Attribute-Based Data Sharing Scheme Revisited in Cloud Computing

Shulan Wang, Kaitai Liang, Joseph K. Liu, Jianping Yu, Jianyong Chen, Weixin Xie

***Abstract*—Ciphertext-policy attribute-based encryption (CP-ABE) is a promising encryption technology for secure data sharing in cloud computing, in which data owner can fully control access structure associated with a ciphertext. However, it brings a major drawback which is known as key escrow problem, since the decryption users' secret keys are issued by an unconditionally trusted third party (i.e. key authority). In addition, there is another problem that most of CP-ABE schemes cannot express arbitrary-state attributes. In this paper, we revisited attribute-based data sharing scheme in order to solve the key escrow issue and improve the ability of attribute expression in cloud computing. An improved two-party key issuing protocol ensures that neither key authority nor cloud service provider can generate the whole secret keys of users individually. Moreover, the function of weighted attribute is provided to enhance the attribute expression, which can not only extend attributes from binary state to arbitrary states, but also reduce the complexity of access policy associated with a ciphertext. Therefore, both ciphertext storage and time cost in encryption are saved. The performance analysis and security proof show that the proposed scheme is efficient to securely achieve data sharing in cloud computing.**

Keywords*—*Cloud computing, Data sharing, Attribute-based encryption, Removing escrow, Weighted attribute.

I. Introduction

CLOUD computing has become a research hot-spot due to its renowned advantages (e.g. convenience, high scalability). One of the most promising cloud applications is online data sharing, such as Online Social Networks with more than one billion users [1]. Data owner (DO) stores large amounts of data in cloud for ease of data sharing and cost saving on local management. Meanwhile, cloud service provider (CSP) becomes the manager of user's data, and key authority (KA) gets more power than ever. However, they cannot be fully trusted since they may reveal DO's data for benefits. Therefore, how to securely and efficiently share user's data becomes one of the most challenges in cloud computing [2], [3].

Ciphertext-policy attribute-based encryption (CP-ABE) [4]–[9] has been a useful encryption technology to solve the challenging problem of secure data sharing. In CP-ABE, user's secret key is described by attributes, and ciphertext is associated with access structure. It enables DO to define access structure over the universe of attribute. User can decrypt ciphertext only if his attributes match the restrictions on predefined access structure.

There will be several challenges if CP-ABE schemes are directly applied in cloud system. Firstly, all users' secret keys are issued by an unconditionally trusted KA. It brings a major drawback which is known as key escrow problem. By knowing users' secret keys, the KA can decrypt all ciphertext addressed to specific users. However, the users may not allow the KA with such a power.

Secondly, attribute expression is another challenge. At present, most CP-ABE schemes [4]–[8], [10]–[12] cannot deal with arbitrary-state attributes, but can only express binary state about attributes: "satisfying" and "not-satisfying". In this paper, the weighted attribute is proposed which can not only extend attributes from binary state to arbitrary states, but also simplify access policy associated with a ciphertext. Thus the storage cost of ciphertext and time cost in encryption can be saved. We use an example to further illustrate our idea.

Suppose in a university, teachers are classified into teaching assistant, lecturer, associated professor and full professor. We distribute the weight of the attributes as 1, 2, 3, and 4, and are denoted as "teacher: 1", "teacher: 2", "teacher: 3" and "teacher: 4", respectively. The four attributes can be denoted by one attribute which has just different weights. Especially, it can be arbitrary-state attributes, such as "Teacher: teaching assistant, lecturer, associate professor, full professor". Furthermore, we assume that an access policy is represented as: $\mathcal{T}$ {("Lecturer" OR "Associate Professor" OR "Full Professor") AND "Male"} and the existing CP-ABE schemes are executed on the form of access policy $\mathcal{T}$. If our proposed scheme is deployed, the $\mathcal{T}$ can be improved as $\mathcal{T}^{'}$ {"teacher: 2" AND "male"}, since the attribute "teacher: 2" denotes the minimum level in the access policy and includes {"teacher: 2", "teacher: 3" "teacher: 4"} by default. Therefore, the storage overhead of ciphertext and computation complexity in encryption can be reduced. These two structures are shown in Fig. 1. In addition, it can express larger attribute space than ever under the same condition. For example, if the attribute space and weighted set include n elements, respectively, the proposed scheme can describe n^2 different possibilities. In contrast, the previous schemes only show $2n$ different possibilities.

A. Related Work

In 2005, Sahai and Waters [13] proposed fuzzy identity-based encryption (IBE), which was the prototype of attribute-based encryption (ABE). Then, two variants of ABE were

Shulan Wang, Jianping Yu, and Weixin Xie are with ATR Key Laboratory of National Defense Technology and College of Information Engineering, Shenzhen University, Shenzhen, P.R. China (e-mail: wangshulan@email.szu.edu.cn, yujp@szu.edu.cn, wxxie@szu.edu.cn).

Kaitai Liang is with Department of Information and Computer Science, Aalto University, Finland (e-mail: kaitai.liang@aalto.fi).

Joseph K. Liu is with Faculty of Information Technology, Monash University, Australia (e-mail: joseph.liu@monash.edu).

Jianyong Chen is with College of Computer and Software Engineering, Shenzhen University, Shenzhen, P.R. China (e-mail: jychen@szu.edu.cn).

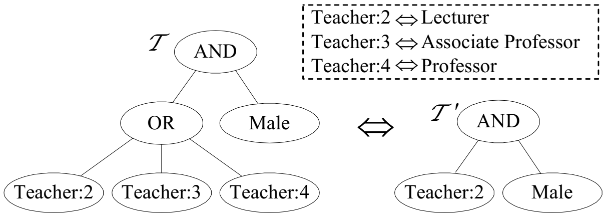


Fig. 1. Two equivalent access structures of a ciphertext. $\mathcal{T}$ represents a general access policy in the existing CP-ABE schemes. $\mathcal{T}'$ denotes an improved access policy in the proposed scheme.

proposed: key-policy ABE (KP-ABE) [14] and CP-ABE [4], [5], depending on how attributes and policy are associated with a ciphertext and secret keys of users. Later, many CP-ABE schemes with specific features have been presented. For example, [12] presented a novel access control scheme in cloud computing with efficient attribute and user revocation. The computational overhead is significantly eliminated from $O(2N)$ to $O(N)$ in user key generation by improving CP-ABE construction, where N is the number of attributes. The size of ciphertext is approximately reduced to 1/2. However, the security of the scheme was given only with discussion.

At present, most CP-ABE schemes are constructed on the architecture where a full trusted authority is required to generate the whole secret keys of users with its master secret key as input [4]–[6], [12]–[17]. Thus, the key escrow issue is inherent such that the authority can decrypt all ciphertext encrypted for users. Chase *et al.* [10] presented a distributed KP-ABE scheme that solved the key escrow problem in a multi-authority system. In this approach, all authorities, which are not joined, are participating in the key generation protocol in a distributed way, such that they cannot pool their data and link multiple attribute sets belonging to the same user. Because there is no centralized authority with master secret information, all attribute authorities should communicate with the other authorities in the system to create a user's secret key. But, a major disadvantage of this approach is the performance degradation [18], [19]. It results in $O(N^2)$ communication overhead on the system setup and on any rekeying phase, and requires each user to store $O(N^2)$ additional auxiliary key components in addition to the attribute keys, where N is the number of authorities in the system. Then, Chow [20] proposed an anonymous private key generation protocol in IBE scheme where KA can issue private key to an authenticated user without knowing the list of the user's identities. It seems that this approach can properly be used in ABE schemes if attributes are treated as identities. However, this scheme cannot be adopted for CP-ABE since the identity of user is a set of attributes which is not public.

Until 2013, [11] provided an improving security data sharing scheme based on the typical CP-ABE [4]. The key escrow issue is resolved by using an escrow-free key issuing protocol where the key generation center and the data storing center work together to generate secret key for user. Therefore, the computational cost in generating user's secret key increases since the protocol requires interactive computation between the both parties.

Besides, Liu *et al.* [15], [16] presented fine-grained access control scheme with attribute hierarchy, where [15] and [16] are constructed based on [5] and [6], respectively. In these two schemes, the attributes are divided into multiple levels to achieve fine-grained access control for hierarchical attributes, but the attributes can only express binary state as before. Later, Fan *et al.* [17] proposed an arbitrary-state ABE to solve the issue of the dynamic membership management in an ABE scheme. In this paper, to express arbitrary-state attributes, the traditional attributes are expanded to two parts: attribute and value. For example, the traditional attributes can be denoted as {“Doctor”,“Professor”,“Engineer”}. In this scheme, the improved attributes are denoted as: {Career: “Doctor”,“Professor”,“Engineer”}, where “Career” represents an attribute and “Doctor”,“Professor” and “Engineer” denote values of the attribute “Career”. Accordingly, the computation cost for attributes is more expensive than the traditional schemes under the same number of attributes.

B. *Our Contributions*

In this study, based on [12], an attribute-based data sharing scheme is proposed in cloud computing, which is denoted as ciphertext-policy weighted ABE scheme with removing escrow (or CP-WABE-RE, for short). It successfully resolves two types of problems: key escrow and arbitrary-sate attribute expression. The contributions of our scheme are as follows:

- we propose an improved key issuing protocol to resolve the key escrow problem of CP-ABE in cloud computing. The protocol can prevent KA and CSP from knowing each other's master secret key so that none of them can create the whole secret keys of users individually. Thus, the fully trusted KA can be semi-trusted in the proposed scheme. In this case, data confidentiality and privacy can be ensured.
- we present weighted attribute to enhance the expression of attribute. The weighted attribute can not only express arbitrary-state attributes instead of the traditional binary state, but also reduce the complexity of access policy. Thus the storage cost of ciphertext and computation complexity in encryption can be reduced. Besides, it can express larger attribute space than ever under the same condition.

  Note that there are significant improvements of our scheme over [17] that the weighted attribute can simplify access policy associated with a ciphertext (as the Fig. 1 shows) under the same ability of expression, which saves the storage cost of ciphertext and time cost in encryption.
- we conduct and implement comprehensive experiment for the proposed scheme. The simulation shows that CP-WABE-RE scheme is efficient both in terms of computation complexity and storage cost. In addition, the security of CP-WABE-RE scheme is also proved under the generic group model.

C. *Organization*

The remaining parts of this paper are organized as follows. Section II discusses preliminaries which contain the related background. Section III proposes system model of CP-WABE-RE scheme. In section IV, the concrete construction of CP-WABE-RE scheme is presented. The theoretical analysis and experimental simulation are given in section V. In section VI, we provide security proof for the proposed scheme. Finally, conclusions are presented in section VII.

II. Preliminaries

A. Access Structure

Let $\{P_1, \ldots, P_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^{\{P_1,\ldots,P_n\}}$ is monotone if $\forall B, C$: if $B \in \mathbb{A}$ and $B \subseteq C$ then $C \in \mathbb{A}$. An access structure (respectively, monotone access structure) is a collection (respectively, monotone collection) $\mathbb{A}$ of non-empty subsets of $\{P_1, \ldots, P_n\}$, i.e., $\mathbb{A} \subseteq 2^{\{P_1,\ldots,P_n\}} \backslash \{\emptyset\}$. The sets in $\mathbb{A}$ are called authorization sets. Otherwise, the sets are called unauthorization sets.

In the proposed scheme, the role of the parties is taken by the attributes. Thus, $\mathbb{A}$ is going to include the authorized sets of attributes. Generally, unless stated in another way, the scheme uses an access structure which is a monotone access structure.

B. Bilinear Maps

Let $\mathbb{G}_0$ and $\mathbb{G}_T$ be two groups of prime order p. The generator of $\mathbb{G}_0$ is g. A bilinear mapping $\hat{e} : \mathbb{G}_0 \times \mathbb{G}_0 \to \mathbb{G}_T$ satisfies the following properties:

- Bilinearity: For any $u, v \in \mathbb{G}_0$ and $a, b \in \mathbb{Z}_p$, it has $\hat{e}(u^a, v^b) = \hat{e}(u, v)^{ab}$.
- Non-degeneracy: There exists $u, v \in \mathbb{G}_0$ such that $\hat{e}(u, v) \neq 1$.
- Computability: For all $u, v \in \mathbb{G}_0$, there is an efficient computation $\hat{e}(u, v)$.

C. Weighted Access Tree

Let $\mathcal{T}$ be a weighted access tree, where root node of the tree is R. To facilitate description of the access tree, several functions and terms are defined as follows.

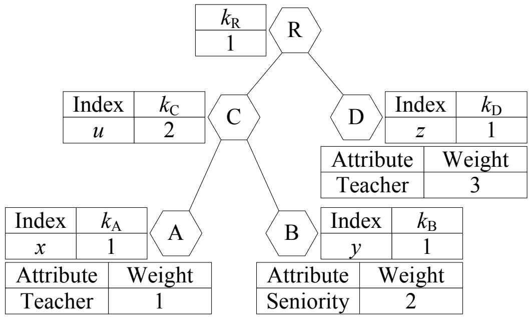


Fig. 2. An example of weighted access tree structure.

- x denotes a node of tree $\mathcal{T}$. If x is a leaf node, it denotes an attribute with weight. If x is a non-leaf node, it denotes a threshold gate, such as "AND", "OR","n-of-m (n<m)". For example, the nodes C and A denote a threshold gate and an attribute respectively in Fig. 2.
- num_x denotes the number of x's children in $\mathcal{T}$. For example, $num_{\text{R}} = 2$ in Fig. 2.
- k_x denotes threshold value of node x, where $0 < k_x \leq num_x$. When $k_x = 1$ and x is a non-leaf node, it is an OR gate. When $k_x = num_x$ and x is a non-leaf node, it is an AND gate. In particular, if x is a leaf node, $k_x = 1$. For example, $k_{\text{R}} = 1$ and $k_{\text{C}} = 2$ denote an OR gate and an AND gate respectively in Fig. 2.
- $parent(x)$ represents the parent of the node x in $\mathcal{T}$. For example, $parent(\text{A}) = \text{C}$ in Fig. 2.
- $att(x)$ denotes an attribute associated with the leaf node x in $\mathcal{T}$.
- $index(x)$ returns an unique value associated with the node x, where the value is assigned to x for a given key in an arbitrary manner.
- $\mathcal{T}_x$ denotes the sub-tree of $\mathcal{T}$ rooted at the node x. If a set of weighted attribute S satisfies the access tree $\mathcal{T}_x$, we denote it as $\mathcal{T}_x(S) = 1$. $\mathcal{T}_x(S)$ is recursively computed as follows. If x is a non-leaf node, $\mathcal{T}_x(S)$ returns 1 if and only if at least k_x children return 1. If x is a leaf node, then $\mathcal{T}_x(S)$ returns 1 if and only if the weight of attribute ω_x from S must be greater than or equal to the weight of the leaf node. That is $weight(\omega_x) \geq weight(att(x))$.

In addition, Morillo *et al.* [21] proved that every weighted value of the threshold access structure can be defined as a natural number. Unless stated otherwise, the value of weight is a natural number in this paper. In Fig. 2, the access policy is denoted as: {("Teacher:1" And "Seniority:2") OR "Teacher:3"}. If one possesses attributes ("Teacher", "Seniority") with weight ("1", "2"), he can satisfy the tree in Fig. 2; If the other one who possesses attribute ("Teacher") with weight ("4"), he can also satisfy the access tree.

III. System Model

As illustrated in Fig. 3 and Fig. 4, the system model and framework of CP-WABE-RE scheme in cloud computing are given, where the system consists of four types of entities: KA, CSP, DO and Users. In addition, we provide the detailed definition of CP-WABE-RE scheme.

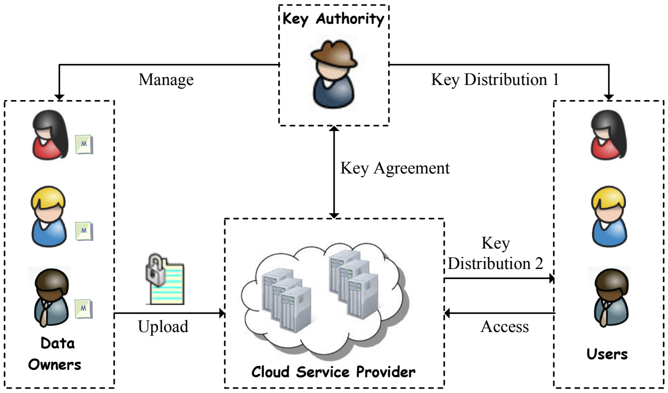


Fig. 3. System model of CP-WABE-RE scheme in cloud computing.

Key Authority (KA). It is a semi-trusted entity in cloud system. Namely, KA is honest-but-curious, which can honestly perform the assigned tasks and return correct results. However, it will collect as many sensitive contents as possible. In cloud system, the entity accepts the users' enrollment. Meanwhile, it not only generates most part of system parameter, but is also in charge of creating most part of secret key for each user.

Cloud Service Provider (CSP). It is the manager of cloud servers. It is also a semi-trusted entity which provides many services such as data storage, computation, and transmission. To solve the key escrow problem, it generates parts of system parameter and secret key for each user.

Data Owners (DO). They are owners of files to be stored in cloud system. They are in charge of defining access structure and executing data encryption operation. They also upload the generated ciphertext to CSP.

Users. They want to access ciphertext stored in cloud system. They download the ciphertext and execute the corresponding decryption operation.

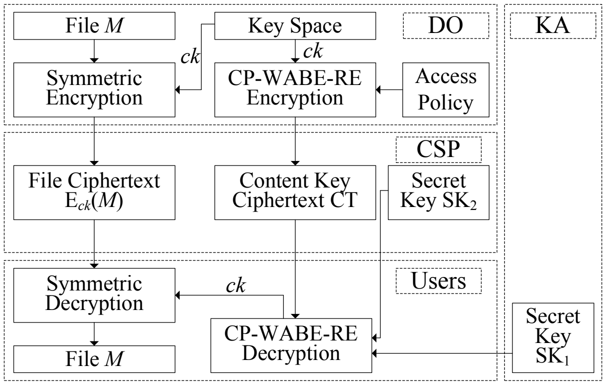


Fig. 4. System framework of CP-WABE-RE scheme.

Definition 1. *(CP-WABE-RE): The proposed scheme contains the following four phases:*

$\mathbf{Phase\ 1}$: $\mathbf{System\ Initialization}$. This phase includes both algorithms: $\mathbf{KA.Setup}$ and $\mathbf{CSP.Setup}$.

(1) $\mathbf{KA.Setup}(1^\kappa) \rightarrow (\mathrm{PP}_1, \mathrm{MSK}_1)$. It is executed by KA. The probabilistic operation inputs security parameter κ. It returns public parameter PP_1 and master secret key MSK_1.

(2) $\mathbf{CSP.Setup(1^\kappa)} \rightarrow (\mathrm{PP}_2, \mathrm{MSK}_2)$. This algorithm is run by CSP. It inputs a security parameter κ and generates PP_2 and MSK_2.

The system public parameter is $\mathrm{PP} = \{\mathrm{PP}_1, \mathrm{PP}_2\}$, while the master secret key of system is $\mathrm{MSK} = \{\mathrm{MSK}_1, \mathrm{MSK}_2\}$ stored by KA and CSP, respectively.

$\mathbf{Phase\ 2}$: $\mathbf{Data\ Encryption}$. To improve efficiency of encryption, DO first encrypts file M with content key ck by using simple symmetric encryption algorithm, where file ciphertext is denoted as $E_{ck}(M)$. Then, the content key ck is encrypted by the following operation.

$\mathbf{DO.Encrypt}(\mathrm{PP}, ck, \mathbb{A}) \rightarrow (\mathrm{CT})$. DO inputs PP, ck, and an access policy $\mathbb{A}$. It encrypts ck and outputs content key ciphertext CT which implicitly contains $\mathbb{A}$. Then, DO delivers $E_{ck}(M)$ and CT to CSP.

$\mathbf{Phase\ 3}$: $\mathbf{User\ Key\ Generation}$. This phase consists of $\mathbf{KA.KeyGen}$ and $\mathbf{CSP.KeyGen}$.

(1) $\mathbf{KA.KeyGen}(\mathrm{MSK}_1, S) \rightarrow (\mathrm{SK}_1)$. KA inputs MSK_1 and a set of weighted attributes S. It creates secret key SK_1 described by S.

(2) In $\mathbf{CSP.KeyGen}$, we propose an improved two-party key issuing protocol to remove escrow. KA and CSP perform the improved protocol with master secret keys of their own. Thus, none of them can create the whole set of secret keys of users individually. Meanwhile, we assume that KA does not collude with CSP since they are honest as in [22] (otherwise, they can obtain the secret keys of each user by sharing their master secret keys).

$\mathbf{CSP.KeyGen}(\mathrm{MSK}_2) \rightarrow (\mathrm{SK}_2)$. CSP inputs MSK_2 and the required information. It produces secret key SK_2 by executing the following key issuing protocol.

- $\mathbf{KeyCom_{KA \leftrightarrow CSP}}(\mathrm{MSK}_1, ID_t, r, \mathrm{MSK}_2) \rightarrow (\mathrm{SK}_2)$. It is an interactive algorithm between KA and CSP. KA inputs MSK_1, a user identity ID_t, and a personalized secret r. CSP inputs MSK_2 and ID_t. At last, only CSP generates a personalized key component SK_2 for the corresponding user.

Then, the user constructs the whole secret key SK with the key components separately receiving from KA and CSP, i.e. $\mathrm{SK} = \{\mathrm{SK}_1, \mathrm{SK}_2\}$.

$\mathbf{Phase\ 4}$: $\mathbf{Data\ Decryption}$. This phase contains both algorithms: $\mathbf{Users.Decrypt}$ and $\mathbf{Data.Decrypt}$. User first downloads file ciphertext $E_{ck}(M)$ and content key ciphertext CT from CSP. If he satisfies conditions, he can get content key ck by calling $\mathbf{Users.Decrypt}$ algorithm. Then, he uses ck to further decrypt file M by using $\mathbf{Data.Decrypt}$ operation.

(1) $\mathbf{Users.Decrypt}(\mathrm{PP}, \mathrm{SK}, \mathrm{CT}) \rightarrow (ck)$. User inputs PP, SK described by S, and CT which includes access policy $\mathbb{A}$. Only when the weighted attribute set S matches the access policy $\mathbb{A}$, the content key ck is obtained.

(2) $\mathbf{Data.Decrypt}(E_{ck}(M), ck) \rightarrow (M)$. User inputs $E_{ck}(M)$ and ck. Based on symmetric decryption algorithm, it outputs file M.

IV. The proposed CP-WABE-RE Scheme

In this section, we present the construction of CP-WABE-RE system, including five procedures: system initialization, new file creation (data encryption), new user authorization (user key generation), data file access (data decryption), and data file deletion. In addition, the revocation scheme of [11] can be directly used in our proposed scheme. The reason is described as below. The revocation scheme is performed in the phase of data encryption. And the removing escrow is operated in the phase of user key generation. Therefore, in [11], the modification of removing escrow does not affect the use of revocation scheme since they are run in different phases.

A. System Initialization

Let $\hat{e} : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_T$ be a bilinear map, and $\mathbb{G}_0$ be bilinear group of prime order p with generator g. Let

$H : \{0,1\}^* \rightarrow \mathbb{G}_0$ be a hash function. For any $i \in \mathbb{Z}_p$ and a set $S = \{s_1, s_2, \ldots, s_m \in \mathbb{Z}_p\}$, the Lagrange coefficient $\Delta_{i,S} = \prod_{l \in S, l \neq i}(x - l)/(i - l)$. In addition, an universe of attribute set $A = \{a_1, \ldots, a_n\}$ and a set of weight $W = \{\omega_1, \ldots, \omega_n\}(\omega_1 \leq \ldots \leq \omega_n)$ are defined. Thus the system contains n^2 weighted attributes which are $\tilde{A} = \{a_1 : \omega_1, \ldots, a_1 : \omega_n, \ldots, a_n : \omega_1, \ldots, a_n : \omega_n\}$, where the higher hierarchy of attributes is used, the bigger weighted value is distributed.

(1) $\mathbf{KA}.\mathbf{Setup}(1^\kappa)$. KA runs the algorithm which inputs security parameter κ. Then, KA chooses random $\alpha_1, \beta \in \mathbb{Z}_p$ and computes $h = g^\beta$ and $u_1 = \hat{e}(g,g)^{\alpha_1}$. Lastly, it obtains PP_1 and MSK_1 as the formula (1):

$$\mathrm{PP}_1 = \{\mathbb{G}_0, g, h, u_1\},\ \mathrm{MSK}_1 = \{\alpha_1, \beta\} \tag{1}$$

(2) $\mathbf{CSP}.\mathbf{Setup}(\mathbf{1}^\kappa)$. CSP executes the operation which inputs security parameter κ. Based on the κ, CSP chooses a random number $\alpha_2 \in \mathbb{Z}_p$ and calculates $u_2 = \hat{e}(g,g)^{\alpha_2}$. Then, it sets PP_2 and MSK_2 as the formula (2):

$$\mathrm{PP}_2 = \{u_2\},\ \mathrm{MSK}_2 = \{\alpha_2\} \tag{2}$$

Finally, the public parameter and master secret key of system are denoted as $\mathrm{PP} = \{\mathbb{G}_0, g, h, u = u_1 \cdot u_2 = \hat{e}(g,g)^\alpha\}$, where $\alpha = \alpha_1 + \alpha_2$, and $\mathrm{MSK} = \{\{\alpha_1, \beta\}, \{\alpha_2\}\}$.

B. New File Creation (Data Encryption)

Before file M is uploaded to CSP, DO processes the file with the following steps: (1) DO picks a unique ID for file M. (2) It encrypts M with content key ck by using symmetric encryption method, where ck is chosen in a key space. The file ciphertext is denoted as $E_{ck}(M)$, where E_{ck} denotes a symmetric encryption operation with the key ck. (3) It defines an access structure $\mathcal{T}$ and encrypts the ck by running the improved encryption operation. Then, content key ciphertext CT is returned.

$\mathbf{DO}.\mathbf{Encrypt}(\mathrm{PP}, ck, \mathcal{T})$. The improved algorithm is executed by DO which inputs PP, ck, and $\mathcal{T}$. It outputs content key ciphertext CT.

Firstly, a polynomial q_x is selected for each node x (including the leaf nodes) in $\mathcal{T}$. From the root node R, the node's information of q_x is randomly selected from top to bottom manner. For each node x in $\mathcal{T}$, degree of the polynomial d_x is set to $k_x - 1$, where k_x is the threshold value.

Then, beginning from the root node R, DO sets $q_R(0) = s(s \in \mathbb{Z}_p)$, where s is randomly selected. And DO randomly selects d_R other points of the polynomial q_R to define it completely. For each non-root node x, it sets $q_x(0) = q_{parent(x)}(index(x))$ and randomly chooses d_x other points to completely define q_x. Meanwhile, each leaf node denotes a weighted attribute.

In the access tree $\mathcal{T}$, let Y be the set of leaf nodes, and ω_i be the minimum weight of each leaf node, which is set by DO. Then DO computes CT as the formula (3). Finally, DO sends the integrated ciphertext $\{ID, \mathrm{CT}, E_{ck}(M)\}$ to CSP.

$$\mathrm{CT} = \left\{ \begin{array}{l} \mathcal{T}, \tilde{C} = ck \cdot \hat{e}(g,g)^{\alpha \cdot s}, C = g^s \\ \left\{ \begin{array}{l} \forall y \in Y, i \in [1,n] : \\ C_y = h^{q_y(0)} \cdot H(att(y))^{-\omega_i s}, \\ \forall j \in (i,n], C_{y,j} = H(att(y))^{-(\omega_j - \omega_i)s} \end{array} \right\} \end{array} \right\} \tag{3}$$

To better understand, let us take the Fig. 2 for example here. DO sets the polynomial $q_{\mathrm{R}}(x)$ with degree 0 as $q_{\mathrm{R}}(x) = s$ for node R. At the same time, the polynomial $q_{\mathrm{C}}(x)$ is set as $q_{\mathrm{C}}(x) = s + c_r x$ with degree 1 where $q_{\mathrm{R}}(u) = q_{\mathrm{C}}(0) = s$ and c_r is randomly chosen in $\mathbb{Z}_p^*$ from node C. In addition, for leaf nodes A, B, and D, they are respectively set as: $q_{\mathrm{A}}(0) = q_{\mathrm{C}}(x) = s + c_r x$, $q_{\mathrm{B}}(0) = q_{\mathrm{C}}(y) = s + c_r y$, and $q_{\mathrm{D}}(0) = q_{\mathrm{R}}(z) = s$.

C. New User Authorization (User Key Generation)

When a user wants to join the cloud system, KA first accepts the user's enrollment. If he is legal, KA authenticates and assigns a set of weighted attributes S to the user in accordance with his identity or role. Then, KA and CSP will generate secret key SK for the user. The phase consists of $\mathbf{CSP}.\mathbf{KeyGen}$ and $\mathbf{KA}.\mathbf{KeyGen}$.

(1) $\mathbf{CSP}.\mathbf{KeyGen}$. We provide an improved key issuing protocol between KA and CSP to execute the work of CSP.

$\mathbf{KeyCom}_{\mathbf{KA} \leftrightarrow \mathbf{CSP}}(\mathrm{MSK}_1, ID_t, r, \mathrm{MSK}_2)$. Assume that user t needs a secret key. Firstly, KA randomly chooses a number $r \in \mathbb{Z}_p$ for the user. Then, KA and CSP perform a secure two-party computation (2PC) protocol, where KA inputs $\mathrm{MSK}_1 = \{\alpha_1, \beta\}$ and CSP inputs $\mathrm{MSK}_2 = \{\alpha_2\}$. After the execution of the 2PC protocol, CSP gets $x = (\alpha_1 + \alpha_2)\beta \bmod p$. This can be done via a general 2PC protocol for a simple arithmetic computation [10], [11], [20]. Alternatively, we can do this more efficiently by using the construction in [23]. Note that during the 2PC protocol, KA knows nothing about α_2 while CSP knows nothing about $\{\alpha_1, \beta\}$. After the 2PC protocol, KA and CSP engage the following interactive protocol in order to generate the personalized key component for the user:

1) CSP selects a random number $\rho_1 \in \mathbb{Z}_p$. It calculates $X_1 = g^{x/\rho_1} = g^{(\alpha_1+\alpha_2)\beta/\rho_1}$ and transmits $\{X_1, \mathrm{PoK}(\rho_1, x)\}$ to KA, where PoK represents a proof of knowledge of the secret values used in the computation. It can be efficiently realized, e.g. via Schnorr protocol.

2) KA chooses $\theta \in \mathbb{Z}_p$ and computes $Y_1 = X_1^{\theta/\beta} = g^{(\alpha_1+\alpha_2)\theta/\rho_1}$ and $Y_2 = h^{r\theta}$. It sends $\{Y_1, Y_2, \mathrm{PoK}(\theta, \beta, r)\}$ to CSP.

3) CSP randomly selects $\rho_2 \in \mathbb{Z}_p$ and computes $X_2 = (Y_1^{\rho_1} Y_2)^{\rho_2} = (g^{(\alpha_1+\alpha_2)\theta} h^{r\theta})^{\rho_2}$. It transfers $\{X_2, \mathrm{PoK}(\rho_2)\}$ to KA.

4) KA calculates $Y_3 = X_2^{1/\theta} = (g^{(\alpha_1+\alpha_2)} h^r)^{\rho_2}$ and sends $\{Y_3, \mathrm{PoK}(\theta)\}$ to CSP.

5) CSP calculates $D = Y_3^{1/\rho_2} = g^{(\alpha_1+\alpha_2)} h^r = g^\alpha h^r$ and sends a personalized key component $\mathrm{SK}_2 = \{D = g^\alpha h^r\}$ to the corresponding user t.

Fig. 5 provides a direct description for the above protocol. Here the first step denotes a 2PC protocol which inputs $\{\alpha_1, \beta\}$

from KA and $\{\alpha_2\}$ from CSP, and returns $x = (\alpha_1 + \alpha_2)\beta \bmod p$ to CSP.

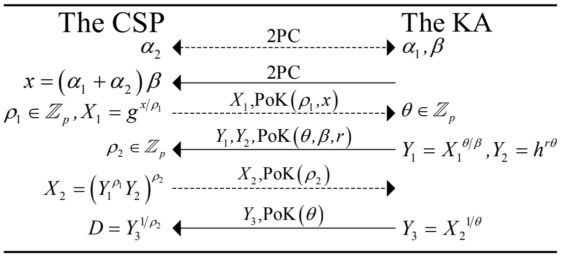

The CSP
The KA
2PC
α_2
α_1, β
2PC
$x = (\alpha_1 + \alpha_2)\beta$
$X_1, \mathrm{PoK}(\rho_1, x)$
$\rho_1 \in \mathbb{Z}_p, X_1 = g^{x/\rho_1}$
$\theta \in \mathbb{Z}_p$
$Y_1, Y_2, \mathrm{PoK}(\theta, \beta, r)$
$\rho_2 \in \mathbb{Z}_p$
$Y_1 = X_1^{\theta/\beta}, Y_2 = h^{r\theta}$
$X_2, \mathrm{PoK}(\rho_2)$
$X_2 = (Y_1^{\rho_1} Y_2)^{\rho_2}$
$Y_3, \mathrm{PoK}(\theta)$
$D = Y_3^{1/\rho_2}$
$Y_3 = X_2^{1/\theta}$

Fig. 5. The improved two-party key issuing protocol. “2PC” denotes a general two-party computation protocol. “Pok” denotes a proof of knowledge of the secret values used in the computation.

(2) $\mathbf{KA}.\mathbf{KeyGen}(\mathrm{MSK}_1, r, S)$. KA executes the algorithm which inputs MSK_1, a number $r \in \mathbb{Z}_p$ which has randomly chosen in $\mathbf{CSP}.\mathbf{KeyGen}$, and a set of weighted attributes S. Then, for each weighted attribute $j \in S$, it possesses weighted value $\omega_j (\omega_j \in W)$. Finally, it computes SK_1 described by S as the formula (4).

$$\mathrm{SK}_1 = \{L = g^r, \forall j \in S : D_j = H(j)^{r\omega_j}\} \tag{4}$$

So, user t can construct the whole secret key SK by using the key components separately receiving from the two entities. It is described as the formula (5).

$$\mathrm{SK} = \{D = g^{\alpha} h^r, L = g^r, \forall j \in S : D_j = H(j)^{r\omega_j}\} \tag{5}$$

D. Data File Access (Data Decryption)

In cloud system, legal users can freely query the ciphertext. When a user requests accessing a ciphertext to CSP, it transmits the corresponding ciphertext $\{ID, \mathrm{CT}, E_{ck}(M)\}$ to the user. The user can obtain content key ck by calling the improved $\mathbf{Users}.\mathbf{Decrypt}$ algorithm. Then, he uses ck to further decrypt the file M using $\mathbf{Data}.\mathbf{Decrypt}$ operation. If not, he cannot decrypt it.

(1) $\mathbf{Users}.\mathbf{Decrypt}(\mathrm{PP}, \mathrm{CT}, \mathrm{SK})$. User inputs PP, CT, and SK described by S. If the weighted attributes S that the user possesses satisfy access policy $\mathcal{T}$, the user can obtain content key ck. The operation is a recursive algorithm which is defined as below.

1) If x is a leaf node. Let $k = att(x)$, ω_k be the weighted value of the user's node x, and ω_i be the weighted value of the access policy $\mathcal{T}$'s node x. If $k \notin S$ or $k \in S, \omega_i > \omega_k$, we note $\mathrm{DecryptNode}(\mathrm{CT}, \mathrm{SK}, x) = \perp$. If $k \in S$ and $\omega_i = \omega_k$, we compute $\mathrm{DecryptNode}(\mathrm{CT}, \mathrm{SK}, x)_1$ as the formula (6). If $k \in S$, $\omega_i < \omega_k$ and $\omega_k = \omega_j$, we compute $\mathrm{DecryptNode}(\mathrm{CT}, \mathrm{SK}, x)_2$ as the formula (7).

$$\begin{aligned} &\mathrm{DecryptNode}(\mathrm{CT}, \mathrm{SK}, x)_1 \\ &= \hat{e}(C_x, L) \cdot \hat{e}(C, D_k) \\ &= \hat{e}(h^{q_x(0)} \cdot H(att(x))^{-\omega_i s}, g^r) \cdot \hat{e}(g^s, H(k)^{r\omega_k}) \\ &= \hat{e}(g^{\beta q_x(0)}, g^r) \cdot \hat{e}(H(k)^{-\omega_i s}, g^r) \cdot \hat{e}(g^s, H(k)^{r\omega_k}) \\ &= \hat{e}(g, g)^{r\beta q_x(0)} \qquad (if\ \omega_k = \omega_i) \end{aligned} \tag{6}$$

$$\begin{aligned} &\mathrm{DecryptNode}(\mathrm{CT}, \mathrm{SK}, x)_2 \\ &= \hat{e}(C_x \cdot C_{x,j}, L) \cdot \hat{e}(C, D_k) \\ &= \hat{e}(h^{q_x(0)} H(att(x))^{-\omega_i s} H(att(x))^{-(\omega_j - \omega_i)s}, g^r) \\ &\quad \cdot \hat{e}(g^s, H(k)^{r\omega_k}) \\ &= \hat{e}(g^{\beta q_x(0)} \cdot H(k)^{-\omega_j s}, g^r) \cdot \hat{e}(g^s, H(k)^{r\omega_k}) \\ &= \hat{e}(g^{\beta q_x(0)}, g^r) \cdot \hat{e}(H(k)^{-\omega_j s}, g^r) \cdot \hat{e}(g^s, H(k)^{r\omega_k}) \\ &= \hat{e}(g, g)^{r\beta q_x(0)} \qquad (if\ \omega_k = \omega_j > \omega_i) \end{aligned} \tag{7}$$

2) If x is a non-leaf node, $\mathrm{DecryptNode}(\mathrm{CT}, \mathrm{SK}, x)$ is defined: for all nodes z that are children of x, it runs $\mathrm{DecryptNode}(\mathrm{CT}, \mathrm{SK}, z)$ and stores the output as F_z. Let S_x be an arbitrary k_x-sized set of child nodes z such that $F_z \neq \perp$. If the nodes don't exist, $F_z = \perp$. If not, F_x is computed as the formula (8), where $k = index(z)$, and $S_x^{'} = \{index(z) : z \in S_x\}$.

$$\begin{aligned} F_x &= \prod_{z \in S_x} F_z^{\Delta_{k, S'_x}(0)} \\ &= \prod_{z \in S_x} (\hat{e}(g, g)^{r \cdot \beta q_z(0)})^{\Delta_{k, S'_x}(0)} \\ &= \prod_{z \in S_x} (\hat{e}(g, g)^{r \cdot \beta q_{parent(z)}(index(z))})^{\Delta_{k, S'_x}(0)} \\ &= \prod_{z \in S_x} (\hat{e}(g, g)^{r \cdot \beta q_x(k)})^{\Delta_{k, S'_x}(0)} \\ &= \hat{e}(g, g)^{r \cdot \beta q_x(0)} \end{aligned} \tag{8}$$

Then, we define the decryption algorithm by calling $\mathrm{DecryptNode}(\mathrm{CT}, \mathrm{SK}, x)_1$ or $\mathrm{DecryptNode}(\mathrm{CT}, \mathrm{SK}, x)_2$ on the root node R of the access tree $\mathcal{T}$. If the $\mathcal{T}$ is satisfied by S we define $A = \mathrm{DecryptNode}(\mathrm{CT}, \mathrm{SK}, R) = \hat{e}(g, g)^{r\beta q_R(0)} = \hat{e}(g, g)^{r\beta s}$. Thus, the user can gain ck with the formula (9).

$$\begin{aligned} \widetilde{C}/(\hat{e}(C, D)/A) &= \widetilde{C}/(\hat{e}(g^s, g^{\alpha} \cdot h^r)/\hat{e}(g, g)^{r\beta s}) \\ &= ck \cdot \hat{e}(g, g)^{\alpha s}/\hat{e}(g, g)^{\alpha s} \\ &= ck \end{aligned} \tag{9}$$

(2) $\mathbf{Data}.\mathbf{Decrypt}(E_{ck}(M), ck)$. User inputs file ciphertext $E_{ck}(M)$ and content key ck. Based on symmetric decryption algorithm, i.e., DES or AES, the file M can be decrypted as the formula (10), where D_{ck} denotes a symmetric decryption operation with the key ck.

$$D_{ck}[E_{ck}(M)] = M \tag{10}$$

E. Data File Deletion

Here, we show that data file deletion contains not only discretionary deletion but also mandatory blocking.

Discretionary Deletion. All of the legal data owners can freely delete ciphertext in cloud system. Assume that a DO wants to delete an encrypted file, the procedures of algorithm between DO and CSP are described as below, where the algorithm can adopt any secure signature scheme such as BLS short signature scheme [24] as the underlying primitive to achieve.

(1) DO sends a request to CSP, which includes file's ID and its signature on the ID.

(2) CSP verifies these request information. If validation, CSP deletes the corresponding ciphertext. Otherwise, CSP will do nothing.

Mandatory Blocking. To provide legal sharing environment, a new function, i.e., mandatory blocking, is added to the proposed system. The steps are described as below.

(1) When accessing a file, user needs to evaluate how well the file is accessed, such as shopping online and teaching online.

(2) CSP synthesizes these assessments for each file. If some files are not consistent with the evaluation standards, those files would be mandatory blocked by CSP. Meanwhile, DO will receive the private messages explaining the reason. If not, CSP will do nothing.

V. Performance Analysis

In this section, we analyze and compare the efficiency of the proposed scheme with the schemes [11], [12], and [17] in theoretical and experimental aspects.

A. Theoretical Analysis

1) Key Escrow and Weighted Attribute: Table I shows the problems of key escrow and weighted attribute in each scheme which will affect the security and practicability of cloud system. The key escrow in CP-WABE-RE scheme can be removed by using an improved key issuing protocol for cloud computing. [11] uses escrow-free key issuing protocol to solve the issue. On the contrary, both [12] and [17] don't involve the problem of key escrow. In addition, the weighted attribute in CP-WABE-RE scheme can not only support arbitrary-state attributes instead of the traditional binary state, but also simplify access policy associated with a ciphertext as opposed to [11] [12]. Unfortunately, [17] can only express arbitrary-state attributes, and cannot simplify the access structure. In Table I, we can find that only CP-WABE-RE scheme can simultaneously support both the features and cloud system. [11] solves the problem of key escrow so it can satisfy environment of cloud system as ours. However, both [12] and [17] cannot remove key escrow. Thus the both schemes cannot be directly applied in cloud computing.

TABLE I. Feature Comparisons

Scheme	Key Escrow	Weighted Attribute	Cloud System
CP-WABE-RE	No	Yes	Yes
[11]	No	No	Yes
[12]	Yes	No	No
[17]	Yes	Yes	No

2) Efficiency: In Table II and Table III, we compare efficiency of the above four schemes on storage overhead and computation cost in theory, where the used symbols are defined in Table IV.

To simplify the comparisons, access structure, data re-encryption of [11] [12], and dynamic membership management (that is, user joining, leaving, and attribute updating) of [17] are not included in the following analysis. In addition, the cost of transmission isn't involved when implementing the interactive protocols in both [11] and our proposed scheme.

TABLE IV. Notations for Efficiency Comparisons

Notation	Definition
$\mathbb{G}_i$	exponentiation or multiplication in $group$ $(i = 0, T)$
$C_{\hat{e}}$	$\hat{e}$ operation, $\hat{e}$ denotes bilinear pairing
$\mathbb{Z}_p$	$Group$ $\{0, 1, \dots, p-1\}$ under multiplication modulo p
S	Least interior nodes satisfying an access structure
$\mathbb{A}_C$	Attributes appeared in ciphertext CT
$\mathbb{A}_u$	Attributes of user u
ω_i	Maximum weight of attribute i in system
ω_{i_1}	Weight of attribute i in ciphertext CT
n	Number of attributes in system
k	Number of users in system
L_*	Bit-Length of element in *
$\lvert * \rvert$	Number of elements in *

In Table II, the schemes are compared in terms of CT size, SK size, PP size and MSK size. CT size represents the storage overhead in cloud computing and implies the communication cost from DO to CSP, or from CSP to users. SK size denotes the required storage cost for each user. PP and MSK sizes represent the storage overhead of KA and CSP in terms of public parameter and master secret key.

As shown in Table II, when $\omega_i = \omega_{i_1}$, all attributes possess equal weights in CP-WABE-RE scheme. Thus our scheme is equivalent to [11] and [12]. Meanwhile, CT size is reduced as $(|\mathbb{A}_C| + 1)L_{\mathbb{G}_0} + L_{\mathbb{G}_T}$ in CP-WABE-RE scheme, which is equal to [12]'s. Comparing with [11] and [17], the CT size in our proposed scheme and [12] is reduced by nearly half. When $\omega_i \neq \omega_{i_1}$, CP-WABE-RE scheme can use an attribute to express $(\omega_i - \omega_{i_1} + 1)$ attributes which has different weights. Therefore, it requires smaller storage cost in CT than the others. Moreover, we can find that the SK size in CP-WABE-RE scheme is equal to [12]'s, which is smaller than [11]'s and [17]'s. Furthermore, when $|\mathbb{A}_u| \to \infty$, the storage overhead in our scheme is reduced by nearly half comparing to [11]'s. And the storage cost in our scheme is decreased nearly by 66.67% comparing to [17]'s in theory. In addition, we can also observe that the PP size is equal among [11], [12], and CP-WABE-RE scheme. And the size of PP in [17] is the longest since it is related to the number of system attributes n and the number of system users k. About the size of MSK, we can find that the parameter in CP-WABE-RE scheme doesn't appear to be

TABLE II. EFFICIENCY COMPARISONS: STORAGE COST

Scheme	Size of CT	Size of SK	Size of PP	Size of MSK
CP-WABE-RE	$[\Sigma_{i=1}^{\lvert\mathbb{A}_C\rvert}(\omega_i - \omega_{i_1} + 1) + 1]L_{\mathbb{G}_0} + L_{\mathbb{G}_T}$	$(\lvert\mathbb{A}_u\rvert + 2)L_{\mathbb{G}_0}$	$3L_{\mathbb{G}_0} + L_{\mathbb{G}_T}$	$3L_{\mathbb{Z}_p}$
[11]	$(2\lvert\mathbb{A}_C\rvert + 1)L_{\mathbb{G}_0} + L_{\mathbb{G}_T}$	$(2\lvert\mathbb{A}_u\rvert + 1)L_{\mathbb{G}_0}$	$3L_{\mathbb{G}_0} + L_{\mathbb{G}_T}$	$L_{\mathbb{Z}_p} + L_{\mathbb{G}_0}$
[12]	$(\lvert\mathbb{A}_C\rvert + 1)L_{\mathbb{G}_0} + L_{\mathbb{G}_T}$	$(\lvert\mathbb{A}_u\rvert + 2)L_{\mathbb{G}_0}$	$3L_{\mathbb{G}_0} + L_{\mathbb{G}_T}$	$L_{\mathbb{G}_0}$
[17]	$2(\lvert\mathbb{A}_C\rvert + 1)L_{\mathbb{G}_0} + 2L_{\mathbb{G}_T}$	$(3\lvert\mathbb{A}_u\rvert + 1)L_{\mathbb{G}_0}$	$(n + 2)L_{\mathbb{G}_0} + 2L_{\mathbb{G}_T} + knL_{\mathbb{Z}_p}$	$L_{\mathbb{G}_0}$

TABLE III. EFFICIENCY COMPARISONS: COMPUTATION COST

Scheme	New File Creation Cost	Data File Access Cost	New User Authorization Cost
CP-WABE-RE	$\{[\Sigma_{i=1}^{\lvert\mathbb{A}_C\rvert}(\omega_i - \omega_{i_1} + 2)] + 1\}\mathbb{G}_0 + 2\mathbb{G}_T$	$(2\lvert\mathbb{A}_u\rvert + 1)C_{\hat{e}} + (2\lvert S\rvert + 2)\mathbb{G}_T$	$(\lvert\mathbb{A}_u\rvert + 9)\mathbb{G}_0$
[11]	$(2\lvert\mathbb{A}_C\rvert + 1)\mathbb{G}_0 + 2\mathbb{G}_T$	$(2\lvert\mathbb{A}_u\rvert + 1)C_{\hat{e}} + (2\lvert S\rvert + 2)\mathbb{G}_T$	$(2\lvert\mathbb{A}_u\rvert + 4)\mathbb{G}_0$
[12]	$(2\lvert\mathbb{A}_C\rvert + 1)\mathbb{G}_0 + 2\mathbb{G}_T$	$(2\lvert\mathbb{A}_u\rvert + 1)C_{\hat{e}} + (2\lvert S\rvert + 2)\mathbb{G}_T$	$(\lvert\mathbb{A}_u\rvert + 3)\mathbb{G}_0$
[17]	$(2\lvert\mathbb{A}_C\rvert + 2)\mathbb{G}_0 + 3\mathbb{G}_T$	$(3\lvert\mathbb{A}_u\rvert + 1)C_{\hat{e}} + (2\lvert S\rvert + 3)\mathbb{G}_T$	$(4\lvert\mathbb{A}_u\rvert + 2)\mathbb{G}_0$

much different from the others.

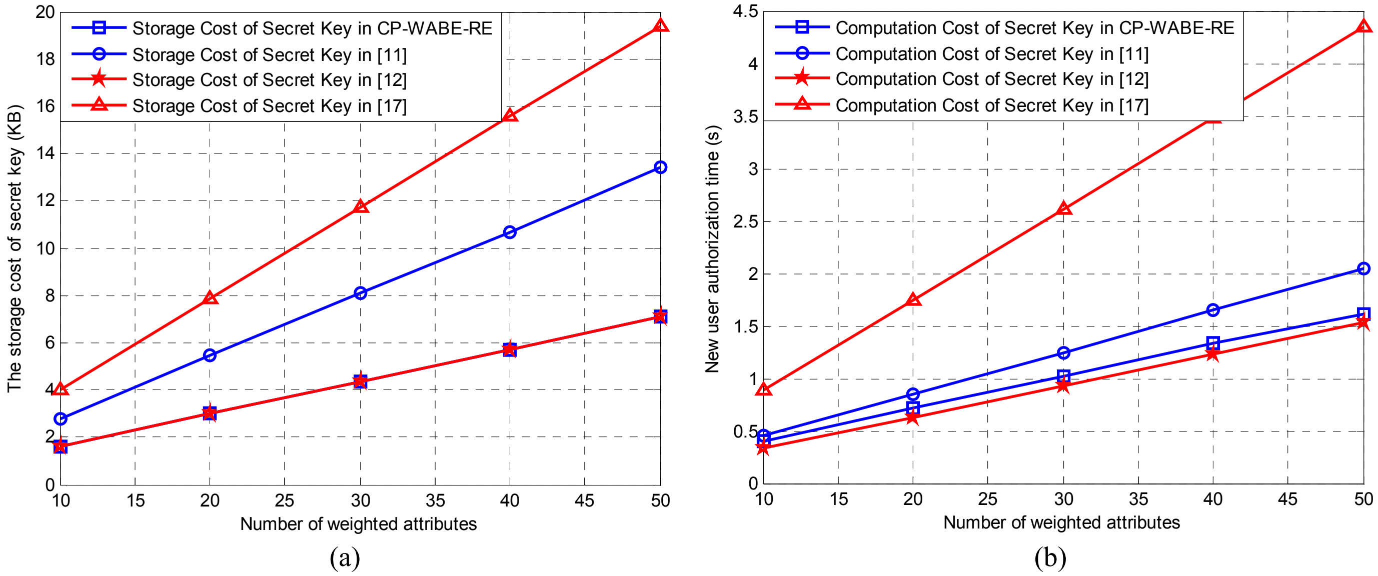
In Table III, we evaluate the computation cost of encryption, decryption and key generation. During the new file creation (data encryption), the computation cost in CP-WABE-RE scheme can be reduced as $(2|\mathbb{A}_C| + 1)\mathbb{G}_0 + 2\mathbb{G}_T$ when $\omega_i = \omega_{i_1}$, which is roughly equal to [11]'s, [12]'s, and [17]'s. Similar to Table II, when $\omega_i \neq \omega_{i_1}$, CP-WABE-RE scheme computes an attribute to represent multiple attributes which possess different weights. Meanwhile, it can simplify access structure associated with a ciphertext. However, the scheme [17] doesn't possess the feature of our scheme, without expressing arbitrary-state attributes. So, when $\omega_i \neq \omega_{i_1}$, the encryption cost in CP-WABE-RE scheme is saved. About data file access (data decryption) cost, the parameter is equal in [11], [12], and CP-WABE-RE scheme. And the decryption cost in [17] is larger than the others. In addition, in the phase of new user authorization (user key generation), our proposed scheme only consumes additional $6\mathbb{G}_0$ of computation cost in solving key escrow issue, comparing with that in [12]. Meanwhile, the key generation cost in CP-WABE-RE scheme is smaller than [11]'s and [17]'s. Furthermore, when $|\mathbb{A}_u| \to \infty$, the computation cost in ours is decreased nearly by 50% in theory than [11]'s, where the cost of transmission isn't involved in the two schemes. At the same time, the cost in ours is reduced by nearly 75% comparing to [17]'s in theory.

B. Experimental Analysis

Now, to validate theoretical analysis proposed in previous subsection, we execute CP-WABE-RE scheme by using the cpabe toolkit and the Java Pairing-Based Cryptography library (JPBC) [25]. Meanwhile, we also simulate the schemes in [11], [12], and [17] at the same condition. The following experiments are conducted using Java on the system with Intel(R) Core(TM) i5-4590 CPU at 3.30 GHz and 8.00GB RAM running Windows 7. To achieve a 80-bit security level, the experiments use a 160-bit elliptic curve group based on the supersingular curve $y^2 = x^3 + x$ over a 512-bit finite field. In addition, all the simulation results are the mean of 10 trials. The units of storage cost and time are Kilobyte (KB) and second (s).

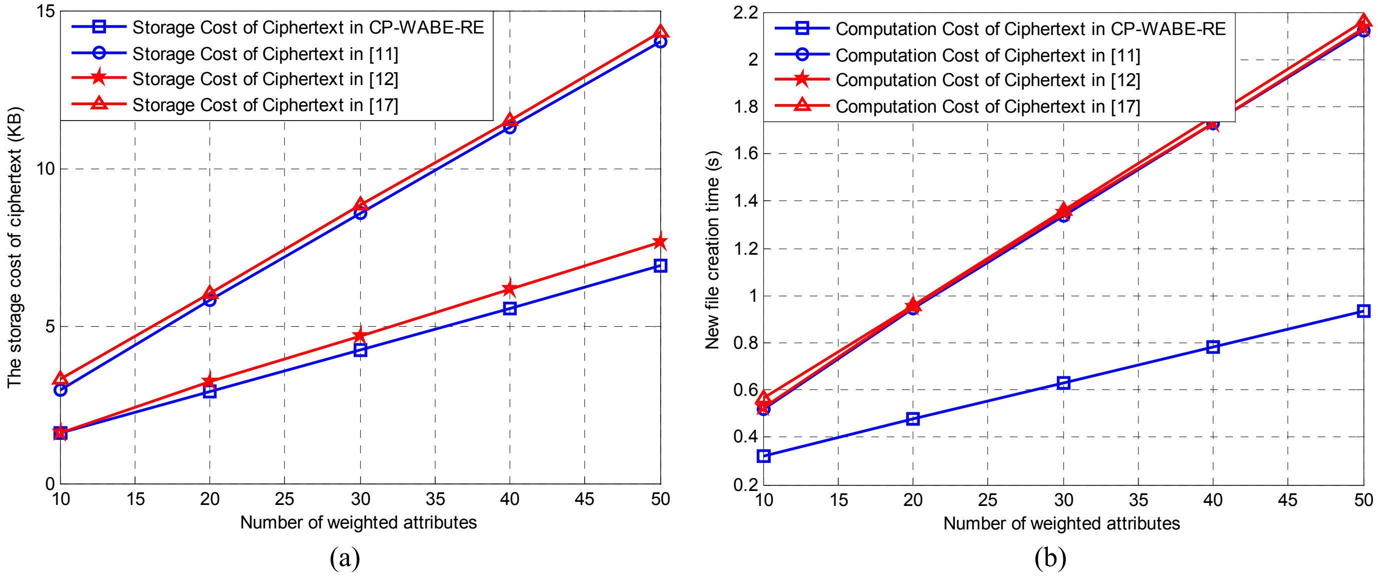
1) Simulation Analysis of Key Escrow: The storage overhead and computation cost of user secret key are compared as plotted in Fig. 6. The number of weighted attributes used in this simulation is $N = \{10, 20, 30, 40, 50\}$.

Fig. 6(a) and Fig. 6(b) intuitively show the experimental results. We find that the storage overhead of user secret key in ours is the same as [12]'s, and it is smaller than [11]'s and [17]'s under the same number of attributes. About the computation cost of secret key, the value of our scheme is larger than [12]'s, where the difference of key generation cost is $6\mathbb{G}_0$ according to the above Table III. At the same time, the parameter in CP-WABE-RE scheme is smaller than [11]'s and [17]'s at the same condition. We also observe that all experimental results are gradually increasing and approximately follow a linear relationship with the number of weighted attributes. Therefore, with a small error tolerance, we estimate their limit values, where the mathematical expressions are computed by using the mean algorithm. When $N \to \infty$, the limit value of space saving in CP-WABE-RE scheme is approximately equal to 48.39% comparing to [11]'s. The cost is reduced by nearly half in theory which is consistent with the above efficiency analysis. Comparing with our scheme and [17], the saved storage cost is approximately 64.47% which matches the corresponding limit value in theory. In addition, when $N \to \infty$, comparing with CP-WABE-RE scheme and [11], the maximum of improved efficiency in computation cost approaches to 23.04%. Comparing with our scheme and [17], the reduced computation cost is approximate to 64.88% in Fig. 6(b). However, in Table III, if $|\mathbb{A}_u| \to \infty$, the corresponding computation cost can be reduced to 50% and 75% in theory, where the cost of transmission isn't involved. Remarkably, the cost of transmission comes from the difference between theoretical value and experimental result.



(a) (b)

Fig. 6. Comparison of experimental results of key escrow. (a)The storage cost of secret key. (b)The time cost of new user authorization. The coordinate is storage overhead or time cost of user's secret key. The abscissa is the number of weighted attributes in user secret key.



(a) (b)

Fig. 7. Comparison of experimental results of weighted attribute. (a)The storage cost of ciphertext. (b)The time cost of new file creation. The abscissa is the number of attributes appeared in ciphertext. The coordinate is storage overhead or time cost of encryption at DO.

2) Simulation Analysis of Weighted Attribute: Next, we measure and analyze the storage overhead and computation cost for encrypting (by a DO) data, where the number of attributes in access policy is $N = \{10, 20, 30, 40, 50\}$. It should be noticed that the CP-WABE-RE scheme is equivalent to the schemes in [11] and [12] when all attributes possess equal weights ($\omega_i = \omega_{i_1}$), where it has been analyzed in Table II and Table III. For simplicity, we have omitted the simulation when $\omega_i = \omega_{i_1}$. To show the advantage of the weighted attribute here, in CP-WABE-RE scheme, the maximum value of each weighted attribute is set to be 5, and the lowest value of each weighted attribute is chosen to encrypt. We implement [11], [12], [17], and our proposed scheme under the equivalent access policy encrypted in ciphertext. Fig. 7 shows the simulation results.

Fig. 7(a) plots the relationship between the storage overhead of ciphertext and the number of weighted attributes in access policy. Fig. 7(b) shows encryption time of ciphertext versus the number of weighted attributes. When $\omega_i \neq \omega_{i_1}$ (here we assume that an attribute can be represented 5 attributes which possess different weights), we find that CP-WABE-RE scheme requires less storage cost and encryption time than the others. We also observe that all results approximately follow a linear relationship with the number of weighted attribute in access trees. Similar as the analysis of Fig. 6, we estimate the limit values with a small error tolerance, where the mathematical expressions are computed by using the mean algorithm. For example, in Fig. 7(a), when $N \to \infty$, comparing with CP-WABE-RE scheme and [11], the limit value of space saving is approximately equal to 52.60%. Similarly, the reduced storage cost in ciphertext is 11.77% comparing our scheme to [12]. And comparing with [17], our proposed scheme can save storage cost approximate to 51.71%. In addition, the reduced storage cost in [12] approaches to 46.28% comparing to [11]. In Fig. 7(b), when $N \to \infty$, CP-WABE-RE scheme can save computation cost approximate to 61.68% comparing with [11], [12], and [17], where the computation cost associated with a ciphertext in [11] is approximately equal to [12]'s and [17]'s. It indicates that the results are consistent with the theoretical analysis presented in previous subsection.

VI. Security Proof

We first present the chosen plaintext attacks (CPA) security proof of our CP-WABE-RE scheme. The security game is identical to those of traditional (fully) CP-ABE systems. We state here the definition of an adaptive CP-ABE security game for the completeness of the security analysis.

1) **System Initi.** The challenger runs the operations of $\mathbf{KA}.\mathbf{Setup}$ and $\mathbf{CSP}.\mathbf{Setup}$ of CP-WABE-RE scheme and sends public parameter PP to the adversary $\mathcal{A}$.
2) **Phase 1.** For the attribute sets $S_1, \ldots, S_{q1} (\forall i \in [1, \ldots, q1])$ chosen by $\mathcal{A}$, he can repeatedly ask $\mathcal{C}$ for the secret key SK. Meanwhile, the challenger answers the secret key SK by running the algorithms of $\mathbf{CSP}.\mathbf{KeyGen}$ and $\mathbf{KA}.\mathbf{KeyGen}$.
3) **Challenge.** $\mathcal{A}$ submits two equal length messages $M_0, M_1 \in \mathbb{G}_T$ and an access tree $\mathbb{A}$ to the challenger, where there should not be any secret key issued to $\mathcal{A}$ such that the key satisfies $\mathbb{A}$. The challenger randomly picks a bit $\mu \in \{0, 1\}$ and encrypts M_μ with $\mathbb{A}$ by using the algorithm $\mathbf{DO}.\mathbf{Encrypt}$.
4) **Phase 2.** Same as the **Phase 1** but with the restriction that the querying key cannot satisfy $\mathbb{A}$.
5) **Guess.** $\mathcal{A}$ outputs a guess $\hat{\mu}$ of μ. In this game, $\mathcal{A}$ can win the game which is defined as $|Pr[\hat{\mu} = \mu] - (1/2)|$.

Definition 2. *The proposed scheme is said to be secure against CPA if no probabilistic polynomial-time adversaries have non-negligible advantage in the above game.*

We use the generic bilinear group model and the random oracle model to prove that no adversary can break the CPA security of our scheme with non-negligible probability. In other words, our security is reduced to mathematical properties of elliptic curve groups as well as security of target collision resistance hash function. We note that our security proof technique follows that of [4].

Consider two random encodings ξ_0, ξ_1 of an additive group $\mathbb{F}_p$, which is injective maps $\xi_0, \xi_1 : \mathbb{F}_p \to \{0, 1\}^m$, where $m > 3log(p)$. We set $\mathbb{G}_0 = \{\xi_0(x) : x \in \mathbb{F}_p\}$ and $\mathbb{G}_T = \{\xi_1(x) : x \in \mathbb{F}_p\}$. In the security game, the simulator is given a random oracle for simulating hash function, and oracles in groups $\mathbb{G}_0, \mathbb{G}_T$ and bilinear map $\hat{e} : \mathbb{G}_0 \times \mathbb{G}_0 \to \mathbb{G}_T$ for computation queries. We are also given a random oracle to represent the hash function H. And we refer to $\mathbb{G}_0$ as a generic bilinear group. Below, we give a lower bound on the advantage of a generic adversary in breaking the security of our scheme.

Theorem 1. *For any adversary $\mathcal{A}$, let q be a bound on the total number of group elements which $\mathcal{A}$ receives from queries to the oracles for the hash function, groups $\mathbb{G}_0$, $\mathbb{G}_T$, the bilinear map $\hat{e}$ and from its interaction with the security game, in which $\mathbb{G}_0$ is bilinear group of prime order p with generator g. We have that the advantage of $\mathcal{A}$ in the game is $O(q^2/p)$.*

Proof. In the challenge phase of a CP-ABE game, the simulator will construct either $M_0\hat{e}(g, g)^{\alpha s}$ or $M_1\hat{e}(g, g)^{\alpha s}$ as the component $\tilde{C}$. Here, we consider a modified game where $\tilde{C}$ is either $\hat{e}(g, g)^{\alpha s}$ or $\hat{e}(g, g)^{\theta}$, and $\theta \in_R \mathbb{F}_p$. Now, the adversary is required to tell if $\tilde{C} = \hat{e}(g, g)^{\alpha s}$ or $\hat{e}(g, g)^{\theta}$. It is not difficult to see that the modified game can be regarded as a hybrid argument in which the adversary is asked to tell $\hat{e}(g, g)^{\theta}$ from $M_0\hat{e}(g, g)^{\alpha s}$, and $\hat{e}(g, g)^{\theta}$ from $M_1\hat{e}(g, g)^{\alpha s}$. Accordingly, an adversary in the CP-ABE game with advantage ϵ is transformed into an adversary $\mathcal{A}$ in the modified game with advantage at least $\frac{\epsilon}{2}$. Below we let $g = \xi_0(1)$, $g^x = \xi_0(x)$ and $\hat{e}(g, g)^x = \xi_1(x)$.

1) **System Initi.** The simulator chooses $\alpha_1, \alpha_2, \beta \in_R \mathbb{F}_p$, and next sets $h = g^\beta$, $u_1 = \hat{e}(g, g)^{\alpha_1}$, $v_1 = g^{\alpha_1}$, $u_2 = \hat{e}(g, g)^{\alpha_2}$, $v_2 = g^{\alpha_2}$ and $\alpha = \alpha_1 + \alpha_2$. It further sends the PP $= \{g, h, u = u_1 u_2\}$ to $\mathcal{A}$.
2) **Hash Queries.** If $\mathcal{A}$ issues a hash query on an attribute $att(y)$, the simulator returns g^{t_i} and stores $(t_i, att(y))$ into $List^H$, where $t_i \in_R \mathbb{F}_p$.
3) **Key Queries.** Here we combine the simulations of the algorithms **CSP.KeyGen** and **KA.KeyGen** as one key query. We note that it will not bring additional advantage for $\mathcal{A}$ in winning the game. When $\mathcal{A}$ queries a user i's secret key for an attribute set S, the simulator works as follows. It chooses $r_i, w_j \in_R \mathbb{F}_p$, and computes $D = g^\alpha g^{\beta r_i}$, $L = g^{r_i}$, $\forall\ j \in S :\ D_j = H(j)^{r_i w_j}$. The simulator finally sends the secret key to $\mathcal{A}$ and stores (SK, i, S) into $List^{SK}$.
4) **Challenge.** $\mathcal{A}$ outputs two equal length messages $M_0, M_1 \in \mathbb{G}_T$ and an access tree $\mathbb{A}$ to the simulator, where there should not be any secret key issued to $\mathcal{A}$ such that the key satisfies $\mathbb{A}$. The simulator chooses an $s \in_R \mathbb{F}_p$, and next uses linear secret sharing technique to construct shares λ_y of s for all attributes y in $\mathbb{A}$ as in the algorithm **DO.Encrypt**, where λ_y is uniformly and independently random in $\mathbb{F}_p$, and λ_y can be seen as a linear combination of independent random variables (in $\mathbb{F}_p$) and s. The simulator then chooses $\theta \in_R \mathbb{F}_p$,

and sets $\tilde{C} = \hat{e}(g,g)^{\theta}$, $C = g^{s}$, $\forall\ y \in Y$, $i \in [1,n]$, $C_y = g^{\beta\lambda_i} H(att(y))^{-w_i s}$, and $\forall j \in (1,n]$, $C_{y,j} = H(att(y))^{-(w_j - w_i)s}$, where $w_i, w_j \in_R \mathbb{F}_p$. The simulator sends the challenge ciphertext to $\mathcal{A}$.

5) **Key Queries.** Same as the previous key queries phase but with the restriction that the querying key cannot satisfy $\mathbb{A}$.
6) **Guess.** $\mathcal{A}$ outputs a guess bit.

Below we consider unexpected collision. An oracle query can be seen as a rational function $\vartheta = \eta/\psi$ in the variables θ, α, β, t_i, w_i, r_i, λ_i and s. Suppose there are two distinct rational functions $\vartheta = \eta/\psi$ and $\vartheta' = \eta'/\psi'$. An unexpected collision event indicates that taking two different queries corresponding to the two functions, we have the same output due to random choice of variables. If the event happens, it means that $\vartheta = \vartheta'$, and further $\eta\psi' - \eta'\psi = 0$. By the Schwartz-Zippel Lemma in [26], [27], the probability of the event is $O(1/p)$. Therefore, the probability of a collision event is at most $O(q^2/p)$. Accordingly, the unexpected collision will not occur in the simulations with probability $1 - O(q^2/p)$.

Remember that each group element is uniformly and dependently chosen in the above simulations. $\mathcal{A}$ can tell the difference between θ and αs elements in $\mathbb{G}_T$ if there are two distinct queries ϑ and ϑ' leading to the same output. Assume $\vartheta' = \gamma'\theta$ and $\vartheta = \gamma\alpha s$, we have $\vartheta - \vartheta' = \gamma\alpha s - \gamma'\theta$ such that $\gamma'\theta + \vartheta - \vartheta' = \gamma\alpha s$, where γ, γ' are non-zero constant. We will show that $\mathcal{A}$ cannot construct a query for $\gamma\alpha s$ in $\mathbb{G}_T$.

We here observe all possible rational function queries in $\mathbb{G}_T$ by means of bilinear map and the group elements given to $\mathcal{A}$. It can be seen that $\mathcal{A}$ can obtain a transcript $\{g$, g^{β}, g^{s}, $g^{\beta\lambda_i} g^{-t_i w_i s}$, $g^{-(w_j - w_i)st_i}$, $g^{\alpha} g^{\beta r}$, g^{r}, $g^{r w_i t_i}\}$ from one query. For another transcript, we set it as $\{g$, g^{β}, g^{s}, $g^{\beta\lambda_{i'}} g^{-t_{i'} w_{i'} s}$, $g^{-(w_{j'} - w_{i'})st_{i'}}$, $g^{\alpha} g^{\beta r'}$, $g^{r'}$, $g^{r' w_{i'} t_{i'}}\}$. We first ignore g^{β} since the elements with α will be tagged with β which is irrelevant to αs. To output a factor αs, we should focus on elements with factors α and s. It is not difficult to see that there are three types of outputs with αs in $\mathbb{G}_T$ from two transcripts. One is $\alpha s + \beta r s$ (resp. $\alpha s + \beta r' s$). To output $\gamma\alpha s$, we need an element $\beta r s$. However the element does not exist. The second format with αs is $(-w_j + w_i)t_i\alpha s + (-w_j + w_i)st_i\beta r'$ (resp. $(-w_{j'} + w_{i'})t_{i'}\alpha s + (-w_{j'} + w_{i'})st_{i'}\beta r$). To eliminate the part right after $+$, we need to concentrate on the elements with t_i. The elements with $\beta\lambda_i - t_i w_i s$, and $r w_i t_i$ fail to construct a cancel-out part as they are lack of a factor w_j. For the element $-(w_j - w_i)st_i$, we need an element with $\beta r'$ (resp. βr). But none of other elements satisfy our requirement. The last format with αs is $-t_i w_i \alpha s + \beta^2\lambda_i r' - t_i w_i s\beta r' + \alpha\beta\lambda_i$ (resp. $-t_{i'} w_{i'} \alpha s + \beta^2 \lambda_{i'} r - t_{i'} w_{i'} s\beta r + \alpha\beta\lambda_{i'}$). If we cannot find elements to cancel out all of terms except for that of αs, it indicates that $\mathcal{A}$ fails to construct $\gamma\alpha s$. For simplicity, we only check with the last term $\alpha\beta\lambda_i$. It can be seen that $\alpha + \beta r$ is the only element with α. Thus, we need a $\beta\lambda_i$ term. Nevertheless, the term does not exist.

From the above observation, we can therefore state that $\mathcal{A}$ fails to construct the query form $\gamma\alpha s$.

In addition, an improved key issuing protocol is proposed to resolve the key escrow problem of CP-ABE in cloud computing. In this paper, we assume that they do not collude with each other to share their master secret keys. We say that our proposed key issuing protocol is secure when the following two aspects are satisfied. The first one is that the KA cannot derive the user secret key if the CSP is honest. The other is that the CSP cannot derive the user secret key while the KA is honest. Security analysis about the protocol is described as below.

Theorem 2. *The proposed key issuing protocol in section IV-C is a secure protocol for computing $g^{\alpha}h^{r}$ by KA and CSP. Assume that the underlying arithmetic 2PC and zero knowledge proofs are secure, and (for security against corrupt CSP) that DDH is hard.*

Proof. First, to note that $D = Y_3^{1/\rho_2} = X_2^{1/\rho_2\tau} = (Y_1^{\rho_1} Y_2)^{1/\tau} = X_1^{\rho_1/\beta} h^{r} = g^{\alpha_1 + \alpha_2} h^{r} = g^{\alpha} g^{\beta r}$. To show the security we consider the cases of corrupting KA and corrupting CSP respectively.

(1) For a corrupted KA, our simulator proceeds as follows:

$\mathsf{Sim}_{\mathsf{C}}$: First, it will run the arithmetic 2PC simulator for computation of $(\alpha_1 + \alpha_2)\beta$. In the process, it will extract α_1. Next, the simulator will choose random values $X_1 \in \mathbb{G}_0$, and send it to KA. It will receive Y_1 and Y_2 from the adversary KA, and two corresponding zero knowledge proofs. We will extract β and r from the corresponding proofs. Then, it will send α_1, β and r to the trusted party, and receive $g^{\alpha_2} \cdot g^{\alpha_1 + \beta r} = g^{\alpha + \beta r}$, which will be CSP's secret key output.

Consider a hybrid simulator Hyb_C that takes as input of CSP's secret α_2. It first runs the arithmetic 2PC simulator for the computation of x with the correct output value according to α_2. Then the simulator completes the protocol as the honest CSP would do. This is clearly indistinguishable from the real CSP's protocol by the security of the arithmetic 2PC.

Now, assuming that the proof of knowledge scheme is secure, Hyb_C should be indistinguishable from the above simulator Sim_C. This is because the value X_1 used by Sim_C will be distributed identically to those in Hyb_C. (Since ρ_1 is chosen at random in the real protocol, X_1 will be distributed uniformly over $\mathbb{G}_0$ in the real protocol as in the simulated protocol.) Thus, interaction with our simulation is indistinguishable from interaction with an honest CSP.

(2) For a corrupted CSP, our simulator proceeds as follows:

$\mathsf{Sim}_{\mathsf{K}}$: First, it will run the arithmetic 2PC simulator for computation of $(\alpha_1 + \alpha_2)\beta$. This 2PC will extract α_2 from CSP and output $x = (\alpha_2 + \alpha_1)\beta \bmod p$. We will choose a random value $x \in \mathbb{Z}_p$, and give it to the arithmetic 2PC simulator. Note that this is correctly distributed, since there is some β such that $x = (\alpha_2 + \alpha_1)\beta \bmod p$ for any x, α_1, α_2. Next, our simulator will receive X_1 from the adversary, and the corresponding zero knowledge proof. ρ_1 is extracted by the proof system. We will select random values $Y_1, Y_2 \in \mathbb{G}_0$, and send them to CSP. (Again, this will be distributed exactly as in a real execution.) We will receive X_2 from the adversary, and use the corresponding proof to extract ρ_2. Then, it will send α_2 to the trusted party, and receive $D = g^{\alpha_2} \cdot g^{\alpha_1 + \beta r} = g^{\alpha + \beta r}$. Finally, it will compute $Y_3 = D^{\rho_2}$ and send it to CSP.

Consider a hybrid simulator Hyb_K that takes as input of KA's secrets α_1, β and r. It will compute $x = (\alpha_2 + \alpha_1)\beta$

using the arithmetic 2PC simulator. When the 2PC simulator provides α_2 and asks for output, it will correctly compute $(\alpha_2 + \alpha_1)\beta$. Then it will complete the execution as in the real protocol. This protocol is clearly indistinguishable from the real KA's protocol by security of the arithmetic 2PC.

In addition, we consider a second hybrid $Hyb_K^{'}$ which is the same as the Hyb_K to proceed the above protocol, but which uses the zero-knowledge simulator for all proofs of knowledge. This must be indistinguishable by the zero-knowledge property of the proof system. Here we only need show that the $Hyb_K^{'}$ is indistinguishable from the interaction with the above simulator.

Consider the reduction from DDH assumption: Given g, $A = g^a$, $B = g^b$, $C = g^c$, and we must decide whether $c = ab$ or $c \in_R \mathbb{Z}_p$. Here we define $X_1 = A = g^a$ and $h = g^{\sigma}$ for $\sigma \in_R \mathbb{Z}_p$. As the $\mathsf{Sim_K}$, we run the arithmetic 2PC simulator for computation of $(\alpha_1 + \alpha_2)\beta$ and extraction α_2. Next we receive $X_1 = A$ and extract ρ_1 from the corresponding proof. Meanwhile, we compute $Y_1 = X_1^{\theta/\beta} = g^{a\theta/\beta} = C^{1/\beta}$, $Y_2 = h^{r\theta} = g^{\sigma r\theta} = B^{\sigma r}$, and send them to the adversary CSP, along with a simulated proof of knowledge. Then we receive X_2 and extract ρ_2 from the proof. At last, we compute $Y_3 = X_2^{1/\theta} = (A^{\rho_1/\beta} \cdot g^{\theta r})^{\rho_2}$ and send it to CSP.

Here we assume that the proofs of knowledge are secure. If $c = ab$, Y_1, Y_2, Y_3 will be distributed correctly, and this will be distinguishable from $Hyb_K^{'}$. If c is a random number (that is, $c \in_R \mathbb{Z}_p$), then Y_1, Y_2 are randomly selected from $\mathbb{G}_0$, as in $\mathsf{Sim_K}$. Thus, any adversary that can distinguish $Hyb_K^{'}$ from $\mathsf{Sim_K}$ will allow us to resolve DDH problem. Under the DDH assumption, interaction with $\mathsf{Sim_K}$ is indistinguishable from interaction with a real KA.

Thus, our construction is a secure two-party key issuing protocol.

VII. Conclusion

In this paper, we revisited an attribute-based data sharing scheme in cloud computing. The improved key issuing protocol was presented to resolve the key escrow problem. It enhances data confidentiality and privacy in cloud system against the managers of KA and CSP as well as malicious system outsiders, where KA and CSP are semi-trusted. In addition, the weighted attribute was proposed to enhance the expression of attribute, which can not only describe arbitrary-state attributes, but also reduce the complexity of access policy, so that the storage cost of ciphertext and time cost in encryption can be saved. Finally, we presented the performance and security analyses for the proposed scheme, in which the results demonstrate high efficiency and security of our scheme.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (61171072, 61472083 and 61170283), the Science and Technology Projects of Shenzhen (ZDSYS20140430164957660), and National High-Technology Research and Development Program ("863" Program) of China under Grant 2013AA01A212.

References

[1] T. Paul, A. Famulari, and T. Strufe, "A survey on decentralized online social networks," *Computer Networks*, pp. 1–16, 2014.

[2] M. Yang and Y. Yang, "An efficient hybrid peer-to-peer system for distributed data sharing," *IEEE Transactions on Computers*, vol. 59, no. 9, pp. 1158–1171, 2010.

[3] C. Wang, S. S. Chow, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for secure cloud storage," *IEEE Transactions on Computers*, vol. 62, no. 2, pp. 362–375, 2013.

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

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

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

[7] H. Deng, Q. Wu, B. Qin, J. Domingo-Ferrer, L. Zhang, J. Liu, and W. Shi, "Ciphertext-policy hierarchical attribute-based encryption with short ciphertexts," *Information Sciences*, vol. 275, pp. 370–384, 2014.

[8] A. Balu and K. Kuppusamy, "An expressive and provably secure ciphertext-policy attribute-based encryption," *Information Sciences*, vol. 276, pp. 354–362, 2014.

[9] Z. Zhou, D. Huang, and Z. Wang, "Efficient privacy-preserving ciphertext-policy attribute based encryption and broadcast encryption," *IEEE Transactions on Computers*, vol. 64, no. 1, pp. 126–138, 2015.

[10] M. Chase and S. S. Chow, "Improving privacy and security in multi-authority attribute-based encryption," *Proceedings of the 16th ACM Conference on Computer and Communications Security*, pp. 121–130, 2009.

[11] J. Hur, "Improving security and efficiency in attribute-based data sharing," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 10, pp. 2271–2282, 2013.

[12] X. Xie, H. Ma, J. Li, and X. Chen, "An efficient ciphertext-policy attribute-based access control towards revocation in cloud computing," *Journal of Universal Computer Science*, vol. 19, no. 16, pp. 2349–2367, 2013.

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

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

[15] X. Liu, J. Ma, J. Xiong, Q. Li, and J. Ma, "Ciphertext-policy weighted attribute encryption for fine-grained access control," *Processing of the 5th International Conference on Intelligent Networking and Collaborative Systems*, pp. 51–57, 2013.

[16] X. Liu, J. Ma, J. Xiong, and G. Liu, "Ciphertext-policy hierarchical attribute-based encryption for fine-grained access control of encryption data," *International Journal of Network Security*, vol. 16, no. 6, pp. 437–443, 2014.

[17] C. Fan, S. Huang, and H. Rung, "Arbitrary-state attribute-based encryption with dynamic membership," *IEEE Transactions on Computers*, vol. 63, no. 8, pp. 1951–1961, 2014.

[18] M. Chase, "Multi-authority attribute based encryption," *Theory of Cryptography*, vol. 4397, pp. 515–534, 2007.

[19] S. Müller, S. Katzenbeisser, and C. Eckert, "Distributed attribute-based encryption," *Information Security and Cryptology–ICISC 2008*, vol. 5461, pp. 20–36, 2009.

[20] S. S. Chow, "Removing escrow from identity-based encryption," *Public Key Cryptography–PKC 2009*, pp. 256–276, 2009.

[21] P. Morillo, C. Padró, G. Sáez, and J. L. Villar, “Weighted threshold secret sharing schemes,” *Information Processing Letters*, vol. 70, no. 5, pp. 211–216, 1999.

[22] L. Ibraimi, M. Petkovic, S. Nikova, P. Hartel, and W. Jonker, “Mediated ciphertext-policy attribute-based encryption and its application,” *Information Security Applications*, pp. 309–323, 2009.

[23] M. Belenkiy, J. Camenisch, M. Chase, M. Kohlweiss, A. Lysyanskaya, and H. Shacham, “Randomizable proofs and delegatable anonymous credentials,” *Advances in Cryptology-CRYPTO 2009*, pp. 108–125, 2009.

[24] D. Boneh, B. Lynn, and H. Shacham, “Short signatures from the weil pairing,” *Advances in cryptology-ASIACRYPT*, pp. 514–532, 2001.

[25] A. De Caro and V. Iovino, “JPBC: java pairing based cryptography,” *IEEE Symposium on Computers and Communications*, pp. 850–855, 2011.

[26] J. T. Schwartz, “Fast probability algorithms for verification of polynomial identities,” *Journal of the ACM*, vol. 27, no. 4, pp. 701–717, 1980.

[27] R. Zippel, “Probabilistic algorithms for sparse polynomials,” *EUROSAM*, vol. 72, pp. 216–226, 1979.