Journal of Universal Computer Science, vol. 19, no. 16 (2013), 2349-2367
submitted: 15/7/13, accepted: 27/9/13, appeared: 1/10/13 © *J.UCS*

An Efficient Ciphertext-Policy Attribute-Based Access Control towards Revocation in Cloud Computing*

Xingxing Xie
(Department of Mathematics
Xidian University, Xi'an 710071, P.R.China
xiexingxing11@163.com)

Hua Ma
(Department of Mathematics
Xidian University, Xi'an 710071, P.R.China
ma_hua@126.com)

Jin Li
(Department of Computer Science
Guangzhou University, Guangzhou 510006, P.R.China
jinli71@gmail.com)

Xiaofeng Chen*
(State Key Laboratory of Integrated Service Networks (ISN)
Xidian University, Xi'an 710071, P.R.China
xfchen@xidian.edu.cn)

Abstract: Attribute-Based Encryption (ABE) is one of the new visions for fine-grained access control in cloud computing. Plenty of research work has been done in both academic and industrial communities. However, before ABE can be deployed in data outsourcing systems, efficient enforcement of authorization policies and policy updates are the main obstacles. Therefore, in order to solve this problem, efficient and secure attribute and user revocation should be proposed in original ABE scheme, which is still a challenge in existing work. In this paper, we propose a new ciphertext-policy ABE (CP-ABE) construction with efficient attribute and user revocation, which largely eliminates the overhead computation at data service manager and data owner. Besides, we present an efficient access control mechanism based on the CP-ABE construction with one outsourcing computation service provider.

Key Words: attribute-based encryption, revocation, outsourcing, re-encryption
Category: E.3

1 Introduction

Cloud computing is a new and promising technology, which is altering the traditional computing paradigm of Internet. Recently, cloud computing, which has

[1] The extended abstract of this paper has been presented in *International Conference, ICTEurAsia,* pp.373-382, Springer, LNCS 7804, 2013.
[2] Corresponding author: Xiaofeng Chen (xfchen@xidian.edu.cn)

attracted much attention from fields of both academia and industry, has appeared as one of the most favorite paradigms in the IT industry. Especially, data outsourcing is becoming more and more significant in the cloud computing. In order to administrate the outsourced data of users, a preferable access control system needs to be put forward. Particularly, in the outsourcing environment, designing an access control will introduce many challenges.

Access control is one of the most common and versatile mechanisms used for security enforcement of information systems. An access control model formally describes how to decide whether an access request should be permitted or repudiated. Attribute-based encryption(ABE) attracting much attention has emerged as a relatively new encryption technology. That is because ABE enables efficient one-to-many broadcast encryption and fine-grained access control. And ABE is a new fancy for public key encryption that allows users to enforce encryption and decryption of messages by means of attributes. Therefore, ABE is expected to be applied in many fields of technology, such as data outsourcing. In the attribute-based access control, as frequently joining or leaving from some attribute groups [Hur and Noh 2011], the user and attribute revocation arise public's attention.

Given its expressiveness, the user and attribute revocation are currently being considered as a challenge in existing ABE schemes. Especially, efficiency of the user and attribute revocation is main drawback, which impedes attribute-based access control from being adopted. Therefore, Many schemes [Hur and Noh 2011, Liang et al. 2011, Boldyreva et al. 2008, Ming et al. 2011] are proposed to cope with attribute-based access control towards efficient revocation. The most remarkable is the scheme proposed by J.Hur and D.K.Noh, which realizes attribute-based access control with efficient fine-grained revocation in outsourcing. However, key update will become a bottleneck for data service manager, which will perform heavy computation at every time of update. Moreover, in the outsourcing environment, external service provider [Wang et al. 2012] is indispensable. Thus, in this paper, we attempt to solve the problem of attribute-based access control using CP-ABE in outsourcing environments.

1.1 Related Work

It is key-policy ABE (KP-ABE) and ciphertext-policy ABE (CP-ABE) that are favorable. In ABE systems, ciphertexts and users' keys are tagged with access policies and sets of descriptive attributes respectively. In CP-ABE, an encipherer defines a policy applied to encrypt a ciphertext and the attributes are used to describe users' keys. Whereas, in KP-ABE, this situation is contrary to that in CP-ABE. To dominate users' privileges, we specify CP-ABE as the data outsourcing architecture.

Attribute Revocation Recently, revocable ABE draws public's attention and it occurs that several schemes have been proposed [Bethencourt et al. 2007, Boldyreva et al. 2008, Pirretti et al. 2006]. It is timed rekeying that is mentioned for realizing revocation, in which each of attribute will be set a expiration time. Because it is not possible to realize immediate rekeying on any member change, it is called as coarse-grained revocation. However, these approaches have two main problems, which consist of scalability problem and security degradation in terms of the backward and forward security [Hur and Noh 2011, Rafaeli and Hutchison 2003], which will be mentioned in the next section.

In regard to scalability, we describe some details about it. The key authority periodically announces a key update so that all users nonrevoked can update their keys, which could be a bottleneck for both parties involving in updating. In addition, in the previous revocation, it doesn't take the scalable distribution of the updated attribute keys to the group of users into account. Thus, it is still a tough problem to design a scalable and efficient revocation mechanism in the data outsourcing architecture by means of ABE.

The other problem is security degradation. In the ABE systems, a group of users share an attribute essentially. In practice, membership may change frequently in the group that shares an attribute. Then, before a new user comes to hold the attribute, he should access the previous data until the next expiration time. On the other hand, a revoked user would still be able to access the data encrypted until the data re-encrypted with the newly updated attribute keys.

In the previous schemes, the key authority periodically announces a key update, which will lead to a bottleneck for the key authority. With the help of semi-honest service provider, two CP-ABE schemes with immediate attribute revocation are proposed in [Ibraimi et al. 2007, Yu et al. 2010]. However, achieving fine-grained user access control failed. In the previous scheme [Wang et al. 2011], a fine-grained attribute revocation is proposed using two trees per user, which is not desirable in practice. Junbeom et al. [Hur and Noh 2011] proposed a CP-ABE scheme with fine-grained attribute revocation with the help of the honest-but-curious proxy. However, in the phase of key update, the overhead computation at data service manager is so large that will lead to a bottleneck for the data service manager.

User Revocation Recently, the importance of user revocation has arisen public's attention in many practical ABE systems. In the [Ostrovsky et al. 2007], a fine-grained user-level revocation is proposed using ABE that supports negative clause. To realizing that, one just adds the AND of negation of revoked user identities. Yet, this method still lacks efficiency performance. Golle et al. [Golle et al. 2008] also proposed a revocable scheme with KP-ABE, which has much limitation. The user revocation is proposed in [Rafaeli and Hutchison 2003, Naor et al. 2001] including ABE systems, because users need to change their attributes frequently in practice.

Attrapadung and Imai [Attrapadung and Imai 2009a] suggested two user-revocable ABE schemes, in which the data owner should take full control of all the membership lists that leads to be not applied in the outsourcing environments. That's because data owner will no longer be directly in control of data after outsourcing. Besides, Attrapadung and Imai [Attrapadung and Imai 2009b] also proposed an identity-based revocation scheme under multicast and constructed user-level revocation under direct revocation mode. In the previous schemes [Liang et al. 2011, Ostrovsky et al. 2007], a user loses all the access rights to the data when he is revoked from a single attribute group, which is limited to the availability. In other words, when a user is revoked from a single attribute group, it is unavoidable to be revoked from the whole system. And the construction lacks efficiency.

1.2 Our Contribution

In our study, we propose a ciphertext policy attribute-based access control with efficient revocation. This construction successfully eliminates the overhead computation at data service manager and data owner. Compared with [Hur and Noh 2011], in our proposed construction, the size of the ciphertext will approximately reduce by half. Correspondingly, in the phase of key update, the computation operated by the data service manager will reduce by half. In addition, we summarize the comparisons between our proposed scheme and [Hur and Noh 2011] in terms of size of ciphertext and key and the computations in the phase of key update. Furthermore, we formally show the security proof based on security requirement in the access control system.

Besides, the proposed construction is secure in the sense of "honest but curious" adversary model. If involving the condition of collusion attack, the proposed construction is not secure. Thus, we propose a new idea involving two data service managers, which lacks efficiency.

Main Difference from the Conference Version Parts of the work in this paper have been presented in [Xie et al. 2013]. We have carefully revised the conference version of this paper and presented the much mores details as compared to [Xie et al. 2013]. The main improvements are given as follows: Firstly, we add some details of a data outsourcing system. Secondly, we add a new [Section 8] and propose a new solution to the collusion attack. Finally, we provide a complete efficiency analysis in the [Section 6].

1.3 Organization

The rest of the paper is organized as follows. In [Section 2], we describe our attribute-based access control system and threat model. Then, we present the related background on access structure, bilinear groups and secret sharing, and state our complexity assumption in [Section 3]. Next, we give our proposed construction , key update of the construction and analysis of the efficiency in [Section 4], [Section 5] and [Section 6] respectively. In [Section 7] and [Section 8], we briefly prove the scheme in the sense of "curious but honesty" adversary model and simply extend our construction. Finally, we give conclusion in [Section 9].

2 Systems and Models

2.1 System Description and Assumptions

As shown in [Fig. 1], four entities are involved in our attribute-based access control system:

- Trusted authority. It generates public parameters and master keys for the system. It is responsible for generating, distributing and updating keys. It is fully trusted by all components in the outsourcing system.

- Data owner. It is a client who owns data, defines access policy and encrypts the outsourced data.

- User. It is an entity who would like to access the cryptographic data. If a user possesses a set of attributes, which satisfy the access policy defined by the data owner, then he will be able to decrypt the ciphertext.

- Service provider. It is an entity that provides data outsourcing service, which consists of data servers and a data service manager. The data servers are responsible for storing the outsourced data. The data service manager is assumed to be honest-but-curious. In other words, the data service manager will perform in accordance to our proposed protocols and returns correct computation results. However, it will try to find out as much sensitive information as possible. The attribute group keys are managed by the data service manager. Access control from outside users is executed by the data service manager.

2.2 Threat Model and Security Requirements

- *Data confidentiality*. It is not allowed to access the plaintext if a user's attributes do not satisfy the access policy. In addition, unauthorized data service manager should be prevented from accessing the plaintext of the encrypted data that it stores.

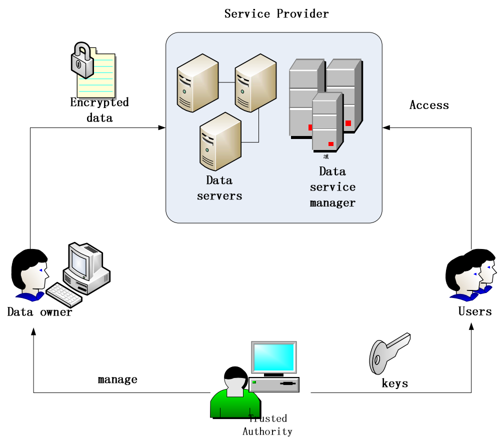

Figure 1: *Architecture of a data outsourcing systems*

- $Collusion - resistance$. Even if multiple users collaborate, they are unable to decrypt encrypted data by combining their attributes private keys. We note that the service provider is honest. In this paper, we do not consider active attacks by colluding with revoked users as in [Ibraimi et al. 2007, Yu et al. 2010].
- $Backward\ and\ forward\ secrecy$. In our context, backward secrecy means that if a new key is distributed for the group when a new member joins, he is not able to decrypt previous messages before the new member holds the attribute. On the other hand, forward secrecy means that a revoked or expelled group member will not be able to continue accessing the plaintext of the subsequent data exchanged (if it keeps receiving the messages), when the other valid attributes that he holds do not satisfy the access policy.

3 Preliminaries and Definitions

We first show the formal definition for access structure. Then, we give the preliminaries about secret sharing, bilinear map and security assumption.

3.1 Access Structure

Definition 1. (Access Structure) Let $\{P_1, P_2, \ldots, P_n\}$ be a set of parties. A collection $\mathring{A} \subseteq 2^{\{P_1, P_2, \ldots, P_n\}}$ is monotone if $\forall B, C : if\ B \in \mathring{A}$ and $B \subseteq C$ then $C \in \mathring{A}$. An access structure (respectively, monotone access structure) is a collection (respectively, monotone collection) $\mathring{A}$ of nonempty subsets of

$\{P_1, P_2, \ldots, P_n\}$, i.e., $\mathring{A} \subseteq 2^{\{P_1,P_2,\ldots,P_n\}} \backslash \{\emptyset\}$. The sets in $\mathring{A}$ are called the authorized sets, and the sets not in $\mathring{A}$ are called the unauthorized sets.

In our context, the role of the parties is taken by the attributes. Thus, the access structure $\mathring{A}$ will contain the authorized sets of attributes. We restrict our attention to monotone access structure. From now on, unless stated otherwise, by an access structure we mean a monotone access structure.

3.2 Bilinear Pairings

Let $\mathbb{G}$ and $\mathbb{G}_T$ be two cyclic group of prime order p. The Discrete Logarithm Problem on both $\mathbb{G}$ and $\mathbb{G}_T$ are hard. A bilinear map e is a map function e: $\mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ with the following properties:

1. Bilinearity: For all $A, B \in \mathbb{G}$, and $a, b \in \mathbb{Z}_p^*$, $e(A^a, B^b) = e(A, B)^{ab}$.
2. Non-degeneracy: $e(g, g) \neq 1$, where g is the generator of $\mathbb{G}$.
3. Computability: There exsits an efficient algorithm to compute the pairing.

3.3 Decisional Bilinear Diffie-Hellman Exponent (BDHE) Assumption [Waters 2011]

The decisional Bilinear Diffie-Hellman Exponent problem computes $e(g, g)^{a^{q+1}s}$ $\in \mathbb{G}_T$, given a generator g of $\mathbb{G}$ and elements $\overrightarrow{y} = (g_1, \cdots, g_q, g_{q+2}, \cdots, g_{2q}, g^s)$ for $a, s \in \mathbb{Z}_p^*$. Let g_i denote g^{a^i}.

An algorithm $\mathcal{A}$ has advantage $\epsilon(\kappa)$ in solving the decisional BDHE problem for a bilinear map group $\langle p, \mathbb{G}, \mathbb{G}_{\mathbb{T}}, e\rangle$, where κ is the security parameter, if $|Pr[\mathcal{B}(\overrightarrow{y}, T = e(g, g)^{a^{q+1}s}) = 0] - Pr[\mathcal{B}(\overrightarrow{y}, T = R) = 0]| \geq \epsilon(\kappa)$.

$\langle p, \mathbb{G}, \mathbb{G}_{\mathbb{T}}, e\rangle$ is deemed to satisfy the decisional BDHE assumption, when for every polynomial-time algorithm (in the security parameter κ) to solve the decisional BDHE problem on $\langle p, \mathbb{G}, \mathbb{G}_{\mathbb{T}}, e\rangle$, the advantage $\epsilon(\kappa)$ is a negligible function.

3.4 Secret Sharing

A (t, n)-secret sharing is used to separate a secret into n shares and any t shares can reconstruct the secret. However, combining less than t shares will not expose any information about the secret. As introduced by Shamir at el. in [Shamir 1979], in a $t-1$ degree polynomial, any t points on the polynomial can be used to reconstruct the secret. We will also make use of Lagrange coefficients $\triangle_{i,\wedge}$ for any $i \in \mathbb{Z}_p^*$ and a set, $\wedge$, of elements in $\mathbb{Z}_p^*$: $\triangle_{i,\wedge}(x) = \Pi_{j\in\wedge, j\neq i} \frac{x-j}{i-j}$.

3.5 System Definition and Our Basic Construction

Let $\mathcal{U} = \{u_1, \cdots, u_n\}$ be the whole of users. Let $\mathcal{L} = \{1, \cdots, p\}$ be the universe of attributes that defines, classifies the users in the system. Let $G_i \subset \mathcal{U}$ be a set of users that hold the attribute i. Let $\mathcal{G} = \{G_1, \cdots, G_p\}$ be the whole of such attribute groups. Let K_i be the attribute group key that is possessed by users, who own the attribute i.

Ciphertext Policy Attribute-Based Access Control with User Revocation

Definition 2. A CP-ABE with user revocation capability scheme consists of six algorithms:

- **Setup:** Taking a security parameter k, this algorithm outputs a public key PK and a master secret key MK.
- **KeyGen(MK, S, U):** Taking the MK, and a set of attributes $S \subseteq \mathcal{L}$ and users $U \subseteq \mathcal{U}$, this algorithm outputs a set of private attributes keys SK for each user.
- **KEKGen(U):** Taking a set of users $U \subseteq \mathcal{U}$, this algorithm outputs KEKs for each user, which will be used to encrypt attribute group keys.
- **Encrypt(PK, M, $\mathcal{T}$):** Taking the PK, a message M and an access structure $\mathcal{T}$, this algorithm outputs the ciphertext CT .
- **Re-Encrypt(CT, G):** Taking ciphertext CT and attributes groups G, this algorithm outputs re-encrypted CT'.
- **Decrypt(CT', SK, K_S):** The decryption algorithm takes as input the ciphertext $CT^{'}$, a private key SK, and a set of attribute group keys K_S. The decryption can be done.

4 Ciphertext Policy Attribute-Based Access Control with Efficient Revocation

4.1 Access Policy Tree

In the section, we briefly describe an access policy tree $\mathcal{T}$. The data access policy tree is composed by leaf nodes associated with attributes, each of which represents an attribute, and internal nodes, each of which is a logical gate, such as "AND", "OR", "n-of-m". Several functions and terms are defined as follows to facilitate the presentation of our scheme:

- $p(x)$: reprents the parent of the node x in the tree.

- x: denotes the attribute associated with the leaf node x.
- i_x: denotes the attribute associated with the leaf node x in the tree.
- If a user possesses a set of attributes that satisfy the internal nodes of the tree to reach the root, he can access the plaintext defined by the access tree $\mathcal{T}$. AND and OR are the most frequently used logical gates.
- num_x is the number of children of a node x and the threshold value is $k_x = num_x - 1$, which is used as the polynomial degree for node x using the threshold secret sharing scheme, when x is an AND. Also, when x is an OR gate or a leaf node, $k_x = 0$.
- $index(x)$: returns such a number associated with the node x, which is uniquely assigned to nodes in the access structure for a given key in an arbitrary manner.

Let $\mathcal{T}_x$ be the subtree of $\mathcal{T}$ at the node x. If a set of attributes i satisfies the access tree $\mathcal{T}_x$, we denote it as $\mathcal{T}_x = 1$. We compute $\mathcal{T}_x(i)$ recursively as follows: If x is a nonleaf node, $\mathcal{T}_x(i)$ return 1 iff at least k_x children return 1. If x is a leaf node, then $\mathcal{T}_x(i)$ return 1 iff $i_x \in i$.

4.2 KEK Construction

In our scheme, KEK tree will be used to re-encrypt the ciphertext encrypted by the owner, which is constructed by the data service manager as see [Fig. 2]. Now, some basic properties of the KEK tree will be presented as follows:

- Every member in $\mathcal{U}$ is distributed to the leaf nodes of the tree. Each of leaf node and internal node represents key, which is generated randomly. In the tree, each node v_j of the tree holds a KEK, denoted by KEK_j.
- Path keys PK_t originate from the leaf node up to the root node of the tree. Every member $u_t \in U$ obtains the relevant path keys. For example, u_3 stores $PK_3 = \{KEK_{10}, KEK_5, KEK_2, KEK_1\}$ as its path keys in [Fig. 2].
- For every G_i, the corresponding minimum cover sets, which cover all of the leaf nodes associated with users in G_i, in the KEK tree will be constructed. we regard a set of KEKs that represent root nodes of subtrees for G_i as $KEK(G_i)$. For instance, if $G_i = \{u_1, u_2, u_5, u_6, u_7, u_8\}$ in Fig.2, then $KEK(G_i) = \{KEK_3, KEK_4\}$. Because v_3 and v_4 are the root nodes of the minimum cover sets that could cover all of the members in G_i. It follows that this collection covers all users in G_i. However, any user $u \notin G_i$ can definitely not know any KEK in $KEK(G_i)$.

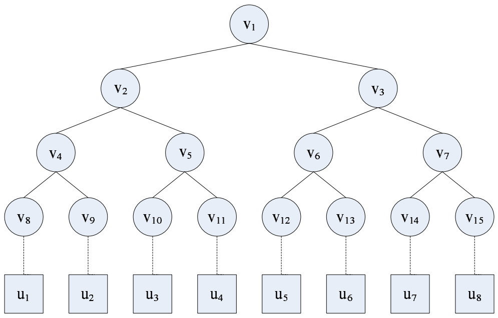

Figure 2: KEK tree attribute group key distribution

4.3 Our Construction

Let $e : \mathbb{G} \times \mathbb{G} \longrightarrow \mathbb{G}_T$ be a bilinear map of prime order p with the generator g. A security parameter, κ, will decide the size of the groups. We will additionally employ a hash function $H : \{0, 1\}^* \longrightarrow \mathbb{G}$ [Yang et al. 2011] that we will model as a random oracle. The function will map any attribute described as a binary string to a random group element.

4.3.1 System Setup and Key Generation

The trusted authority (TA) first runs Setup algorithm by choosing a bilinear map e: $\mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ of prime order δ with the generator g. Then, TA chooses two random $\alpha, a \in Z_p$. The public parameters are $PK = \{\mathbb{G}, g, h = g^a, e(g, g)^\alpha\}$. The master key is $MK = \{g^\alpha\}$, which is only konwn by the TA.

After executing the Setup algorithm producing PK and MK, each user in U needs to register with the TA, who verifies the user's attributes and issues proper private keys for the user. Running $KeyGen(MK, S, U)$, the TA inputs a set of users $U \subseteq \mathcal{U}$ and attributes $S \subseteq \mathcal{L}$, and outputs a set of private key components corresponding to each attribute $j \in S$. The key generation algorithm is presented as follows:

1. Choose a random $r \in \mathbb{Z}_p^*$, which is unique to each user.

2. Compute the following secret value to the user $u \in U$ as:

$$SK = (K = g^\alpha g^{ar}, L = g^r, \forall j \in S : D_j = H(j)^r)$$

After implementing above operations, TA sends the attribute groups G_j for each $j \in S$ to the data service manager. For instance, if u_1, u_2, u_3 are labeled with $\{1, 2\}, \{1, 2, 3\}, \{2, 3\}$, respectively, the data service manager will receives $G_1 = \{u_1, u_2\}, G_2 = \{u_1, u_2, u_3\}, G_3 = \{u_2, u_3\}$ from the TA.

4.3.2 KEK Generation

After obtaining the attribute groups G_j for each $j \in S$ from the TA, the data service manager runs KEKGen(U) and generates KEKs for users in U. Firstly, the data service manager sets a binary KEK tree for the unniverse of users $\mathcal{U}$ just like that described above. The KEK tree is responsible for distributing the attribute group keys to users in $U \subseteq \mathcal{U}$. For instance, u_3 stores $PK_3 = \{KEK_{10}, KEK_5, KEK_2, KEK_1\}$ as its path keys in [Fig. 2].

Then, in the data re-encryption phase, the data service manager will encrypt the attribute group keys by means of the path keys, i.e. KEKs. The method of the key assignment is that keys are assigned randomly and independently from each other, which is similar to information theoretic.

4.3.3 Data Encryption

To encrypt the data M, a data owner needs to specify a policy tree $\mathcal{T}$ over the universe of attributes $\mathcal{L}$. Running $Encrypt(PK, M, \mathcal{T})$, the data M is enforced attribute-based access control. The policy tree $\mathcal{T}$ is defined as follows.

For each node x in the tree $\mathcal{T}$, the algorithm chooses a polynomial q_x, which is chosen in a top-down manner, starting from the root node R and its degree d_x is one less than the threshold value k_x of the node, that is, $d_x = k_x - 1$. For the root node R, it randomly chooses an $s \in \mathbb{Z}_p^*$ and sets $q_R(0) = s$. Except the root node R, it sets $q_x(0) = q_{p(x)}(index(x))$ and chooses d_x other points randomly to completely define q_x for any other node x. Let Y be the set of leaf nodes in $\mathcal{T}$. The ciphertext is then constructed by giving the policy tree $\mathcal{T}$ as follows:

$$CT = (\mathcal{T}, \widetilde{C} = Me(g,g)^{\alpha s}, C = g^s,$$

$$\forall y \in Y : C_y = g^{a q_x(0)} \cdot H(y)^{-s})$$

After constructing CT, the data owner outsources it to the service provider securely.

4.3.4 Data Re-Encryption

On receiving the ciphertext CT, the data service manager re-encrypts CT using a set of the membership information for each attribute group $G \subseteq \mathcal{G}$ emerging from the access structure implanted in CT. The outsourced ciphertext was encrypted under the attribute-level access control by the data owner in the request of data outsourcing. However, the re-encryption algorithm enforces user-level access control. The re-encryption algorithm progresses as follows:

1. For all $G_y \in G$, chooses a random $K_y \in \mathbb{Z}_p^*$ and re-encrypts CT as follows:

$$CT' = (\mathcal{T}, \widetilde{C} = Me(g,g)^{\alpha s}, C = g^s,$$

$$\forall y \in Y : C_y = (g^{aq_x(0)} \cdot H(y)^{-s})^{K_y})$$

2. After re-encrypting CT, we firstly need select root nodes of the minimum cover sets in the KEK tree mentioned in [Section 4]. Then, the data service manager needs to employ a method to deliver the attribute group keys to valid users. The method we used is a symmetric encryption of a message M under a key K, in other words, $E_K : \{0,1\}^k \longrightarrow \{0,1\}^k$, as follow:

$$Hdr = (\forall y \in Y : \{E_K(K_y)\}_{K \in KEK(G_y)})$$

After above all operations, the data service manager responds with (Hdr, CT') to the user sending any data request. In practice, if users are not revoked from any of the attribute groups and authorized to decrypt it, they will be able to decrypt the attribute group key from Hdr, in despite of they cannot update their key constantly. We call the attribute group key distribution protocol through Hdr as a stateless appraoch.

4.3.5 Data Decryption

Data decryption phase consists of the attribute group key decryption from Hdr and message decryption.

Attribute group key decrypt. To execute data decryption, a user u_t first decrypts the attribute group key for all attributes in S that the user holds from Hdr. If the user $u_t \in G_j$, he can decrypt the attribute group key K_j from Hdr using a KEK that is possessed by the user. For example, if $G_j = \{u_1, u_2, u_5, u_6\}$ in [Fig. 2], u_5 can decrypt the K_j using the path key $KEK_6 \in PK_5$. Next, u_t updates its secret key as follows:

$$SK = (K = g^{\alpha} g^{ar}, \forall j \in S : D_j = H(j)^r, L = (g^r)^{1/K_j})$$

It is important to state that no $u \notin G_j$ is contained in any of the subsets whose root node is holding any KEK in $KEK(g_j)$, which means that for every KEK in $KEK(G_j)$, KEK is not divided from a random key given all the information of all users that not belongs to G_j. It is important to note that any user $u \notin G_j$ can not obtain K_j even if he colludes with other users $u' \notin G_j$.

Message decrypt. Once the user updates its secret key, he then runs the $Decrypt(CT', SK, K_S)$ algorithm as follows. The user runs a recursive function $DecryptNode(CT', SK, R)$, R is the root of $\mathcal{T}$. The recursion function is the same as defined in [Bethencourt et al. 2007]. And if x is a leaf node, then $DecryptNode(CT', SK, x)$ is proceeded as follow when $x \in S$ and $u_t \in G_x$:

$$\begin{aligned} Decrypt(CT', SK, x) &= e(C_x, L) \cdot e(C, D_x) \\ &= e((H(x)^{-s} \cdot g^{aq_x(0)})^{K_x}, (g^r)^{1/K_x}) \cdot e(g^s, H(x)^r) \\ &= e(g,g)^{raq_x(0)} \end{aligned}$$

Now we consider the recursion when x is a nonleaf node processed as follows: $\forall z$ is the child of x, it calls $DecryptNode(CT', SK, z)$ and stores the output as F_z. Let S_x be an arbitrary k_x-sized set of child nodes z, then computes:

$$\begin{aligned} F_x &= \prod_{z \in S_x} F_z^{\Delta_{i,S'_x}(0)}, where^{i=index(x),}_{S'_x=\{index(z):z\in S_x\}} \\ &= \prod_{z \in S_x} (e(g,g)^{r \cdot aq_z(0)})^{\Delta_{i,S'_x}(0)} \\ &= \prod_{z \in S_x} (e(g,g)^{r \cdot aq_{p(z)}(index(z))})^{\Delta_{i,S'_x}(0)} \\ &= \prod_{z \in S_x} (e(g,g)^{r \cdot aq_x(i)})^{\Delta_{i,S'_x}(0)} \\ &= e(g,g)^{r \cdot aq_x(0)} \end{aligned}$$

Where $i = index(z)$ and $S'_x = \{index(z) : z \in S_x\}$. Finally, if x is the root node R of the access tree $\mathcal{T}$, the recursive algorithm returns $A = DecryptNode$ $(CT', SK, R) = e(g,g)^{ras}$. And the algorithm decrypts the ciphertext by computing

$$\widetilde{C}/(e(C,K)/A) = \widetilde{C}/(e(g^s, g^\alpha g^{ra})/e(g,g)^{ras}) = M.$$

5 Key Update

In this section, when a user wants to leave or join several attribute groups, he needs to send the attributes changed to TA. On receiving the membership changes request for some attribute groups, TA informs the data service manager and sends the updated membership list of the attribute group to it. Afterwards, it changes the attribute group key for the attribute. Without loss of generality, suppose there is any membership change in G_j, which is equal to that a user comes to hold or drop an attribute j at the some instance. Next, we progress the update procedure as follows:

1. The data service manager selects a random $s' \in \mathbb{Z}_p^*$, and K'_i and re-encrypts the ciphertext CT' using PK as

$$\begin{aligned} CT^{'} &= (\mathcal{T}, \widetilde{C} = Me(g,g)^{\alpha(s+s')}, C = g^{(s+s')}, \\ C_i &= (g^{a(q_x(0)+s')} \cdot H(i)^{-s})^{K_i'} \\ &\quad \forall y \in Y \setminus \{i\} : C_y = (g^{a(q_x(0)+s')} \cdot H(y)^{-s})^{K_y}). \end{aligned}$$

It is important to note that the attribute group keys that are not affected by the membership changes do not need to be updated.

2. After updating the ciphertext, the data service manager selects new minimum cover sets for G_i changed and generates a new header message as follows:

$$\begin{aligned} Hdr &= (\{E_K(K_i')\}_{K \in KEK(G_i)}, \\ &\quad \forall y \in Y \setminus \{i\} : \{E_K(K_y)\}_{K \in KEK(G_y)}). \end{aligned}$$

On receiving a data request query from a user, the above Hdr and CT', which is updated afterwards, will respond to the user.

The key update procedure guarantees the user-level fine-grained access control, and also achieves the attribute revocation. In addition, the user revocation can be done in each attribute level. Thus, when the attributes possessed by a user satisfy the access policy, he may be able to decrypt the outsourced data, though he is revoked from some attribute groups.

6 Efficiency Analysis

Table 1: *Result Comparison of the Number of Exponentiations*

Scheme	Private Key Generation	Ciphertext Generation	key update
Scheme one	$2u+3$	$2\omega+2$	$2\omega+3$
Our scheme	$u+2$	$\omega+2$	$\omega+3$

In this section, we analyze the efficiency of the proposed scheme with the previous scheme [Hur and Noh 2011]. [Tab. 1] shows the comparisons between our proposed scheme and [Hur and Noh 2011] in terms of size of ciphertext and key and the computations in the phase of key update. We first summarize the computation cost for key generation. In the stage of key generation, supposing that the number of attributes possessed by a user is u, the number of exponentiations of our proposed scheme is $u + 2$. However, in the previous scheme, it is $2u + 3$. Then, in the phase of data encryption, in which we assume that the number of all leaf nodes is ω, the number of exponentiations of our proposed scheme is less ω than that of previous scheme.

Finally, we mainly analyze the computational workload in the phase of key update. In the proposed scheme, the number of exponentiations is reduced to ω+3. However, in the previous scheme, the number of exponentiations unexpectedly is 2ω+3. Thus, it will enormously improve computational efficiency. As shown in Table 1, our scheme performs better than scheme one at every stage, reducing nearly half of its computation cost. That is because every component corresponding to attributes or leaf nodes is one in our scheme. But it is two in scheme one.

7 Security

In this section, the security of the proposed scheme is given based on the security requirements discussed in [Section 2].

Theorem 3. ***Collusion Resistance.*** *The proposed scheme is secure to resist collusion attack.*

Proof. In this paper, it is important to resist to the colluding users for the sake of realizing fine-grained access control with revocation. In CP-ABE, the secret s sharing is embedded into a ciphertext, and to decrypt a ciphertext, a user or a colluding attacker needs to recover $e(g, g)^{\alpha s}$. To recover $e(g, g)^{\alpha s}$, the attacker must pair C_x from the ciphertext and D_x from the other colluding users' private key for an attribute x that the attacker does not hold. However, every user's private key is uniquely generated by a random r. Thus, even if the colluding users are all valid, the attacker can not recover $e(g, g)^{\alpha s}$. Only when the attributes possessed by the attacker satisfy the tree policy, the attacker could recover $e(g, g)^{\alpha s}$. So, the desired value $e(g, g)^{\alpha s}$ cannot be recovered by collusion attack.

Theorem 4. ***Data Confidentiality.*** *The proposed scheme prevents unauthorized users and the curious service provider from acquiring the privacy of the outsourced data.*

Proof. Firstly, if the attributes holden by a user don't satisfy the tree policy $\mathcal{T}$, the user will not recover the value $e(g, g)^{ras}$, which leads the ciphertext not to be deciphered. Secondly, when a user is revoked from some attribute groups that satisfy the access policy, he will lose the updated attribute group key. If the user would like to decrypt a node x for corresponding attribute, he needs to pair C_x from the ciphertext and L encrypted by K_x from its private key. As the user cannot obtain the updated attribute group key K_x, he cannot decrypt the value $e(g,g)^{raq_x(0)}$. Ultimately, Since we assume that the service provider is honest-but-curious, the service provider cannot be totally trusted by users. The service provider is available to the ciphertext and each attribute group key. However, any of the private keys for the set of attributes are not given to the data service manager. Thus, the service provider will not decrypt the ciphertext.

Theorem 5. ***Backward and Forward Secrecy.*** *For backward and forward secrecy of the outsourced data, the proposed scheme is secure against any newly joining and revoked users, respectively.*

Proof. When a user comes to join some attribute groups, the corresponding attribute group keys are updated and delivered to the user securely. In addition, the data service manager updates the ciphertext with random s' as described above. Even if the user has stored the previous ciphertext exchanged before he obtains the attribute keys and the holding attributes satisfy the access policy, he cannot decrypt the pervious ciphertext. That is because, in the light of the user's ability, though he could succeed in computing $e(g,g)^{ra(s+s')}$, it will not help to recover the value $e(g,g)^{\alpha s}$ from the updated ciphertext. We could ensure the backward secrecy of the outsourced data in the proposed scheme.

Furthermore, when a user comes to leave some attribute groups, the corresponding attribute group keys are updated and not delivered to the user. In addition, the data service manager updates the ciphertext with random s' as described above. As the user cannot obtain the updated attribute group keys, he cannot decrypt any nodes corresponding to the updated attributes. Moreover, even if the user has stored $e(g,g)^{\alpha s}$, he cannot decrypt the subsequent value $e(g,g)^{\alpha(s+s')}$. Because he is not available to random s'. Therefore, we could ensure the forward secrecy of the outsourced data in the proposed scheme.

8 Extension

We have proven that the proposed scheme is secure in the sense of "honest but curious" adversary model, in which we assume that data service manager will follow our proposed rules but try to find out as much secret information as possible. However, if a curious user and an illegal data service manager collaborate, the user will obtain KEKs for attribute groups, in which he is illegitimate.

In [Yang et al. 2011], another variant of such collusion attack appears as well. Colluding with the encryption service provider, a curious user is able to successfully decrypt the ciphertext even if he is not specified as a valid decryptor.

In order to prevent such collusion attack, we specify a second data service manager and suppose that user and the two data service managers will not collude.

To realize such advanced construction, the both KEK trees respectively possessed by data service managers cannot be leaked at the same time. To achieve this goal and build new construction, in the phase of data encryption, we randomly split s into s_1 and s_2 [Li et al. 2012], which will be respectively used as root node. Thus, the two data service managers re-encrypt the ciphertext obtained from the owner using a set of membership information for each attribute group $G \subseteq \mathcal{G}$.

We note that such method of coping with collusion attack will lack of efficiency. Thus, the construction needs to be improved. We hope to seek a new and efficient scheme in the later.

9 Conclusions

In data outsourcing environment, users' concerns about the enforcement of authorization policies and the support of policy updates are one of the main obstacles. In this paper, aiming at improving the efficiency of revocation to make CP-ABE widely deployed in access control, we introduce a new CP-ABE construction, which realizes a fine-grained access control based on CP-ABE on the outsourced data. In this construction, the size of the ciphertext and key and the overall computation of key update become less. Furthermore, the security proof is also shown based on access control security requirements. Moreover, the proposed scheme guarantees that a data owner takes control of the access policy and employs it to his outsourced data. We demonstrated that the proposed scheme is efficient to manage the outsourced data.

Acknowledgements

We are grateful to the anonymous referees for their invaluable suggestions. This work is supported by the National Natural Science Foundation of China (Nos. 61272455 and 61100224), the Fundamental Research Funds for the Central Universities (K50511010001 and JY10000901034), and China 111 Project (No. B08038).

References

[Attrapadung and Imai 2009a] Attrapadung, N., Imai, H.: "Conjunctive Broadcast and Attribute-Based Encryption"; Proc.Int'l Conf.Palo Alto on Pairing-Based Cryptography, Springer-Verlag, Berlin (2009), 248-265.

[Attrapadung and Imai 2009b] Attrapadung, N., Imai, H.: "Attribute-based encryption supporting direct/indirect revocation modes"; Proceedingd of Cryptography and Coding, Springer-Verlag, Berlin (2009), 278-300.

[Bethencourt et al. 2007] Bethencourt, J., Sahai, A., Waters, B.: "Ciphertext-Policy Attribute-Based Encryption"; Proc. IEEE Symp. Security and Privacy, New York (2007), 321-334.

[Boldyreva et al. 2008] Boldyreva, A., Goyal, V., Kumar, V.: "Identity-Based Encryption with Efficient Revocation"; Proc. ACM Conf. Computer and Comm. Security, New York (2008), 417-426.

[Golle et al. 2008] Golle, P., Staddon, J., Gagne, M., Rasmussen, P.: "A Content-Driven Access Control Systems"; Proc. Symp. Identity and Trust on the Internet, New York (2008), 26-35.

[Hur and Noh 2011] Hur, J., Noh, D.K.: "Attribute-based Access Control with Efficient Revocation in Data Outsourcing System"; IEEE Transactions on Parallel and Distributed System 22, 7 (2011), 1214-1221.

[Ibraimi et al. 2007] Ibraimi, L., Petkovic, M., Nikova, S., Hartel, P., Jonker, W.: "Mediated Ciphertext-Policy Attribute-Based Encryption and Its Application"; Proc. Int'l Workshop Information Security Architecture, Springer-Verlag, Berlin (2007), 309-323.

[Li et al. 2012] Li, J., Li, J., Chen, X., Jia, C., Wong, D.S.: "Secure Outsourced Attribute-based Encryption"; IACR Cryptology ePrint Archive 2012: 635, (2012).

[Liang et al. 2011] Liang, X., Lu, R., Lin, X., Shen, X.: "Ciphertext Policy Attribute Based Encryption with Efficient Revocation"; technical report, Univ. of Waterloo (2011), http://bbcr.uwaterloo.ca/~x27liang/papers/abe/\%20with\%20revocation.pdf.

[Ming et al. 2011] Ming, Y., Fan, L., Li, H., Li, H.: "An Efficient Attribute based Encryption Scheme with Revocation for Outsourced Data Sharing Control"; Int Conf.Ins.Mea.Computer,Com. and Con., IEEE Computer Society, CA, USA (2011), 516-520.

[Naor et al. 2001] Naor, D., Naor, M., Lotspiech, J.: "Revocation and Tracing Schemes for Stateless Receivers"; Proc. Int'l Cryptology Conf. Advances in Cryptology, Springer-Verlag, London (2011), 41-62.

[Ostrovsky et al. 2007] Ostrovsky, R., Sahai, A., Waters, B.: "Attribute-Based Encryption with Non-Monotonoc Access Structures" Proc. ACM Conf. Computer and Comm. Security, New York, USA (2007), 195-203.

[Pirretti et al. 2006] Pirretti, M., Traynor, P., McDaniel, P., Waters, B.: "Secure Attribute-Based Systems"; Proc. ACM Conf. Computer and Comm. Security, IOS Press, Netherlands (2006), 799-837.

[Rafaeli and Hutchison 2003] Rafaeli, S., Hutchison, D.: "A Survey of Key Management for Secure Group Communication"; ACM Computing Surveys 35, 3 (2003), 309-329.

[Shamir 1979] Shamir, A.: "How to Share a Secret"; Communications of the ACM, 22, 11 (1979), 612-613.

[Wang et al. 2011] Wang, P., Feng, D., Zhang, L.: "Towards Attribute Revocation in Key-Policy Attribute Based Encryption"; Cryptology and Network Security Lecture Notes in Computer Science, Springer-Verlag, Berlin (2011), 272-291.

[Wang et al. 2012] Wang, J., Ma, H., Tang, Q., Li, J., Zhu, H., Ma, S., Chen, X.: "A New Efficient Verifiable Fuzzy Keyword Search Scheme"; Journal of wireless mobile Networks, Ubiquitous Computing and Dependable Applications, 3, 4 (2012), 61-71.

[Waters 2011] Waters, B.: "Ciphertext-Policy Attribute-Based Encryption: An Expressive, Effcient, and Provably Secure Realization"; Communications of the ACM, Springer, Berlin (2011), 53-70.

[Xie et al. 2013] Xie, X., Ma, H., Li, J., Chen, X.: "New Ciphertext-Policy Attribute-Based Access Control with Efficient Revocation"; International Conference on Information and Communication Technology, ICT-EurAsia 2013, Springer-Verlag, Heidelberg 2013, 373-382.

[Yang et al. 2011] Yang, F-Yi., Liu, Z-Wei., Chiu, S-Hui.: “Mobile Banking Payment System”; Journal of wireless mobile Networks, Ubiquitous Computing and Dependable Applications, 2, 3 (2011), 85-95.

[Yu et al. 2010] Yu, S., Wang, C., Ren, K., Lou, W.: “Attribute Based Data Sharing with Attribute Revocation”; Proc. ACM Symp. Information, Computer and Comm. Security, New York (2010), 261-270.

[Zhou and Huang 2011] Zhou, Z., Huang, D.: “Efficient and Secure Data Storage Operations for Mobile Cloud Computing”; Proc. Network and Service Management, IEEE, NJ, USA (2011), 37-45.