Computer Science and Information Systems 16(3):831–847 https://doi.org/10.2298/CSIS180830029Y

# Privacy-preserving Multi-authority Attribute-based Encryption with Dynamic Policy Updating in PHR

Xixi Yan[1], Hao Ni[1], Yuan Liu[1], and Dezhi Han[2]

[1] School of Computer Science and Technology, Henan Polytechnic University, Jiaozuo 454003, China
yanxx@hpu.edu.cn
[2] College of Information Engineering, Shanghai Maritime University, Shanghai 201306, China
dzhan@shmtu.edu.cn

**Abstract.** As a new kind of patient-centred health-records model, the personal health record (PHR) system can support the patient in sharing his/her health information online. Attribute-Based Encryption (ABE), as a new public key cryptosystem that guarantees fine-grained access control of outsourced encrypted data, has been used to design the PHR system. Considering that privacy preservation and policy updating are the key problems in PHR, a privacy-preserving multi-authority attribute-based encryption scheme with dynamic policy updating in PHR was proposed. In the scheme, each of the patient's attributes is divided into two parts: attribute name and attribute value. The values of the user's attributes will be hidden to prevent them from being revealed to any third parties. In addition, the Linear Secret-Sharing Scheme (LSSS) access structure and policy-updating algorithms are designed to support all types of policy updating (based on "and", "or", and "not" operations). Finally, the scheme is demonstrated to be secure against chosen-plaintext attack under the standard model. Compared to the existing related schemes, the sizes of the user's secret key and ciphertext are reduced, and the lower computing cost makes it more effective in the PHR system.

**Keywords:** attribute-based encryption, privacy preserving, policy dynamic updating, fine-grained access control, personal health record.

## 1. Introduction

As a kind of patient-centred health-record model, the Personal Health Record (PHR) system has been gradually popularized. PHR is a system that is created and maintained by a patient or medical consumer, and the health information is based on his/her own health status, medications, laboratory tests, diagnostic studies, questions, allergies, vaccinations, etc. In the PHR system, a patient can maintain his/her personal health information from every channel for local storage, organization, management, sharing and tracking. Patients can access the system online anytime and anywhere. Besides, these records can help patients explain their health problems simply when they go to the doctor, and can also provide useful information when they are filing medical insurance claims.

In practical applications, to reduce the PHR centers' overhead of creation and control, large amounts of data are often outsourced to third-party PHR cloud servers. However, this comes with some additional security and privacy concerns, especially since the patients

do not want their health information to be completely transparent to the cloud server. And worst of all, once the cloud server is hacked, all the patients' health information will be leaked. In addition, when doctors, family members, insurance company staff or other users access a patient's health information, the PHR cloud server needs to interact with them to allow or restrict the visitor's access ability and scope. This will lead to lower efficiency. Obviously, considering the growth of user dimensions and data traffic, and the privacy requirements of the user's personal health information, new encryption technology and access-control mechanisms have been put forward for information management in the PHR system.

Attribute-based encryption (ABE)[1], a new public-key encryption technology, ties the user's identity with a series of attributes. The user's private key or ciphertext is defined according to the attribute set or access structure, and only when the attribute set and the access structure match can the user decrypt to obtain the message. It can guarantee the confidentiality of patient health information data in the PHR cloud server, and realize non-interactive access control with ABE. However, in practice, some sensitive private information is closely related to the patient's attributes in the access policy, so it will lead to privacy disclosure. For example, if a patient sets his/her access policy as People's Hospital, Doctor, Psychiatry, only those that match the three attributes can access his/her health records, yet this attribute information easily reveals the patient's private data. Moreover, the access policy may be changed when a patient needs a referral for treatment. If a user who satisfies Red Cross Hospital, Doctor, Psychiatry or Doctor, Psychiatry or Red Cross Hospital, Doctor, Internal Medicine should be able to access his/her health records, it will require the PHR system to support changing the access policy. Therefore, determining how to both protect privacy and support access-policy dynamic updating is a key problem when the ABE mechanism is applied in the PHR system.

On the other hand, in the pratice of ABE in PHR, user's attributes may be issued by different authorities. For instance, Alice wants to encrypt a message under access policy("DOCTORATE" and "DOCTOR"). Only the recipient who has received a doctorate and now is a doctor, can recover the message. School is responsible to issue attributes (bachelor's degree, master's degree or doctorate) to student , while hospital is responsible to issue attributes (doctor or nurse) to its employees. Therefore, the scheme in which the attribute universe is assumed to be managed by a single authority is not apply this scenarios.

In this paper, our proposed scheme aims to construct the MA-ABE scheme for PHR scenario which has the requirement for privacy preserving and policy updating. In the paper, a partially hidden policy mechanism is used to protect the user's GID and attribute privacy. Futhermore, our scheme also allows policy updating when patient wants to change the access policy associated with ciphertext.

### 1.1. Related Work

The traditional ABE [2-5] has only one credible organization to manage all the attributes, but in practice, the attributes are often managed by multiple authorities. Moreover, the attribute authority needs to distribute keys to all authenticated users. This will lead to a heavy workload that will become the system-performance bottleneck. Hence, Chase [6] proposed the first multi-authority attribute-based encryption scheme. In the scheme, there are a trusted central authority and some attribute authorities, and it also allows

any polynomial number of independent authorities to manage attributes and distribute privacy keys. In MA-ABE, attribute authorities take on the tasks of monitoring attributes and issuing keys. This results in a significant reduction in the system burden and high flexibility, so MA-ABE is more suitable for the PHR system. In terms of privacy protection, Chase and Chow [7] improve privacy and security in the MA-ABE scheme by removing the trusted central authority and introducing an anonymous credential to protect the users' global identifiers (GID). However, it is only supported an AND policy. Lewko and Waters [8] proposed a decentralizing attribute-based encryption scheme. In the scheme, the authorities work independently without coordination among them. It actually generalizes to handle any policy that can be expressed as a Linear Secret Sharing Scheme (LSSS) or equivalently a monotone span program. Han et al. [9] proposed a privacy-preserving decentralized key-policy attribute-based encryption scheme. But they[8-9] also consider the privacy of the GID. Recently, Xhafa et al. [10] proposed a privacy-aware MA-ABE PHR scheme that incorporates user accountability in cloud computing. In the scheme, the access policy is hidden to protect the patient's privacy. But it is only support AND gates and can't support policy updating. Han et al. [11] proposed an improved decentralized ciphertext-policy attribute-based encryption scheme, which is the first scheme to protect the privacy of attributes in MA-ABE. In the scheme, a central authority is required to generate the global key and issue secret keys to users. Commitment schemes and zero-knowledge proof technology are introduced to protect the user's GID and attributes. However, note that it has been proved not to protect the user's attributes effectively in scheme [12]. Qian et al. [13] proposed a privacy-preserving personal health record scheme using multi-authority attribute-based encryption with revocation. An anonymous key-issuing protocol, which is used to generate a secret key for the patient, is introduced to protect the privacy, but it is also only protects users' GIDs. Moreover, the scheme support policy update when patient wants to change the access policy associated with ciphertext, but simple access structures can be implemented. Xia et al. [14] proposed attribute-based access control scheme with efficient revocation in cloud computing. It could support an efficient user revocation mechanism, but it can't protect the user's attribute privacy.

The idea of policy updating for ABE originates from the delegation of the ciphertext in [15]. A proxy method is designed to update the ciphertext, but its new policy is more stringent than the original one. Yang et al. [16] proposed a scheme that supports verifiable policy-update outsourcing for big-data access control in the cloud. In this scheme, they analyzed in detail how to update the ciphertext, which is encrypted with a Boolean formula and a linear secret-sharing scheme (LSSS) structure separately. Then they proposed policy-updating algorithms for all types of policies. However, their scheme is secure only under the general group model. Ying et al. [17] proposed a partially policy-hidden CP-ABE scheme that supports dynamic policy updating. It can support any type of policy updating and the users' privacy is effectively protected by using partial policy hiding, and it is proved to be chosen-plaintext attack (CPA) secure in the standard model. Li et al. [18] proposed a scheme enabling fine-grained access control with efficient attribute revocation and policy updating in the smart grid. It not only defines the system model in the smart grid, but also leverages Third-party Auditor to support attribute revocation. Moreover, it can support dynamic policy updating and apply ABE to the smart grid effectively. Wu [19] constructed a multi-authority CP-ABE scheme

with policy updating in cloud storage. The multi-authority established by the scheme can prevent the key from being leaked and it can support policy updates under the access-tree structure. Ying et al. [20] designed an adaptively secure ciphertext-policy attribute-based encryption with dynamic policy updating. This scheme is regarded as the first MA-ABE scheme with dynamic policy updating that is proved to be adaptively secure under the standard model. A signature subsystem is introduced to resist collusion attacks. However, the system consists of multiple central authorities and multiple attribute authorities. The users' attribute keys are generated by all the central authorities, which leads to high communication cost and storage cost. Liu et al. [21] proposed an attribute-based encryption with outsourcing decryption, attribute revocation and policy updating. It is claimed that the scheme is more useful and flexible in practice, but it is lack of privacy protection. Jiang et al. [22] presented a CP-ABE supporting access policy update and its extension with preserved attributes. However , it dose not take into account the application scenarios of multi-authority.

**Table 1.** Comparison Among Related MA-ABE Schemes and Ours

| Scheme | Multi-authority | Without an authentication center | Authority independence | Flexible access policy | Privacy protection | Policy updating |
|---|---|---|---|---|---|---|
| Scheme [7] | √ | √ | - | AND | GID | - |
| Scheme [8] | √ | √ | √ | √ | GID | - |
| Scheme [10] | √ | √ | √ | AND | Access policy | - |
| Scheme [11] | √ | √ | √ | √ | GID, attribute | - |
| Scheme [17] | - | - | - | √ | attribute | √ |
| Scheme [18] | √ | - | √ | √ | version number | √ |
| Scheme [20] | √ | - | √ | √ | GID | √ |
| Scheme [21] | √ | - | √ | √ | - | √ |
| Our Scheme | √ | √ | √ | √ | GID | √ |

Based on the above depiction, the comparisons among related MA-ABE schemes and our proposed scheme are listed in Table 1. From the above table, it is shown that the most MA-ABE scheme towards privacy protection [7-8,11, 20] generally protect the user's privacy by protecting the user's GID, and rarely consider the privacy of attributes. Besides, the second problem is that the scheme towards policy updating [17-18,20-21] needs an authentication center which is responsible to integrate the secret keys. The authentication center can decrypt every ciphertext in the system, which would endanger the whole system if it's corrupted. Futhermore, it would also increase the computation and communication cost to run and maintain such a fully trusted authority. In contrast, our work in this paper addresses both of these limitations simultaneously.

### 1.2. This Study's Contribution

On addressing the above issues, we propose a privacy-preserving multi-authority attribute-based encryption with dynamic policy updating in PHR. It can protect the privacy of the patient's attributes and support dynamic policy updating. The theoretical innovations of this study are as follows:

(1)The privacy of attributes is considered in our scheme, and the GID is also considered as an attribute that would be protected. In the scheme, we utilize a partially hidden policy mechanism[23] to encrypt the patient's health information. Each of the patient's attributes is divided into two parts: attribute name and attribute value. The values of the user's attributes will be hidden to prevent them from being revealed to any third parties, so the user's privacy will be effectively preserved. In this way, it is more effective in attribute preservation than the traditional privacy-preserving ABE scheme, and it has lower storage cost.

(2)To meet the requirements for policy updating, we design a dynamic policy updating algorithm for Linear Secret-Sharing Scheme (LSSS) access structure in PHR. Our policy updating scheme supports the updating of any type of fine-grained policy involving 'AND', 'OR', or 'NOT', as compared to the ciphertext delegation method[15] which can only re-encrypt the ciphertext under a more restrictive policy. Other, the scheme is designed for LSSS access structure, which is more efficient than the straight-forward policy updating implementation.

(3)Our scheme has no central authority, and it has no private interactivity among attribute authorities. Unlike[17-18,20], there isn't a central authority in our scheme, while there are multiple authorities which generate user's private key by their monitored attributes without any interactivity. The security could be enhanced and the communication cost and computing cost will be reduced since the central authority is removed. Therefore, our scheme does not require central authority and it is more efficient and acceptable for real-world applications.

### 1.3. Organization

The rest of this paper is organized as follows. In Section 2, we recall the basic definition of cryptographic primitives used in this paper. The definition and security model for our scheme is given in Section 3. In Section 4, a privacy-preserving multi-authority attribute-based encryption with dynamic policy updating in PHR is proposed, and it is proved to be correct and secure in Section 5. Subsequently, we give a performance analysis in Section 6. Finally, we conclude this paper in Section 7.

## 2. Background

### 2.1. Bilinear Maps

$G_0$ and $G_T$ are two multiplicative cyclic groups of prime order $p$.Let $g_0$ be the generator of $G_0$.There is a bilinear map,$e : G_0 \times G_0 \to G_T$, with the following properties:

(1) Bilinearity: $\forall x, y \in Z_P$,$\forall a, b \in G_0$,we have $e(a^x, b^y) = e(a, b)^{xy}$;

(2) Computability: $\forall a, b \in G_0$, we have an effective algorithm to compute $e(a, b)$;

(3) Non-degeneracy: $e(g_0, g_0) \neq 1$.

### 2.2. Access Structure

Suppose $\{p_1, p_2, ..., p_n\}$ is a set of parties. A set $P \subseteq 2^{\{p_1,p_2,...,p_n\}}$ is monotone if $\forall X, Y$: if $X \in P$ and $X \subseteq Y$, then $Y \in P$. An access structure is a set $P$ of non-empty subsets of $\{p_1, p_2, ..., p_n\}$, i.e.,$P \subseteq 2^{\{p_1,p_2,...,p_n\}} \backslash \{\emptyset\}$. The collections in $P$ are called the authorized sets, and the collections not in $P$ are called the unauthorized sets.

### 2.3. Linear Secret-Sharing Scheme (LSSS)

A secret-sharing scheme over a collection $P$ is linear if: (1) the shares for each party form a vector over $Z_p$ ; and (2) there exists a matrix $M$ of size $l \times n$ such that for all $i = 1, ..., l$, the $i'$th row is labeled with a function $\rho(i)$.Randomly choose $s \in Z_p$ and a vector $v = (s, v_2, ..., v_n) \in Z_p^n$, where $s$ is the secret to be shared. The share $\lambda_i = M_i \cdot v$ belongs to party $\rho(i)$, where $M_i$ is the $i'$th row of $M$.

Linear reconstruction property: Suppose a scheme's access structure is LSSS. Let $S$ be an authorized set and $I = \{i|\rho(i) \in S\}$. There exists a set of constants $\{\omega_i \in Z_p\}_{i \in I}$ that can be used to compute the secret $s : \sum_{i \in I} \omega_i \lambda_i = s$.

### 2.4. Decisional q-Parallel Bilinear Diffie-Hellman Exponent Assumption (q-PBDHE)

Let $G$ and $G_T$ be two multiplicative cyclic groups of prime order $p$. Let $g$ be the generator of $G$. There is a bilinear map $e : G_0 \times G_0 \rightarrow G_T$. Choose $a, s, b_1, ..., b_n \in Z_p$ and $T \in G_T$. For a tuple, $y = (g, g^s, g^a, ..., g^{(a^q)}, g^{(a^{q+2})}, ..., g^{(a^{2q})})$; $\forall 1 \leq j \leq q$ $g^{s \cdot b_j}, g^{\frac{a}{b_j}}, ..., g^{\frac{a^q}{b_j}}, g^{\frac{a^{q+2}}{b_j}}, ..., g^{\frac{a^{2q}}{b_j}}$; $\forall 1 \leq j, k \leq q, k \neq j, g^{\frac{a \cdot s \cdot b_k}{b_j}}, ..., g^{\frac{a^q \cdot s \cdot b_k}{b_j}}$). It is hard to distinguish $T$ and $e(g, g)^{a^{q+1}S}$ in the group $G_T$.

We say the q-PBDHE assumption holds on the bilinear group $(e, p, G, G_T)$ if there is no polynomial-time adversary $A$ that can distinguish $(y, e(g, g)^{a^{q+1}S})$ and $(y, T)$ with a negligible advantage $Adv_A = |Pr[A(y, e(g, g)^{a^{q+1}S}) = 1] - Pr[A(y, T) = 1]| \geq \varepsilon$.

## 3. System Model and Design Model

### 3.1. System Model

As shown in Figure 1, our PHR system model consists of four parts: Attribute authority, PHR cloud server, patients and PHR users. The N attribute authorities manage users' attributes and generate privacy keys. The PHR cloud server is responsible for providing health-information data-storage services and outsourcing services for the authorized user, such as ciphertext updating and fine-grained access control. The patient sends the encrypted PHR data to the PHR cloud server. If the PHR user applies for access to the patient's PHR data, the attribute authority will check whether his private-key attributes satisfy the access structure. Only an authorized user can decrypt the patient's private information. When the patient wants to alter the access policy, he only calculates the update key and sends it to the PHR cloud server. The ciphertext updating is performed by the PHR cloud server.

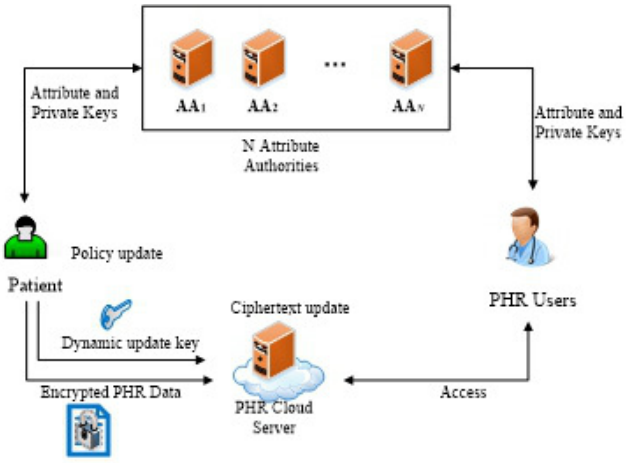


**Fig. 1.** System model

### 3.2. Definition of the System Algorithm

The scheme consists of the following algorithms:

1. $Initialization$

(1)$Global\ Setup(1^{\lambda}) \rightarrow PP$ : Taking as input a security parameter $1^{\lambda}$, the algorithm output the public parameter $PP$.

(2)$Authority\ Setup(PP) \rightarrow (PK_i, SK_i)$ : This algorithm is run by the attribute authority. Taking the public parameter $PP$ as input, it outputs a secret-public key pair $(PK_i, SK_i)$ for each attribute authority.

2. $Encrypt(PP, m, (M, \rho, Z), PK_i) \rightarrow C$ : This algorithm is run by the patient. Taking as input the public parameter $PP$, the PHR message $m$, access structure $(M, \rho, Z)$, and the attribute-authority public keys $PK_i$, it outputs the ciphertext $C$, and the encryption information $En(m)$ is preserved by the patient.

3. $KeyGen(PP, SK_i, A_u) \rightarrow SK_u$ : This algorithm inputs the public parameter $PP$, the attribute-authority secret keys $SK_i$, and the user's attributes $A_u$,and outputs the user's privacy key $SK_u$.

4. $Decrypt(PP, C, SK_u) \rightarrow m$ : This algorithm inputs the public parameter $PP$, the user's privacy key $SK_u$, and the ciphertext $C$ and outputs the message $m$.

5. $Policyupdate$

(1)$DK_mGen(En(m), (M', \rho', Z'), (M, \rho, Z)) \rightarrow DK_m$ : This algorithm is run by the patient. Taking as input the encryption information $En(m)$ of $m$, the current policy $(M, \rho, Z)$ and the new policy $(M', \rho', Z')$, it outputs the dynamic policy-update key $DK_m$.

(2)$CUpdate(C, DK_m) \rightarrow C'$ : This algorithm is run by the PHR cloud server. Taking as input the current ciphertext $C$ and the dynamic policy-update key $DK_m$, it outputs the new ciphertext $C'$.

### 3.3. Security Model

The semantic security against chosen-plaintext attack (CPA) is modeled in the selective access structure model (SAS), in which the adversary must provide the access structure he wishes to attack before he receives the public parameters from the challenger. The game is carried out between a challenger and an adversary. The game is as follows.

Initialization : The adversary $A$ sends the challenge access structure $(M^*, \rho^*, Z^*)$ to the challenger.

Global Setup : The challenger runs the Global Setup algorithm to generate the public parameter $PP$ and sends it to $A$.

Authorities Setup : The challenger runs the Authorities Setup algorithm to generate the attribute authority's secret-public keys and sends them to $A$.

Phase 1 : The adversary makes many attribute private-key queries and determines which attributes do not appear in $(M^*, \rho^*, Z^*)$. The challenger generates the privacy key and sends it to $A$.

Challenge : The adversary submits two equal-length messages $m0$ and $m1$. The challenger randomly chooses and encrypts mc under $(M^*, \rho^*, Z^*)$. The ciphertext is given to $A$.

Phase 2 : Phase 1 is repeated.

Guess : outputs his guess $c'$ on $c$.

Definition 1: Our scheme is secure if any probably polynomial-time adversary $A$ making $q$ secret-key queries can win the above game with a negligible advantange $\varepsilon = |Pr[c = c'] - \frac{1}{2}|$ .

## 4. Scheme of the PHR System

**Table 2.** Notations

| Symbols | Definition |
|---|---|
| $Z_q$ | An integer set of mod $q$ |
| $AA_i(i = 1, ..., n)$ | Attribute authority |
| $a_i(i = 1, ..., n)$ | Attribute name |
| $a_{i,j}(i = 1, ..., n, j = 1, ..., n_i)$ | Attribute value |
| $(M, \rho, Z)(M', \rho', Z')$ | Old access policy and new access policy. $M$ is a matrix, $\rho$ is a function which maps each row $M_i$ to an attribute name, $Z$ denotes the set of attribute values that the patient designs and hides in the policy. $(M', \rho', Z')$ is defined similar to $(M, \rho, Z)$ |
| $PP$ | Public parameter |
| $PK_i, SK_i$ | The secret-public key pair for each attribute authority $AA_i$ |
| $SK_u$ | The user's privacy key |
| $En(m)$ | The encryption information of $m$, and $m$ is the patient's health data |

Before the PHR system operation, we assume that the universe of attributes consists of $n$ attribute names and $n = (a_1, a_2, \ldots, a_n)$ such that each $a_i$ has $n_i$ attribute values and $A_i = (a_{i,1}, a_{i,2}, ..., a_{i,n_i})$. We use $A_u = (a_1 : a_{1,t_1}, a_2 : a_{2,t_2}, ..., a_n : a_{n,t_n})$ to denote the user's attribute set. It is obvious that $a_{i,t_i} \in A_i$. Suppose that there are $N$ authorities $\{AA_1, AA_2, ..., AA_N\}$ and $AA_i$ monitors for attribute name $a_i$ and $a$ set of attribute values corresponding to $a_i$.

A patient's access structure is defined as $(M, \rho, Z)$. $M$ is a matrix with $l$ rows and $n$ columns. The function maps each row $M_i$ to an attribute name. $Z$ denotes the attribute values that the patient designs and hides in the policy, and $Z = (Z_{\rho(1)}, ..., Z_{\rho(l)})$. A PHR user's attribute set matches $(M, \rho, Z)$ if and only if for all $i \in \{1, ..., l\}$ , $a_{\rho(i)} = Z_{\rho(i)}$ , and there exist constants $\omega_i \in Z_p$ such that $\sum_{i \in I} \omega_i \lambda_i = s$.

The algorithms of our N-authority CP-ABE scheme are presented in the rest of the section.

### 4.1. Initialization

1)$Global\ Setup(1^\lambda) \rightarrow PP$: Taking as input a security parameter $1^\lambda$, the algorithm outputs the public parameter $PP = (e, p, g, h, G, G_T)$, where $e : G \times G \rightarrow G_T$. $G$ and $G_T$ are two multiplicative cyclic groups of prime order $p$. Let $g$ and $h$ be the generators of $G$.

2)$Authority\ Setup(PP) \rightarrow (PK_i, SK_i)$: This algorithm is run by the attribute authority. Each attribute authority $AA_i$ chooses $\alpha_i \in Z_p$ and computes

$$A_i = e(g, g)^{\alpha_i} \tag{1}$$

For attribute name $a_i$ and each attribute value $a_{i,n_i}$, attribute authority $AA_i$ chooses $r_i, t_{i,n_i} \in Z_p$ and computes

$$R_i = g^{r_i} \tag{2}$$

$$T_{i,n_i} = g^{t_{i,n_i}} \tag{3}$$

Then, the attribute authority $AA_i$ publishes the public key $PK_i = \{A_i, R_i,$ $(T_{i,n_i})_{\forall a_{i,n_i}} \in A_i\}$ and keeps the private key $SK_i = \{\alpha_i, r_i, (t_{i,n_i})_{\forall a_{i,n_i}} \in A_i\}$.

### 4.2. Encrypt

This algorithm is run by the patient. The patient encrypts his or her PHR data with an access policy $(M, \rho, Z)$. The ciphertext is sent to the PHR cloud server. The algorithm is run as follows:

Step 1: The patient chooses $s \in Z_p$ and a vector $v = (s, v_2, ..., v_n) \in Z_p^n$ randomly.

Step 2: For each row of the matrix, compute $\lambda_i = M_i \cdot v$.

Step 3: The patient chooses $q_1, ..., q_l \in Z_p$ randomly and computes

$$c_0 = m \cdot \prod\nolimits_{i \in I_A} e(g, g)^{a_i s} \tag{4}$$

$$c_1 = g^s \tag{5}$$

for $i \in (1, ..., l)$

$$C_{2,i} = h^{\lambda_i}(T_{\rho(i)}^{Z_{\rho(i)}})^{-q_i} \tag{6}$$

$I_A$ is a set that consists of the indices of the authorities that are selected to encrypt $m$.

Step 4: Finally, the patient sends the ciphertext $C = \{C_0, C_1, (C_{2,i}, C_{3,i})_{i\in(1,\ldots,l)}\}$ to the PHR cloud server, and the encryption information $En(m)$ is preserved by the patient.

### 4.3. KeyGen

When a doctor or other PHR users request access to the patient's PHR data, the $KeyGen$ algorithm is run to generate the doctor's privacy key. This algorithm is executed by the attribute authority. $AA_i$ checks whether it handles the doctor's attribute set $A_u$. If this is the case, it computes

$$D_i = g^{\alpha_i} h^{r_i} \tag{7}$$

$$R_{i,t_i} = (T_{i,t_i}^{a_{i,t_i}})^{r_i} \tag{8}$$

Then $AA_i$ sends the user's privacy key $SK_u = \{D_i, R_{i,t_i}\}_{1\le i\le n}$ to the doctor. Otherwise, it outputs NULL.

### 4.4. Decrypt

When the doctor obtains his privacy key, he can decrypt the ciphertext. The Decrypt algorithm is executed as follows:

Step 1: The system checks whether the conjunction of $(M, \rho, Z)$ can be satisfied by the doctor's attributes.

Step 2: If this is the case, it computes

$$\begin{aligned} e(C_{2,i}, R_i) &= e(h^{\lambda_i}(T_{\rho(i)}^{Z_{\rho(i)}})^{-q_i}, g^{r_i}) = e(h^{\lambda_i}(g^{t_{\rho(i)}Z_{\rho(i)}})^{-q_i}, g^{r_i}) \\ &= e(h,g)^{\lambda_i r_i} e(g,g)^{-q_i t_{\rho(i)} Z_{\rho(i)} r_i} \end{aligned} \tag{9}$$

$$e(C_{3,i}, R_{i,n_i}) = e(g^{q_i}, (T_{\rho(i)}^{u_{\rho(i)}})^{r_i}) = e(g,g)^{q_i t_{\rho(i)} Z_{\rho(i)} r_i} \tag{10}$$

Step 3:Finally, he computes $m = \frac{C_0 \cdot \prod_{i\in I_A}(e(C_{2,i},R_i)e(C_{3,i},R_{i,n_i}))^{\omega_i}}{\prod_{i\in I_A} e(C_1,D_i)}$, where $\sum_{i=1}^{l} \omega_i \lambda_i = s$.

### 4.5. Policy Update

In this subsection, we describe the access policy update in detail. When a patient needs to convert a current access structure $(M, \rho, Z)$ to a new one $(M', \rho', Z')$, he only needs to use the encryption information $En(m)$ and run the $DK_mGen$ algorithm to construct the update keys and send them to the PHR cloud server. The PHR cloud server will run the $CUpdate$ algorithm to update the ciphertext. The algorithm is defined as follows.

1) $DK_mGen$: This algorithm is run by the patient. Taking as input the encryption information $En(m)$ of $m$, the current policy $(M, \rho, Z)$ and the new policy $(M', \rho', Z')$, it

outputs the dynamic policy-update key $DK_m$. $M'$ is a new $l' \times n'$ matrix with $\rho'$ mapping the rows to the attribute names , and $Z'$ is the new attribute values.

Step 1: It first executes the policy-comparison algorithm to compare the new access policy $(M', \rho', Z')$ with the current one $(M, \rho, Z)$, and outputs three sets of row indices $A'_1$,$A'_2$, and $A'_3$ of $M'$. Let $n_{\rho(i),M}$ and $n_{\rho(i),M'}$ denote the numbers of attribute names $\rho(i)$ in $M$ and $M'$, respectively. $A'_1$ and $A'_2$ represent the sets of indices $j$ where $\rho(j)'$ exists in $M$ and $\rho(i) = \rho(j)'$. If $n_{\rho(j)',M'} \leq n_{\rho(j)',M}$, the indices $j$ are put into $A'_1$. If $n_{\rho(j)',M'} > n_{\rho(j)',M}$, will include the indices $j$ that exceed $(n_{\rho(j)',M'} - n_{\rho(j)',M})$. $A'_3$ denotes the set of indices $j$ such that $\rho(j)$ is a new attribute name.

Step 2: The $DK_mGen$ algorithm chooses a random vector $v' \in Z_p^{n'}$ and $s$ is the first entry. Let $\lambda'_j = M'_j \cdot v'$, and $M'_j$ is the $j$'th row of $M'$. For each $j \in [1, l']$, the update key can be divided into three types.

(1) Type 1: If $(j, i) \in A'_1$, the update key will be $DK = (DK = h^{\lambda'_j - \lambda_i})$,and let $q'_j = q_i$:

(2) Type 2: If $(j, i) \in A'_2$,the algorithm chooses $x_j, q'_j \in Z_p$ randomly and computes the update key $DK = (x_j, DK = h^{\lambda'_j - x_j \lambda_i})$;

(3) Type 3: If $(j, i) \in A'_3$,the algorithm chooses $q'_j \in Z_p$ and computes the update key $DK = (DK(1) = h^{\lambda'_j}(T_{\rho'(j)}^{Z'_\rho(j)})^{-q'_j}, DK(2) = g^{q'_j})$.

Step 3: Finally, the patient sends the update key $DK_m = \{(DK)_{(j,i)\in A'_1}$, $(DK)_{(j,i)\in A'_2}, (DK)_{(j,i)\in A'_3}\}$to the PHR cloud server.

2) $CUpdate$: This algorithm is run by the PHR cloud server. Taking as input the current ciphertext $C$ and the dynamic policy-update key $DK_m$ , it outputs the new ciphertext $C'$.

(1) Type 1: For each $j \in A'_1$, it computes the ciphertext elements

$$C'_{2,j} = C_{2,i} \cdot DK = h^{\lambda'_j}(T_{\rho'(j)}^{Z'_\rho(j)})^{-q'_j} \tag{11}$$

$$C'_{3,j} = C_{3,j} = g^{q'_j} \tag{12}$$

where $q'_j = q_i$;

(2) Type 2: For each $j \in A'_2$, it computes

$$C'_{2,j} = (C_{2,i})^{x_j} \cdot DK = h^{\lambda'_j}(T_{\rho'(j)}^{Z'_\rho(j)})^{-q'_j} \tag{13}$$

$$C'_{3,j} = g^{q'_j} \tag{14}$$

where $q'_j = x_j q_i$;

(3) Type 3: For each $j \in A'_3$, it computes

$$C'_{2,j} = DK(1) = h^{\lambda'_j}(T_{\rho'(j)}^{Z'_\rho(j)})^{-q'_j} \tag{15}$$

$$C'_{3,j} = DK(2) = g^{q'_j} \tag{16}$$

Finally, the new ciphertext is $C' = \{C_0, C_1, (C'_{2,j}, C'_{3,j})_{j\in(1,\ldots,l')}\}$

## 5. Security Analysis

In this section, our scheme is demonstrated to be secure against chosen-plaintext attack under the standard model. The specific certification process is as follows.

Definition 2: Our scheme is secure in the selective-access structure and chosen-plaintext attack game if the decisional q-PBDHE assumption holds. Then, no polynomial-time adversary can break our scheme with a challenge structure $(M^*, \rho^*, Z^*)$.

Initialization: The challenger initiates a q-parallel BDHE challenge $(y, T)$. The adversary submits the challenge structure $(M^*, \rho^*, Z^*)$, where $M^*$ is an $l^* \times n^*$ matrix and $l^*, n^* \leq q$.

Global Setup: The challenger chooses random $m \in Z_p$ and computes $h = g^m$. Then it sends the public parameter $PP = (e, p, g, h, G, G_T)$ to $A$.

Authority Setup: For every $AA_i$, it chooses random $\alpha_i' \in Z_p$ and sets $\alpha_i = \alpha_i' + a^{q+1}$. Then, it computes

$$A_i = e(g,g)^{\alpha_i} = e(g,g)^{\alpha_i'} e(g^a, g^{a^q}) \tag{17}$$

Let $X$ be the set of the indices $i$ with $\rho^*(i) = x$ for $i = 1, ..., l^*$. For each attribute name $a_i$ with $\rho^*(i) = x$,it chooses random $r_i \in Z_p$ and computes

$$R_i = g^{r_i} \prod_{i \in X} g^{aM_{i,1}^*/b_i} \cdot g^{a^2 M_{i,2}^*/b_i} \cdot ... \cdot g^{a^{n^*} M_{i,n^*}^*/b_i} \tag{18}$$

For each attribute name $a_i$ with $\rho^*(i) \neq x$,it chooses random $r_i \in Z_p$ and computes $R_i = g^{r_i}$. For each attribute value $a_{i,n_i}$, it chooses random $t_{i,n_i} \in Z_p$ and computes $T_{i,n_i} = g^{t_{i,n_i}}$. The master secret key of $AA_i$ is $SK_i = \{a_i, r_i, (t_{i,n_i})_{\forall a_{i,n_i} \in A_i}\}$ and the public key is $PK_i = \{A_i, R_i, (T_{i,n_i})_{\forall a_{i,n_i} \in A_i}\}$. The challenger sends $PK_i$ to the adversary A.

Phase 1: The adversary $A$ can adaptively query the secret key for a set $S = (a_i : a_{i,t_i})$, where $S$ does not satisfy $(M^*, \rho^*, Z^*)$. The challenger first checks the set $S$. For each attribute name $a_i$ with $\rho^*(i) = x$, it computes

$$D_i = g^{\alpha_i'} g^{a^{q+1}} g^{mr_i} \prod_{i \in X} g^{aM_{i,1}^*/b_i} \cdot g^{a^2 M_{i,2}^*/b_i} \cdot ... \cdot g^{a^{n^*} M_{i,n^*}^*/b_i} \tag{19}$$

For each attribute value $a_{i,t_i}$, it computes

$$R_{i,t_i} = (T_{i,n_i}^{a_{i,t_i}})^{r_i} \tag{20}$$

For each attribute name $a_i$ with $\rho^*(i) \neq x$, it computes

$$D_i = g^{\alpha_i} h^{r_i} = g^{\alpha_i'} g^{a^{q+1}} g^{mr_i} \tag{21}$$

For each attribute value $a_{i,t_i}$, it computes $R_{i,t_i} = (T_{i,n_i}^{a_{i,t_i}})^{r_i}$. Then the challenger sends the privacy key $SK_A = \{D_i, R_{i,t_i}\}_{a_{i,t_i} \in S \cap A_i}$ to the adversary.

Challenge: The adversary submits two equal-length messages $m_0$ and $m_1$. The challenger chooses random $c \in \{0, 1\}$, and computes

$$C_0 = m_c \cdot T \cdot \prod\nolimits_i e(g,g)^{\alpha_i' S} \tag{22}$$

$$C_1 = g^S \tag{23}$$

Then it chooses $v^* = (s, sa + v_2^*, sa^2 + v_3^*, ..., sa^{n-1} + v_{n^*}^*) \in Z_p^{n^*}$ and $q_1^*, ..., q_{l^*}^* \in Z_p$. For $i = 1, ..., n^*$, we define $H_i$ as the set of all $i \neq j$ such that $\rho^*(i) = \rho^*(j)$. The challenge ciphertext elements are

$$\begin{aligned} C_{2,i} &= (T_{\rho^*(i)}^{Z_{\rho^*(i)}})^{q_i^*} (\prod_{j=1,...,n^*} (h)^{M_{i,j}^* v_j^*})(g^{b_i s})^{-t_{\rho^*(i)}} (\prod_{k \in H_i} \prod_{j=1,...,n^*} (g^{a^j s \cdot (b_i/b_k)})^{M_{k,j}^*}) \\ &= T_{\rho^*(i)}^{Z_{\rho^*(i)}})^{q_i^*} (\prod_{j=1,...,n^*} (g^m)^{M_{i,j}^* v_j^*})(g^{b_i s})^{-t_{\rho^*(i)}} (\prod_{k \in H_i} \prod_{j=1,...,n^*} (g^{a^j s \cdot (b_i/b_k)})^{M_{k,j}^*}) \end{aligned} \tag{24}$$

$$C_{3,i} = g^{-q_i^*} g^{-s b_i} \tag{25}$$

The ciphertext $C = \{C_0, C_1, (C_{2,i}, C_{3,i})_{i \in (1,...,l^*)}\}$ is given to $A$.

Phase 2: Phase 1 is repeated.

Guess: outputs his guess $c'$ on $c$. The challenger outputs $\theta = 0$ to guess that $T = e(g,g)^{a^{q+1} s}$ if $c' = c$. Therefore, the adversary can output $c' = c$ with the advantage $Pr[c' = c|\theta = 0] = 1/2 + \varepsilon$. Otherwise, it outputs $\theta = 1$ to guess that T is a random group element in $G_T$, and the advantage is $Pr[c' = c|\theta = 1] = 1/2$.

Therefore, the adversary can break the decisional q-PBDHE assumption with a negligible advantage $|\frac{1}{2} Pr[c' = c|\theta = 0] + \frac{1}{2} Pr[c' = c|\theta = 1] - \frac{1}{2}| = \frac{1}{2} \times \frac{1}{2} + \frac{1}{2} \times (\frac{1}{2} + \varepsilon) - \frac{1}{2} = \frac{1}{2}\varepsilon$.

## 6. Performance Analysis

As shown in Table 3 and Table 4, we compare our scheme with the previous methods [8, 10, 11, 15-17, 19] in terms of system function, communication cost and security model. The function includes whether it supports multiple authorities, privacy protection and dynamic policy updating. The communication cost refers to the sizes of the user's secret keys, ciphertext and dynamic update keys. Let $S$ be the number of user attributes, $D$ be the number of CAs, $N$ be the number of AAs, $n$ be the number of attributes of the system,$\rho$ be the bit length of the PHR user's GID, $l$ be the number of attributes encrypted by the patient, $I_c$ be the number of $AA_i$ encrypted by the patient, $k$ be the number of attributes decrypted by the PHR user, and $n_i$ be the number of attributes managed by $AA_i$. The decrypted cost is the number of bilinear pairing operations, and $l_1'$, $l_2'$, and$l_3'$ are the numbers of attributes of types 1, 2, and 3, respectively.

From Table 3, we can see that our scheme supports the protection of attribute privacy and dynamic policy updating. The schemes in [8, 10, 11] have privacy protection but they can't support policy updating. In the schemes in [8, 12, 17], only the privacy of the GID has been considered. Obviously, they can't protect the privacy of attributes. The whole access policy is hidden to protect privacy in the scheme in [10]. The scheme in [11] can protect the privacy of the GID and attributes. This is the first scheme that considers the privacy of attributes. However, it has been proved in the scheme in [12] that the scheme proposed in [11] can't effectively protect the user's attributes. In our scheme, not only the privacy of GID has been considered, but also the attribute values are hidden in the access structure to prevent the attributes from being revealed to others. In terms of security, the

**Table 3.** Comparison of Function and Security Model

| Scheme | Function | | | Security model |
|---|---|---|---|---|
| | Multi-authority | Privacy protection | Policy updating | |
| Scheme [8] | $\surd$ | Protect user's GID | $\times$ | Random Oracle |
| Scheme [10] | $\surd$ | Hide the access policy to protect privacy | $\times$ | Standard |
| Scheme [11] | $\surd$ | Protect user's GID and attribute | $\times$ | Standard |
| Scheme [15] | $\surd$ | Protect user's GID | $\surd$ | Random Oracle |
| Scheme [16] | $\times$ | Protect user's attribute | $\surd$ | Standard |
| Scheme [17] | $\surd$ | Protect version number | $\surd$ | Standard |
| Scheme [19] | $\surd$ | Protect user's GID | $\surd$ | Standard |
| Our Scheme | $\surd$ | Protect user's GID and attribute | $\surd$ | Standard |

schemes in [8,15] are only secure under the random oracle model. Our scheme and the schemes in [10, 11, 16, 17, 19] are secure under the standard model.

**Table 4.** Comparison of Communication Cost

| Scheme | Communication cost | | | | | |
|---|---|---|---|---|---|---|
| | The size of user's privacy key | The size of ciphertext | The size of dynamic update key | | | Decryption cost |
| | | | Type 1 | Type 2 | Type 3 | |
| Scheme[8] | $S$ | $3l+1$ | $\times$ | $\times$ | $\times$ | $2k$ |
| Scheme[10] | $4S+4\rho+1$ | $4n+4\rho+2$ | $\times$ | $\times$ | $\times$ | $\sum_{i\in[N]} 4n_i+4\rho+2$ |
| Scheme[11] | $S+6$ | $(2l+3)I_c+1$ | $\times$ | $\times$ | $\times$ | $2k+5I_c$ |
| Scheme[15] | $S$ | $3l+1$ | $2l'_1$ | $3l'_2$ | $3l'_3$ | $2k$ |
| Scheme[16] | $S+2$ | $2(2l+2)$ | $5l'_1$ | $6l'_2$ | $4l'_3$ | $4k+2$ |
| Scheme[17] | $2S$ | $3l+2$ | $2l'_1$ | $3l'_2$ | $3l'_2$ | $2k+1$ |
| Scheme[19] | $S+D(N+2)$ | $2l+2$ | $l'_1$ | $2l'_2$ | $2l'_3$ | $2k+1$ |
| Our Scheme | $2S+1$ | $2l+2$ | $l'_1$ | $2l'_2$ | $2l'_3$ | $2k+2I_c$ |

The complexity comparison of communication cost is presented in Table 4. The user's private key in the schemes in [8, 11, 15, 16] is relatively small, but the ciphertext is relatively long. The whole communication cost is high in the scheme in [10]; particularly, the ciphertext length is related to the number of attributes of the system, while in the other schemes, it is related to the number of encrypted attributes in the access policy. Considered the policy updating, the schemes in [8, 10, 11] do not provide the policy-updating function, and the update keys in our scheme are shorter than those in [15, 16, 17] and the same length as those in the scheme in [19]. Moreover, the decryption cost of the scheme in [10] is related to the number of system attributes and user GID length, and the cost is higher. In [11] and in our scheme, the decryption cost is related to the number of encrypted attributes and the number of encrypted attribute authorities, but our

scheme has $3I + c$ lower cost than that in [11]. Compared with the scheme in [19], the length of the user's keys in our scheme is much smaller, because it is only related to the number of user attributes. Instead, the user's private keys in [19] are generated by the $D$ central authorities and $N$ attribute authorities. The product of $D$ and $N$ is multiples of $S$, therefore, the length of the user's keys in our scheme is much smaller. The size of ciphertext and dynamic update key are same. As a whole, it is showed that our scheme has optimized in terms of both the communication and computational costs.

Based on the comprehensive analysis, our PHR system scheme can protect the patient's attributes and support access-policy dynamic updating. Moreover, the proposed scheme also has certain advantages in performance when compared with other schemes. It is applicable to the PHR system, which needs to protect the privacy of the user and update the access policy.

## 7. Conclusions

To protect the patient's attribute privacy and support policy updating in the PHR cloud environment, a privacy-preserving multi-authority attribute-based encryption scheme with dynamic policy updating in PHR is proposed. A semi-policy-hidden technology is introduced to protect the privacy of the patient's attributes. The access policy of ciphertext updating is divided into three types, and for each type, there is a method to update the policy. In this paper, a method of access-policy dynamic updating is designed, with which the patient only needs to send the update key to the PHR cloud server. The implementation delegates the most computationally intensive jobs to the PHR cloud server, so the communication cost and computational cost of our system are greatly reduced. The users not only require strong functioning of the system, but also high performance in the big-data environment. Therefore, a secure MA-ABE scheme with shorter ciphertext, shorter private keys and richer expression ability would be worthwhile to investigate in our future work.

**Acknowledgments** This work was supported in part by the "13th Five-Year" National Crypto Development Foundation under Grant MMJJ20170122, in part by the National Natural Science Foundation of China under Grant 61802117, and in part by the Projects of Henan Provincial Department of Science and Technology under Grant 192102210280.

## References

1. Sahai, A, Waters, B.: Fuzzy identity-based encryption. Advances in Cryptology - EUROCRYPT 2005. Berlin, Heidelberg, Springer - Verlag, 457-473. (2005)
2. Goyal, V, Pandey, O, Sahai, A, et al.: Attribute-based Encryption for Fine Grained Access Control of Encrypted Data. Proceedings of the ACM Conference on Computer and Communications Security (CCS 2006). ACM Press, 89-98. (2006)
3. Agrawal, S, Boyen, X, Vaikuntanathan, V, et.al.: Functional Encryption for Threshold Functions (or Fuzzy IBE) from Lattices. Proceedings of Public Key Cryptography (PKC 2012). Berlin: Springer-Verlag, 280–297. (2012)
4. Boyen, X.: Attribute-Based Functional Encryption on Lattices. Proceedings of the 10th theory of cryptography conference on Theory of Cryptography. Berlin: Springer-Verlag, 122-142. (2013)

5. Zhang, L, Wu, Q, Mu, Y, et al.: Privacy-Preserving and Secure Sharing of PHR in the Cloud. Journal of Medical Systems, Vol. 40, No. 12, 1-13. (2016)
6. Chase, M.: Multi-authority attribute based encryption. Proceedings of Theory of Cryptography Conf. (TCC '07), S.P. Vadhan, ed., 515-534. (2007)
7. Chase, M, Chow, S.: Improving privacy and security in multi-Authority attribute-Based encryption. Proceedings of the 16th ACM confenence of computer and communication Security. E. Al-Shaer, S. Jha, and A.D. Keromytis, eds., 121-130. (2009)
8. Lewko, A, Waters, B.: Decentralizing attribute - based encryption. Proceedings of Eurocrypt 2011: Proc. 30th Ann. Int'l Conf. Theory and Applications of Cryptographic Techniques: Advances in Cryptology, K.G. Paterson, ed., 568-588 . (2011)
9. Han, J, Susilo, W, Mu, Y, et al.: Privacy-Preserving Decentralized Key-Policy Attribute-Based Encryption. IEEE Transactions on Parallel & Distributed Systems, Vol. 23, No. 11, 2150-2162. (2012)
10. Xhafa, F, Feng, J, Zhang, Y, et al.: Privacy-aware attribute-based PHR sharing with user accountability in cloud computing. Journal of Supercomputing, Vol. 71, No. 5, 1607–1619. (2015)
11. Han, J, Susilo, W, Mu, Y, et al.: Improving privacy and security in decentralized ciphertext-policy attribute-based encryption. IEEE Transactions on Information Forensics and Security, Vol. 10, No. 3, 665-678. (2015)
12. Wang, M, Zhang, Z, Chen, C.: Security analysis of a privacy-preserving decentralized ciphertext-policy attribute-based encryption scheme. Concurrency & Computation Practice & Experience, Vol. 28, No. 4, 1237-1245. (2016)
13. Qian, H, Li, J, Zhang, Y, et al.: Privacy-preserving personal health record using multi-authority attribute-based encryption with revocation. International Journal of Information Security, Vol. 14, No. 6, 487-497. (2015)
14. Xia, Z, Zhang, L, Liu. D.: Attribute-based access control scheme with efficient revocation in cloud computing. China Communications, Vol. 13, No. 7, 92-99. (2016)
15. Sahai, A, Seyalioglu, H, Waters, B.: Dynamic credentials and ciphertext delegation for attribute-based encryption. Advances in Cryptology – CRYPTO 2012. Springer Berlin Heidelberg, 199-217. (2012)
16. Yang, K, Jia, X, Ren, K.: Secure and verifiable policy update outsourcing for big data access control in the cloud. IEEE Transactions on Parallel and Distributed Systems, Vol. 26, No. 12, 3461-3470. (2015)
17. Ying, Z, Ma, J, Cui, J.: Partially policy hidden CP-ABE supporting dynamic policy updating. Journal on Communications. Vol. 36, No. 12, 178-189. (2015)
18. Li, H, Liu, D, Alharbi, K, et al. Enabling fine-grained access control with efficient attribute revocation and policy updating in smart grid[J]. KSII Transactions on Internet and Information Systems, Vol, 9, No. 4, 1404-1423. (2015)
19. Wu, G.: Multi-Authority CP-ABE with policy update in cloud storage. Journal of Computer Research and Development. Vol. 53, No. 10, 2393-2399. (2016)
20. Ying, Z, Li, H, Ma, J, et al.: Adaptively secure ciphertext-policy attribute-based encryption with dynamic policy updating. Science China-Information Sciences, Vol. 59, No. 4, 1-16. (2016)
21. Liu ZC, Jiang ZL, Wang X, et al.:Practical attribute-based encryption: outsourcing decrtyption, attribute revocation and policy updating. Journal of Network and Computer Applications, Vol. 108, 112-123.(2018)
22. Jiang YH, Susilo W, Mu Y, et al.:Ciphertext-policy attribute-based encryption supporting access policy update and its extension with preserved attributes. International Journal of Information Security, Vol. 17, No. 5, 533-548.(2018)
23. Lai, J, Deng, H, Li, Y.: Expressive CP-ABE with partially hidden access structures. Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security. ACM, 18-19. (2012)

**Xixi Yan**, received her Ph.D. degree from Beijing University of Posts and Telecommunications. She is currently an associate professor in the School of Computer Science and Technology, Henan Polytechnic University. She is mainly engaged in the digital content security and cloud computing.

**Hao Ni**, received his Bachelor of Engineering degree from Henan Polytechnic University. At present, he is a postgraduate of Henan Polytechnic University, and mainly engaged in Network and Information Security, Cryptography.

**Yuan Liu**, received her Bachelor of Engineering degree from Henan Polytechnic University. At present, she is a doctoral student of Beijing University of Posts and Telecommunications, and mainly engaged in Network and Information Security, Cryptography.

**Dezhi Han** (corresponding author), received the Ph.D. degree from Huazhong University of Science and Technology. He is currently a professor of computer science and engineering at Shanghai Maritime University. His research interests include cloud computing, mobile networking and cloud security.

*Received: August 30, 2018; Accepted: May 25, 2019.*