Reverse Outsourcing: Reduce the Cloud's Workload in Outsourced Attribute-Based Encryption Scheme

Fei Meng, Leixiao Cheng, and Mingqiang Wang*

School of Mathematics, Shandong University, Jinan 250100, China
Key Laboratory of Cryptologic Technology and Information Security, Ministry of Education, Jinan 250100, China
School of Mathematical Sciences, Fudan University, Shanghai 200433, China
wangmingqiang@sdu.edu.cn

Abstract. Attribute-based encryption (ABE) is a cryptographic technique known for ensuring fine-grained access control on encrypted data. One of the main drawbacks of ABE is the time required to decrypt the ciphertext is considerably expensive, since it grows with the complexity of access policy. Green et al. [USENIX, 2011] provided the outsourced ABE scheme, in which most computational overhead of ciphertext decryption is outsourced from end user to the cloud. However, their method inevitably increases the computational burden of the cloud. While millions of users are enjoying cloud computing services simultaneously, it may cause huge congestion and latency.

In this paper, we propose a heuristic primitive called *reverse outsourcing* to reduce the cloud's workload. Specifically, the cloud is allowed to transform the ciphertext decryption outsourced by the end user into several computing tasks and dispatches them to idle users, who have some smart devices connected to the internet but not in use. These devices can provide computing resources for the cloud, just like the cloud hires many employees to complete the computing work. Besides, the computing results returned by the idle users should be verified by the cloud.

We propose a reverse outsourced CP-ABE scheme in the *rational idle user model*, where idle users will be rewarded by the cloud after returning the correct computing results and they prefer to get rewards instead of saving resources. According to the Nash equilibrium, we prove that the best strategy for idle users is to follow our protocol honestly, because the probability of deceiving the cloud with incorrect computing results is negligible. Therefore, in our scheme, most computational overhead of ciphertext decryption is shifted from the cloud to idle users, leaving a constant number of operations for the cloud.

Keywords: Fine-grained access control · Attribute-based encryption · Cloud computing · Reverse outsourcing.

* Corresponding author

1 Introduction

Cloud computing is widely used to store or share data and provide computing services. However, data and services on the cloud are usually open and accessible to anyone, so data owners are often advised to encrypt their data before uploading it to the cloud to protect sensitive information. In many application scenarios, such as in industrial, academic and medical fields, it is necessary for the data owner to establish a specific access control policy to decide who can decrypt the ciphertext. To solve these problems, attribute-based encryption (ABE) [18], initially introduced by Sahai et al., is an expansion of public key encryption that allows users to encrypt and decrypt data based on user attributes. There are two types of ABE schemes, key-policy ABE (KP-ABE) [7] and ciphertext-policy ABE (CP-ABE) [3,19]. In CP-ABE, the data owner embeds an access policy in the ciphertext and the private key of end user is associated with an attribute set. A user can decrypt the ciphertext if his/her attributes satisfy the access policy. It is on the contrary for the KP-ABE. Since the access policy is defined by the data owner, CP-ABE is more suitable for the data sharing scenario than KP-ABE.

Currently, ABE schemes with various functionalities have been widely constructed, e.g., supporting regular languages [1, 20], with unbounded attribute size [1, 13, 17], with constant-size ciphertext [2], with multi-authority [4, 5, 12], and with adaptive security [11, 14, 16].

One of the main drawbacks in the existing ABE schemes [3, 7, 19] is that the number of pairing and exponentiation operations for ciphertext decryption is linear with the complexity of access policy, which means the computation cost of end user is quite expensive. This defect becomes more severe for users on their resource-constrained devices, such as smart phones. To reduce the computation cost of end user, some cryptographic operations with heavy computational load can be outsourced to third-party service [8, 21]. Combined with proxy re-encryption technique, Yu et al. [21] designed a KP-ABE scheme with fine-grained data access control. In this scheme, the root node of the access tree is an AND gate with one child is a leaf node associated with the dummy attribute and another one is the root of the access policy. The dummy attribute is authorized to every end user. The cloud service provider stores all the user's private key components except for the one corresponding to the dummy attribute and doesn't learn any information about the plaintext. Green et al. [8] provided a new methods for efficiently and securely outsourcing decryption of ABE ciphertexts. In their scheme, users private key is blinded by a random number. Both the private key and the random number are kept secret by the user. and the blinded private key is shared with the cloud to transform the original ciphertext into a simple and short El Gamal [6] type ciphertext. Therefore, most of the heavy cryptographic operations of decryption algorithm are shifted to the cloud, leaving only a small number of operations for the end user to recover the plaintext. In addition, Li et al. [15] also considered to outsource key-issuing for ABE schemes by introducing two cloud service providers to perform the outsourced key-issuing and decryption.

Although in the outsourced ABE schemes [8, 15, 21], the cloud is supposed to have almost unlimited computing capabilities, it is obviously not the case in reality. Outsourcing technique inevitably increases the computational burden of cloud computing, especially when millions of users are simultaneously connected to cloud server to enjoy data processing services, which may cause huge network congestion and delays and cloud server crashes. However, little efforts has been paid to reduce the burden on cloud server. There are countless smart devices in the world connected to the Internet, such as smart home appliances, smart cars, and even our personal computers. These devices have certain computing power and maybe not in use most of the time. Is there any way to aggregate these devices to provide computing services for the cloud?

1.1 Contribution

Motivated by the above issues, we initially introduced the primitive of reverse outsourcing to reduce cloud's workload. Specifically, the main contribution of our paper are briefly shown as follows:

In this paper, we propose a heuristic primitive called *reverse outsourcing* to reduce the cloud's workload which means that the cloud is allowed to transform the ciphertext decryption outsourced by the end user into several computing tasks and dispatches them to idle users (with smart devices online but not in use). The cloud should verify the computing results returned by idle users and reward them if the results are valid. We also propose the *rational idle user model* [9], in which idle users prefer to get rewards instead of saving resources.

To demonstrate how reverse outsourcing works, we apply reverse outsourcing to our proposed reverse outsourced CP-ABE scheme. Using Game theory [10] and Nash equilibrium to analyze each rational idle user's strategy, we prove that the best strategy for idle users is to follow our protocol honestly, since the probability of deceiving the cloud by incorrect computing results is negligible. When and only when all idle users follow our protocol honestly, each one can get the maximum benefit.

In the outsourced ABE schemes [8, 15, 21], the number of pairing operations to partially decrypt a ciphertext is linear to the complexity of the access policy. In our reverse outsourced CP-ABE scheme, most computational overhead of ciphertext decryption is shifted from the cloud to idle users, leaving a constant number of operations for both the cloud and the end user. As far as we know, this work is the first attempt to reduce the workload of the cloud in ABE.

1.2 Organization

This paper is organized as follows. Section 2 describes the necessary preliminaries. Section 3 presents the system and security model. We give a concrete construction and explicit analysis of our scheme in section 4 and section 5 respectively. In the end, section 6 summarizes the paper and prospects for the future research.

2 Preliminaries

2.1 Access Structures

Definition 1 (Access structure [3]). *Let $\{P_1, P_2, \ldots, P_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^{\{P_1,P_2,\ldots,P_n\}}$ is monotone if $\forall B, C$: if $B \in \mathbb{A}$ and $B \subseteq C$ then $C \in \mathbb{A}$. An access structure (respectively, monotone access structure) is a collection (respectively, monotone collection) $\mathbb{A}$ of non-empty subsets of $\{P_1, P_2, \ldots, P_n\}$, i.e., $\mathbb{A} \subseteq 2^{\{P_1,P_2,\ldots,P_n\}} \setminus \{\emptyset\}$. The sets in $\mathbb{A}$ are called the authorized sets, and the sets not in $\mathbb{A}$ are called the unauthorized sets.*

In this paper, attributes take the role of the parties and we only focus on the monotone access structure $\mathbb{A}$, which consists of the authorized sets of attributes. Obviously, attributes can directly reflect a user's authority.

Definition 2 (Access tree [3]). *Let $\mathcal{T}$ 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 n_x is the number of children of a node x and k_x is its threshold value, then $0 \leq k_x \leq n_x$. When $k_x = 1$, the threshold gate is an OR gate and when $k_x = n_x$, it is an AND gate. Each leaf node x of the tree is describe by an attribute and a threshold value $k_x = 1$.*

We introduce some functions that will be used in scheme construction and security proof. $\mathsf{parent}(x)$ denotes the parent of the node x in the access tree. $\mathsf{att}(x)$ 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 $\mathcal{T}$ also defines an ordering between the children of every node, that is, the children of a node are numbered from 1 to n_x. The function $\mathsf{index}(x)$ 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.

Definition 3 (Satisfying an access tree [3]). *Let $\mathcal{T}$ be an access tree with root r. Denote by $\mathcal{T}_x$ the subtree of $\mathcal{T}$ rooted at the node x. Hence $\mathcal{T}$ is the same as $\mathcal{T}_r$. If a set of attributes γ satisfies the access tree $\mathcal{T}_x$, we denote it as $\mathcal{T}_x(\gamma) = 1$. We compute $\mathcal{T}_x(\gamma)$ recursively as follows. If x is a non-leaf node, evaluate $\mathcal{T}_{x'}(\gamma) = 1$ for all children x' of node x. $\mathcal{T}_x(\gamma)$ returns 1 if and only if at least k_x children return 1. If x is a leaf node, then $\mathcal{T}_x(\gamma)$ returns 1 if and only if $\mathsf{att}(x) \in \gamma$.*

2.2 Bilinear Map and DBDH Assumption

Algorithms in our scheme are mainly implemented by bilinear maps, which is presented as follows:

Let $\mathbb{G}_0$ and $\mathbb{G}_T$ be two multiplicative cyclic groups of prime order p. Let g be a generator of $\mathbb{G}_0$ and e be a efficient computable bilinear map, $e : \mathbb{G}_0 \times \mathbb{G}_0 \to \mathbb{G}_T$. The bilinear map e has a few properties: (1) Bilinearity: for all $u, v \in \mathbb{G}_0$ and $a, b \in \mathbb{Z}_p$, we have $e(u^a, v^b) = e(u, v)^{ab}$. (2) Non-degeneracy: $e(g, g) \neq 1$. We say

that $\mathbb{G}_0$ is a bilinear group if the group operation in $\mathbb{G}_0$ and the bilinear map $e : \mathbb{G}_0 \times \mathbb{G}_0 \to \mathbb{G}_T$ are both efficiently computable. Notice that the map e is symmetric since $e(g^a, g^b) = e(g, g)^{ab} = e(g^b, g^a)$.

The security of our scheme is based on the decisional bilinear Diffie-Hellman (DBDH) assumption, which is defined as follows: Given the bilinear map parameter $(\mathbb{G}_0, \mathbb{G}_T, p, e, g)$ and three random elements $(x, y, z) \in \mathbb{Z}_p^3$, if there is no probabilistic polynomial time (PPT) adversary $\mathcal{B}$ can distinguish between $(g, g^x, g^y, g^z, e(g, g)^{xyz})$ and $(g, g^x, g^y, g^z, \vartheta)$, we can say that the DBDH assumption holds, where ϑ is randomly selected from $\mathbb{G}_T$. More specifically, the advantage ϵ of $\mathcal{B}$ in solving the DBDH problem is defined as

$$\Big|\Pr[\mathcal{A}(g, g^x, g^y, g^z, Z = e(g, g)^{xyz}) = 1] - \Pr[\mathcal{A}(g, g^x, g^y, g^z, Z = R) = 1]\Big|. \quad (1)$$

Definition 4 (DBDH). *We say that the DBDH assumption holds if no PPT algorithm has a non-negligible advantage ϵ in solving DBDH problem.*

3 System and Security Model

This section describes the fundamental descriptions of system parties, system model, threat model and security model.

3.1 System Parties

As shown in Fig. 1, our proposed reverse outsourced CP-ABE scheme requires the following five parties: Key Generation Center (KGC), Data Owner (DO), Cloud Server (CS), End User (EU), and Idle Users (IU). The specific role of each party is given as follows:

- Key Generation Center (KGC): The KGC is a fully trusted party which is in charge of generating public parameters and secret keys.
- Data Owner (DO): The DO owns data files. He/She encrypts the data with a specific access structure and uploads the ciphertext to the his/her own cloud storage account.
- Cloud Server (CS): The CS is an entity that provides computing and storage services. In this paper, we assume that the computational power of the CS is huge but not unlimited. The CS can help end user to partially decrypt the ciphertext added in his/her cloud storage account. The CS may choose to accomplish this computing task on its own or assign it to idle users and it also needs to check whether the results returned by idle users are correct or not.
- End User (EU): The end user can select and save the ciphertext in the DO's storage account into his / her own storage account. If his/her authority satisfies the access structure embedded in the given ciphertext, he/she can decrypt it and obtain the plaintext. Moreover, the EU is allowed to submit a partial decryption key to the CS to help him/her partially decrypt the ciphertext.
- Idle Users (IU): Idle Users are those who have smart devices connected to the network and not in use. These devices can be aggregated to provide computing services for the CS.

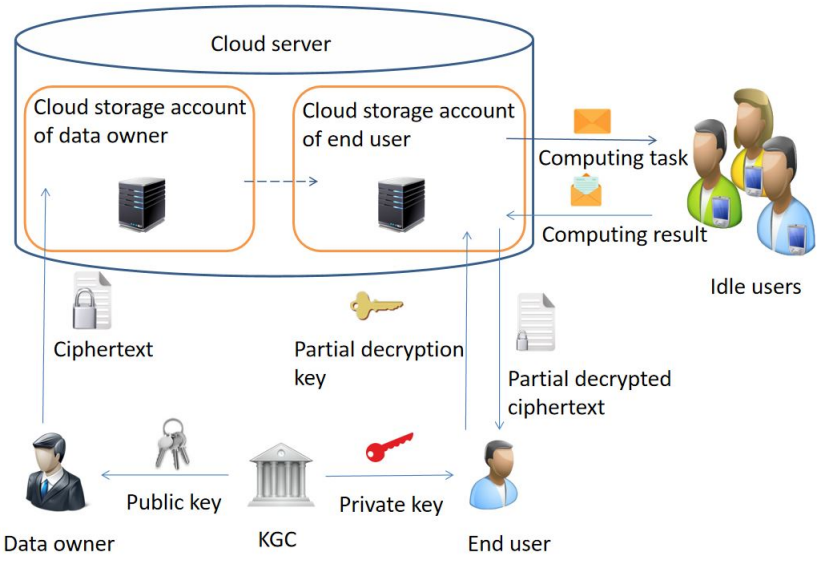

Fig. 1. System description of reverse outsourced CP-ABE scheme.

3.2 System Model

The proposed reverse outsourced CP-ABE scheme mainly consists of the following five fundamental algorithms:

- **Setup**$(1^\lambda, \mathcal{L}) \rightarrow (PK, MSK)$: Given security parameter λ and a set of all possible attributes $\mathcal{L}$, the KGC runs this algorithm to generate the public key PK and the master secret key MSK.
- **KeyGen**$(MSK, S) \rightarrow SK$: On input the master secret key MSK and an attribute set S, the KGC runs this algorithm and returns SK to the EU, where the partial decryption key SK_{pd} in contained in SK.
- **Enc**$(PK, \mathbb{A}, M) \rightarrow CT$: On input PK, an access structure $\mathbb{A}$ and the message M, the DO runs this algorithm to generate the ciphertext CT.
- **Tran**$(CT) \rightarrow CT'$: On input CT, the CS runs this algorithm to randomize CT to generate CT'.
- **PreDec**$(CT', SK_{pd}) \rightarrow CT''$ **or** $\perp$: On input CT' and SK_{pd}, the CS runs this algorithm to partially decrypt CT' and outputs CT'', if the attribute set S contained in SK_{pd} satisfies the access structure $\mathbb{A}$ embedded in CT. Otherwise, $\perp$. This algorithm consists of the following two steps:
  1. **Reverse outsource**: The CS divides the partial decryption task into several parts and assigns them to the IU;
  2. **Verification**: Gathering the results returned by IU, the CS checks whether the partial decryption task is accomplished correctly.
- **Dec**$(CT'', SK) \rightarrow M$: On input CT'' and SK, the EU decrypts CT'' and outputs M if his/her attribute set S satisfies the access structure $\mathbb{A}$ embedded CT.

3.3 Threat Model

In this paper, we assume that the KGC is a fully trusted third party, while the CS is an honest-but-curious entity, which exactly follows the protocol specifications but also are curious about the sensitive information of ciphertext. EU is not allowed to collude with CS. Nevertheless, malicious users may collude with each other to access some unauthorized ciphertexts. IU are rational and selfish, which means that they may cheat, but they are more willing to get rewards than to save computing resources and they are also not allowed to collude with CS.

3.4 Security Model

Our proposed scheme achieves chosen plaintext security, and the security game between a PPT adversary $\mathcal{A}$ and the challenger $\mathcal{C}$ is as follows.

- *Initialization*: $\mathcal{A}$ chooses and submits a challenge access structure $\mathbb{A}^*$ to its challenger $\mathcal{C}$.
- *Setup*: $\mathcal{C}$ runs **Setup** algorithm and returns the public key PK to $\mathcal{A}$.
- *Phase* 1: $\mathcal{A}$ adaptively submits any attribute set S to $\mathcal{C}$ with the restriction that S doesn't satisfy $\mathbb{A}^*$. In response, $\mathcal{C}$ runs **KeyGen** algorithm and answers $\mathcal{A}$ with the corresponding SK.
- *Challenge*: $\mathcal{A}$ submits two equal-length challenge messages (m_0, m_1) to $\mathcal{C}$. Then $\mathcal{C}$ picks a random bit $\vartheta \in \{0, 1\}$ and runs **Enc** algorithm to generate the challenge ciphertext CT^* of m_ϑ.
- *Phase* 2: This phase is the same as Phase 1.
- *Guess*: $\mathcal{A}$ outputs a guess bit ϑ' of ϑ. We say that $\mathcal{A}$ wins the game if and only if $\vartheta' = \vartheta$. The advantage of $\mathcal{A}$ to win this security game is defined as $Adv(\mathcal{A}) = \left| \Pr[\vartheta' = \vartheta] - \frac{1}{2} \right|$.

Definition 5. *The proposed scheme achieves IND-CPA security if there exist no PPT adversary winning the above security game with a non-negligible advantage ϵ under the DBDH assumption.*

4 Reverse Outsourcing

4.1 Definitions

We found that some computing tasks of the CS could be assigned to IU to reduce its own overhead, which we called reverse outsourcing. IU are those idle users with online smart devices that are not in use. Each of them can provide a small amount of computing resources for the cloud, but if millions these resources are brought together, the computing power will be considerable. Here, we formally introduce the concept of *reverse outsourcing*.

Definition 6 (Reverse Outsourcing). *For relieving the computational overhead, the cloud is allowed to divide a computing task into several sub-tasks and assign them to idle users. Each idle user returns a sub-result after completing his/her own assignment. The cloud combines all the sub-results to obtain the result of the original computing task, and then checks whether it is valid.*

While receiving the computing assignments from the CS, the IU should follow protocol specifications. If the results they returned are valid, they will be rewarded by the CS. In this paper, the reverse outsourcing is applied to the *rational idle user model*, which is defined as follows.

Definition 7 (Rational Idle User Model). *Rational idle user is selfish, lazy and always attempts to maximize his/her own profits, which means that he/she prefers to get rewards from the cloud rather than save the computing resources of his/her smart devices. Therefore, for each rational idle user* IU_i*, it holds that* $ut_i^{++} > ut_i^{+} > ut_i^{-} > ut_i^{--}$*, where*

- ut_i^{++} *is the utility of* IU_i *when he/she gets rewards without following the protocol specification.*
- ut_i^{+} *is the utility of* IU_i *when he/she follows the protocol specification and gets rewards.*
- ut_i^{-} *is the utility of* IU_i *when he/she doesn't get rewards without following the protocol specification.*
- ut_i^{--} *is the utility of* IU_i *when he/she follows the protocol specification but doesn't get rewards.*

In the rational idle user model, anyone is independent from another. Since the performance of IU_i satisfies $ut_i^{++} > ut_i^{+} > ut_i^{-} > ut_i^{--}$, he/she may try to defraud rewards without following the protocol, for example, by returning a random result, or he/she may collude with others to trick the cloud with the wrong results. So each IU_i has two strategies: follow the protocol or not. In order to analyze the best strategy for each IU_i, we formalize the *reverse outsourcing game* by means of Game theory [10] and introduce the notion of Nash equilibrium.

Definition 8 (Reverse Outsourcing Game). *The reverse outsourcing game is a tuple* $G_{RO} = \{\mathsf{IU}, T, ST, R, V\}$*, where*

- $\mathsf{IU} = \{\mathsf{IU}_1, \mathsf{IU}_2, \ldots, \mathsf{IU}_k\}$ *is the set of* k *rational idle users, where* $k \geq 1$*.*
- T *is a computing task, which can be divided into* k *sub-tasks* $\{T_1, T_2, \ldots, T_k\}$*. Each sub-task* T_i *is assigned to* IU_i *and the entire task* T *can be completed by completing each* T_i*.*
- $ST = \{st_1, st_2, \ldots, st_k\}$ *is the set of rational idle users' strategies in* G_{RO}*. In particular,* $st_i \in \{st_i^0, st_i^1\}$ *is the set of* IU_i*'s strategies.* st_i^0 *denotes that* IU_i *wants to be rewarded without following the protocol specification;* st_i^1 *denotes that* IU_i *follows the protocol honestly.*
- R *is the computational result of* T*, which can be obtained by gathering each sub-result* R_i *of* T_i*.*
- V *is a verification algorithm to check whether* R *is valid or not. If* R *is valid, every rational idle user will get the same reward. Otherwise, none of them will get reward.*

Definition 9 (Nash Equilibrium of G_{RO}). *For a given strategy* $ST^* = (st_1^*, st_2^*, \ldots, st_k^*)$*,* ST^* *is Nash equilibrium for* G_{RO}*, if and only if for any rational idle user* $\mathsf{IU}_i \in \mathsf{IU}$*, when the game* G_{RO} *is finished, for any* $st_i \in \{st_i^0, st_i^1\}$

and $i \in [1,k]$*, it holds that*

$$ut_i(st_i^* \mid ST^* \setminus \{st_i^*\}) \geq ut_i(st_i \mid ST^* \setminus \{st_i^*\}), \tag{2}$$

where $st_i^* \in \{st_i^0, st_i^1\}$.

4.2 Construction of Reverse Outsourced CP-ABE Scheme

Without loss of generality, we suppose that there are n possible attributes in total and $\mathcal{L} = \{\mathbf{a}_1, \mathbf{a}_2, ..., \mathbf{a}_n\}$ is the set of all possible attributes. Assume $\mathbb{G}_0, \mathbb{G}_T$ are multiplicative cyclic groups with prime order p and the generator of $\mathbb{G}_0$ is g. Let λ be the security parameter which determines the size of groups. Let $e : \mathbb{G}_0 \times \mathbb{G}_0 \to \mathbb{G}_T$ be a bilinear map and $H : \{0,1\}^* \longrightarrow \mathbb{Z}_p$ be a hash function which maps any string to a random element of $\mathbb{Z}_p$. Set the Lagrange coefficient as $\Delta_{i,L}(x) = \prod_{j \in L, j \neq i} \frac{x-j}{i-j}$, where $i \in \mathbb{Z}_p$ and a set, L, of elements in $\mathbb{Z}_p$.

We extract the notion that we called **Lagrange-route product** from [3]. Actually in [3], the calculation of the lagrange-route product is implied in the decryption process of ciphertexts.

Definition 10 (Lagrange-Route Product). *For an access tree* $\mathcal{T}$ *and each leaf node* y *of* $\mathcal{T}$*, there is only one route from* y *to the root node* R*. We define the route as a set* $S_{y \to R} = (y_o, y_1, y_2, \ldots, y_{R-1})$*, where* $y_0 = y$ *and* y_{i-1} *is the child node of* y_i*,* y_R *is the root node* R*. Then, the Lagrange-route product is defined as:*

$$\pi_y = \prod_{x \in S_{y \to R}} \Delta_{i,q_x}(0), \tag{3}$$

where q_x is a polynomial and its definition will be given in the following **Enc** algorithm. The details of our improved scheme are shown as follows.

- **Setup**$(1^\lambda, \mathcal{L}) \to (PK, MSK)$: Given a security parameter λ and all possible attributes $\mathcal{L}$, this algorithm chooses a bilinear group $\mathbb{G}_0$ of prime order p with generator g. Next it picks random exponents $\alpha, \beta, v_j \in_R \mathbb{Z}_p^*$, where $j \in [1,n]$. The public key is generated as:

$$PK = \left\{\mathbb{G}_0, g, h, g^\alpha, e(g,g)^\beta, e(g,h)^\beta, \{PK_j = g^{v_j} \mid \forall \mathbf{a}_j \in \mathcal{L}\}\right\}, \tag{4}$$

  and the master key as:

$$MSK = \left\{\alpha, \beta, \{v_j\}_{j \in [1,n]}\right\}. \tag{5}$$

- **KeyGen**$(MSK, S) \to SK$: While receiving an attribute set S from the EU, the key generation algorithm is run by KGC to output a key that identifies with S. The algorithm randomly chooses $r, r'x', y', z' \in_R \mathbb{Z}_p^*$. Then it returns the partial decryption key SK_{pd} as

$$SK_{pd} = \left\{ \begin{array}{l} SK_0 = x' + y', SK_1 = g^{(\beta+\alpha r)x'+z'}, SK_2 = g^{(\beta+\alpha r)y'}, \\ SK_3 = (gh)^{z'}, SK_4 = g^{\alpha r x'} h^{r'x'}, SK_5 = g^{r'x'}, \\ \left\{SK_{1,j} = g^{\frac{\alpha r x'}{v_j}}, SK_{2,j} = g^{\frac{\alpha r y'}{v_j}} \cdot h^{\frac{\alpha r (x'+y')}{v_j}} \mid \forall \mathbf{a}_j \in S\right\}, \end{array} \right\} \tag{6}$$

and the user secret key

$$SK = \{x', z', SK_{pd}\}. \tag{7}$$

- **Enc**$(PK, M, \mathcal{T}) \rightarrow CT$: The encryption algorithm encrypts a message M under the tree access structure $\mathcal{T}$. The algorithm first chooses a polynomial q_x for each node x (including the leaves) in $\mathcal{T}$. These polynomials are chosen from the root node R in a top-down manner: for each node x of $\mathcal{T}$, the degree of q_x is $d_x = k_x - 1$, where k_x is the threshold value of x; beginning with root node R, it picks $s_1, s_2 \in_R \mathbb{Z}_p^*$, sets $q_R(0) = s_1$ and randomly chooses $d_R = k_R - 1$ other points of q_R to define the polynomial completely; for any other node x, it sets $q_x(0) = q_{\mathsf{parent}(x)}(\mathsf{index}(x))$ and chooses d_x other points to define q_x completely. Let, Y be the set of all leaf nodes in $\mathcal{T}$ and $\mathcal{X}$ be the set of all attributes in $\mathcal{T}$. The algorithm computes $\{\pi_y \mid \forall y \in Y\}$. Then the ciphertext is constructed as

$$CT = \left\{ \begin{array}{l} \mathcal{T}, \tilde{C} = M \cdot e(g,g)^{\beta s_2}, C = e(g, gh)^{\beta s_1}, g^{s_1}, g^{s_2}, h^{s_1}, h^{s_2}, \\ \{C_y = g^{v_j q_y(0) \pi_y} \mid \forall y \in Y, \mathbf{a}_j = \mathsf{att}(y) \in \mathcal{X}\} \end{array} \right\}. \tag{8}$$

- **Tran**$(CT) \rightarrow CT'$: The CS picks $d_1, d_2 \in_R \mathbb{Z}_p^*$. For each CT uploaded by the DO, the CS generates CT' by only replacing C_y of CT with $\{C_{y,1}, C_{y,2}\}$, where $C_{y,1} = C_y{}^{d_1}$ and $C_{y,2} = C_y{}^{d_2}$.
- **PreDec**$(SK_{pd}, CT') \rightarrow CT''$ **or** $\perp$: To partially decrypt CT'' in the end user's storage account, which would like to be decrypted by the EU, the algorithm conducts as follows.
  In $\mathcal{T}$, for each leaf node $y \in Y$, let $\mathbf{a}_j = \mathsf{att}(y) \in \mathcal{X}$. The the CS picks out the minimized set $Y' \subseteq Y$ such that $\{\mathsf{att}(y)\}_{y \in Y'} \subseteq S \bigcap \mathcal{X}$ and $\{\mathsf{att}(y)\}_{y \in Y'}$ satisfies $\mathcal{T}$. If there is no such set, the algorithm returns $\perp$. Otherwise, it conducts the following steps:
  1. **Reverse outsource**: For each $i \in [1,2]$, we define the function $F_{i,y}$ as $F_{i,y}(C_{y,i}, SK_{i,j}, y) = e(C_{y,i}, SK_{i,j})$. The CS sets the computing task $T = \{T_i\}_{i \in [1,2]}$, where each sub-task $T_i = (F_{i,y}, \{C_{y,i}, SK_{1,j}\}_{\mathbf{a}_j = \mathsf{att}(y), y \in Y'})$ is assigned to the corresponding idle user IU_i.
     Given assignment as above, the IU_j computes and returns the sub-result R_i to the CS, where

$$R_i = \prod_{\mathbf{a}_j = \mathsf{att}(y), y \in Y'} F_{i,y}(C_{y,i}, SK_{i,j}, y). \tag{9}$$

     Receiving each R_i outputted by IU_i, the CS computes the result R corresponding with the original task T, where

$$R = \prod_{i \in [1,2]} R_i^{\frac{1}{d_i}}. \tag{10}$$

  2. **Verification**: The CS checks whether the result R is valid or not as follows:

(1) The CS interacts CT' and SK_{pd} to compute A and A' as:

$$\begin{aligned} A &= e(g^{s_1}h^{s_1}, SK_1) \cdot e(g^{s_1}h^{s_1}, SK_2) \\ &= e(g^{s_1}h^{s_1}, g^{(\alpha r+\beta)x'+z'}) \cdot e(g^{s_1}h^{s_1}, g^{(\alpha r+\beta)y'}) \\ &= e(g, gh)^{(\alpha r+\beta)(x'+y')s_1+z's_1}. \end{aligned} \tag{11}$$

$$A' = e(g^{s_2}, SK_1) \cdot e(g^{s_2}, SK_2) = e(g, g)^{(\alpha r+\beta)(x'+y')s_2+z's_2} \tag{12}$$

(2) The result R is valid only if the following equation holds,

$$\frac{A}{R} = C^{SK_0} \cdot e(g^{s_1}, SK_3). \tag{13}$$

For each $i \in [1, 2]$, if each IU_i follows the protocol specification and outputs the correct sub-result R_i, which means that

$$\begin{aligned} R_1 &= \prod_{y\in Y'} F_{1,y}(C_{y,1}, SK_{1,j}, y) = \prod_{y\in Y'} e(g^{v_j q_y(0)\pi_y d_1}, g^{\frac{\alpha r x'}{v_j}}) \\ &= e(g, g)^{\alpha r x' d_1 \sum\limits_{y\in Y'} q_y(0) \prod\limits_{x\in S_{y\to R}} \Delta_{i,q_x}(0)} = e(g, g)^{\alpha r x' s_1 d_1}, \end{aligned} \tag{14}$$

and similarly,

$$\begin{aligned} R_2 &= \prod_{y\in Y'} F_{2,y}(C_{y,2}, SK_{2,j}, y) = \prod_{y\in Y'} e(g^{v_j q_y(0)\pi_y d_2}, g^{\frac{\alpha r y'}{v_j}} \cdot h^{\frac{\alpha r(x'+y')}{v_j}}) \\ &= e(g, g)^{\alpha r y' s_1 d_2} \cdot e(g, h)^{\alpha r(x'+y')s_1 d_2} \end{aligned} \tag{15}$$

Then R is valid, since

$$\begin{aligned} \frac{A}{R} &= \frac{e(g, gh)^{(\alpha r+\beta)(x'+y')s_1+z's_1}}{e(g, g)^{\alpha r x' s_1} \cdot e(g, g)^{\alpha r y' s_1} \cdot e(g, h)^{\alpha r(x'+y')s_1}} \\ &= e(g, gh)^{\beta(x'+y')s_1} \cdot e(g, gh)^{z's_1} \\ &= C^{SK_0} \cdot e(g^{s_1}, SK_3). \end{aligned} \tag{16}$$

The CS rewards every idle user involved in this computing task and computes B as

$$\begin{aligned} B &= \frac{e(SK_4, g^{s_1}g^{s_2})}{R_1^{\frac{1}{d_1}} \cdot e(SK_5, h^{s_1}h^{s_2})} \\ &= \frac{e(g^{\alpha r x'}h^{r'x'}, g^{s_1}g^{s_2})}{e(g, g)^{\frac{\alpha r x' s_1 d_1}{d_1}} \cdot e(g^{r'x'}, h^{s_1}h^{s_2})} \\ &= e(g, g)^{\alpha r x' s_2}. \end{aligned} \tag{17}$$

3. Finally, the CS submits the partial decrypted $CT'' = \{\tilde{C}, g^{s_2}, \frac{A'}{B}\}$ to the EU.

- **Dec**$(CT'', SK) \rightarrow M$: The EU derives M as

$$\frac{\tilde{C}}{(\frac{A'}{B \cdot e(g^{s_2}, g^{z'})})^{\frac{1}{x'}}} = \frac{M \cdot e(g,g)^{\beta s_2}}{(\frac{e(g,g)^{(\alpha r+\beta)x's_2+z's_2}}{e(g,g)^{\alpha r x' s_2} \cdot e(g^{s_2}, g^{z'})})^{\frac{1}{x'}}} = M. \tag{18}$$

Remark 1. Compared with the traditional CP-ABE schemes, we introduce the concept of Lagrange-route product into the encryption algorithm, which will cause the DO to increase operations of multiplication while encrypting the data, but greatly reduce exponential operations when the EU decrypts the data. Because EU doesn't need to recursively conduct the exponential operations associated with the Lagrange coefficients during the decryption process.

Remark 2. In the algorithm **Tran**, the CS picks d_1, d_2 to randomize C_y of CT. It actually increases the computational overhead of the CS because of a lot of additional exponential operations. However, this algorithm can be run at any time before the ciphertext needs to be partially decrypted, such as after uploading the ciphertext or when the cloud's computing pressure is not high. Moreover, each ciphertext only needs to be randomized once, instead of being randomized each time it is decrypted. The randomization process is necessary. If the CS doesn't do so, then $R = R_1 \cdot R_2$ and the IU might compute the correct R_1, R_2 and deceive the CS with $R_1' = R_1 \cdot R_3$ and $R_2' = R_2 \cdot {R_3}^{-1}$, where $R_3 \in_R \mathbb{G}_T^*$

Remark 3. In many practical situations, the end user will first save the ciphertext that may be used in the future in his/her storage account, and then decrypt it when it is actually used later. In this case, the CS can reverse outsource the randomized ciphertext to idle users for partial decryption. However, in other application scenarios, the end user needs the cloud to partially decrypt the selected ciphertext immediately. In this case, the CS directly performs the partial decryption operations on the randomized/unrandomized ciphertext by its own just like [8].

5 Analysis of Reverse Outsourced CP-ABE Scheme

In this section, we provide a security analysis of our proposed reverse outsourced CP-ABE scheme and demonstrate its performance from in a theoretical point of view.

5.1 Security Analysis

Theorem 1. *Supposed that a PPT adversary $\mathcal{A}$ can break the IND-CPA security of our proposed scheme with a non-negligible advantage $\epsilon > 0$, then there exists a PPT simulator $\mathcal{B}$ that can distinguish a DBDH tuple from a random tuple with an advantage $\frac{\epsilon}{2}$.*

Proof. Given the bilinear map parameter $(\mathbb{G}_0, \mathbb{G}_T, p, e, g)$. The DBDH challenger $\mathcal{C}$ selects $a', b', c' \in \mathbb{Z}_p$, $\theta \in \{0,1\}$, $\mathcal{R} \in \mathbb{G}_T$ at random. Let $\mathcal{Z} = e(g,g)^{a'b'c'}$, if $\theta = 0$, $\mathcal{R}$ else. Next, $\mathcal{C}$ sends $\mathcal{B}$ the tuple $\langle g, g^{a'}, g^{b'}, g^{c'}, \mathcal{Z}\rangle$. Then, $\mathcal{B}$ plays the role of challenger in the following security game.

- *Initialization*:$\mathcal{A}$ submits a challenge access structure $\mathbb{A}^*$ to $\mathcal{B}$.
- *Setup*: $\mathcal{B}$ chooses $\beta', t \in \mathbb{Z}_p$ at random and sets $h = g^t$, $g^\alpha = g^{a'}$, $e(g,g)^\beta = e(g,g)^{\beta'+a'b'} = e(g,g)^{\beta'} e(g^{a'}, g^{b'})$, $e(g,h)^\beta = (e(g,g)^\beta)^t$. For each attribute $a_j \in \mathcal{L}$, $\mathcal{B}$ picks a random $s_j \in Z_p$. If $a_j \in \mathbb{A}^*$, set $PK_j = g^{v_j} = g^{\frac{a'}{s_j}}$; otherwise, $PK_j = g^{v_j} = g^{s_j}$. Then, $\mathcal{B}$ sends $PK = \{\mathbb{G}_0, g, h, g^\alpha, e(g,g)^\beta, e(g,h)^\beta, \{PK_j \mid \forall a_j \in \mathcal{L}\}\}$ to $\mathcal{A}$.
- *Phase* 1: $\mathcal{A}$ adaptively submits any attribute set $S \in \mathcal{L}$ to $\mathcal{B}$ with the restriction that $S \nvDash \mathbb{A}^*$. In response, $\mathcal{B}$ picks $\hat{r}, \tilde{r}, x', y', z' \in_R \mathbb{Z}_p$ at random, and computes $g^r = \frac{g^{\hat{r}}}{g^{b'}} = g^{\hat{r}-b'}$, $SK_0 = x' + y'$, $SK_1 = g^{(\beta+\alpha r)x'+z'} = g^{(\beta'+a'b'+a'(\hat{r}-b'))x'+z'} = (g^{\beta'+a'\hat{r}})^{x'} \cdot g^{z'}$, $SK_2 = g^{(\beta+\alpha r)y'} = (g^{\beta'+a'\hat{r}})^{y'}$, $SK_3 = (gh)^{z'} = g^{(1+t)z'}$,$SK_4 = g^{\alpha r x'} h^{r'x'} = g^{\alpha' \hat{r} x'} h^{\tilde{r}x'}$,$SK_5 = g^{r'x'} = g^{\tilde{r}x'}$.
  For each $\mathbf{a}_j \in S$, if $\mathbf{a}_j \in \mathbb{A}^*$, $\mathcal{B}$ computes $SK_{1,j} = g^{\frac{\alpha r' x'}{v_j}} = g^{s_j x' \tilde{r}}$ and $SK_{2,j} = g^{\frac{\alpha r' y'}{v_j}} h^{\frac{\alpha r'(x'+y')}{v_j}} = g^{s_j y' \tilde{r}} h^{s_j(x'+y')\tilde{r}}$; otherwise, $SK_{1,j} = g^{\frac{\alpha r' x'}{v_j}} = g^{\frac{a' \tilde{r} x'}{s_j}}$ and $SK_{2,j} = g^{\frac{\alpha \tilde{r} y'}{v_j}} h^{\frac{\alpha r'(x'+y')}{v_j}} = g^{\frac{a' \tilde{r} y'}{s_j}} h^{\frac{a' \tilde{r}(x'+y')}{s_j}}$.
  Afterwards, $\mathcal{B}$ answers $\mathcal{A}$ with the corresponding secret key $SK = (x', z', SK_{pd})$, where the partial decryption key $SK_{pd} = \{SK_0, SK_1, SK_2, SK_3, SK_4, SK_5, \{SK_{1,j}, SK_{2,j} \mid \forall \mathbf{a}_j \in S\}\}$.
- *Challenge*: $\mathcal{A}$ submits two equal-length challenge messages (M_0, M_1) to $\mathcal{B}$. Then, $\mathcal{B}$ describes the challenge access structure $\mathbb{A}^*$ as an access tree $\mathcal{T}^*$, picks $s_1 \in_R \mathbb{Z}_p$ and sets $g^{s_2} = g^{c'}, h^{s_2} = g^{tc'}$, $e(g,g)^{\beta s_2} = \mathcal{Z} \cdot e(g,g)^{\beta' c'}$, $e(g,h)^{\beta s_2} = \mathcal{Z}^t \cdot e(g,g)^{\beta' c' t}$, $e(g, gh)^{\beta s_2} = e(g,g)^{\beta s_2 (1+t)}$.
  Finally, $\mathcal{B}$ randomly picks $\theta' \in_R \{0,1\}$, and the challenge ciphertext CT^* is constructed as $\Big\{\mathcal{T}^*, \tilde{C}^* = M_{\theta'} \cdot e(g,g)^{\beta s_2}, C^* = e(g,gh)^{\beta s_1}, g^{s_1}, g^{s_2}, h^{s_1}, h^{s_2}, \{C_y^* = g^{v_j q_y(0) \pi_y} \mid \forall y \in Y^*, \mathbf{a}_j = \mathsf{att}(y) \in \mathcal{X}^*\}\Big\}$.
- *Phase* 2: This phase is the same as Phase 1.
- *Guess*: $\mathcal{A}$ outputs a guess bit θ'' of θ'. If $\theta'' = \theta'$, $\mathcal{B}$ guesses $\theta = 0$ which indicates that $\mathcal{Z} = e(g,g)^{a'b'c'}$ in the above game. Otherwise, $\mathcal{B}$ guesses $\theta = 1$ i.e., $\mathcal{Z} = \mathcal{R}$.

If $\mathcal{Z} = e(g,g)^{a'b'c'}$, then CT^* is available and $\mathcal{A}'s$ advantage of guessing θ' is ϵ. Therefore, $\mathcal{B}'s$ probability to guess θ correctly is

$$\Pr\left[\mathcal{B}\left(g, g^{a'}, g^{b'}, g^{c'}, \mathcal{Z} = e(g,g)^{a'b'c'}\right) = 0\right] = \frac{1}{2} + \epsilon. \tag{19}$$

Else $\mathcal{Z} = \mathcal{R}$, then CT^* is random from the view of $\mathcal{A}$. Hence, $\mathcal{B}'s$ probability to guess θ correctly is

$$\Pr\left[\mathcal{B}\left(g, g^{a'}, g^{b'}, g^{c'}, \mathcal{Z} = \mathcal{R}\right) = 1\right] = \frac{1}{2}. \tag{20}$$

In conclusion, $\mathcal{B}'s$ advantage to win the above security game is

$$\begin{aligned} Adv(\mathcal{B}) = \frac{1}{2}\Big(& \Pr\left[\mathcal{B}\left(g, g^{a'}, g^{b'}, g^{c'}, \mathcal{Z} = e(g,g)^{a'b'c'}\right) = 0\right] \\ & + \Pr\left[\mathcal{B}\left(g, g^{a'}, g^{b'}, g^{c'}, \mathcal{Z} = \mathcal{R}\right) = 1\right]\Big) - \frac{1}{2} = \frac{1}{2}\epsilon. \end{aligned} \tag{21}$$

□

5.2 Strategy Analysis

Theorem 2. *In our reverse outsourced CP-ABE scheme,* $R = \prod_{i\in[1,2]} {R_i}^{\frac{1}{d_i}}$, *the probability that* IU *does not follow the protocol but returns the sub-results* $\{R_i^*\}_{i\in[1,2]}$ *such that* $\{R_i^* \neq R_i\}_{i\in[1,2]}$ *and* $R = \prod_{i\in[1,2]} {R_i^*}^{\frac{1}{d_i}}$ *is negligible.*

Proof. Without loss of generality, we assume that idle users $\{\mathsf{IU}_i\}$ generate two computing results R_1^* and R_2^* to deceive the CS. Since $d_1, d_2 \in_R \mathbb{Z}_p^*$ are kept secret and $R = \prod_{i\in[1,2]} {R_i}^{\frac{1}{d_i}}$ is random in the view of idle users, they can successfully trick the CS with the following probability

$$\begin{aligned} & \Pr_{R_i^* \in \mathbb{G}_T^* \setminus R_i, i\in[1,2]}[R = \prod_{i\in[1,2]} {R_i^*}^{\frac{1}{d_i}}] \\ = & \sum_{t_1^* \in \mathbb{G}_T^* \setminus \{R_1\}} \Pr[R_1^* = t_1^*] \cdot \Pr[R_2^* = (\frac{R}{{R_1^*}^{\frac{1}{d_1}}})^{d_2} \mid R_1^* = t_1^*] \\ = & \frac{1}{p-1}. \end{aligned} \tag{22}$$

□

In our reverse outsourced CP-ABE scheme, if any IU_i cheats, for example, submitting an incorrect sub-result, the cheating behavior can be detected by the CS. Then, all participants $\{\mathsf{IU}_i\}_{i\in[1,2]}$ will not be rewarded. Thus, the utility of IU_i for choosing a strategy $st_i \in \{st_i^0, st_i^1\}$ satisfies

$$ut_i(st_i \mid ST \setminus \{st_i\}) = \begin{cases} ut_i^+, & st_i = st_i^1 \text{ and } \forall st_j = st_j^1 \text{ for } st_j \in ST \setminus \{st_i\}; \\ ut_i^-, & st_i = st_i^0 \text{ or } \exists st_j = st_j^0 \text{ for } st_j \in ST \setminus \{st_i\}. \end{cases}$$

The utility of IU_i reaches the maximum when all participants $\{\mathsf{IU}_i\}_{i\in[1,2]}$ follow the protocol specification. According to Nash equilibrium theory, following the protocol honestly and returning the correct sub-result, is the best strategy for each IU_i . Any other strategy will not only decrease the utility of each IU_i but waste his/her computational resources. Therefore, in the rational idle user model, in order to maximize it's own profits, each idle user has to follow the protocol specification of our reverse outsourced CP-ABE scheme.

6 Conclusion

In this paper, we propose a heuristic primitive called reverse outsourcing, and we also provide a reverse outsourcing CP-ABE scheme. In our scheme, the decryption work is outsourced from end user to the cloud and then to idle users, who have some online devices with a certain amount of computing power and not in use. A challenging issue is how to ensure that idle users return the correct computing results. To deal with this issue, we apply our scheme to the rational idle user model, in which idle users will be rewarded if they all return correct results and they prefer earning rewards to saving computing power, and the ciphertext will be randomized by the cloud so that any idle user cannot deceive the cloud. Therefore, the computational overhead at both end users and cloud sides is being minimized. During this paper, we only focus on reverse outsourced CP-ABE. The same approach applies to reverse outsourced KP-ABE, which we will omit here in order to keep the paper compact.

Acknowledgments

The authors are supported by National Cryptography Development Fund (Grant No. MMJJ20180210) and National Natural Science Foundation of China (Grant No. 61832012 and No. 61672019).

References

1. Attrapadung, N.: Dual system encryption via doubly selective security: Framework, fully secure functional encryption for regular languages, and more. In: Nguyen, P.Q., Oswald, E. (eds.) Advances in Cryptology - EUROCRYPT 2014 - 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, May 11-15, 2014. Proceedings. Lecture Notes in Computer Science, vol. 8441, pp. 557–577. Springer (2014)
2. Attrapadung, N., Libert, B., de Panafieu, E.: Expressive key-policy attribute-based encryption with constant-size ciphertexts. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) Public Key Cryptography - PKC 2011 - 14th International Conference on Practice and Theory in Public Key Cryptography, Taormina, Italy, March 6-9, 2011. Proceedings. Lecture Notes in Computer Science, vol. 6571, pp. 90–108. Springer (2011)
3. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: 2007 IEEE Symposium on Security and Privacy (S&P 2007), 20-23 May 2007, Oakland, California, USA. pp. 321–334 (2007)
4. Chase, M.: Multi-authority attribute based encryption. In: Vadhan, S.P. (ed.) Theory of Cryptography, 4th Theory of Cryptography Conference, TCC 2007, Amsterdam, The Netherlands, February 21-24, 2007, Proceedings. Lecture Notes in Computer Science, vol. 4392, pp. 515–534. Springer (2007)
5. Chase, M., Chow, S.S.M.: Improving privacy and security in multi-authority attribute-based encryption. In: Al-Shaer, E., Jha, S., Keromytis, A.D. (eds.) Proceedings of the 2009 ACM Conference on Computer and Communications Security, CCS 2009, Chicago, Illinois, USA, November 9-13, 2009. pp. 121–130. ACM (2009)

6. Gamal, T.E.: A public key cryptosystem and a signature scheme based on discrete logarithms. In: Blakley, G.R., Chaum, D. (eds.) Advances in Cryptology, Proceedings of CRYPTO '84, Santa Barbara, California, USA, August 19-22, 1984, Proceedings. Lecture Notes in Computer Science, vol. 196, pp. 10–18. Springer (1984)
7. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Proceedings of the 13th ACM Conference on Computer and Communications Security, CCS 2006, Alexandria, VA, USA, Ioctober 30 - November 3, 2006. pp. 89–98 (2006)
8. Green, M., Hohenberger, S., Waters, B.: Outsourcing the decryption of ABE ciphertexts. In: 20th USENIX Security Symposium, San Francisco, CA, USA, August 8-12, 2011, Proceedings (2011)
9. Halpern, J.Y., Teague, V.: Rational secret sharing and multiparty computation: extended abstract. In: Proceedings of the 36th Annual ACM Symposium on Theory of Computing, Chicago, IL, USA, June 13-16, 2004. pp. 623–632 (2004)
10. Kol, G., Naor, M.: Cryptography and game theory: Designing protocols for exchanging information. In: Theory of Cryptography, Fifth Theory of Cryptography Conference, TCC 2008, New York, USA, March 19-21, 2008. pp. 320–339 (2008)
11. Lewko, A.B., Okamoto, T., Sahai, A., Takashima, K., Waters, B.: Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In: Gilbert, H. (ed.) Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Monaco / French Riviera, May 30 - June 3, 2010. Proceedings. Lecture Notes in Computer Science, vol. 6110, pp. 62–91. Springer (2010)
12. Lewko, A.B., Waters, B.: Decentralizing attribute-based encryption. In: Paterson, K.G. (ed.) Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings. Lecture Notes in Computer Science, vol. 6632, pp. 568–588. Springer (2011)
13. Lewko, A.B., Waters, B.: Unbounded HIBE and attribute-based encryption. In: Paterson, K.G. (ed.) Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings. Lecture Notes in Computer Science, vol. 6632, pp. 547–567. Springer (2011)
14. Lewko, A.B., Waters, B.: New proof methods for attribute-based encryption: Achieving full security through selective techniques. In: Safavi-Naini, R., Canetti, R. (eds.) Advances in Cryptology - CRYPTO 2012 - 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings. Lecture Notes in Computer Science, vol. 7417, pp. 180–198. Springer (2012)
15. Li, J., Chen, X., Li, J., Jia, C., Ma, J., Lou, W.: Fine-grained access control system based on outsourced attribute-based encryption. In: Crampton, J., Jajodia, S., Mayes, K. (eds.) Computer Security - ESORICS 2013 - 18th European Symposium on Research in Computer Security, Egham, UK, September 9-13, 2013. Proceedings. Lecture Notes in Computer Science, vol. 8134, pp. 592–609. Springer (2013)
16. Okamoto, T., Takashima, K.: Fully secure unbounded inner-product and attribute-based encryption. In: Wang, X., Sako, K. (eds.) Advances in Cryptology - ASIACRYPT 2012 - 18th International Conference on the Theory and Application of Cryptology and Information Security, Beijing, China, December 2-6, 2012. Pro-

ceedings. Lecture Notes in Computer Science, vol. 7658, pp. 349–366. Springer (2012)
17. Rouselakis, Y., Waters, B.: Practical constructions and new proof methods for large universe attribute-based encryption. In: Sadeghi, A., Gligor, V.D., Yung, M. (eds.) 2013 ACM SIGSAC Conference on Computer and Communications Security, CCS'13, Berlin, Germany, November 4-8, 2013. pp. 463–474. ACM (2013)
18. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings. pp. 457–473 (2005)
19. Waters, B.: Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In: Public Key Cryptography - PKC 2011 - 14th International Conference on Practice and Theory in Public Key Cryptography, Taormina, Italy, March 6-9, 2011. Proceedings. Lecture Notes in Computer Science, vol. 6571, pp. 53–70. Springer (2011)
20. Waters, B.: Functional encryption for regular languages. In: Safavi-Naini, R., Canetti, R. (eds.) Advances in Cryptology - CRYPTO 2012 - 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings. Lecture Notes in Computer Science, vol. 7417, pp. 218–235. Springer (2012)
21. Yu, S., Wang, C., Ren, K., Lou, W.: Achieving secure, scalable, and fine-grained data access control in cloud computing. In: INFOCOM 2010. 29th IEEE International Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies, 15-19 March 2010, San Diego, CA, USA. pp. 534–542. IEEE (2010)