Key-Policy Weighted Attribute Based Encryption for Fine-Grained Access Control

Ximeng Liu, Hui Zhu
School of Telecommunications Engineering
Xidian University
Xi'an, China
snbnix@gmail.com

Jianfeng Ma, Jun Ma
School of Computer Science and Technology
Xidian University
Xi'an, China
jfma@mail.xidian.edu.cn
sijunhan@163.com

Siqi Ma
School of information system
Singapore Management University
Singapore
siqi.ma.2013@phdis.smu.edu.sg

***Abstract*—When we step into the new internet era, large numbers of new techniques have emerged in order to make our life better. However, these new techniques require some new properties in order to keep personal information confidentially which the traditional encryption method cannot catch up with. Recently, a new encryption primitive called Attribute Based Encryption(ABE) have appeared because it can achieve both information security and fine-grained access control. Although the ABE scheme has these new characters which can keep the information security that can fit for the new widely used technique, nevertheless, the characteristic of attributes are treated in the identical level in most of traditional schemes. In the real scenario, the importance of each attributes is always different. In this paper, we propose a scheme called Key-Policy Weighted Attribute based Encryption (KP-WABE) while the attributes have different weights according to their importance in the system. The KP-WABE scheme is proved to be secure under the l-th Bilinear Diffie-Hellman Inversion Assumption. Our scheme can be considered as the generalization of traditional KP-ABE scheme when all attributes have equal weights.**

***Keywords*-Bilinear pairings; Attribute-based; Access structure; Weighted attribute; Encryption.**

I. Introduction

In order to achieve confidence of the communications, encryption is considered as one of the most widespread used mechanism to protect the transmitted message. As many new application break out in recent year, new challenges are appear in order to protect the confidence of information. As we stay in the age of cloud computing era, many companies outsource their storage ability to storage private information. However, the cloud sever is business company which cannot be fully trusted. Also, due to the character of cloud computing, different users want to share their document to a group of users which satisfy some specific policy. It is hard by achieve these character both conveniently at same time by using the traditional methods. Recently, attribute based cryptography has attracted much attention as a new public key primitive. Attribute Based Encryption (ABE) has great advantage over the traditional PKC due to it can achieve both information security and fine-grained access control. When data provider wants to share some information with user by using the traditional method, the provider must know the exactly one whom he/she want to be shared with. ABE scheme can can solve this problem easily by achieving flexible one-to-many encryption instead of one-to-one. In this scheme, the ciphertext is labeled with a set of attributes defined for the system users or access structure. The particular user hold private key can decrypt the particular ciphertext only if the two match. For example, the headmaster wants to encrypt a document to the entire professor with age above 45 in the computer science (CS) department. In this situation, the document would be encrypted to the attribute subset {"professor", "CS department", "age above 45"}, and only if users hold the private key contain these three attributes can decrypt the document while others cannot.

In this paper, we propose a scheme called Key-Policy Weighted Attribute Based Encryption(KP-WABE) scheme. In the real circumstance, the attributes are not always in the same position. Different attributes have different importance in the system. Every data receiver holds a set of weighted attributes. The data owner wants to encrypt a data to all the receivers who have a certain set of weighted attributes. In the KP-WABE scheme, data receiver's private key has certain kinds of weighted access structure. In order to decrypt the message successfully, the ciphertext with a set of weighted attribute must satisfy the weighted access structure. We will give an example to explain why attribute is very important in the practical scenario. For instance, the headmaster wants both staff and assistant professor in CS department to decrypt the document. We can give staff and assistant professor with weight 1 and 2, denote "employee(1)" and "employee(2)" respectively. Both two employees can decrypt the document with access structure {"employee(1)" AND "CS department} while staff can not decrypt the the document with access structure {"employee(2)" AND "CS department}. Also, one attribute may have different nickname in the system. For example, both "Kitty" and "Pussycat" represent a little cat. We can use "little cat(1)" to represent them. In all, the weight can represent the importance of the attribute in the system. Also, a attribute with different nicknames can be represented by one attribute with same weight. The notion of KP-WABE can be considered as the generalization of traditional KP-ABE scheme while all attributes are in the same level. In KP-WABE, the universal attributes have different weighted according to their importance defined in the access control system. This

978-1-4799-4640-2/14/$31.00 ©2014 IEEE

kind of ABE could be used in distributes systems which our scheme is more appropriate for the actual scenario.

A. Related Work

The ABE scheme is considered as one of the important cryptography primitive in the cloud computing era. As one of the application of fuzzy identity-based encryption(FIBE)[1](also called threshold attribute based encryption, t-ABE), it original is come from identity-based encryption[2][3]. In order to achieve the error tolerance property, Sahai and Waters introduce the fuzzy identity into the identity-based encryption system. So as to let the authorized user to decrypt the message correctly, intersection between the identity in the ciphertext and the identity in the private key must exceed a threshold. But this scheme, it can only support threshold which can not achieve fine-grained access control. Later, Goyal et al.[4] turned t-ABE into a fine-grained level by using the access structure. In this scheme, Ostrovsky combine user's private key with an access tree while the ciphertext is associated with an attribute set. The user can decrypt the ciphertext if and only if the attribute set satisfied the access tree. This scheme is called Key-Policy Attribute Based Encryption(KP-ABE) in short. Another kinds of ABE scheme is called Ciphertext-Policy Attribute Based Encryption(CP-ABE)[5] [6]. It performs in the reverse way: the ciphertext is associated with the access structure while the private key is associated with a set of attribute. Bethencourt et al.[5] proposed the first construction of the CP-ABE scheme. In this paper, the authors constructed the CP-ABE Toolkit based on the Pairing Based Cryptography (PBC) library. They also used this toolkit to test the preference of the CP-ABE scheme proposed in this paper. Recently, Waters [7] proposed a efficient and provably secure CP-ABE realization by using the Linear Secret Sharing Scheme (LSSS) matrix. He used LSSS matrix instead of access tree proposed in the previous scheme due to LSSS matrix can express any any attribute access structure. There exist some drawbacks in the scheme mention above. One is that the length of ciphertext is increase linearly with the number of attributes. When the number of attributes involved in the system is very large, it can greatly increase the transmitting cost. In order to solve this problem, threshold attribute based encryption scheme[8] and CP-ABE scheme[9] with constant size were proposed in these papers. In order to discuss the relationship between the attribute, Li et al. [10] proposed a scheme called ABE with attribute hierarchy which the attributes in this scheme were classified into trees according to their relationship. Due to it can achieve the fine-grained control level, many application can be realize by using the ABE scheme. Yu et al.[11] use KP-ABE scheme to encrypt the message in order to achieve fine-grained data access control in cloud computing environment. Li et al.[12] use the multiple-authority ABE scheme which allow patients to outsource own personal health record(PHR) to the cloud sever and let they control the PHR by themselves. due to it can achieve both fine-grained access control and data integrity. In order to indicate the importance of the attributes, Liu et al.[13] proposed a scheme called Ciphertext-Policy Weighted Attribute Based Encryption (CP-WABE) which introduce the weighted attribute into the ABE system. However, the computation cost is high and the ciphertext is long which is not suitable for practical scenario.

B. Our contribution

We will list the contribution of this paper: (1)We formalize the KP-WABE model and its corresponding security model for KP-WABE. (2)We give the specific construction of the KP-WABE scheme (3)We make analysis of KP-WABE scheme to show our scheme is secure base on a hard problem describe in this paper

C. Organization

The rest of this paper is organized as follow: In section 2, we give the formal model of the KP-WABE scheme and its security model. In section 3, some basic concept will be introduced in this section. It is important because we use this basic concept to construct the scheme and prove the security of KP-WABE scheme. In section 4, the specific construction of KP-WABE scheme is given in this section. In section 5, we give security proof of the KP-WABE scheme in the standard model by using the hard assumption. Section 7 we conclude this paper.

II. KP-WABE scheme and Security model

In this section, we introduce KP-WABE scheme and its security model.

A. KP-WABE scheme

A KP-WABE scheme consists four fundamental algorithms: *Setup*, *KeyGen*, *Encrypt* and *Decrypt*.

Setup$(1^\lambda, U)$: The setup algorithm is run by central authority which inputs security parameter 1^λ and weighted attribute universe description U. It will generate the public parameters pms and master key MSK as output after the *Setup* algorithm run completely.

KeyGen(pms, MSK, Γ): The key generation algorithm is run by central authority which takes public parameters pms, master key MSK and weighted access structure Γ as input. It will generate a private key SK which contains Γ as output after executing the *KeyGen* algorithm.

Encrypt(pms, S'_0, m): This encryption algorithm is run by a user who wants to protect the message which takes the public parameters pms, the message m and a set S'_0 of weighted attributes S as input. It will generate a ciphertext CT as output after executing the *Encrypt* algorithm.

Decrypt(CT, SK): The decryption algorithm is run by decryptor which takes the ciphertext CT and the private key SK as input. If a set of weighted attribute satisfies the weighted access structure, it will generate the ciphertext and return message m as output after executing the *Decrypt* algorithm.

In the KP-WABE scheme, the ciphertext is related with a weighted attribute set while the private key is associated with the weighted access structure. In our security model, the

adversary will choose a challenged access structure Γ^* and ask for any private key SK contain weighted attribute set S which does not satisfy Γ^*. We now give the formal security game for KP-WABE scheme below.

B. Security model for KP-WABE

Init. The adversary first declares a challenged weighted attribute set S which he wants to be challenged upon.

Setup. The challenger runs the *Setup* algorithm and outputs the public parameter pms. The pms will given to the adversary after executing *Setup* algorithm successfully.

Phase 1. In this phase, the adversary will make polynomially private key queries for many weighted access structures Γ_j repeatedly. It is worth noting that S is not satisfied any weighted access structure Γ_j.

Challenge. After finishing Phase 1, the adversary $\mathcal{A}$ will submits two equal length message $m_0, m_1 \in \mathcal{M}$ to the challenger. Then, challenger flips a random coin β and encrypts m_β with S. After flipping the coin, the encrypted message is given to the adversary $\mathcal{A}$.

Phase 2. Same as phase 1.

Guess. The adversary outputs a guess whether $\beta' = \beta$ or not.

If $\beta' = \beta$, we say the adversary $\mathcal{A}$ wins this game. We defined the advantage of $\mathcal{A}$ as $\mathrm{Adv}_{\mathcal{A}} = \Pr[\beta' = \beta] - \frac{1}{2}$ in this game.

Definition 1. *We said that our KP-WABE scheme is IND-sAtr-CPA secure if there does not exist polynomial time adversaries which have non-negligible advantage win the game above.*

III. Preliminaries:

In this section, we introduce some notions called bilinear maps, l-th bilinear Diffie-Hellman inversion assumption and weighted access tree which we use to construct and prove the KP-WABE scheme.

A. Bilinear Maps

Denote $\mathbb{G}$ and $\mathbb{G}_T$ as two cyclic groups of prime order p with the multiplication, g is a generator of $\mathbb{G}$ and e is called a bilinear map. Let $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a bilinear map which the following properties hold:

1) Bilinearity: $e(u^a, v^b) = e(u, v)^{ab}$ for all $u, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}_p$.
2) Non-degeneracy: $e(g, g) \neq 1$.
3) Computability: There exists a algorithm to compute bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ efficiently.

Notice that the map e is symmetric since $e(u^a, v^b) = e(u, v)^{ab} = e(u^b, v^a)$.

B. l-th Bilinear Diffie-Hellman Inversion Assumption

We used l-th bilinear Diffie-Hellman inversion assumption which is introduce in [14]. The l-th bilinear Diffie-Hellman inversion problem define as follows:

Denote g and h are two random generators of $\mathbb{G}$ and choose α as a random number from Z_p^*. Assume the adversary is given $\mathbf{y} = (g, h, g^\alpha, \cdots, g^{\alpha^\omega})$ where $g^{\alpha^i} \in \mathbb{G}$. It will still remain hard to distinguish $T = e(g, h)^{\alpha^{\omega+1}} \in \mathbb{G}_T$ from a random element in $\mathbb{G}_T$.

An algorithm $\mathcal{B}$ outputs $z \in \{0, 1\}$ in solving l-wBDHI* in $\mathbb{G}_T$ with advantage ϵ if

$$\left|\Pr\left[\mathcal{B}\left(\mathbf{y}, T = e(g,h)^{\alpha^{\omega+1}} = 0\right)\right] - \Pr\left[\mathcal{B}\left(\mathbf{y}, T = R\right) = 0\right]\right| \geq \epsilon$$

We say that the l-th bilinear Diffie-Hellman inversion assumption holds if there exist a polynomial time algorithm which can solve the l-th bilinear Diffie-Hellman inversion problem with a negligible advantage.

C. Access structure and weighted access tree

1) Access structure:

Definition 2 (Access structure[15])**.** *We define that $\{P_1, P_2, \cdots, P_n\}$ is a set of parties. A collection $\Gamma \subseteq 2^{\{P_1, \cdots, P_n\}}$ is monotone for all the B, C if it satisfies the following conditions: if $B \in \Gamma$ and $B \subseteq C$ then $C \in \Gamma$. An access structure is a collection Γ of non-empty subsets of $\{P_1, P_2 \cdots, P_n\}$, that is to say, $\Gamma \subseteq 2^{\{P_1, \cdots, P_n\}} \backslash \{\emptyset\}$. We say that sets in Γ are called the authorized sets while others not in Γ are called the unauthorized sets.*

2) Weighted access tree:

In this subsection, we will define a weighted access tree Γ. All the non-leaf node of Γ represents a threshold gate. We denote that num_x is represented as the number of children of a node x, then the threshold value k_x is belong to $0 < k_x < num_x$. We give two special case of the threshold gate call OR gate and AND gate. When $k_x = 1$, it represent OR gate and $k_x = num_x$ otherwise. Each leaf node x of the tree is described by an attribute with weight. If the weight of attribute from the attribute set exceed the weight of the leaf node, we let value $k_x = 1$.

We will define some function in order to describe this work simplicity. The parent of the node x is denoted as $parent(x)$ in the weighted access tree. The function $att(x)$ is defined only if x is a leaf node which x is related with a weighted attribute. The order among the children of x are also defined from 1 to num_x. The function $index(x)$ returns such a number associated with the node x which are uniquely assigned to nodes in the weighted access tree for a given key in an arbitrary manner.

Satisfying a weighted access tree. We denote the symbol Γ as a weighted access tree with root r. Also, we define Γ_x as the subtree of Γ with root node x. If a set of weighted attributes S satisfies the access tree Γ_x, we denote it as $\Gamma_x(S) = 1$. In order to compute $\Gamma_x(S)$, we will compute it recursively as below: For all children x' of non-leaf node x, $\Gamma_x(S)$ returns 1 if and only if the number of x' return 1 exceed threshold of x. If x is a leaf node, then $\Gamma_x(S)$ returns 1 if and only if the weight of attribute λ_x from S great than or equal to the weight of the leaf node, that is $weight(\lambda_x) \geq weight(att(x))$.

IV. Our construction

In our construction, the attributes are assumed to be divided into n chains according to their weight. For each attribute

λ_i, we assume the length of the attribute chain is ω_i based on the weight of λ_i. Let $\lambda_{i,k}$ be an attribute of depth k with path $(\lambda_{i,1}, \lambda_{i,2}, \cdots, \lambda_{i,k}, \cdots, \lambda_{i,\omega_i})$. We show how these attributes categorized and construct in Figure 1. If all the attribute have same weight, our KP-WABE scheme can be convert into traditional KP-ABE scheme.

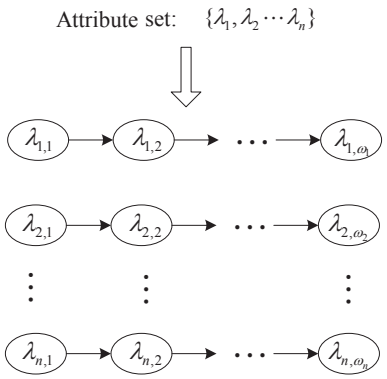

Fig. 1. weighted attributes set

The concrete construction is as follows:

Setup$(1^\lambda, S)$: In this algorithm, two important parameters will be inputted into the algorithm: security parameter 1^λ and weighted attribute set which contains all attributes in the system. Each attributes λ_i have weight $\omega_i = weight(\lambda_i)$ according to its importance in the system and we assume $\omega = \max\{\omega_1, \omega_2 \cdots \omega_l\}$. Then it will choose a group $\mathbb{G}$ with prime order p. Also, random group elements $g_2, u'_1, u'_2 \cdots u'_n, u_1 \cdots u_\omega$ will be choose from $\mathbb{G}$. These element are chosen according to $|\mathcal{U}|$ in the system. Finally, an exponent $\alpha \in Z_p$ is randomly chosen. The construction of public parameters pms is described as:

$$pms = \{g, g_1, g_2, g_3, e, (u'_i)_{1 \le i \le n}, (u_i)_{1 \le i \le \omega}\}$$

Also, the master secret key is described as

$$MSK = g_2^\alpha.$$

Encrypt(pms, S'_0, m): In this algorithm, three parameters are inputted into the algorithm: public parameters pms, weighted attributes set S'_0 and message m belong to $\mathbb{G}_T$. We use $(\lambda'_i, \lambda'_{i,1}, \cdots \lambda'_{i,\omega'_i})$ denote attribute chain for λ'_i in S'_0 with weight ω'_i. It computes $E' = me(g_1, g_2)^s$, $T = g^s$. For all $\lambda'_i \in S'_0$, it calculates $E_{\lambda'_i} = (u'_i \prod_{\delta=1}^{\omega'_i} u_\delta^{\lambda'_{i,\delta}} \cdot g_3)^s$.

The ciphertext is denoted as the following tuples:

$$CT = (E', T, \{E_{\lambda'_i}\}_{\lambda'_i \in S'_0}).$$

Keygen(pms, MSK, Γ): In this algorithm, three parameters are inputted into the algorithm: the pms, master key MSK and weighted access tree Γ. For each node x in the tree Γ, the algorithm will generate a polynomial q_x with degree $d_x = k_x - 1$. The k_x is represented as threshold value of that node x. This construction is from top-to-down manner. It first begin from root node r where the algorithm sets $q_r(0) = \alpha$. In order to define it completely, random d_r other points of the polynomial q_r should be chosen. In order to complete define q_x for any other node x, the algorithm set $q_x(0) = q_{\text{parent}}(index(x))$ and choose d_x other points randomly.

Finally, the algorithm will handle the set of the leaf nodes Y_0 in Γ which each leaf node is related with a weighted attribute. The key is construct as following:

1) For each $\lambda_i \in Y_0$, we denote that λ_i be an attribute of depth k_i with path $(\lambda_i, \lambda_{i,1}, \cdots \lambda_{i,k_i})$. Then CA chooses $r_i \in Z_p$ randomly and calculates $D_\lambda = \{d_{i0}, d_i, d_{i,k-1}, \cdots d_{i,\omega_i - 1}\}$, where $d_{i0} = g_2^{q_{\lambda_i}(0)} \cdot (u'_i \prod_{j=1}^{k_i} u_j^{\lambda_{i,j}} \cdot g_3)^{r_i}, d_i = g^{r_i}, d_{i,k_i+1} = u_{k_i+1}^{r_i}, \cdots, d_{i,\omega_i} = u_{\omega_i}^{r_i}$.

2) The private key is denoted as the following:

$$SK_{\Gamma_0} = \{D_\lambda\}_{\lambda \in \Gamma_0}.$$

Decrypt(CT, SK): Suppose that a ciphertext is encrypted with the access tree where ω'_i is the depth of leaf node. Assume user has a private key $SK_{\Gamma'_0} = \{D_{\lambda_i}\}_{\lambda_i \in \Gamma'_0}$ for attribute set S'_0 and k_i is the weight of attribute $\lambda_i \in S'_0$. If $k_i \ge \omega'_i$, the authorized user can calculate $d'_{i0} = d_{i0} \cdot d_{i,\omega'_i}^{\lambda_{i,\omega'_i}} \cdots d_{i,k_i}^{\lambda_{i,k_i}}$. The decrypt algorithm is calculate with down-to-top method. It first computes from leaf node of the access tree:

$$\frac{e(d_{i0}, T)}{e(d_i, E'_{\omega'})} = e(g, g_2)^{s \cdot q_x(0)}.$$

For each non-leaf node, the algorithm compute:

$$\begin{aligned} F_w &= \prod_{k \in S_w} F_k^{\Delta_{i,S'_w}(0)}, where \begin{matrix} i = index(k) \\ S'_w = \{index(k) : k \in S_w\} \end{matrix} \\ &= \prod_{k \in S_w} \left(e(g, g_2)^{s \cdot q_k(0)}\right)^{\Delta_{i,S'_w}(0)} \\ &= \prod_{k \in S_w} \left(e(g, g_2)^{s \cdot q_{paraent(k)}(0)}\right)^{\Delta_{i,S'_w}(0)} = e(g, g_2)^{s \cdot q_w(0)}. \end{aligned}$$

If the weighted attribute set in the ciphertext satisfies the access tree, we can successfully calculate the F_r:

$$F_r = e(g, g_2)^{s \cdot q_R(0)} = e(g, g_2)^{s \cdot \alpha}.$$

Finally, the algorithm divide out F_r from CT and obtain the message m.

V. Proof

Theorem 1. *We say that our KP-WABE scheme is IND-sAtr-CPA secure if the Bilinear Diffie-Hellman Inversion Assumption holds.*

Proof: We assume there exist an adversary $\mathcal{A}$ with non-negligible advantage $\epsilon = \text{Adv}_{\mathcal{A}}$ in the selective security game against KP-WABE scheme. We will show how can we build a simulator $\mathcal{B}$ to play the l-wBDHI* problem.

Init: In this stage, the selective weighted attribute set game begins with $\mathcal{A}$. It first outputs an attribute set S^* with attribute λ_i^* weight $k_i \le \omega$ that it intends to attack. $(\lambda_i^*, \lambda_{i,1}^*, \cdots \lambda_{i,k_i}^*)$ is attribute chain which represent attribute λ_i^* with weight k_i.

If $k_i \leq \omega$ then $\mathcal{B}$ pads with $\omega - k_i$ zeroes on the right to make a vector of length ω.

Setup: In order to generate the system parameters, algorithm $\mathcal{B}$ sets $g_1 = y_1 = g^{\alpha}$ and $g_2 = y_\omega \cdot g^{\gamma} = g^{\alpha^{\omega}+\gamma}$. Next, $\mathcal{B}$ picks $\gamma_1, \cdots, \gamma_\omega$ randomly and sets $u_j = g^{\gamma_j}/y_{\omega-j+1}$ for $j = 1, \cdots, \omega$. Algorithm $\mathcal{B}$ also picks a random δ in Z_p and set $u'_i = g^{\delta_i} \cdot \prod_{j=1}^{\omega} y_{\omega-j+1}^{\lambda^*_{i,j}}$.

Finally, $\mathcal{B}$ sends the public parameters $pms = \{g, g_1, g_2, u', u_1 \cdots u_\omega\}$ to $\mathcal{A}$. The master key can be calculated as $g_2^{\alpha} = g^{\alpha(\alpha^l+\gamma)} = y_{l+1}y_1^{\gamma}$.

Phase 1: In this phase, $\mathcal{A}$ answers up to polynomial q_s private key queries such that the challenge set does not satisfied Γ.

For simplicity, we first define the following two algorithm: PSat and PUnsat.

PSat(Γ_x, S, λ_x): In order to set up the polynomials for the root node of the sub-tree such that $\Gamma_x(S) = 1$, this algorithm will input a weighted access tree Γ_x along with a weighted attribute set S and an integer $\lambda_x \in Z_p$. In order to generate a polynomial q_x for root node, the algorithm chooses q_x of degree d_x such that $q_x(0) = \lambda_x$ with rest of point choose randomly. As long as the algorithm reached leaf node, it sets $q_x(0) = \lambda_x$. We can sets polynomials for each child node x' of x by calling the algorithm *PSat*$(\Gamma_{x'}, S, q_x(index(x'))$. where $q_{x'} = q_x(index(x'))$.

PUnsat$(\Gamma_x, S, g^{\lambda_x})$: In order to set up the polynomials for the root node of the sub-tree such that $\Gamma_x(S) = 0$, this algorithm will input a weighted access tree Γ_x along with a weighted attribute set S and an element $g^{\lambda_x} \in \mathbb{G}$ (where $\lambda_x \in_R Z_p$). Firstly, the algorithm chooses q_x of degree d_x such that $q_x(0) = \lambda_x$ with rest of point choose randomly. No more than d_x children of x are satisfied due to $\Gamma_x(S) = 0$. We assume that there exist h_x $(h_x \leq d_x)$ number of children of x satisfied the access tree. The algorithm then chooses a point $\lambda_{x'} \in_R Z_p$ and sets $q_x(index(x')) = \lambda_{x'}$ for each satisfied child node x' of x. Later, it will choose the $d_x - h_x$ points of q_x randomly to completely fix the q_x. In order to set polynomials for the rest of the nodes in the tree, the algorithm run call the following algorithm recursively as follows. For each child node x' of x:

1) It calls the algorithm PSat$(\Gamma_{x'}, S, q_x(\text{index}(x'))$ if x' is a satisfied node where $q_x(\text{index}(x'))$ is known in this case.

2) It calls the algorithm PUnsat$(\Gamma_{x'}, S, g^{q_x(\text{index}(x'))})$ if x' is not a satisfied node where only $g^{q_x(0)}$ can be known in this case.

In order to set a polynomial q_x for each node x of access structure Γ, the simulator first runs PUnsat (Γ, S, A) to give private keys for Γ. If x is satisfied for each leaf node x of Γ, we know q_x completely. If x is not satisfied, then at least $g^{q_x(0)}$ is known (in some cases q_x might be known completely). Furthermore, $q_r(0) = \alpha$.

For the leaf node, we simulate as the following two cases.

Case1: We simulate the attribute λ_i with weight $k_i \leq \omega$ where $\lambda_i \in Y_0$. Simulator first define the polynomial as $Q_x(\cdot) = \alpha q_x(\cdot)$. Attribute chain is denoted as $ATC_{\lambda_i} = (\lambda_i, \lambda_{i,1}, \cdots, \lambda_{i,u_i}) \in (Z_p^*)^{u_i}$ where $u_i \leq \omega'_i$. We say that only restriction is that ATC_{λ_i} is not $ATC^*_{\lambda_i}$ or a prefix of $ATC^*_{\lambda_i}$. This restriction ensures that there exists a $k_i \in \{1, \cdots, u_i\}$ such that $\lambda_{i,u_i} \neq \lambda^*_{i,u_i}$ (Otherwise, ATC_{λ_i} would be a prefix of $ATC^*_{\lambda_i}$). We set k_i it is the smallest index. To respond to the query, algorithm $\mathcal{B}$ first derives a private key for the attribute chain $(\lambda_i, \lambda_{i,1}, \cdots, \lambda_{i,u_i})$ from which it then constructs a private key for the requested attribute $(\lambda_i, \lambda_{i,1}, \cdots, \lambda_{i,k_i}, \cdots, \lambda_{i,u_i})$. In order to generates the private key, $\mathcal{B}$ first sets r'_i randomly from Z_p. We assume $r_i = r'_i + \frac{q_{\lambda_i}(0)\cdot\alpha^{k_i}}{(\lambda_{i,k_i} - \lambda^*_{i,k_i})}$. Next $\mathcal{B}$ generates the private key as $\left(g_2^{Q_{\lambda_i}(0)} \cdot (u'_i \prod_{j=1}^{k_i} u_j^{\lambda_{i,j}})^{r_i}, g^{r_i}, u_{k_i+1}^{r_i}, \cdots, u_{\omega_i}^{r_i}\right)$. We show that $\mathcal{B}$ can compute all the elements of the private key and use the fact that $y_i^{\alpha^j} = y_{i+j}$ for any i, j. We will generate the first component of the private key as follows:

$$
\begin{aligned}
&(u'_i \prod_{j=1}^{k_i} u_j^{\lambda_{i,j}})^{r_i} \\
&= \left(g^{\delta_i + \sum_{j=1}^{k} \lambda_{i,j}\gamma_j} \cdot \prod_{j=1}^{k_i-1} y_{\omega-j+1}^{(\lambda^*_{i,j} - \lambda_{i,j})} \cdot y_{\omega-k_i+1}^{\lambda^*_{i,k_i} - \lambda_{i,k_i}} \cdot \prod_{j=k_i+1}^{\omega} y_{\omega-j+1}^{\lambda^*_{i,j}} \right)^{r_i}
\end{aligned}
$$

We denote Z as the product of the first, second, and fourth terms:

$$
Z = \left(g^{\delta_i + \sum_{j=1}^{k} \lambda_{i,j}\gamma_j} \cdot \prod_{j=1}^{k_i-1} y_{\omega-j+1}^{(\lambda^*_{i,j} - \lambda_{i,j})} \cdot \prod_{j=k_i+1}^{\omega} y_{\omega-j+1}^{\lambda^*_{i,j}} \right)^{r_i}
$$

Next, we handle the term $y_{\omega-k_i+1}^{r_i(\lambda^*_{i,k_i} - \lambda_{i,k_i})}$, that is,

$$
\begin{aligned}
y_{\omega-k_i+1}^{r_i(\lambda^*_{i,k_i} - \lambda_{i,k_i})} &= y_{\omega-k_i+1}^{r'(\lambda^*_{i,k_i} - \lambda_{i,k_i})} \cdot y_{\omega-k_i+1}^{(\lambda^*_{i,k_i} - \lambda_{i,k_i})\frac{\alpha^{k_i}}{(\lambda_{i,k_i} - \lambda^*_{i,k_i})}} \\
&= y_{\omega-k_i+1}^{r'(\lambda^*_{i,k_i} - \lambda_{i,k_i})} / y_{\omega+1}.
\end{aligned}
$$

Hence, we denote the first component in the private key as:

$$
\begin{aligned}
&g_2^{Q_{\lambda_i}(0)} \cdot (u'_i \prod_{j=1}^{k_i} u_j^{\lambda_{i,j}} \cdot g_3)^{r_i} \\
=&g_1^{\gamma q_{\lambda_i}(0)} \cdot Z \cdot y_{\omega-k_i+1}^{r'(\lambda^*_{i,k_i} - \lambda_{i,k_i})}.
\end{aligned}
$$

$\mathcal{B}$ can compute the first private key component since the $y_{\omega+1}$ cancel out with all the terms in this expression are knows to $\mathcal{B}$. The second component g^{r_i} is $y_{k_j}^{1/(\lambda_{i,k_i} - \lambda^*_{i,k_i})} \cdot g^{r'}$ which $\mathcal{B}$ can compute. Similarly, the remaining elements $u_{k_j+1}^{r_i}, \cdots, u_{\omega}^{r_i}$ can be computed by $\mathcal{B}$ since they do not contain the element $y_{\omega+1}$.

Case2: We simulate the attribute λ_i where $\lambda_i \notin Y_0$. In order to generates the private key, $\mathcal{B}$ first picks r'_i randomly from Z_p. We assume $r_i = r'_i + \frac{q_{\lambda_i}(0)\cdot\alpha^{k_i}}{-\lambda^*_{i,k_i}} \in Z_p$ and $\mathcal{B}$ generates the private key as follows:

$$
\left(g_2^{Q_{\lambda_i}(0)} \cdot (u'_i)^{r_i}, g^{r_i}, u_1^{r_i}, \cdots, u_{k_i+1}^{r_i}, \cdots, u_{\omega_i}^{r_i}\right).
$$

It can be easily verified that $\mathcal{B}$ can compute all the elements of the private key since $y_i^{\alpha^j} = y_{i+j}$ for any i, j. We expand

first component of the private key as:

$$g_2^{Q_{\lambda_i}(0)} \cdot (u_i')^{r_i} = (y_{\omega+1}^{q_{\lambda_i}(0)} \cdot g_1^{\gamma q_{\lambda_i}(0)}) \cdot (g^{\delta_i} \cdot \prod_{j=1}^{\omega} y_{\omega-j+1}^{\lambda^*_{i,j}})^{r_i}$$

$$= (g_1^{\gamma q_{\lambda_i}(0)}) \cdot (g^{\delta_i} \cdot \prod_{j=1, i\neq k}^{\omega} y_{\omega-j+1}^{\lambda^*_{i,j}})^{r_i} y_{\omega-k_i+1}^{r_i' \lambda^*_{i,j}}.$$

We observe the second component g^{r_i} is $y_{\omega-k_i+1}^{r_i'} / y_{k_i}^{q_{\lambda_i}(0)}$ which $\mathcal{B}$ can compute. Similarly, the remaining element $u_1^{r_i}, \cdots, u_{k_i+1}^{r_i}, \cdots, u_{\omega_i}^{r_i}$ can also be compute by $\mathcal{B}$.

Challenge: In order to build the challenge ciphertext by adversary, $\mathcal{A}$ gives two message m_0 and m_1 to the simulator. $\mathcal{B}$ flips a coin β and creates the ciphertext as:

$$CT = \{T \cdot M_\beta \cdot e(y_1, h^\gamma), h, \{h^{\delta_i + \sum_{j=1}^{\omega} \lambda^*_{i,j} \gamma_j}\}\}$$

where h and T are form the input element given to $\mathcal{B}$. We note that if $h = g^c$ which c is unknown in Z_p.

$$h^{\delta_i + \sum_{j=1}^{\omega} \lambda^*_{i,j} \gamma_j} = \left(\prod_{j=1}^{\omega} (g^{\gamma_j} / y_{\omega-j+1})^{\lambda^*_{i,j}} \cdot (g^{\delta_i} \prod_{j=1}^{\omega} y_{\omega-j+1}{}^{\lambda^*_{i,j}}) \right)^c$$

$$= (u_1^{\lambda^*_{i,1}} \cdots u_{m_j}^{\lambda^*_{i,m_j}} \cdots u_\omega^{\lambda^*_{i,\omega}} \cdot u_i')^c$$

$$e(g,h)^{\alpha^{\omega+1}} e(y_1, h^\gamma) = (e(y_1, y_\omega) \cdot e(y_1, g^\gamma))^c = (g_1, g_2)^c$$

Finally, we observe that challenge ciphertext is a valid tuple of m_β if $T = e(g,h)^{\alpha^{\omega+1}}$. Otherwise, we say the ciphertext contain no information about m_β from adversary view because T is random element from $\mathbb{G}_T$.

Guess: The adversary will guess whether the β' and β equal or not. If $\beta = \beta'$, $\mathcal{B}$ outputs 0 to guesses that $T = (g,h)^{\alpha^{\omega+1}}$. Otherwise, it outputs 1 to indicate T is a random group element in $\mathbb{G}_T$.

If T is a valid tuple, we say that $\mathcal{B}$ gives a perfect simulation and has the following equation hold below

$$\Pr\left[\mathcal{B}(\vec{y}, T = (g,h)^{\alpha^{\omega+1}}) = 0\right] = \tfrac{1}{2} + \mathrm{Adv}_{\mathcal{A}}$$

Otherwise, T is a random element and we say the message m_b is hidden from $\mathcal{A}$ and we have the following equation:

$$\Pr\left[\mathcal{B}(\vec{y}, T = R) = 0\right] = \frac{1}{2}.$$

Therefore, the overall advantage of the simulator in the l-th bilinear Diffie-Hellman inversion game is non-negligible. ■

VI. Conclusion

In this paper, we propose a scheme called KP-WABE scheme due to attributes in the system is not always in the same position. Weighted access structure is used to construct the KP-WABE scheme in this paper. In order to decrypt the ciphertext successfully, a weighted attribute set contains in the ciphertext should be satisfied the weighted access tree which is contained in user's private key. Our KP-WABE can be transform into traditional KP-ABE easily when all the attributes in the system have equal weight. We also give a security model for KP-WABE and use l-th bilinear Diffie-Hellman inversion assumption to prove the security of our scheme. More importantly, our construction can exhibit significant improvement over the traditional ABE scheme accordant with the practical scenario.

Acknowledgment

This research is supported by Changjiang Scholars and Innovative Research Team in University (IRT1078); the Key Program of NSFC-Guangdong Union Foundation under grant No. U1135002; Major national S&T program(2011ZX03005002, 2012ZX03001009); the Fundamental Research Funds for the Central Universities (JY10000903001, K5051301017); the National Natural Science Foundation of China (61303218, 61370078).

References

[1] A. Sahai and B. Waters, "Fuzzy identity-based encryption," *Advances in Cryptology–EUROCRYPT 2005*, pp. 557–557, 2005.

[2] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Advances in cryptology*. Springer, 1985, pp. 47–53.

[3] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in *Advances in CryptologyłCRYPTO 2001*. Springer, 2001, pp. 213–229.

[4] 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.

[5] 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.

[6] L. Cheung and C. Newport, "Provably secure ciphertext policy abe," in *Proceedings of the 14th ACM conference on Computer and communications security*. ACM, 2007, pp. 456–465.

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

[8] J. Herranz, F. Laguillaumie, and C. Ràfols, "Constant size ciphertexts in threshold attribute-based encryption," in *Public Key Cryptography–PKC 2010*. Springer, 2010, pp. 19–34.

[9] K. Emura, A. Miyaji, A. Nomura, K. Omote, and M. Soshi, "A ciphertext-policy attribute-based encryption scheme with constant ciphertext length," in *Information Security Practice and Experience*. Springer, 2009, pp. 13–23.

[10] J. Li, Q. Wang, C. Wang, and K. Ren, "Enhancing attribute-based encryption with attribute hierarchy," *Mobile Networks and Applications*, vol. 16, no. 5, pp. 553–561, 2011.

[11] S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving secure, scalable, and fine-grained data access control in cloud computing," in *INFOCOM, 2010 Proceedings IEEE*. IEEE, 2010, pp. 1–9.

[12] M. Li, S. Yu, Y. Zheng, K. Ren, and W. Lou, "Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption," 2013.

[13] X. Liu, J. Ma, J. Xiong, Q. Li, and J. Ma, "Ciphertext-policy weighted attribute based encryption for fine-grained access control," in *Intelligent Networking and Collaborative Systems (INCoS), 2013 5th International Conference on*. IEEE, 2013, pp. 51–57.

[14] D. Boneh, X. Boyen, and E.-J. Goh, "Hierarchical identity based encryption with constant size ciphertext," in *Advances in Cryptology–EUROCRYPT 2005*. Springer, 2005, pp. 440–456.

[15] A. Beimel, "Secure schemes for secret sharing and key distribution," Ph.D. dissertation, PhD thesis, Israel Institute of Technology, Technion, Haifa, Israel, 1996.

[16] A. Sahai, "Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security," in *Foundations of Computer Science, 1999. 40th Annual Symposium on*. IEEE, 1999, pp. 543–553.

[17] R. Canetti, S. Halevi, and J. Katz, "Chosen-ciphertext security from identity-based encryption," in *Advances in Cryptology-Eurocrypt 2004*. Springer, 2004, pp. 207–222.