Achieving Collaborative Cloud Data Storage by Key-Escrow-Free Multi-Authority CP-ABE Scheme with Dual-Revocation

Nyamsuren Vaanchig, Hu Xiong, Wei Chen, and Zhiguang Qin

(Corresponding author: Nyamsuren Vaanchig)

School of Information and Software Engineering & University of Electronic Science and Technology of China
No. 4, North Jianshe Road, Chenghua District, Chengdu, Sichuan 610054, China
(Email: nyamsuren.v@gmail.com)

(Received Oct. 4, 2016; revised and accepted Feb. 20, 2017)

Abstract

Nowadays, more and more users store their data in cloud storage servers for great convenience and real benefits offered by the service, so cloud data storage becomes one of the desirable services provided by cloud service providers. Multi-Authority Ciphertext-Policy Attribute-Based Encryption (MA-CP-ABE) is an emerging cryptographic solution to data access control for large-scale collaborative cloud storage service, which allows any data owner to outsource the data to cloud data storage in order to enable users from collaborating domains or organizations to access the outsourced data. However, the existing MA-CP-ABE schemes cannot be directly applied to collaborative cloud storage services as data access control due to the key escrow problem and the absence of dual revocation mechanism (user revocation and attribute revocation). By addressing these issues, this paper presents a Key-Escrow-Free Multi-Authority Ciphertext-Policy Attribute-Based Encryption Scheme with Dual-Revocation by introducing "the essential attribute" and making use of a certificate authority apart from attribute authorities. Compared with the existing MA-CP-ABE schemes, the proposed scheme is the most suitable one to enable data access control for collaborative cloud storage systems. Furthermore, the security and performance analysis indicates that our scheme is more secure and reasonably efficient to be applied to practical scenarios as collaborative cloud storage systems.

Keywords: Access Control; Attribute Revocation; Collaborative Data Storage; Key Escrow; Multi-authority Ciphertext-policy Attribute-based Encryption; User Revocation

1 Introduction

Cloud storage is one of the popular services offered by cloud service providers, which allows data owners to store their data in third party storage servers for great convenience and real benefits offered by the service. However, this service introduces a great challenge to data access control which addresses how data owners ensure that their data stored in the third party storage servers are accessed by only authorized users [1, 9, 23, 30]. Since data storage servers are not in the same domain with data owners, they can not be fully trusted by data owners to be in charge of making data access decisions on behalf of the data owners.

Ciphertext-Policy Attribute-Based Encryption (CP-ABE) scheme is a great achievement to solve the challenge of data access control over data stored in cloud storage; it is a public key encryption technique designed for one-to-many communications, where data owners hold direct control on their access policies and the access policies are enforced cryptographically [2, 17, 25, 39]. In CP-ABE, each user is entitled to a set of attributes which are associated with the user's secret key. The data owner chooses an access policy over a set of attributes and encrypts the data under the access policy. A user is able to decrypt the ciphertext as long as the set of attributes associated with the user's secret key satisfies the access policy of the ciphertext. This desirable property of CP-ABE makes it suitable for data access control for cloud storage. Since its first introduction [4], there have been extensive research [7, 11, 15, 21, 33] regarding its various aspects. In all these schemes, one trusted central authority is required to handle all attributes in the system and issue a secret key for each user in the system. These schemes thus can be utilized as a data access control for cloud storage where data is encrypted under an access policy over attributes issued by a domain or an organization. However, in large-scale collaborative cloud storage systems, where data owners want to share their data according to the access policies described over attributes issued across different domains and organizations, the trusted central authority may become the bottleneck in the system per-

formance. Furthermore, it is not proper from the point of security due to the single point of trust.

Multi-Authority CP-ABE (MA-CP-ABE) scheme is introduced to overcome the issues mentioned above. In MA-CP-ABE schemes, users may hold attributes issued by multiple attribute authorities and data owners may also encrypt their data under access policies defined over attributes from different attribute authorities. Since, in MA-CP-ABE schemes, attributes are independently managed by different attribute authorities, both the workload and trust are distributed over multiple attribute authority instead of being centralized on a single authority. Since the introduction of MA-ABE [6] , researchers have done wide-ranging research on multi-authority CP-ABE schemes [8, 10, 18, 19, 20, 24, 26, 35, 36] by considering various challenges facing to it.

However, the existing MA-CP-ABE schemes cannot be directly applied to a collaborative cloud storage as data access control due to the key escrow problem and the absence of dual revocation mechanism (user revocation and attribute revocation). Although the existing MA-CP-ABE schemes overcome the issue of the single point of trust, they still hold the key escrow problem in the scope of any attribute authority. Consequently, any attribute authority could decrypt any data which encrypted under an access policy over a set of attributes from only the attribute authority, and the failure or corruption of any attribute authority also raises security issue for the data that encrypted under an access policy described by a set of attributes from one attribute authority. The consequences of these issues are not acceptable for collaborative cloud storage where data owners would like to make their data only accessible to designated users. Moreover, the existing MA-CP-ABE schemes lack dual revocation mechanism dealing with the invalidation of users' access privilege at either attribute level and system level. The mechanism invalidating a user's access privilege at the system level is called user revocation, which addresses the problem of revoking a user's full access privilege from the system when the user is detected as malicious, or the user leaves the system. Whereas the mechanism invalidating a user's access privilege at the attribute level is called attribute revocation, which concerns the issue of revoking a user's partial access privilege when the user's attribute set can be changed dynamically due to the user's role change in the system. Several works [16, 22, 29, 32, 34] have been proposed by solely concerning user revocation issue but in single-authority CP-ABE since the first discussion on user revocation in [4]. However, they are not directly applicable to collaborative cloud data storage systems due to the following drawbacks: vulnerable against collusion attack, scalability problem, and security degradation. Only considering attribute revocation issue, CP-ABE schemes with immediate attribute revocation mechanisms [13, 14, 31, 37, 38] and CP-ABE schemes with time-based attribute revocation mechanisms [27, 31] have been proposed in single-authority CP-ABE settings. Nevertheless, the time-based approaches cause the security degradation problem in terms of forward secrecy until the next expiration time, and the immediate revocation approaches except for the scheme in [37] make the system impurely CP-ABE-based such that it requires the server to be fully trusted. Moreover, the scheme in [37] cannot resist collusion attack from the server and non-revoked users since the updated ciphertexts are transformable to their old versions if the server misuses the update key, and also this scheme results in the stateless user problem. Later on, the similar attribute revocation solutions as those in [13, 31, 37] have been used in the MA-CP-ABE schemes [10, 20, 36], respectively, but these schemes inherit the drawbacks from the corresponding attribute revocation solutions. To the best of our knowledge, there is no MA-CP-ABE scheme which overcomes the key escrow problem and supports secure and scalable dual revocation mechanism (user revocation and attribute revocation).

In this paper, we propose a Key-Escrow-Free Multi-Authority CP-ABE Scheme with Dual-Revocation to enable a secure and scalable data access control for collaborative cloud storage systems. To accomplish the goal of this study, we encounter the following major challenges: (a) how to tie a user's secret keys together to prevent the collusion attack, (b) how to overcome the key escrow problem, (c) how to achieve dual revocation mechanism in which the trust of the cloud server must be reduced. In the proposed scheme, we thus overcome these challenges by introducing a dummy attribute called "the essential attribute" and user-central-key. In addition, our scheme makes use of a certificate authority apart from attribute authorities. The user-central-key is used to tie user's secret keys issued by different authorities; which is issued by the certificate authority. The essential attribute, which is handled by the certificate authority, is used in our construction for the dual-purpose of overcoming the key escrow problem and achieving user revocation mechanism.

For these purposes, in our scheme, each user holds an additional user-attribute-key associated with the essential attribute, and each data is encrypted under an access policy formed from a boolean formula that is expressed as a binary access tree which has two subtrees connected to an AND root gate. While one subtree consists of only leaf node associated with the essential attribute, the other subtree is formed from an access policy defined by the data owner over a set of meaningful attributes. That is to say, in our scheme, an access policy is defined over a set of attributes from at least two authorities: one must be the certificate authority and the other can be any attribute authority from which some attributes are involved in the access policy. Therefore, a data owner uses the public keys of those authorities together as a message encapsulation key, which results in that our scheme overcomes the key escrow problem. To achieve dual revocation mechanism, we assign an attribute secret for each attribute in the system including the essential attribute. Note that certificate authority is the revocation controller in user revocation mechanism, while attribute authorities are the charge of the revocation controller in attribute revoca-

tion mechanism. When either of user revocation or attribute revocation event takes place, the corresponding revocation controller redefines the attribute secret for the involved attribute. Accordingly, the attribute secret for the corresponding component in the public parameters, ciphertexts, and non-revoked users' secret keys must be updated in order to guarantee backward secrecy and forward secrecy.

To do so, we must consider the following practical problems that exist in the existing revocable CP-ABE schemes: the frequent update of public parameters, presence of a fully trusted cloud server, collusion attack that might be triggered by partially revoked or fully revoked user, and stateless user problem. The frequent update of the public parameters that exists in the revocation mechanisms of the previous works is not practical because it requires data owners to be aware of each revocation event and get the updated public parameters to avoid generating an invalid ciphertext no one could decrypt. To solve this impracticality, we introduce ill-formed ciphertext and ciphertext-heal-key components in our construction. The fully trusted cloud server that is considered in revocation mechanism of the previous works makes the system not purely CP-ABE based as well as vulnerable to the collusion attack from the cloud server and any revoked users. Although in our scheme, the cloud server is delegated to update ciphertexts upon any revocation event as similar as in previous works, we must prevent any unauthorized decryption by any fully revoked or partially revoked user with colluding with cloud server which is provided with a ciphertext-update key. Our solution for this challenge is that the ciphertext-update key generated by the revocation controller cannot be misused to transform the updated ciphertext to its old version. Moreover, the revocation controller generates a unique key-update-key for each non-revoked user, which cannot be used by any other user, such that our scheme can resist the collusion attack from any revoked user and non-revoked user. However, key-update-key generation for each user and its distribution to each non-revoked user bring more overhead on the revocation controller upon each revocation event.

In addition, it might result in stateless user problem such that authorized users can not decrypt the ciphertexts to which they authorized to access if the non-revoked users miss some key-update-keys due to some reasons. By simultaneously considering the overhead and the stateless user problem, our scheme uses a system-wide version number for each attribute in the system including the essential attribute to indicate the evolution of the attribute secret of the attribute. As a result, the revocation controller generates a unique key-update-key with the latest version number for the non-revoked user on demand whenever the user requests. The user request is triggered by the detection of the fact that any attribute component of his/her secret key is stateless by checking the corresponding version values in the both ciphertext and user-attribute-key upon a decryption process.

The main contributions of this work are summarized as follows. We provide the first construction of Key-Escrow-Free Multi-Authority CP-ABE Scheme with Dual-Revocation and summarize the comparisons between the proposed scheme and the existing MA-CP-ABE schemes regarding their functionalities, as in Table 1. Dual revocation mechanism of the proposed scheme guarantees both backward and forward secrecy, resists the potential collusion attacks and deals with practical problems that exist in the existing revocable CP-ABE schemes such as frequent update of the public parameters, the presence of a fully trusted cloud server, and the stateless user problem. Furthermore, we prove that the proposed scheme is selectively secure in the standard model under the q-parallel BDHE assumption as well as provide security and performance analysis compared with the existing works. The security and performance analysis show that our scheme is more secure and reasonably efficient to be applied to practical scenarios as collaborative cloud storage systems.

We organize the rest of the paper as follows. In Section 2, we provide some preliminary knowledge used in this work. The system model, algorithm definitions, security model and security properties are described in Section 3. The proposed Key-Escrow-Free Multi-Authority CP-ABE Scheme with Dual-Revocation is presented in Section 4 as a data access control for collaborative cloud storage. The security and performance analysis of the proposed scheme is provided in Section 5. Finally, a conclusion is given in Section 6.

2 Preliminaries

2.1 Access Structures

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

In our context, the role of the parties is taken by the attributes. Thus, the access structure $\mathbb{A}$ will contain the authorized set of attributes. We only consider monotone access structures.

2.2 Linear Secret Sharing Scheme

Definition 2. *(Linear Secret Sharing Scheme (LSSS) [3]). A secret-sharing scheme* $\prod$ *over a set of parties* $\mathcal{P}$ *is called linear over* $\mathbb{Z}_p$ *if*

1) *The shares of each party form a vector over* $\mathbb{Z}_p$*.*

2) *There exists a matrix an* M *with* l *rows and* n *columns called the secret-generating matrix for* $\prod$*. For all* $i = 1, ..., l$*, the* i*-th row of* M *is labeled by a party* $\rho(i)$*, where* ρ *is a function that maps a row to a*

Table 1: Comparison of multi-authority CP-ABE schemes

Properties	**[26]**	**[18]**	**[24]**	**[36]**	**[28]**	**[20]**	**[10]**	**Our**
Master Authority Free	No	Yes	No	Yes	Yes	Yes	Yes	Yes
Scalability	Yes	Yes	No	No	Yes	No	Yes	Yes
Key Escrow Free	No	No	Yes	No	No	No	No	Yes
Attribute Revocation	No	No	No	Yes	No	Yes	Yes	Yes
User Revocation	No	No	No	No	No	No	No	Yes
Bilinear Group Order	Prime	Composite	Composite	Prime	Prime	Composite	Composite	Prime
Security Model	GGM	ROM	ROM	ROM	ROM	SM	ROM	SM

* GGM: Generic Group Model, ROM: Random Oracle Model, SM: Standard Model

party for labeling. When we consider the column vector $\vec{v} = (s, r_2, ..., r_n)$, where $s \in \mathbb{Z}_p$ is the secret to be shared and $r_2, ..., r_n \in \mathbb{Z}_p$ are randomly chosen, then $M\vec{v}$ is the vector of l shares of the secret s according to $\prod$. The share $(M\vec{v})_i$ belongs to party $\rho(i)$.

According to [3], each linear secret sharing scheme meets *linear reconstruction* property defined as follows: Suppose that $\prod$ is a LSSS for the access structure $\mathbb{A}$. Let $\mathcal{S} \in \mathbb{A}$ be any authorized set, and let $I \subset \{1, 2, ..., l\}$ be the set of rows whose labels are in $\mathcal{S}$, i.e. $I = \{i : \rho(i) \in \mathcal{S}\}$. Then there exists constants $\{w_i \in \mathbb{Z}_p\}_{i \in I}$ such that if $\{\lambda_i\}$ are valid shares of any secret s according to $\prod$, then $\sum_{i \in I} w_i \lambda_i = s$. Additionally, it is shown in [3] that the constants $\{w_i\}_{i \in I}$ can be found in time polynomial in the size of the secret-generating matrix M.

2.3 Bilinear Map

Let $\mathbb{G}_0$ and $\mathbb{G}_1$ be two multiplicative cyclic groups of prime order p. Let g be a generator of $\mathbb{G}_0$ and e be a bilinear map, $e : \mathbb{G}_0 \times \mathbb{G}_0 \to \mathbb{G}_1$ with the following properties:

1) *Bilinearity*: for all $u, v \in \mathbb{G}_0$ and $a, b \in \mathbb{Z}_p$, we have $e(u^a, v^b) = e(u, v)^{ab}$.

2) *Non-degeneracy*: $e(g, g) \neq 1$.

3) *Computability*: There is an efficient algorithm to compute $e(u, v)$ for $\forall u, v \in \mathbb{G}_0$.

2.4 Decisional q-Parallel Bilinear Diffie-Hellman Exponent Assumption

Choose a group $\mathbb{G}_0$ of prime order p according to the security parameter. Let $a, s, b_1, ..., b_q \in \mathbb{Z}_p$ be chosen at random and g be a generator of $\mathbb{G}_0$. If an adversary is given $\vec{y} =$

$$g, g^s, g^a, ..., g^{(a^q)}, , g^{(a^{q+2})}, ..., g^{(a^{2q})}$$
$$\forall_{1 \leq j \leq q} g^{s \cdot b_j}, ..., g^{a/b_j}, ..., g^{(a^q/b_j)}, , g^{(a^{q+2}/b_j)}, ..., g^{(a^{2q}/b_j)}$$
$$\forall_{1 \leq j,k \leq q, k \neq j} g^{a \cdot s \cdot b_k / b_j}, ..., g^{(a^q \cdot s \cdot b_k / b_j)}$$

it must remain hard to distinguish $e(g, g)^{a^{q+1}s} \in \mathbb{G}_1$ from a random element in $\mathbb{G}_1$.

An algorithm $\mathcal{B}$ that outputs $z \in \{0, 1\}$ has advantage ϵ in solving decisional q-parallel BDHE in $\mathbb{G}_0$ if

$$\left| Pr[\mathcal{B}(\vec{y}, T = e(g, g)^{a^{q+1}s}) = 0] - Pr[\mathcal{B}(\vec{y}, T = R) = 0] \right| \geq \epsilon.$$

Definition 3. *The decisional q-parallel BDHE assumption holds if no polynomial time algorithm has a non-negligible advantage in solving the q-parallel BDHE problem.*

3 System and Security Models

3.1 System Model

We consider a collaborative cloud storage system as illustrated in Figure 1, and the system consists of the following entities:

Certificate Authority (CA): It sets up the system. In addition, CA is in charge of authorizing and revoking users' access privileges to and from the system. For this purpose, it handles the essential attribute such that it acts as an attribute authority handling only one attribute.

Attribute Authorities (AA): It sets up its own domain. Each AA is responsible for entitling and revoking attributes for and from users according to their roles in its domain. In our system, every attribute belongs to a single AA, but each AA manages an arbitrary number of attributes.

Cloud Server (CS): It stores data owners' data in encrypted form and provides data storage service to data users. The CS is not involved in access control enforcement or data decryption process. It means that, in our system, data access decisions are made cryptographically such that only authorized users can obtain data without depending any decision or process done by CS. We assume that CS is minimally trusted such that it converts ill-formed ciphertexts to well-formed ciphertexts upon data publication phase and updates well-formed ciphertexts to their latest version upon any revocation event.

Data Owners (DO): A data owner defines an access policy over a set of attributes from the relevant AAs

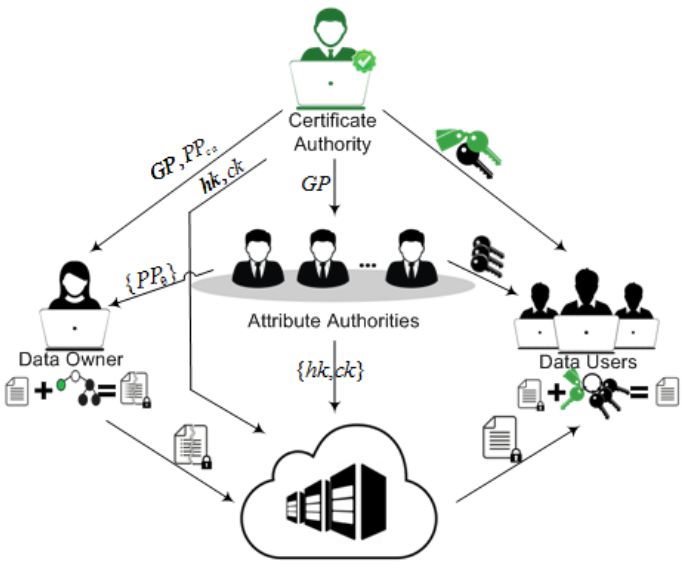

Figure 1: System architecture of collaborative cloud data storage

including CA and encrypts his/her data under the access policy. Then the DO publishes the encrypted data in ill-formed way to CS. The data access decision to this data is enforced cryptographically according to the access policy in the ciphertext without any dependence from CS. It means that any authorized user who has sufficient attributes can decrypt the ciphertext to obtain the data.

Data Users (DU): Each DU is assigned with a global identity from CA and entitled to a set of attributes from multiple AAs including the essential attribute from CA. Any DU can freely get any ciphertext from CS and can decrypt the ciphertext if and only if the DU's attribute set satisfies the access policy of the ciphertext. However, any DU's attribute set may dynamically change due to his/her role change in the different domains in the system, and this case will be handled by the attribute revocation mechanism of our system. Moreover, any DU can leave the system or detected as a malicious DU, and this case will be handled by the user revocation mechanism of our system.

3.2 Algorithms

The proposed scheme consists of the following algorithms:

- $GSetup(\lambda) \rightarrow GP, SK, VK$. This algorithm, run by CA, takes the security parameter λ as input and outputs the global parameters GP for the system. In addition, CA generates a pair of sign and verification keys, SK, VK.
- $ASetup(GP) \rightarrow MK_\theta, PP_\theta, \{hk_{x_\theta}^{h(ver_{x_\theta})}\}$. This algorithm, run by an AA, takes the global parameters GP as input, and it outputs the AA's master key MK_θ and public parameters PP_θ. It also outputs a set of ciphertext-heal keys $\{hk_{x_\theta}^{h(ver_{x_\theta})}\}$ for all the attributes $\{x_\theta\}$ that the AA handles, where $h(ver_{x_\theta})$ is the version value of the attribute x_θ.
- $GKeyGen(GP, gid) \rightarrow G_{gid}$. This algorithm, run by CA, takes the global parameters GP and a global identifier gid of a DU as input, and it outputs a user-central-key G_{gid} for the DU.
- $AKeyGen(GP, MK_\theta, G_{gid}, \mathcal{S}_{gid,\theta}) \rightarrow AK_{gid,\theta}$. This algorithm, run by an AA, takes the global parameters GP, the AA's master key MK_θ and a DU's user-central-key G_{gid} and an attribute set $\mathcal{S}_{gid,\theta}$ that describes the DU's role in the domain as input. It then outputs an user-attribute-key $AK_{gid,\theta}$ for the DU.
- $Encrypt(m, (M, \rho), GP, \{PP_\theta\}_{\theta \in F}) \rightarrow \tilde{ct}$. This algorithm, run by a DO, takes data m, an LSSS access structure (M, ρ) , the global parameters GP, and a set of public parameters $\{PP_\theta\}$ of the relevant AAs including CA as input. Here, θ is index of the relevant authorities including both CA and the AAs. It outputs an ill-formed ciphertext $\tilde{ct}$ that no one can decrypt.
- $CTConvert(\tilde{ct}, \{hk_{x_\theta}\}_{x_\theta=\rho(i),\theta \in F}^{h(ver_{x_\theta})}) \rightarrow CT$. This algorithm, run by CS, takes an ill-formed ciphertext $\tilde{ct}$ and a set of ciphertext-heal keys $\{hk_{x_\theta}\}_{x_\theta=\rho(i),\theta \in F}^{h(ver_{x_\theta})}$ for all the attributes included in the access structure of the ciphertext as input. It then outputs a well-formed ciphertext CT.
- $Decrypt(CT, G_{gid}, \{AK_{gid,\theta}\}_{\theta \in F}) \rightarrow m|\perp$. This algorithm, run by a DU, takes a well-formed ciphertext CT, a DU's user-central-key G_{gid} and a set of user-attribute-keys $\{AK_{gid,\theta}\}$ of the DU as input. It then outputs the data m when the DU's attribute set $\mathcal{S}_{gid}$ satisfies the access structure (M, ρ) of the CT. Otherwise the decryption fails and returns $\perp$.
- $CKeyGen(MK_\theta, y_\theta) \rightarrow hk_{y_\theta}^{h(ver'_{y_\theta})}, ck_{y_\theta}^{h(ver'_{y_\theta})}, MK'_\theta$. This algorithm, run by an AA, takes the AA's master key MK_θ, an attribute y_θ involved in the revocation event as input. It then outputs a ciphertext-heal-key $hk_{y_\theta}^{h(ver'_{y_\theta})}$, a ciphertext-update-key $ck_{y_\theta}^{h(ver'_{y_\theta})}$, and an updated master key MK'_θ.
- $CTUpdate(CT, ck_{y_\theta}^{h(ver'_{y_\theta})}) \rightarrow CT'$. This algorithm, run by CS, takes a ciphertext CT and a ciphertext-update key $ck_{y_\theta}^{h(ver'_{y_\theta})}$ as input, and it outputs an updated ciphertext CT'.
- $UKeyGen(MK_\theta, y_\theta, G_{gid}) \rightarrow uk_{gid,y_\theta}^{h(ver'_{y_\theta})}$. This algorithm, run by an AA, takes the AA's master key

MK_θ, an attribute y_θ, and an user-central-key G_{gid} as input. It then outputs a key-update-key $uk_{gid,y_\theta}^{h(ver'_{y_\theta})}$ associated with the attribute y_θ that a DU with the user-central-key G_{gid} is requested.

- $AKUpdate(AK_{gid,\theta}, uk_{gid,y_\theta}^{h(ver'_{y_\theta})}) \rightarrow AK'_{gid,\theta}$. This algorithm, run by a DU, takes the DU's user-attribute-key $AK_{gid,\theta}$ and a key-update-key $uk_{gid,y_\theta}^{h(ver'_{y_\theta})}$ for the DU as input, and it outputs an updated user-attribute-key $AK'_{gid,\theta}$ for the DU.

Definition 4. *The Key-Escrow-Free MA-CP-ABE Scheme with Dual-Revocation is correct if for any GP generated by GSetup(λ) algorithm, for any set of $\{PP_\theta, MK_\theta, hk_{x_\theta}\}$ generated by ASetup(GP) algorithm, for any CT encrypted by Encrypt$(m, (M, \rho), GP, \{PP_\theta\}_{\theta \in F})$ algorithm on data m and converted by CTConvert$(\tilde{ct}, \{hk_{x_\theta}\}_{x_\theta = \rho(i), \theta \in F}^{h(ver_{x_\theta})})$ algorithm, for any secret key $(G_{gid}, \{AK_{gid,\theta}\})$ generated by GKeyGen(GP, gid) and AKeyGen$(GP, MK_\theta, G_{gid}, \mathcal{S}_{gid,\theta})$ algorithms, it is true that Decrypt$(CT, G_{gid}, \{AK_{gid,\theta}\}_{\theta \in F}) = m$ if and only if the attribute set $\mathcal{S}_{gid} = \cup_{\theta \in F} \mathcal{S}_{gid,\theta}$ satisfies (M, ρ) and the version values of the corresponding components in the both of $\{AK_{gid,\theta}\}$ and CT are match.*

3.3 Security Model and Security Requirements

We consider potential attackers in the collaborative cloud storage system as follows: 1) The CA and each AA are assumed to be honest such that each of them does not collude with any other entity. However, CA or any AA can be corrupted by attackers, and also it should be prevented from decrypting any ciphertexts individually. 2) The CS is assumed to be minimally trusted. It might attempt to obtain the content of the encrypted data although it correctly performs the tasks assigned by legitimate entities. 3) Each DU is assumed to be dishonest and malicious, and he/she might attempt to obtain access to data beyond his/her access privilege. To simplicity, we classify dishonest and malicious DUs in the system into three categories: (a) unauthorized DU is a user who does not have sufficient attributes satisfying the access policy of the encrypted data. (b) partially revoked DU is a user whose attribute set no longer satisfies the access policy of the encrypted data. (c) fully revoked DU is a user whose access privilege is no longer valid in the system.

Now, we present a security model for the proposed Key-Escrow-Free Multi-Authority CP-ABE Scheme with Dual-Revocation. The security model is described by a game between a challenger $\mathcal{B}$ and an adversary $\mathcal{A}$. The phases of the game are following:

Global Setup. The $\mathcal{B}$ runs *GSetup* algorithm and sends the global parameters to the $\mathcal{A}$.

Init. In this phase, The $\mathcal{A}$ specifies a set of corrupted authorities $\mathcal{C} \subseteq \mathcal{N}$, where $\mathcal{N}$ is the set of all authorities. We assume that the $\mathcal{A}$ can corrupt at most $|\mathcal{N}| - 1$ number of authorities. In addition, the $\mathcal{A}$ declares a challenge access structure (M^*, ρ^*) along with the target version numbers $\{ver^*_{x_\theta}\}_{x_\theta = \rho^*(i)}$ for each attribute included in the challenge access structure, which he will try to attack.

Authority Setup. For non-corrupted authorities $\mathcal{N} - \mathcal{C}$, the $\mathcal{B}$ obtains the public parameters and master key by running *ASetup* algorithm and gives the public parameters to the $\mathcal{A}$.

Phase 1. The $\mathcal{A}$ makes secret key and key-update-key queries adaptively as follows:

- Secret key queries: A secret key consists of a user-central-key and a set of user-attribute-keys for an attribute set $\mathcal{S}_{gid}$. The $\mathcal{A}$ makes secret key queries by submitting $\mathcal{S}_{gid} = \bigcup_{\theta \in \mathcal{N} - \mathcal{C}} S_{gid,\theta}$, where $\mathcal{S}_{gid} \cup \bigcup_{\theta \in \mathcal{C}} U_\theta$ does not satisfy the challenge access structure. The $\mathcal{B}$ responds to each query by returning a user-central-key G_{gid} along with set of user-attribute-keys $\{AK_{gid,\theta}\}_{\theta \in \mathcal{N} - \mathcal{C}}$.
- Key-update-key queries: The $\mathcal{A}$ makes key-update-key queries by submitting (x_θ, G_{gid}) along with a version number ver_{x_θ} for the attribute x_θ, where x_θ must be in $S_{gid} = \bigcup_{\theta \in \mathcal{N} - \mathcal{C}} S_{gid,\theta}$ and $2 \leq ver_{x_\theta} \leq ver^*_{x_\theta}$. The $\mathcal{B}$ responds by returning the corresponding key-update-key $uk_{gid,x_\theta}^{h(ver_{x_\theta})}$ to the $\mathcal{A}$.

Challenge. The $\mathcal{A}$ submits two equal length messages m_0 and m_1 to the $\mathcal{B}$. The $\mathcal{B}$ flips a random coin $b \in \{0, 1\}$, encrypts m_b under the access structure (M^*, ρ^*) and sends the ciphertext CT^* to the $\mathcal{A}$, where the version numbers of each attribute in the access structure are equal to the corresponding version number $ver^*_{p(\cdot)}$ given with the challenge access structure.

Phase 2. Phase 1 is repeated.

Guess. The $\mathcal{A}$ makes a guess b' for b and it wins if $b = b'$.

The advantage of an adversary $\mathcal{A}$ in this game is defined as $Pr[b = b'] - 1/2$.

Definition 5. *Key-Escrow-Free Multi-Authority CP-ABE Scheme with Dual-Revocation is selectively secure (against static corruption of authorities) if all polynomial time adversaries have at most a negligible advantage in this security game.*

Our scheme also guarantees the following security requirements which are basic requirements for revocation:

Collusion Resistance: Any fully revoked/partially revoked DU should be prevented from decrypting any

ciphertext, to which he/she is not authorized to access, by colluding with CS (type-A) or any non-revoked DU (type-B).

Forward Secrecy: Any partially revoked DU should be prevented from decrypting any ciphertext which requires the attributes which are revoked from the DU to decrypt. Any fully revoked DU should be prevented from decrypting any ciphertext.

Backward Secrecy: Any new/non-revoked DU should be able to decrypt any ciphertext as long as the DU has sufficient attribute set satisfying the access structure of the ciphertext.

4 Data Access Control by Key-Escrow Free Multi-Authority CP-ABE with Dual Revocation

4.1 Overview

We now present our Key-Escrow-Free Multi-Authority CP-ABE Scheme with Dual-Revocation that can enable a secure and scalable data access control for collaborative cloud storage systems. As previously mentioned, our scheme introduces a dummy attribute called the essential attribute and makes use of a certificate authority (CA) apart from attribute authorities (AAs). Although our scheme separates authorities into a CA and multiple AAs, CA acts as an AA by handling the essential attribute besides setting up the system. The essential attribute is used in our scheme for dual-purpose: to achieve user revocation mechanism and key escrow free property. The essential attribute, therefore, must be included in the access structure of each ciphertext. Moreover, each user is required to possess an additional user-attribute-key which is associated with the essential attribute besides the DU's user-attribute-keys that are associated with a different set of attributes from different domains. Note that CA will be included in all the AA notation throughout this paper since it is involved in two different roles in the system.

In our construction, an access structure is formed from a boolean formula that expressed as a binary access tree which has two subtrees connected to an AND root gate. While one subtree consists of only leaf node associated with the essential attribute, the other subtree is formed from an access policy defined by the DO over a set of meaningful attributes, where interior nodes are AND or OR gates and the leaf nodes correspond to the attributes. Using the LSSS generation algorithm [18], one can convert any access trees into their equivalent LSSS matrices. Since the access structure is defined over a set of attributes from different AAs including CA, a DO encrypts his/her data under the access structure by using the public parameters of the relevant AAs including CA; it results in an ill-formed ciphertext. Before publishing the ill-formed ciphertext, CS converts it into a well-formed ciphertext by using a set of the corresponding ciphertext-heal-keys . To decrypt any ciphertext, a DU must have a user-central-key issued by CA and a set of user-attribute-keys satisfying the access structure of the ciphertext, in which the user-attribute-key associated with the essential attribute must be valid.

Our scheme supports both user revocation and attribute revocation, and we call it dual revocation. We achieve both user revocation and attribute revocation mechanism by applying the same technique, but CA is in charge of user revocation controller by handling the essential attribute and any AA is in charge of attribute revocation controller by handling the corresponding attribute. In our revocation mechanism, the revocation controller defines an attribute secret and a system-wide version number for each attribute in the system at the domain setup phase. The version number of the attribute indicates the evolution of the attribute's attribute secret, and all the version numbers are initially set to 1. Moreover, each version number is hashed to a version value which will be embedded into the corresponding attribute component in ciphertexts and user-attribute-keys. Whenever any revocation event takes place, the attribute secret for the involved attribute is replaced with a new one, and its version number is increased by 1. Since the attribute secret for the involved attribute is redefined, the corresponding attribute component in the public parameters, ciphertexts and non-revoked DUs' attribute-secret-keys must be updated in order to guarantee backward secrecy. In order to avoid the impracticality of the frequent update of the public parameters, we introduce ill-formed ciphertext and ciphertext-heal key components in our scheme, as mentioned above. The ciphertext-heal keys are generated by the revocation controller for the purpose of the subsequent data publication data instead of updating the corresponding component in the public parameters. The revocation controller also generates a ciphertext-update key for updating the corresponding attribute component in ciphertexts. Once CS receives the ciphertext-update key, it updates the corresponding attribute component in the ciphertexts by using proxy re-encryption technique [5] but in a unidirectional way. For the purpose of updating the corresponding attribute component in non-revoked DUs' user-attribute-keys, the revocation controller generates a unique key-update-key for each non-revoked DU on demand whenever the DU requests. The DU's request is triggered by the detection of whether any attribute component of his/her user-attribute-key is stateless. This detection is made upon the decryption process by matching the corresponding version values in both the ciphertext and the user-attribute-key. Upon receiving the key-update-key, the DU updates his/her key to its latest version. The increased version values are embedded within ciphertext-heal-keys, ciphertext-update-keys and key-update-keys; With these keys, the version values of the corresponding attribute components in ciphertexts and non-revoked DUs' user-attribute-keys are updated to their latest version as well.

4.2 Construction

Let $\mathcal{N}$ denote the set of all the authorities including CA and A_θ be an authority with the index θ. In addition, let U_θ denote a set of attributes managed by an A_θ and let $F \subseteq \mathcal{N}$ denote a set of the relevant authorities including CA from them some attributes are involved in the access structure for the encryption.

The construction of our scheme is presented as follows:

System Initialization: The CA sets up the system by running the *GSetup* algorithm and also sets up its own domain by running the *ASetup* algorithm. After setting up its domain, CA sends a ciphertext-heal-key related to the essential attribute to CS. In the Collaborative Cloud Storage System, any entity can simply act as an AA by setting up its own domain with the *ASetup* algorithm. After setting up its domain, it sends a set of ciphertext-heal-keys, each is related to an attribute of the domain, to CS.

- $GSetup(\lambda)$. It first chooses a bilinear group $\mathbb{G}_0$ of prime order p with generator g and a bilinear map $e : \mathbb{G}_0 \times \mathbb{G}_0 \to \mathbb{G}_1$. In addition, CA generates a pair of sign SK and verification VK keys. The global parameters are published as: $GP = (e, g, \mathbb{G}_0, \mathbb{G}_1, VK)$.
- $ASetup(GP)$. It first chooses randomly $\alpha_\theta, \beta_\theta, \delta_\theta, a_\theta \in \mathbb{Z}_p$ as its master key and $\{t_{x_\theta}, v_{x_\theta}\}_{x_\theta \in U_\theta} \in \mathbb{Z}_p$ as attribute secrets for the attributes in the domain. Next, it sets each attribute's version number ver_{x_θ} as 1 and selects a non-cryptographic hash function $h(\cdot)$ that maps a version number to a hash value. It then outputs the authority's master key MK_θ and public parameters PP_θ as follows:

$$\begin{aligned} MK_\theta &= (\alpha_\theta, \beta_\theta, \delta_\theta, a_\theta, \\ &\quad \{ver_{x_\theta} = 1, t_{x_\theta}, v_{x_\theta}\}_{x_\theta \in U_\theta}); \\ PP_\theta &= (e(g,g)^{\alpha_\theta}, g^{a_\theta}, g^{\delta_\theta \beta_\theta}, \\ &\quad \{h(ver_{x_\theta}), T_{x_\theta} = g^{t_{x_\theta}\beta_\theta}\}_{x_\theta \in U_\theta}). \end{aligned}$$

  Finally, it generates a set of ciphertext-heal-keys for all attributes in the domain as follows:

$$\{hk_{x_\theta}^{h(ver_{x_\theta})} = \frac{v_{x_\theta} - t_{x_\theta}}{\delta_\theta}\}_{x_\theta \in U_\theta}$$

User and Attribute Authorization: When a new DU joins the system, CA assigns a globally unique identifier *gid* to the DU and issues a user-central-key by running the *GKeyGen*, on which a signature of CA. Also, CA generates a user-attribute-key associated with the essential attribute for the DU by running the *AKeyGen* algorithm. Then, CA sends them to the DU through a secure channel. To get a user-attribute-key, any DU submits its user-central-key to an AA. Each AA authenticates any DU by verifying the signature on the DU's user-central-key with the verification key VK issued by CA. If the DU is a legal DU, then the AA entitles a set of attributes $\mathcal{S}_{gid,\theta}$ to the DU according to his/her role in the domain and then generates a user-attribute-key for the DU by running the *AKeyGen* algorithm. Then, the AA sends the user-attribute-key to the DU through a secure channel.

- $GKeygen(GP, gid)$. It first chooses a random $u_{id} \in \mathbb{Z}_p$ and generates a user-central-key for the DU as follows:

$$G_{gid} = g^{u_{id}}.$$

  Then, it signs on it with the sign-key SK to ensure its integrity.
- $AKeyGen(GP, MK_\theta, G_{gid}, \mathcal{S}_{gid,\theta})$. It generates a user-attribute-key which is associated with the DU's attribute set in the domain as follows:

$$\begin{aligned} AK_{gid,\theta} &= \Big(D_{1_\theta} = g^{\alpha_\theta}(G_{gid})^{a_\theta}, \forall x_\theta \in \mathcal{S}_{gid,\theta} : \\ &\quad \{h(ver_{x_\theta}), D_{x_\theta} = (G_{gid})^{v_{x_\theta}\beta_\theta}\}\Big) \end{aligned}$$

Data Publication: Before outsourcing data m, a DO defines an access structure (M, ρ). Then, the DO encrypts the data m under the (M, ρ) by running the *Encrypt* algorithm and submits an ill-formed ciphertext as the output to CS. After receiving the ill-formed ciphertext, CS converts it to a well-formed ciphertext by running the *CTConvert* algorithm. Finally, CS publishes the well-formed ciphertext to the collaborative cloud storage.

- $Encrypt(m, (M, \rho), GP, \{PP_\theta\}_{\theta \in F})$. Let M be an $l \times n$ matrix. The function ρ associates rows of M to attributes. Then, it chooses a random vector $\vec{v} = (s, y_2, ..., y_n) \in \mathbb{Z}_p^n$, where $y_2, ..., y_n$ will be used to share the encryption exponent s. For $i = 1$ to l, it calculates $\lambda_i = \vec{v} \cdot M_i$, where M_i is the vector corresponding to the i^{th} row of M. In addition, it randomly chooses $r_1, r_2, ..., r_l \in \mathbb{Z}_p^l$ and computes an ill-formed ciphertext as:

$$\begin{aligned} \tilde{ct} &= \Big(C_0 = m\Big(\prod_{\theta \in F} e(g,g)^{\alpha_\theta}\Big)^s, C_1 = g^s, \\ &\quad \{h(ver_{\rho(i)}), C_{i,1} = g^{r_i}, C_{i,2} = g^{-\delta_\theta \beta_\theta r_i}, \\ &\quad \tilde{C}_{i,3} = (\prod_{\theta \in F} g^{a_\theta})^{\lambda_i} T_{\rho(i)}^{-r_i}\}_{i=1...l}\Big). \end{aligned}$$

  Here, $h(ver_{\rho(i)})$ is set by the version value of the corresponding T_{x_θ} in the PP_θ, where $x_\theta = \rho(i)$.
- $CTConvert(\tilde{ct}, \{hk_{x_\theta}\}_{x_\theta = \rho(i), \theta \in F}^{h(ver_{x_\theta})})$. By using a set of ciphertext-heal keys $\{hk_{x_\theta}\}_{x_\theta = \rho(i), \theta \in F}$, it converts an ill-formed ciphertext to a well-formed ciphertext as:

$$\begin{aligned} CT &= \Big(C_0, C_1, \{h(ver_{\rho(i)}), C_{i,1}, C_{i,2}, \\ &\quad C_{i,3} = \tilde{C}_{i,3} \cdot (C_{i,2})^{hk_{\rho(i)}^{h(ver_{x_\theta})}}\}_{i=1...l}\Big). \end{aligned}$$

Here, $h(ver_{\rho(i)})$ is set by the version value of the corresponding ciphertext-heal-key $hk_{x_\theta}^{h(ver_{x_\theta})}$, where $x_\theta = \rho(i)$.

Data Retrieval: Any DU can freely get any ciphertext CT from CS and obtain the data by running the *Decrypt* algorithm if and only if the DU possesses sufficient attributes $\mathcal{S}_{gid} = \bigcup_{\theta \in F} \mathcal{S}_{gid,\theta}$ that satisfy the access structure (M, ρ) of the CT. Note that, in decryption process, the version values of the attribute components in the user-attribute-keys are matched with the version values of the corresponding components in the ciphertext for the purpose of detecting whether there is any component stateless in the DU's user-attribute-keys. If there is any mismatch, then it will trigger the DU to request a key-update-key for the attribute on which the version number mismatch occurs from CA or the corresponding AA.

- $Decrypt(CT, G_{gid}, \{AK_{gid,\theta}\}_{\theta \in F})$. Suppose that $\mathcal{S}_{gid} = \bigcup_{\theta \in F} \mathcal{S}_{gid,\theta}$ satisfies the access structure (M, ρ) and let $I \subseteq 1, 2, ..., l$ be defined as $I = \{i : \rho(i) \in \mathcal{S}_{gid}\}$. Then, it chooses a set of constants $\{w_i \in \mathbb{Z}_p\}_{i \in I}$ and reconstructs the secret s as $s = \sum_{i \in I} w_i \lambda_i$ if λ_i are valid shares of the secret s according to M. The decryption algorithm first computes:

$$\begin{aligned} K &= \frac{e(C_1, \prod_{\theta \in F} D_{1_\theta})}{\prod_{i \in I} \left(e(C_{i,1}, D_{\rho(i)}) e(C_{i,3}, G_{gid})\right)^{w_i}} \\ &= e(g,g)^{s \sum_{\theta \in F} \alpha_\theta}. \end{aligned}$$

Finally, it recovers the data by computing:

$$m = C_0 / K.$$

Attribute and User Revocation: When an attribute revocation event takes place, the AA that manages the involved attribute runs the $CKeyGen$ algorithm to generate a pair of ciphertext-heal-key and ciphertext-update-key for CS, where the keys are related to the involved attribute. In the case of user revocation event, the involved attribute will be the essential attribute and CA runs the $CKeyGen$ algorithm to generate a pair of ciphertext-heal-key and ciphertext-update-key related the essential attribute. Upon receiving the pair of keys, CS updates all the ciphertexts which include the involved attribute in their access structures by running the $CTUpdate$ algorithm. In the case of user revocation, all ciphertexts are updated since all ciphertexts include the essential attribute in its access structure. Note that CS keeps the new ciphertext-heal-key for subsequent data publications instead of the old one. The AA (or CA) generates a unique key-update-key for the non-revoked DU on demand by running the $UKeyGen$ algorithm. That is to say that the key-update-key generation is triggered by the request from any non-revoked DU. Upon receiving the unique key-update-key from the AA, the DU updates his/her corresponding user-attribute-key by running the $AKUpdate$ algorithm. To simplicity, we suppose that y_θ is the attribute involved in the revocation event.

- $CKeyGen(MK_\theta, y_\theta)$. It first randomly chooses $v'_{y_\theta} \in \mathbb{Z}_p$ as new attribute secret for y_θ, which is different from the previous secret $v'_{y_\theta} \neq v_{y_\theta}$ and increases the attribute's version number by 1 as $ver'_{y_\theta} = ver_{y_\theta} + 1$. Then, it generates a ciphertext-heal-key as follows:

$$hk_{y_\theta}^{h(ver'_{y_\theta})} = \frac{v'_{y_\theta} - t_{y_\theta}}{\delta_\theta}.$$

Next, it generates a ciphertext-update-key as follows:

$$ck_{y_\theta}^{h(ver'_{y_\theta})} = (ck' = \frac{v'_{y_\theta} - \delta_\theta}{\delta_\theta}, ck'' = v_{y_\theta}\beta_\theta - \delta_\theta\beta_\theta).$$

Finally, it updates the master key MK_θ into MK'_θ by replacing the old attribute secret v_{y_θ} for y_θ with new one v'_{y_θ} with the increased version number ver'_{y_θ}.

- $CTUpdate(CT, ck_{y_\theta}^{h(ver'_{y_\theta})})$. It updates only the corresponding component associated with y_θ and sets its version value as the version value associated with $ck_{y_\theta}^{h(ver'_{y_\theta})}$ as:

$$\begin{aligned} CT' = (C_0, C_1, \forall i : \{ &\text{ if } \rho(i) \neq y_\theta : \\ &h(ver(\rho(i)), C_{i,1}, C_{i,2}, C_{i,3}, \\ &\text{if } \rho(i) = y_\theta : h(ver'(\rho(i)), C_{i,1}, C_{i,2}, \\ &C'_{i,3} = C_{i,3} \cdot (C_{i,1})^{ck'} \cdot (C_{i,2})^{ck''} \}). \end{aligned}$$

- $UKeyGen(MK_\theta, y_\theta, G_{gid})$. It generates a unique key-update-key for the non-revoked DU as follows:

$$uk_{gid,y_\theta}^{h(ver'_{y_\theta})} = (G_{gid})^{v'_{y_\theta}\beta_\theta}.$$

- $AKUpdate(SK_{gid,\theta}, uk_{gid,y_\theta}^{h(ver'_{y_\theta})})$. It only replaces the corresponding component associated with the involved attribute and its version value with the corresponding values in $uk_{gid,y_\theta}^{h(ver'_{y_\theta})}$ as follows:

$$\begin{aligned} SK'_{gid,\theta} = (&D_{1_\theta}, y_\theta \in \mathcal{S}_{gid,\theta} : h(ver'_{y_\theta}), \\ &D'_{y_\theta} = uk_{gid,y_\theta}, \\ &\forall x_\theta \in \mathcal{S}_{gid,\theta} \setminus y_\theta : \{h(ver_{x_\theta}), D_{x_\theta}\}). \end{aligned}$$

4.3 Correctness

If the attribute set $\mathcal{S}$ satisfies the access structure, we have that $\sum_{i \in I} \lambda_i w_i = s$. Therefore:

$$\begin{aligned} K &= \frac{e(C_1, \prod_{\theta \in F} D_{1_\theta})}{\prod_{i \in I} \left(e(C_{i,1}, D_{\rho(i)}) e(C_{i,3}, G_{gid})\right)^{w_i}} \\ &= \frac{e(g^s, \prod_{\theta \in F} g^{\alpha_\theta} g^{u_{id}\alpha_\theta})}{\prod_{i \in I} \left(e(g^{r_i}, g^{u_{id} v_{x_\theta} \beta_\theta}) e((\prod_{\theta \in F} (g^{a_\theta})^{\lambda_i} g^{-v_{x_\theta}\beta_\theta r_i}, g^{u_{id}})\right)^{w_i}} \end{aligned}$$

Table 2: Security property comparison

Properties	[36]	[20]	[10]	Our
Collusion Resistance (type-A)	No	No	Yes	Yes
Collusion Resistance (type-B)	Yes	No	Yes	Yes
Forward Secrecy	Yes	No	No	Yes
Backward Secrecy	Yes	No	No	Yes

$$= \frac{e(g^s, g^{\sum_{\theta\in F}\alpha_\theta} g^{u_{id}\sum_{\theta\in F} a_\theta})}{\prod_{i\in I}\left(e(g^{r_i}, g^{u_{id}v_{x_\theta}\beta_\theta}) e((g^{\lambda_i\sum_{\theta\in F} a_\theta} g^{-v_{x_\theta}\beta_\theta r_i}, g^{u_{id}})\right)^{w_i}}$$
$$= \frac{e(g,g)^{s\sum_{\theta\in F}\alpha_\theta} e(g,g)^{u_{id}s\sum_{\theta\in F} a_\theta}}{\prod_{i\in I}\left(e(g,g)^{r_i u_{id} v_{x_\theta}\beta_\theta} e(g,g)^{\lambda_i u_{id}\sum_{\theta\in F} a_\theta} e(g,g)^{-v_{x_\theta}\beta_\theta r_i u_{id})}\right)^{w_i}}$$
$$= \frac{e(g,g)^{s\sum_{\theta\in F}\alpha_\theta} e(g,g)^{u_{id}s\sum_{\theta\in F} a_\theta}}{\prod_{i\in I}\left(e(g,g)^{\lambda_i u_{id}\sum_{\theta\in F} a_\theta}\right)^{w_i}}$$
$$= \frac{e(g,g)^{s\sum_{\theta\in F}\alpha_\theta} e(g,g)^{u_{id}s\sum_{\theta\in F} a_\theta}}{e(g,g)^{u_{id}\sum_{\theta\in F} a_\theta \sum_{i\in I}\lambda_i w_i}}$$
$$= \frac{e(g,g)^{s\sum_{\theta\in F}\alpha_\theta} e(g,g)^{u_{id}s\sum_{\theta\in F} a_\theta}}{e(g,g)^{u_{id}s\sum_{\theta\in F} a_\theta}}$$
$$= e(g,g)^{s\sum_{\theta\in F}\alpha_\theta}$$

Then, the data m is recovered as:

$$m = \frac{C_0}{K} = \frac{m\prod_{\theta\in F}(e(g,g)^{\alpha_\theta})^s}{e(g,g)^{s\sum_{\theta\in F_A}\alpha_\theta}} = \frac{me(g,g)^{s\sum_{\theta\in F}\alpha_\theta}}{e(g,g)^{s\sum_{\theta\in F}\alpha_\theta}}$$

5 Analysis of Our System

5.1 Security Analysis

As can be seen in Table 1, our scheme is the only scheme that supports both user revocation and attribute revocation. Therefore, we first give a comparison between the proposed scheme and the existing attribute-revocable MA-CP-ABE schemes [10, 20, 36] in terms of the security properties defined in Section 3.3.

We will prove the following theorem regarding the security of our Key-Escrow-Free Multi-Authority CP-ABE Scheme with Dual-Revocation.

Theorem 1. *If the decisional q-parallel BDHE assumption holds then all PPT adversary with challenge matrix of size $l^* \times n^*$, where $l^*, n^* \leq q$, have negligible advantage in selectively breaking our system.*

Proof. To prove the theorem, we will assume that there exists a PPT attacker $\mathcal{A}$ which has a non-negligible advantage $Adv_{\mathcal{A}}$ in selectively breaking our system. Moreover, we suppose the $\mathcal{A}$ chooses a challenge matrix (M^*, ρ^*) where both dimensions are at most q. Using this attacker, we will build a PPT simulator $\mathcal{B}$ that plays the decisional q-parallel BDHE problem with non-negligible advantage. Since we assume that the attacker can corrupt at most $|\mathcal{N}-1|$ authorities, the attacker cannot know the the master key of only non-corrupted authority. It implies that the security proof of our scheme can be proved under q-parallel BDHE assumption. Thus, in security proof, we will use only the terms of the q-parallel BDHE assumption instead of all the different g^{a_θ} terms of the relevant authorities.

Global Setup. The $\mathcal{B}$ sends the global parameters GP of the system to the $\mathcal{A}$.

Init. The $\mathcal{B}$ takes the q-parallel BDHE challenge $\vec{y}, T$. It receives a set of corrupted authorities $\mathcal{C} \subseteq \mathcal{N}$ and a challenge access structure (M^*, ρ^*) along with the target version numbers $\{ver^*_{x_\theta}\}_{x_\theta=\rho^*(i)}$ from the $\mathcal{A}$. We have that M^* is a $l^* \times n^*$ matrix, where $l^*, n^* < q$.

Authority Setup. For each non-corrupted authority $A_\theta \in \mathcal{N} - \mathcal{C}$, the $\mathcal{B}$ picks random exponents $\alpha'_\theta, \beta_\theta, \delta_\theta \in \mathbb{Z}_p$ and implicitly sets the master key of the authority to be $\alpha_\theta = \alpha'_\theta + a^{q+1}$ by letting

$$e(g,g)^{\alpha_\theta} = e(g^a, g^{a^q}) e(g,g)^{\alpha'_\theta}.$$

We describe how the simulator programs the public attribute keys $\{T_{x_\theta}\}_{x_\theta\in U}$, where $U = \bigcup_{\theta\in\mathcal{N}_A-\mathcal{C}_A} U_\theta$. For each $x_\theta \in U$, it chooses random values $z_{x_\theta}, v_{x_\theta} \in \mathbb{Z}_p$. Let X denote the set of indices i, such that $\rho^*(i) = x_\theta$. The simulator programs T_{x_θ} as:

$$T_{x_\theta} = \left(g^{z_{x_\theta}} \prod_{i\in X} g^{aM^*_{i,1}/b_i} \cdot g^{a^2 M^*_{i,2}/b_i} \cdots g^{a^{n^*} M^*_{i,n^*}/b_i}\right)^{\beta_\theta}$$

Note that if $X = \emptyset$ then we have $T_{x_\theta} = g^{z_{x_\theta}\beta_\theta}$. Also note that the public attribute keys are distributes randomly due to the $g^{z_{x_\theta}}$ and β_θ values.

Therefore, the public parameters of each non-corrupted authority A_θ are:

$$PP_\theta = \left(e(g,g)^{\alpha'_\theta}, g^{\delta_\theta\beta_\theta}, \{h(1), T_{x_\theta}\}_{x_\theta\in U_\theta}\right).$$

Moreover, the simulator selects randomly $ver^*_{x_\theta} - 1$ number of values $\{v'_{x_\theta}\}_{ver=2,\ldots,ver^*_{x_\theta}} \in \mathbb{Z}_p$ for each $x_\theta \in U_A$ and keeps them. Then, the simulator can calculate a set of ciphertext-heal-keys for all attributes that appear in the challenge access structure, where each ciphertext-heal-key can be calculated on its target version number by using v'_{x_θ} for version number $ver^*_{x_\theta}$.

$$hk_{x_\theta}^{h(ver^*_{x_\theta})} = \frac{v'_{x_\theta} - z_{x_\theta}}{\delta_\theta}.$$

Phase 1. In this phase, the simulator answers secret key and key-update key queries from the adversary. Without loss of generality, suppose the simulator is given a secret key query for $\mathcal{S}_{gid}$ for the version numbers of all the attributes in $\mathcal{S}_{gid}$ is 1, where $\mathcal{S}_{gid} = \bigcup_{\theta\in\mathcal{N}-\mathcal{C}} \mathcal{S}_{gid,\theta}$ is a set of attributes belonging to several non-corrupted authorities. Suppose $\mathcal{S}_{gid}$ does not satisfy M^* in combination with any keys that can be obtained from corrupted authorities.

Since $\mathcal{S}_{gid}$ does not satisfy $(M^*, \rho^*(i))$, there exist a vector $\vec{w} = (w_1, ..., w_{n^*}) \in \mathbb{Z}_p^{n^*}$ such that $w_1 = -1$ and $\vec{w} \cdot M^*_i = 0$ for all $i \in I = i | i \in [l] \wedge \rho^*(i) \in \mathcal{S}_{gid}$.

Then, the simulator chooses a random $u'_{id} \in \mathbb{Z}_p$ and implicitly sets u_{id} as

$$\begin{aligned} u_{id} &= u'_{id} + w_1 a^q + w_2 a^{q-1} + \cdots + w_{n^*} a^{q-n^*+1} \\ &= u'_{id} + \sum_{i=1\ldots n^*} w_i a^{q+1-i} \end{aligned}$$

This is properly distributed due to the u'_{id}. Then it calculates:

$$G_{gid} = g^{u_i d} = g^{u'_{id}} \prod_{i=1\ldots n^*} \left(g^{a^{q+1-i}}\right)^{w_i}$$

Now we describe how to create an user-attribute-key for every $\mathcal{S}_{gid,\theta} \subseteq \mathcal{S}_{gid}$. It has to create the user-attribute-keys separately since each subset of attributes $\mathcal{S}_{gid,\theta} \subseteq \mathcal{S}_{gid}$ belongs to different authority A_θ. Each user-attribute key consists of $D_{1_\theta}, \{D_{x_\theta}\}_{x_\theta \in \mathcal{S}_{gid,\theta}}$ components. Note that all attribute components in the user-attribute-keys are created on the version number 1. Then using the suitable terms from the assumption, the simulator calculates:

$$\begin{aligned} D_{1_\theta} &= g^{\alpha_\theta} g^{a u_{id}} = g^{a^{q+1}} g^{\alpha'_\theta} g^{a u'_{id}} \prod_{i=1\ldots n^*} g^{w_i a^{q+2-1}} \\ &= g^{\alpha'_\theta} (g^a)^{u'_{id}} \prod_{i=2\ldots n^*} \left(g^{a^{q+2-i}}\right)^{w_i} \end{aligned}$$

Now, for all $x_\theta \in \mathcal{S}_{gid,\theta}$, the simulator has to compute D_{x_θ} component. If $x_\theta \in \mathcal{S}_{gid,\theta}$ is not used in the access structure, for which there is no i such that $\rho^*(i) = x_\theta$, the the simulator can simply calculate:

$$D_{x_\theta} = \left(G_{gid}\right)^{v_{x_\theta}\beta_\theta}$$

If $x_\theta \in \mathcal{S}_{gid,\theta}$ is used in the access structure, then the simulator computes D_{x_θ} as follows:

$$D_{x_\theta} = \left(G_{gid}\right)^{v_{x_\theta}\beta_\theta} \prod_{i\in X} \prod_{j=1,\ldots,n^*} \Bigg(g^{(a^j/b_i)r} \cdot \prod_{k=1,\ldots,n^*, k\neq j} (g^{a^{q+1+j-k}/b_i})^{w_k}\Bigg)^{M^*_{i,j}\beta_\theta}$$

Towards key-update key queries, the simulator returns a key-update-key for the given attribute and on the given version number. Suppose that the simulator is given a key-update-key for (x_θ, G_{gid}) along with a version number t_{x_θ}, where $x_\theta \in \mathcal{S}_{gid}$ and $2 \leq t_{x_\theta} \leq ver^*_{x_\theta}$. By using v'_{x_θ} for the version number t_{x_θ}, the simulator calculates a key-update key $uk^{h(t_{x_\theta})}_{gid,x_\theta}$ for each x_θ in the similar way of calculating D_{x_θ} depending on whether x_θ is in the challenge access structure or not. If x_θ is not used in the access structure, the key-update key can calculate as:

$$uk^{h(t_{x_\theta})}_{gid,x_\theta} = \left(G_{gid}\right)^{v'_{x_\theta}\beta_\theta}$$

If x_θ is used in the access structure, then the key-update key can calculate as:

$$uk^{h(t_{x_\theta})}_{gid,x_\theta} = \left(G_{gid}\right)^{v'_{x_\theta}\beta_\theta} \prod_{i\in X} \prod_{j=1,\ldots,n^*} \Bigg(g^{(a^j/b_i)r} \cdot \prod_{k=1,\ldots,n^*, k\neq j} (g^{a^{q+1+j-k}/b_i})^{w_k}\Bigg)^{\beta_\theta M*_{i,j}}$$

Challenge. In this phase, we build the challenge ciphertext. The adversary gives two messages m_0, m_1 with equal length to the simulator, The simulator flips a coin b and contructs

$$C_0 = m_b \cdot T \cdot \prod_{\theta\in F_A} e(g, g^s)^{\alpha'_\theta}$$

and $C_1 = g^s$, where T is the challenge term and g^s is the corresponding term of the assumption.

The tricky part is to simulate the $C_{i,3}$ values since this contains the terms that must be cancelled out. However, the simulator can choose the secret splitting, such that these can be cancelled out. The simulator sets

$$\vec{v} = (s, sa + y'_2, sa^2 + y'_3, ..., sa^{n*-1} + y'_{n^*}) \in \mathbb{Z}_p^{n^*},$$

where $y'_2, ..., y'_{n^*} \in \mathbb{Z}_p$. We see that the secret s and the vector $\vec{v}$ are properly distributed, since s is information theoretically hidden from $\mathcal{A}$ and y'_i's are picked uniformly at random. As as result, since $\lambda_i = \vec{v} \cdot M^*_i$, we can construct the share of the secret as:

$$\lambda_i = \sum_{j=1\ldots n^*} M^*_{i,j} s a^{j-1} + \sum_{j=2,\ldots,n^*} M^*_{i,j} y'_j.$$

For each row, the simulator chooses a random r'_i and implicitly sets $r_i = r'_i + sb_i$. For $i = 1, ..., n^*$, we define R_i as the set of all $k \neq i$ such that $\rho^*(i) = \rho^*(k)$. That is the set of all other row indices that have the same attributes as row i. Using the above, the $\mathcal{B}$ calculates

$$\begin{aligned} C_{i,1} &= g^{r_i} g^{s b_i} \\ C_{i,2} &= (g^{-r_i} \cdot g^{-s b_i})^{\delta_\theta \beta_\theta} \\ \tilde{C}_{i,3} &= T^{-r'_i}_{\rho^*(i)} \Bigg(\prod_{j=2,\ldots,n^*} (g^a)^{M^*_{i,j} y'_j}\Bigg) (g^{b_i s})^{-z_{\rho^*(i)\beta_\theta}} \cdot \\ &\quad \Bigg(\prod_{k\in R_i} \prod_{j=1,\ldots,n^*} (g^{a^j \cdot s \cdot (b_i/b_k)})^{M^*_{k,j}\beta_\theta}\Bigg). \end{aligned}$$

Since $\tilde{C}_{i,3}$ is simulated in a ill-formed way, the simulator also must convert $\tilde{C}_{i,3}$ to a well-formed $C_{i,3}$ by using a set of ciphertext-heal keys $\{hk^{ver^*_{x_\theta}}_{x_\theta}\}$, where $x_\theta = \rho^*(i)$.

$$C_{i,3} = \tilde{C}_{i,3} \cdot (C_{i,2})^{hk^{ver^*_{\rho^*(i)}}_{\rho^*(i)}}.$$

Therefore, the $\mathcal{B}$ hands over the ciphertext $CT = ((M^*, \rho^*), C_0, C_1, \{C_{i,1}, C_{i,2}, C_{i,3}\})$ to the $\mathcal{A}$.

Phase 2. The same as Phase 1.

Guess. The adversary eventually outputs a guess $b^{'}$ for the challenge bit. If $b^{'} = b$ the simulator outputs 0 to guess that $T = e(g,g)^{a^{q+1}s}$. Otherwise, it outputs 1 to indicate that it believes T is a random group element in $\mathbb{G}_1$.

When T is a tuple the simulator $\mathcal{B}$ gives a perfect simulation so we have that

$$Pr\left[\mathcal{B}(\vec{y}, T = e(g,g)^{a^{q+1}s}) = 0\right] = \frac{1}{2} + Adv_{\mathcal{A}}.$$

When T is a random group element the message m_b is completely hidden from the adversary and we have that

$$Pr\left[\mathcal{B}(\vec{y}, T = R) = 0\right] = \frac{1}{2}.$$

Therefore, the $\mathcal{B}$ can play the decisional q-parallel BDHE game with non-negligible advantage.

□

5.2 Performance Analysis

As can bee seen in Table 1, our scheme is the only scheme that supports both user revocation and attribute revocation, whereas MA-CP-ABE schemes in [10, 20, 36] only support attribute revocation. However, the schemes in [10, 20] utilize composite order groups. The group operations in composite order groups –group exponentiations and group pairings– are several orders of magnitude slower than those in prime order groups. More information on the comparison between prime and composite groups can be found [12, 28]. Thus, we exclude those schemes that are set in composite order groups from performance comparisons, and we give performance analysis of our scheme by comparing with Yang et al.'s scheme [36] that utilizes prime order groups as the same as our scheme does. The comparison is made in terms of storage overhead, communication cost, and computational efficiency. Now, we describe the notations used in the comparisons. Let $|G|$, $|G_T|$, and $|Z_p|$ denote the size of an element in the source group, the target group, and the field Z_p, respectively. In addition, let u and a be the total number of users and attributes in the system, respectively. Moreover, let s, l, and f denote the number of attributes a user possess, the number of attributes in an access structure and the number of matched attributes in a decryption, respectively. Besides, r, k, a_k, and n_k denotes the number of AAs involved in an encryption, the number of AAs in the system, the number of attributes handled by the AA and the number of AAs from that a user holds his/her secret key, respectively. Furthermore, let n_x and n_c be the number of users who possess an attribute x and the number of ciphertexts including an attribute in its access structure, respectively. Finally, let E, E_T, and P denote exponentiation in the source group, exponentiation in the target group, and pairing operations, respectively.

Storage overhead. From Table 3, we can see the storage overhead on each entity in the schemes, namely cloud server (CS), certificate authority (CA), attribute authority (AA), data owner (DO), and data user (DU). In both schemes, the main storage overhead on CS comes from ciphertexts, but our scheme brings additional overhead on CS due to ciphertext-heal-keys. However, it helps to reduce the communication cost between DOs and the AA on each revocation event; this communication cost exists in the scheme [36]. As for the overhead on CA, the scheme [36] brings a considerable amount of overhead than our scheme does because of a pair of global public and secret keys of each user in the system. In both schemes, the main storage overhead on each AA comes from the attribute secrets of all the attributes managed by the AA. Besides that, the scheme in [36] causes a great deal of extra overhead on each AA because this scheme requires each AA to keep a pair of global public and secret keys of each user in the system. While only the public parameters and public attribute keys of the AAs involved in the encryption contribute the main storage overhead on a DO in the scheme [36], in our scheme the public parameters and public attribute keys of the AAs involved in encryption including CA contribute the main storage overhead on a DO. However, the total overhead on a DO in our scheme is less than that in [36]. In both schemes, the storage overhead on a DU comes from a secret key that consists of the DU's global key and keys related to his/her attribute set from different AAs. However, in our scheme the components of secret key associated to the essential attribute brings slightly additional overhead on a DU than that in [36]. In Table 3, the size of a secret key is expressed by all components possessed by a DU who has s number of attributes in the system.

Communication cost. In Table 4, we discuss the communication cost between the entities in the schemes. To distinguish the communication cost at initialization phase from that at revocation phase, we use the notations **I:** and **R:**, respectively. In the scheme [36], CA sends a pair of global public and secret keys of each user in the system to each AA; it brings considerable higher communication cost between CA and an AA. In both schemes, sending user-central-key/global-key to a DU results in the communication cost between CA and the DU, and the cost in our scheme is slightly higher than that in [36]. Due to the user revocation mechanism –the scheme [36] does not support– our scheme brings the communication cost between CA and CS, DU regarding the essential attribute that CA handles. In both schemes, the communication cost between CS and a DO, a DU mainly comes from the transmission of a ciphertext, and it is higher in the scheme [36] than that in our scheme. The communication cost between an AA

Table 3: Comparison of storage overhead

Ents	The scheme in [36]	Our scheme
CS	$(2+4l)\|G\|+\|G_T\|$	$(4+3l)\|G\|+\|G_T\|; a\|Z_p\|$
CA	$u\|Z_p\|+u\|G\|$	$6\|Z_p\|$
AA	$(2+a_k+u)\|Z_p\|+u\|G\|$	$(4+2a_k)\|Z_p\|$
DO	$((2+2a_k)\|G\|+\|G_T\|)r$	$((5+a_k)\|G\|+2\|G_T\|)r$
DU	$\|Z_p\|+(1+s+2n_k)\|G\|$	$(4+s+2n_k)\|G\|$

and a DO comes from the public parameters and public attribute keys managed by the AA, which is lower in our scheme. In both schemes, a secret key associated with a set of attributes from the AA contributes communication cost between the AA and the DU, whereas key-update-key brings the cost upon each attribute revocation event. This cost is nearly close in both schemes.

Computational efficiency. We present a theoretical analysis of the computational efficiency of our scheme in terms of key generation, encryption, decryption, ciphertext update and update key generation, as shown in Table 5 by comparing with the scheme [36]. The most time-consuming operations in ABE settings are group exponentiation and pairing. Therefore, we present only those operations in the analysis. In both schemes, the cost of key generation depends on the number of attributes the DU possess from the AA. The cost of update key generation upon each attribute revocation event in the scheme [36] depends on the number of non-revoked users for the involved attribute. In our scheme, the computational cost coming from update key generation is less than that in [36] since a key-update-key for a non-revoked DU is generated on demand in our scheme. In both schemes, the computational cost on a DO is occurred by encryption, and the cost linearly increases with the number of attributes expressing the access structure in the ciphertext. In both schemes, the cost by decryption depends on the number of attributes satisfying the access structure in the ciphertext, and the computation cost occurred by ciphertext update upon each attribute revocation event increases linearly as the number of ciphertexts including the involved attribute in their access structures. As can be seen from Table 5, our scheme is more efficient than Yang et al.'s scheme [36].

Table 4: Comparison of communication cost

Comp's	The scheme in [36]	Our scheme
CA&AA	$u\|Z_p\|+u\|G\|$	-
CA&CS	-	**I:** $\|Z_p\|$; **R:** $3\|Z_p\|$
CA&DU	$\|Z_p\|+\|G\|$	$5\|G\|$
CA&DO	-	$2\|G\|+\|G_T\|$
AA&CS	**R:** $2\|Z_p\|$	**I:** $a\|Z_p\|$; **R:** $3\|Z_p\|$
AA&DU	**I:** $(2+s_k)\|G\|$;**R:** $\|G\|$	**I:** $(2+s_k)\|G\|$; **R:** $\|G\|$
AA&DO	$(2+2n_k)\|G\|+\|G_T\|$	$\|Z_p\|+(1+n_k)\|G\|+\|G_T\|$
DO&CS	$(1+4l)\|G\|+\|G_T\|$	$(5+3l)\|G\|+\|G_T\|$
CS&DU	$(1+4l)\|G\|+\|G_T\|$	$(5+3l)\|G\|+\|G_T\|$

Table 5: Comparison of computational efficiency

Components	The scheme in [36]	Our scheme
KeyGeneration	$(3+2s)E$	$(1+s)E$
Encryption	$(2+5l)E+E_T$	$(1+4l)E+ET$
Decryption	$fE_T+(4f+2k)P$	$fE_T+(2f+1)P$
CTUpdate	$2n_cE$	$2n_cE$
KeyUpdate	n_xE	$\ll n_xE$

6 Conclusion

The existing MA-CP-ABE schemes cannot be applied to collaborative cloud data storage as data access control due to the key escrow problem and the absence of dual revocation mechanism. In this paper, we proposed a Key-Escrow-Free MA-CP-ABE scheme with Dual Revocation to overcome the issues existing in the previous works. The proposed scheme is key escrow free due to the fact that any access structure is formed in a way that attributes from an individual authority cannot satisfy it. In addition, dual revocation mechanism in the proposed scheme guarantees both forward and backward secrecy, and it resists any potential collusion attacks as well. Moreover, our scheme is set in groups of prime order and proved in the standard model under a standard assumption. Based on the comparison of MA-CP-ABE schemes regarding their functionalities, the comparison of attribute-revocable MA-CP-ABE schemes regarding security requirements for revocation, and performance analysis, we demonstrate that our proposed scheme is more suitable to be applied to collaborative cloud data storage as a secure and scalable access control.

Acknowledgments

This study was supported by International (Regional) Joint Research Project of China National Science Foundation under Grant No.61520106007. The authors gratefully acknowledge the anonymous reviewers for their valuable comments.

References

[1] B. Balusamy, P. V. Krishna, G. S. T. Arasi, and V. Chang, "A secured access control technique for cloud computing environment using attribute based hierarchical structure and token granting system," *International Journal of Network Security*, vol. 19, no. 4, pp. 559–572, 2017.

[2] M. Bayat, M. Aref, "An attribute based key agreement protocol resilient to KCI attack," *International*

Journal of Electronics and Information Engineering, vol. 2, no. 1, pp. 10–20, 2015.

[3] A. Beimel, *Secure Schemes for Secret Sharing and Key Distribution*. Technion, Haifa, Isreal: Ph.D Thesis, Isreal Institute of Technology, 1996.

[4] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proceedings of 2007 IEEE Symposium on Security and Privacy (SP'07)*, pp. 321–334, Berkeley, CA, May 2007.

[5] M. Blaze, G. Bleumer, and M. Strauss, "Divertible protocols and atomic proxy cryptography," in *Proceedings of International Conference on the Theory and Application of Cryptographic Techniques*, pp. 127–144, Espoo, Finland, May-June 1998.

[6] M. Chase, "Multi-authority attribute based encryption," in *Proceedings of 4th Theory of Cryptography Conference (TCC'07)*, pp. 515–534, Amsterdam, The Netherlands, Feb. 2007.

[7] L. Cheung and C. Newport, "Provably secure ciphertext policy abe," in *Proceedings of the 14th ACM conference on Computer and communications security (CCS07)*, pp. 456–465, Alexandria, Virginia, USA, Oct. 2007.

[8] S. Chow, "A framework of multi-authority attribute-based encryption with outsourcing and revocation," in *Proceedings of the 21st ACM on Symposium on Access Control Models and Technologies (SACMAT'16)*, pp. 215–226, Shanghai, China, June 2016.

[9] P. S. Chung, C. W. Liu, and M. S. Hwang, "A study of attribute-based proxy re-encryption scheme in cloud environments," *International Journal of Network Security*, vol. 16, no. 1, pp. 1-13, 2014.

[10] H. Cui and R. H. Deng, "Revocable and decentralized attribute-based encryption," *The Computer Journal*, vol. 59, no. 8, pp. 1220–1235, 2016.

[11] V. Goyal, A. Jain, O. Pandey, and A. Sahai, "Bounded ciphertext policy attribute based encryption," in *Proceedings of the 35th International Colloquium, ICALP 2008*, pp. 579–591, Reykjavik, Iceland, July 2008.

[12] A. Guillevic, "Comparing the pairing efficiency over composite-order and prime-order elliptic curves," in *Proceedings of the 11th International Conference on Applied Cryptography and Network Security (ACNS'13)*, pp. 357–372, 2013.

[13] J. Hur and D. Noh, "Attribute-based access control with efficient revocation in data outsourcing systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 7, pp. 1214–1221, 2011.

[14] L. Ibraimi, M. Petkovic, S. Nikova, P. Hartel, and W. Jonker, "Mediated ciphertext-policy attribute-based encryption and its application," in *10th International Workshop on Information Security Applications (WISA'09)*, pp. 309–323, 2009.

[15] L. Ibraimi, Q. Tang, P. Hartel, and W. Jonker, "Efficient and provable secure ciphertext-policy attribute-based encryption schemes," in *5th International Conference on Information Security Practice and Experience (ISPEC'09)*, pp. 1–12, Xi'an, China, Apr. 2009.

[16] S. Jahid, P. Mittal, and N. Borisov, "Easier: Encryption-based access control in social networks with efficient revocation," in *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security (ASIACCS'11)*, pp. 411–415, Hong Kong, China, Mar. 2011.

[17] C. C. Lee, P. S. Chung, M. S. Hwang, "A survey on attribute-based encryption schemes of access control in cloud environments," *International Journal of Network Security*, vol. 15, no. 4, pp. 231–240, July 2013.

[18] A. Lewko and B. Waters, "Decentralizing attribute-based encryption," in *Proceedings of 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 568–588, Tallinn, Estonia, May 2011.

[19] K. Li and H. Ma, "Outsourcing decryption of multi-authority ABE ciphertexts," *International Journal of Network Security*, vol. 16, pp. 286–294, 2014.

[20] Q. Li, J. Ma, R. Li, X. Liu, J. Xiong, and D. Chen, "Secure, efficient and revocable multi-authority access control system in cloud storage," *Computers & Security*, vol. 59, pp. 45–59, 2016.

[21] X. Liang, Z. Cao, H. Lin, and D. Xing, "Provably secure and efficient bounded ciphertext policy attribute based encryption," in *Proceedings of the 4th International Symposium on Information, Computer, and Communications Security (ASIACCS'09)*, pp. 343–452, Sydney, NSW, Australia, Mar. 2009.

[22] X. Liang, X. Lin R. Lu, and X. Shen, *Ciphertext Policy Attribute Based Encryption with Efficient Revocation*, Waterloo, Ontario, Canada: Technique Report, University of Waterloo, 2011.

[23] C. W. Liu, W. F. Hsien, C. C. Yang, and M. S. Hwang, "A survey of attribute-based access control with user revocation in cloud data storage", *International Journal of Network Security*, vol. 18, no. 5, pp. 900–916, 2016.

[24] Z. Liu, Z. Cao, Q. Huang, D. S. Wong, and T. H. Yuen, "Fully secure multi-authority ciphertext-policy attribute-based encryption without random oracles," in *Proceedings of 16th European Symposium on Research in Computer Security*, pp. 278–297, Leuven, Belgium, Sept. 2011.

[25] H. Ma, T. Peng, Z. Liu, "Directly revocable and verifiable key-policy attribute-based encryption for large universe," *International Journal of Network Security*, vol. 19, no. 2, pp. 272–284, 2017.

[26] S. Muller, S. Katzenbeisser, and C. Eckert, "On multi-authority ciphertext-policy attribute-based encryption," *Bulletin of the Korean Mathematical Society*, vol. 46, no. 4, pp. 803–819, 2009.

[27] M. Pirretti, P. Traynor, and P. Mcdaniel, "Secure attribute-based systems," in *Proceedings of the 13th ACM conference on Computer and communications*

security (CCS06), pp. 99–112, Alexandria, Virginia, USA, Oct. 2006.

[28] Y. Rouselakis and B. Waters, "Efficient statically-secure large-universe multi-authority attribute-based encryption," in *Proceedings of 19th International Conference (FC'15)*, pp. 315–332, San Juan, Puerto Rico, Jan. 2015.

[29] A. Sahai, H. Seyalioglu, and B. Waters, "Dynamic credentials and ciphertext delegation for attribute-based encryption," in *Proceedings of the 32nd Annual Cryptology Conference on Advances in Cryptology (CRYPTO'12)*, pp. 199–217, 2012.

[30] Y. Tian, Y. Peng, G. Gao, X. Peng, "Role-based access control for body area networks using attribute-based encryption in cloud storage," *International Journal of Network Security*, vol. 19, no. 5, pp. 720–726, 2017.

[31] L. Touati and Y. Challal, "Batch-based cp-abe with attribute revocation mechanism for the internet of things," in *Proceedings of International Conference on Computing, Networking and Communications (ICNC'15)*, pp. 1044–1049, Garden Grove, CA, Feb. 2015.

[32] G. Wang, Q. Liu, J. Wu, and M. Guo, "Hierarchical attribute-based encryption and scalable user revocation for sharing data in cloud servers," *Computers & Security*, vol. 30, no. 5, pp. 320–331, 2011.

[33] B. Waters, "Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization," in *Proceedings of the 14th International Conference on Practice and Theory in Public Key Cryptography Conference on Public Key Cryptography (PKC'11)*, pp. 53–70, Taormina, Italy, Mar. 2011.

[34] Z. Xu and K. Martin, "Dynamic user revocation and key refreshing for attribute-based encryption in cloud storage," in *Proceedings of the 11th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom'12)*, pp. 844–849, Liverpool, UK, June 2012.

[35] K. Xue, Y. Xue, J. Hong, W. Li, H. Yue, D. S. Wei, and P. Hong, "RAAC: Robust and auditable access control with multiple attribute authorities for public cloud storage," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 4, pp. 953–967, 2017.

[36] K. Yang and X. Jia, "Expressive, efficient, and revocable data access control for multi-authority cloud storage," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 7, pp. 1735–1744, 2014.

[37] K. Yang, X. Jia, and K. Ren, "Attribute-based fine-grained access control with efficient revocation in cloud storage systems," in *Proceedings of the 8th ACM SIGSAC Symposium on Information, Computer and Communications Security (ASIACCS'13)*, pp. 523–528, Hanzhou, China, May 2013.

[38] S. Yu, C. Wang, K. Ren, and W. Lou, "Attribute based data sharing with attribute revocation," in *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security (ASIACCS'10)*, pp. 261–270, Beijing, China, Apr. 2010.

[39] Y. Zhao, P. Fan, H. Cai, Z. Qin and H. Xiong, "Attribute-based encryption with non-monotonic access structures supporting fine-grained attribute revocation in M-healthcare," *International Journal of Network Security*, vol. 19, no. 6, pp. 1044–1052, 2017.

Biography

Nyamsuren Vaanchig is a Ph.D. candidate at the School of Information and Software Engineering, University of Electronic Science and Technology of China (UESTC). She received her M.Sc. degree from the Mongolian State University of Education (MSUE) in 2007. Her research interests include information security, cryptography, and cloud computing.

Hu Xiong is an associate professor at the School of Information and Software Engineering, UESTC. He received his Ph.D. degree from the School of Computer Science and Engineering, UESTC in 2009. His research interests include cryptographic protocols and network security.

Wei Chen is an associate professor at the School of Information and Software Engineering, UESTC. He received his Ph.D. degree from UESTC in 2010. His research interests include information security and software assurance.

Zhiguang Qin is a professor and the dean of School of Information and Software Engineering, UESTC. He received his Ph.D. degree from UESTC in 1996. He has engaged in distributed computing, information security, and electronic commerce research.