International Journal of Security and Its Applications
Vol. 10, No. 10 (2016) pp.303-320
http://dx.doi.org/10.14257/ijsia.2016.10.10.27

Fine-Grained Access Control for Cloud Data Sharing by Secure and Efficient Attribute-Revocable Ciphertext-Policy Attribute-Based Encryption

Nyamsuren Vaanchig[1*], Wei Chen[2] and Zhiguang Qin[3]

School of Information and Software Engineering, University of Electronic Science and Technology of China
[1]nyamsuren.v@gmail.com, [2]chenwei@uestc.edu.cn, [3]qinzg@uestc.edu.cn

Abstract

Nowadays, more and more users outsource their data to third party cloud storage servers for the purpose of sharing, so cloud data sharing becomes one of the popular services offered by cloud service providers. However, the third party storage servers in cloud data sharing systems, which are not fully trusted by data owners, make access control to the shared data a challenging issue. Although Ciphertext-Policy Attribute-Based Encryption (CP-ABE) is an emerging cryptographic solution for this issue, dealing with dynamic changes to users' access privileges (attribute revocation) in its practical applications as cloud data sharing systems is a real challenge. To overcome this challenge, we propose a fine-grained access control scheme for cloud data sharing systems by designing secure and efficient attribute-revocable CP-ABE scheme. Our scheme only allows non-revoked users in the attribute group to update their secret key by themselves using their unique key-update keys and the ciphertexts are updated by minimally trusted cloud server using a ciphertext-update key. Compared with the existing access controls achieved by attribute-revocable CP-ABE schemes, our proposed access control scheme reduces the trust degree of the cloud server in the attribute revocation mechanism. Furthermore, the analysis indicates that our access control scheme is more secure and efficient to apply to practical scenarios.

Keywords*: Access control, attributes revocation, ciphertext-policy attribute-based encryption, and cloud data sharing*

1. Introduction

Cloud data sharing is one of the popular services offered by cloud service providers; it allows data owners to store their data in third party data storage servers for the purpose of sharing. Moreover, cloud data sharing service offers great convenience to users such as anytime and anywhere access to the shared data and gives real benefits such as cost savings, productivity enhancements and so on. However, security concerns over the shared data come along with the benefits since cloud storage servers and data owners are not in the same domain; the cloud storage servers are not fully trusted by data owners. It raises a challenging issue for access control over the shared data, which addresses how data owners ensure that their data stored in cloud storage servers are accessed by authorized users.

CP-ABE scheme is a great achievement to resolve the issue of fine-grained access control over shared data in one-to-many communications, where data owners hold direct control over their access policies and the access policies are enforced cryptographically. In CP-ABE, each user is entitled to a set of attributes which are associated with the user's secret key, and a data owner chooses an access policy over a

* Corresponding Author

ISSN: 1738-9976 IJSIA

set of attributes and encrypts the message 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 fine-grained access control for cloud data sharing service. Since the first construction of CP-ABE proposed by J. Bethencourt *et. al.,* [2], researchers have studied and improved it by considering various aspects of it. The further improvements on CP-ABE can be found in [5], [7], [14], [12], [23], *etc*.

However, one of the major hurdles for practical applications of CP-ABE is how to revoke the attributes from the users. Whenever a user loses a part of his/her access privilege due to the users' role change in the system, the corresponding one or some attributes must be revoked from the user, and the user should not be able to decrypt the ciphertext which requires those attributes in decryption. The mechanism dealing with this issue is called attribute revocation. Designing attribute revocation mechanism in CP-ABE setting is not a straightforward task because each attribute can be shared by multiple users (an attribute group) and each user can possess more than one attribute (the user's attribute set). In other words, it is because of the fact that different users may have the same secret key in function related to the same attribute set. It also brings security concerns in the system in terms of collusion-resistance, forward secrecy, and backward secrecy. Moreover, dealing with attribute revocation in CP-ABE settings in a naive way causes massive overhead on both the key authority and data owners due to re-keying secret keys for a large number of non-revoked users and updating the ciphertexts, respectively. Therefore, attribute revocation issue is a major hindrance to enable CP-ABE as a fine-grained access control for cloud data sharing systems.

Attribute revocation issue was first addressed by M.Pirretti *et. al.,* [19] by setting an expiration date to users' attributes. Later on, L. Touati *et. al.,* [22] improved this solution by refreshing the necessary parts of the user secret keys at every time slot in order to reduce the overhead and complexity by comparing with [19]. However, these time-based approaches cause a security degradation problem in terms of forward secrecy until the next expiration time. Subsequently, many works have been proposed to achieve immediate attribute revocation mechanism in CP-ABE setting by concerning the drawbacks of the time-based attribute revocation mechanisms and the overhead on TA in the naïve attribute revocation mechanism. L. Ibraimi *et. al.,* [11] and F. Xu *et. al.,* [6] proposed similar solutions separately by splitting each user's secret key into two shares and by requiring the server to be in charge of the attribute revocation controller. In these two schemes, the server keeps one share of users' secret keys as decryption token and a list of revoked users for each attribute (user blacklist) and participates in each decryption process to decide whether to issue or refuse decryption token for the user according to the blacklists. S.Yu *et. al.,* [26] proposed a different solution, in which the tasks of ciphertext re-encryption and secret key update are shifted from trusted authority to proxy servers. The proxy servers are provided with a proxy re-key and a user blacklist for each attribute, and then decide that which users can update their keys according to the blacklists. To improve the scheme in [26], several works [9], [15], [16], and [17] have been proposed by concerning different aspects. J. Hur *et. al.,* [10] proposed another solution by also placing the server to be in charge of the attribute revocation controller as the same as the previous schemes. But, in this scheme, the server is provided with non-revoked user lists for each attribute (user white lists) instead of blacklists, and it delivers attribute group keys to the users those who are in the white list through attribute key distribution protocol. Referring to the scheme [10], the works in [24] and [4] have been proposed their solutions. Later on, L. Zu *et. al.,* [27] achieved attribute revocation in their scheme as similar as in [10], but the difference of this scheme is that the approach of delivering attribute group keys is through CP-ABE scheme. By concerning the massive overhead that occurred by attribute revocation event, S. Jahid *et. al.,* [13] proposed their architecture without

rekeying to non-revoked users and without re-encrypting existing ciphertexts. This scheme requires the server to participate decryption process by using a proxy key including user blacklists for the purpose of converting ciphertext in a form that the non-revoked user can decrypt. Although all the schemes mentioned above aimed to immediate attribute revocation mechanism in CP-ABE scheme in different ways, they require that the cloud server to be fully trusted. In other words, those schemes place the cloud server to be in charge of the revocation controller in attribute revocation mechanism such that the cloud server decides who can update their keys or who can decrypt the ciphertext according to the user black/white lists once attribute revocation event happens. As a result, these schemes make the system impurely CP-ABE based. Moreover, these schemes cannot guarantee forward secrecy of the shared data. Afterwards, K.Yang *et. al.,* [25] proposed an attribute-revocable CP-ABE scheme in order to deal with the problem of the fully trusted server by placing trusted authority to be in charge of updating non-revoked users' secret keys, and delegating the server to update ciphertexts with an update key provided by trusted authority. However, this scheme cannot resist the collusion attack from the server and any revoked user. As the updated ciphertexts are transformable to its old version if the server misuses the update key, the revoked users still can decrypt the ciphertexts from which the user's access privilege is revoked. Moreover, this scheme results in stateless user problem such that the users can not decrypt the ciphertexts to which they authorized to access if the non-revoked users miss more than one attribute revocation event without requesting trusted authority to update their keys. To the best of our knowledge, attribute revocation in CP-ABE scheme is still an open problem for further improvements in order to apply it as a fine-grained access control into cloud data sharing systems.

In this paper, we propose a scalable and fine-grained access control for cloud data sharing systems by designing an efficient and secure attribute-revocable CP-ABE scheme that overcomes the drawback of the existing attribute-revocable CP-ABE schemes. To accomplish our goal, we shall consider the following challenges: (1) how to achieve immediate attribute revocation mechanism without fully trusted server, (2) how to avoid stateless user problem that arises due to the attribute revocation event, (3) how to minimize the load on trusted authority in case of its' charge of revocation controller. In our scheme, we thus overcome the problem of the fully trusted server by placing a trusted authority to be in charge of revocation instead of the fully trusted server; the trusted authority generates a ciphertext-update key for the cloud server upon each attribute revocation event and a unique key-update key for each non-revoked user on-demand way. In addition, our scheme makes use of a minimally trusted server instead of fully trusted server to update the ciphertexts with the ciphertext-update key; the minimally trusted server has no power to give any access to the shared data by colluding with others. Furthermore, the load on the trusted authority is reduced in our scheme due to the fact that non-revoked users update their keys by themselves by requesting trusted authority to generate a unique key-update key on-demand way instead of requesting them upon each attribute revocation event. Moreover, by concerning the stateless user problem, we use a system-wide version number for each of attributes in our scheme, which indicates that the evolution of the attribute secret key. As a result, the analysis indicates that our scheme overcomes the drawbacks of the existing schemes, and it is secure and more efficient to be applied to the practical scenarios as cloud data sharing systems.

We organize the rest of the paper as follows. In Section 2, we review some preliminary knowledge used in this work. The definition of the system model, framework, security model, and security requirements are described in Section 3. We describe the main construction of our scheme in Section 4. Section 5 provides the analysis of our scheme in terms of comprehensive, security, and performance. Finally, we give a conclusion in Section 6.

2. Preliminaries

2.1. Access Structure

Let $\{P_1, P_2, ..., P_n\}$ *be a set of parties. A collection* $\mathrm{A} \subseteq 2^{\{P_1, P_2, ..., P_n\}}$ *is monotone if* $B \in \mathrm{A}$ *and* $B \subseteq C$ *implies* $C \in \mathrm{A}$. *An access structure [1] is a monotone collection* A *of non-empty subsets of* $\{P_1, P_2, ..., P_n\}$, *i.e.* $\mathrm{A} \subseteq 2^{\{P_1, P_2, ..., P_n\}}$, $\{\varnothing\}$. *The sets in* A *are called the authorized sets, and the sets not in* A *are called the unauthorized set.* In our context, the role of the parties is taken by the attributes. Thus, the access structure A will contain the authorized sets of attributes. Our scheme only supports monotone access structures.

2.2. Lagrange Interpolation

Let $i \in \square$ and $S \subseteq \square$, the Lagrange basis polynomial is defined as $\square_{i,S}(z) = \prod_{j \in S, j \neq i} (\frac{z-j}{i-j})$. Let $f(z) \in \square[z]$ be a d-th degree polynomial. If $|S| = d+1$, from a set of $d+1$ points $(i, f(i))_{i \in S}$, one reconstruct $f(z)$ as follows:

$$f(z) = \sum_{i \in S} f(i) \cdot \square_{i,S}(z) \tag{1}$$

2.3. Shamir Secret Sharing Scheme

In Shamir Secret Sharing Scheme [22], a secret s is shared among n parties by creating a polynomial f of degree t, such that $f(0) = s$ and each party i is given by the share $(i, f(i))$. Any group of t parties, where $t \leq n$, can reconstruct the secret s by using the Lagrange interpolation.

2.4. Bilinear Maps

Let G_0 and G_1 be two multiplicative cyclic groups of prime order p. Let g be a generator of G_0 and e be a bilinear map, $e: \mathrm{G}_0 \times \mathrm{G}_0 \to \mathrm{G}_1$ with the following properties:

1. *Bilinearity*: for all $u, v \in \mathrm{G}_0$ and $a, b \in \square_p$, 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 $\forall u, v \in \mathrm{G}_0$.

2.5. Decisional Bilinear Diffie-Hellman (DBDH) Assumption.

Suppose a challenger chooses a group G_0 of prime order p according to security parameter. Let $a, b, c, \theta \in \square_p$ be chosen at random and g be a generator of G_0. The DBDH assumption [21] states that no polynomial-time adversary is to be able to distinguish the tuple $(A = g^a, B = g^b, C = g^c, Z = e(g, g)^{abc})$ from the tuple $(A = g^a, B = g^b, C = g^c, Z = e(g, g)^{\theta})$ with more than a negligible advantage.

3. Definitions and Models

3.1. System Model

We consider a scalable and fine-grained access control scheme for cloud data sharing system as described in Figure 1. The system consists of four entities:

- ***Trusted Authority*** **(TA)** generates public and secret parameters of the system. It is in charge of entitling and revoking attributes to and from users according to their roles. TA is the only entity that is fully trusted by all entities participating in the system.

- ***Cloud Server*** **(CS)** stores data owners' data and provides data sharing service to data users. CS is not involved in access control enforcement and data decryption process. We assume that CS is minimally trusted hence it updates the ciphertexts with the ciphertext-update key provided by TA when attribute revocation takes place, but no power to give any access to the shared data by colluding with others.

- ***Data Owners*** **(DO)** define their access policies over a set of attributes, encrypt their data under the access policies, and publish the ciphertexts to CS for the purpose of sharing. Data access decisions are enforced cryptographically according to the access policies in the ciphertexts without depending on CS's decision and participation even if attribute revocation takes place.

- ***Data Users*** **(DU)** access the shared data stored in the system. Each user is entitled a set of attributes according to its role and unique identifier in the system. Any DU can freely get any ciphertext from CS and can decrypt it if and only if the set of attributes associated with the DU's secret key satisfies the access policy of the ciphertext. However, DUs' attribute set may dynamically change due to their role change in the system.

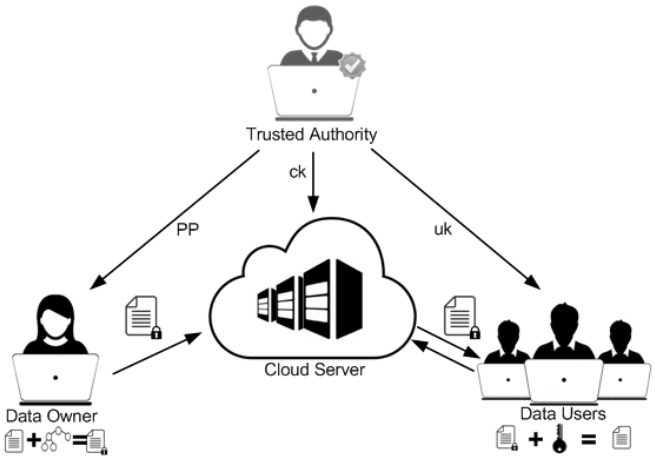

Figure 1. System Architecture of Cloud Data Sharing

3.2. Algorithm Definitions

The proposed access control scheme consists of the following algorithms:

- $Setup(\lambda) \rightarrow MK, PP, PK$. This algorithm, run by TA, takes as input security parameter λ , and it outputs the system master key MK , public parameters PP , and public attribute keys PK .

- $SKeygen(MK, ID, S_{ID}) \rightarrow SK_{ID}$. This algorithm, run by TA, takes as input the master key MK , a unique user identifier ID and a set of attributes S_{ID} , and it outputs a secret key SK_{ID} .

- $Encrypt(PP, PK, M, \mathrm{A}) \rightarrow CT$. This algorithm, run by DO, takes as input the public parameters PP , public attribute keys PK , a message M , an access

policy A , and it outputs a ciphertext CT that implicitly contains the access policy A .

- $Decrypt(CT, SK_{ID}) \rightarrow M \mid \perp$. This algorithm, run by DU, takes as input a ciphertext CT and the DU's secret key SK_{ID}, and it outputs a message M or $\perp$.

- $CKeygen(MK, PK, y) \rightarrow ck_y^{h(ver')}, MK', PK'$. This algorithm, run by TA, takes as input the master key MK , public attribute keys PK , and the revoked attribute y. It outputs a ciphertext-update key $ck_y^{h(ver')}$, an updated master key MK', and updated public attribute keys PK'.

- $UKeygen(MK', y, u_{id}) \rightarrow uk_{y,ID}^{h(ver')}$. This algorithm, run by TA, takes as input the updated master key MK', a secret value of the DU u_{id}, and an attribute y for which key-update key is requested. It outputs a key-update key $uk_{y,ID}^{h(ver')}$ for the non-revoked DU.

- $CTUpdate(CT, ck_y^{h(ver')}) \rightarrow CT'$. This algorithm, run by CS, takes as input a ciphertext CT and a ciphertext-update key $ck_y^{h(ver')}$, and it outputs an updated ciphertext CT'.

- $SKUpdate(SK_{ID}, uk_{y,ID}^{h(ver')}) \rightarrow SK'_{ID} \mid \perp$. This algorithm, run by the DU, takes as input the DU's secret key SK_{ID} and a unique key-update key $uk_{y,ID}^{h(ver')}$, and it outputs an updated secret key SK'_{ID}.

Definition 1(Correctness): *Our scheme is correct if PP , PK , and MK are generated by $Setup(\lambda)$, SK_{ID} is generated by $SKeygen(MK, ID, S_{ID})$ and CT is generated by $Encrypt(PP, PK, M, A)$, then*

1. $Decrypt(CT, SK_{ID}) = M$ if and only if the attribute set S_{ID} satisfies A and the version numbers of the corresponding components in the both of SK_{ID} and CT are equal.

2. Let $ck_y^{h(ver')}$ be generated by $CKeygen(MK, PK, y)$, $uk_{y,ID}^{h(ver')}$ be generated by $UKeygen(MK', y, u_{id})$, CT' be computed by $CTUpdate(CT, ck_y^{h(ver')})$, and SK'_{ID} be computed by $SKUpdate(SK_{ID}, uk_{y,ID}^{h(ver')})$, then $Decrypt(CT', SK'_{ID}) = M$ if and only if the attribute set S_{ID} satisfies A and the version numbers of the corresponding components in the SK'_{ID} and CT' are equal.

3.3. Security Model and Security Requirements

Here, we consider potential attackers in cloud data sharing system: 1) CS is assumed to be minimally trusted, but it might attempt to obtain the content of the shared data and give access permission to the unauthorized users even though it properly performs all tasks assigned by legitimate entities. 2) Users are assumed to be dishonest and malicious, so unauthorized users might attempt to obtain access to the shared data beyond their access privileges. We classified unauthorized users into two different types: a) an illegitimate user which refers to any user who does not have enough attributes satisfying the access policy of the ciphertext, and b) a revoked user which relates to any user from whom one or more attributes are revoked.

3.3.1. Security Model: Now we describe a security model for our attribute-revocable CP-ABE scheme. CPA security of our scheme under selective-attribute model [12] can

be defined by the following game between a challenger B and an adversary A, where the challenger simulates the protocol execution and answers queries from the adversary.

Init. The adversary A chooses the challenge access tree A^* and submits it to the B.

Setup. The B runs *Setup* algorithm to generate PP, PK, MK and gives PP, PK to the A.

Phase1. The A makes secret key queries by submitting any attribute set $S = \{x \mid x \in N\}$ with restriction that $x \notin \mathrm{A}^*$. The simulator B returns the corresponding secret keys.

Challenge. The adversary A submits two equal length messages M_0, M_1. The B flips a random coin b and encrypts M_b under the access tree A^*. Then, the challenge ciphertext CT^* is given to the adversary A.

Phase2. Phase 1 is repeated.

Guess. The adversary A outputs a guess b' of b.

The advantage of the adversary A in this game is defined as $\left| Pr[b' = b] - \frac{1}{2} \right|$.

Definition 2: *Attribute-revocable CP-ABE scheme is selectively secure if any polynomial time adversary has only a negligible advantage in the above game*.

3.3.2. Security Requirements: Our scheme also guarantees the following security requirements which are basic requirements of attribute revocation:

- ***Collusion-Resistance:*** Any unauthorized user should be prevented from decrypting any ciphertexts by colluding with authorized users or the cloud server.
- ***Forward Secrecy:*** Any revoked user who no longer possesses some attributes should be prevented from decrypting any ciphertext which requires those attributes to decrypt.
- ***Backward Secrecy***: Any new user or non-revoked user can decrypt any ciphertext if and only if the user has sufficient attributes to decrypt. Here, we use 'new user' to denote any user who newly joins an attribute group and 'non-revoked user' to denote any user who still possesses an attribute which is involved in any attribute revocation event.

4. Access Control by Secure and Efficient Attribute-Revocable CP-ABE

4.1. Scheme Overview

As described previously, we propose a scalable and fine-grained access control for cloud data sharing systems by designing a secure and efficient attribute-revocable CP-ABE scheme that does not require fully trusted cloud server in its attribute revocation mechanism. In our scheme, TA defines a secret attribute key and a system-wide version number for each attribute in the system. The version number indicates the evolution of the secret attribute key. Initially, all the version numbers are set to 1. When an attribute revocation takes place, the secret attribute key for the revoked attribute is replaced with a new one, and its version number is increased by 1. The corresponding component in the public attribute keys, ciphertexts, and secret keys must

be updated accordingly in order to guarantee backward secrecy and forward secrecy. To do so, TA updates the corresponding public attribute key of the revoked attribute and generates a ciphertext-update key for CS. Once CS receives the ciphertext-update key, it updates the corresponding attribute component in the ciphertexts by using proxy re-encryption technique [4] in a unidirectional way. TA generates key-update key uniquely for each non-revoked DU on-demand way. Any non-revoked DU updates his/her secret key by using his/her unique key-update key with the latest version after receiving it from TA. By considering the collusion attacks from a revoked DU and a non-revoked DU, TA is required to keep a list of secret values of the system users for the purpose of generating key-update key uniquely for each non-revoked DU.

4.2. Scheme Construction

Let G_0 and G_1 be multiplicative groups of prime order p, and let g be a generator of G_0. In addition, let $e: G_0 \times G_0 \to G_1$ denote the bilinear map and let h be a hash function that hashes a version number. In our construction, attributes are represented by their index values x and let $N = \{1, 2, ..., n\}$ be the attribute universe in the system. Let U denote the user universe in the system and ver_x denote the version number of an attribute x. In addition, let U_x denote a set of users who possess the attribute x.

The construction of our access control scheme consists of five phases: System Initialization, Key Generation, Data Encryption, Data Decryption, and Attribute Revocation.

4.2.1. System Initialization: TA initializes the system by running $Setup(\lambda)$ algorithm. It first chooses a bilinear group G_0 of prime order p with a generator g and a bilinear map $e: G_0 \times G_0 \to G_1$. Then, it selects a random element $\alpha \in \mathbb{Z}_p$ as the system master secret key and $\{\beta_x, t_x\}_{x \in N} \in \mathbb{Z}_p$ as secret attribute keys for all the attributes in the system. Next, it sets the version numbers of all the secret attribute keys $\{ver_x\}_{x \in N}$ as 1. Finally, it outputs the system public parameters PP, public attribute keys PK and the system master key MK as follows:

$$PP = (g, e(g,g)^{\alpha}, \{T_{x,1} = g^{\beta_x}\}_{x \in N}), PK = (\{h(ver_x), T_{x,2} = g^{t_x}\}_{x \in N}),$$
$$MK = (\alpha, \{ver_x, \beta_x, t_x\}_{x \in N}, h).$$

4.2.2. Key Generation: When a DU joins the system, TA first assigns a set of attributes $S_{ID} \subseteq N$ and a unique identifier ID to the DU. We assume that each user has a unique identifier $ID \in U$, but it is not involved in encryption and decryption algorithms. Then, TA generates a secret key associated with the attribute set S_{ID} by running $SKeygen(MK, ID, S_{ID})$ algorithm. The algorithm runs as follows: Firstly, it chooses random elements $u_{id} \in \mathbb{Z}_p$ and generates secret keys SK_{ID} as follows:

$$SK_{ID} = (D_0 = g^{\alpha - u_{id}}, \forall x \in S_{ID} : \{h(ver_x), D_x = g^{\frac{u_{id}}{t_x}}\}),$$

where u_{id} is a secret value for the DU with $ID \in U$. Note that TA is required to keep the secret value for each DU for the purpose of generating a unique key-update key for the DU on-demand way.

4.2.3. Data Encryption: Before publishing the data M to CS, the DO defines a access policy A over a set of attributes $Y \subseteq N$ and encrypts the data M under the A running by $Encrypt(PP, PK, M, \mathrm{A})$ algorithm. In our scheme, the access policy A is expressed

by an n-ary tree, in which each non-leaf node represents a threshold gate and each leaf node describes an attribute. Threshold gate is represented by the total number of child nodes n and the number of child node t necessary to reconstruct the secret assigned to the node, where $0 < t \le n$. The threshold gate is OF gate when $t \ne n$, the threshold gate is AND gate when $t = n$, and the threshold gate is OR gate when $t = 1$. The algorithm runs as follows: Firstly, it chooses a random element $s \in \square_p$, sets s to be the secret of the root node, marks all child nodes as unassigned, and marks the root node assigned. Next, starting from the root node to each unassigned non-leaf node, it recursively performs the following: It divides the secret s by using (t, n) Shamir's secret sharing technique, where t is defined differently depending on the threshold gate of the node. It assigns a share secret $s_x = f(x)$ to each child node and marks this node assigned. By using the secret share values of the leaves $j_x \in Y$, it finally computes a ciphertext CT as follows:

$$CT = (\mathrm{A}, C_0 = Me(g,g)^{\alpha s}, C_1 = g^s, \forall x_j \in Y : \{h(ver_{x_j}), C_{x_j,1} = T_{x,1}^{s_j}, C_{x_j,2} = T_{x,2}^{s_j}),$$

where j denote the index of the attribute in the access tree, and $h(ver_{x_j})$ is set by $h(ver_x)$ of the public component of the attribute x.

4.2.4. Data Decryption: Any DU can freely get any CT from CS and can obtain the data by running $Decrypt(CT, SK_{ID})$ algorithm if and only if the set of attributes associated with the DU's secret key SK_{ID} satisfies the access policy in the ciphertext CT. The algorithm runs as follows: It firstly checks whether the attribute set S_{ID} satisfies the access policy A. If not, it returns $\perp$. Otherwise, it chooses the smallest set $S_{ID}^* \subseteq S_{ID}$ that satisfies A. Next, it computes the following K_1 if and only if the version number of each attribute in S_{ID}^* is equal to the version number of the corresponding attributes $x_j \in Y$:

$$K_1 = \prod_{x \in S_{ID}^*} \left(e(C_{x_j,2}, D_x) \right)^{l_j(0)} = e(g,g)^{u_{id} s}, \quad (1)$$

where $l_j(0)$ is a Lagrange coefficient and can be computed by everyone who knows the index j of the attribute in the access tree. Note that this computation is completed by an optimized technique called "flatten" in [3]. Next, it computes

$$K_2 = K \cdot e(C_1, D_0) = e(g,g)^{\alpha s} \quad (2)$$

Finally, it can decrypt the data M by computing $M = C_0 / K_2$. Note that the version number match is made only for the purpose of detecting whether the DU's secret key is stateless or not. If the DU's secret key is detected as stateless, the DU asks TA to generate a unique key-update key with the latest version for the corresponding attribute component and update his/her secret keys by himself/herself.

4.2.5. Attribute Revocation: When a DU loses a part of his/her access privileges, the corresponding some attributes must be revoked from the DU and the DU should not be able to decrypt the ciphertexts which require those attributes in decryption. To simplicity, we suppose that an attribute y is revoked from a DU with ID'. Our attribute revocation mechanism consists of three phases: Ciphertext-Update Key Generation, Ciphertext Update, Key-Update Key Generation, and Secret Key Update.

Ciphertext-Update Key Generation. When attribute revocation takes places, TA runs $CKeyGen(MK, PK, y)$ algorithm. The algorithm runs as follows: It first randomly

chooses a new secret attribute key $t'_y \in \square_p (t'_y \neq t_y)$ for y. Then, it updates the corresponding components in MK and PK accordingly as $(ver_y + 1, t'_y)$ and $(h(ver_y + 1), T'_y = g^{t'_y})$ for the purpose of backward secrecy. Next, it computes a ciphertext-update key $ck_y^{h(ver')}$ for CS as follows:

$$ck_y^{h(ver')} = \frac{t'_y - \beta_y}{t_y},$$

where the version number $h(ver')$ is set by the version number of the attribute secret key of y, as $h(ver') = h(ver_y + 1)$. Finally, TA sends $ck_y^{h(ver')}$ to CS.

Ciphertext Update. Upon receiving $ck_y^{h(ver')}$, CS updates all the ciphertexts which include the revoked attribute y by running $CTUpdate(CT, ck_y^{h(ver')})$ algorithm. The algorithm only updates the corresponding component associated with the revoked attribute y in the ciphertext CT and it sets its version number as the version number associated with $ck_y^{h(ver')}$ as follows:

$$CT' = (\mathrm{A}, C_0, C_1, y_j \in Y : h(ver'_{y_j}), C_{y_j,1}, C'_{y_j,2} = (C_{y_j,2})^{ck_y^{h(ver')}} \cdot C_{y_j,1},$$
$$\forall x_j \in Y, \quad y_j : \{h(ver_{x_j}), C_{x_j,1}, C_{x_j,2}\})$$

Update Key Generation. Upon receiving a request from any non-revoked DU, TA runs $UKeyGen(MK', u_{id}, y)$ algorithm. The algorithm generates a unique key-update key by using the DU's secret value u_{id} and the secret attribute key t'_y for y, which is in MK', as follows:

$$uk_{y,ID}^{h(ver')} = g^{\frac{u_{id}}{t'_y}}.$$

Here, the version number $h(ver')$ is set by the version number of the attribute secret key of y. Finally, TA returns $uk_{y,ID}^{h(ver')}$ to the DU.

Secret Key Update. After receiving a unique key-update key from TA, the DU updates his/her secret key to the latest version by running $SKUpdate(SK_{ID}, uk_{y,ID}^{ver+1})$ algorithm. It updates the secret key as follows:

$$SK'_{ID} = (D_0, y \in S : h(ver'_y), D'_y = uk_{y,ID}^{h(ver')}, \forall x \in S, \quad y : \{h(ver_x), D_x\},$$

where the version number of the updated component is set by the version number associated with $uk_{y,ID}^{h(ver')}$.

4.3. Correctness. We can verify the correctness of the decryption process. Each component of formula (1) is extended as:

$$K_1 = \prod_{x \in S_{ID}^*} \left(e(C_{x_j,1}, D_x)\right)^{l_j(0)} = \prod_{x \in S_{ID}^*} \left(e(T_{x,2}^{s_j}, g^{\frac{u_{id}}{t_x}})\right)^{l_j(0)}$$
$$= \prod_{x \in S_{ID}^*} \left(e(g^{t_x s_j}, g^{\frac{u_{id}}{t_x}})\right)^{l_j(0)} = \prod_{x \in S_{ID}^*} \left(e(g,g)^{s_j u_{id}}\right)^{l_j(0)}$$
$$= e(g,g)^{u_{id} \sum_{x \in S_{ID}^*} s_j l_j(0)} = e(g,g)^{u_{id} s}$$

Then, the following computation according to formula (2) gives us $e(g,g)^{\alpha s}$ which is used to recover M from C_0.

$$K_2 = K_1 \cdot e(C_1, D_0)$$
$$= e(g,g)^{u_{id}s} e(g^s, g^{\alpha - u_{id}})$$
$$= e(g,g)^{u_{id}s} e(g,g)^{\alpha s} e(g,g)^{-u_{id}s}$$
$$= e(g,g)^{\alpha s} \quad .$$

5. Analysis of Our Proposed Access Control

5.1. Comprehensive Analysis

We now provide a comprehensive analysis by comparing the proposed access control for cloud data sharing system with the existing access controls that are based on attribute-revocable CP-ABE schemes, as shown in *Table1*. In the comprehensive analysis, scalability refers to flexibility concerning the number of users in the system and the number of revocable users for each attribute, and independent retrieval term relates to the fact that whether authorized users can retrieve data in the nature of CP-ABE that is without depending on the cloud server's decision and participation. The rest of the terms refer to the security requirements defined in Section 3.3 and the roles of the entities in the attribute revocation mechanism.

Table 1. Comparison of Access Controls with Attribute-Revocable CP-ABE

Properties	Our scheme	[11], [6]	[26],[9],[15], [16],[17]	[13]	[10],[24],[4]	[25]
Scalability	Yes	Yes	Yes	No	No	Yes
Independent Retrieval	Yes	No	No	No	No	Yes
Forward secrecy	Yes	No	Yes	Yes	No	Yes
Backward secrecy	Yes	Yes	Yes	Yes	Yes	No
Collusion resistance	Yes	No	No	Yes	No	No
Revocation controller	TA	CS	CS	CS	CS	TA
Ciphertext Updater	CS	-	CS	CS	CS	CS
Key Updater	DU	-	CS	-	DU	TA

5.2. Security Analysis

Now, we analyze the security properties of our scheme by the following theorems.

Theorem 1: *Suppose the DBDH assumption holds. Then no polynomial time adversary can selectively break our system with a challenge access tree* A^*.

Proof. For our scheme under the DBDH assumption, no polynomial-time adversary is to be able to distinguish $e(g,g)^{as}$ from any random element in the group G_1 with more than a negligible advantage. Therefore, our scheme guarantees data confidentiality of the shared data against any unauthorized access. a). Any illegitimate user who does not hold sufficient attributes that satisfy the access policy of the ciphertext and any revoked user whose attribute set no longer satisfies the access policy of the ciphertext cannot recover $e(g,g)^{u_{id}s}$. Without $e(g,g)^{u_{id}s}$, the unauthorized/revoked users cannot compute message encapsulation key $e(g,g)^{as}$ such that they cannot succeed in the decryption. The full proof follows from [12]. b). Cloud

server that is not fully trusted by data owners has no power to decrypt any ciphertext since data access decisions are made cryptographically without depending on it□ .

Theorem 2: *Our scheme is collusion-resistance against any colluders.*

Proof. a). Different legitimate users who do not have sufficient attributes individually satisfying the access policy A of the ciphertext *CT* cannot succeed in decrypting it by pooling their attribute sets. In order to decrypt the *CT* the colluding users need to recover $e(g,g)^{u_{id}s}$ by pairing $C_{x_j,2}$ and D_x, respectively from the colluding users' secret keys for each attribute x in the pooled attribute set S^* satisfying A of the *CT* . However, it cannot be recovered by the colluding users since $SKeygen(MK, ID, S_{ID})$ algorithm generates each user's secret key by using a secret key randomization technique. b). The collusion between revoked users from whom one or more attribute are revoked and non-revoked users who still hold those attributes cannot succeed in our scheme since each key-update key is distinguished for each non-revoked users by using the user's secret value u_{id} . b). The collusion between the cloud server and illegitimate/revoked users cannot succeed in our scheme. The server has no power to give access privileges to the users even though the server is a charge of updating ciphertexts *CT* to *CT'* with a ciphertext-update key $ck_x^{h(ver')}$ in a unidirectional way. Thus, collusion-resistance against the above types of colluders is guaranteed in our scheme.□

Theorem 3: *Our scheme guarantees forward secrecy of any shared data against revoked users.*

Proof. When a user is revoked from an attribute group, the secret attribute key t_x for the involved attribute group x is replaced with new one t_x', and the cloud server updates the corresponding attribute component in ciphertexts by using a ciphertext-update key $ck_x^{h(ver')} = (t_x' - \beta_x) / t_x$. Due to the different values of secret attribute keys ($t_x' \neq t_x$) in the updated ciphertexts and the secret key of the revoked user, the revoked users cannot succeed in decrypting the ciphertexts. Therefore, our scheme guarantees forward secrecy of the shared data which was previously able to be accessed by the revoked users. □

Theorem 4: *Our scheme guarantees the backward secrecy of any shared data in cloud data sharing system.*

Proof. In the context of cloud data sharing with CP-ABE, both non-revoked users who still belong to and new users who newly joins an attribute group *x* are able to decrypt the ciphertexts that require the attribute *x* in decryption as long as the users individually have sufficient attributes S_{ID} satisfying the access policies A of the ciphertexts. It refers to backward secrecy in our system. When an attribute revocation event takes place, the authority chooses a new secret attribute key t_x' for the attribute x, and then it updates the corresponding public attribute key in PK . TA also generates a unique key-update key $uk_{x,ID}^{h(ver')} = g^{u_{id}/t_x'}$ for the non-revoked user on-demand way. With the unique key-update key, each non-revoked user can update his/her key by himself/herself. Moreover, the attribute component associated with the revoked attribute x in the ciphertexts is updated to the latest version with a new ciphertext-update key. Thus, our scheme guarantees backward secrecy of any shared data such that they are able to be accessed by any authorized users. □

5.3. Performance Analysis

From *Table 1*, we can see that the schemes except for our scheme and the scheme in [25] are not applicable to cloud data sharing system because they make the system impurely CP-ABE based system that require fully trusted cloud server to be in charge of revocation controller. Thus, we give performance analysis of our scheme by comparing with the scheme in [25] that addresses the problem of the fully trusted cloud server in attribute revocation mechanism as similar as our scheme. The comparisons are made in terms of storage overhead, communication cost, and computational efficiency. Let $|p|, |g_0|$, and $|g_1|$ denote the size of an element in $\square_p$, G_0, and G_1 respectively. Also, let s and l denote the number of attributes that are associated with the secret key and the access policy in the ciphertext, respectively. Moreover, let n_a denote the total number of attributes in the system and n_u denote the total number of users in the system.

5.3.1. Storage Overhead: From *Table* 2, we can see the storage overhead on each entity in the schemes. In both schemes, the main storage overhead on TA comes from the master key and a secret attribute key for each attribute. Besides this, our scheme brings additional storage overhead on TA due to secret value for each user in the system. In both schemes, the public parameters and public attribute keys contribute the storage overhead on DO. The storage overhead on DU comes from secret key, whereas ciphertexts contribute the storage overhead on CS. The comparison shows that our scheme brings less storage overhead on DU, DO and CS than the scheme in [25].

Table 2. Comparison of Storage Overhead

Entity	Our scheme	Yang's scheme[25]
TA	$(1+2n_a)\|p\|+n_u\|p\|$	$(4+n_a)\|p\|$
DO	$(1+2n_a)\|g_0\|+\|g_1\|$	$(4+2n_a)\|g_0\|+\|g_1\|$
DU	$(1+s)\|g_0\|$	$(2+s)\|g_0\|$
CS	$(1+2l)\|g_0\|+\|g_1\|$	$(1+3l)\|g_0\|+\|g_1\|$

5.3.2. Communication Cost: *Table* 3 describes a comparison of the size of the components involved in the schemes: master key *MK* , public parameters *PK*, ciphertext *CT*, secret key *SK* and update key *UK*. The *PK* that includes public parameters and public attribute keys contribute to the communication cost between TA and DO. The communication cost between CS and DO, DU comes from the transmission of *CT*. The main communication cost between TA and DU comes from both *SK* and *UK*. The comparison shows that our scheme incurs less communication cost than [25].

Table 3. Comparison of Component Size

Component	Our scheme	Yang's scheme[25]
PK	$(1+2n_a)\|g_0\|+\|g_1\|$	$(4+2n_a)\|g_0\|+\|g_1\|$
SK	$(1+s)\|g_0\|$	$(2+s)\|g_0\|$
CT	$(1+2l)\|g_0\|+\|g_T\|$	$(1+3l)\|g_0\|+\|g_T\|$
UK	$\|g_0\|$	$3\|g_0\|$

5.3.3. Computational Efficiency: We implemented the constructions of our scheme and Yang's scheme [25] to provide some information on the computational efficiency of the schemes. The implementation uses the PBC library and 160-bit elliptic curve group based on the supersingular curve $y^2 = x^3 + x$ over a 512-bit finite field. All the experiments were carried out on 2.50GHz, Intel(R) Core(TM) i5, 4GB memory and running Ubuntu 14.04. The size of the plaintext is set to be 1KB. All the experiment results are the average of 30 trials.

We compare the performance of our scheme with the performance of Yang's scheme [25] in terms of key generation, data encryption, data decryption, and attribute revocation phases. *Figure 2* displays measurements of time to generate a secret key, time to encrypt data, time to update secret key, and time to update ciphertext. From *Figure 2.a*, we can see that key generation time is linear with the number of attributes in a secret key in both schemes, but key generation phase of our scheme is more efficient than the one in [25]. To test encryption and decryption, we randomly generated 10 different policies for each of the desired number of leaves (10, 20... 100). *Figure 2.b* presents that encryption time of both schemes scales linearly with the number of leaf nodes in the access policy, but data encryption phase of our scheme takes less time than that in [25]. The decryption time depends on the access policy used in the encryption and the attributes involved, and decryption phases of both schemes take nearly the same amount of time (not shown in the Figure). In attribute revocation phase, both schemes require to update those components associated with the revoked attributes in both the ciphertexts and secret keys. Thus, the computational efficiency of attribute revocation phase is shown by the results of both secret key update and ciphertext update sub-phases. *Figure 2.c* shows the total time of generating a key-update key and updating a secret key by using the key-update key, and *Figure 2.d* shows that the total time of generating a ciphertext-update key and updating a ciphertext by using the ciphertext-update key. From the experiment results, we can see that attribute revocation phase of our scheme is efficient than that in [25].

Therefore, the comparison of computational efficiency demonstrates that our scheme is more efficient than [25] in each phase of the construction.

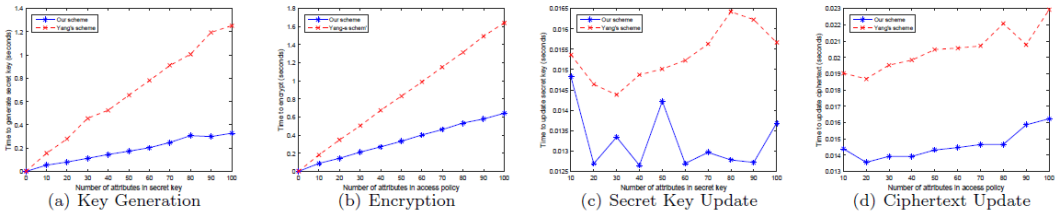

Figure 2. Comparison of Computational Efficiency

6. Conclusion

Addressing attribute revocation issue in CP-ABE settings, recently many attribute-revocable CP-ABE schemes have been proposed. However, the existing attribute-revocable CP-ABE schemes are not applicable to cloud data sharing systems as a secure and efficient fine-grained access control due to the fact that those schemes require that the cloud server is required to be fully trusted in order to achieve attribute revocation mechanism. In this paper, we proposed a fine-grained access control for cloud data sharing systems by designing a secure and efficient attribute-revocable CP-ABE scheme. One desirable property of our proposed scheme is that trust degree of the cloud server is reduced in attribute revocation mechanism such that data access decision is made in nature of CP-ABE scheme without relying on the cloud server's decision and participation even if any attribute revocation event takes place. In addition,

the comprehensive analysis, security analysis, and performance analysis indicate that our scheme is scalable, secure and efficient. Therefore, we demonstrate that our proposed access control scheme is more suitable to apply into cloud data sharing systems.

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] A. Beimel, "Secure Schemes for Secret Sharing and Key Distribution", Ph.D. Thesis, Israel Institute of Technology, **(1996)**.
[2] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption", Proceedings of 2007 IEEE Symposium on Security and Privacy (SP '07), Berkeley, CA, **(2007)** May 20-23.
[3] M. Blaze, G. Bleumer and M. Strauss, "Divertible protocols and atomic proxy cryptography", Proceedings of International Conference on the Theory and Application of Cryptographic Techniques, Espoo, Finland, **(1998)** May 31-June 4.
[4] C. Bodong, X. LiaoTe, Y. Junyan, M. Meng and T. Zhuo, "A scheme supporting efficient attribute revocation for cloud storage based on cpabe", Proceedings of 3rd International Conference on Computer Science and Service System (CSSS 2014), **(2014)**.
[5] L. Cheung and C. Newport, "Provably Secure Ciphertext-Policy ABE", in Proceedings of the 14th ACM conference on Computer and communications security (CCS07), Alexandria, Virginia, USA, **(2007)** October 29 – November 2.
[6] X. Fu and Z. Wu, "Ciphertext-Policy Attribute-Based Encryption with Immediate Attribute Revocation for Fine-Grained Access Control in Cloud Storage", Proceedings of International Conference on Communications, Circuits and Systems (ICCCAS), Chengdu, China, **(2013)** November 15-17.
[7] V. Goyal, A. Jain, O. Pandey and A. Sahai, "Bounded Ciphertext Policy Attribute Based Encryption", Proceedings of 35th International Colloquium (ICALP 2008), Reykjavik, Iceland, **(2008)** July 7-11.
[8] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data", Proceedings of the 13th ACM conference on Computer and communications security (CCS06), Alexandria, Virginia, USA, **(2006)** October 30-November 3.
[9] L. Hui and G. Lijun, "An Efficient Ciphertext-Policy Attribute-Based Encryption Scheme with Attribute Adjunction/Revocation in Data Sharing System", http://www.paper.edu.cn, **(2009)**, pp. 1-13.
[10] 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, **(2011)**, pp. 1214-1221.
[11] L. Ibraimi, M. Petkovic, S. Nikova, P. Hartel and W. Jonker, "Mediated Ciphertext-Policy Attribute-Based Encryption and Its Application", 10th International Workshop (WISA 2009), Busan, Korea, **(2009)** August 25-27.
[12] L. Ibraimi, Q. Tang, P. Hartel, and W. Jonker, "Efficient and Provable Secure Ciphertext-Policy Attribute-Based Encryption Schemes", Proceedings of the 5th International Conference (ISPEC 2009), Xian, China, **(2009)** April 13-15.
[13] S. Jahid, P. Mittal and N. Borisov, "EASiER: Encryption-Based Access Control in Social Networks with Efficient Revocation" Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security ASIACCS '11, Hong Kong, China, **(2011)** March 22-24.
[14] X. Liang, Z. Cao, H. Lin and D. Xing, "Provably Secure and Efficient Bounded Ciphertext Policy Attribute Based Encryption", Proceedings of the 4th International Symposium on Information, Computer, and Communications Security ASIACCS '09, Sydney, NSW, Australia, **(2009)** March 10-12.
[15] Y. Ming, L. Fan, H. Jing-Li and W. Zhao-Li, "An Efficient Attribute Based Encryption Scheme with Revocation for Outsourced Data Sharing Control", Proceedings of First International Conference on Instrumentation, Measurement, Computer, Communication and Control, Beijing, China, **(2011)** October 21-23.
[16] T. Naruse, M. Mohri and Y. Shiraishi, "Attribute-Based Encryption with Attribute Revocation and Grant Function Using Proxy Re-encryption and Attribute Key for Updating", Future Information Technology, vol. 276, **(2014)**, pp. 119-125.

[17] N. Pan, L. Sun and X. Mao, "CP-ABE Scheme with Revocation for Cloud Storage", Proceedings of International Conference on Computer Science and Information Technology, Kunming, China, **(2013)** September 21-23.
[18] L. Pang, J. Yang and Z. Jiang, "A Survey of Research Progress and Development Tendency of Attribute-Based Encryption", The Scientific World Journal, vol. 2014, **(2014)**, 13 pages.
[19] M. Pirretti, P. Traynor and P. Mcdaniel, "Secure Attribute-Based Systems", Proceedings of the 13th ACM conference on Computer and communications security (CCS06), Alexandria, Virginia, USA, **(2006)** October 30-November 3.
[20] A. Sahai and B. Waters, "Fuzzy Identity-Based Encryption", Proceedings of 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, **(2005)** May 22-26.
[21] A. Shamir, "How To Share a Secret", Communications of the ACM (CACM), vol. 22, no. 11, **(1979)**, pp. 612-613.
[22] L. Touati and Y. Challal, "Batch-based CP-ABE with attribute revocation mechanism for the Internet of Things", Proceedings of International Conference on Computing, Networking and Communications (ICNC), Garden Grove, CA, **(2015)** February 16-19.
[23] B. Waters, "Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization", Proceedings of the 14th International conference on Practice and theory in public key cryptography conference on Public key cryptography PKC'11, Taormina, Italy, **(2011)** March 6-9.
[24] X. Xie, H. Ma, J. Li and X. Chen, "An Efficient Ciphertext-Policy Attribute-Based Access Control towards Revocation in Cloud Computing", Journal of Universal Computer Science, vol. 19, no. 16, **(2013)**, pp. 2349-2367.
[25] K. Yang, X. Jia and K. Ren, "Attribute-Based Fine-grained Access Control With Efficient Revocation in Cloud Storage Systems", Proceedings of the 8th ACM SIGSAC symposium on Information, computer and communications security ASIA CCS'13, Hanzhou, China, **(2013)** May 8-10.
[26] S. Yu, C. Wang, K. Ren and W. Lou, "Attribute based data sharing with attribute revocation", Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security ASIACCS '10, Beijing, China, **(2010)** April 13-16.
[27] L. Zu, Z. Liu and J. Li, "New Ciphertext-Policy Attribute-Based Encryption with Efficient Revocation", in Proceedings of IEEE International Conference on Computer and Information Technology (CIT), Xi'an, China, **(2014)** September 11-13.

Authors

Nyamsuren Vaanchig, she is a Ph.D. candidate at School of Information and Software Engineering, University of Electronic Science and Technology of China (UESTC). She received the B.S. degree from the Mongolian State University of Education (MSUE), Ulaanbaatar, in 2004, and the M.S. degree from the same university, in 2007, both in information technology. Her research interests include information security, cryptography, and cloud computing.

Wei Chen, he is an associate professor at UESTC. He received his B.S. degree from the UESTC in 2001 and his M.S. degree from UESTC, in 2003, both in Computer Science and Engineering. He received his Ph.D. degree from UESTC, in 2010, in Information and Communication Engineering. His research interests include information security and software assurance.

Zhiguang Qin, he a professor and the dean of School of Information and Software Engineering, UESTC. He received his MS (1989) and Ph.D. (1996) degree separately at Department of Mathematics from Hunan Xiangtan University and Department of Computer Science from UESTC. He has engaged in distributed computing, information security, and electronic commerce research.