Ciphertext-Policy Attribute-Based Encryption with User and Authority Accountability

Xing Zhang[1](✉), Cancan Jin[2], Cong Li[2], Zilong Wen[2], Qingni Shen[2], Yuejian Fang[2], and Zhonghai Wu[2]

[1] School of Electronics Engineering and Computer Science, Peking University, Beijing, China
novostary@163.com
[2] School of Software and Microelectronics, Peking University, Beijing, China
jincancan1992@126.com, li.cong@pku.edu.cn, 450275803@qq.com, {qingnishen,fangyj,zhwu}@ss.pku.edu.cn

Abstract. To ensure the security of sensitive data, people need to encrypt them before uploading them to the public storage. Attribute-based encryption (ABE) is a promising cryptographic primitive for fine-grained sharing of encrypted data. However, ABE lacks user and authority accountability. The user can share his/her secret key without being identified, while key generation center (KGC) can generate any user's secret key. In this paper, we propose a practical large universe ciphertext-policy ABE (CP-ABE) with user and authority accountability in the white-box model. As embedding the user's identity information into this user's secret key directly, the trace stage has only $O(1)$ time overhead. The property of accountability is proved against the dishonest user and KGC in the standard model. We implement our scheme in Charm. Experiments show that CP-ABE of Rouselakis and Waters in CCS 2013 is enhanced in user and authority accountability by our method with small computational cost.

Keywords: Attribute-Based Encryption · User accountability · Authority accountability · White-box model

1 Introduction

Cloud computing is changing the way we deliver large-scale web applications. Various computing resources are delivered as services over the Internet. The openness and sharing of cloud has caused important issues of information security. More and more enterprises and individuals choose to put their data into the cloud. However, cloud service providers are generally assumed to be untrusted parties, that is, they may be curious about the content of their users' data for advertising or even sell the data to data owner's competitors. A natural solution is that data owners should encrypt sensitive data before outsourcing them. Attribute-based encryption (ABE), as an excellent cryptographic access control mechanism, is quite preferable for sharing of encrypted outsourced data.

The concept of ABE was first proposed by Sahai and Waters in 2005 [21]. Then ABE comes into two flavors, key-policy ABE (KP-ABE) [10,19,2] and ciphertext-policy ABE (CP-ABE) [4,8,25]. In KP-ABE, ciphertexts are associated with sets of attributes and user's secret keys are associated with access structures. When ciphertexts are created, data owners do not know who will have access to them later.

B. Thuraisingham et al. (Eds.): SecureComm 2015, LNICST 164, pp. 500–518, 2015.
DOI: 10.1007/978-3-319-28865-9_27

KP-ABE focuses on the specific need of user. Whatever user needs, key generation center (KGC) will generate secret keys corresponding to the proper access structures. In CP-ABE, the situation is the opposite. Users' secret keys are labeled by attributes and ciphertexts are associated with access structures. Before encrypting, the data owner clearly knows what kind of people is allowed to access.

Nevertheless, ABE has a major drawback which is known as the lack of user accountability. As secret keys do not include identity information, a dishonest user need not worry about being caught if this user shares his/her secret key with others or produces a pirate decryption device and sells it on the Internet. Almost all ABE systems suffer from this problem which does not exist in traditional public key encryption (PKE) as users' public keys are certificated with their identities by public key infrastructure (PKI). Thus the general method for user accountability is to embed the identity-related information to the user's secret key. Notice that ABE is a one-to-many communication and its public key in the conventional sense consists of public parameters and attribute sets.

In addition, there is also another problem named the lack of authority accountability. As KGC in ABE has the power to generate secret key for any user with any attribute set, it is hard to distinguish whether the traitor founded by using the technique of user accountability is innocent or not. The general method is to embed secret information which is hidden from the KGC's view into the user's secret key. That secret information can be called key family number [9], which means there are a cluster of secret keys related with each user. We can tell that KGC is to blame if the key family number of the suspected key does not match with that of the accessible users.

There are two models about accountability, white-box model and black-box model. In white-box model, we can get the content of secret key of suspected user. While in black-box model, the secret key is encapsulated in a decryption box. A judge should be able to decide if this box was created by a dishonest user or KGC only by constructing the input and observing the output of the box. Notice that Liu et al. [16] use the word "traceability" other than "accountability". In this paper, they are used interchangeably.

1.1 Related Work

In ABE, most of the concern is user accountability [11] which assuming that the KGC can be trusted. Hinek et al. [11] proposed a token-based ABE. When decrypting, users must request a decryption token from a third party token server. Therefore, the token server is required to be online. Yu et al. [26] proposed a KP-ABE scheme by combining anonymous ABE with traitor tracing in broadcast environments. The content provider would choose particular types of ciphertexts and trick pirate devices into decrypting them. Li et al. [13] proposed an accountable, anonymous CP-ABE. User accountability can be achieved in black-box model by embedding additional user-specific information into the attribute secret key. Liu et al. proposed white-box [16] and black-box [15] traceable CP-ABE respectively. Both can support any monotone access structures while the schemes prior to Liu et al.'s work only support AND gate with wildcards. However, both schemes use bilinear groups of large composite order and are inefficient. Ning et al. [18] proposed a large universe CP-ABE with user accountability in white-box model on bilinear groups of prime order. "Large universe" means that a scheme can support flexible number of attributes. Liu and Wong [17] proposed both large universe KP-ABE and

CP-ABE with user accountability in black-box model on bilinear groups of prime order. The scheme supports revocation for the dishonest user.

Wang et al. [24] achieved authority accountability in white-box model by combining accountable authority identity-based encryption (IBE) [14] and KP-ABE [10]. As the user's secret key contains the secret information unknown to KGC, if KGC forges secret key in accordance with the user's identity, we can find whether KGC or the user is dishonest according the key family number. But yet it does not support large universe.

In multi-authority ABE [6,7], different authorities operate simultaneously and each hands out a user's partial secret key for a different set of attributes. Li et al. [12] proposed a multi-authority CP-ABE scheme with user accountability. However, it only supports access structure with AND gate with wildcards.

1.2 Our Contributions

The main contributions of our work can be summarized as follows.

1) We propose a ciphertext-policy attribute-based encryption scheme with user and authority accountability (UaAA-CP-ABE) in white-box model.

2) Our scheme has the property of large universe and is proved selectively secure in the standard model. The accountability property is also proved against dishonest user and KGC in the standard model.

3) By embedding a user's identity into this user's secret key directly, the only thing needed to do is to check whether the suspected secret key is well-formed at trace stage. If that key is well-formed, we can easily find out the dishonest user or KGC. It is more practical than existing ones [16,18]. More analysis can be seen at Section 1.3.

4) Our scheme is very efficient. We enhance CP-ABE of Rouselakis and Waters [20] in user and authority accountability with small computational cost.

We compare our work with other related works in Table 1.

1.3 Our Main Ideas

In this section we will briefly describe the main ideas in our scheme.

We extend large universe CP-ABE of Rouselakis and Waters [20] to support accountability for user and authority. To find out the identity of the dishonest user, Liu et al. [16] use an identity table to connect the user's identity with secret key. Therefore, the table grows linearly with the number of users in the system. To address this issue, Ning et al. [18] remove the identity table and use Shamir's threshold scheme [23] to trace the dishonest user. As every user has a unique identity ID in the system, can we embed ID into the user's secret key directly? If succeeded, the trace stage would become very simple, the only thing is to check whether the suspected secret key is well-formed or not. Liu et al. [16] in their extensions give some suggestions by using another signature scheme in [5]. However, they do not give a complete construction and proof. And their scheme uses bilinear groups of composite order and merely supports user accountability in white-box model. In our scheme, we successfully embed the signature scheme in [5] into our prime order construction and give complete proof.

Table 1. Comparisons with other related works

Schemes	Category	Large Universe	Supporting Monotonic Access Structure	Order of Bilinear Groups	User Accountability	Authority Accountability	Security Model[1]
LRZ+[13]	CP-ABE	×	×	prime	black-box[2]	×	selectively secure
WCL+[24]	KP-ABE	×	✓	prime	white-box	white-box	selectively secure
LCW[16]	CP-ABE	×	✓	compo-site	white-box	×	Fully secure
LCW[15]	CP-ABE	×	✓	compo-site	black-box	×	fully secure
NCD+[18]	CP-ABE	✓	✓	prime	white-box	×	selectively secure
LW[17]	KP-ABE CP-ABE	✓	✓	prime	black-box	×	selectively secure
RW[20]	CP-ABE	✓	✓	prime	×	×	selectively secure
Ours	CP-ABE	✓	✓	prime	white-box	white-box	selectively secure

[1]All schemes are secure in the standard model.

[2][16] gives a "compare-before-output" technique to avoid the tracing algorithm from identifying the dishonest user in Appendix A.

In order to achieve authority accountability, we borrow some ideas from accountable authority IBE [9]. Nevertheless, in IBE, both secret key and ciphertext contain the user's specific identity information. In ABE, the ciphertext is used for sharing and cannot contain the user's specific identity information. However, we finally succeed in embedding secret information hidden from the KGC's view into the user's secret key. We owe it to the secret key and ciphertext structure of Rouselakis and Waters [20] which employ "attribute" layer and "secret sharing" layer and use a "binder term" to connect them. We can embed secret information into the "secret sharing" layer in the user's secret key and need not change the ciphertext. This trick does not affect the normal computation in the decryption phase other than a change in exponential factor.

1.4 Organization

The remainder of the paper is organized as follows. Section 2 introduces the background. In Section 3, we give the formal definition of UaAA-CP-ABE and its security model. Section 4 proposes the construction of our UaAA-CP-ABE scheme. In Section 5, we analyze our proposed scheme in terms of security and performance. Finally, we give a brief conclusion in Section 6.

2 Background

2.1 Access Structures and Linear Secret Sharing Schemes

Definition 1. *(Access Structures [3])* Let $\{P_1, P_2, \ldots, P_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^{\{P_1, P_2, \ldots, P_n\}}$ is monotone if $\forall \mathrm{B}, \mathrm{C}$: 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 $\{P_1, P_2, \ldots, P_n\}$, i.e., $\mathbb{A} \subseteq 2^{\{P_1, P_2, \ldots, P_n\}} \backslash \{\emptyset\}$. The sets in $\mathbb{A}$ are called the authorized sets, and the sets not in $\mathbb{A}$ are called the unauthorized sets.

In our context, the role of the parties is taken by the attributes. In this paper, we mainly focus on monotone access structure.

Definition 2. *(Linear Secret Sharing Schemes (LSSS) [3])* A secret sharing scheme Π over a set of parties $\mathcal{P}$ is called linear (over $\mathbb{Z}_p$) if

1) The shares for each party form a vector over $\mathbb{Z}_p$.

2) There exists a matrix an M with l rows and n columns called the share-generating matrix for Π. For all $i = 1, \ldots l$, the i^{th} row of M we let the function ρ defined the party labeling row i as $\rho(i)$. When we consider the column vector $\vec{z} = (s, z_2, \ldots, z_n)^T$, where $s \in \mathbb{Z}_p$ is the secret to be shared, and $z_2, \ldots, z_n \in \mathbb{Z}_p$ are randomly chosen, the $\mathrm{M}\vec{z}$ is the vector of l shares of the secret s according to Π. The share $(\mathrm{M}\vec{z})_i$ belongs to party $\rho(i)$.

According to [3], every LSSS according to the above definition also enjoys the linear reconstruction property. Suppose that Π is an LSSS for the access structure $\mathbb{A}$. Let $\mathcal{S}$ be any authorized set if $\mathbb{A}(\mathcal{S}) = 1$, and let $I \subset \{1,2, \ldots, l\}$ be defined as $I = \{i : \rho(i) \in \mathcal{S}\}$. Then, there exist constants $\{\omega_i \in Z_p\}_{i \in I}$ such that, if $\{\lambda_i\}$ are valid shares of any secret s according to Π, then $\sum_{i \in I} \omega_i \cdot \lambda_i = s$.

2.2 Bilinear Maps

Definition 3. *(Bilinear Maps)* Let $\mathbb{G}_0$ and $\mathbb{G}_1$ be two multiplicative cyclic groups of prime order p. Let g be a generator of $\mathbb{G}_0$ and e be a bilinear map $e: \mathbb{G}_0 \times \mathbb{G}_0 \to \mathbb{G}_1$. The bilinear map e has the following properties:

1) Bilinearity: for all $u, v \in \mathbb{G}_0$ and $a, b \in \mathbb{Z}_p$, we have $e(u^a, v^b) = e(u, v)^{ab}$.

2) Non-degeneracy: $e(g, g) \neq 1$.

3) Computable: there exists an efficient algorithm for $e: \mathbb{G}_0 \times \mathbb{G}_0 \to \mathbb{G}_1$.

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

2.3 Assumptions

In our paper, we adopt the q-type assumption of Rouselakis and Waters' scheme [20].

Assumption 1. $\boldsymbol{q}$**-type assumption**

Initially the challenger calls the group generation algorithm with input the security parameter, picks a random group element $g \in \mathbb{G}_0$, and $q+2$ random exponents $a, s, b_1, b_2, \dots, b_q \in \mathbb{Z}_p$. Then he sends to the adversary the group description $(p, \mathbb{G}_0, \mathbb{G}_1, e)$ and all of the following terms:

$$
\begin{array}{ll}
g, g^s & \\
g^{a^i}, g^{b_j}, g^{sb_j}, g^{a^i b_j}, g^{a^i b_j^2} & \forall (i,j) \in [q,q] \\
g^{a^i b_j / b_{j'}^2} & \forall (i,j,j') \in [2q,q,q]\ with\ j \neq j' \\
g^{a^i / b_j} & \forall (i,j) \in [2q,q]\ with\ i \neq q+1 \\
g^{sa^i b_j / b_{j'}}, g^{sa^i b_j / b_{j'}^2} & \forall (i,j,j') \in [q,q,q]\ with\ j \neq j'
\end{array}
$$

It is hard for the adversary to distinguish $e(g,g)^{sa^{q+1}} \in \mathbb{G}_1$ from an element which is randomly chosen from $\mathbb{G}_1$.

Definition 4. We say that the q-type assumption holds if no probabilistic polynomial time (PPT) adversary has a non-negligible advantage in solving the q-type problem.

Assumption 2. $\boldsymbol{l}$**-Strong Diffie-Hellman assumption** [5]

Given a $(l+1)$-tuple $g, g^x, g^{x^2}, \dots, g^{x^l}$ as input, it is hard for the adversary to output a pair $(d, g^{1/(x+d)})$ where $d \in \mathbb{Z}_p^*$.

Definition 5. We say that the l-SDH assumption holds if no PPT adversary has a non-negligible advantage in solving the l-SDH problem.

2.4 Miscellaneous Primitives

Zero-knowledge Proof of Knowledge of Discrete Log. A zero-knowledge proof[1] is a method by which one party (the prover) can prove to another party (the verifier) that a given statement is true, without conveying any information apart from the fact that the statement is indeed true. As a realistic cryptography application, a zero-knowledge proof of knowledge (ZK-POK) of discrete log protocol [9,22] enables a prover to prove to a verifier that it possesses the discrete log r of a given group element R in question.

3 CP-ABE with User and Authority Accountability

In this section we give the definition and security model of a large universe CP-ABE scheme with user and authority accountability (UaAA-CP-ABE).

[1] http://en.wikipedia.org/wiki/Zero-knowledge_proof

3.1 Definition

A UaAA-CP-ABE scheme consists of the following five algorithms:

Setup $(1^{\lambda}) \rightarrow (PK, MK)$: This is a randomized algorithm that takes a security parameter $\lambda \in \mathbb{N}$ encoded in unary. It outputs the public parameters PK and master key MK.

KeyGen $(PK, MK, ID, \mathcal{S}) \rightarrow SK$: This is a randomized algorithm that takes as input the public parameters PK, the master key MK, a user's identity ID and a set of attributes $\mathcal{S}$. It outputs this user's secret key SK.

Encrypt $(PK, M, \mathbb{A}) \rightarrow CT$: This is a randomized algorithm that takes as input the public parameters PK, a plaintext message M, and an access structure $\mathbb{A}$. It outputs the ciphertext CT.

Decrypt $(PK, SK, CT) \rightarrow M$: This algorithm takes as input the public parameters PK, a secret key SK for a user ID with a set of attributes $\mathcal{S}$, and a ciphertext CT encrypted under access structure $\mathbb{A}$. It outputs the message M if $\mathbb{A}(\mathcal{S}) = 1$.

Trace $(SK_{suspected}) \rightarrow$ (A user's ID or "KGC" or $\perp$) : This algorithm has two stages. In the first stage, it takes as input a decryption key $SK_{suspected}$ and outputs a user's identity ID with a key family number o or the special symbol $\perp$ if $SK_{suspected}$ is ill-formed. In the second stage, it compares the key family number o' of the secret key of the user ID with o. If $o' = o$, it outputs ID assuming the user ID is dishonest. Otherwise, it outputs "KGC". This definition of Trace is for the white-box setting.

3.2 Selective Security Model for UaAA-CP-ABE

In this part, we will define selective security for our UaAA-CP-ABE scheme. This is described by a game between an adversary $\mathcal{A}$ and a challenger $\mathcal{B}$ and is parameterized by the security parameter $\lambda \in \mathbb{N}$. The phases of the game are as follows.

Init: The adversary $\mathcal{A}$ declares the challenge access structure $\mathbb{A}^*$ which he wants to attack, and then sends it to the challenger $\mathcal{B}$.

Setup: The challenger $\mathcal{B}$ runs the Setup (1^{λ}) algorithm and gives the public parameters PK to the adversary $\mathcal{A}$.

Phase 1: The adversary $\mathcal{A}$ is allowed to issue queries for secret keys for users with sets of attributes $(ID_1, \mathcal{S}_1), (ID_2, \mathcal{S}_2), \ldots, (ID_{Q_1}, \mathcal{S}_{Q_1})$. For each $(ID_i, \mathcal{S}_i)$, the challenger $\mathcal{B}$ calls KeyGen $(PK, MK, ID_i, \mathcal{S}_i) \rightarrow SK_i$ and sends SK_i to $\mathcal{A}$. The only restriction is that $\mathcal{S}_i$ does not satisfy $\mathbb{A}^*$.

Challenge: The adversary $\mathcal{A}$ submits two equal length message M_0 and M_1. The challenger $\mathcal{B}$ flips a random coin $b \in \{0,1\}$, and encrypts M_b with $\mathbb{A}^*$. The ciphertext is passed to $\mathcal{A}$.

Phase 2: Phase 1 is repeated.

Guess: The adversary $\mathcal{A}$ outputs a guess b' of b.

The advantage of an adversary $\mathcal{A}$ in this game is defined as $|\Pr[b' = b] - 1/2|$.

Definition 6. A ciphertext-policy attribute-based encryption scheme with user and authority accountability is selectively secure if all PPT adversaries have at most negligible advantage in λ in the above security game.

3.3 Accountability Model for UaAA-CP-ABE

In this part, we will define three games for accountability, one for the dishonest KGC and two for the dishonest user.

a) The DishonestKGC Game

The intuition behind this game is that an adversarial KGC attempts to calculate user's key family number o in the user's secret key. The DishonestKGC Game for our scheme is defined as follows.

Setup: The adversary (acting as an adversarial KGC) runs the Setup (1^{λ}) algorithm and gives the public parameters PK and a user's identity ID to the challenger. The challenger checks that PK and ID are well-formed and aborts if the check fails.

Key Generation: The challenger chooses $o \in \mathbb{Z}_p$ randomly and sends w^o to the adversary. The challenger also need to give to the adversary a zero-knowledge proof of knowledge of the discrete log of w^o with respect to w. Then the adversary calls KeyGen $(PK, MK, ID, \mathcal{S}) \rightarrow SK$ and sends SK to the challenger. The challenger also check that SK is well-formed and aborts if the check fails.

Key Forgery: The adversary will output a decryption key SK' related with ID. The challenger checks that SK' is well-formed and aborts if the check fails.

Let KW denote the event that the adversary wins this game which happens the key family number of SK' equivalent to SK's. The advantage of an adversary in this game is defined as $Pr\ [KW]$.

b) The DishonestUser-1 Game

The intuition behind this game is that the adversary cannot create a new ID's secret key or even generate a new key SK'_{ID} with an existed ID appeared at Key Query stage. At Key Query stage, the adversary has already got SK_{ID}. In this game, a new key with an existed ID means that the identity-related information in SK_{ID} is successfully changed by the adversary. A tuple (ID, c) represents identity ID with the identity-related information. The DishonestUser-1 Game for our scheme is defined as follows.

Setup: The challenger runs the Setup (1^{λ}) algorithm and gives the public parameters PK to the adversary.

Key Query: The adversary issues queries for secret keys for users with sets of attributes $(ID_1, \mathcal{S}_1), (ID_2, \mathcal{S}_2), \dots, (ID_q, \mathcal{S}_q)$. The challenger responds to each query by calling KeyGen $(PK, MK, ID_i, \mathcal{S}_i) \rightarrow SK_i$.

Key Forgery: Eventually, the adversary outputs a decryption key SK related with (ID, c) and wins the game if

(1) (ID, c) is not any of $(ID_1, c_1), \dots, (ID_q, c_q)$, and

(2) SK is well-formed.

Let $UW1$ denote the event that the adversary wins this game. The advantage of an adversary in this game is defined as $Pr\ [UW1]$.

c) The DishonestUser-2 Game

As the same with DishonestKGC Game, we must assure a dishonest user cannot create another key family number (denoted by o) in that user's secret key. The DishonestUser-2 Game for our scheme is defined as follows.

Setup: The challenger runs the Setup (1^{λ}) algorithm and gives the public parameters PK to the adversary (acting as an adversarial user). The adversary checks that PK are well-formed and aborts if the check fails.

Key Query: The adversary issues queries for secret keys for users with sets of attributes $(ID_1, \mathcal{S}_1), (ID_2, \mathcal{S}_2), \ldots, (ID_q, \mathcal{S}_q)$. The challenger responds to each query by calling KeyGen $(PK, MK, ID_i, \mathcal{S}_i) \rightarrow SK_i$.

Key Forgery: The adversary will output a decryption key SK related with (ID, c, o) and wins the game if

(1) (ID, c) is one of $(ID_1, c_1), \ldots, (ID_q, c_q)$, we assume (ID, c) is equivalent to (ID_i, c_i), and

(2) o does not equal to o_i, and

(3) SK is well-formed.

Let $UW2$ denote the event that the adversary wins this game. The advantage of an adversary in this game is defined as $Pr\ [UW2]$.

Definition 7. A ciphertext-policy attribute-based encryption scheme with user and authority accountability is fully accountable if all PPT adversaries have negligible advantage in the above three security games.

4 Our Construction

Let $\mathbb{G}_0$ be a bilinear group of prime order p, and let g be a generator of $\mathbb{G}_0$. In addition, let $e: \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$ denote the bilinear map. A security parameter λ will determine the size of the groups. For the moment we assume that users' identity IDs and attributes are elements in $\mathbb{Z}_p^*$, however, IDs and attributes can be any meaningful unique strings using a collision resistant hash function $H: \{0,1\}^* \rightarrow \mathbb{Z}_p^*$.

Our construction follows.

Setup $(1^{\lambda}) \rightarrow (PK, MK)$: The algorithm calls the group generator algorithm $\mathcal{G}(1^{\lambda})$ and gets the descriptions of the groups and the bilinear mapping $D = (p, \mathbb{G}_0, \mathbb{G}_1, e)$. Then it picks the random terms $g, u, h, w, v \in \mathbb{G}_0$ and $\alpha, x, y \in \mathbb{Z}_p$. The published public parameters PK are

$$(D, g, u, h, w, v, X = g^x, Y = g^y, e(g, g)^{\alpha}).$$

The master key MK are (α, x, y).

KeyGen $(PK, MK, ID, \mathcal{S} = \{A_1, A_2, \ldots, A_k\} \subseteq \mathbb{Z}_p) \rightarrow SK$: After the user ID is authenticated, the KGC gets w^o from ID where ID chooses $o \in \mathbb{Z}_p$ randomly. ID also needs to give to KGC a zero-knowledge proof of knowledge of the discrete log (as in Section 2.5) of w^o with respect to w. Then it picks $k+2$ random nents $c, r, r_1, r_2, \dots, r_k \in \mathbb{Z}_p$. It outputs this user's secret key SK (Notice that $N_3 = o$ is owned by the user secretly, and is part of SK):

$$\mathcal{S}, K_1 = g^{\alpha/(x+ID+yc)} w^{o\cdot r}, N_1 = ID, N_2 = c, N_3 = o, L_1 = g^r, L_2 = g^{xr}, L_3 = g^{yr},$$
$$\{K_{i,2} = g^{r_i}, K_{i,3} = (u^{A_i}h)^{r_i} v^{-(x+ID+yc)r}\}_{i\in[k]}.$$

Here $1/(x+ID+yc)$ is computed modulo p. In the unlikely event that $x+ID+yc=0$ we will pick another random c.

Encrypt $(PK, m, (M,\rho)) \rightarrow CT$: To encrypt a message $m \in \mathbb{G}_1$ under an access structure encoded in an LSSS policy (M,ρ). Let the dimensions of M be $l \times n$. Each row of M will be labeled by an attribute and $\rho(i)$ denotes the label of i^{th} row $\vec{M_i}$. Choose a random vector $\vec{z} = (s, z_2, \dots, z_n)^T$ from $\mathbb{Z}_p^n$ where s is the random secret to be shared among the shares. The vector of the shares is $\vec{\lambda} = (\lambda_1, \lambda_2, \dots, \lambda_l)^T = M\vec{z}$. It then chooses l random value $t_1, t_2, \dots, t_l \in Z_p$ and publishes the ciphertext as:

$$CT = ((M,\rho), C = me(g,g)^{\alpha s}, D_1 = g^s, D_2 = g^{xs}, D_3 = g^{ys},$$
$$\{C_{i,1} = w^{\lambda_i} v^{t_i}, C_{i,2} = \left(u^{\rho(i)}h\right)^{-t_i}, C_{i,3} = g^{t_i}\}_{i\in[l]}).$$

Decrypt $(PK, SK, CT) \rightarrow m$: To decrypt the ciphertext CT with the decryption key SK, proceed as follows. Suppose that $\mathcal{S}$ satisfies the access structure and let $I = \{i : \rho(i) \in \mathcal{S}\}$. Since the set of attributes satisfy the access structure, there exist coefficients $\omega_i \in \mathbb{Z}_p$ such that $\sum_{\rho(i)\in I} \omega_i \cdot \vec{M_i} = (1,0,\dots,0)$. Then we have that $\sum_{\rho(i)\in I} \omega_i \lambda_i = s$. Now it calculates

$$E = e(K_1, D_1^{ID} D_2 D_3^c) = e(g,g)^{\alpha s} e(g,w)^{(x+ID+yc)rso}.$$
$$F = \prod_{i\in I} (e\left(L_1^{ID} L_2 L_3^c, C_{i,1}\right) e(K_{i,2}, C_{i,2}) e(K_{i,3}, C_{i,3}))^{\omega_i} = e(g,w)^{(x+ID+yc)rs}.$$
$$m = CF^o/E.$$

Trace $(SK_{suspected}) \rightarrow$ (A user's ID or "KGC" or $\perp$) : If $SK_{suspected}$ is ill-formed, the algorithm will output the special symbol $\perp$. Otherwise, it outputs $N_1 = ID$ and key family number $N_3 = o$ in $SK_{suspected}$. If ID does not exist, the algorithm outputs "KGC" which means the dishonest KGC create a fake user's identity. Otherwise, it compares o with the key family number o_{ID} of the secret key of a real user ID. If $o = o_{ID}$, it outputs ID assuming the user ID is dishonest. Otherwise, it outputs "KGC". Notice that we do not need to compare the signature part $N_2 = c$ in these two keys, because key family number N_3 is enough to distinguish dishonest user or KGC.

5 Analysis of Our Proposed Scheme

5.1 Selective Security Proof

In our original scheme, the KGC does not have complete control over *SK* because it does not know o in w^o. For this reason, the scheme is difficult to be proved selectively secure. A similar situation occurs in accountable authority identity-based encryption (A-IBE) scheme [9]. In the part of security proof of A-IBE, the simulator uses a knowledge extractor to extract the discrete log. In our proof, we will use the same technology and assume that the simulator knows o.

In the selective security proof, we will reduce the selective security of our CP-ABE scheme to that of Rouselakis and Waters' [20] which is proved selectively secure under Assumption 1.

Theorem 1. If Rouselakis and Waters' scheme [20] is selectively secure, then all PPT adversaries with a challenge matrix of size $l \times n$, where $l, n \leq q$, have a negligible advantage in selectively breaking our scheme.

Proof. To prove the theorem we will suppose that there exists a PPT adversary $\mathcal{A}$ with a challenge matrix that satisfies the restriction, which has a non-negligible advantage $Adv_{\mathcal{A}}$ in selectively breaking our scheme. Using this adversary we will build a PPT simulator $\mathcal{B}$ that attacks Rouselakis and Waters' scheme (Sim_{RW}) [20] with a non-negligible advantage.

Init: The adversary $\mathcal{A}$ declares a challenge access policy $\mathbb{A}^* = (M^*, \rho^*)$ which he wants to attack, and then sends it to the challenger $\mathcal{B}$. $\mathcal{B}$ sends this received challenge access policy to Sim_{RW}. Notice that M^* is a $l \times n$ matrix, where $l, n \leq q$. Each row of M^* will be labeled by an attribute and $\rho^*(i)$ denotes the label of i^{th} row of M^*.

Setup: $\mathcal{B}$ gets the public parameters $PK_{RW} = (D, g, u, h, w, v, e(g,g)^{\alpha})$ from Sim_{RW}. Then $\mathcal{B}$ chooses $x, y \in \mathbb{Z}_p$ randomly, and gives the public parameters PK= $(D, g, u, h, w, v, g^x, g^y, e(g,g)^{\alpha})$ to $\mathcal{A}$. Notice that this way α is information-theoretically hidden from $\mathcal{B}$.

Phase 1: Now $\mathcal{B}$ has to produce secret keys for tuples which consists of non-authorized sets of attributes $\mathcal{S} = \{A_1, A_2, \ldots, A_k\}$, a user's identity ID, and an element w^o computed with a zero-knowledge proof. The only restriction is that $\mathcal{S}$ does not satisfy $\mathbb{A}^*$. As analysis in the beginning part of this section, we assume $\mathcal{B}$ knows o. At first, $\mathcal{B}$ will issue $\mathcal{S}$ to Sim_{RW} and get the corresponding decryption key as follows:

$$\mathcal{S}, \widetilde{K_1} = g^{\alpha} w^{\tilde{r}}, \widetilde{L_1} = g^{\tilde{r}}, \{\widetilde{K}_{i,2} = g^{\tilde{r}_i}, \widetilde{K}_{i,3} = (u^{A_i}h)^{\tilde{r}_i} v^{-\tilde{r}}\}_{i \in [k]}.$$

Then $\mathcal{B}$ picks random exponents $c \in \mathbb{Z}_p$, and sets $r = \tilde{r}/((x + ID + yc) \cdot o)$ and $\{r_i = \tilde{r}_i/o\}_{i \in [k]}$ implicitly. Here $1/(x + ID + yc)$ is computed modulo p. In the unlikely event that $x + ID + yc = 0$, $\mathcal{B}$ will pick another random c. Then $\mathcal{B}$ computes

$$K_1 = \widetilde{K_1}^{1/(x+ID+yc)} = g^{\alpha/(x+ID+yc)} w^{\tilde{r}/(x+ID+yc)} = g^{\alpha/(x+ID+yc)} w^{o \cdot r}.$$

$$L_1 = \widetilde{L_1}^{1/((x+ID+yc)\cdot o)} = g^{\tilde{r}/((x+ID+yc)\cdot o)} = g^r, L_2 = L_1^x = g^{xr}, L_3 = L_1^y = g^{yr}.$$

$$\{K_{i,2} = (\widetilde{K}_{i,2})^{1/o} = g^{\tilde{r}_i/o} = g^{r_i}\}_{i \in [k]}.$$

$$\{K_{i,3} = (\widetilde{K}_{i,3})^{1/o} = (u^{A_i}h)^{\tilde{r}_i/o} v^{-\tilde{r}/o} = (u^{A_i}h)^{r_i} v^{-(x+ID+yc)r}\}_{i \in [k]}.$$

Finally, $\mathcal{B}$ sends the decryption key $SK = (\mathcal{S}, K_1, N_1 = ID, N_2 = c, L_1, L_2, L_3,$ $\{K_{i,2}, K_{i,3}\}_{i \in [k]})$ to $\mathcal{A}$. Notice that $N_3 = o$ is owned by $\mathcal{A}$.

Challenge: The adversary $\mathcal{A}$ submits two equal length message m_0 and m_1. Then $\mathcal{B}$ submits m_0 and m_1 to Sim_{RW}, and gets the challenge ciphertext as follows:

$$((M^*, \rho^*), C, D_1 = g^s, \{C_{i,1} = w^{\lambda_i} v^{t_i}, C_{i,2} = \left(u^{\rho(i)}h\right)^{-t_i}, C_{i,3} = g^{t_i}\}_{i \in [l]}).$$

Notice that C has two forms indeed according to the proof part of Rouselakis and Waters' scheme [20], one is well-formed $(m_b e(g,g)^{\alpha s})$, and the other is random.

Then $\mathcal{B}$ computes $D_2 = D_1^x = g^{xs}, D_3 = D_1^y = g^{ys}$. Finally, $\mathcal{B}$ sends the challenge ciphertext $CT = ((M^*, \rho^*), C, D_1, D_2, D_3, \{C_{i,1}, C_{i,2}, C_{i,3}\}_{i\in[l]})$ to $\mathcal{A}$.

Phase 2: Phase 1 is repeated.

Guess: The adversary $\mathcal{A}$ outputs a guess b' of b to $\mathcal{B}$. Then $\mathcal{B}$ sends b' to Sim_{RW}.

Since the distributions of the public parameters, secret keys and ciphertexts of our scheme and Rouselakis and Waters' in the above game are the same, the adversary in selectively breaking Rouselakis and Waters' scheme has the same advantage as adversary $\mathcal{A}$ in selectively breaking our scheme. As Rouselakis and Waters' scheme is selectively secure, so do ours. □

5.2 Accountability Proof

a) Analysis of the DishonestKGC Game

Theorem 2. Assuming that computing discrete logarithm is hard in $\mathbb{G}_0$, the advantage of an adversary in the DishonestKGC Game is negligible for our scheme.

Proof. To prove the theorem we will suppose that there exists a PPT adversary $\mathcal{A}$ which has a non-negligible advantage $Adv_{\mathcal{A}}$ in the DishonestKGC Game in our scheme. Using this adversary we will build a PPT simulator $\mathcal{B}$ that attacks the discrete logarithm problem with a non-negligible advantage. $\mathcal{B}$ proceeds as follows.

Setup: The adversary $\mathcal{A}$ (acting as an adversarial KGC) runs the Setup (1^λ) algorithm and gives the public parameters PK= $(D, g, u, h, w, v, g^x, g^y, e(g,g)^\alpha)$ and a user's identity ID to the simulator $\mathcal{B}$. $\mathcal{B}$ checks that PK and ID are well-formed and aborts if the check fails.

Key Generation: $\mathcal{B}$ invokes the challenger $\mathcal{C}$, passes on w to it and gets a challenge $W = w^o \in \mathbb{G}_0$. Then $\mathcal{B}$ engages in the key generation protocol with $\mathcal{A}$ to get a decryption key for ID as follows. Notice that $\mathcal{B}$ should give to $\mathcal{A}$ a zero-knowledge proof of knowledge of the discrete log of w^o with respect to w, however, $\mathcal{B}$ does not know o. A similar situation occurs in A-IBE [9]. In the part of security proof of the FindKey game in A-IBE, $\mathcal{B}$ simulates the required proof without knowledge of o. In our proof, we will use the same technology and assume that $\mathcal{B}$ successfully gives to $\mathcal{A}$ a zero-knowledge proof of knowledge. Then $\mathcal{A}$ calls KeyGen $(PK, MK, ID, \mathcal{S}) \to SK$ and sends SK to $\mathcal{B}$.

Key Forgery: $\mathcal{A}$ will output a decryption key $SK' = (\mathcal{S}, K_1, N_1 = ID, N_2 = c, N_3 = o', L_1, L_2, L_3, \{K_{i,2}, K_{i,3}\}_{i\in[k]})$ related with ID. $\mathcal{B}$ checks that SK' is well-formed and aborts if the check fails. If SK' is well-formed, $\mathcal{B}$ sends o' to $\mathcal{C}$.

If $Adv_{\mathcal{A}}$ in the DishonestKGC Game is non-negligible, we have built a PPT simulator $\mathcal{B}$ that attacks the discrete logarithm problem with a non-negligible advantage. Since computing discrete logarithm is believed to be difficult, there does not exist a PPT adversary $\mathcal{A}$ which has a non-negligible advantage $Adv_{\mathcal{A}}$ in the DishonestKGC Game in our scheme. □

b) Analysis of the DishonestUser-1 Game

Theorem 3. The advantage of an adversary in the DishonestUser-1 Game is negligible for our CP-ABE scheme under the l-SDH assumption.

Proof. To prove the theorem we will suppose that there exists a PPT adversary $\mathcal{A}$ which has a non-negligible advantage $Adv_{\mathcal{A}}$ in the DishonestUser-1 Game in our scheme (the probability that $\mathcal{A}$ wins the game is at least ϵ). Using this adversary we will show how to build a PPT simulator $\mathcal{B}$ that is able to solve the l-SDH assumption with a non-negligible advantage.

We first give some intuition for the proof. Assuming $\mathcal{A}$ issues q queries, For each secret key, we record a tuple $(ID_i, c_i, d_i = ID_i + yc_i)$. At Key Forgery stage, the adversary outputs a decryption key SK related with $(ID^*, c^*, d^* = ID^* + yc^*)$. There are two possibilities when the adversary wins the game, $d^* \in \{d_i\}_{i\in[q]}$ or $d^* \notin \{d_i\}_{i\in[q]}$. We distinguish between two types of adversaries.

Type-1 adversary: an adversary that either

1) makes a secret key query for user's identity $ID = -x$ at Key Query stage, or

2) outputs a decryption key SK related with $d^* \notin \{d_i\}_{i\in[q]}$ at Key Forgery stage.

Type-2 adversary: an adversary that both

1) never makes a secret key query for user's identity $ID = -x$ at Key Query stage, and

2) outputs a decryption key SK related with $d^* \in \{d_i\}_{i\in[q]}$ at Key Forgery stage.

We will show that either adversary can be used to solve the l-SDH assumption. However, the simulator $\mathcal{B}$ works differently for each adversary type. Thus, $\mathcal{B}$ will choose a random bit $b_{mode} \in \{1,2\}$ that indicates its guess for the type of adversary that $\mathcal{A}$ will emulate.

$\mathcal{B}$ is given a bilinear mapping $D = (p, \mathbb{G}_0, \mathbb{G}_1, e)$ and a random instance $(A_0 = g', A_1 = (g')^x, A_2 = (g')^{x^2}, \ldots, A_l = (g')^{x^l}) \in \mathbb{G}_0^{l+1}$ of the l-SDH problem for some unknown $x \in \mathbb{Z}_p^*$. Then $\mathcal{B}$ proceeds as follows.

Setup: $\mathcal{B}$ chooses $q = l - 1$ random elements $d_1, d_2, \ldots, d_q \in \mathbb{Z}_p^*$. Let $f(z)$ be the polynomial $f(z) = \prod_{i=1}^{l-1}(z + d_i)$. Expand $f(z)$ and write $f(z) = \sum_{i=0}^{l-1}\eta_i z^i$ where $\eta_0, \eta_1, \ldots, \eta_{l-1} \in \mathbb{Z}_p$ are the coefficients of the polynomial $f(z)$. Compute:

$$\boldsymbol{g} \leftarrow \prod^{l-1} A_i^{\eta_i} = (\boldsymbol{g}')^{f(x)} \textbf{ and } \boldsymbol{Z} \leftarrow \prod_{i=1}^{l} A_i^{\eta_{i-1}} = (\boldsymbol{g}')^{xf(x)} = \boldsymbol{g}^x.$$

Notice that we may assume that $\boldsymbol{f(x) \neq 0}$, otherwise, $\boldsymbol{x = -d_i}$ for some i which means that $\mathcal{B}$ just obtains the secret key x of the l-SDH problem.

Then $\mathcal{B}$ picks the random terms $u, h \in \mathbb{G}_0$, $\alpha, \mu, \pi \in \mathbb{Z}_p$ and

If $b_{mode} = 1$, $\mathcal{B}$ picks a random $y \in \mathbb{Z}_p^*$ and gives $\mathcal{A}$ the public parameters $PK_1 = (D, g, u, h, w = g^\mu, v = g^\pi, X = Z = \boldsymbol{g}^x, Y = g^y, e(g,g)^\alpha)$.

If $b_{mode} = 2$, $\mathcal{B}$ picks a random $x' \in \mathbb{Z}_p^*$ and gives $\mathcal{A}$ the public parameters $PK_2 = (D, g, u, h, w = g^\mu, v = g^\pi, X = g^{x'}, Y = Z = \boldsymbol{g}^x, e(g,g)^\alpha)$.

Notice that in either case, $\mathcal{B}$ provides the adversary $\mathcal{A}$ with a valid public parameters.

Key Query: The adversary $\mathcal{A}$ can issue up to q queries for secret keys adaptively. In order to respond, $\mathcal{B}$ maintains a list H-list of tuples (ID_i, c_i, W_i). Then for the ith query $(ID_i, \mathcal{S}_i)$:

Let $f_i(z)$ be the polynomial $f_i(z) = f(z)/(z+d_i) = \prod_{j=1,j\neq i}^{l-1}(z+d_i)$. Expand $f_i(z)$ and write $f_i(z) = \sum_{j=0}^{l-2}\beta_j z^j$ where $\beta_0, \beta_1, \dots, \beta_{l-2} \in \mathbb{Z}_p$ are the coefficients of the polynomial $f_i(z)$. Compute

$$\sigma_i \leftarrow \prod_{j=0}^{l-2} \boldsymbol{A}_j^{\beta_j} = (\boldsymbol{g}')^{f_i(\boldsymbol{x})} = \boldsymbol{g}^{\mathbf{1}/(\boldsymbol{x}+\boldsymbol{d}_i)}.$$

If $b_{mode} = 1$, check if $g^{-ID} = X$. If so, $\mathcal{B}$ just obtains the secret key x of the l-SDH problem which allows it to compute $(d, g^{1/(x+d)})$ for any d easily. At this point $\mathcal{B}$ successfully solves the l-SDH assumption.

Otherwise, $\mathcal{B}$ sets $c_i = (d_i - ID_i)/y \in \mathbb{Z}_p^*$. If $c_i = 0$, $\mathcal{B}$ reports failure and aborts. Otherwise, it picks $k+1$ random exponents $r, r_1, r_2, \dots, r_k \in \mathbb{Z}_p$ and outputs ID_i's secret key SK_i (o is owned by the adversary secretly, and is part of SK_i)

$$\mathcal{S}, K_1 = \sigma_i^{\alpha} w^{o\cdot r} = g^{\alpha/(x+ID_i+yc_i)} w^{o\cdot r}, N_1 = ID_i, N_2 = c_i, N_3 = o,$$
$$L_1 = g^r, L_2 = X^r = g^{xr}, L_3 = Y^r = g^{yr},$$
$$\{K_{i,2} = g^{r_i}, K_{i,3} = (u^{A_i}h)^{r_i} X^{-\pi r} v^{-(ID+yc)r} = (u^{A_i}h)^{r_i} v^{-(x+ID_i+yc_i)r}\}_{i\in[k]}.$$

Apparently, this is a valid user's secret key.

If $b_{mode} = 2$, $\mathcal{B}$ sets $c_i = (x' + ID_i)/d_i \in \mathbb{Z}_p^*$. If $c_i = 0$, $\mathcal{B}$ reports failure and aborts. Otherwise, it picks $k+1$ random exponents $r, r_1, r_2, \dots, r_k \in \mathbb{Z}_p$ and outputs ID_i's secret key SK_i (o is owned by the adversary secretly, and is part of SK_i)

$$\mathcal{S}, K_1 = \sigma_i^{\alpha/c_i} w^{o\cdot r} = g^{\alpha/(x'+ID_i+xc_i)} w^{o\cdot r}, N_1 = ID_i, N_2 = c_i, N_3 = o,$$
$$L_1 = g^r, L_2 = X^r = g^{x'r}, L_3 = Y^r = g^{xr},$$
$$\{K_{i,2} = g^{r_i}, K_{i,3} = (u^{A_i}h)^{r_i} v^{-(x'+ID_i)r} Y^{-\pi c_i r} = (u^{A_i}h)^{r_i} v^{-(x'+ID_i+xc_i)r}\}_{i\in[k]}.$$

Apparently, this is a valid user's secret key, too.

In either case $\mathcal{B}$ adds the tuple $(ID_i, c_i, W_i = g^{ID_i}Y^{c_i})$ to the H-list.

Key Forgery: Eventually, the adversary outputs a decryption key SK related with (ID^*, c^*) where SK is well-formed and (ID^*, c^*) is not any of $(ID_1, r_1), \dots, (ID_q, r_q)$. Notice that by adding dummy queries as necessary, we may assume that the adversary made exactly $l-1$ queries. Let $W^* = g^{ID^*}Y^{c^*}$. Then $\mathcal{B}$ searches W^* from the H-list. There are two possibilities:

Type-1 adversary: No tuple of the form $(\cdot,\cdot, W^*)$ appears on the H-list.

Type-2 adversary: The H-list contains at least one tuple (ID_j, c_j, W_j) such that $W_j = W^*$.

Let $B_{type} = 1$ if $\mathcal{A}$ produced a type-1 adversary. Otherwise, set $B_{type} = 2$. If $b_{mode} \neq B_{type}$, $\mathcal{B}$ reports failure and aborts.

If $b_{mode} = B_{type} = 1$, check if $g^{-ID} = X$. If so, $\mathcal{B}$ can solve the l-SDH assumption successfully. Otherwise, compute

$$\sigma^* = (K_1 L_1^{-\mu N_3})^{1/\alpha} = g^{1/(x+ID^*+yc^*)} = (\boldsymbol{g}')^{f(x)/(x+ID^*+yc^*)}.$$

Let $d^* = ID^* + yc^*$. Notice that $d^* \notin \{d_i\}_{i\in[l-1]}$ when adversary is type-1.

Using long division we write the polynomial f as $f(z) = \gamma(z)(z + d^*) + \gamma_{-1}$ for some polynomial $\gamma(z) = \sum_{i=0}^{l-2} \gamma_i z^i$ and $\gamma_{-1} \in \mathbb{Z}_p$. Then $f(z)/(z + d^*) = \gamma_{-1}/(z + d^*) + \sum_{i=0}^{l-2} \gamma_i z^i$ and hence

$$\sigma^* = (\boldsymbol{g}')^{\gamma_{-1}/(x+d^*) + \sum_{i=0}^{l-2} \gamma_i x^i}.$$

Notice that $\gamma_{-1} \neq 0$, since $f(z) = \prod_{i=1}^{l-1}(z + d_i)$ and $d^* \notin \{d_i\}_{i\in[l-1]}$. Then $\mathcal{B}$ computes

$$(\sigma^* \cdot \prod_{i=0}^{l-2} A_i^{-\gamma_i})^{1/\gamma_{-1}} = ((\boldsymbol{g}')^{\gamma_{-1}/(x+d^*)} \cdot (\boldsymbol{g}')^{\sum_{i=0}^{l-2} \gamma_i x^i} \cdot \prod_{i=0}^{l-2} (\boldsymbol{g}')^{-\gamma_i x^i})^{1/\gamma_{-1}}$$
$$= (\boldsymbol{g}')^{1/x+d^*}.$$

and returns $(d^*, (\boldsymbol{g}')^{1/x+d^*})$ as the solution to the l-SDH problem.

If $b_{mode} = B_{type} = 2$, let (ID_j, c_j, W_j) be a tuple on the H-list where $W_j = W^*$. Since $Y = g^x$, we know that $g^{ID_j} g^{xc_j} = g^{ID^*} g^{xc^*} \Rightarrow ID_j + xc_j = ID^* + xc^*$. We know that $(ID_j, c_j) \neq (ID^*, c^*)$, otherwise, the adversary failed to forge a secret key SK and would lose the game. Therefore, $x = (ID^* - ID_j)/(c_j - c^*) \in \mathbb{Z}_p^*$. As $\mathcal{B}$ knows x, $\mathcal{B}$ can solve the l-SDH assumption successfully.

Now we complete the description of simulator $\mathcal{B}$. Notice that,

1) the view from $\mathcal{A}$ is independent of the choice of b_{mode},

2) the public parameters are uniformly distributed, and

3) the secret keys that $\mathcal{A}$ queries are well-formed.

Therefore, $\mathcal{A}$ produces a valid secret key with probability at least ϵ.

It remains to bound the probability that $\mathcal{B}$ does not abort. We argue as follows:

If $b_{mode} = B_{type} = 1$, $\mathcal{B}$ aborts when $\mathcal{A}$ forged a secret key with $d^* \in \{d_i\}_{i\in[l-1]}$. This happens with probability at most $(l-1)/p$.

If $b_{mode} = B_{type} = 2$, $\mathcal{B}$ does not abort.

Since b_{mode} is independent of B_{type} we have that $\Pr[b_{mode} = B_{type}] = 1/2$. It now follows that $\mathcal{B}$ produces a valid tuple $(d, (g')^{1/(x+d)})$ with probability

$$\Pr[\mathcal{B} \text{ not abort \&\& win} | b_{mode} = B_{type} = 1] \cdot \Pr[b_{mode} = B_{type} = 1] +$$
$$\Pr[\mathcal{B} \text{ not abort \&\& } win | b_{mode} = B_{type} = 2] \cdot \Pr[b_{mode} = B_{type} = 2]$$
$$= \epsilon \cdot (1 - (l-1)/p) \cdot 1/4 + \epsilon \cdot 1/4 = \epsilon/2 - (l-1) \cdot \epsilon/(4p) \approx \epsilon/2. \qquad \square$$

c) Analysis of the DishonestUser-2 Game

Theorem 4. Assuming that computing discrete logarithm is hard in $\mathbb{G}_0$, the advantage of an adversary in the DishonestUser-2 Game is negligible for our scheme.

Proof. To prove the theorem we will suppose that there exists a PPT adversary $\mathcal{A}$ which has a non-negligible advantage $Adv_{\mathcal{A}}$ in the DishonestUser-2 Game in our scheme. Using this adversary we will build a PPT simulator $\mathcal{B}$ that attacks the discrete logarithm problem (g, g^z) with a non-negligible advantage. $\mathcal{B}$ proceeds as follows.

Setup: $\mathcal{B}$ runs the Setup (1^λ) algorithm and gives the public parameters PK to the adversary $\mathcal{A}$. Notice that the generation of w, v is different from the original Setup. $\mathcal{B}$ picks the random terms $\omega, \mu \in \mathbb{Z}_p$ and calculates $w = g^\omega, v = g^\mu$. However, in $\mathcal{A}$'s view, they are identical.

Key Query: The adversary $\mathcal{A}$ issues queries for secret keys for users with sets of attributes $(ID_1, \mathcal{S}_1), (ID_2, \mathcal{S}_2), \dots, (ID_q, \mathcal{S}_q)$. As space limited, we only give the different parts from the original KeyGen here. For query i, when $\mathcal{A}$ gives $\mathcal{B}$ a zero-knowledge proof of knowledge of the discrete log of w^{o_i} with respect to w, $\mathcal{B}$ will use a knowledge extractor [9] to extract the discrete log o_i. Then $\mathcal{B}$ chooses $\gamma_i \in \mathbb{Z}_p$ and implicitly sets $r^{(i)} = \gamma_i \cdot z$. $\mathcal{B}$ can calculate $w^{o_i \cdot r^{(i)}}$ by $(g^z)^{\omega \cdot o_i \cdot \gamma_i}$. $\mathcal{B}$ can use the same method to calculate $g^{r^{(i)}}, g^{xr^{(i)}}, g^{yr^{(i)}}, v^{-(x+ID+yc)r^{(i)}}$ even if $\mathcal{B}$ does not know z. Other parts in the secret key will follow the same method in KeyGen. Finally, $\mathcal{B}$ sends SK_i to $\mathcal{A}$.

Key Forgery: The adversary $\mathcal{A}$ outputs a decryption key SK related with (ID, c, o). We assume (ID, c) is equivalent to (ID_i, c_i) and o does not equal to o_i. In this case, $\mathcal{A}$ generates a new secret key successfully.

Now we will analyze the security of the discrete logarithm problem. Let's review the user's secret key firstly. For simplicity, we omit i and (i) in o_i and $r^{(i)}$:

$$\mathcal{S}, K_1 = g^{\alpha/(x+ID+yc)} w^{o \cdot r}, N_1 = ID, N_2 = c, N_3 = o,$$

$$L_1 = g^r, L_2 = g^{xr}, L_3 = g^{yr}, \{K_{i,2} = g^{r_i}, K_{i,3} = (u^{A_i} h)^{r_i} v^{-(x+ID+yc)r}\}_{i \in [k]}.$$

And the adversary $\mathcal{A}$ outputs a forged secret key SK' where $o' \neq o$:

$$\mathcal{S}', K_1' = g^{\alpha/(x+ID+yc)} w^{o' \cdot r'}, N_1' = ID, N_2' = c, N_3' = o',$$

$$L_1' = g^{r'}, L_2' = g^{xr'}, L_3' = g^{yr'}, \{K_{i,2}' = g^{r_i'}, K_{i,3}' = (u^{A_i} h)^{r_i'} v^{-(x+ID+yc)r'}\}_{i \in [k']}.$$

Firstly, we will analyze K_1 and K_1'. As $\alpha/(x + ID + yc)$ and r in K_1 is information-theoretically hidden from $\mathcal{A}$. If $\mathcal{A}$ can forge K_1' successfully, then we can assume that $K_1' = K_1 \cdot w^{f_1} \Rightarrow o \cdot r + f_1 = o' \cdot r'$. Similarly, since $x + ID + yc$ in $K_{i,3}$ is information-theoretically hidden from $\mathcal{A}$, if $\mathcal{A}$ can forge $K_{i,3}'$ successfully, we can assume that $K_{i,3}' = (K_{i,3})^{f_2} \Rightarrow r \cdot f_2 = r'$. Then we get two equations:

$$\begin{cases} o \cdot r + f_1 = o' \cdot r' \\ \quad r \cdot f_2 = r' \end{cases}.$$

From $\mathcal{A}$'s view, $\mathcal{A}$ knows o, o', f_1, f_2. If $o' \cdot f_2 \neq o$, then $r = f_1/(o' \cdot f_2 - o)$. Apparently, the probability of $o' \cdot f_2 = o$ is negligible. Then $\mathcal{A}$ can compute r. As r equals to $\gamma_i \cdot z$, then $z = r/\gamma_i$. Therefore, if $\mathcal{A}$ forges a secret key SK' where $o' \neq o$, we can conclude that $\mathcal{A}$ have solved the discrete logarithm problem. However, as we assumed that computing discrete logarithm is hard in $\mathbb{G}_0$, then $\mathcal{A}$ cannot forge a secret key SK' where $o' \neq o$. Therefore, the advantage of an adversary in the DishonestUser-2 Game is negligible for our scheme. □

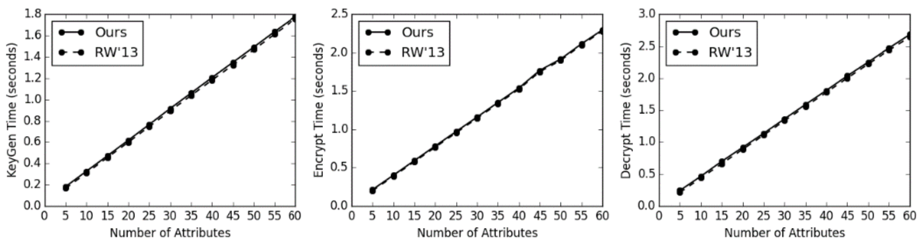
5.3 Performance Analysis

There are two aspects to consider for performance analysis, the performance of normal functions and the capability of the accountability. As for accountability, the advantage of our scheme is obvious and we have explained it in Section 1.3. Therefore, we mainly focus on the performance of normal functions in this section. We compared our scheme with Rouselakis and Waters (RW'13) [20] as ours is based on RW'13. We wanted to know how much computational efficiency to lose for security enhancements of RW'13. We implemented both schemes in Charm[2] [1]. We use

[2] You can download our codes from https://github.com/zlwen/charm-example.

"SS512" elliptic curve group. All our tests were executed on a Intel(R) Core(TM) i7-3770 CPU (3.40GHz) with 8.0GB RAM running Windows 8.1 Pro and Python 3.4.3 and sampled 20 times.

As the Setup stage is stable, we do not show the time in the figure. Ours spend 71.5 milliseconds and 56 milliseconds for RW'13. These are very small values. Fig. 1 shows the computation cost in KeyGen, Encrypt, and Decrypt under various conditions. In the Setup stage, attribute number of users starts from 5 to 60 and increases 5 every time. In the Encrypt stage, attribute number of ciphertext policies starts from 5 to 60 and increases 5 every time. They are connected by the AND gate. As can be seen from the figure, the time is very close to each experiment. We find that the differences of Encrypt and Decrypt time are nearly constant in these test cases. The Encrypt time of our scheme is 0.015s bigger than RW'13 [20] and the Decryption time is 0.032s. The difference of KeyGen time between our scheme and RW'13 [20] grows slowly from 0.016s to 0.021s. Therefore, our scheme is very efficient. Notice that the encryption time of our scheme in the figure is only for KGC. Users also need to give to KGC a zero-knowledge proof of knowledge of the discrete log of w^o with respect to w.



(a) KeyGen (b) Encrypt (c) Decrypt

Fig. 1. Comparison of KeyGen, Encrypt and Decrypt

6 Conclusion

The lack of user and authority accountability is an important challenging issue in ABE. The user is able to share his/her secret key and abuse his/her access privilege without being identified, and KGC can generate any user's secret key. In this paper, we propose a practical large universe CP-ABE with user and authority accountability. We can trace the dishonest user or KGC in white-box model. We prove our scheme selectively secure in the standard model under q-type assumption. We also prove the accountability property against dishonest user and KGC in the standard model. In the future work, we intend to construct a scheme which can support user and authority accountability in black-box model. And another future research direction is how to revoke the dishonest user after the user is found.

Acknowledgments. This work is supported by the National High Technology Research and Development Program ("863" Program) of China under Grant No. 2015AA016009, the National Natural Science Foundation of China under Grant No. 61232005, and the Science and Technology Program of Shen Zhen, China under Grant No. JSGG2014051 6162852628.

References

1. Akinyele, J.A., Garman, C., Miers, I., Pagano, M.W., Rushanan, M., Green, M., Rubin, A.D.: Charm: a framework for rapidly prototyping cryptosystems. Journal of Cryptographic Engineering **3**(2), 111–128 (2013)
2. Attrapadung, N., Libert, B., de Panafieu, E.: Expressive key-policy attribute-based encryption with constant-size ciphertexts. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC 2011. LNCS, vol. 6571, pp. 90–108. Springer, Heidelberg (2011)
3. Beimel, A.: Secure schemes for secret sharing and key distribution. Ph.D. thesis, Israel Institute of Technology, Technion, Haifa, Israel (1996)
4. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: IEEE Symposium on Security and Privacy, pp. 321–334 (2007)
5. Boneh, D., Boyen, X.: Short signatures without random oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 56–73. Springer, Heidelberg (2004)
6. Chase, M.: Multi-authority Attribute Based Encryption. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 515–534. Springer, Heidelberg (2007)
7. Chase, M., Chow, S.S.: Improving privacy and security in multi-authority attribute-based encryption. In: ACM Conference on Computer and Communications Security, pp. 121–130 (2009)
8. Cheung, L., Newport, C.: Provably secure ciphertext policy ABE. In: ACM Conference on Computer and Communications Security, pp. 456–465 (2007)
9. Goyal, V.: Reducing trust in the PKG in identity based cryptosystems. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 430–447. Springer, Heidelberg (2007)
10. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: ACM Conference on Computer and Communications Security, pp. 89–98 (2006)
11. Hinek, M.J., Jiang S., Safavi-Naini, R., Shahandashti, S.F.: Attribute-based encryption with key cloning protection. Cryptology ePrint Archive, Report 2008/478 (2008). http://eprint.iacr.org/
12. Li, J., Huang, Q., Chen, X., Chow, S.S., Wong, D.S., Xie, D.: Multi-authority ciphertext-policy attribute-based encryption with accountability. In: ACM Conference on Computer and Communications Security, pp. 386–390 (2011)
13. Li, J., Ren, K., Zhu, B., Wan, Z.: Privacy-aware attribute-based encryption with user accountability. In: Samarati, P., Yung, M., Martinelli, F., Ardagna, C.A. (eds.) ISC 2009. LNCS, vol. 5735, pp. 347–362. Springer, Heidelberg (2009)
14. Libert, B., Vergnaud, D.: Towards black-box accountable authority ibe with short ciphertexts and private keys. In: Jarecki, S., Tsudik, G. (eds.) PKC 2009. LNCS, vol. 5443, pp. 235–255. Springer, Heidelberg (2009)
15. Liu, Z., Cao, Z., Wong, D.S.: Blackbox traceable CP-ABE: how to catch people leaking their keys by selling decryption devices on eBay. In: ACM Conference on Computer and Communications Security, pp. 475–486 (2013)
16. Liu, Z., Cao, Z., Wong, D.S.: White-box traceable ciphertext-policy attribute-based encryption supporting any monotone access structures. IEEE Transactions on Information Forensics and Security **8**(1), 76–88 (2013)
17. Liu, Z., Wong, D.S.: Practical attribute based encryption: traitor tracing, revocation, and large universe. Cryptology ePrint Archive, Report 2014/616 (2014). http://eprint.iacr.org/

18. Ning, J., Cao, Z., Dong, X., Wei, L., Lin, X.: Large universe ciphertext-policy attribute-based encryption with white-box traceability. In: Kutyłowski, M., Vaidya, J. (eds.) ICAIS 2014, Part II. LNCS, vol. 8713, pp. 55–72. Springer, Heidelberg (2014)
19. Ostrovsky, R., Sahai, A., Waters, B.: Attribute-based encryption with non-monotonic access structures. In: ACM Conference on Computer and Communications Security, pp. 195–203 (2007)
20. Rouselakis, Y., Waters, B.: Practical constructions and new proof methods for large universe attribute-based encryption. In: ACM Conference on Computer and Communications Security, pp. 463–474 (2013)
21. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)
22. Schnorr, C.-P.: Efficient identification and signatures for smart cards. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 239–252. Springer, Heidelberg (1990)
23. Shamir, A.: How to share a secret. Communications of the ACM **22**(11), 612–613 (1979)
24. Wang, Y., Chen, K., Long, Y., Liu, Z.: Accountable authority key policy attribute-based encryption. Science China Information Sciences **55**(7), 1631–1638 (2012)
25. Waters, B.: Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC 2011. LNCS, vol. 6571, pp. 53–70. Springer, Heidelberg (2011)
26. Yu, S., Ren, K., Lou, W., Li, J.: Defending against key abuse attacks in KP-ABE enabled broadcast systems. In: Chen, Y., Dimitriou, T.D., Zhou, J. (eds.) SecureComm 2009. LNICST, vol. 19, pp. 311–329. Springer, Heidelberg (2009)