10-2016

Attribute-based encryption with granular revocation

Hui CUI
Singapore Management University, hcui@smu.edu.sg

DENG, Robert H.
Singapore Management University, robertdeng@smu.edu.sg

Xuhua DING
Singapore Management University, xhding@smu.edu.sg

Yingjiu LI
Singapore Management University, yjli@smu.edu.sg

Citation

CUI, Hui; DENG, Robert H.; DING, Xuhua; and Yingjiu LI. Attribute-based encryption with granular revocation. (2016). *Security and Privacy in Communication Networks: 12th International Conference, SecureComm 2016, Guangzhou, China, October 10-12: Proceedings*. 198, 165-181.
Available at: https://ink.library.smu.edu.sg/sis_research/3399

Attribute-Based Encryption with Granular Revocation

Hui Cui, Robert H. Deng, Xuhua Ding, and Yingjiu Li

Secure Mobile Centre, School of Information Systems, Singapore Management University, Singapore
hcui@smu.edu.sg,robertdeng@smu.edu.sg,xhding@smu.edu.sg,yjli@smu.edu.sg

Abstract. Attribute-based encryption (ABE) enables an access control mechanism over encrypted data by specifying access policies over attributes associated with private keys or ciphertexts, which is a promising solution to protect data privacy in cloud storage services. As an encryption system that involves many data users whose attributes might change over time, it is essential to provide a mechanism to selectively revoke data users' attributes in an ABE system. However, most of the previous revokable ABE schemes consider how to disable revoked data users to access (newly) encrypted data in the system, and there are few of them that can be used to revoke one or more attributes of a data user while keeping this user active in the system. Due to this observation, in this paper, we focus on designing ABE schemes supporting selective revocation, i.e., a data user's attributes can be selectively revoked, which we call ABE with granular revocation (ABE-GR). Our idea is to utilize the key separation technique, such that for any data user, key elements corresponding to his/her attributes are generated separately but are linkable to each other. To begin with, we give a basic ABE-GR scheme to accomplish selective revocation using the binary tree data structure. Then, to further improve the efficiency, we present a server-aided ABE-GR scheme, where an untrusted server is introduced to the system to mitigate data users' workloads during the key update phase. Both of the ABE-GR constructions are formally proved to be secure under our defined security model.

Key words: Granular revocation, ABE, Efficiency, Cloud storage

1 Introduction

Attribute-based encryption (ABE) [21] provides a promising solution to preserve data privacy in a scenario (e.g., cloud storage services [23]) where data users are identified by their attributes (or credentials), and data owners want to share their data according to some policy based on the attributes of data users. In a ciphertext-policy ABE (CP-ABE) system, each data user is given a private attribute-key reflecting his/her attributes generated by the attribute authority (AA), and each data owner specifies an access policy to the message over a set of attributes[1]. A data user will be able to decrypt a ciphertext if and only if the attributes ascribed to his/her private attribute-key satisfy the access policy (or access structure) associated with the ciphertext. Though ABE favorably solves the problem arising in the situations where different users with different attributes are given access to different levels of the encrypted data, it fails to address the issue of dynamic credentials where the attributes of every data user change with time. This challenge motivates the study of revocation mechanisms [1], where a periodical key update process disables revoked data users to update their decryption keys to decrypt newly encrypted data.

In terms of attribute-based setting, user revocation is divided into indirect revocation and direct revocation [1]. Regarding indirect revocation, one solution is to ask data users to periodically renew their private attribute-keys [7], but this requires the update key size to be $O(N-R)$ group elements where N is the number of all users and R is the number of revoked users. To reduce the cost of key update from linear to logarithmic (i.e., $O(R\log(\frac{N}{R})))$, Boldyreva, Goyal and Kumar [5] put forth a revocation methodology by combining the fuzzy IBE scheme [21] with the binary tree data structure [18] where the AA publicly broadcasts the key update information for each time period, but only non-revoked data users can update their decryption keys to decrypt a newly generated ciphertext. In direct revocation [2, 1], data owners possess a current revocation list, and specify the revocation list directly when running the encrypting algorithm so that user revocation can be done instantly without requiring the key update phase as in the indirect method[2]. There are also constructions (e.g., [25]) that delegate the direct revocation ability to a semi-trusted server who cannot collude with the data users, where the server helps data users with decryption but terminates the decryption operation for any revoked data user.

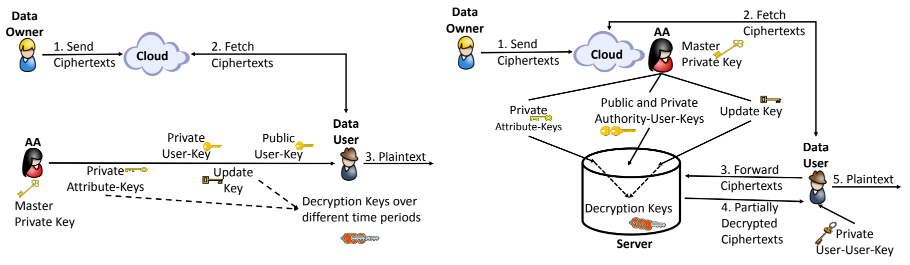
Since an attribute-based encryption system might involve a large number of data users whose attributes change over time, it is desirable to build an attribute-based encryption scheme that the credentials possessed by data users can be selectively revoked. However, most of the previous revocable ABE systems [5, 2, 1, 20, 25] only consider efficient user revocation to prevent revoked data users from accessing the encrypted data, and there is little attention on how to independently revoke one or more attributes from a data user, i.e., selective revocation on attributes. Due to this observation, in this paper, we focus on the design of efficient and revocable attribute-based encryption schemes where the

[1] There are two complimentary forms of ABE: CP-ABE and key-policy ABE (KP-ABE). In a KP-ABE system, the situation is reversed that the keys are associated with the access policies and the ciphertexts are associated with the attributes. In the rest of this paper, unless otherwise specified, what we talk about is CP-ABE.

[2] Note that direct revocation can be done immediately without the key update process which asks for the communication from the AA to all the non-revoked users over all the time periods, but it requires all the data owners to keep the current revocation list. This makes the system impurely attribute-based, since data owners in the attribute-based setting create ciphertext based only on attributes without caring revocation. In this paper, unless otherwise specified, the revocation mechanism we talk about is indirect revocation.

attributes possessed by each data user can be selectively revoked via a periodical key update phase, which we call attribute-based encryption with granular revocation (ABE-GR). Notice that ABE-GR can achieve user revocation by revoking all credentials possessed by a data user.

1.1 Our Contributions



(a) ABE-GR (b) Server-Aided ABE-GR

Fig. 1: System architectures of ABE-GR (left) and server-aided ABE-GR (right).

We describe the system architecture of an ABE-GR scheme in Fig. 1-(a). In an ABE-GR system, each data user's key is divided into three parts: private user-key (with a corresponding public user-key), private attribute-keys (associated with different attributes) and public key update information, from the latter two of which a data user can extract a decryption key. The AA, who keeps the master private key and publishes the public parameter, is responsible for the distribution of personalized pairs of public and private user-keys and private attribute-keys. In addition, the AA regularly posts the public key update information. Each data owner encrypts a message under an access structure and a time period using the public parameter. To decrypt a newly generated ciphertext, a data user needs to possess a pair of public and private user-keys as well as a decryption key on the current time period satisfying the access policy of this ciphertext. The key challenge in building an ABE-GR scheme is to prevent a data user from using his/her revoked attributes to decrypt any newly generated ciphertext. Traditionally, in an ABE scheme, each attribute possessed by a data user corresponds to one element in his/her private attribute-key, and these key elements are tied together through a random value. In order to support granular revocation in ABE, we need a technique to enable different key components on different attributes to be created separately but linkable to each other. Thanks to the key separation technique in distributed ABE [17] where the task of the single AA is split across multiple AAs and each attribute is controlled by one specific AA, we can equip an ABE scheme with a similar technique but under a single AA. Thus, each key component associated with the corresponding attribute will be created separately, but they still bind together due to the sharing of the same identification information (i.e., the public user-key) which is unique to each data user. As a result, we build an ABE-GR scheme by combining an ABE scheme with the key separation technique in distributed ABE [17]. To reduce the size of key update for the AA from linear to logarithmic in the number of users, we apply the binary tree data structure [18] in the algorithms of our ABE-GR scheme. Details about this ABE-GR scheme, which we will refer to as a basic ABE-GR scheme, is given in Section 4.

As alluded in [19], binary tree data structure [18] is useful in alleviating the workload of the AA, but it could not mitigate the workload of each data user who needs to periodically update the decryption key. Is it possible to fix the keys stored by data users such that they are not required to frequently update their decryption keys while without affecting the revocation? To give an affirmative answer to this question, we bring in an untrusted server[3] to the basic ABE-GR system to mitigate the workloads of data users. We depict the system architecture in Fig. 1-(b), which involves four entities: an AA, data owners, data users, and a server. Different from that in the basic ABE-GR construction, the public and private user-key pair is divided into a pair of public and private user-user-keys and a pair of public and private authority-user-keys, of which the former is generated by each data user himself/herself[4] and the latter is extracted by the AA based on the public user-user-key. The server is given the public and private authority-user-keys and private attribute-keys of data users as well as the key update information. A data user fetches a ciphertext from the cloud, and sends it to the server for partial decryption. For any non-revoked user, from the private attribute-keys and key update information, the server can generate a collection of decryption keys (associated with a set of attributes), which, combining with the public and private authority-user-keys, can partially decrypt a ciphertext forwarded by this user if his/her non-revoked attributes satisfy the access structure ascribed to the ciphertext. A data user can obtain the plaintext by decrypting the partially decrypted ciphertext using his/her self-generated private user-user-key. This does not compromise the security, because the public user-user-key is embedded in the private authority-user-key, the server cannot fully unwrap the ciphertext without the private user-user-key. A detailed description of the construction is presented in Section 4.

Since both our constructions are built on an ABE scheme that is selectively secure [24, 17], where the adversary has to commit the challenge access structure in advance, we can only achieve selective security in our ABE-GR schemes. Note that the techniques can be applied to fully secure ABE schemes (e.g., [20]) to obtain fully secure (server-aided) ABE-GR schemes.

[3] The server is untrusted in the sense that it honestly follows the protocol but without holding any secret information (i.e., it may collude with data users). Besides, all operations done by the server can be performed by anyone, including data users (i.e., any dishonest behaviour from the server can be easily detected).

[4] This pair of user-user-keys can also be generated by the AA, but this requires a secure channel between each data user and the AA for private key distribution.

1.2 Related Work

Revocable IBE. With regard to revocable IBE, Boneh and Franklin [7] suggested that users renew their private keys periodically, but this requires that all users have to regularly contact the key generation centre (KGC) to obtain new private keys, and a secure channel must be established between the KGC and each user for such transactions. Hanaoka et al. [10] presented a convenient method for the users to periodically renew their private keys without interacting with the KGC, where the KGC publicly posts the key update information. However, each user needs to posses a tamper-resistant hardware device, making the solution very cumbersome. Boldyreva, Goyal and Kumar [5] presented an efficient revocable IBE scheme to reduce the size of key update from linear to logarithmic, but all non-revoked users need to periodically update their decryption keys. To better address the revocation issue, revocation with a third party [6, 9, 16, 3, 15, 13, 19] has been proposed, where a semi-trusted[5] (or untrusted) third party is assigned to hold the shares of all users' private keys and help users decrypt each ciphertext. Once an identity is revoked, the mediator stops helping the user with decryption.

Revocable ABE. Regarding user revocation in ABE, there are two revocation mechanisms [1, 8]: direct and indirect revocation. Considering direct revocation, in which data owners keep the current revocation list, and specify the revocation list directly when encrypting, there are schemes in [2, 14, 11]. In addition, Yang et al. [25] proposed an approach by asking a semi-trusted server to share the decryption capability with data users, and thus when a data user is revoked, the server terminates the decryption for the user. Regarding indirect revocation, which we intend to achieve in this paper, Boldyreva, Goyal and Kumar [5] proposed a revocable KP-ABE scheme where the AA indirectly enables the revocation by forcing revoked users to be unable to update their keys. Later, based on the same technique adopted in [5], Attrapadung and Imai [1] raised a hybrid revocable KP-ABE system under selective security model which allows a data owner to select whether to use either direct or indirect revocation mode when encrypting a message. Sahai, Seyalioglu and Waters [20] provided a generic way to build ABE schemes supporting dynamic credentials, where the AA indirectly accomplishes revocation by stopping updating the keys for revoked users. Cui and Deng [8] gave two indirectly revocable and decentralized ABE schemes in composite-order groups where the AA'role is split across multiple AAs.

1.3 Organization

The remainder of this paper is organized as follows. In Section 2, we briefly review the notions and definitions relevant to this paper. In Section 3, we describe the framework for ABE-GR, and then present its security model. In Section 4, we

[5] In this paper, unless otherwise specified, "semi-trusted" means that the security is guaranteed under the assumption that the corresponding entity is disallowed to collude with the malicious data users.

give two concrete constructions of ABE-GR, and provably reduce their security. In addition, we compare our ABE-GR schemes with previous revocable ABE schemes in Section 4. We conclude the paper in Section 5.

2 Preliminaries

In this section, we review some basic cryptographic notions and definitions that are to be used in this paper.

2.1 Bilinear Pairings and Complexity Assumptions

Let p be a prime number, and G be a group of order p that is generated from g. We define $\hat{e} : G \times G \rightarrow G_1$ to be a bilinear map if it has two properties [7].

– Bilinear: for all $g \in G$, and a, $b \in Z_p^*$, we have $\hat{e}(g^a, g^b) = \hat{e}(g, g)^{ab}$.
– Non-degenerate: $\hat{e}(g, g) \neq 1$.

We say that G is a bilinear group if the group operation in G is efficiently computable and there exists a group G_1 and an efficiently computable bilinear map $\hat{e} : G \times G \rightarrow G_1$ as above.

Decisional Parallel Bilinear Diffie-Hellman Exponent Assumption [24]. The decisional q-parallel bilinear Diffie-Hellman exponent (BDHE) problem is that for any probabilistic polynomial-time algorithm, given $\overrightarrow{y} =$

$$\begin{array}{ll} & g, g^{\mu}, g^{a}, ..., g^{a^q}, g^{a^{q+2}}, ..., g^{a^{2q}}, \\ \forall\ j \in [1, q] & g^{\mu \cdot b_j}, g^{a/b_j}, ..., g^{a^q/b_j}, g^{a^{q+2}/b_j}, ..., g^{a^{2q}/b_j}, \\ \forall 1 \leq j, k \leq q, k \neq j & g^{a \cdot \mu \cdot b_k/b_j}, ..., g^{a^q \cdot \mu \cdot b_k/b_j}, \end{array}$$

it is difficult to distinguish $(\overrightarrow{y}, \hat{e}(g, g)^{a^{q+1}\mu})$ from $(\overrightarrow{y}, Z)$, where $g \in G$, $Z \in G_1$, a, μ, b_1, ..., $b_q \in Z_p$ are chosen independently and uniformly at random.

2.2 Access Structures and Linear Secret Sharing

Definition 1. ***(Access Structure [12, 24])****. Let $\{P_1, ..., P_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^{\{P_1,...,P_n\}}$ is monotone if $\forall B$, C : if $B \in \mathbb{A}$ and $B \subseteq C$, then $C \subseteq \mathbb{A}$. A monotone access structure is a monotone collection $\mathbb{A}$ of non-empty subsets of $\{P_1, ..., P_n\}$, i.e., $\mathbb{A} \subseteq 2^{\{P_1,...,P_n\}} \backslash \{\emptyset\}$. The sets in $\mathbb{A}$ are called the authorized sets, and the sets not in $\mathbb{A}$ are called the unauthorized sets.*

Definition 2. ***(Linear Secret Sharing Schemes (LSSS) [12, 24])****. Let P be a set of parties. Let $\mathbb{M}$ be a matrix of size $l \times n$. Let $\rho : \{1, ..., l\} \rightarrow P$ be a function that maps a row to a party for labeling. A secret sharing scheme Π over a set of parties P is a linear secret-sharing scheme over Z_p if*

1. The shares for each party form a vector over Z_p.

2. *There exists a matrix $\mathbb{M}$ which has l rows and n columns called the share-generating matrix for Π. For $x = 1, ..., l$, the x-th row of matrix $\mathbb{M}$ is labeled by a party $\rho(i)$, where $\rho : \{1, ..., l\} \to P$ is a function that maps a row to a party for labeling. Considering that the column vector $\overrightarrow{v} = (\mu, r_2, ..., r_n)$, where $\mu \in Z_p$ is the secret to be shared and $r_2, ..., r_n \in Z_p$ are randomly chosen, then $\mathbb{M}\overrightarrow{v}$ is the vector of l shares of the secret μ according to Π. The share $(\mathbb{M}\overrightarrow{v})_i$ belongs to party $\rho(i)$.*

It has been noted in [12] that every LSSS also enjoys the linear reconstruction property. Suppose that Π is an LSSS for access structure $\mathbb{A}$. Let $\mathbf{A}$ be an authorized set, and define $I \subseteq \{1, ..., l\}$ as $I = \{i|\rho(i) \in \mathbf{A}\}$. Then the vector (1, 0, ..., 0) is in the span of rows of matrix $\mathbb{M}$ indexed by I, and there exist constants $\{w_i \in Z_p\}_{i \in I}$ such that, for any valid shares $\{v_i\}$ of a secret μ according to Π, we have $\sum_{i \in I} w_i v_i = \mu$. These constants $\{w_i\}$ can be found in polynomial time with respect to the size of the share-generating matrix $\mathbb{M}$ [4].

Boolean Formulas [12]. Access policies can also be described in terms of monotonic boolean formulas. LSSS access structures are more general, and can be derived from representations as boolean formulas. There are standard techniques to convert any monotonic boolean formula into a corresponding LSSS matrix. The boolean formula can be represented as an access tree, where the interior nodes are AND and OR gates, and the leaf nodes correspond to attributes. The number of rows in the corresponding LSSS matrix will be the same as the number of leaf nodes in the access tree.

2.3 Binary Tree

We recall the definitions related to binary tree in [5, 19]. Denote BT by a binary tree with N leaves corresponding to N users. Let **root** be the root node of the tree BT. If θ is a leaf node, then Path(θ) denotes the set of nodes on the path from θ to **root**, which includes both θ and **root**. If θ is a non-leaf node, then θ_l, θ_r denote left and right child of θ. Assume that nodes in the tree are uniquely encoded as strings, and the tree is defined by all of its nodes descriptions. The KUNodes algorithm is used to compute the minimal set of nodes for which key update needs to be published so that only the non-revoked users at time period t are able to decrypt the ciphertexts. This algorithm takes a binary tree BT, a revocation list rl and a time period t as the input, and it outputs a set of nodes, which is the minimal set of nodes in BT such that none of the nodes in rl with corresponding time period at or before t (users revoked at or before t) have any ancestor (or, themselves) in the set, and all other leaf nodes (corresponding to non-revoked users) have exactly one ancestor (or, themselves) in the set. The KUNodes algorithm works as follows: it firstly marks all the ancestors of the revoked nodes as revoked, and then it outputs all the non-revoked children of revoked nodes. We give a formal definition as follows.

```
KUNodes(BT, $rl$, $t$)
  $X, Y \leftarrow \emptyset$.
  $\forall\ (\theta_i, t_i) \in rl,\ \text{if } t_i \leq t,\ \text{then add Path}(\theta_i) \text{ to } X$.
  $\forall\ x \in X,\ \text{if } x_l \notin X,\ \text{then add } x_l \text{ to } Y;\ \text{if } x_r \notin X, \text{then add } x_r \text{ to } Y$.
  If $Y = \emptyset$, then add root to $Y$.
  Return $Y$.
```

3 System Architecture and Security Model

In this section, we describe the system architecture and formal security definition of attribute-based encryption with granular revocation.

3.1 Framework

An ABE-GR scheme involves three entities: attribute authority (AA), data owners and data users, where the algorithms run by these parties are described as follows.

- GSetup$(1^\lambda) \rightarrow (par,\ msk)$. Taking a security parameter λ as the input, this algorithm outputs the public parameter par and the master private key msk. This algorithm is run by the AA.
- ASetup$(par,\ A_i) \rightarrow (\text{PK}_{A_i},\ \text{SK}_{A_i},\ rl_i,\ st_i)$. Taking the public parameter par and an attribute A_i as the input, this algorithm outputs a public and private key pair $(\text{PK}_{A_i}, \text{SK}_{A_i})$ along with an initially empty revocation list rl_i and a state st_i. This algorithm is run by the AA.
- UserKG$(par,\ msk,\ id) \rightarrow (sk_{id},\ pk_{id})$. Taking the public parameter par, the master private key msk and an identity id as the input, this algorithm outputs a private and public user-key pair $(sk_{id},\ pk_{id})$ for user id. This algorithm is run by the AA.
- PrivKG$(par, \text{SK}_{A_i}, pk_{id}, st_i) \rightarrow (pk_{id}^{A_i},\ st_i)$. Taking the public parameter par, the private key SK_{A_i}, a pubic user-key pk_{id} and a state st_i as the input, this algorithm outputs a private attribute-key $pk_{id}^{A_i}$ and an updated state st_i for user id possessing an attribute A_i. This algorithm is run by the AA.
- TKeyUp$(par,\ \text{SK}_{A_i},\ t,\ rl_i,\ st_i) \rightarrow (ku_t^{(i)},\ st_i)$. Taking the public parameter par, the private key SK_{A_i}, a time period t, a revocation list rl_i and a state st_i as the input, this algorithm outputs the key update information $ku_t^{(i)}$ and an updated state st_i. This algorithm is run by the AA.
- DecKG$(par, pk_{id}^{A_i}, tku_t^{(i)}) \rightarrow dk_{id,t}^{(i)}$. Taking the public parameter par, a private attribute-key $pk_{id}^{A_i}$ and the key update information $tku_t^{(i)}$ as the input, this algorithm outputs a decryption key $dk_{id,t}^{(i)}$ for user id at time period t. This algorithm is run by each data user.

– Encrypt(par, ($\mathbb{M}$, ρ), $\{\mathrm{PK}_{A_i}\}$, t, M) $\rightarrow$ CT. Taking the public parameter par, an access structure ($\mathbb{M}$, ρ), a set of public keys $\{\mathrm{PK}_{A_i}\}$ for relevant attributes, a time period t and a message M as the input, this algorithm outputs a ciphertext CT. This algorithm is run by each data owner, and CT will be stored to the cloud.
– Decrypt(par, pk_{id}, sk_{id}, $\{dk_{id,t}^{(i)}\}$, CT) $\rightarrow M/\perp$. Taking the public parameter par, a public and private user-key pair (pk_{id}, sk_{id}), a collection of decryption keys $\{dk_{id,t}^{(i)}\}$ corresponding to the same id and a ciphertext CT as the input, this algorithm outputs a message M when the collection of attributes $\{A_i\}$ satisfies the access matrix corresponding to the ciphertext or a failure symbol $\perp$. This algorithm is run by each data user.
– Revoke(id, A_i, t, rl_i, st_i) $\rightarrow rl_i$. Taking an attribute A_i of identity id to be revoked, a time period t, a revocation list rl_i and a state st_i, this algorithm outputs an updated revocation list rl_i. This algorithm is run by the AA.

Notes and Comments. Note that in order to create public and private keys, private attribute-keys, key update information and decryption keys corresponding to multiple attributes, the corresponding algorithms ASetup, PrivKG, TKeyUp and DecKG are extended to take in many attributes by running the "single attribute" version once for each attribute.

The correctness of an ABE-GR scheme requires that for any security parameter λ and any message M, if the data user is not revoked at time period t, and if all the parties follow the described algorithms as above, then we have Decrypt(par, sk_{id}, $\{dk_{id,t}^{(i)}\}$, CT) $= M$ if $\{dk_{id,t}^{(i)}\}$ is a set of decryption keys for the same identity id over a set of attributes satisfying the access structure of the ciphertext CT.

3.2 Security Model

Below we describe the security definition of indistinguishability under chosen plaintext attacks, i.e., IND-CPA security, for ABE-GR between an adversary algorithm $\mathcal{A}$ and a challenger algorithm $\mathcal{B}$.

– Setup. Algorithm $\mathcal{B}$ runs the GSetup algorithm, and gives the public parameter par to algorithm $\mathcal{A}$ whilst keeps the master private key msk. In addition, algorithm $\mathcal{B}$ runs the ASetup algorithm, and keeps the private keys $\{\mathrm{SK}_{A_i}\}$, initially empty revocation lists $\{rl_i\}$ and states $\{st_i\}$ whilst gives algorithm $\mathcal{A}$ the public keys $\{\mathrm{PK}_{A_i}\}$.
– Phase 1. Algorithm $\mathcal{A}$ adaptively issues a sequence of queries to algorithm $\mathcal{B}$.
  1. Private-User-Key oracle. Algorithm $\mathcal{A}$ issues a private user-key query on an identity id. Algorithm $\mathcal{B}$ returns sk_{id} by running UserKG(par, msk, id), and adds (id, pk_{id}) to the user list.
  2. Private-Attribute-Key oracle. Algorithm $\mathcal{A}$ issues a public attribute-key query on an identity id with an attribute set $\{A_i\}$. Algorithm $\mathcal{B}$ returns $\{pk_{id}^{A_i}\}$ by running UserKG(par, msk, id) (if id does not exist in the user list), PrivKG(par, SK_{A_i}, pk_{id}, st_i) for each A_i of id.

3. Key-Update oracle. Algorithm $\mathcal{A}$ issues a key update query on a time period t. Algorithm $\mathcal{B}$ returns $\{tku_t^{(i)}\}$ by running TKeyUp(par, SK_{A_i} t, rl_i, st_i) for each A_i.
4. Decryption-Key oracle. Algorithm $\mathcal{A}$ issues a decryption key query on a time period t and an identity id with an attribute set $\{A_i\}$. Algorithm $\mathcal{B}$ returns $\{dk_{id,t}^{(i)}\}$ by running UserKG(par, msk, id) (if id does not exist in the user list), PrivKG(par, SK_{A_i}, pk_{id}, st_i), TKeyUp(par, SK_{A_i}, t, rl_i, st_i), DecKG(par, $pk_{id}^{A_i}$, $tku_t^{(i)}$) on each A_i of id. Note that this oracle cannot be queried on a time period t which has not been issued to the Key-Update oracle.
5. Revocation oracle. Algorithm $\mathcal{A}$ issues a revocation query on an attribute A_i of identity id and a time period t. Algorithm $\mathcal{B}$ runs Revoke(id, t, rl_i, st_i) and outputs an updated revocation list rl_i. Note that this oracle cannot be queried on a time period t if a key update query has been issued on t.

– Challenge. Algorithm $\mathcal{A}$ outputs two messages M_0^*, M_1^* of the same size, an access structure ($\mathbb{M}^*$, ρ^*) and a time period t^* with the following constraint. Denote $I_{\mathbb{M}^*,\rho^*} = \{\mathcal{I}_1, ..., \mathcal{I}_\chi\}$ by a set of minimum subsets of attributes satisfying ($\mathbb{M}^*$, ρ^*). For each id, if algorithm $\mathcal{A}$ asks for a collection of private attribute-keys on an attribute set covering an $\mathcal{I}_j \in I_{\mathbb{M}^*,\rho^*}$ ($j \in [1, \chi]$), then (1) the revocation oracle must be queried on some tuple (id, t, A_i) where t happens at or before t^* and $A_i \in \mathcal{I}_j$ ($j \in [1, \chi]$), and (2) the Decryption-Key oracle cannot be queried on (id, t, $\{A_i\}$) for any $t = t^*$ and $\mathcal{I}_j \subseteq \{A_i\}$ ($j \in [1, \chi]$). Algorithm $\mathcal{B}$ randomly chooses $\gamma \in \{0, 1\}^*$, runs Encrypt(par, ($\mathbb{M}^*$, ρ^*), $\{\text{PK}_{A_i}\}$, t^*, M_γ^*) to obtain the challenge ciphertext CT^*, and sends CT^* to algorithm $\mathcal{A}$.
– Phase 2. Algorithm $\mathcal{A}$ continues issuing a sequence of queries to algorithm $\mathcal{B}$ as in Phase 1, following the restriction defined in the Challenge phase.
– Guess. Algorithm $\mathcal{A}$ makes a guess γ' for γ, and it wins the game if $\gamma' = \gamma$.

The advantage of algorithm $\mathcal{A}$ in this game is defined to be $\Pr[\gamma = \gamma'] - 1/2$. We say that an ABE-GR scheme is secure under the IND-CPA security model if all probabilistic polynomial time (PPT) adversaries have at most a negligible advantage in the security parameter λ. In addition, an ABE-GR scheme is said to be selectively secure under the IND-CPA security model if an Init stage is added before the Setup phase where algorithm $\mathcal{A}$ commits to the challenge access structure ($\mathbb{M}^*$, ρ^*) which it attempts to attack.

Remarks. Note that our security definition is different from those presented in previous revocable ABE schemes. The definitions in [1, 20, 8] do not take a realistic threat called decryption key exposure attacks [22][6] into consideration, while our model allows an additional Decryption-Key oracle to cover such kind of attacks so that no information of the plaintext is revealed from a ciphertext even if all (short-term) decryption keys of different time periods are exposed.

[6] This does not contradict the security proofs of these schemes, because such attack is excluded from their security models.

4 Attribute-Based Encryption with Granular Revocation

In this section, we present two ABE-GR constructions and their security analysis. Also, we compare them with several existing revocable ABE schemes.

4.1 Basic Construction

Assume that the attribute space is Z_p, the time space is Z_p, and the message space is G_1. Our basic attribute-based encryption scheme supporting granular revocation is composed of the following algorithms, which is built upon the CP-ABE scheme presented in [24].

- GSetup. This algorithm takes a security parameter λ as the input. It randomly chooses a group G of prime order p with $g \in G$ being the corresponding generator, and defines a bilinear map $\hat{e} : G \times G \to G_1$. Additionally, it randomly chooses u, $h \in G$, a, $\alpha \in Z_p$. Define a function F to map an element y in Z_p to an element in G by $F(y) = u^y h$. The public parameter is $par = (g, g^a, u, h, \hat{e}(g,g)^\alpha)$. The master private key is $msk = \alpha$.
- ASetup. This algorithm takes the public parameter par and an attribute i as the input. It randomly chooses $\alpha_i \in Z_p$, and computes $\mathrm{PK}_{A_i} = g^{\alpha_i}$. Let rl_i be an empty list storing revoked users and BT_i be a binary tree with at least N leaf nodes. It outputs the public key PK_{A_i} along with rl_i and st_i where st_i is a state which is set to be BT_i, and keeps α_i as the private key SK_{A_i}.
- UserKG. This algorithm takes the public parameter par, the master private key msk and an identity id as the input. It randomly chooses $\beta \in Z_p$, and outputs a private and public user-key pair $(sk_{id}, pk_{id}) = (g^\alpha (g^a)^\beta, g^\beta)$.
- PrivKG. This algorithm takes the public parameter par, the private key SK_{A_i}, a public user-key pk_{id}, an attribute A_i with a private key SK_{A_i} and a state st_i as the input. It firstly chooses an undefined leaf node $\theta^{(i)}$ from the binary tree BT_i, and stores id in this node. Then, for each node $x^{(i)} \in \mathrm{Path}(\theta^{(i)})$, it runs as follows.
  1. It fetches $g_{i,x}$ from the node $x^{(i)}$. If $x^{(i)}$ is undefined, it randomly chooses $g_{i,x} \in G$, and computes $P_x^{(i)} = (g^\beta / g_{i,x})^{\alpha_i}$. It stores $g_{i,x}$ in the node $x^{(i)}$.
  2. It outputs the private attribute-key $pk_{id}^{A_i} = \{x^{(i)}, P_x^{(i)}\}_{x^{(i)} \in \mathrm{Path}(\theta^{(i)})}$ and an updated state st_i.
- TKeyUp. This algorithm takes the public parameter par, a private key SK_{A_i}, a time period t, a revocation list rl_i, and a state BT_i as the input. For all $x^{(i)} \in \mathrm{KUNodes}(\mathrm{BT}_i, rl_i, t)$, it fetches $g_{i,x}$[7] from the node x. Then, it randomly chooses $s_{i,x} \in Z_p$, and computes

$$Q_{x,1}^{(i)} = {g_{i,x}}^{\alpha_i} \cdot F(t)^{s_{i,x}}, \quad Q_{x,2}^{(i)} = g^{s_{i,x}}.$$

  It outputs $ku_t^{(i)} = \{x^{(i)}, Q_{x,1}^{(i)}, Q_{x,2}^{(i)}\}_{x^{(i)} \in \mathrm{KUNodes}(\mathrm{BT}_i,\, rl_i,\, t)}$ as the key update information.

[7] Note that $g_{i,x}$ is always predefined in the PrivKG algorithm.

– DecKG. This algorithm takes the public parameter par, a private attribute-key $pk_{id}^{A_i}$ and the key update information $tku_t^{(i)}$ as the input. It parses each $pk_{id}^{A_i}$ as $\{x^{(i)},\ P_x^{(i)}\}_{x^{(i)} \in I}$, $tku_t^{(i)}$ as $\{x^{(i)},\ Q_{x,1}^{(i)},\ Q_{x,2}^{(i)}\}_{x^{(i)} \in J}$ for some set of nodes $I = \mathrm{Path}(\theta^{(i)})$, $J = \mathrm{KUNodes}(\mathrm{BT}_i,\ rl_i,\ t)$. If $I \cap J = \emptyset$, it returns $\perp$. Otherwise, for any $x^{(i)} \in I \cap J$, it randomly chooses $s'_{i,x} \in Z_p$, and computes

$$dk_1^{(i)} = P_x^{(i)} \cdot Q_{x,1}^{(i)} \cdot F(t)^{s'_{i,x}} = {pk_{id}}^{\alpha_i} \cdot F(t)^{s_{i,x}+s'_{i,x}},$$
$$dk_2^{(i)} = Q_{x,2}^{(i)} \cdot g^{s'_{i,x}} = g^{s_{i,x}+s'_{i,x}}.$$

It outputs the decryption key $dk_{id,t}^{(i)} = (dk_1^{(i)},\ dk_2^{(i)})$.

– Encrypt. This algorithm takes the public parameter par, an LSSS access structure $(\mathbb{M},\ \rho)$, a set of public keys $\{\mathrm{PK}_{A_i}\}$ for relevant attributes, a time period t and a message M as the input. Let $\mathbb{M}$ be an $l \times n$ matrix. It randomly chooses a vector $\overrightarrow{v} = (\mu,\ y_2,\ ...,\ y_n)^{\perp} \in Z_p^n$. These values will be used to share the encryption exponent μ. For $i = 1$ to l, it calculates $v_i = \mathbb{M}_i \cdot \overrightarrow{v}$ where $\mathbb{M}_i$ is the i-th row of $\mathbb{M}$. Also, it randomly chooses $\mu,\ \mu_1,\ ...,\ \mu_k \in Z_p$, and computes

$$C_0 = \hat{e}(g,g)^{\alpha\mu} \cdot M, \quad C_2^{(i)} = (g^a)^{v_i} \cdot (\mathrm{PK}_{\rho(i)})^{-\mu_i},$$
$$C_1 = g^{\mu}, \quad C_3^{(i)} = g^{\mu_i}, \quad C_4^{(i)} = F(t)^{\mu_i}.$$

It outputs the ciphertext $\mathrm{CT} = ((\mathbb{M},\ \rho),\ t,\ C_0,\ C_1,\ \{C_2^{(i)},\ C_3^{(i)},\ C_4^{(i)}\}_{i \in [i,l]})$.

– Decrypt. This algorithm takes the public parameter par, a public and private user pair $(pk_{id},\ sk_{id})$, a set of decryption keys $\{dk_{id,t}^{(i)}\}$ and a ciphertext CT as the input. Suppose that $\{A_i\}$ associated with $\{pk_{id}^{A_i}\}$ satisfies the access structure $(\mathbb{M},\ \rho)$. Let I be defined as $I = \{i : \rho(i) \in \{A_i\}\}$. Denote by $\{w_i \in Z_p\}_{i \in I}$ a set of constants such that if $\{v_i\}$ are valid shares of any secret μ according to $\mathbb{M}$, then $\sum_{i \in I} w_i v_i = \mu$. It computes

$$\frac{\hat{e}(C_1, sk_{id}) \prod_{i \in I} \hat{e}(C_4^{(i)}, dk_2^{(i)})}{(\prod_{i \in I} \hat{e}(C_2^{(i)}, pk_{id}) \hat{e}(C_3^{(i)}, dk_1^{(i)}))^{w_i}} = \hat{e}(g,g)^{\alpha\mu},$$

and then cancels out this value from C_0 to obtain the plaintext M.

– Revoke. This algorithm takes an attribute A_i of identity id, a time period t, a revocation list rl_i and a state st_i as the input. For all the nodes $x^{(i)}$ associated with identity id, it adds $(x^{(i)},\ t)$ to rl_i, and outputs the updated rl_i.

Theorem 1. *Under the decisional q-parallel BDHE assumption, our basic ABE-GR scheme is selectively IND-CPA secure.*

Proof. In the proof, it is assumed that if an adversary has issued a private user-key query on an identity id, and a private attribute-key query on attributes $\{A_i\}$ of this identity id satisfying the challenge access structure $(\mathbb{M}^*,\ \rho^*)$, then at least one attribute in each set of minimum attributes satisfying $(\mathbb{M}^*,\ \rho^*)$ of this identity id is revoked at or before the challenge time period t^*. We detail the proof in the full version of this paper due to the space limit[8].

[8] Please contact the author for the full version.

4.2 Construction with Improved Efficiency

The main drawback in our previous construction lies in that all non-revokes data users need to periodically update their decryption keys. To remove such cumbersome workloads from data users, we give another ABE-GR scheme, which we call a server-aided ABE-GR scheme. Our method is to introduce an untrusted server to the basic ABE-GR scheme such that the server will help data users with the workloads in key update stage. The algorithms of our server-aided ABE-GR scheme mostly follow those in the basic ABE-GR scheme except with two differences.

– Firstly, the user-key generation algorithm is replaced by two algorithms, where one is run by each data user himself/herself called UUserKG, and the other one is run by the AA called AUserKG. The UUserKG algorithm outputs a public and private user-user-key pair. On input a public user-user-key and the master private key of the AA, the AUserKG algorithm outputs a public and private authority-user-key pair and publicly transmits them to the server.
– Secondly, the decryption algorithm is divided into two parts, of which one is run by the server called SDecrypt using the public and private authority-user-keys and decryption key, and the other one is run the data user called UDecrypt with the private user-user-key. The SDecrypt algorithm takes a ciphertext as the input, and outputs a partially decrypted ciphertext. The UDecrypt algorithm takes a partially decrypted ciphertext as the input, and outputs the plaintext.

Assume that for each data user, the server keeps a list of tuples (identity, attributes, public and private authority-user-keys, a set of private attribute-keys), i.e., $(id, \{A_i\}, (pk_{id}, sk_{id}), \{pk_{id}^{A_i}\})$. We detail the concrete construction as follows.

– GSetup. The same as that in the basic ABE-GR construction.
– ASetup. The same as that in the basic ABE-GR construction.
– UUserKG. The data user id randomly chooses $\tau \in Z_p$, and outputs a public and private user-user-key pair $(pk'_{id}, sk'_{id}) = (g^{\tau}, \tau)$.
– AUserKG. The AA randomly chooses $\beta \in Z_p$, and outputs a private and public authority-user-key pair $(sk_{id}, pk_{id}) = ((pk'_{id})^{\alpha}(g^{a})^{\beta}, g^{\beta})$. The AA will publicly send (sk_{id}, pk_{id}) to the server.
– PrivKG. The same as that in the basic ABE-GR construction. The AA will publicly send $pk_{id}^{A_i}$ to the server.
– TKeyUp. The same as that in the basic ABE-GR construction. The AA will publicly send $ku_t^{(i)}$ to the server.
– DecKG. The same as that in the basic ABE-GR construction except that it is run by the server rather than the data user.
– Encrypt. The same as that in the basic ABE-GR construction.
– SDecrypt. Given the private authority-user-key and decryption key, the server computes

$$C_0' = \frac{\hat{e}(C_1, sk_{id}) \prod_{i\in I} \hat{e}(C_4^{(i)}, dk_2^{(i)})}{(\prod_{i\in I} \hat{e}(C_2^{(i)}, pk_{id}) \hat{e}(C_3^{(i)}, dk_1^{(i)}))^{w_i}} = \hat{e}(pk_{id}', g)^{\alpha\mu},$$

and sends $\mathrm{CT}' = (id, C_0, C_0')$ to the data user.

– UDecrypt. The data user computes $M = C_0/(C_0')^{\frac{1}{\tau}}$ using the private user-user-key.
– Revoke. The same as that in the basic ABE-GR construction.

Remarks. It is worth noticing that the server-aided ABE-GR scheme has an edge over the basic ABE-GR one in both storage and computation overheads. Firstly, each data user in the server-aided ABE-GR construction only needs to keep one short private key, while in the basic ABE-GR one each data user keeps a private key of large size (depending on the size of attribute sets he/she owns and the total number of data users allowed in the system). Secondly, each data user in the server-aided ABE-GR system only needs to perform one exponentiation and no pairing computation to decrypt a ciphertext, while in the basic ABE-GR one each data user needs to perform many exponentiation and pairing computations. Thirdly, there is no secure channel required in the server-aided ABE-GR scheme for private key transmission, but the AA in the basic ABE-GR one needs to send the private user-key and attribute-keys to each data user via a secure channel.

Theorem 2. *Under the decisional q-parallel BDHE assumption, our server-aided ABE-GR scheme is selectively IND-CPA secure.*

Proof. In the proof, it is assumed that if an adversary has issued a private user-user-key query, a private authority-user-key query, and a private attribute-key on attributes $\{A_i\}$ satisfying the challenge access structure $(\mathbb{M}^*, \rho^*)$ on an identity id, then at least one attribute in each set of minimum attributes satisfying $(\mathbb{M}^*, \rho^*)$ of this identity id is revoked at or before the challenge time period t^*. The proof is similar to that in Theorem 1, and we detail it in the full version of this paper due to the space limit.

4.3 System Analysis

To the best of our knowledge, in addition to our work in this paper, [5], [1], [20] and [25] are also about constructions on revocable ABE built from the standard bilinear maps in the prime-order groups. This paper aims to achieve granular revocation in a CP-ABE system such that the AA can selectively revoke specific attributes of data users. In [5], a KP-ABE scheme with user revocation is proposed using indirect revocation where the AA enables the revocation by forcing revoked users to be unable to update their keys. A KP-ABE system with hybrid user revocation is raised in [1] which allows a data owner to select to use either direct or indirect revocation mode when encrypting a message. A generic way to build ABE schemes supporting dynamic credentials is provided in [20], where the AA indirectly accomplishes revocation by stopping updating the keys for revoked data users. In [25], a semi-trusted server is asked to share the

Table 1: Comparison of properties among revocable ABE (RABE) schemes.

	RABE in [5]	RABE in [1]	RABE in [25]	RABE in [20]	Basic ABE-GR	Server -aided ABE-GR
Revocation Mode	Indirect	Indirect & Direct	Direct	Indirect	Indirect	Indirect
Selective Revocation	No	No	No	No	Yes	Yes
Type of ABE	KP-ABE	KP-ABE	CP-ABE	KP-ABE & CP-ABE	CP-ABE	CP-ABE
Key Exposure Resistance	No	No	No	No	Yes	Yes
Secure Channel	Yes	Yes	Yes	Yes	Yes	No
Server	NA	NA	Semi-trust	NA	NA	Untrust
Size of Key Updates	$O(R\log(\frac{N}{R}))$	$O(R\log(\frac{N}{R}))$	NA	$O(R\log(\frac{N}{R}))$	$O(mR\cdot \log(\frac{N}{R}))$	$O(mR\cdot \log(\frac{N}{R}))$
Size of Key Stored by User	$O(l\log N)$	$O(l\log N)$	$O(1)$	$O(l\log N)$ & $O(k\log N)$	$O(k\log N)$	$O(1)$
Data User's Computation Overhead	$\geq 2(\mathrm{E}+\mathrm{P})$	$\geq 3\mathrm{E}+4\mathrm{P}$	E	$\geq \mathrm{E}+\mathrm{P}$	$\geq \mathrm{E}+4\mathrm{P}$	E

decryption capability with data users, and thus the server can indirectly revoke a data user by terminating the decryption for this data user.

Denote "NA" by the meaning of not-applicable. Let R be the number of revoked users, N be the number of all data users, l be the number of attributes presented in the access structure, k be the size of attribute set possed by each data user, and m be the maximum size allowed for k. In Table 1, we compare our revocable systems with the revocable ABE constructions in [5], [1], [25] and [20], where "E" and "P" denote the calculation of exponentiation and pairing, respectively. It is straightforward to see that our notion of ABE-GR is the first that achieves selective revocation while preserving desirable properties in terms of both security and efficiency. Additionally, our server-aided ABE-GR scheme greatly reduces the storage and computation overhead incurred to each data user with the help of an untrusted server.

5 Conclusions

In this paper, we introduced a notion called attribute-based encryption with granular revocation (ABE-GR) to achieve selective revocation, where each data user's attributes (or credentials) can be selectively revoked. To our knowledge, there are few works on such a revocation mechanism, and most of the existing revocable ABE schemes aim to revoke a data user from the system such that a revoked data user will become underprivileged to all (newly) encrypted data in the system. Motivated by the key separation technique in distributed ABE [17] where one single AA's workload is split across several AAs and each AA is responsible for at least one specific attribute, we equipped a normal ABE system with a similar technique such that each data user's attribute-keys are composed of key elements (corresponding to different attributes) generated separately but essentially linkable to each other. Thus, each data user's attributes can be selectively revoked by the AA, and a data user can be revoked from the system by separately revoking all of his/her attributes. After the description of security model for SR-ABE, we presented a basic construction of ABE-GR, which utilizes the binary tree data structure to reduce the workload of the AA. Then, we further improved the efficiency by introducing an untrusted server to the proposed ABE-GR scheme to help data users with the workloads incurred in key update and decryption, which we call server-aided ABE-GR. In addition, we formally proved the security of our ABE-GR and server-aided ABE-GR schemes, and compared them with other concrete constructions of revocable ABE that are related to our work.

Acknowledgements

This research work is supported by the Singapore National Research Foundation under the NCR Award Number NRF2014NCR-NCR001-012.

References

1. N. Attrapadung and H. Imai. Attribute-based encryption supporting direct/indirect revocation modes. In *Cryptography and Coding, 12th IMA International Conference, Cryptography and Coding 2009, Cirencester, UK, December 15-17, 2009. Proceedings*, volume 5921 of *Lecture Notes in Computer Science*, pages 278–300. Springer, 2009.
2. N. Attrapadung and H. Imai. Conjunctive broadcast and attribute-based encryption. In *Pairing-Based Cryptography - Pairing 2009, Third International Conference, Palo Alto, CA, USA, August 12-14, 2009, Proceedings*, volume 5671 of *Lecture Notes in Computer Science*, pages 248–265. Springer, 2009.
3. J. Baek and Y. Zheng. Identity-based threshold decryption. In *Public Key Cryptography - PKC 2004, 7th International Workshop on Theory and Practice in Public Key Cryptography, Singapore, March 1-4, 2004*, volume 2947 of *Lecture Notes in Computer Science*, pages 262–276. Springer, 2004.
4. A. Beimel. *Secure Schemes for Secret Sharing and Key Distribution.* PhD thesis, Israel Institute of Technology, Israel Institute of Technology, June 1996.
5. A. Boldyreva, V. Goyal, and V. Kumar. Identity-based encryption with efficient revocation. In *Proceedings of the 2008 ACM Conference on Computer and Communications Security, CCS 2008, Alexandria, Virginia, USA, October 27-31, 2008*, pages 417–426. ACM, 2008.

6. D. Boneh, X. Ding, G. Tsudik, and C. Wong. A method for fast revocation of public key certificates and security capabilities. In *10th USENIX Security Symposium, August 13-17, 2001, Washington, D.C., USA*. USENIX, 2001.
7. D. Boneh and M. Franklin. Identity-based encryption from the weil pairing. In *CRYPTO*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–219. Springer-Verlag, 2001.
8. H. Cui and R. H. Deng. Revocable and decentralized attribute-based encryption. *The Computer Journal, doi: 10.1093/comjnl/bxw007.*
9. X. Ding and G. Tsudik. Simple identity-based cryptography with mediated RSA. In *Topics in Cryptology - CT-RSA 2003, The Cryptographers' Track at the RSA Conference 2003, San Francisco, CA, USA, April 13-17, 2003, Proceedings*, volume 2612 of *Lecture Notes in Computer Science*, pages 193–210. Springer, 2003.
10. Y. Hanaoka, G. Hanaoka, J. Shikata, and H. Imai. Identity-based hierarchical strongly key-insulated encryption and its application. In *Advances in Cryptology - ASIACRYPT 2005, 11th International Conference on the Theory and Application of Cryptology and Information Security, Chennai, India, December 4-8, 2005, Proceedings*, volume 3788 of *Lecture Notes in Computer Science*, pages 495–514. Springer, 2005.
11. M. Horváth. Attribute-based encryption optimized for cloud computing. In *SOFSEM 2015: Theory and Practice of Computer Science - 41st International Conference on Current Trends in Theory and Practice of Computer Science, Pec pod Sněžkou, Czech Republic, January 24-29, 2015. Proceedings*, volume 8939 of *Lecture Notes in Computer Science*, pages 566–577. Springer, 2015.
12. A. B. Lewko and B. Waters. Decentralizing attribute-based encryption. In *Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings*, volume 6632 of *Lecture Notes in Computer Science*, pages 568–588. Springer, 2011.
13. J. Li, J. Li, X. Chen, C. Jia, and W. Lou. Identity-based encryption with outsourced revocation in cloud computing. *IEEE Trans. Computers*, 64(2):425–437, 2015.
14. Q. Li, H. Xiong, and F. Zhang. Broadcast revocation scheme in composite-order bilinear group and its application to attribute-based encryption. *IJSN*, 8(1):1–12, 2013.
15. K. Liang, J. K. Liu, D. S. Wong, and W. Susilo. An efficient cloud-based revocable identity-based proxy re-encryption scheme for public clouds data sharing. In *Computer Security - ESORICS 2014 - 19th European Symposium on Research in Computer Security, Wroclaw, Poland, September 7-11, 2014. Proceedings, Part I*, volume 8712 of *Lecture Notes in Computer Science*, pages 257–272. Springer, 2014.
16. B. Libert and J. Quisquater. Efficient revocation and threshold pairing based cryptosystems. In *Proceedings of the Twenty-Second ACM Symposium on Principles of Distributed Computing, PODC 2003, Boston, Massachusetts, USA, July 13-16, 2003*, pages 163–171. ACM, 2003.
17. S. Müller, S. Katzenbeisser, and C. Eckert. Distributed attribute-based encryption. In *Information Security and Cryptology - ICISC 2008, 11th International Conference, Seoul, Korea, December 3-5, 2008, Revised Selected Papers*, volume 5461 of *Lecture Notes in Computer Science*, pages 20–36. Springer, 2008.
18. D. Naor, M. Naor, and J. Lotspiech. Revocation and tracing schemes for stateless receivers. In *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*, volume 2139 of *Lecture Notes in Computer Science*, pages 41–62. Springer, 2001.

19. B. Qin, R. H. Deng, Y. Li, and S. Liu. Server-aided revocable identity-based encryption. In *Computer Security - ESORICS 2015 - 20th European Symposium on Research in Computer Security, Vienna, Austria, September 21-25, 2015, Proceedings, Part I*, volume 9326 of *Lecture Notes in Computer Science*, pages 286–304. Springer, 2015.
20. A. Sahai, H. Seyalioglu, and B. Waters. Dynamic credentials and ciphertext delegation for attribute-based encryption. In *Advances in Cryptology - CRYPTO 2012 - 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings*, volume 7417 of *Lecture Notes in Computer Science*, pages 199–217. Springer, 2012.
21. A. Sahai and B. Waters. 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*, volume 3494 of *Lecture Notes in Computer Science*, pages 457–473. Springer, 2005.
22. J. H. Seo and K. Emura. Revocable identity-based encryption revisited: Security model and construction. In *Public-Key Cryptography - PKC 2013 - 16th International Conference on Practice and Theory in Public-Key Cryptography, Nara, Japan, February 26 - March 1, 2013. Proceedings*, volume 7778 of *Lecture Notes in Computer Science*, pages 216–234. Springer, 2013.
23. Z. Wan, J. Liu, and R. H. Deng. HASBE: A hierarchical attribute-based solution for flexible and scalable access control in cloud computing. *IEEE Trans. Information Forensics and Security*, 7(2):743–754, 2012.
24. B. Waters. 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*, volume 6571 of *Lecture Notes in Computer Science*, pages 53–70. Springer, 2011.
25. Y. Yang, X. Ding, H. Lu, Z. Wan, and J. Zhou. Achieving revocable fine-grained cryptographic access control over cloud data. In *Information Security, 16th International Conference, ISC 2013, Dallas, Texas, USA, November 13-15, 2013, Proceedings*, volume 7807 of *Lecture Notes in Computer Science*, pages 293–308. Springer, 2013.