Noname manuscript No.
(will be inserted by the editor)

Energy-efficient Secure Outsourcing Decryption of Attribute Based Encryption for Mobile Device in Cloud Computation

Jindan Zhang · Baocang Wang · Fatos Xhafa · Xu An Wang · Cong Li

Received: date / Accepted: date

Abstract In this paper, we propose two new ways for efficient secure outsourcing the the decryption of key-policy attribute based encryption ($KP-ABE$) with energy efficiency. The first way base on an observation about the permutation property of the access structure of the attribute based encryption schemes. We propose a high efficient way for outsourcing the decryption for KP-ABE based on this observation, and it can be used for mobile devices for its high efficiency, but it can only be used for the ABE schemes having tree-like access structure and the self-enclosed system. The second way base on another observation that almost all the previous work on outsourcing the decryption of $KP-ABE$ cares little about the ciphertext length. Almost all the previous schemes have linear length ciphertext with the attributes or the policy. But we know that transferring so long ciphertexts via wireless network for mobile phone can easily run out the energy of the battery, which hesitates the adaption of these solutions in actual scenarios. Thus we propose another new scheme to outsource the decryption of ABE but with constant-size ciphertexts, which can achieve high energy efficiency. Furthermore, we propose a new very efficient way to secure outsource the decryptor's secret key to the cloud, which cost only one modular exponentiation. We roughly evaluate the efficiency of our proposals and the results show that our proposals are practical.

Jindan Zhang
State Key Laboratory of Integrated Service Networks, Xidian University, Xi'an 710071, China; Guangxi Key Laboratory of Cryptography and Information Security, Guilin University of Electronic Technology, P. R. China; Xianyang Vocational Technique College, P. R. China
E-mail: 69957106@qq.com

Baocang Wang
State Key Laboratory of Integrated Service Networks, Xidian University, Xi'an 710071, China
E-mail: bcwang79@aliyun.com

Fatos Xhafa
Department of Computer Science, Universitat Politcnica de Catalunya, Spain
E-mail: fatos@cs.upc.edu

Xu An Wang
Key Laboratory of Information and Network Security, Engineering University of Chinese Armed Police Force, P. R. China; Guangxi Key Laboratory of Cryptography and Information Security, Guilin University of Electronic Technology, P. R. China
E-mail: wangxazjd@163.com

Cong Li
Key Laboratory of Information and Network Security, Engineering University of Chinese Armed Police Force, P. R. China

1 Introduction

In these days, cloud computation is a very hot research topic for its promising properties of cheap management cost for users, any where/any time access, and very scalable software and hard ware investigation. However, before adapting cloud computation, data owners should ensure their data shall be secure and well protected. Attribute based encryption is a very promising technique to store data owner's data for flexible secure access controlling on the ciphertexts. However, attribute based encryption is a public key primitive and use tools needing huge computation cost like bilinear pairings. Thus it is desired to outsource this huge cost to the cloud. In this paper, we concentrate on one scenario of outsouring for cloud computation: how to securely outsourcing the decryption of attribute based encryption to the cloud,

which has important application for mobile users. Consider the following scenario:

"Data owner Alice outsource her data sets to the cloud, before outsourcing her data to the cloud, she first encrypt her data by using attribute based encryption such as KP-ABE and outsource the ciphertexts to the cloud. Data user Bob want to use the data items for which his credential can decrypt, but he is on a vacation and can only use mobile phone. Thus if he decrypt the downloading ciphertexts directly, his mobile phone can not implement this for the huge computation load. Thus he needs some mechanism to outsouce the decrypting to the cloud, this is called the mechanism of outsourcing decryption of attribute based encryption to the cloud. Another important issue is the length of the ciphertexts, if the length is very long outsoucing the ciphertexts to the cloud will cost lots of energy for the mobile phone, which will sharply shorten the living time of the mobile phone and this is intolerable. Thus we also care about the length of ciphertexts when outsourcing the decryption of attribute based encryption to the cloud."

Here we review some previous work on this topic. Green et al. [14] in 2011 proposed a very efficient way for outsourcing decryption of attribute based encryption, which is a wonderful result for adapting ABE for mobile equipments. However, this work care little about the verifiability for the computation results, the cloud can cheat the data users without any noticing. Therefore, Lai et al. [15] proposed a new way to outsource decryption of ABE with verifiability, but their scheme is not very efficient. Li et al. [19] also proposed a way to outsource encryption/key generation/decryption of ABE with checkability, but their scheme need often interaction with the PKG. Recently, Qin et al. [22] proposed an very efficient way to outsource decryption of ABE with verifiability.

1.1 Our Contribution

In this paper, we propose two new ways for efficient secure outsourcing the the decryption of key-policy attribute based encryption ($KP-ABE$) with energy efficiency. The first way base on an observation about the permutation property of the access structure of the attribute based encryption schemes, that is, the sequence of attributes are tightly associated with the access structure. If the sequence of attributes have been permutated, the access structure should be changed too. From the permutated sequence of attributes and the changed access structure, the adversary can not easily deduce the original sequence of attributes and the original access structure if there are many attributes like 100 ones in the access structure. We propose a high efficient way for outsourcing the decryption for KP-ABE based on this observation, and it can be used for mobile devices for its high efficiency, but it has its own restriction: it can only be used for the ABE schemes having tree-like access structure while not supporting LSSS structure, and also it can only be used for self-enclosed system. The second way base on another observation that almost all the previous work on outsourcing the decryption of $KP-ABE$ cares little about the ciphertext length. Almost all the previous schemes have linear length ciphertext with the attributes or the policy. But we know that transferring so long ciphertexts via wireless network for mobile phone can easily run out the energy of the battery, which hesitates the adaption of these solutions in actual scenarios. Thus we propose another new scheme to outsource the decryption of ABE but with constant-size ciphertexts, which can achieve high energy efficiency. Furthermore, we propose a new very efficient way to secure outsource the decryptor's secret key to the cloud, which cost only one modular exponentiation. We roughly evaluate the efficiency of our proposals, compared with the pervious work [14, 15, 19, 22] on outsourcing the decryption of ABE, our work can achieve high energy efficiency for the client's mobile devices. Thus our proposals are practical. This paper is based on our previous work [28] but with significant extension, concretely we add the first proposal which is not considered in our previous work, formally prove the security of the second proposal, and evaluate the performance of the second proposal etc.

1.2 Related Work

In 2005, Sahai and Waters [23] first proposed the concept of attribute based encryption and give a concrete such construction, which is an extension of fuzzy identity based encryption they also first proposed. In 2006, according to how to embed the access control in the algorithm, Goyal et al. [13] categorized ABE as the key policy ABE and the ciphertext policy ABE, and they also gave the first concrete key policy ABE based on bilinear pairings, this scheme can be proved selective secure in the standard model. Since then, many research results have been achieved. Waters in 2009 [24]first proposed the dual system encryption framework and later [16,18,20] use this technique to design several fully secure ABE schemes. Recently, Attrapadung [1] give a interesting framework to transform selective ABE to fully secure ABE. Chase et al. in 2007 [5] proposed the concept of ABE with multi-authority and [6, 17] continues this line of research. Attrapadung et al. [2–4,25,26] proposed serveral variants of ABE with interesting proper-

ties such as dual-policy ABE, chosen ciphertext secure ABE, constant size ciphertext ABE, they also discussed some relationship about ABE and other primitives such as conjunctive broadcast encryption and predicate encryption etc. In 2012, Parno et al. [21]first consider ABE's application in secure verifiable outsourcing computation. Garg et al. [10] first proposed the ABE scheme which can support any access control structure which can be expressed as circuits by using the multilinear map, this work can resist the backward-and-shift attack while construction on bilinear map can not. However, almost all the ABE scheme for circuits [12] can only achieve selective security, which is not satisfying. Recently, Garg et al. [11] proposed the first fully secure ABE schemes for circuits and proved the construction's full security by relying on some novel interesting techniques. In another line of research work, Chen et al. [7–9] try to give high efficient ABE schemes with full security and there are serval notable research results have been achieved. Recently, how to efficient outsource the decryption of ABE has gained great attention and many wonderful results have been achieved [14, 15, 19, 22].

1.3 Organization

We organize our paper as the following. In section II, we give the definition and security model for outsouced attribute based encryption. In section III, we give our first proposal, roughly analysis its security and give the comparison results. In section IV, we give our second proposal and its security proof, we also give a new way for outsourcing the decryption key, finally we give the performance and comparison results. In the last section, we conclude our paper with many interesting open problems.

2 Definition and Security Model

2.1 Definition

Let S represent a set of attributes, and A an accesss tructure. For generality, we will define (I_{enc}, I_{key}) as the inputs to the encryption and key generation function respectively. In a $KP-ABE$ scheme $(I_{enc}, I_{key}) = (S, A)$ while in a $CP-ABE$ scheme $(I_{enc}, I_{key}) = (A, S)$. A $CP-ABE(KP-ABE)$ scheme with outsourcing functionality consists of five algorithms:

1. $\mathsf{Setup}(\lambda, U)$. The setup algorithm takes security parameter and attribute universe description as input. It outputs the public parameters PK and a master key MK.
2. $\mathsf{Encrypt}(PK, M, I_{enc})$. The encryption algorithm takes as input the public parameters PK, a message M, and an access structure (resp. attribute set)I. It outputs the ciphertext CT.
3. $\mathsf{KeyGen}_{out}(MK, I_{key})$. The key generation algorithm takes as input the master key MK and an attribute set (resp. access structure) I_{key} and outputs a private key SK and a transformation key TK.
4. $\mathsf{Transform}(TK, CT)$. The ciphertext transformation algorithm takes as input a transformation key TK for I_{key} and a ciphertext CT that was encrypted under I_{enc}. It outputs the partially decrypted ciphertext CT' if $S \in A$ and the error symbol $\perp$ otherwise.
5. $\mathsf{Decrypt}(SK, CT)$. The decryption algorithm takes as input a private key SK for I_{key} and a partially decrypted key ciphertext CT' that was originally encrypted under I_{enc}. It outputs the message M if $S \in A$ and the error symbol $\perp$ otherwise.

2.2 Security Model

Definition 1 A $CP-ABE$ or $KP-ABE$ scheme with outsourcing is selective CPA-secure (or selective secure against chosen-plaintext attacks) if all polynomial time adversaries have at most a negligible advantage in the below game.

The formal security game consists of the following phases:

1. Init. $\mathcal{A}$ declares a set of encryption attribute S that will be used to create the challenge ciphertext during Challenge phase, submits S to the challenger.
2. Setup. The challenger runs the Setup algorithm and gives the pubilc parameters to the adversary $\mathcal{A}$.
3. $\mathsf{Phase\ 1}$. The challenger initializes an empty table T, an empty set D and an integer $j = 0$. Proceeding adaptively, the adversary can repeatedly make any of the following queries:
   - $\mathsf{Creat}(I_{key})$: The challenger sets $j := j+1$. It runs the outsourced key generation algorithm on I_{key} to obtain the pair (SK, TK) and stores in table T the entry (j, I_{key}, SK, TK). It then returns to the adversary the transformation key TK. Note: Create can be repeatedly queried with the same input.
   - $\mathsf{Corrupt}(i)$: If there exists an $i-th$ entry in table T, then the challenger obtains the entry (i, I_{key}, SK, TK) and sets $D := D \cup \{I_{key}\}$. It then returns to the adversary the private key SK. If no such entry exists, then it returns $\perp$.
4. $\mathsf{Challenge}$. The adversary submits two messages M_0, M_1. In addition the adversary gives a value I^*_{enc} such

that for all $I_{key} \in D$, $f_{I_{key}}, I^*_{enc}) \neq 1$. The challenger flips a random coin b, and encrypts M_b under I^*_{enc}. The resulting ciphertext CT^* is given to the adversary.

5. Phase 2. Phase 1 is repeated with the restrictions that the adversary cannot trivially obtain a private key for the challenge ciphertext. That is, it can not issue a Corrupt query that would result in a value I_{key} which satisfies $f(I_{key}, I^*_{enc}) = 1$ being added to D.
6. Guess. The adversary outputs a guess b' for b. The adversarys advantage in this game is defined as $|Pr[b' = b] - 1/2|$.

3 The First Proposal

3.1 System model

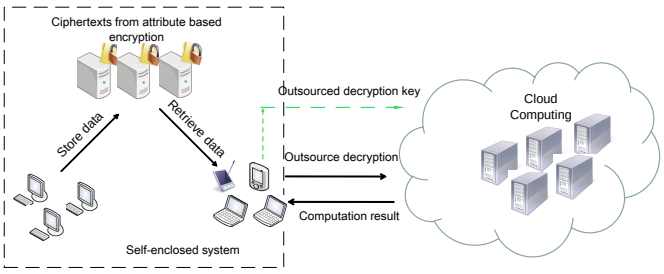

Fig. 1: Outsourced attributed based encryption for self-enclosed system

In this subsection, we describe the system model for our proposal. Our first proposal is mainly aiming at relieve the decryption overhead of mobile users for secure access control system using attribute based encryption in a self-enclosed setting. The self-enclosed system here refers that the cloud can not access the information flow in this system. In this self-enclosed system, there are three roles: the data owners who upload their files to the storage server using attribute based encryption, the data users can scalable access these encrypted files by using their secret keys if the attributes associated with the ciphertext satisfying the access structure tree associated with their secret keys, and the storage servers for storing the uploaded ciphertexts. Outside the self-enclosed system, there exists another party named the cloud service provider. The data users maybe use mobile devices which shall greatly restrict their computation ability, thus it need to outsource the computation work to the cloud, here we mainly focus on how to outsource the decryption of ciphertexts for the mobile devices. In this model, the cloud is a semi-trusted party, which shall be interested to derive the private keys of the data users, note here we do not consider the verifiability of the computation results, which is another issue many other papers have solved.

3.2 Review of GPSW'S Key-Policy ABE

3.2.1 Access Tree

Ciphertexts are labeled with sets of attributes and private keys are associated with access structures that control which ciphertexts a user is able to decrypt.

Access tree Γ. Let Γ be a tree representing an access structure. Each non-leaf node of the tree represents a threshold gate, described by its children and a threshold value. If num_x is the number of children of a node x and k_x is its threshold value, then $0 \leq k \leq num_x$. When $k_x = 1$, the threshold gate is an OR gate and when $k_x = num_x$, it is an AND gate. Each leaf node x of the tree is described by an attribute and a threshold value $k_x = 1$.

To facilitate working with the access trees, we define a few functions. We denote the parent of the node x in the tree by $parent(x)$. The function $att(x)$ is defined only if x is a leaf node and denotes the attribute associated with the leaf node x in the tree. The access tree T also defines an ordering between the children of every node, that is, the children of a node are numbered from 1 to num. The function $index(x)$ returns such a number associated with the node x. Where the index values are uniquely assigned to nodes in the access structure for a given key in an arbitrary manner.

Satisfying an access tree. Let Γ be an access tree with root r. Denote by Γ_x the subtree of Γ rooted at the node x. Hence Γ is the same as Γ_r. If a set of attributes γ satisfies the access tree Γ_x, we denote it as $\Gamma_x(\gamma) = 1$. We compute $\Gamma_x(\gamma)$ recursively as follows. If x is a non-leaf node, evaluate $\Gamma'_x(\gamma)$ for all children x' of node x. $\Gamma_x(\gamma)$ returns 1 if and only if at least k_x children return 1. If x is a leaf node, then $\Gamma_x(\gamma)$ returns 1 if and only if $att(x) \in \gamma$.

3.2.2 GPSW Scheme

1. Setup: Let G_1 be a bilinear group of prime order p, and let g be a generator of $\mathbb{G}_1$. In addition, let $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ denote the bilinear map. We also define the Lagrange coefficient $\Delta_{i,S}$ for $i \in Z_p$ and a set, S, of elements in Z_p: $\Delta_{i,S(x)} = \Pi_{j \in S, j \neq i} \frac{x-j}{i-j}$. Define the universe of attributes $U = \{1, 2, \cdots, n\}$.
2. Key Generation(Γ):
   - For each attribute $i \in U$, choose random number t_i uniformly from Z_p.

- Choose random number y uniformly in Z_p.
- The published public parameters PK is

$$T_1 = g^{t_1}, \cdots, T_{|U|} = g^{t_{|U|}}, Y = g^y$$

And the authority's master key MK is:

$$(t_1, \cdots, t_{|U|}, y)$$

- Then the authority generates a key that enables the User to decrypt a message encrypted under a set of attributes γ if and only if $\Gamma(\gamma) = 1$. The algorithm proceeds as follows.
  (a) Choose a polynomial q_x for each node x (including the leaves) in the tree T .These polynomials are chosen in the following way in a top-down manner, starting from the root node r. For each node x in the tree, set the degree d_x of the polynomial q_x to be one less than the threshold value k_x of that node, that is, $d_x = k_x - 1$. Now, for the root node r, set $q_r(0) = y$ and d_r other points of the polynomial q_r randomly to define it completely. For any other node x, set $q_x(0) = q_{parent(x)}(index(x))$ and choose d_x other points randomly to completely define q_x.
  (b) Once the polynomials have been decided, for each leaf node x, the authority give the following secret key to the user:

$$D_x = g^{\frac{q_x(0)}{t_i}}$$

where $i = att(x)$.

3. Encryption(M, γ, PK): To encrypt a message $M \in \mathbb{G}_2$ under a set of attributes γ, choose a random value $r \in Z_p$ and publish the ciphertext as:

$$E = (\gamma, E' = Me(g^r, Y) = Me(g, g)^{ry},$$
$$\{E_i = T_i^r (i \in \gamma)\})$$

4. Decryption(E, D): We specify our decryption procedure as a recursive algorithm.
   - We first define a recursive algorithm $DecryptNode(E, D, x)$ that takes as input the ciphertext E, the private key D (we assume the access tree Γ is embedded in the private key), and a node x in the tree. It outputs a group element of $\mathbb{G}_2$ or $\perp$.
   - Let $i = att(x)$, if the node x is a leaf node then:
     – If $i \in \gamma$,
     $$Decrypt(E, D, x) = e(D_x, E_i)$$
     $$= e(g^{\frac{q_x(0)}{t_i}}, g^{rt_i}) = e(q, q)^{rq_x(0)}$$
     – Otherwise, return $\perp$.

We now consider the recursive case when x is a non-leaf node. The algorithm $DecryptNode(E, D, x)$ then proceeds as follows: For all nodes z that are children of x, it calls $DecryptNode(E, D, z)$ and stores the output as F_z. Let S_x be an arbitrary k_x -sized set of child nodes z such that $F_z \neq \perp$. If no such set exists then the node was not satisfied and the function returns $\perp$. Otherwise, we compute

$$\begin{aligned} F_x &= \prod_{z \in S_x} F_z^{\delta_{i,S'_x}(0)} \\ &= \prod_{z \in S_x} (e(g,g)^{rq_z(0)})^{\delta_{i,S'_x}(0)} \\ &= \prod_{z \in S_x} (e(g,g)^{rq_{parent(z)}(index(z))})^{\delta_{i,S'_x}(0)} \\ &= \prod_{z \in S_x} e(g,g)^{rq_x(i)\delta_{i,S'_x}(0)} \\ &= e(g,g)^{rq_x(0)} \\ &= e(g,g)^{ry} \end{aligned}$$

3.3 Main Idea

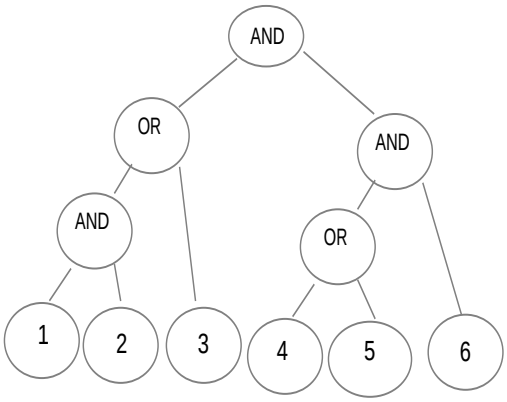

Fig. 2: Original Access Structure I

Our proposal is based on the following observation:

The access structure is tightly associated with the order of the leaf nodes. For example, in KP-ABE, the ciphertexts are associated with the attribute sets, and the policy is embedded in the private key of the users. But the point is that the ciphertexts shoud have some order, if we change the ciphertexts' order, which mean-

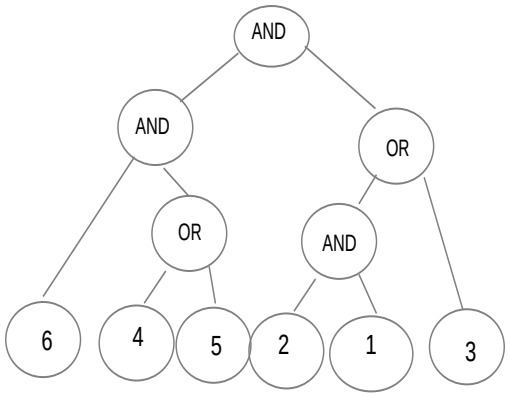

Fig. 3: Permutated Access Structure II

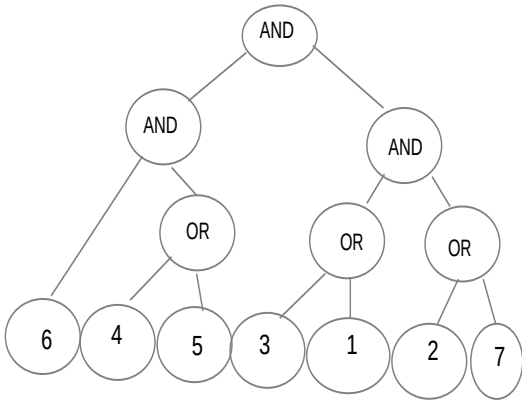

Fig. 4: Permutated Access Structure III

s the ciphertext corresponding to the attribute are no longer correct, the decryption result can no longer be correct! CP-ABE also has this feature.

A more detailed description is the following:

In Fig.1, which is the original access structure I, and the leaf nodes are $\{1, 2, \cdots, 6\}$, and the order of them is $\{1-2-3-4-5-6\}$. In Fig.2, we permutated the access structure, which denoted as II, the leaf nodes are still $\{1, 2, \cdots, 6\}$, but the order of them is $\{6-4-5-2-3-1\}$. In Fig.3, we further permutated the access structure II to get III, which has more difference. It has more leaf nodes and gates, which hide the topology of I. The leaf nodes are now $\{1, 2, \cdots, 6, 7\}$, and the order of them is $\{6-4-5-3-1-2-7\}$. To get the same derived boolean value, leaf node 3 should have the same boolean value with leaf node 7.

3.4 A Concrete Outsourced ABE Scheme

In this section, we give a concrete example to show how the ABE scheme with permutated access structure II works. Our scheme is based on the GPSW scheme [13] (Note GPSW scheme has a threshold access structure, but our Fig.1, 2, 3 all have the boolean formula structure. We emphasis this does not matter for OR gate can be seen as 1-to-1 threshold access structure and AND gate can be seen as a 2-to-1 threshold access structure.)

1. Setup: Let G_1 be a bilinear group of prime order p, and let g be a generator of $\mathbb{G}_1$. In addition, let $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ denote the bilinear map. We also define the Lagrange coefficient $\Delta_{i,S}$ for $i \in Z_p$ and a set, S, of elements in Z_p: $\Delta_{i,S(x)} = \Pi_{j \in S, j \neq i} \frac{x-j}{i-j}$. Define the universe of attributes $U = \{1, 2, \cdots, n\}$.
2. Key Generation(Γ)):
   - For each attribute $i \in U$, choose random number t_i uniformly from Z_p.
   - Choose random number y uniformly in Z_p.
   - The published public parameters PK is

     $$T_1 = g^{t_1}, \cdots, T_{|U|} = g^{t_{|U|}}, Y = g^y$$

     And the authority's master key MK is:

     $$(t_1, \cdots, t_{|U|}, y)$$

   - Then the authority generates a key that enables the User to decrypt a message encrypted under a set of attributes γ if and only if $\Gamma(\gamma) = 1$. The algorithm proceeds as follows.
     (a) Choose a polynomial q_x for each node x (including the leaves) in the tree T .These polynomials are chosen in the following way in a top-down manner, starting from the root node r. For each node x in the tree, set the degree d_x of the polynomial q_x to be one less than the threshold value k_x of that node, that is, $d_x = k_x - 1$. Now, for the root node r, set $q_r(0) = y$ and d_r other points of the polynomial q_r randomly to define it completely. For any other node x, set $q_x(0) = q_{parent(x)}(index(x))$ and choose d_x other points randomly to completely define q_x.
     (b) Once the polynomials have been decided, for each leaf node x, the authority generates the following keys:

     $$D_x = g^{\frac{q_x(0)}{t_i}}$$

where $i = att(x)$.

Note here Γ = *Access Structure I* and should be kept private by the decryptor.

3. Outsource KeyGen (D, Γ, P, Q) The decryptor first permutes his access structure and the sequence of the attributes, that is,

$$P(Access\ Structure\ I) = Access\ Structure\ \ II$$

$$Q(Order\ of\ leaf\ nodes\ in\ I) = Order\ of\ leaf\ nodes\ in\ II$$

Here P, Q denote the permutation of the access structure and permutation of the order of leaf nodes and is kept secret by the decryptor. Furthermore the following condition should be satisfied:

$$Q(Order\ of\ leaf\ nodes\ in\ I)$$

satisfying

$$P(Access\ Structure\ I)$$

if and if

$$Order\ of\ leaf\ nodes\ in\ I$$

satisfying

$$Access\ Structure\ I$$

The data user's outsourced decryption key will be

$$D^{outsource} = (Q(D_x, \cdots D_{x'}) = Q(g^{\frac{q_x(0)}{t_1}}, \cdots, g^{\frac{q_x(0)}{t_n}}), Access\ Structure\ II)$$

4. Encryption(M, γ, PK): To encrypt a message $M \in \mathbb{G}_2$ under a set of attributes γ, choose a random value $r \in Z_p$ and compute:

$$E = (\gamma, E' = Me(g^r, Y) = Me(g,g)^{ry}, \{E_i = T_i^r (i \in \gamma)\})$$

then the encrypter sends these ciphertexts to the data user.

5. Outsource Ciphertext(E, Q): After obtaining the ciphertext E, the decryptor permutes E by using permutation Q

$$E^{outsource} = (Q(\gamma), Q(\{E_i = T_i^r (i \in \gamma)\}))$$

and outsource them to the cloud.

6. Decryption(E', D'): After obtaining (E', D'), the cloud runs the decryption algorithm as following:
   - We first define a recursive algorithm $DecryptNode(E', D', x)$ that takes as input the ciphertext E', the private key D (we assume the access tree II is embedded in the private key), and a node x in the tree. It outputs a group element of $\mathbb{G}_2$ or $\perp$.
   - Let $i = att(x)$, if the node x is a leaf node then:
     - If $i \in Q(\gamma)$, $Decrypt(E^{outsource}, D^{outsource}, x) = = e(g^{\frac{q_x(0)}{t_i}}, g^{rt_i}) = e(q,q)^{rq_x(0)}$
     - Otherwise, return $\perp$.

   We now consider the recursive case when x is a non-leaf node. The algorithm $DecryptNode(E', D', x)$ then proceeds as follows: For all nodes z that are children of x, it calls $DecryptNode(E', D', z)$ and stores the output as F_z. Let S_x be an arbitrary k_x -sized set of child nodes z such that $F_z \neq \perp$. If no such set exists then the node was not satisfied and the function returns $\perp$. Otherwise, the cloud compute

$$\begin{aligned} F_x &= \prod_{z \in S_x} F_z^{\delta_{i,S'_x}(0)} \\ &= \prod_{z \in S_x} (e(g,g)^{rq_z(0)})^{\delta_{i,S'_x}(0)} \\ &= \prod_{z \in S_x} (e(g,g)^{rq_{parent(z)}(index(z))})^{\delta_{i,S'_x}(0)} \\ &= \prod_{z \in S_x} e(g,g)^{rq_x(i)\delta_{i,S'_x}(0)} \\ &= e(g,g)^{rq_x(0)} \\ &= e(g,g)^{ry} \end{aligned}$$

and returns it to the user, the user then computes

$$M = \frac{E'}{e(g,g)^{ry}} = \frac{Me(g^r, Y)}{e(g,g)^{ry}}$$

3.5 Security Analysis

Theorem 1 *Our construction of outsourced attribute based encryption is secure if the underlying permutation of order of leaves and permutation of access structure can not be inverted (e. g. number of the attributes is very large like 100 and the access structure is complex enough).*

Proof Here we roughly analysis our proposal's security. For the malicious cloud, although it can get the permutated secret key,

$$D^{outsource} = (Q(D_x, \cdots D_{x'}) = Q(g^{\frac{q_x(0)}{t_1}}, \cdots, g^{\frac{q_x(0)}{t_n}}), Access\ Structure\ II)$$

it can not know the concrete permutation, when the attributes are large enough such as 100 ones, it needs to try $100! = 9.33 \times 10^157$ times to knowing the concrete permutation, which shall be too large for a rational cloud service. But we also note our proposal can only be secure in the self-enclosed system, that is, the attributed based encryption ciphertexts can only

be known by the data owners and data users, and can not be accessed by the clouds. Otherwise, the malicious cloud may derive the permutation from the original sequence of ciphertexts and the permutated sequence of ciphertexts. We remark here such self-enclosed system is very common in our real life, such as E-health information system in hospital, which can be accessed by Doctors and Patients, but can not be accessed by others outside the hospital like the cloud service providers. Although our proposal is not a perfect solution for generic construction of outsourced attribute based encryption, but it can find its application in many real setting. Note here the mobile data users only need to permutate his secret keys and keep the permutation as the secret key, while all previous proposals for outsourced ABE at least require the data users to do a modular exponential operation for outsourcing the decryption key, thus our proposal is high efficient for the data users.

3.6 Comparison

Table 1: Comparison Result

Proposals	Method	Assumption	Efficiency
Previous Work	Cryptographic Way	Computation Intractable Problem	Not very high
Our	Permutation	Combinatorial Intractable Problem	High

Here we give a roughly comparison results with previous work, our work uses the permutation approach, while all previous work use the cryptographic approach, thus our proposal can be high efficient for the data users.

4 The Second Proposal

4.1 System model

Here we describe the system model for our second proposal, which is mainly for reducing the decryption overhead of mobile users for secure access control system using attribute based encryption. The open system here refers that the cloud can easily access the information flow (e.g. ABE ciphertexts) in this system. In this open system, there are four parties: the data owners, the data users, and the cloud storage service provider, and the cloud computing service provider. The data owners first use attribute based encryption to share the datum with the data users via storage server, the data users are often mobile users and need to use the cloud computing service to relieve the decryption overhead. Here the

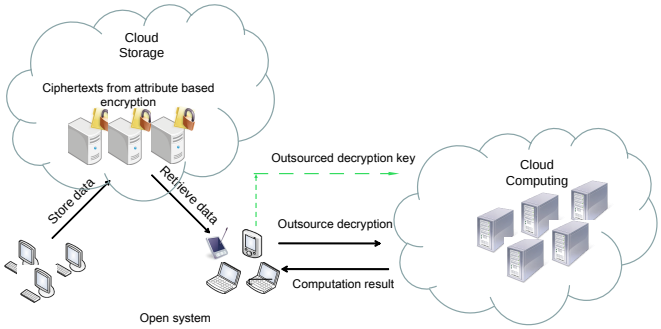

Fig. 5: Outsourced attributed based encryption for open system

cloud computing service is a semi-trusted party, which is curious about deriving the secret keys of the data users or the decryption results.

4.2 Review of Generic Construction of Expressive Key-Policy Attribute-Based Encryption with Constant-Size Ciphertexts

Attrapadung et al. construct monotonic KP-ABE with short ciphertexts by showing a general transformation that automatically turns any IBBE scheme fitting a certain template into a KP-ABE in the selective security model. Here we review this generic construction.

4.2.1 Linear ID-based Broadcast Encryption Template

They define a template that IBBE schemes should comply with in order to give rise to (selectively secure) KP-ABE schemes. They call this a linear IBBE template. Let (G, G_T) be underlying bilinear groups of order p. A linear IBBE scheme is determined by parameter $n_1, n_2 \in N$, a family F of vectors of functions, and a function D of which the latter two are specified by

$$\mathcal{F} \subset \{f_1, f_2, F\} | f_1 : Z_p^* \to \mathbb{G}, f_2 : Z_p^* \to \mathbb{G}^{n_1},$$
$$F : (Z_p^*)^{\leq n-1} \to \mathbb{G}^{\leq n_2},$$
$$\mathcal{D} : \mathbb{G}^{n_1+2} \times I \times \mathbb{G}^{n_2+1} \times \begin{pmatrix} I \\ < N \end{pmatrix} \to \mathbb{G}_T$$

with requirements specified below. A linear IBBE scheme works as follows.

1. $\mathsf{Setup}(\lambda, n)$. Given a security parameter $\lambda \in N$ and a bound $n \in N$ on the number of identities per ciphertext, the algorithm selects bilinear groups $(\mathbb{G}, \mathbb{G}_T)$ of prime order p and a generators $g \leftarrow R\mathbb{G}$. It computes $e(g,g)^\alpha$ for a random $\alpha \in \mathbb{Z}_p^*$ and chooses functions $(f_1, f_2, F) \leftarrow \mathcal{F}$. The master secret key consists of $msk = g^\alpha$ while the public key is $mpk := (g, e(g,g)^\alpha, f_1, f_2, F, n, n_1, n_2)$.

2. $\mathsf{Keygen}(msk, \mathsf{ID})$. It picks $r \in_R \mathbb{Z}_p^*$ and computes

$$sk_{\mathsf{ID}} = (d_1, d_2, d_3)$$
$$= (g^{\alpha} \cdot f_1(\mathsf{ID})^{\mathsf{r}}, \mathsf{g}^{\mathsf{r}}, \mathsf{f_2}(\mathsf{ID})^{\mathsf{r}}) \in \mathbb{G}^{\mathsf{n_1}+2}$$

3. $\mathsf{Encrypt}(mpk, M, S)$. It parses S as $S = \{\mathsf{ID_1}, \cdots, \mathsf{ID_q}\}$, where $q \leq n$. To encrypt $M \in \mathbb{G}_T$, it chooses a random exponent $s \in_R \mathbb{Z}_p^*$ and computes the ciphertext as

$$C = (C_0, C_1, C_2)$$
$$= (M \cdot e(g,g)^{\alpha s}, g^s, F(\mathsf{ID_1}, \cdots, \mathsf{ID_q})^{\mathsf{s}})$$

4. $\mathsf{Decrypt}(mpk, sk_{\mathsf{ID}}, \mathsf{ID}, \mathsf{C}, \mathsf{S})$. It parses $sk_{\mathsf{ID}} = (d_1, d_2, d_3)$ and $C = (C_0, C_1, C_2)$ then runs

$$\mathcal{D}((d_1, d_2, d_3), \mathsf{ID}, (\mathsf{C_1}, \mathsf{C_2}), \mathsf{S}) \rightarrow \mathsf{e(g,g)^{\alpha s}}$$

and obtains $M = C_0 / e(g,g)^{\alpha s}$. We are now ready to state the requirements: for all $(f_1, f_2, F) \in \mathcal{F}$, the following properties must hold

(a) $\mathsf{Correctness}$. For all $\alpha, r, s \in \mathbb{Z}_p^*$, $ID \in I$, $S = \{\mathsf{ID_1}, \cdots, \mathsf{ID_q}\} \in \begin{pmatrix} I \\ < N \end{pmatrix}$ and $\mathsf{ID} \in \mathsf{S}$, We have

$$\mathcal{D}((g^{\alpha} f_1(\mathsf{ID})^{\mathsf{r}}, \mathsf{g}^{\mathsf{r}}, \mathsf{f_2}(\mathsf{ID})^{\mathsf{r}}), \mathsf{g}^{\mathsf{r}}, \mathsf{f_2}(\mathsf{ID})^{\mathsf{r}})),$$
$$\mathsf{ID}, (\mathsf{g}^{\mathsf{s}}, \mathsf{F}(\mathsf{ID_1}, \cdots, \mathsf{ID_q})^{\mathsf{s}}), \mathsf{S}) = \mathsf{e(g,g)^{\alpha s}}$$

(b) $\mathsf{Linearity}$. For all $\gamma \in \mathbb{Z}_p^*$, $\mathsf{ID} \in \mathsf{I}$, $S \in (\begin{smallmatrix} I \\ < N \end{smallmatrix})$, $\mathsf{ID} \in \mathsf{S}$, $(d_1, d_2, d_3) \in G^{n_1+2}$, and $(C_1, C_2) \in \mathbb{G}^{\leq n_2+1}$, we have

$$\mathcal{D}((d_1, d_2, d_3)^{\gamma}, \mathsf{ID}, (C_1, C_2), S)$$
$$= \mathcal{D}((d_1, d_2, d_3), \mathsf{ID}, (C_1, C_2), S)^{\gamma}$$

4.2.2 Generic Conversion from Linear IBBE to KP-ABE

Let $\Pi_{IBBE} = (Setup', Keygen', Encrypt', Decrypt')$ be a linear IBBE system. They construct a KP-ABE scheme from Π_{IBBE} as follows:

1. $\mathsf{Setup}(\lambda, n)$: It simply outputs $Setup'(\lambda, n) \rightarrow (msk, mpk)$.
2. $\mathsf{KeyGen}(msk, (L, \pi))$: The algorithm computes a private key for an access structure that is associated with LSSS scheme (L, π) as follows. Let L be $l \times k$ matrix. First it generates shares of 1 with the LSSS scheme (L, π) as follows. Let L be $l \times k$ matrix. First, it generates shares of 1 with the $LSSS(L, \pi)$. Namely, it chooses a vector $\beta = (\beta_1, \beta_2, \cdots, \beta_k)^T \in_R (Z_p)^k$ subject to the constraint $\beta_1 = 1$. Then for each $i = 1$ to l, it calculates $\lambda_i =< L_i, \beta >$, picks $r' \in_R \mathbb{Z}_p$ and sets D_i as follows:

$$Keygen'(msk, \pi(i)) \rightarrow (d_{i,1}, d_{i,2}, d_{i,3})$$

$$D_i = (d_{i,1}^{\lambda_i} \cdot f_1(\pi(i))^{r'}, d_{i,2}^{\lambda_i} \cdot g^{r'}, d_{i,3}^{\lambda_i} \cdot f_2(\pi(i))^{r'})$$

It then outputs the private key as

$$sk_{(L,\pi)} = \{D_i\}_{i=1,\cdots,l}$$

3. $\mathsf{Encrypt}(mpk, M, \omega)$: It simply outputs

$$Encrypt'(mpk, M, \omega) \rightarrow (C_0, C_1, C_2)$$

4. $\mathsf{Decrypt}(mpk, sk_{(L,\pi)}, (L, \pi), C, \omega)$: Assume first that the policy (L, π) is satisfied by the attribute set ω, so that decryption is possible. Let $I = \{i | \pi(i) \in \omega\}$. It calculates the reconstruction constants $\{(i, u_i)\}_{i \in I} = Recon_{L,\pi}(\omega)$. It parses C as (C_0, C_1, C_2) and $sk_{L,\pi}$ as $\{D_i\}_{i=1,\cdots,l}$ where $D_i = (d'_{i,1}, d'_{i,2}, d'_{i,3})$. For each $i \in I$, it computes

$$\mathcal{D}((d'_{i,1}, d'_{i,2}, d'_{i,3}), ID, (C_1, C_2), S) \rightarrow e(g,g)^{\alpha \cdot s \cdot \lambda_i} \quad (1)$$

which we prove correctness below. It compute $e(g,g)^{\alpha s} = \prod_{i \in I} (e(g,g)^{\alpha s \lambda_i})^{u_i}$ and finally obtains $M = \frac{C_0}{e(g,g)^{\alpha s}}$, where we recall that $\Sigma_{i \in I} \lambda_i u_i = 1$.

$\mathsf{Correctness}$. They now verify that equation (1) is correct. First from a property of keys in linear IBBE, we have that $(d_{i,1}, d_{i,2}, d_{i,3})$ will be in the form $(g^{\alpha} \cdot f_1(\pi(i))^{r_i}, g^{r_i}, f_2(\pi(i))^{r_i})$ for some $r_i \in_R \mathbb{Z}_p^*$. Therefore we have

$$D_i = (g^{\alpha \lambda_i} \cdot f_1(\pi(i))^{\bar{r} \lambda_i}, g^{\bar{r} \lambda_i}, f_2(\pi(i))^{\bar{i} \lambda_i}) = (d_1^{\lambda_i}, d_2^{\lambda_i}, d_3^{\lambda_i})$$

with $\bar{r_i} = r_i + r'/\lambda_i$ and $(d_1, d_2, d_3) = sk_{\pi(i)}$ with randomness $\bar{r_i}$. Hence

$$\mathcal{D}((d'_{i,1}, d'_{i,2}, d'_{i,3}), ID, (C_1, C_2), S)$$
$$= \mathcal{D}((d_1, d_2, d_3), ID, (C_1, C_2), S)^{\lambda_i}$$
$$= (e(g,g)^{\alpha \cdot s})^{\lambda_i}$$

where each equality holds from linearity and correctness of $\mathcal{D}$ respectively.

The authors also give the security theorem of the above generic construction:

Theorem 2 *If the underlying IBBE scheme is selectively secure, then the resulting KP-ABE system is also selectively secure.*

4.2.3 IBBE Instantiation with Short Ciphertexts

This subsection presents an IBBE scheme with short ciphertexts and shows how to apply the KP-ABE conversion. This specific IBBE can be seen as an instance of the functional encryption (FE) for zero inner-product, which itself is implied by spatial encryption of [10]. A FE system for zero inner- product is defined by a relation $R^{ZIP} : \mathbb{Z}_p \times \mathbb{Z}_p \rightarrow \{0,1\}$ where $R^{ZIP}(\boldsymbol{X}, \boldsymbol{Y}) = 1$ if $< \boldsymbol{X}, \boldsymbol{Y} >= 0$. The technique of deriving an IBBE scheme from a FE scheme for zero inner-product can

be traced to [23]. A private key for an identity ID is defined by setting $X = (x_1, \cdots, x_n)^T$, with $x_i = \mathsf{ID}_{i-1}$. To encrypt to a set $S = \{ID_1, \cdots, ID_q\}$, one defines $Y = (y_1, \cdots, y_n)^T$ as a coefficient vector from

$$P_S[Z] = \Sigma_{i=1}^{q+1} y_i Z^{i-1} = \prod_{ID_j \in S} (Z - ID_j) \quad (2)$$

where, if $q+1 \leq n$, the coordinates $y_{q+2}, \cdots, y_n$ are set to 0. By doing so, we note that $P_S[\mathsf{ID}] =< \mathbb{X}, \mathbb{Y} >$ evaluates to 0 iff $\mathsf{ID} \in \mathsf{S}$. We now describe the IBBE instantiated from the FE system of [4]. Its selective security is an immediate consequence of [4], where it is proved under the DBDHE assumption.

1. $\mathsf{Setup}(\lambda, n)$: It chooses bilinear groups $(\mathbb{G}, \mathbb{G}_T)$ of prime order $p \geq 2^\lambda$ with $g \leftarrow_R \mathbb{G}$. It randomly chooses $\alpha, \alpha_0 \in \mathbb{Z}_p$, $\boldsymbol{\alpha} = (\alpha_1, \cdots, \alpha_n)^T \leftarrow_R \mathbb{Z}_p^n$. It then sets $H = (h_1, \cdots, h_n)^T = g^{\boldsymbol{\alpha}}$. The master secret key is $msk = \alpha$, and the public key is $mpk = (g, e(g,g)^\alpha, h_0 = g^{\alpha_0}, H = g^{\boldsymbol{\alpha}})$.
2. $\mathsf{KeyGen}(msk, \mathsf{ID})$: The algorithm first defines a vector $X = \{x_1, \cdots, x_n\}^T$ such that $x_i = \mathsf{ID}^{i-1}$ for $i = 1$ to n. It chooses $r \in_R \mathbb{Z}_p$ and outputs the private key $sk_{\mathsf{ID}} = (D_1, D_2, K_2, \cdots, K_n)$ where

$$D_1 = g^\alpha \cdot h_0^r, D_2 = g^r, \{K_i = (h_1^{\frac{-x_i}{x_1}})^r\}_{i=2,\cdots,n}$$

3. $\mathsf{Encrypt}(mpk, M, S)$: To encrypt M to the receiver set S (where $|S| \leq n$), the algorithm defines $\mathbb{Y} = (y_1, y_2, \cdots, y_n)^T$ as the coefficient vector of $P_S[Z]$ from equation (2). It then picks $s \leftarrow_R \mathbb{Z}_p$ and computes the ciphertext as

$$C = (C_0, C_1, C_2) = (M \cdot e(g,g)^{\alpha s}, g^s, (h_0 \cdot h_1^y \cdots h_n^{y_n})^s)$$

4. $\mathsf{Decrypt}(mpk, sk_{\mathsf{ID}}, \mathsf{ID}, \mathsf{C}, \mathsf{S})$: It defines the vector $Y = (y_1, \cdots, y_n)^T$ from the polynomial $P_S[Z]$ as usual. It then computes

$$e(g,g)^{\alpha s} = \frac{e(C_1, D_1 \cdot K_2^{y_2} \cdots K_n^{y_n})}{e(C_2, D_2)}$$

and recovers $M = \frac{C_0}{e(g,g)^{\alpha s}}$.

Correctness. If $< X, Y >= 0$, then decryption recovers M since

$$D_1 \cdot \prod_{i=2}^n K_i^{y_i} = g^\alpha (h_0 \cdot h_1^{-\frac{1}{x_1}(<X,Y> - x_1 y_1)} \prod_{i=2}^n h_i^{y_i})^r$$

$$= g^\alpha (h_0 \cdot \prod_{i=1}^n h_i^{y_i})^r$$

so that $e(C_1, D_1 \cdot \prod_{i=1}^n K_i^{y_i}) = e(g,g)^{\alpha s} \cdot e(h_0 \prod_{i=1}^n h_i^{y_i}, g^{rs})$ equals the product $e(g,g)^{\alpha s} \cdot e(C_2, D_2)$.

Applying the KP-ABE Conversion. The above IBBE can be considered as a linear IBBE system with $n_1 = n-1$, $n_2 = 1$ and the family $\mathcal{F}$ is defined by taking all functions of the following forms ranging over $h_0, h_1, \cdots, h_n \in G$:

$$f_1(\mathsf{ID}) = \mathsf{h_0}, \mathsf{f_2}(\mathsf{ID})$$
$$= (h_1^{-\mathsf{ID}} h_2, \cdots, h_1^{-ID^{n-1}} h_n), F(\mathsf{ID_1}, \cdots, \mathsf{ID_q})$$
$$= h_0 \prod_{i=1}^{q+1} h_i^{y_i}$$

where the vector $Y = \{y_1, \cdots, y_n\}^T$ is defined from the polynomial $P_S[Z]$ in equation (2) as usual. In addition, the function D is the computation in equation (3), which can be show to have linearity as required. The resulting KP-ABE has constant-size ciphertexts. This comes with the expense of longer private keys of size $O(tn)$, where t is the number of attributes in the access structure.

4.3 Our Proposal

Here we give a generic construction just like the above transformation: Let $\Pi_{IBBE} = (Setup', Keygen', Encrypt', Decrypt')$ be a linear IBBE system. Then we construct an efficient outsourcing KP-ABE scheme with energy efficiency from Π_{IBBE} as follows:

1. $\mathsf{Setup}(\lambda, n)$: It simply outputs $Setup'(\lambda, n) \rightarrow (msk, mpk)$.
2. $\mathsf{KeyGen}(msk, (L, \pi))$: The algorithm computes a private key for an access structure that is associated with LSSS scheme (L, π) as follows. Let L be $l \times k$ matrix. First it generates shares of 1 with the LSSS scheme (L, π) as follows. Let L be $l \times k$ matrix. First, it generates shares of 1 with the $LSSS(L, \pi)$. Namely, it chooses a vector $\beta = (\beta_1, \beta_2, \cdots, \beta_k)^T \in_R (Z_p)^k$ subject to the constraint $\beta_1 = 1$. Then for each $i = 1$ to l, it calculates $\lambda_i =< L_i, \beta >$, picks $r' \in_R \mathbb{Z}_p$ and sets D_i as follows:

$$Keygen'(msk, \pi(i)) \rightarrow (d_{i,1}, d_{i,2}, d_{i,3})$$
$$D_i = (d_{i,1}^{\lambda_i} \cdot f_1(\pi(i))^{r'}, d_{i,2}^{\lambda_i} \cdot g^{r'}, d_{i,3}^{\lambda_i} \cdot f_2(\pi(i))^{r'})$$

It then outputs the private key as $sk_{(L,\pi)} = \{D_i\}_{i=1,\cdots,l}$.
3. $\mathsf{Outsourcing\ KeyGen}(sk_{(L,\pi)}, Temp)$: This algorithm computes the outsourcing private key for data user. After input the secret key $sk_{(L,\pi)}$ and a random $Temp \in \mathbb{Z}_p^*$ which acts as the key to mask the real secret key, this algorithm outputs the outsourced secret key $sk_{(L,\pi)}^{outsource}$ for this data user

$$D_i^{\frac{1}{Temp}} = ((d_{i,1}^{\lambda_i} \cdot f_1(\pi(i))^{r'})^{\frac{1}{Temp}}, (d_{i,2}^{\lambda_i} \cdot g^{r'})^{\frac{1}{Temp}}, (d_{i,3}^{\lambda_i} \cdot f_2(\pi(i))^{r'})^{\frac{1}{Temp}})$$

The data user preserves $Temp$ for final decryption.

4. Encrypt(mpk, M, ω): It simply outputs $Encrypt'(mpk, M, \omega) \rightarrow (C_0, C_1, C_2)$.
5. Outsourcing Decrypt($mpk, sk_{(L,\pi)}, (L, \pi), C, \omega$): Assume first that the policy (L, π) is satisfied by the attribute set ω, so that decryption is possible. Let $I = \{i|\pi(i) \in \omega\}$. The cloud calculates the reconstruction constants $\{(i, u_i)\}_{i\in I} = Recon_{L,\pi}(\omega)$. It parses C as (C_0, C_1, C_2) and $sk_{L,\pi}^{outsource}$ as $\{D_i^{\frac{1}{Temp}}\}_{i=1,\cdots,l}$ where $D_i^{\frac{1}{Temp}} = ((d'_{i,1})^{\frac{1}{Temp}}, (d'_{i,2})^{\frac{1}{Temp}}, (d'_{i,3})^{\frac{1}{Temp}})$. For each $i \in I$, it computes

$$\mathcal{D}(((d'_{i,1})^{\frac{1}{Temp}}, (d'_{i,2})^{\frac{1}{Temp}}, (d'_{i,3})^{\frac{1}{Temp}}), ID, (C_1, C_2), S) \rightarrow e(g, g)^{\frac{\alpha\cdot s\cdot\lambda_i}{Temp}} \quad (3)$$

which we prove correctness below. It compute

$$TempDecrypt = e(g, g)^{\frac{\alpha s}{Temp}} = \prod_{i\in I}(e(g, g)^{\frac{\alpha\cdot s\cdot\lambda_i}{Temp}})^{u_i}$$

where we recall that $\Sigma_{i\in I}\lambda_i u_i = 1$.

6. Decrypt($TempDecrypt, Temp$): On inputs $TempDecrypt$ and $Temp$, this algorithm finally obtains $e(g, g)^{\alpha s} = TempDecrypt^{Temp} = (e(g, g)^{\frac{\alpha s}{Temp}})^{Temp}$ and thus $M = \frac{C_0}{e(g,g)^{\alpha s}}$.

Correctness. We can verify that equation (3) is correct. First from a property of keys in linear IBBE, we have that $(d_{i,1}, d_{i,2}, d_{i,3})$ will be in the form $(g^{\alpha} \cdot f_1(\pi(i))^{r_i}, g^{r_i}, f_2(\pi(i))^{r_i})$ for some $r_i \in_R \mathbb{Z}_p^*$. Therefore we have

$$D_i = (g^{\alpha\lambda_i}\cdot f_1(\pi(i))^{\bar{r}\lambda_i}, g^{\bar{r}\lambda_i}, f_2(\pi(i))^{\bar{i}\lambda_i}) = (d_1^{\lambda_i}, d_2^{\lambda_i}, d_3^{\lambda_i})$$

with $\bar{r_i} = r_i + r'/\lambda_i$ and $(d_1, d_2, d_3) = sk_{\pi(i)}$ with randomness $\bar{r_i}$. Hence

$$\mathcal{D}(((d'_{i,1})^{\frac{1}{Temp}}, (d'_{i,2})^{\frac{1}{Temp}}, (d'_{i,3})^{\frac{1}{Temp}}), ID, (C_1, C_2), S) = \mathcal{D}((d_1, d_2, d_3), ID, (C_1, C_2), S)^{\frac{\lambda_i}{Temp}} = (e(g, g)^{\alpha\cdot s})^{\frac{\lambda_i}{Temp}}$$

where each equality holds from linearity and correctness of $\mathcal{D}$ respectively. And the correctness of Decrypt algorithm follows easily. Our technique is same as [14]

4.4 Security Analysis

We have the following security theorem on our proposal (denoted as KP-OABE):

Theorem 3 *If the underlying IBBE scheme is selectively secure, then so is the resulting KP-OABE system. More precisely, for any selective-set adversary $\mathcal{A}$ against the KP-OABE construction, there is an IND-sID-CPA adversary $\mathcal{B}$ against the IBBE scheme and we have:*

$$Adv_{\mathcal{B}}^{IBBE-sID-CPA}(\lambda) \geq Adv_{\mathcal{A}}^{KP-OABE-sCPA}(\lambda)$$

More concretely, our KP-ABE scheme with outsourced decryption is selectively CPA-secure assuming that the scheme of Attrapdung [4] is an selectively CPA-secure IBBE scheme.

Proof We construct a simple IND-sID-CPA adversary $\mathcal{B}$ against the IBBE scheme assuming that a selective-set attacker $\mathcal{A}$ has non-negligible advantage against the KP-OABE system. Namely, $\mathcal{B}$ plays the role of $\mathcal{A}$s challenger and interacts with his own challenger in the IBBE security game. Here, we call the challenger of IBBE as $\mathcal{C}_I$.

The game begins with our KP-OABE adversary $\mathcal{A}$ choosing an attribute set ω^* that intends to attack. The Attrapdung IBBE adversary $\mathcal{B}$ then announces $S^* = \{i \in \omega^*\}$ as her target set of receivers. The system-wide Attrapdung IBBE public key that $\mathcal{B}$ receives from her challenger are relayed to $\mathcal{A}$ as system-wide parameters for the our KP-OABE scheme.

1. Init. KP-OABE adversary $\mathcal{A}$ chooses the set of attribute set ω^* it wishes to be challenged upon and sends to $\mathcal{B}$.
2. Setup. The challenger $\mathcal{C}_I$ computes the public key $MPK = (g, e(g, g)^{\alpha}, f_1, f_2, F, n, n_1, n_2)$ and sends these to the adversary $\mathcal{A}$.
3. Phase 1. In this phase the simulator $\mathcal{B}$ answers the following queries by $\mathcal{A}$. And $\mathcal{B}$ constructs the corresponding key table Q, then it answers the adversarys queries as follows:
   $\mathcal{A}$ may ask for the private key of any access structure (L, π) such that ω^* does not satisfy (L, π). To answer such a query, let L_{ω^*} be the sub-matrix formed by the rows of L that correspond to an attribute in $\omega*$. Since $\mathbf{1} = (1, 0, \cdots, 0)^T$ is not in the row space of L_{ω^*}, there must exist an efficiently computable vector w such that $L_{\omega*} \cdot w = 0$ and $< 1, w > \neq 0$. Let h denote the value of $< 1, w > \neq 0$. Let h denote the value $< 1, w >$. To construct a private key, $\mathcal{B}$ has to define a vector $\boldsymbol{u} = a \cdot \boldsymbol{\beta}$ such that . $\mathcal{B}$ implicitly sets $\boldsymbol{u}$ as $\boldsymbol{u} = \boldsymbol{v} + \psi \cdot \boldsymbol{w}$, where $\boldsymbol{v} = (v_1, \cdots, v_k)^T$ is a randomly chosen vector and $\psi = (\alpha - v_1)/h$, so that $(1, w) = \alpha$. To generate triples $(D_{i,1}, D_{i,2}, D_{i,3})$ for each row of L, $\mathcal{B}$ proceeds as follows:
   (a) Let $\Gamma_1 = \{j \in \{1, \cdots, l\}|\pi(j) \in \omega^*\}$. For each $j \in \Gamma_1$, if $\Gamma_1^T = (m_{j1}, \cdots, m_{jk})$ denotes the jth row of L, we have $< \boldsymbol{L}_j, \boldsymbol{u} > = < \boldsymbol{L}_j, \boldsymbol{u} > = \sum_{t_1=1}^{k} m_{j_{t_1} v_{t_1}}$ and the share $\lambda_j = < \boldsymbol{L}_j, \boldsymbol{u} >$ is thus computable, so that $\mathcal{B}$ can pick integers

$\lambda_j, r_j \in_R Z_p^*$ and define $(D_j = D_{j,1}, D_{j,2}, D_{j,3}) = (g^{\lambda_j} \cdot f_1(\pi(j))^{r_j}, g^{r_j}, f_2(\pi(j))^{r_j})$. $\mathcal{B}$ calls the the IBBE key generation oracle on (L, π) to obtain the private key $sk_{L,\pi}$. Then $\mathcal{B}$ chooses a random value $Temp \in Z_p^*$ and sets the outsourced secret key $sk_{L,\pi}^{outsource}$ as $D_j^{1/Temp}$. Finally $\mathcal{B}$ stores $(sk_{L,\pi}, sk_{L,\pi}^{outsource}, Temp)$ in table Q.

(b) Let $\Gamma_2 = \{j \in \{1, \cdots, l\} | \pi(j) \notin \omega^*\}$. For each $j \in \Gamma_2$, $\mathcal{B}$ is allowed to query its own challenger $\mathcal{C}_I$ to extract

$$(d_{j,1}, d_{j,2}, d_{j,3}) \leftarrow \prod_{IBBE} Keygen(msk, \pi(j))$$

We have $< \boldsymbol{L}_j, \boldsymbol{u} >=< \boldsymbol{L}_j, \boldsymbol{v} > + \psi < \boldsymbol{L}_i, \bar{w} >= \sum_{t_1=1}^{k} m_{jt_1} \left(v_{t_1} + \frac{\alpha - v_1}{h} \omega_{t_1}\right) = \mu_1 \alpha + \mu_2$ where the coefficients $\mu_1 = (\sum_{t_1=1}^{k} m_{j_{t_1}} w_{t_1}) \cdot h^{-1}$ and $\mu_2 = h^{-1} \sum_{t_1=1}^{k} m_{j_{t_1}} w_{t_1} (h v_{t_1} - v_1 w_{t_1})$ are both computable, so that $\mathcal{B}$ can obtain a well-formed triple

$D_j = (D_{j,1}, D_{j,2}, D_{j,3})$
$= (d_{j,1}^{\mu_1} g^{\mu_2} f_1(\pi(j))^{r'_j}, d_{j,2}^{\mu_1} \cdot g^{r'_j}, d_{j,2}^{\mu_1} \cdot g^{r'_j},$
$d_{j,3}^{\mu_1} \cdot f(\pi(j))^{r'_j})$

$\mathcal{B}$ calls the the IBBE key generation oracle on (L, π) to obtain the private key $sk(L, \pi)$. Then, $\mathcal{B}$ chooses a random value $Temp' \in Z_p^*$ and sets the outsourced secret key $(sk'_{(L,\pi)})^{outsource}$ as $D_j^{\frac{1}{Temp'}}$. Finally, $\mathcal{B}$ stores

$$(sk'_{(L,\pi)}, (sk'_{(L,\pi)})^{outsource}, Temp)$$

in table Q.

4. Challenge. The adversary $\mathcal{A}$ submits two equal length messages M_0, M_1 and attribute S to $\mathcal{B}$ to obtain the challenge ciphertext CT^*. Then, $\mathcal{B}$ sends the M_0, M_1 to C_I before relaying the challenge ciphertexts back to $\mathcal{A}$.
5. Phase 2. The adversary $\mathcal{A}$ continues to adaptively queries as in Phase 1, but with the restriction that the adversary cannot violate the constraint on the challenge attribute S. B responds the queries as in Phase 1. And B eventually outputs the same result $\sigma \in \{0, 1\}$ as $\mathcal{A}$ does. It is easy to see that $\mathcal{B}$ never has to query her challenger to extract the private key for an identity of the target attribute set $S^* = \omega^*$.
6. Guess. Eventually, $\mathcal{A}$ outputs a bit β', then $\mathcal{B}$ outputs β'. This ends the description of the simulation. Thus, if $\mathcal{A}$ has advantage ϵ in the selective security game against our scheme, then $\mathcal{B}$ breaks the $IBBE$ scheme with the same probability. It comes that $\mathcal{B}$ is successful whenever $\mathcal{A}$ is so, this ends our proof.

4.5 A New Way to Outsource the Decryption Key with Only One Modular Exponentiation Cost

Here we give a new way for secure efficient outsourcing the decryptor's decryption key for KP-ABE, which only cost one modular exponentiation, concretely the outsourced KP-ABE scheme is the following:

1. Setup(λ, n): It simply outputs

$$Setup'(\lambda, n) \rightarrow (msk, mpk)$$

2. KeyGen$(msk, (L, \pi))$: The algorithm computes a private key for an access structure that is associated with LSSS scheme (L, π) as follows. Let L be $l \times k$ matrix. First it generates shares of 1 with the LSSS scheme (L, π) as follows. Let L be $l \times k$ matrix. It chooses a vector $\beta = (\beta_1, \beta_2, \cdots, \beta_k)^T \in_R (Z_p)^k$ subject to the constraint $\beta_1 = 1$. Then for each $i = 1$ to l, it calculates $\lambda_i =< L_i, \beta >$, picks $r' \in_R \mathbb{Z}_p$ and sets D_i as follows:

$$Keygen'(msk, \pi(i)) \rightarrow (d_{i,1}, d_{i,2}, d_{i,3})$$

$$D_i = (d_{i,1}^{\lambda_i} \cdot f_1(\pi(i))^{r'}, d_{i,2}^{\lambda_i} \cdot g^{r'}, d_{i,3}^{\lambda_i} \cdot f_2(\pi(i))^{r'})$$

It then outputs the private key as $sk_{(L,\pi)} = \{D_i\}_{i=1,\cdots,l}$.

3. Outsourcing KeyGen$(sk_{(L,\pi)}, z)$: This algorithm computes the outsourcing private key for data user. After input the secret key $sk_{(L,\pi)}$ and a random $z \in \mathbb{Z}_p^*$ which acts as the key to mask the real secret key, this algorithm outputs the outsourced secret key $sk_{(L,\pi)}^{outsource}$ for this data user

$D_i^{outsource} =$
$((d_{i,1}^{\lambda_i} \cdot f_1(\pi(i))^{r'}) \cdot g^z, (d_{i,2}^{\lambda_i} \cdot g^{r'}) \cdot g^z, (d_{i,3}^{\lambda_i} \cdot f_2(\pi(i))^{r'}) \cdot g^z)$

The data user preserves z for final decryption.

4. Encrypt(mpk, M, ω): It simply outputs

$$Encrypt'(mpk, M, \omega) \rightarrow (C_0, C_1, C_2)$$

5. Outsourcing Decrypt$(mpk, sk_{(L,\pi)}, (L, \pi), C, \omega)$: Assume first that the policy (L, π) is satisfied by the attribute set ω, so that decryption is possible. Let $I = \{i | \pi(i) \in \omega\}$. The cloud calculates the reconstruction constants $\{(i, u_i)\}_{i \in I} = Recon_{L,\pi}(\omega)$. It parses C as (C_0, C_1, C_2) and $sk_{L,\pi}^{outsource}$ as $((d'_{i,1}) \cdot g^z, (d'_{i,2}) \cdot g^z, (d'_{i,3}) \cdot g^z)$. For each $i \in I$, it computes

$$X_i = \mathcal{D}(((d'_{i,1}) \cdot g^z, (d'_{i,2}) \cdot g^z, (d'_{i,3}) \cdot g^z), ID, (C_1, C_2), S) \quad (4)$$

$$Y_i = \mathcal{D}((g, g, g), ID, (C_1, C_2), S) \quad (5)$$

which we prove correctness below. It compute

$$TempDecrypt_1 = \prod_{i \in I} (e(g,g)^{\alpha \cdot s \cdot \lambda_i} (Y_i^z))^{u_i}$$

and $TempDecrypt_2 = \prod_{i \in I} (Y_i)^{u_i}$ where we recall that $\Sigma_{i \in I} \lambda_i u_i = 1$.

6. Decrypt($TempDecrypt_1$, $TempDecrypt_2$, z): On inputs $TempDecrypt_1$, $TempDecrypt_2$, z, this algorithm finally obtains $e(g,g)^{\alpha s} = \frac{TempDecrypt_1}{TempDecrypt_2^z}$ and thus $M = \frac{C_0}{e(g,g)^{\alpha s}}$.

Correctness. We can verify that equation (3) is correct. First from a property of keys in linear IBBE, we have that $(d_{i,1}, d_{i,2}, d_{i,3})$ will be in the form $(g^{\alpha} \cdot f_1(\pi(i))^{r_i}, g^{r_i}, f_2(\pi(i))^{r_i})$ for some $r_i \in_R \mathbb{Z}_p^*$. Therefore we have

$$D_i = (g^{\alpha \lambda_i} \cdot f_1(\pi(i))^{\bar{r} \lambda_i}, g^{\bar{r} \lambda_i}, f_2(\pi(i))^{\bar{i} \lambda_i}) = (d_1^{\lambda_i}, d_2^{\lambda_i}, d_3^{\lambda_i})$$

with $\bar{r_i} = r_i + r'/\lambda_i$ and $(d_1, d_2, d_3) = sk_{\pi(i)}$ with randomness $\bar{r_i}$. Hence

$$\mathcal{D}((g^z, g^z, g^z), ID, (C_1, C_2), S)$$
$$= \mathcal{D}((g,g,g), ID, (C_1, C_2), S)^z$$

where each equality holds from linearity and correctness of $\mathcal{D}$ respectively. And the correctness of Decrypt algorithm follows easily. Security. Here we roughly analysis the security of this proposal. The main difference between this proposal and the above one is this: in Outsourcing KeyGen algorithm, the outsourced key is

$$D_i^{outsource} =$$
$$((d_{i,1}^{\lambda_i} \cdot f_1(\pi(i))^{r'}) \cdot g^z, (d_{i,2}^{\lambda_i} \cdot g^{r'}) \cdot g^z, (d_{i,3}^{\lambda_i} \cdot f_2(\pi(i))^{r'}) \cdot g^z)$$

For z is unknown to the adversary, thus the adversary can not easily derive the real decryption key.

4.6 Feature and Performance Analysis

4.6.1 Feature Comparison

Table 2: Comparison Result

Proposals	Constant Ciphertext Length	Verifiability	Contribution
[14]	No	No	First OD-ABE
[15]	No	Yes	First OD-ABE-V
[19]	No	Yes	First OD/E/K-ABE-V
[22]	No	Yes	Efficient OD-ABE-V
Ours	Yes	No	Efficient OD-C-ABE

In this subsection, we compare our work with the previous work, with the emphasis on the ciphertext length. OD-ABE denotes outsourcing decryption of ABE, OD-ABE-V denotes outsourcing decryption of ABE with verifiability, OD/E/K-ABE-V denotes outsourcing decryption/encryption/key generation of ABE with verifiability, OD-C-ABE denotes outsourcing decryption of ABE with constant ciphertexts. We also note our scheme can easily achieve verifiability property by adapting the technique presenting in [22], which is the most efficient one until now. From the above table, we can see our proposal is the most energy efficient scheme for mobile users when outsourcing the decryption of attribute based encryption to the cloud, which will has important applications for mobile users, while this is a more and more common setting in our life.

4.6.2 Performance Analysis

We roughly evaluate the performance of our second proposal. According to the benchmark of JPBC, based on the testbed3 which with the following hardware platforms: HTC Desire HD A9191, Android 2.2 (Java Port) [27], we give the comparison results between Green et al.'s scheme [14] and our scheme, which can be seen in Table II, III, IV and Fig. 6, 7, 8, 9. These performance evaluation results show our second proposal is practical.

Table 3: ABE Ciphertext Size (Kbytes)

Scheme/Number of Attributes	20	40	60	80	100
GHW [14]	1.385	2.645	3.905	5.165	6.425
Ours	0.188	0.188	0.188	0.188	0.188

Table 4: Outsourcing KeygenTime(seconds)

Scheme/Number of Attributes	20	40	60	80	100
GHW [14]	0.118	0.176	0.234	0.292	0.350
Our first way 4.3	0.089	0.089	0.089	0.089	0.089
Our another new way4.5	0.029	0.029	0.029	0.029	0.029

Table 5: Final Decryption time (seconds)

Scheme/Number of Attributes	20	40	60	80	100
Ours	0.0198	0.0198	0.0198	0.0198	0.0198

5 Conclusion

In this paper, we consider the issue of outsourcing decryption of ABE for mobile devices with energy efficiency. We give two proposals. The first one is based on a novel permutation technique for access structure and attributes, which can be used for self-enclosed system. The second one is based on ABE schemes with

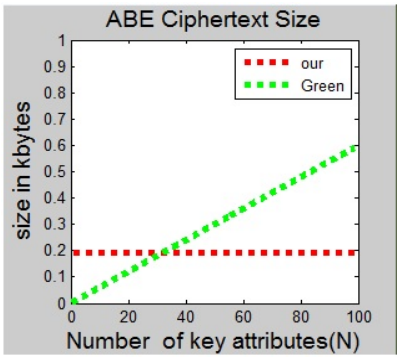

Fig. 6: ABE cipertext size comparison

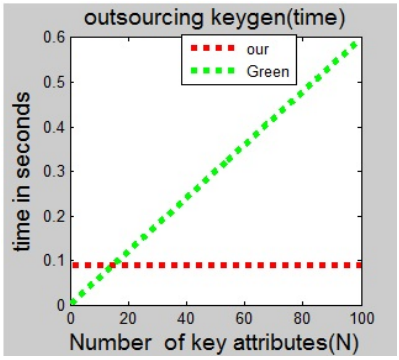

Fig. 7: Outsourcing keygen time comparison

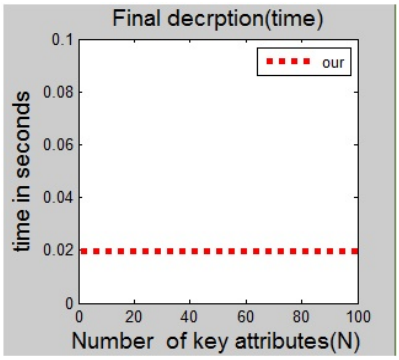

Fig. 8: Final decryption time

constant ciphertexts for mobile users in the cloud setting. We propose a generic construction for outsourced ABE based on Attrapadung et al. generic construction of ABE. We think the ciphertext length is an important issue for outsourcing for it directly affect the battery's energy consuming, the more shorter the ciphertexts are, the more surviving time the mobile phones can support. However, we also note our result is very basic, many open problems are leaving such as proving proposing outsourcing decryption of ABE candidates with constant ciphertexts and constant private

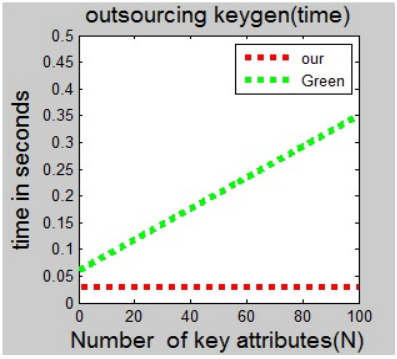

Fig. 9: Outsourcing keygen time comparison between GHW and our new way

keys etc, extending the idea of permutation technique to ciphertext-policy attribute based encryption etc.

Acknowledgements

The second author and third author are the corresponding authors. This work is supported by the National Cryptography Development Fund of China Under Grants No. MMJJ20170112, National Key Research and Development Program of China Under Grants No. 2017YFB0802002, National Nature Science Foundation of China (Grant Nos. U1636114, 61402531, 61572521), the Nature Science Basic Research Plan in Shaanxi Province of china (Grant Nos. 2014JM8300, 2014JQ8358, 2015JQ6231, 2016JQ6037) and Guangxi Key Laboratory of Cryptography and Information Security (No. GCIS201610).

References

1. Nuttapong Attrapadung. Dual system encryption via doubly selective security: Framework, fully secure functional encryption for regular languages, and more. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 557–577. Springer, May 2014.
2. Nuttapong Attrapadung and Hideki Imai. Conjunctive broadcast and attribute-based encryption. In Hovav Shacham and Brent Waters, editors, *PAIRING 2009*, volume 5671 of *LNCS*, pages 248–265. Springer, August 2009.
3. Nuttapong Attrapadung and Hideki Imai. Dual-policy attribute based encryption. In Michel Abdalla, David Pointcheval, Pierre-Alain Fouque, and Damien Vergnaud, editors, *ACNS 09*, volume 5536 of *LNCS*, pages 168–185. Springer, June 2009.
4. Nuttapong Attrapadung, Benoît Libert, and Elie de Panafieu. Expressive key-policy attribute-based encryption with constant-size ciphertexts. In Dario Catalano, Nelly Fazio, Rosario Gennaro, and Antonio Nicolosi, editors, *PKC 2011*, volume 6571 of *LNCS*, pages 90–108. Springer, March 2011.

5. Melissa Chase. Multi-authority attribute based encryption. In Salil P. Vadhan, editor, *TCC 2007*, volume 4392 of *LNCS*, pages 515–534. Springer, February 2007.
6. Melissa Chase and Sherman S. M. Chow. Improving privacy and security in multi-authority attribute-based encryption. In Ehab Al-Shaer, Somesh Jha, and Angelos D. Keromytis, editors, *ACM CCS 09*, pages 121–130. ACM Press, November 2009.
7. Jie Chen and Hoeteck Wee. Fully, (almost) tightly secure IBE and dual system groups. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 435–460. Springer, August 2013.
8. Jie Chen and Hoeteck Wee. Doubly spatial encryption from DBDH. Cryptology ePrint Archive, Report 2014/199, 2014. `http://eprint.iacr.org/2014/199`.
9. Jie Chen and Hoeteck Wee. Dual system groups and its applications — compact HIBE and more. Cryptology ePrint Archive, Report 2014/265, 2014. `http://eprint.iacr.org/2014/265`.
10. Sanjam Garg, Craig Gentry, Shai Halevi, Amit Sahai, and Brent Waters. Attribute-based encryption for circuits from multilinear maps. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 479–499. Springer, August 2013.
11. Sanjam Garg, Craig Gentry, Shai Halevi, and Mark Zhandry. Fully secure functional encryption without obfuscation. Cryptology ePrint Archive, Report 2014/666, 2014. `http://eprint.iacr.org/2014/666`.
12. Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Attribute-based encryption for circuits. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 545–554. ACM Press, June 2013.
13. Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM CCS 06*, pages 89–98. ACM Press, October / November 2006. Available as Cryptology ePrint Archive Report 2006/309.
14. Matthew Green, Susan Hohenberger, and Brent Waters. Outsourcing the decryption of abe ciphertexts. Proc. USENIX Security Symposim, San Francisco, CA, USA, 2013.
15. Junzuo Lai, Robert Deng, Caowen Guan, and Jian Weng. Attribute-based encryption with verifiable outsourced decryption. *IEEE Transaction on Information Forensics and Security*, 8(8):1343–1354, 2013.
16. Allison B. Lewko, Tatsuaki Okamoto, Amit Sahai, Katsuyuki Takashima, and Brent Waters. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 62–91. Springer, May 2010.
17. Allison B. Lewko and Brent Waters. Decentralizing attribute-based encryption. In Kenneth G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 568–588. Springer, May 2011.
18. Allison B. Lewko and Brent Waters. New proof methods for attribute-based encryption: Achieving full security through selective techniques. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 180–198. Springer, August 2012.
19. Jin Li, Xinyi Huang, Jingwei Li, Xiaofeng Chen, and Yang Xiang. Securely utsourcing attribute-based encryption with checkability. *IEEE Transation on Parallel and Distributed Systems, In Press, doi: 10.1109/TPDS.2013.27*, 2013.
20. Tatsuaki Okamoto and Katsuyuki Takashima. Fully secure functional encryption with general relations from the decisional linear assumption. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 191–208. Springer, August 2010.
21. Bryan Parno, Mariana Raykova, and Vinod Vaikuntanathan. How to delegate and verify in public: Verifiable computation from attribute-based encryption. In Ronald Cramer, editor, *TCC 2012*, volume 7194 of *LNCS*, pages 422–439. Springer, March 2012.
22. Baodong Qin, Robert H. Deng, Shengli Liu, and Siqi Ma. Attribute-based encryption with efficient verifiable outsourced decryption. *IEEE Transaction on Information Forensics and Security*, 10(7):1384–1393, 2015.
23. Amit Sahai and Brent R. Waters. Fuzzy identity-based encryption. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 457–473. Springer, May 2005.
24. Brent Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 619–636. Springer, August 2009.
25. Shota Yamada, Nuttapong Attrapadung, Goichiro Hanaoka, and Noboru Kunihiro. Generic constructions for chosen-ciphertext secure attribute based encryption. In Dario Catalano, Nelly Fazio, Rosario Gennaro, and Antonio Nicolosi, editors, *PKC 2011*, volume 6571 of *LNCS*, pages 71–89. Springer, March 2011.
26. Shota Yamada, Nuttapong Attrapadung, Goichiro Hanaoka, and Noboru Kunihiro. A framework and compact constructions for non-monotonic attribute-based encryption. In Hugo Krawczyk, editor, *PKC 2014*, volume 8383 of *LNCS*, pages 275–292. Springer, March 2014.
27. Angelo De Caro, Vincenz Iovino. jPBC: java pairing based cryptography. In *Proceeding of the 16th IEEE symposium on Comuputers and Communications, ISCC 2011*, pages 850–855. IEEE, 2011. `http://gas.dia.unisa.it/projects/jpbc/`.
28. Xu An Wang, Jianfeng Ma, Fatos Xhafa. Outsourcing decryption of attribute based encryption with energy efficiency. In *Proceeding of the International Conference on P2P, Parallel, Grid, Cloud and Internet Computing 3PGCIC 2015*, pages 444-448. IEEE, 2015.