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

Zhangx

**Abstract.** To ensure the security of sensitive data, people need to encrypt them before uploading them to a public storage. Attribute-based encryption (ABE) is a promising cryptographic primitive for fine-grained sharing of encrypted data. In ABE, data owner can encrypt data through attributes or access structures. Only users who satisfy ciphertext’s access structures or attributes can decrypt them. However, ABE has a major drawback - the problem of user and authority accountability. 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 CP-ABE with user and authority accountability in white-box model. As embedding user’s identity ID into user’s secret key directly, the trace stage is very efficient. We prove our scheme selectively secure in the standard model under q-type assumption. The accountability property is also proven against dishonest user and KGC in the standard model. We implement our scheme in Charm. Experiments show that we enhance CP-ABE of Rouselakis and Waters in CCS 2013 with small computational cost.

**Keywords:** attribute-based encryption, user accountability, authority accountability, while-box model

1. Introduction

Cloud computing, as a new business computing mode, is changing the way we de-liver 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 or individuals choose to put their outsourced data into the cloud. However, cloud service providers are generally assumed to be untrusted parties, which mean they may be curious about the content of their users’ data for recommending advertisements or even leak the data to data owner’s competitors driven by interests. 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 encryption and sharing of outsourced data.

The concept of ABE was first proposed by Sahai and Waters in 2005 [1], as a generalization of identity-based encryption (IBE) [2]. In [1], the access policy is restricted to threshold. Goyal et al. [3] proposed a more expressive key-policy ABE (KP-ABE) scheme and formalized the notion of ciphertext-policy ABE (CP-ABE) [4]. The latter is conceptually closer to traditional role-based access control methods. In a CP-ABE system, a user’s access permissions are fully determined by his/her secret key.

Nevertheless, ABE has a major drawback which is known as user accountability problem. As user’s secret keys do not include the information which indicates the user’s identity, a dishonest user even need not to 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. Imagine that company A adopts CP-ABE system to share classified documents on cloud with the access policy such as {(Accountant AND Department of Finance) OR Manager}. Suppose Alice and Bob has the attribute set {Account, Department of Finance} while Lucy has the attribute set {Manager, Department of Marketing}. A key generation center (KGC), for instance, Department of Personnel, generates the corresponding secret key to them. They all have the access rights to this document. Once the content of this document is leaked to company B, then who is the traitor, Alice, Bob or Lucy? 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 respective user identities by public key infrastructure (PKI). Thus the general method for user accountability is to embed the identity-related information to 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 to the user accountability problem, there is also another problem named authority accountability. Suppose that we find the traitor is Bob by using the technology of user accountability. However, Bob may claim that he is innocent and framed by the KGC. As KGC has the power to generate secret key for any user with any attribute set, it is hard to distinguish whether Bob is innocent or not. For authority accountability, 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 [5], which means there are a cluster of secret keys related with each user, a user is innocent if the key family number of this user is not the same with the suspected one, and then we can accuse of the KGC.

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 input and observing the output of the box. Notice that Liu et al. [6] use the word “traceability” other than “accountability”, and in our context, these two words have the same meaning.

* 1. Related Work

ABE comes into two flavors, key-policy ABE (KP-ABE) [3, 7, 8] and ciphertext-policy ABE (CP-ABE) [4, 9, 10]. In KP-ABE, ciphertexts are associated with sets of attributes and user’s secret keys are associated with access structures. When ciphertext is created, data owner does not know who will access later. KP-ABE focuses on the specific need of user. Whatever user needs, KGC will generate secret keys corresponding to proper access structure. In CP-ABE, the situation is reversed, user’s secret keys are labeled by attributes and ciphertexts are associated with access structures. Before encrypting, data owner clearly knows that what kind of people are allowed access. Lewko and Waters [11] proposed the first unbounded large universe KP-ABE scheme in composite order bilinear groups. Unbounded means no bound on the size of attribute sets used for encryption while a bound was fixed at setup on the number of attributes that can be used for encryption large universe KP-ABE of [3]. Okamoto and Takashima [12] proposed the first unbounded large universe CP-ABE by using dual pairing vector space (DPVS). Rouselakis and Waters [13] proposes practical constructions of unbounded large universe KP-ABE and CP-ABE, which achieves remarkable efficiency improvement over [11, 12].

In ABE, most of the concern is user accountability [14] which assuming KGC is trusted. Hinek et al. [14] proposed a token-based ABE. When decrypting, user must request a decryption token from a third party token server. Therefore, the token server is required to be online. Yu et al. [15] proposed a novel 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. [16] proposed an accountable, anonymous, and ciphertext-policy 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 [6] and black-box [17] traceable CP-ABE respectively. It can support any monotone access structures while it only supports AND gate with wildcards prior to Liu et al.’s work. However, these two schemes use bilinear groups of large composite order and are very inefficient. Ning et al. [18] proposed a large universe CP-ABE with user accountability in white-box model on bilinear groups of prime order. Liu and Wong [19] proposed both large universe KP-ABE and CP-ABE with user accountability in black-box model on bilinear groups of prime order. The scheme supported revocation for dishonest user.

Wang et al. [20] achieved authority accountability in white-box model by combining Libert and Vergnaud’s IBE scheme [21] and KP-ABE [3]. As the user’s secret key contains the secret information that the KGC does not know, if KGC forges secret key in accordance with the user’s identity, we can fine whether the KGC or the user is dishonest according the key family number. But yet it does not support large universe.

Authority accountability can be considered as accounting the abused part after the problem is occurring. There is also another perspective - avoiding the problem before occurring. In addition to that the KGC can generate user’s secret key with arbitrary access structures or set of attributes, the KGC can decrypt the ciphertext directly by using its master key. It is known as the key escrow problem. Hur [22] proposed a secure two party computation protocol for resolving the key escrow problem. The KGC is divided into two parts: KGC and the data storing center. The secure 2PC protocol ensures that neither of them could generate the user’s secret key all alone. Nevertheless, this scheme still suffers from the problem of user accountability.

In multi-authority ABE [23, 24], different authorities operate simultaneously and each hands out user’s partial secret keys for a different set of attributes. Li et al. [25] proposed a multi-authority CP-ABE scheme with user accountability. However, it only supports access structure with AND gate with wildcards.

Goyal [5] firstly presented the notion of accountable authority IBE (A-IBE). Nevertheless, it merely allows tracing well-formed decryption keys. This while-box model seems unlikely to suffice in practice since malicious parties can rather release an obfuscated program that only decrypts with small but noticeable probability. The black-box accountability was achieved in [26, 27].

* 1. Our Contributions

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

1) We propose an attribute-based encryption scheme with user and authority accountability 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 proven against dishonest user and KGC in the standard model.

3) By embedding user’s identity ID into 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.

4) Our scheme is very efficient. We enhance CP-ABE of Rouselakis and Waters [13] with small computational cost.

5) We extend our scheme to support online/offline encryption and outsourcing decryption.

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

* 1. Our Main Ideas

Now we will briefly describe the main ideas in our scheme.

We extends large universe CP-ABE of Rouselakis and Waters [13] to support accountability for user and authority. To find out the identity of dishonest user, Liu et al. [6] use an identity table to connect 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 the Shamir’s threshold scheme [28] to trace the

Table 1. Comparisons with other related works

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Schemes | Category | Large Universe | Supporting  Monotonic  Access Structure | Order of Bilinear Groups | User  Accountability | Authority  Accountability | Security Model1 |
| [16] | CP-ABE | 🗴 | 🗴 | prime | black-box2 | 🗴 | selectively secure |
| [20] | KP-ABE | 🗴 | 🗸 | prime | white-box | white-box | selectively secure |
| [6] | CP-ABE | 🗴 | 🗸 | composite | white-box | 🗴 | fully  secure |
| [17] | CP-ABE | 🗴 | 🗸 | composite | black-box | 🗴 | fully  secure |
| [18] | CP-ABE | 🗸 | 🗸 | prime | white-box | 🗴 | selectively secure |
| [19] | KP-ABE  CP-ABE | 🗸 | 🗸 | prime | black-box | 🗴 | selectively secure |
| Ours | CP-ABE | 🗸 | 🗸 | prime | white-box | white-box | selectively secure |

1All schemes are secure in the standard model.

2[6] gives a “compare-before-output” technique to avoid the tracing algorithm from identifying the dishonest user in Appendix A.

dishonest user. As every user has a unique identity ID in the system, can we embed ID into user’s secret key directly? If succeeded, the trace stage would become very simple, the only thing needed to do is to check whether the suspected secret key is well-formed or not. Liu et al. [6] in their extensions gives some suggestions by using another signature scheme[[1]](#footnote-1) in [29]. However, they do not give complete proof. Their scheme uses bilinear groups of composite order and merely supports user accountability in white-box model. In our scheme, we successfully embed that signature scheme into our prime order construction and give complete proof.

In order to achieve authority accountability, we borrow some idea from A-IBE [5]. Nevertheless, in IBE, both secret key and ciphertext contain user’s specific identity information. In ABE, ciphertext is used for sharing and cannot contain user’s specific identity information. Thanks to the secret key structure of Rouselakis and Waters [13], it employs “attribute” layer and “secret sharing” layer, and uses a “binder term” to connect them. Then, if we change something in “secret sharing” layer, it does not affect the “attribute” layer. Therefore, we embed secret information which is hidden from the KGC’s view into the “secret sharing” term of the user’s secret key. This trick does not affect the computation of the “attribute” layer in the decryption.

* 1. Organization

The remainder of the paper is organized as follows. Section 2 introduces the background including access structure, linear secret sharing scheme, bilinear groups and some assumptions. In Section 3, we gives the formal definition of CP-ABE with user and authority accountability (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. Section 6 discusses some extends. Finally, we give a brief conclusion in Section 7.

1. Background
   1. Access Structure

**Definition 2.1 Access Structure** [30]

Let be a set of parties. A collection is monotone if : if and then . An access structure (respectively, monotone access structure) is a collection (respectively, monotone collection) of non-empty subsets of , i.e., . The sets in are called the authorized sets, and the sets not in are called the unauthorized sets.

In our context, the role of the parties is taken by the attributes. Thus, the access structure will contain the authorized sets of attributes. From now on, unless stated otherwise, by an access structure we mean a monotone access structure.

* 1. Linear Secret Sharing Schemes

**Definition 2.2 Linear Secret Sharing Schemes (LSSS)** [30]

A secret sharing scheme over a set of parties is called linear (over ) if

1. The shares for each party form a vector over .
2. There exists a matrix an M with rows and columns called the share-generating matrix for. For all *i = 1,…l*, the *i*th row of M we let the function defined the party labeling row *i* as . When we consider the column vector , where is the secret to be shared, and are randomly chosen, the is the vector of *l* shares of the secret s according to. The share belongs to party .

According to [30], every LSSS according to the above definition also enjoys the linear reconstruction property. Suppose thatis an LSSS for the access structure . Let be any authorized set if , and let be defined as . Then, there exist constants such that, if are valid shares of any secret s according to, then .

* 1. Bilinear Maps

**Definition 2.3 Bilinear Maps**

Let and be two multiplicative cyclic groups of prime order . Let be a generator of and be a bilinear map, . The bilinear map has the following properties:

1. Bilinearity: for all and , we have .
2. Non-degeneracy: .
3. Computable: there exists an efficient algorithm for the bilinear map .

Notice that the map is symmetric since .

* 1. Assumptions

In our construction, we adopt the q-type assumption of Rouselakis and Waters’ scheme [13].

**Assumption 2.1 q-type assumption**

Initially the challenger calls the group generation algorithm with input the security parameter, picks a random group element , and q+2 random exponents . Then he sends to the adversary the group description and all of the following terms:

It is hard for the adversary to distinguish from an element which is randomly chosen from .

**Definition 2.4** 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 2.2 *l*-Strong Diffie-Hellman assumption** [29]

Given a (*l*+1)-tuple as input, it is hard for the adversary to output a pair where .

**Definition 2.5** We say that the *l*-SDH assumption holds if no PPT adversary has a non-negligible advantage in solving the *l* -SDH problem.

* 1. Miscellaneous Primitives

**Zero-knowledge Proof of Knowledge of Discrete Log** A zero-knowledge proof[[2]](#footnote-2) 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 [5, 31] enables a prover to prove to a verifier that it possesses the discrete log r of a given group element R in question.

1. 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. Definition

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

* Setup : This is a randomized algorithm that takes a security parameter encoded in unary. It outputs the public parameters PK and master key MK.
* KeyGen : 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. It outputs this user’s secret key SK.
* Encrypt : This is a randomized algorithm that takes as input the public parameters PK, a plaintext message M, and an access structure. It outputs the ciphertext CT.
* Decrypt : This algorithm takes as input the public parameters PK, a secret key SK for user ID with a set of attributes, and a ciphertext CT that was encrypted under access structure . It outputs the message M if.
* Trace : This algorithm has two stages. In the first stage, it takes as input a decryption key and outputs a user’s identity with a key family number or the special symbol if is ill-formed. In the second stage, it compares the key family number of the secret key of user ID with . If, it outputs ID assuming the user ID is dishonest. Otherwise, it outputs “KGC”. This definition of Trace is for the white-box setting.
  1. **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 and a challenger and is parameterized by the security parameter. The phases of the game are as follows.

* **Init:** The adversary declares the challenge access structure which he wants to attack, and then sends it to the challenger.
* **Setup:** The challenger runs the Setup algorithm and gives the public parameters PK to the adversary.
* **Phase 1:** The adversary is allowed to issue queries for secret keys for users with sets of attributes,,…,. For each , the challenger calls KeyGen and sends to . The only restriction is that does not satisfy .
* **Challenge:** The adversary submits two equal length message and . The challenger flips a random coin, and encrypts with. The ciphertext is passed to .
* **Phase 2:** Phase 1 is repeated.
* **Guess:** The adversary outputs a guess of .

The advantage of an adversary in this game is defined as .

**Definition 3.1** 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 in the above security game.

* 1. **Accountability Model for UaAA-CP-ABE**

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

### a) The DishonestKGC Game**.**

The intuition behind this game is that an adversarial KGC attempts to calculate user’s key family number 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 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 randomly and sends to the adversary. The challenger also need to give to the adversary a zero-knowledge proof of knowledge of the discrete log of with respect to Then the adversary calls KeyGen and sends 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 related with ID. The challenger checks that  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 equivalent to ’s. The advantage of an adversary in this game is defined as .

### 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 with an existed ID appeared at Key Query stage. At Key Query stage, the adversary has already got . In this game, a new key with an existed ID means that the identity-related information in is successfully changed by the adversary. A tuple 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 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. The challenger responds to each query by calling KeyGen .
* **Key Forgery:** Eventually, the adversary outputs a decryption key related with and wins the game if

(1)  is not any of , and

(2) 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 .

### 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 ) in that user’s secret key. The DishonestUser-2 Game for our scheme is defined as follows.

* **Setup:** The challenger runs the Setup 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. The challenger responds to each query by calling KeyGen .
* **Key Forgery:** The adversary will output a decryption key related with and wins the game if

(1) is one of , we assume is equivalent to , and

(2) does not equal to , and

(3) 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 .

**Definition 3.2** 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.

1. Our Construction

Let be a bilinear group of prime order , and let be a generator of . In addition, let denote the bilinear map. A security parameter will determine the size of the groups. For the moment we assume that users’ identity IDs and attributes are elements in, however, IDs and attributes can be any meaningful unique strings using a collision resistant hash function.

Our construction follows.

* Setup : The algorithm calls the group generator algorithm and gets the descriptions of the groups and the bilinear mapping . Then it picks the random terms and . The published public parameters PK are

The master key MK are .

* KeyGen : After user is authenticated, the KGC gets  from user ID where user chooses randomly. User ID also need to give to KGC a zero-knowledge proof of knowledge of the discrete log (as in Section 2.5) of with respect to Then it picks random exponents. It outputs this user’s secret key SK (Notice that is owned by the user secretly, and is part of SK):

Here is computed modulo . In the unlikely event that we will pick another random .

* Encrypt : To encrypt a message under an access structure encoded in an LSSS policy . Let the dimensions of M be . Each row of M will be labeled by an attribute and denotes the label of row . Choose a random vector from, s is the random secret to be shared among the shares. The vector of the shares is. It then chooses random value and publish the ciphertext as:
* Decrypt : To decrypt the ciphertext CT with the decryption key SK, proceed as follows. Suppose that satisfies the access structure and let . Since the set of attributes satisfy the access structure, there exist coefficients such that . Then we have that . Now it calculates
* Trace : If is ill-formed, the algorithm will output the special symbol . Otherwise, it outputs and key family number in . If ID does not exist, the algorithm outputs “KGC” which means the dishonest KGC create a fake user’s identity. Otherwise, it compares with the key family number of the secret key of real user ID. If, it outputs ID assuming the user ID is dishonest. Otherwise, it outputs “KGC”. Notice that we do not need to compare the signature part in these two keys, because key family number is enough to distinguish dishonest user or KGC.

1. Analysis of Our Proposed Scheme
   1. Selective Security Proof

In our original scheme, the KGC does not have complete control over SK because it does not know  in . 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 [5]. 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 .

In the selective security proof, we will reduce the selective security of our CP-ABE scheme to that of Rouselakis and Waters’ [13] which is proved selectively secure under Assumption 2.1.

**Theorem 5.1** If Rouselakis and Waters’ scheme [13] is selectively secure, then all PPT adversaries with a challenge matrix of size , where , have a negligible advantage in selectively breaking our scheme.

**Proof.** To prove the theorem we will suppose that there exists a PPT adversary with a challenge matrix that satisfies the restriction, which has a non-negligible advantage in selectively breaking our scheme. Using this adversary we will build a PPT simulator that attacks Rouselakis and Waters’ scheme () [13] with a non-negligible advantage.

* **Init:** The adversary declares a challenge access policy which he wants to attack, and then sends it to the challenger. sends this received challenge access policy to . Notice that is a matrix, where . Each row of will be labeled by an attribute and denotes the label of row of .
* **Setup:** gets the public parameters from . Then chooses randomly, and gives the public parameters PK= to . Notice that this way is information-theoretically hidden from .
* **Phase 1:** Now has to produce secret keys for tuples which consists of non-authorized sets of attributes , users’ identity ID, and an element computed with a zero-knowledge proof. The only restriction is that does not satisfy . As analysis in the beginning part of this section, we assume knows . At first, will issue to and get the corresponding decryption key as follows:

Then picks random exponents, and sets and implicitly. Here is computed modulo . In the unlikely event that , will pick another random . Then computes

Finally, sends the decryption key to . Notice that is owned by .

* **Challenge:** The adversary submits two equal length message and . Then submits and to , and gets the challenge ciphertext as follows:

Notice that has two forms indeed according to the proof part of Rouselakis and Waters’ scheme [13], one is well-formed (), and the other is random.

Then computes . Finally, sends the challenge ciphertext to .

* **Phase 2:** Phase 1 is repeated.
* **Guess:** The adversary  outputs a guess of to . Then sends to .

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 in selectively breaking our scheme. As Rouselakis and Waters’ scheme is selectively secure, so do ours. □

* 1. Accountability Proof

### a) Analysis of The DishonestKGC Game**.**

**Theorem 5.2** Assuming that computing discrete logarithm is hard in , 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 which has a non-negligible advantage in the DishonestKGC Game in our scheme. Using this adversary we will build a PPT simulator that attacks the discrete logarithm problem with a non-negligible advantage. proceeds as follows.

* **Setup:** The adversary (acting as an adversarial KGC) runs the Setup algorithm and gives the public parameters PK= and a user’s identity ID to the simulator . checks that PK and ID are well-formed and aborts if the check fails.
* **Key Generation:** invokes the challenger , passes on to it and gets a challenge . Then engages in the key generation protocol with to get a decryption key for ID as follows. Notice that should give to a zero-knowledge proof of knowledge of the discrete log of with respect to, however, does not know . A similar situation occurs in A-IBE [5]. In the part of security proof of the FindKey game in A-IBE, simulate the required proof without knowledge of . In our proof, we will use the same technology and assume that successfully gives to a zero-knowledge proof of knowledge. Then calls KeyGen and sends to .
* **Key Forgery:** will output a decryption key related with ID. checks that is well-formed and aborts if the check fails. If is well-formed, sends to .

If in the DishonestKGC Game is non-negligible, we have built a PPT simulator 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 which has a non-negligible advantage in the DishonestKGC Game in our scheme. □

### b) Analysis of The DishonestUser-1 Game**.**

**Theorem 5.3** The advantage of an adversary in the DishonestUser-1 Game is negligible for our CP-ABE scheme under the *l*-SDH assumption.

Due to space limitations, we refer the interested reader to Appendix A for the proof of this theorem.

### c) Analysis of The DishonestUser-2 Game**.**

**Theorem 5.4** Assuming that computing discrete logarithm is hard in , the advantage of an adversary in the DishonestUser-2 Game is negligible for our scheme.

**Proof.** It is very difficult to prove this theorem using the formal proof method. We will use plausible reasoning to explain why the advantage of an adversary in the DishonestUser-2 Game is negligible informally. Now let’s review the user’s secret key SK’s format:

And the target of adversary is to output a forged secret key where :

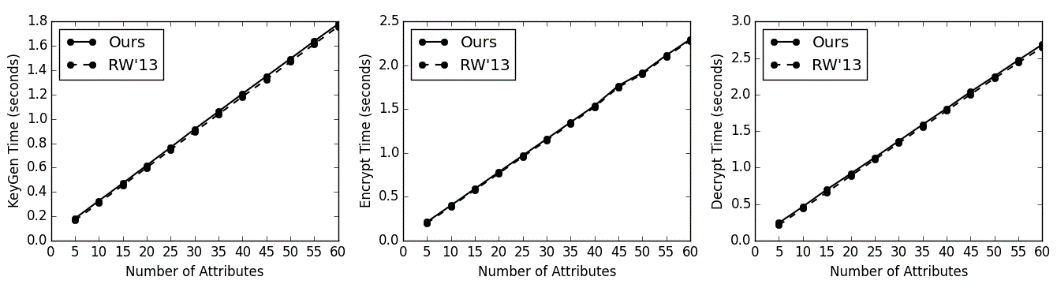
Firstly, we will analysis and . As and in is information-theoretically hidden from . If can forge successfully, then we can assume that . Similarly, since in is information-theoretically hidden from , if can forge successfully, we can assume that . Then we get two equations:

From ’s view, knows . If , then . Apparently, the probability of is negligible. Then can compute which is discrete logarithm of . However, as we assume, computing discrete logarithm is hard in , then cannot forge a secret key where . Therefore, the advantage of an adversary in the DishonestUser-2 Game is negligible for our scheme. □

* 1. Performance Analysis

We implemented our scheme and Rouselakis and Waters (RW’13) [13] in Charm[[3]](#footnote-3) [32]. We wanted to know how much computational efficiency to lose for security enhancements of RW’13. We use “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 millisecond and 56 millisecond 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. Therefore, our scheme is very efficient. Notice that the encryption time of our scheme in the figure is only for KGC. Users



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

Fig. 1. Comparison of KeyGen, Encrypt and Decrypt

also need to give to KGC a zero-knowledge proof of knowledge of the discrete log of with respect to w.

1. Extensions
   1. Online/Offline

In our proposed scheme, it is feasible to trace both malicious user and authority reliably. With the mobile devices being widely used, a remarkable issue should be concerned. How to make the devices which are not powerful run our system efficiently? Some works [33, 34] have done in implementing an online/offline system by splitting the encryption into two phases: offline and online. The core idea of online/offline is going to put most of the heavy computations, such as modular exponentiation and hashing into the offline phase, and then the online phase only performs some light-weighted computations, such as integer addition and multiplication. Using online/offline ABE [33], we can naturally transform our construction into the unbounded “online-offline” style to realize a “light-weighted” and efficient encryption computation. At the same time, another issue also needs to be paid attention to. Key generation algorithm may become a bottleneck in such a scene that a large number of users in the system are reissued secret keys frequently for the purpose of revocation. In order to prevent it from occurring, the KeyGen stage also needs to be spitted into the online phase and offline phase. The proof of security about the construction mentioned above can be easily to reduce security to CP-ABE of Rouselakis and Waters [13] just like [33].

* 1. Outsourcing Decryption

Besides online/offline technique, we can use the technique of outsourcing decryption to save user’s bandwidth and decryption time in mobile scenario. As computing power is limited and battery life is a persistent problem in mobile device, the decryption time in ABE is unacceptable. Green et al. [35] proposed a method for outsourcing the decryption of ABE ciphertexts. User has a transformation key and an ElGamal-style key. The transformation key can be given to the cloud to translate any ABE ciphertext satisfied by that user’s attributes into an ElGamal-style ciphertext without revealing ciphertext’s content to the cloud. Then user can use the ElGamal-style key to decrypt that ElGamal-style ciphertext quickly. The traditional process is as follows. Suppose is a randomly ElGamal-style key. We can perform exponential operations on every element in of the original scheme’s user’s key by using . Then we have generated the transformation key. During applying this technique to our scheme, we are surprised to find that our scheme is naturally support outsourcing the decryption of ciphertexts. The key family number is already our ElGamal-style key. The transformation key is user’s secret key except . The cloud computes and in our decryption algorithm in Section 4. Then user can recover the message .

1. Conclusion

The problem of user and authority accountability is an important challenging issue in ABE. User is able to share his/her secret key and abuse his/her access privilege without being identified, and KGC can generate user’s secret key with arbitrary access structure or set of attributes. In this paper, we propose a practical large universe CP-ABE with user and authority accountability. We can trace 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. We extend our scheme to support online/offline encryption and outsourcing decryption. 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.

### References

1. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)
2. Boneh, D., Franklin, M.: Identity-based encryption from the Weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
3. 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)
4. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: IEEE Symposium on Security and Privacy, pp. 321–334 (2007)
5. 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)
6. 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)
7. 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)
8. 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)
9. Cheung, L., Newport, C.: Provably secure ciphertext policy ABE. In: ACM Conference on Computer and Communications Security, pp. 456–465 (2007)
10. 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)
11. Lewko, A., Waters, B.: Unbounded HIBE and attribute-based encryption. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 547–567. Springer, Heidelberg (2011)
12. Okamoto, T., Takashima, K.: Fully secure unbounded inner-product and attribute-based encryption. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 349–366. Springer, Heidelberg (2012)
13. 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)
14. 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/>
15. 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.) Security and Privacy in Communication Networks, vol. 19, pp. 311-329. Springer, Heidelberg (2009)
16. 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.) Information Security 2009. LNCS, vol. 5735, pp. 347-362. Springer, Heidelberg (2009)
17. 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)
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.) ESORICS 2014. LNCS, vol. 8713, pp. 55-72. Springer, Heidelberg (2014)
19. 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/>
20. Wang Y., Chen K., Long Y., Liu Z.: Accountable authority key policy attribute-based encryption. Science China Information Sciences 55 (7), 1631-1638 (2012)
21. 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)
22. Hur, J.: Improving security and efficiency in attribute-based data sharing. IEEE Transactions on Knowledge and Data Engineering 25 (10), 2271-2282 (2013)
23. Chase, M.: Multi-authority attribute based encryption. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 515–534. Springer, Heidelberg (2007)
24. 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)
25. 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)
26. Goyal V., Lu S., Sahai A., Waters B.: Black-box accountable authority identity-based encryption. In: ACM Conference on Computer and Communications Security, pp. 427-436 (2008)
27. Libert, B., Vergnaud, D.: Towards practical black-box accountable authority IBE: Weak black-box traceability with short ciphertexts and private keys. IEEE Transactions on Information Theory 57 (10), 7189-7204 (2011)
28. Shamir, A.: How to share a secret. Communications of the ACM 22(11), 612–613 (1979)
29. 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)
30. Beimel, A.: Secure schemes for secret sharing and key distribution. PhD thesis, Israel Institute of Technology, Technion, Haifa, Israel (1996)
31. Schnorr C.: Efficient identification and signatures for smart cards. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 239-252. Springer, Heidelberg (1990)
32. 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)
33. Hohenberger S., Waters B.: Online/offline attribute-based encryption. In: Krawczyk, H. (ed.) PKC 2014. LNCS, vol. 8383, pp. 293-310. Springer, Heidelberg (2014)
34. Guo, F., Mu, Y., Chen, Z.: Identity-based online/Offline encryption. In: Tsudik, G. (ed.) FC 2008. LNCS, vol. 5143, pp. 247–261. Springer, Heidelberg (2008)
35. Green, M., Hohenberger, S., Waters, B.: Outsourcing the decryption of ABE ciphertexts. In: USENIX Security Symposium (2011)

A Analysis of The DishonestUser-1 Game

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

We first give some intuition for the proof. Assuming issues q queries, For each secret key, we record a tuple . At Key Forgery stage, the adversary outputs a decryption key related with . There are two possibilities when the adversary wins the game, or . We distinguish between two types of adversaries.

Type-1 adversary: an adversary that either

(1) makes a secret key query for user’s identity at Key Query stage, or

(2) outputs a decryption key related with at Key Forgery stage.

Type-2 adversary: an adversary that both

(1) never makes a secret key query for user’s identity at Key Query stage, and

(2) outputs a decryption key related with at Key Forgery stage.

We will show that either adversary can be used to solve the *l*-SDH assumption. However, the simulator works differently for each adversary type. Thus, will choose a random bit that indicates its guess for the type of adversary that will emulate.

is given a bilinear mapping and a random instance of the *l*-SDH problem for some unknown . Then proceeds as follows.

* **Setup:** chooses random elements . Let be the polynomial . Expand and write where are the coefficients of the polynomial . Compute:

Notice that we may assume that , otherwise, for some *i* which means that just obtains the secret key of the *l*-SDH problem.

Then picks the random terms, and

If , picks a random and gives the public parameters .

If , picks a random and gives the public parameters .

Notice that in either case, provides the adversary with a valid public parameters.

* **Key Query:** The adversary can issue up to queries for secret keys adaptively. In order to respond, maintains a list H-list of tuples . Then for the ith query :

Let be the polynomial . Expand and write where are the coefficients of the polynomial . Compute

* If , check if . If so, just obtains the secret key of the *l*-SDH problem which allows it to compute for any easily. At this point successfully solves the *l*-SDH assumption.

Otherwise, sets . If , reports failure and aborts. Otherwise, it picks random exponents and outputs ’s secret key ( is owned by the adversary secretly, and is part of )

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

* If , sets . If , reports failure and aborts. Otherwise, it picks random exponents and outputs ’s secret key ( is owned by the adversary secretly, and is part of )

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

In either case adds the tuple to the H-list.

* **Key Forgery:** Eventually, the adversary outputs a decryption key related with where is well-formed and is not any of . Notice that by adding dummy queries as necessary, we may assume that the adversary made exactly queries. Let . Then searches from the H-list. There are two possibilities:

Type-1 adversary: No tuple of the form appears on the H-list.

Type-2 adversary: The H-list contains at least one tuple such that .

Let if produced a type-1 adversary. Otherwise, set . If , reports failure and aborts.

* If , check if . If so, can solve the *l*-SDH assumption successfully. Otherwise, compute

Let . Notice that when adversary is type-1.

Using long division we write the polynomial f as for some polynomial and . Then

and hence

Notice that , since and . Then computes

and returns as the solution to the *l*-SDH problem.

* If , let be a tuple on the H-list where . Since , we know that . We know that , otherwise, the adversary failed to forge a secret key SK and would lose the game. Therefore, . As knows x, can solve the *l*-SDH assumption successfully.

Now we complete the description of simulator . Notice that,

(1) the view from is independent of the choice of ,

(2) the public parameters are uniformly distributed, and

(3) the secret keys that queries are well-formed.

Therefore, produces a valid secret key with probability at least .

It remains to bound the probability that does not abort. We argue as follows:

If , aborts when forged a secret key with . This happens with probability at most .

If , does not abort.

Since is independent of we have that . It now follows that produces a valid tuple with probability

□

1. [6, 18] use the same signature scheme in [29] to achieve user accountability. [↑](#footnote-ref-1)
2. http://en.wikipedia.org/wiki/Zero-knowledge\_proof [↑](#footnote-ref-2)
3. You can download our code from https://github.com/zlwen/charm-example [↑](#footnote-ref-3)