A^2BE: Accountable Attribute-Based Encryption for Abuse Free Access Control

Jin Li[1], Kui Ren[1], and Kwangjo Kim[2]

[1] Department of Electrical and Computer Engineering
Illinois Institute of Technology
Chicago, Illinois 60616, USA
{jin.li@iit.edu,kren@ece.iit.edu}

[2] International Research center for Information Security (IRIS)
Information and Communications University(ICU)
103-6 Munji-Dong, Yuseong-Gu, Daejeon, 305-732, Korea
{kkj@icu.ac.kr}

Abstract. As a recently proposed public key primitive, attribute-based encryption (ABE) (including Ciphertext-policy ABE (CP-ABE) and Key-policy ABE (KP-ABE)) is a highly promising tool for secure fine-grained access control. For the purpose of secure access control, there is, however, still one critical functionality missing in the existing ABE schemes, which is the prevention of key abuse. In particular, two kinds of key abuse problems are considered in this paper, i) illegal key sharing among colluding users and ii) misbehavior of the semi-trusted attribute authority including illegal key (re-)distribution. Both problems are extremely important as in an ABE-based access control system, the attribute private keys directly imply users' privileges to the protected resources. To the best of our knowledge, such key abuse problems exist in all current ABE schemes as the attribute private keys assigned to the users are never designed to be linked to any user specific information except the commonly shared user attributes.
To be concrete, we focus on the prevention of key abuse in CP-ABE in this paper [3]. The notion of accountable CP-ABE (CP-A^2BE, in short) is first proposed to prevent illegal key sharing among colluding users. The accountability for user is achieved by embedding additional user specific information in the attribute private key issued to the user. To further obtain accountability for the attribute authority as well, the notion of Strong CP-A^2BE is proposed, allowing each attribute private key to be linked to the corresponding user's secret that is unknown to the attribute authority. We show how to construct such a Strong CP-A^2BE scheme and prove its security based on the computational Diffie-Hellman assumption.

Keywords: Access control, Key abuse, Attribute-based, Ciphertext-policy, Encryption, Trace

1 Introduction

Today's computing and electronic technology innovations have unprecedentedly enabled ubiquitous information generation, processing, and distribution in both volume and speed. Vast amounts of information resources are made available and readily accessible to individuals and organizations through various computer systems and the Internet. This trend, however, also poses new challenges in designing suitable secure access control mechanisms.

Recently, the notion of ABE, which was proposed by Sahai and Waters [28], has attracted much attention in the research community to design flexible and scalable access control systems. For the first time, ABE enables public key based one-to-many encryption. Therefore,

[3] Our technique can easily be extended to KP-ABE as well.

it is envisioned as a highly promising public key primitive for realizing scalable and fine-grained access control systems, where differential yet flexible access rights can be assigned to individual users. To address complex and general access policy, two kinds of ABE have been proposed: KP-ABE and CP-ABE. In KP-ABE, access policy is assigned in attribute private key, whereas, in CP-ABE, the access policy is specified in the ciphertext.

In this paper, we address the problem of accountability in ABE-based secure access control systems. The accountability is critical in access control systems because it deals with the key abuse problem, which includes i) illegal key sharing among users and ii) misbehavior of the attribute authority, e.g., illegal key (re-)distribution. In an ABE-based access control system, the attribute private keys directly imply users' privileges to the protected resources. On the one hand, the dishonest users may illegally share their attribute private keys with other users and thus abuse the system without being detected. On the other hand, a semi-trusted attribute authority may illegally generate and distribute legitimate keys to unauthorized users. These accountability problems by far are not yet to be properly considered in existing ABE-based access control schemes. Clearly, to securely deploy an ABE-based access control system, it is imperative to guarantee that 1) the key issued to each user cannot be shared; 2) the attribute authority's misbehavior, i.e., distributing decryption keys or decrypting ciphertext arbitrarily for unauthorized users, should be prevented. To the best knowledge of ours, such key abuse problem exists in all ABE-based access control schemes, as their attribute keys are never designed to be linked to any user specific information except the commonly shared attributes.

Our Contribution The notion of accountable CP-ABE (CP-A^{2}BE) is first proposed to address the key abuse problems existed in access control based on ABE. In CP-A^{2}BE, the user accountability is achieved. To obtain further accountability for both users and the attribute authority, Strong CP-A^{2}BE is presented. Constructions of (Strong) CP-A^{2}BE are given, as well as their security analysis. The approaches designing the (Strong) CP-A^{2}BE schemes are new and different from techniques used in the previous accountable IBE schemes [2, 18, 20]. The key points for these constructions are described as follows:

- In CP-A^{2}BE, the user's identity is embedded into the attribute private key issued to him from the attribute authority. The CP-A^{2}BE can be used to prevent the key sharing among users based on the following observation: The user's decryption key consists of the attribute private key and the user's identity. The user cannot change his attribute private key to another one embedded with a different identity. As a result, the identity in the decryption key cannot be changed either. Therefore, if the user shares his decryption key, the identity will be detected from the pirate device, and the user will be punished.

- In Strong CP-A^{2}BE, the accountability of the attribute authority is further achieved in addition to user accountability. The Strong CP-A^{2}BE has the assumption that each user should get a higher level secret before requesting an attribute private key (We just consider the public key certificate for simplicity in this work). The public key is embedded into the user's attribute private key. Now, the user's decryption key contains the attribute private key and secret key corresponding to his certificate, which cannot be changed. As a consequence, the user's secret key in the certificate will be leaked if sharing his decryption key. Moreover, since the user's decryption key contains his secret key that is unknown to the attribute authority, the accountability for the semi-trusted attribute authority can be obtained too.

1.1 Related Work

Since the introduction of ABE in implementing fine-grained access control systems, a lot of works have been proposed to design efficient and flexible ABE schemes. There are two methods to realize the fine-grained access control based on ABE: KP-ABE and CP-ABE. They were both mentioned in [21] by Goyal *et al.* In KP-ABE, ciphertext is labeled with sets of attributes, and each attribute private key is associated with an access structure that specifies which type of ciphertexts the key is able to decrypt. In a CP-ABE system, on the contrary, each user's key is associated with a set of attributes and ciphertext will specify an access policy over attributes. Therefore, CP-ABE is different from KP-ABE in the sense that, in CP-ABE, it is the encryptor who assigns certain access policy in the ciphertext. When a message is being encrypted, it will be associated with an access structure over a predefined set of attributes. The user will only be able to decrypt a given ciphertext if his attributes match the access structure specified in the ciphertext. The first KP-ABE construction [21] realized the monotonic access structures for key policies. To enable more flexible access policy, Ostrovsky *et al.* [26] presented the first KP-ABE system that supports the expression of non-monotone formulas in key policies. However, KP-ABE is less flexible than CP-ABE because the policy is determined once the user's attribute private key is issued. Later, Bethencourt *et al.* [4] proposed the first CP-ABE construction. However, the construction [4] is only proved secure under the generic group model. To overcome this weakness, Cheung and Newport [12] presented another construction that is proved to be secure under the standard model. The construction supports the types of access structures that are represented by AND of different attributes. Later, in [19], the authors gave another construction for more advanced access structures based on number theoretic assumption. To further achieve receiver-anonymity, Boneh and Waters [8] proposed a predicate encryption scheme based on the primitive called Hidden Vector Encryption. The scheme in [8] can also realize the anonymous CP-ABE by using the opposite semantics of subset predicates. Katz, Sahai, and Waters [24] proposed a novel predicate encryption scheme supporting inner product predicates. Their scheme is very general and can achieve both KP-ABE and hidden CP-ABE schemes. However, the constructions of [8, 24] are very inefficient compared to [25]. In [22], they also mentioned the key abuse problem of users, and another third party should be involved in each user's decryption in their scheme, which makes it impractical. To reduce the trust of attribute authority in ABE, Chase [10] proposed a multi-authority ABE scheme, where each authority controls a subset of the attributes. If one wants to decrypt a ciphertext, he has to get enough attributes from each attribute authority.

Organization In Section 2, we show how to unify the definitions and security models for CP-ABE and KP-ABE , based on which we present the security models of CP-A^{2}BE and Strong CP-A^{2}BE. In addition, their constructions and security analysis are given. In Section 3, the implementation of Strong CP-A^{2}BE to access control are described. In Section 4, we present two interesting applications by utilizing the techniques used in the construction of (Strong) CP-A^{2}BE. This paper ends with concluding remarks.

2 Strong CP-A^{2}BE

In this Section, the system model of CP-A^{2}BE and its construction are firstly proposed to achieve user accountability. Then, we present how to get a Strong CP-A^{2}BE construction achieving accountability for both users and attribute authority. The security models and results of (Strong) CP-A^{2}BE are also demonstrated.

2.1 The CP-A^{2}BE

We first give the unified definitions for CP-ABE and KP-ABE, as an independent interest. Based on that, we define the proposed CP-A^{2}BE and show the construction with security result.

2.1.1 System Model

In both CP-ABE and KP-ABE architectures, there are two entities: the attribute authority and users. The attribute authority is in charge of the issue of attribute private keys to users requesting them. A message can be encrypted under a specified ciphertext-policy such that only users whose attribute private keys match the policy, are able to decrypt the ciphertext. The user can get his attribute private key from the attribute authority before decryption. The definitions of CP-ABE and KP-ABE are unified as X-ABE, where X means CP or KP here. A binary relation R is also introduced. We denote $R(L, W) = 1$ if L matches W, where L and R represent general access structure (or key-policy) and ciphertext-policy, respectively.

Definition 1. *An X-ABE system consists of four algorithms, namely, Setup, KeyGen, Enc, and Dec, which are defined as follows:*

Setup(1^λ) *The setup algorithm, takes as input security parameter 1^λ, outputs a master secret key sk and public key pk.*

KeyGen$(L,\ sk)$ *The key generation algorithm, takes as input L and sk, outputs sk_L as the attribute private key for L.*

Enc$(M,\ W,\ pk)$ *The encryption algorithm, takes as input a message M together with W, outputs $\mathcal{C}$ as the ciphertext.*

Dec$(W,\ \mathcal{C},\ sk_L)$ *The decryption algorithm, takes as input $\mathcal{C}$, W, and the attribute private key sk_L. If $R(L, W) = 1$, it outputs a plaintext M. Otherwise, it returns $\perp$.*

Definitions for KP-ABE and CP-ABE can be derived from the above unified definition.

1. *It is the definition for CP-ABE, if L is just a set of attributes (or, attribute list), and W denotes general ciphertext-policy.*

2. *It is the definition for KP-ABE, if L is defined as a general access structure (key-policy) and W is an attribute list.*

In CP-A^{2}BE, as explained, we consider how to obtain accountability for users. The definition for CP-A^{2}BE is almost the same with CP-ABE, except the decryption key $sk_{ID,L}$ is issued on attribute L with the user's identity ID, and the algorithm for tracing is required additionally. The Trace algorithm is defined as follows:

Trace. *This algorithm is used to trace a decryption key to its original holder. It takes as input a decryption key, and outputs an identity associated with this decryption key.*

The decryption key in our following constructions consists of two parts, namely, attribute private key and identity. The attribute authority issues an attribute private key only when the user's identity is what he alleged, and he is eligible to get the attributes requesting. Of course, it is the basic requirement to ensure that the user cannot change his attributes or identity information to get another valid attribute private key. In this work, for simplicity, it specifically assumes that the decryption key is well-formed when a user shares his key with others, which can be seen from the above definition for tracing algorithm. In fact, this assumption can be avoided because there already exists an approach proposed in [20] to get rid off the assumption of well-formed decryption key by introducing default attributes. In fact, this well-formed decryption key assumption has also been used in [18, 2] to reduce the trust of PKG in IBE. As mentioned in [18, 2], the user could still construct a malformed decryption key which, when used in conjunction with some other decryption process, is still able to decrypt ciphertexts. Therefore, we also need to consider the extreme case of black box which is able to decrypt the ciphertexts in practice, which is very similar to the technique of black-box traitor tracing [13]. As just mentioned, recently Goyal et al. [20] showed an approach on how to achieve black-box tracing from a scheme with the well-formed decryption key assumption. Based on this method, we can also achieve the black-box tracing by using the method as in [20].

2.1.2 The Construction

In this construction, the ciphertext-policy has the same fine-grained access structure as [12]. Next, we detail the access structure in [12]. Assume attributes in universe form a set $U = \{w_1, w_2, \cdots, w_n\}$. To encrypt a message, it specifies the ciphertext-policy $W = [W_1, W_2, \cdots, W_n]$, where W_i is equal to w_i or $*$. The notion of wildcard $*$ in the ciphtertext-policy means the value of "don't care". For example, let the ciphertext-policy $W = [1, 0, 1, *]$ for $n = 4$. This ciphertext-policy W means that the recipient who wants to decrypt must have the value 1 for W_1 and W_3, the value 0 for W_2, and any possible values for W_4. Therefore, if the receiver has an attribute private key for $[1, 0, 1, 0]$, he can decrypt the ciphertext because the first three values for W_1, W_2 and W_3 are equivalent to the corresponding values in ciphertext-policy. Moreover, the fourth value 0 in the private key satisfies the ciphertext-policy because $W_4 = *$. If an attribute private key is associated with the attribute list $[1, 1, 1, 0]$, this attribute private key will not match the ciphertext-policy since $W_2 \neq 0$. To be more generalized, given an attribute list $L = [L_1, L_2, \cdots, L_n]$ and a ciphertext-policy $W = [W_1, W_2, \cdots, W_n]$, we say that L matches W if for all $i \in [1, n]$, $W_i = L_i$ or $W_i = *$. In [12], each attribute can take two values 1 and 0. In our construction, we generalize the access structures such that each attribute can take multiple values. More formally, let $S_i = \{v_{i,1}, v_{i,2}, \cdots, v_{i,n_i}\}$ be a set of possible values for attribute w_i where n_i is the number of the possible values for w_i. Now, the attribute list L for a user is $L = [L_1, L_2, \cdots, L_n]$ where $L_i \in S_i$ for $1 \leq i \leq n$. The generalized ciphertext-policy W, then, is $W = [W_1, W_2, \cdots, W_n]$ where where W_i is equal to some value in S_i or $*$. The attribute list L matches the ciphertext-policy W (that is, $R(L, W) = 1$) if $W_i = L_i$ or $W_i = *$ for $1 \leq i \leq n$.

Main Idea In this system, each user is issued an attribute private key on $L \| ID$, where L denotes an attribute list and ID denotes the user's identity, respectively. When a message is to be encrypted under a ciphertext-policy W, a ciphertext is created with the underlying policy $W' = W \| *$. Users with attributes L satisfying $R(L, W) = 1$ can decrypt the ciphertext, even if their identities are different. This holds because the second part in W' is $*$, namely, "don't care" (This technique is used to keep the one-to-many property in ABE, even though different identities have been embedded in the attribute private keys for the same attributes).

In addition, the user cannot change his attribute-based private key on $L\|ID$ to $L^*\|ID^*$, where $(L, ID) \neq (L^*, ID^*)$. Therefore, the identity ID will be detected from the decryption key embedded in the illegal decryption device.

Before presenting the construction, we give a brief review on the property of pairings. Let $\mathbb{G}_1, \mathbb{G}_2$ be cyclic groups of prime order p, writing the group action multiplicatively. Let g be a generator of $\mathbb{G}_1$, and $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ be a map with the following properties: i) *Bilinearity* $\hat{e}(g_1^a, g_2^b) = \hat{e}(g_1, g_2)^{ab}$ for all $g_1, g_2 \in \mathbb{G}_1$, and $a, b \in_R \mathbb{Z}_p$; ii) *Non-degeneracy* There exist $g_1, g_2 \in \mathbb{G}_1$ such that $\hat{e}(g_1, g_2) \neq 1$, namely, the mapping does not send all pairs in $\mathbb{G}_1 \times \mathbb{G}_1$ to the identity in $\mathbb{G}_2$; iii) *Computability* There exists an efficient algorithm to compute $\hat{e}(g_1, g_2)$ for all $g_1, g_2 \in \mathbb{G}_1$. In this CP-A^{2}BE construction, we assume there are n attributes in universe. That is, let the universal attributes set be $U = \{\omega_1, \omega_2, \cdots, \omega_n\}$. Define S_i to be the multi-value set for each attribute ω_i in U. The CP-A^{2}BE construction is as follows:

Setup *To generate system parameters, a trusted authority selects random generators* $g, g_2, g_3, u_0, u_1, \ldots, u_n \in \mathbb{G}_1$*, picks a random* $\alpha \in \mathbb{Z}_p$*, and sets* $g_1 = g^\alpha$*. He also defines a cryptographic hash function* $H : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$*. The system parameter is* $param = (g, g_1, g_2, g_3, u_0, u_1, \cdots, u_n, H)$*, and the master secret key is* g_2^α*.*

KeyGen *To issue an attribute private key for a user with identity* ID *and an attribute list* $L = [L_1, L_2, \cdots, L_n]$*, the attribute authority picks up a random* $r \in \mathbb{Z}_p$ *and computes* $(d_0, d_1) = (g_2^\alpha (u_0^{ID} u_1^{H(L_1)} \cdots\ u_n^{H(L_n)}\ g_3)^r,\ g^r)$ *as the attribute private key. The validity of* $(d_0,\ d_1)$ *can be verified through the following equation:* $\hat{e}(d_0, g) = \hat{e}(g_1, g_2)\hat{e}(u_0^{ID} u_1^{H(L_1)} \cdots u_n^{H(L_n)} \cdot g_3, d_1)$*. Finally, the user retains the decryption key* $sk_{ID,L} = (d_0, d_1,\ d_2,\ d_3)$ *on decryption device, where* $d_2 = ID$ *and* $d_3 = L$*.*

Enc *To encrypt a message* $M \in \mathbb{G}_2$ *under ciphertext-policy* $W = [W_1, W_2,\ \cdots,\ W_n]$*, pick up a random value* $s \in \mathbb{Z}_p$ *and set* $C_0 = M\hat{e}(g_1, g_2)^s$*,* $C_1 = g^s$*,* $C_2 = (\prod_{W_i \neq *} u_i^{H(W_i)} \cdot g_3)^s$*,* $T_i = \{u_i^s\}_{W_i = *}$*,* $E = u_0^s$*. The ciphertext for* M *on* W *is* $C = (C_0, C_1, C_2, \{T_i\}_{W_i=*}, E)$*.*

Dec *To decrypt the ciphertext* $C = (C_0, C_1, C_2, \{T_i\}_{W_i=*}, E)$*, the recipient with identity* ID *and attribute list* L *first checks* W *to know whether* $R(L, W) = 1$*. If* $R(L, W) = 1$*, he proceeds as follows: Let* $sk_{ID,L} = (d_0, d_1, d_2, d_3)$ *be the decryption key deposited in decryption device, where* $d_2 = ID$ *and* $d_3 = L$*. He computes* $C_2' = C_2 \prod_{W_i=*} T_i^{H(L_i)} E^{d_2}$ *and decrypts with* $sk_{ID,L}$ *to get* $M = C_0\hat{e}(d_1, C_2')/\hat{e}(d_0, C_1)$*.*

Trace *Let* $sk_{ID,L} = (d_0, d_1, d_2, d_3)$ *be a valid decryption key in an illegal decryption device, where* $d_3 = [L_1, L_2,\ \cdots,\ L_n]$*. It means that* $\hat{e}(d_0, g) = \hat{e}(g_2, g_1)\ \hat{e}(u_0^{d_2} u_1^{H(L_1)} \cdots\ u_n^{H(L_n)}\ g_3,\ d_1)$*. Then, just reveal* d_2 *as the identity of the dishonest user who shares the decryption key.*

2.1.3 Security Analysis

We show the definitions of CDH/DBDH problems and assumptions in the bilinear groups before the security result.

CDH Problem. *The Computational Diffie-Hellman (CDH) problem is that, given* $g,\ g^x,\ g^y \in \mathbb{G}_1$ *for unknown random* $x, y \in \mathbb{Z}_p^*$*, to compute* g^{xy}*.* We say that the (t, ϵ)-CDH assumption holds in $\mathbb{G}_1$ if no t-time algorithm has the probability at least ϵ in solving the CDH problem for non-negligible ϵ.

DBDH Problem. *The Decision Bilinear Diffie-Hellman (DBDH) problem is that, given* $g,\ g^x,\ g^y,\ g^z \in \mathbb{G}_1$ *for unknown random* $x, y, z \in \mathbb{Z}_p^*$*,* $T \in \mathbb{G}_2$*, to decide if* $T = \hat{e}(g, g)^{xyz}$*.* We say that a polynomial-time adversary $\mathcal{A}$ has advantage ϵ in solving the DBDH problem in groups $(\mathbb{G}_1, \mathbb{G}_2)$ if $\mid Pr[\mathcal{A}(g, g^x, g^y, g^z, \hat{e}(g,g)^{xyz}) = 1] - Pr[\mathcal{A}(g, g^x, g^y, g^z, \hat{e}(g,g)^r) = 1] \mid \geq 2\epsilon$, where the probability is taken over the randomly chosen x, y, z, r and the random bits consumed by $\mathcal{A}$. (t, ϵ)-DBDH assumption holds in $(\mathbb{G}_1, \mathbb{G}_2)$ if no t-time algorithm has the probability at least ϵ in solving the DBDH problem for non-negligible ϵ.

We first describe the security definition of CP-ABE here. As defined in [4, 12, 21], the security requirement for CP-ABE is indistinguishability against chosen message attack (IND-CPA). Its formal definition is given based on the following IND-CPA game involving a simulator and an adversary $\mathcal{A}$: The simulator chooses a sufficiently large security parameter 1^λ, and runs Setup to get a master key sk and public key pk. The master key sk is kept secret and pk is sent to $\mathcal{A}$. $\mathcal{A}$ can perform a polynomially bounded number of queries for key generation on attribute list L. The simulator answers the queries by returning indistinguishable and correct attribute private keys. After these queries, $\mathcal{A}$ tries to distinguish a ciphertext is an encryption to which one of two adaptively chosen messages under some chosen ciphertext-policy. In concrete, the adversary outputs a challenge W^* with two messages M_0 and M_1, on which he wishes to be challenged. The simulator randomly chooses a bit $b \in \{0, 1\}$, computes the challenge ciphertext $\mathcal{C}=$ Enc(M_b, W^*, pk), and sends $\mathcal{C}$ to $\mathcal{A}$. $\mathcal{A}$ is allowed to issue more queries for attribute private keys. Finally, $\mathcal{A}$ outputs a guess bit b'. $\mathcal{A}$ wins the game if $b = b'$ and L that satisfies $R(L,W^*)$=1 has never been submitted to request an attribute private key. The advantage of $\mathcal{A}$ in Game IND-CPA is defined as the probability that $\mathcal{A}$ wins the game minus 1/2.

Definition 2. *A CP-ABE satisfies IND-CPA if no polynomial time adversary can win the above game with non-negligible probability.*

Another stronger security notion for CP-ABE is IND-CCA, in which the adversary is allowed to query decryption oracle. As there are many generic techniques [15, 27] to transform schemes with IND-CPA security to IND-CCA, we just prove the IND-CPA security of our constructions for simplicity. In this work, we use another weaker security model, called selective-IND-CPA. This model is analogous to the selective-ID model [5] utilized in IBE protocols. In selective-IND-CPA, the adversary should commit to the challenge W^* before Setup compared with the IND-CPA model.

Next, we define two security requirements for CP-A^{2}BE, that is, IND-CPA and Key Unforgeability. The security of IND-CPA is required since the CP-A^{2}BE is still one kind of CP-ABE. The definition of IND-CPA is almost the same as the corresponding one in CP-ABE, except that user's identity will be included in the attribute private key query. To trace the identity who shares the decryption key, security requirement of user accountability is defined additionally. User accountability means that if a user has a decryption key, he cannot output and share a new decryption key for a different identity. The user accountability is defined through the following game of Key Unforgeability. Notice that the security model of IND-CPA cannot imply Key Unforgeability. This is because in IND-CPA model, it only guarantees that the user is unable to decrypt a ciphertext if his attributes does not match the ciphertext-policy. The IND-CPA model does not consider the case when the user decrypts ciphertext with the same attribute list, but replaces the identity in the attribute private key with another one to avoid tracing. Therefore, to achieve user accountability, it should be ensured that the user can not change the identity inserted in his attribute private key. In this work, we also consider a weaker security notion called selective-key unforgeability. The formal definition for selective-Key Unforgeability is based on the following game involving an adversary $\mathcal{A}$: The adversary outputs the target identity ID^* before Setup. The simulator chooses a sufficiently large security parameter 1^λ, and run Setup to get a master secret key sk and public key pk. Retain sk and give pk to $\mathcal{A}$. $\mathcal{A}$ can perform a polynomially bounded number of queries to key generation oracle for private key on (ID, L). Finally, $\mathcal{A}$ outputs a decryption key sk_{ID^*,L^*} for identity ID^* on attributes L^*. The challenger runs a sanity check on sk_{ID^*,L^*} to ensure that it is well-formed. It aborts if the check fails. $\mathcal{A}$ wins the game if ID^* has not been submitted to key generation oracle. The advantage of $\mathcal{A}$ in Game selective-Key Unforgeability is defined as the probability that $\mathcal{A}$ wins the game. The CP-A^{2}BE is accountable if there is no adversary wins the above game with non-negligible probability.

Theorem 1. *The CP-A^{2}BE construction is secure in selective-IND-CPA and selective-Key Unforgeability models, under the DBDH and CDH assumptions, respectively.*

Proof. See Appendix A.

Actually, if the hash function H plays the role of random oracle during proof, the scheme can achieve the security of full Key Unforgeability. To get IND-CCA security from IND-CPA, we can use the most efficient transformations-Fujisaki-Okamoto technique [15], which adds only a little computation on the original CP-A^{2}BE scheme. Therefore, the resulted IND-CCA construction is very efficient.

2.2 The Strong CP-A^{2}BE

In CP-A^{2}BE, only the accountability for users is achieved. To further reduce the trust and get accountability for the attribute authority, the notion of Strong CP-A^{2}BE is given in this Section. To construct such a Strong CP-A^{2}BE scheme, the assumption that users in this system should have a higher level secret information, is required. As we explained, the secret in public key certificate functions as a higher level secret information in this work for simplicity. This assumption is reasonable because, before the user is issued attribute private key, the attribute authority should know he is the right user as alleged. To achieve the authentication, the user should demonstrate proof that he is the holder of a public key by using standard proof of knowledge [2]. The construction is based on the scheme in Section 2.1. The advantage of this method is that the tracing algorithm is no longer required for user accountability based on that the user will not leak his secret in certificate. However, another trace algorithm is required to detect the misbehavior of attribute authority. We detail the analysis later in this Section.

2.2.1 The Construction

Main Idea In Strong CP-A^{2}BE, each user should have a registered public key pk_u before requesting private key for attribute list L. The attribute authority issues an attribute private key on $L\|sk_u$ to the user with L and public key pk_u, where sk_u is the secret key corresponding to pk_u. The attribute authority can generate the attribute private key, without the information of sk_u. When a message is encrypted under a ciphertext-policy W, compute the ciphertext with respect to the underlying ciphertext-policy $W' = W\|*$. Any user with $L\|sk_u$ satisfying $R(L\|sk_u, W\|*) = 1$ (*i.e.* $R(L, W) = 1$), is able to decrypt the ciphertext regardless of the user's secret key sk_u. Moreover, the user can only decrypt the ciphertext with the attribute private key on $L\|sk_u$. In other words, the user cannot change his private key on $L\|sk_u$ to another one on $L^*\|sk_u^*$ such that $(L, sk_u) \neq (L^*, sk_u^*)$. As a result, the user will not share his decryption key as it will reveal his secret key in the certificate. The accountability for attribute authority can be achieved too because he does not know the user's secret key in the certificate. If the attribute authority generates and distributes another valid decryption on L on a randomly chosen sk_u, the misbehavior will be detected since only the attribute authority can generate such a decryption key on the random sk_u without corresponding registered public key certificate on pk_u.

Setup *This algorithm is the same as Section 2.1. The system parameter is param* $= (g, g_1, g_2, g_3, u_0, u_1, \cdots, u_n, H)$ *and the master key is* g_2^α.

KeyGen *To issue an attribute private key to a user with public key* $u = u_0^x$ *on an attribute list* $L = [L_1, L_2, \cdots, L_n]$*, the user proves that he is the holder of public key* u *by using proof of knowledge technique. If the proof passes, the attribute authority picks up a random* $r \in \mathbb{Z}_p$ *and computes* $d_0 = g_2^\alpha \cdot (uu_1^{H(L_1)} \cdots u_n^{H(L_n)} g_3)^r$, $d_1 = g^r$. *Then,* (d_0, d_1) *is sent to the user as the attribute private key. Finally, the user retains the decryption key* $sk_{u,L} = (d_0, d_1, d_2, d_3)$ *in decryption device, where* $d_2 = x$, $d_3 = [L_1, L_2, \cdots, L_n]$. *The correctness of* $sk_{u,L}$ *can be verified if* $\hat{e}(d_0, g) = \hat{e}(g_1, g_2)\hat{e}(u_0^{d_2} u_1^{H(L_1)} \cdots u_n^{H(L_n)} \cdot g_3, d_1)$.

Enc *To encrypt a message* $M \in G_2$ *under a ciphertext-policy* $W = [W_1, W_2, \cdots, W_n]$*, the algorithm proceeds the same as the one described in Section 2.1. Finally, it outputs the ciphertext as* $C = (C_0, C_1, C_2, \{T_i\}_{W_i = *}, E)$.

Dec *To decrypt the ciphertext* $C = (C_0, C_1, C_2, \{T_i\}_{W_i = *}, E)$*, the recipient with public key* u *and attribute list* L *can check* W *to know whether* $R(L, W) = 1$. *If* $R(L, W) = 1$*, he computes* $C_2' = C_2 \prod_{W_i = *} T_i^{H(L_i)} E^{d_2}$ *and decrypts the ciphertext by using* $sk_{u,L} = (d_0, d_1, d_2, d_3)$ *as* $M = C_0 \hat{e}(d_1, C_2') / \hat{e}(d_0, C_1)$.

The above construction can be proved to be a secure CP-A^{2}BE scheme, without algorithm of Trace. However, to achieve Strong CP-A^{2}BE, it is necessary to detect the misbehavior of the attribute authority. Therefore, the algorithm Trace will be given here to achieve accountability for the attribute authority.

Trace *Let* $sk_{u,L} = (d_0, d_1, d_2, d_3)$ *be a decryption key in an illegal decryption device, where* $d_3 =[L_1, L_2, \cdots, L_n]$. *It means that* $\hat{e}(d_0, g) = \hat{e}(g_2, g_1)\ \hat{e}(u_0^{d_2} u_1^{H(L_1)} \cdots\ u_n^{H(L_n)}\ g_3,\ d_1)$. *Then, check if* $u_0^{d_2}$ *is a valid public key. If not, it is the attribute authority who generates and distributes this decryption key.*

Note that in KeyGen algorithm, the user with public key u should prove he is the holder of this public key by proving the knowledge of x. The secret key x can be viewed as another attribute issued in the attribute private key though the attribute authority does not know the attribute. Therefore, the proof can be easily get from the proof in Section 2.1.

2.2.2 Security Analysis

The accountability for users is described through the following game Key Unforgeability. This game has some difference from the same game in Section 2.1. In that game in Section 2.1, key abuse problem can be prevented because identity of the user will be detected if he shares his decryption key. In Strong CP-A^{2}BE, the decryption key contains the attribute private key and secret key corresponding to the public key certificate. Assume the users do not leak the secret key in the certificate, the users are not willing to share his decryption key. Therefore, in the game of Key Unforgeability, we only need to guarantee that the user can not change the secret key in his decryption key. Since the public key certificate is required before issued attribute private key, the user has to prove he knows the secret key in certificate by using proof of knowledge technique. The adversary will try to output a decryption key with some secret key without relation to his own public key. We will also define a weaker security notion called selective-Key Unforgeability as in Section 2.1. The only difference is that, in the game of selective-Key Unforgeability, the adversary should output the key that he will use in the forged decryption key in advance.

In more details, the adversary outputs a value sk^*, corresponding to some unregistered public key pk^* that will be shared as a part of the decryption key. The simulator chooses a sufficiently large security parameter 1^λ, and runs Setup to get a master key sk and public key pk. The simulator retains the secret key sk and sends pk to $\mathcal{A}$. $\mathcal{A}$ can perform a polynomially bounded number of queries to key generation oracle for private key on attribute list L with valid proof of knowledge to a public key pk. Finally, $\mathcal{A}$ outputs a decryption key sk_{pk^*,L^*} on L^* and pk^*. $\mathcal{A}$ wins the game if sk_{pk^*,W^*} is a valid decryption key, and the public key pk^* has not been submitted to ask for an attribute private key (There is no requirement of certificate for this public key). The advantage of $\mathcal{A}$ in Game selective-Key Unforgeability is defined as the probability that $\mathcal{A}$ wins the game. A Strong CP-A^{2}BE achieves user accountability if no adversary wins the above game with non-negligible probability. One may argue in case that, after a user get an attribute private key on attributes L on pk^* (Here, to query attribute private key on pk^*, it should be a valid public key), the user forges and shares another valid decryption key on attributes L^* and pk^*. From the decryption key, we can get pk^* and trace its owner.

From the above two games, it can achieve the security of CP-A^{2}BE. To get the Strong CP-A^{2}BE, the game FindKey should be defined additionally. The accountability for attribute authority can be guaranteed based on the following game FindKey. This game is utilized to detect the misbehavior of attribute authority: The challenger runs Setup to get master key sk and public key pk. It gives pk, sk to the adversary $\mathcal{A}$ ($\mathcal{A}$ plays the role of semi-trused attribute authority in this game). Finally, $\mathcal{A}$ outputs a well-formed decryption key sk_{pk^*,W^*} on some attributes W^* with respect to pk^*. The advantage of $\mathcal{A}$ in Game Findkey is defined as the probability that $\mathcal{A}$ outputs a well-formed decryption key with respect to some user's valid public key pk^*. A Strong CP-A^{2}BE satisfies accountability for attribute authority if there is no adversary wins the above game with non-negligible probability. In case that the adversary outputs some decryption key with respect to an invalid public key, we define it loses the game according to the above definition. And, from this kind of forgery, we say that the attribute authority misbehaves. In the following construction, we assume the attribute authority uses the same group as the users' public keys in public key certificate.

Theorem 2. *The Strong CP-A^{2}BE construction is secure in* selective-IND-CPA *model, and has accountability for users and attribute authority, under the* DBDH *assumption and* CDH *assumption, respectively.*

Proof. We show how to prove that the scheme achieves user accountability under selective-Key Unforgeability model. First, the adversary outputs some value sk^* in private key it wants to share. Then, the simulator sets the public parameters and simulates the private key generation oracle. It is the same as the proof in Theorem 1. Notice that sk^* is viewed as corresponding position of ID^* in the proof of Theorem 1. The simulator only needs to simulate the key generation oracle. The challenge ciphertext oracle is not required here because the goal of adversary is to output a private key for any attributes with some randomly chosen value sk^*. The adversary could ask private key for any L on a valid public key pk_u. During key generation queries, the user will not give simulator the secret key sk_u. However, as mentioned in KeyGen, the user has to use some proof of knowledge to show he is the holder of public key pk_u. By using the knowledge extractor, the simulator can extract sk_u and simulate key generation oracle in the same way as Theorem 1, where ID is viewed as sk_u here. The accountability of attribute authority can be easily proved from the definition of game FindKey. The decryption key includes the secret key sk_u from pk_u. To avoid detection, the attribute authority has to output a valid decryption key on some user's public key. This implies the attribute authority has to compute the user's secret key corresponding to some public key certificate. Based on the security of public key system, the above CP-A^{2}BE has accountability for the attribute authority.

3 Implementation for Access Control

We show how to apply the Strong CP-A^{2}BE to achieve abuse free access control in this Section. For CP-A^{2}BE, it can be implemented in a similar way. Initially, assume some data has been stored on a remote data server. The owner may wish to allow users only with specific privileges to access the data. In the ABE-based access control system, these privileges can be categorized via the users' attributes. There are three entities in the system: A public key certificate center, an attribute authority and users. First, individual users in this system should have a registered public key from the public key certificate center. The secret key in the certificate functions as a higher level private information compared to the attribute-based system. With such a certificate, the user can prove to the attribute authority that he is the owner of the public key in the certificate, using proof of knowledge technique. If a user is eligible to some attributes, he will be issued a private key for these attributes from the attribute authority. This can be achieved by the KeyGen algorithm in the above construction. If a user, after uploading his data, wants to enforce some access control policy to the data, he can simply apply the algorithm of Enc in Strong CP-A^{2}BE. As a result, users with the attributes that matches the policy defined in ciphertext, can decrypt and read the data. The user would not share his decryption key with others because his secret key, which is implied in his certificate, is embedded in the decryption key. Moreover, the user cannot compute a new decryption key with a different invalid public key (This property can be achieved based on the security game of FindKey). Thus, if a valid decryption key with respect to some invalid public key was found, we can tell it is the misbehavior of attribute authority. Otherwise, if the public key corresponding to the decryption key is valid, then, it is the user with this public key shares his decryption key.

The underlying scheme without identity and tracing algorithm is an improvement to the CP-ABE construction in [12]. We discuss and compare the CP-ABE scheme [12] with ours in terms of key size and computation overhead. In our scheme, the attribute private key consists of only two group elements, which is constant with the number of user's attribute. However, in [12], there are $2n$ group elements in the user's attribute private key. To issue an attribute private key, our schemes needs two exponentiations in group $\mathbb{G}_1$ for the attribute authority to generate an attribute private key for user. However, $2n$ exponentiations in $\mathbb{G}_1$ are required in the key generation algorithms of [12], where n is the number of attributes in universe. The computational cost [12] is linear with the number of universal attributes. In our construction, ciphertext consists of $3 + k$ group elements and the encryption algorithm needs $3+k$ exponentiations in $\mathbb{G}_1$, where k denotes the number of wildcards in ciphertext. Decryption requires two pairing computation. However, in [12], there are $3n$ group elements in ciphertext and encryption algorithm performs $n + 1$ exponentiations in group $\mathbb{G}_1$. For decryption, it performs $n + 1$ pairings. Therefore, overall, the underlying CP-ABE construction is more efficient than [12].

4 Extensions

In this Section, we show how to use the technique in Section 2.2 to solve some open problems existed in accountable IBE. Note that accountable ABE cannot be constructed directly from accountable IBE [2, 18, 20] since only one private key will be issued for each identity in IBE, whereas in ABE, many private keys will be issued for each attribute to different users. We also propose another new notion called conditional IBE. In conditional IBE, an encryptor can specify the ciphertext such that it can only be decrypted by the user with identity ID and additional several conditions. This notion is different from ordinary IBE in that the encryptor can also add requirements in the ciphertext, such as specific attributes, not only the identity information. Security of the following two constructions can be easily derived based on the above results in Section 2.

4.1 Accountable IBE

Identity-based cryptosystem [30] is a public key cryptosystem where the public key can be an arbitrary string such as an email address. It was proposed to simplify key management procedures of certificate-based public key infrastructures. In identity-based cryptosystem, a private key generator (PKG) uses a master secret key to issue private keys to identities that request them. Boneh and Franklin [7] proposed the first practical IBE scheme based on pairing, which is provably secure in the random oracle model. The first IBE without random oracles was proposed by Waters [31]. Later, this construction was further generalized and analyzed in [11]. To reduce the trust of PKG in IBE, Goyal [18] proposed another notion called accountable IBE and strengthened by [2, 20]. These IBE can only be used to encrypt a message to a single user. To encrypt a message to a group, the notion of IBE with wildcard [1] was proposed. In this kind of IBE, one can encrypt a message for a group of users with some common properties, in which the different parts are viewed as "don't care" components. Recently, Goyal [18] proposed the first method on how to reduce the trust of PKG in IBE. Though later construction with black-box accountability based on DBDH assumption was proposed [20], the scheme is very inefficient.

An open question was left in [18]: How to construct a more efficient IBE with minimum trust to PKG based on standard assumption such as DBDH assumption? In this Section, we solve this open problem by using the technique in the above Sections. We combine the user's public key certificate with IBE system, while keeping all the properties of IBE. Actually, this approach is not new and has been used in [16] to construct identity-based cryptosystem with special properties.

Another open problem can also be solved by using our method: In case of some user loses private key for its identity ID, then, how to achieve accountability? The papers [18, 20] cannot solve this problem because they require the user's decryption key to detect the misbehavior of

PKG. In our approach, even with only one private key for ID detected from a pirate device, this kind of key abuse can be found because the value in the forged decryption key is different from the public key in the public key certificate for ID. There are five algorithms in accountable IBE, that is, setup algorithm Setup, key generation algorithm KeyGen, encryption algorithm Enc, decryption algorithm Dec, and tracing algorithm Trace. The algorithm of Setup is almost the same as in Section 2.2. The system parameter is $param = (g, g_1, g_2, g_3, u_0, u_1, H)$ and the master key is g_2^{α}. Let the identity be ID for the user with public key $u = u_0^x$. To extract a private key from PKG, the user should show his identity and prove he is the holder of the public key u by using the proof of knowledge. If the proof passes, the attribute authority picks up a random $r \in \mathbb{Z}_p$ and computes $(d_0, d_1) = (g_2^{\alpha} \cdot (uu_1^{ID} g_3)^r,\ g^r)$. Finally, the user retains the decryption key $sk_{ID} = (d_0, d_1, d_2)$, where $d_2 = x$. The correctness of (d_0, d_1, d_2) can be verified by checking if the following equation holds: $\hat{e}(d_0, g) = \hat{e}(g_1, g_2)\hat{e}(u_0^{d_2} u_1^{ID} g_3, d_1)$. To encrypt a message $M \in \mathbb{G}_2$ for user ID, pick up a random value $s \in \mathbb{Z}_p$, set $C_0 = M\hat{e}(g_1, g_2)^s$, $C_1 = g^s$, $C_2 = (u_1^{ID} g_3)^s$, and $E = u_0^s$. The cipertext is $C = (C_0, C_1, C_2, E)$. In Dec algorithm, after receiving the ciphertext $C = (C_0, C_1, C_2, E)$, the user ID with public key u proceeds as follows: Let $sk_{ID} = (d_0, d_1, d_2)$ be his decryption key. The user computes $C_2' = C_2 E^x$ and decrypts the ciphertext as $M = C_0 \frac{\hat{e}(d_1, C_2')}{\hat{e}(d_0, C_1)}$. In algorithm of Trace, it takes as input a valid decryption key sk_{ID} =(d_0, d_1, d_2), which means $\hat{e}(d_0, g) = \hat{e}(g_1, g_2)\hat{e}(u_0^{d_2} u_1^{ID} g_3, d_1)$. It is the PKG who forges and distributes the decryption key for the user with identity ID if $u_0^{d_2} \neq u$ if $u_0^{d_2}$ is equal to u in public key certificate for ID.

We describe briefly why this technique can prevent PKG from generating private key and decrypting ciphertext on behalf of users. If PKG outputs a forged valid decryption key for ID, another value $u_0^{x'}$ will be inserted in the decryption key. From the public certificate, we know the public key u_0^x and identity ID is connected to the same user. Therefore, from these two valid decryption keys, it can tell that PKG forges the decryption key for the user with identity ID. In our system, even if the user with identity ID has not requested private key, we still can tell PKG behaves illegally because there is no certificate for public key u' and ID from the forged decryption key for ID. However, the scheme of [18] can not prevent such forgery because two different decryption keys for the same identity ID are required to decide if it is the PKG who forges the detected decryption key.

4.2 Conditional IBE

Consider the following scenario: One wants to encrypt a message to some identity ID. He also wants to ensure that the user can only decrypt if he not only has identity ID, but also satisfies some additional conditions. For example, the ciphertext can be created for ID with additional attributes "Ph.D degree" and "Staff in University A". However, in traditional IBE, a message can only be encrypted to some user with ID, without other conditions. From the above CP-A^{2}BE scheme, such conditional IBE can be constructed as follows.

The Setup and KeyGen algorithms, including the definition for ciphertext-policy, are the same as the scheme in Section 2.1. To encrypt a message $M \in \mathbb{G}_2$ for identity ID with ciphertext-policy $W = [W_1, W_2, \cdots, W_n]$, pick up a random value $s \in \mathbb{Z}_p$ and set $C_0 = M\hat{e}(g_1, g_2)^s$, $C_1 = g^s$, $C_2 = (\prod_{W_i \neq *} u_0^{ID} u_i^{H(W_i)} g_3)^s$, $T_i = \{u_i^s\}_{W_i = *}$. The cipertext is $C = (C_0, C_1, C_2, \{T_i\}_{W_i = *})$. To decrypt the ciphertext $C = (C_0, C_1, C_2, \{T_i\}_{W_i = *})$, the user with identity ID and attribute list $L = [L_1, L_2, \cdots, L_n]$ can check W to know whether $R(L, W) =$

1. If $R(L, W) = 1$, the user computes $C_2' = C_2 \prod_{W_i=*} T_i^{H(L_i)}$ and decrypts the ciphertext as $M = C_0 \frac{\hat{e}(d_1, C_2')}{\hat{e}(d_0, C_1)}$ with his key $sk_{ID,L} = (d_0, d_1)$.

5 Conclusion

In this paper, we discussed the problem of key abuse existed in access control that is based on CP-ABE. Two kinds of accountability are considered in this work: The accountability for users and accountability for the semi-trusted attribute authority. First, we showed how to construct CP-A^{2}BE to achieve accountability for users, by inserting user's specific information into the attribute private keys. To further obtain accountability for both users and the semi-trusted attribute authority, we proposed and formulated the notion of Strong CP-A^{2}BE by letting the attribute private key contains the user's secret unknown to the attribute authority. A Strong CP-A^{2}BE scheme was also constructed based on the assumption that each user has registered a public key. The key point to these constructions is that the user's specific information or secret could be viewed as another default attribute. We also solved some open problems in accountable IBE and proposed another notion of conditional IBE scheme, as the extensions of the new techniques in the constructions of (Strong) CP-A^{2}BE.

References

1. Michel Abdalla, Dario Catalano, Alexander W. Dent, John Malone-Lee, Gregory Neven, and Nigel P. Smart. *Identity-Based Encryption Gone Wild.* ICALP'06, LNCS 4052, pp. 300-311, Springer, 2006.
2. Man Ho Au, Qiong Huang, Joseph K. Liu, Willy Susilo, Duncan S. Wong, and Guomin Yang. *Traceable and Retrievable Identity-Based Encryption.* ACNS'08, LNCS 5037, pp. 94-110, Springer, 2008
3. Joonsang Baek, Willy Susilo, and Jianying Zhou. *New Constructions of Fuzzy Identity-Based Encryption.* ASIACCS'07, pp. 368-370, ACM, 2007.
4. John Bethencourt, Amit Sahai, and Brent Waters. *Ciphertext-Policy Attribute-Based Encryption.* IEEE Symposium on Security and Privacy'07, pp. 321-334, IEEE, 2007.
5. Dan Boneh and Xavier Boyen. *Efficient Selective-ID Secure Identity Based Encryption Without Random Oracles.* EUROCRYPT'04, LNCS 3027, pp. 223-238, Springer, 2004.
6. Dan Boneh, Ran Canetti, Shai Halevi, and Jonathan Katz. *Chosen-Ciphertext Security from Identity-Based Encryption.* SIAM J. Comput. 36(5): 1301-1328, 2007.
7. Dan Boneh and M. Franklin. *Identity-Based Encryption from The Weil Pairing*, Crypto'01, LNCS 2139, pp.213-229, Springer-Verlag, 2001.
8. Dan Boneh and Brent Waters. *Conjunctive, Subset, and Range Queries on Encrypted Data.* TCC'07. LNCS 4392, pp. 535-554. Springer, 2007.
9. Xavier Boyen, Qixiang Mei, and Brent Waters. *Direct Chosen Ciphertext Security from Identity-Based Techniques.* CCS'05, pp. 320-329, ACM press, 2005. Full version at http://eprint.iacr.org/2005/288.
10. Melissa Chase. *Multi-Authority Attribute Based Encryption.* TCC'07, LNCS 4392, pp. 515-534, Springer, 2007.
11. Sanjit Chatterjee and Palash Sarkar. *Trading Time for Space: Towards an Efficient IBE Scheme with Short(er) Public Parameters in the Standard Model*, ICISC'05, LNCS 3935, pp. 424-440, 2005.
12. Ling Cheung and Calvin Newport. *Provably Secure Ciphertext Policy ABE.* In CCS'07, Proceedings of the 14th ACM conference on Computer and communications security, pages 456-465, ACM, 2007.
13. Benny Chor, Amos Fiat, and Moni Naor. *Tracing Traitor.* CRYPTO'94, LNCS 839, pp. 257-270, Springer, 1994.
14. Paul Feldman. *A Practical Scheme for Non-Interactive Verifiable Secret Sharing.* In Proc. 28th FOCS, pp. 427-437, 1987.
15. Eiichiro Fujisaki and Tatsuaki Okamoto. *Secure Integration of Asymmetric and Symmetric Encryption Schemes.* CRYPTO'99, LNCS 1666, pp. 537-554, Springer, 1999.
16. David Galindo, Javier Herranz, and Eike Kiltz. *On the Generic Construction of Identity-Based Signatures with Additional Properties.* ASIACRYPT'06, LNCS 4284, pp. 178-193, Springer, 2006.

17. Craig Gentry. *Practical Identity-based Encryption without Random Oracles.* EUROCRYPT'06. LNCS, vol. 4004, pp. 445-464. Springer, 2006.
18. Vipul Goyal. *Reducing Trust in the PKG in Identity Based Cryptosystems.* CRYPTO'07, LNCS 4622, pp. 430-447, 2007. Extension is available at http://eprint.iacr.org/2007/368.
19. Vipul Goyal, Abhishek Jain, Omkant Pandey, and Amit Sahai. *Bounded Ciphertext Policy Attribute Based Encryption.* ICALP'08. LNCS 5126, pp. 579-591, 2008.
20. Vipul Goyal, Steve Lu, Amit Sahai, and Brent Waters. *Black-Box Accountable Authority Identity-Based Encryption.* CCS'08, pp. 427-436, ACM, 2008.
21. Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. *Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data.* CCS'06, pp. 89-98, ACM, 2006.
22. M. Jason Hinek, Shaoquan Jiang, Reihaneh Safavi-Naini, and Siamak Fayyaz Shahandashti. *Attribute-Based Encryption with Key Cloning Protection.* Available at http://eprint.iacr.org/2008/478.
23. Apu Kapadia, Patrick P. Tsang, and Sean W. Smith. *Attribute-based Publishing with Hidden Credentials and Hidden Policies.* In Proc. of Network and Distributed System Security Symposium (NDSS), pp. 179-192, 2007.
24. Jonathan Katz, Amit Sahai, and Brent Waters. *Predicate Encryption Supporting Disjunctions, Polynomial Equations, and Inner Products.* EUROCRYPT'08, LNCS 4965, pp. 146-162, Springer, 2008.
25. Takashi Nishide, Kazuki Yoneyama, and Kazuo Ohta. *ABE with Partially Hidden Encryptor-Specified Access Structure.* ACNS'08, LNCS 5037, pp. 111-129, Springer, 2008.
26. Rafail Ostrovsky, Amit Sahai, and Brent Waters. *Attribute-based Encryption with Non-Monotonic Access Structures.* CCS'07, pp. 195-203, ACM, 2007.
27. Amit Sahai. *Non-Malleable Non-Interactive Zero Knowledge and Adaptive Chosen Ciphertext Security.* IEEE Symp. on Foundations of Computer Science, 1999.
28. Amit Sahai and Brent Waters. *Fuzzy Identity-Based Encryption.* EUROCRYPT'05, LNCS 3494, pp. 457-473, Springer, 2005.
29. Adi Shamir. *How to Share a Secret.* vol. 22, pp. 612-613, ACM, 1979.
30. Adi Shamir. *Identity-Based Cryptosystems and Signature Schemes.* CRYPTO'84, LNCS 196, pp. 47-53, Springer, 1984.
31. Brent Waters. *Efficient Identity-Based Encryption Without Random Oracles.* EUROCRYPT'05, LNCS 3494, pp. 114-127, Springer, 2005.

Appendix A: Proof of Theorem 1

The proof of Theorem 1 can be derived from the following two Lemmas.

Lemma 1. *The CP-A^{2}BE is* selective-IND-CPA *secure under the DBDH assumption.*

Proof. Assume that an attacker $\mathcal{A}$ breaks selective-IND-CPA with probability greater than ϵ within time t making q_d private key extraction queries. We show that using $\mathcal{A}$, one can construct a DBDH attacker $\mathcal{A}'$ with almost the same probability with ϵ. First, $\mathcal{A}$ outputs the target ciphertext-policy $W^*{=}(W_1^*, \cdots, W_n^*)$. Suppose that $\mathcal{A}'$ is given $(g, \hat{e}, \mathbb{G}_1, \mathbb{G}_2,\ A = g^x,\ B = g^y,\ C = g^z,\ T)$, where T is either $\hat{e}(g,g)^{xyz}$ or $\hat{e}(g,g)^{\gamma}$ for random $\gamma \in \mathbb{Z}_p$, as an instance of the DBDH problem. By ϵ' and t', we denote the winning probability and running time of $\mathcal{A}'$, respectively. $\mathcal{A}'$ can simulate the challenger's execution of each phase of selective-IND-CPA game for $\mathcal{A}$ as follows: $\mathcal{A}'$ sets $g_1 = g^x$ and $g_2 = g^y$. For $1 \le i \le n$ and $W_i^* \neq *$, let $u_i = g_1^{a_i} g^{b_i}$ by choosing $a_i, b_i \in Z_p^*$. For $1 \le i \le n$ and $W_i^* = *$, $u_i = g^{b_i}$ by choosing $b_i \in Z_p^*$. Then, choose a random $b_0 \in Z_p^*$ and let $u_0 = g^{b_0}$. Assign $g_3 = g_1^{-\sum_{1 \le i \le n \wedge W_i^* \neq *} a_i H(W_i^*)} g^{b'}$. The system parameters $para{=}\ (g, g_1, g_2, g_3, (u_i)_{0 \le i \le n})$ are sent to $\mathcal{A}$.

$\mathcal{A}'$ answers $\mathcal{A}$'s key generation queries as follows. Upon receiving a key generation query for $L{=}(L_1, \cdots, L_n)$ with respect to ID, $\mathcal{A}'$ checks if $R(L, W^*) = 1$. If $R(L, W^*) = 1$, $\mathcal{A}'$ aborts. Otherwise, $\mathcal{A}'$ chooses $r = (-y)(\sum_{1 \le i \le n} a_i H(L_i) - \sum_{W_i^* \neq *} a_i H(W_i^*))^{-1} + r'$. Let

$R = \sum_{1\leq i\leq n} a_i H(L_i) - \sum_{W_i^* \neq *} a_i H(W_i^*)$ and $R' = (-b_0 ID - \sum_{1\leq i\leq n} b_i H(L_i) + b')R^{-1}$. It outputs the simulated private key as
$sk_{ID,L}{=}(a_0, a_1){=}((u_0^{ID} u_1^{H(L_1)} \cdots u_n^{H(L_n)} g_3)^{r'} g_2^{R'}, g_2^{\frac{-1}{R}} g^{r'})$. We need to check if $a_0{=}g_2^x\ (u_0^{ID}\ u_1^{H(L_1)}\ u_n^{H(L_n)}\ g_3)^r$ and $a_1 = g^r$. Since the choose of R' and r, we have $g^r{=}g^{\frac{-y}{R}+r'}\ {=}g_2^{\frac{-1}{R}} g^{r'}$, and

$$\begin{aligned}
g_2^x(u_0^{ID} u_1^{H(L_1)} \cdots u_n^{H(L_n)} \cdot g_3)^r &= g_2^x(g_1^R g^{b_0 ID + \sum_{1\leq i\leq n} b_i H(L_i) + b'})^r \\
&= g_2^x(g_1^R g^{b_0 ID + \sum_{1\leq i\leq n} b_i H(L_i) + b'})^{\frac{-y}{R}+r'} \\
&= g_2^x(g_1^R g^{b_0 ID + \sum_{1\leq i\leq n} b_i H(L_i) + b'})^{r'} g^{-xy} g_2^{R'} \\
&= (g_1^R g^{b_0 ID + \sum_{1\leq i\leq n} b_i H(L_i) + b'})^{r'} g_2^{R'} \\
&= (u_0^{ID} u_1^{H(L_1)} \cdots u_n^{H(L_n)} g_3)^{r'} g_2^{R'}
\end{aligned}$$

After these queries for attribute private keys, $\mathcal{A}$ outputs two equal length messages M_0, M_1, identity ID^*, and the challenge ciphertext-policy $W^*{=}(W_1^*, \cdots, W_n^*)$. The challenger chooses randomly $b \in \{0,1\}$, and outputs the ciphertext as $(TM_b,\ C,\ C^{\sum_{W_i^* \neq *} b_i H(W_i^*) + b'}$, $\{C^{b_i}\}_{W_i^* *}$, $C^{b_0})$. Correctness of the ciphertext could be verified as follows: If $T = \hat{e}(g,g)^{xyz}$, $(TM_b, C, C^{\sum_{1\leq i\leq n, W_i^* \neq *} b_i H(W_i^*) + b'}$, $\{C^{b_i}\}_{W_i^* *}$, $C^{b_0}){=}(e(g_1,g_2)^s M_b$, g^s, $C_2{=}(\prod_{i=1, W_i^* \neq *}^n u_i^{H(W_i^*)} \cdot g_3)^s$, $T_i = \{u_i^s\}_{W_i = *}$, $u_0^s)$ by just letting $s = z$. Therefore, the simulated challenge ciphertext is correct when $T = \hat{e}(g,g)^{xyz}$.

$\mathcal{A}$ can still query key generation. $\mathcal{A}'$ answers key generation queries as above. Finally, $\mathcal{A}$ outputs a bit b'. Then, $\mathcal{A}'$ also outputs b' as the answer to the DBDH problem. For the simulation to complete without aborting. It is easy to verify that we can get the probability of breaking the DBDH problem as $\epsilon' \approx \epsilon$ if the adversary successes with probability ϵ.

Lemma 2. *The CP-A^{2}BE is selective-Key Unforgeability under the CDH assumption.*

Proof. Assume that an attacker $\mathcal{A}$ breaks selective-Key Unforgeability with probability greater than ϵ within time t making q_d private key generation queries. We show that using $\mathcal{A}$, one can break the CDH problem by constructing another attacker $\mathcal{A}'$ with approximately the same success probability. First, $\mathcal{A}$ outputs the target identity ID^*. Suppose that $\mathcal{A}'$ is given $g, \hat{e}, \mathbb{G}_1, \mathbb{G}_2$, $A = g^x$, $B = g^y$ and asked to compute g^{xy}. $\mathcal{A}'$ can simulate the challenger's execution of each phase for $\mathcal{A}$ as follows: $\mathcal{A}'$ sets $g_1 = A$ and $g_2 = B$. It chooses $r_0, r_0', r_1, r_2, \cdots, r_n \in \mathbb{Z}_p^*$. Let $u_0 = A^{r_0}$ and $g_3 = A^{-ID^* r_0} g^{r_0'}$. For $1 \leq i \leq n$, let $u_i = g^{r_i}$. The system parameters $para{=}\ (g, g_1, g_2, g_3, u_0, (u_i)_{1\leq i\leq n})$ are sent to $\mathcal{A}$. $\mathcal{A}'$ answers $\mathcal{A}$'s key generation queries as Lemma 1. Upon receiving a key generation query for ID with attributes $L = [L_1, L_2, \cdots, L_n]$, $\mathcal{A}'$ chooses $r' \in \mathbb{Z}_p^*$ and lets $r = \frac{y}{(ID^* - ID) r_0} + r'$. The private key can be simulated private key as $(g_1^{r_0(ID - ID^*)} g^{\sum_{i=1}^n r_i H(L_i)})^{r'} g_2^{r_0' + \sum_{i=1}^n r_i H(L_i)/(ID - ID^*) r_0}$, $g_2^{\frac{1}{(ID-ID^*) r_0}} g^{r'})$. The correctness can be verified as the same way in Lemma 1. Finally, $\mathcal{A}$ outputs a forged decryption key $sk_{ID^*, L^*} = (d_0, d_1, d_2, d_3)$ that $\mathcal{A}$ will share for attribute list $d_3 = L^* = [L_1^*, L_2^*, \cdots, L_n^*]$ on identity $d_2 = ID^*$. Because the decryption is valid and well-formed, then, we have $d_0 = g_2^x(u_0^{d_2^*} u_1^{H(L_1^*)} \cdots u_n^{H(L_n^*)} \cdot g_3)^r$ and $d_1 = g^r$ for some r. Since the simulation of public parameters in setup, we have that $d_0 = g_2^x(g^{r_0'} g^{\sum_{i=1}^n r_i H(L_i^*)})^r$. From this observation, $\mathcal{A}'$ can compute $g_2^x = d_0/(d_1)^{r_0' + \sum_{i=1}^n r_i H(L_i^*)}$ and output it as the solution to the CDH problem.