or string value. To achieve this, we devise our own two-layer encryption scheme, the ciphertext is neither order-preserving nor deterministic, since we embed randomness there. Additionally, the inference attack described in our paper is launched by observing the role attributes, access policy, and access pattern(access frequency). With our constructions, we can prevent the attackers from achieving the inference attacks.

8 Conclusion

In this paper, for the first time, we design an inference attack resistant e-healthcare cloud system with fine-grained access control. We first propose a two-layer encryption scheme. In the first-layer encryption, we propose to define a specialized access policy for each data attribute in the EHR, generate a secret share for every distinct role attribute, and reconstruct the secret to encrypt each data attribute, which ensures a fine-grained access control, saves much encryption time, and conceals the frequency of role attributes occurring in the EHR. In the second-layer encryption, we propose to preserve the privacy of role attributes and access policies used in the first-layer encryption. Additionally, to take full advantage of the cloud server, we propose to let the cloud execute computationally intensive works on behalf of the data user without knowing any sensitive information. To preserve the access pattern of the data attributes in the EHR, we construct a blind data retrieving protocol based on the Paillier encryption. Furthermore, we show that our scheme can be easily extended to support search functionality. Finally, we conduct extensive security analyses and performance evaluations, which confirm the efficacy and efficiency of our schemes.

Acknowledgment

This work is partly supported by the National Natural Science Foundation of China under Grant No. 61472125, the Research Foundation of Chinese Ministry of Education and China Mobile Communications Corporation under Grant No. MCM20122061, and the scholarship from China Scholarship Council.

References

[1] Google fit. [Online]. Available: https://developers.google.com/fit

[2] Healthkit. [Online]. Available: https://developer.apple.com/healthkit

[3] Ibm watson health cloud. [Online]. Available: http://www.ibm.com/smarterplanet/us/en/ibmwatson/health

[4] Y. Zhu, G.-J. Ahn, H. Hu, S. S. Yau, H. G. An, and C.-J. Hu, "Dynamic audit services for outsourced storages in clouds," *IEEE Transactions on Services Computing*, vol. 6, no. 2, pp. 227–238, 2013.

[5] H. Tian, Y. Chen, C.-C. Chang, H. Jiang, Y. Huang, Y. Chen, and J. Liu, "Dynamic-hash-table based public auditing for secure cloud storage," *IEEE Transactions on Services Computing*, pp. 1–10, 2015.

[6] W. Zhang, Y. Lin, S. Xiao, Q. Liu, and T. Zhou, "Secure distributed keyword search in multiple clouds," in *Proc. IEEE/ACM IWQOS'14*. Hongkong: IEEE/ACM, May 2014, pp. 370–379.

[7] W. Zhang, S. Xiao, Y. Lin, T. Zhou, and S. Zhou, "Secure ranked multi-keyword search for multiple data owners in cloud computing," in *Proc. 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN2014)*. Atlanta, USA: IEEE, jun 2014, pp. 276–286.

[8] D. Nascimento and M. Correia, "Shuttle: Intrusion recovery for paas," in *Proc. IEEE Distributed Computing Systems (ICDCS'15)*, Ohio, USA, Jun. 2015, pp. 10–20.

[9] At risk of exposure -in the push for electronic medical records, concern is growing about how well privacy can be safeguarded. [Online]. Available: http://articles.latimes.com/2006/jun/26/health/he-privacy26

[10] J. Benaloh, M. Chase, E. Horvitz, and K. Lauter, "Patient controlled encryption: ensuring privacy of electronic medical records," in *Proceedings of the 2009 ACM workshop on Cloud computing security*. ACM, 2009, pp. 103–114.

[11] M. Li, S. Yu, K. Ren, and W. Lou, "Securing personal health records in cloud computing: Patient-centric and fine-grained data access control in multi-owner settings," in *Security and Privacy in Communication Networks*. Springer, 2010, pp. 89–106.

[12] J. Sun, X. Zhu, C. Zhang, and Y. Fang, "Hcpp: Cryptography based secure ehr system for patient privacy and emergency healthcare," in *Distributed Computing Systems (ICDCS), 2011 31st International Conference on*. IEEE, 2011, pp. 373–382.

[13] J. Zhou, Z. Cao, X. Dong, and X. Lin, "Tr-mabe: White-box traceable and revocable multi-authority attribute-based encryption and its applications to multi-level privacy-preserving e-healthcare cloud computing systems," in *INFOCOM, 2015 Proceedings IEEE*. Hong Kong: IEEE, 2015, pp. 2398–2406.

[14] M. Naveed, S. Kamara, and C. V. Wright, "Inference attacks on property-preserving encrypted databases," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2015, pp. 644–655.

[15] A. Beimel, "Secure schemes for secret sharing and key distribution," Ph.D. dissertation, Technion-Israel Institute of technology, Faculty of computer science, 1996.

[16] Z. Liu, Z. Cao, and D. S. Wong, "Efficient generation of linear secret sharing scheme matrices from threshold access trees," *IACR Cryptology ePrint Archive*, 2010. [Online]. Available: http://eprint.iacr.org/2010/374

[17] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Advances in cryptology-EUROCRYPT99*. Springer, 1999, pp. 223–238.

[18] B. Wang, W. Song, W. Lou, and Y. T. Hou, "Inverted index based multi-keyword public-key searchable encryption with strong privacy guarantee," in *INFOCOM, 2015 Proceedings IEEE*. Hong Kong: IEEE, 2015, pp. 2092–2110.

[19] W. Zhang, Y. Lin, S. Xiao, J. Wu, and S. Zhou, "Privacy preserving ranked multi-keyword search for multiple data owners in cloud computing," *IEEE Transactions on Computers*, 2015.

[20] L. Zhang, T. Jung, C. Liu, X. Ding, X. Y. Li, and Y. Liu, "Pop: Privacy-preserving outsourced photo sharing and searching for mobile devices," in *Proc. IEEE Distributed Computing Systems (ICDCS'15)*, Ohio, USA, Jun. 2015, pp. 10–20.

[21] Z. Qin, H. Xiong, S. Wu, and J. Batamuliza, "A survey of proxy re-encryption for secure data sharing in cloud computing," *IEEE Transactions on Services Computing*, pp. 1–18, 2016.

[22] W. Zhang, S. Zhou, A. Srinivasan, J. Wu, and Y. Lin, "Detecting attacks smartly in vehicle cloud computing," in *Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress*, 2017, pp. 245–252.

[23] Y. Tang and L. Liu, "Privacy-preserving multi-keyword search in information networks," *IEEE transactions on knowledge and data engineering*, vol. 27, no. 9, pp. 2424–2437, 2015.

[24] Y. Tang, L. Liu, A. Iyengar, K. Lee, and Q. Zhang, "e-ppi: Locator service in information networks with personalized privacy preservation," in *Distributed Computing Systems (ICDCS), 2014 IEEE 34th International Conference on*. IEEE, 2014, pp. 186–197.

[25] B. Waters, "Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization," in *Public Key Cryptography–PKC 2011*. Springer, 2011, pp. 53–70.

[26] J. A. Akinyele, C. Garman, I. Miers, M. W. Pagano, M. Rushanan, M. Green, and A. D. Rubin, "Charm: a framework for rapidly prototyping cryptosystems," *Journal of Cryptographic Engineering*, vol. 3, no. 2, pp. 111–128, 2013.

[27] L. Guo, C. Zhang, J. Sun, and Y. Fang, "A privacy-preserving attribute-based authentication system for mobile health networks," *Mobile Computing, IEEE Transactions on*, vol. 13, no. 9, pp. 1927–1941, 2014.

[28] L. Guo, Y. Fang, M. Li, and P. Li, "Verifiable privacy-preserving monitoring for cloud-assisted mhealth systems," in *INFOCOM, 2015 Proceedings IEEE*. Hong Kong: IEEE, 2015, pp. 1026–1034.

[29] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Advances in Cryptology–EUROCRYPT 2005*. Springer, 2005, pp. 457–473.

[30] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of the 13th ACM conference on Computer and communications security*. Acm, 2006, pp. 89–98.

[31] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Security and Privacy, 2007. SP'07. IEEE Symposium on*. IEEE, 2007, pp. 321–334.

[32] M. S. Islam, M. Kuzu, and M. Kantarcioglu, "Inference attack against encrypted range queries on outsourced databases," in *Proceedings of the 4th ACM conference on Data and application security and privacy*. ACM, 2014, pp. 235–246.

Wei Zhang received his B.S. degree in Computer Science from Hunan University, China, in 2011. Since 2011, he has been a Ph.D. candidate in College of Computer Science and Electronic Engineering, Hunan University. Since 2014, he has been a visiting student in Department of Computer and Information Sciences, Temple University. His research interests include cloud computing, network security and data mining.

Yaping Lin received his B.S. degree from Hunan University and his M.S. degree from National University of Defense Technology, in 1982 and 1985, respectively. He received his Ph.D degree from Hunan University in 2000. He has been a professor and PhD supervisor in Hunan University since 1996. During 2004-2005, he worked as a visiting researcher at University of Texas at Arlington. His research interests include cloud computing, network security and machine learning. He is a member of IEEE.

Jie Wu is the chair and a Laura H. Carnell professor with the Department of Computer and Information Sciences, Temple University, Philadelphia, PA. Prior to joining Temple University, he was a program director with the National Science Foundation and a distinguished professor with Florida Atlantic University, Boca Raton, FL. He regularly publishes in scholarly journals, conference proceedings, and books. His research interests include mobile computing and wireless networks, routing protocols, cloud and green computing, network trust and security, and social network applications. He serves on several editorial boards, including IEEE Transactions on Computers, IEEE Transactions on Service Computing, and Journal of Parallel and Distributed Computing. He was general cochair/chair for IEEE MASS 2006, IEEE IPDPS 2008, and IEEE ICDCS 2013, as well as program cochair for IEEE INFOCOM 2011 and CCF CNCC 2013. he served as a general chair forACM MobiHoc 2014. He was an IEEE Computer Society distinguished visitor, ACM distinguished speaker, and chair for the IEEE Technical Committee on Distributed Processing (TCDP). He is a CCF distinguished speaker and a fellow of the IEEE. He is the recipient of the 2011 China Computer Federation (CCF) Overseas Outstanding Achievement Award.

Ting Zhou received her B.S. degree from University of South China in 2011. Since 2011, she has been a M.S. candidate in College of Computer Science and Electronic Engineering, Hunan University. Her research interests include cloud computing, Internet of Things.