Hindawi
Mathematical Problems in Engineering
Volume 2017, Article ID 4070616, 12 pages
https://doi.org/10.1155/2017/4070616

Research Article

Research on Ciphertext-Policy Attribute-Based Encryption with Attribute Level User Revocation in Cloud Storage

Guangbo Wang and Jianhua Wang

Zhengzhou Information Science and Technology Institute, Zhengzhou, Henan 450004, China

Correspondence should be addressed to Guangbo Wang; 691759571@qq.com

Received 17 February 2017; Revised 1 April 2017; Accepted 5 April 2017; Published 23 May 2017

Academic Editor: Liu Yuhong

Attribute-based encryption (ABE) scheme is more and more widely used in the cloud storage, which can achieve fine-grained access control. However, it is an important challenge to solve dynamic user and attribute revocation in the original scheme. In order to solve this problem, this paper proposes a ciphertext-policy ABE (CP-ABE) scheme which can achieve attribute level user attribution. In this scheme, if some attribute is revoked, then the ciphertext corresponding to this attribute will be updated so that only the individuals whose attributes meet the access control policy and have not been revoked will be able to carry out the key updating and decrypt the ciphertext successfully. This scheme is proved selective-structure secure based on the q-Parallel Bilinear Diffie-Hellman Exponent (BDHE) assumption in the standard model. Finally, the performance analysis and experimental verification have been carried out in this paper, and the experimental results show that, compared with the existing revocation schemes, although our scheme increases the computational load of storage service provider (CSP) in order to achieve the attribute revocation, it does not need the participation of attribute authority (AA), which reduces the computational load of AA. Moreover, the user does not need any additional parameters to achieve the attribute revocation except for the private key, thus saving the storage space greatly.

1. Introduction

With the advent of big data era, there is an increasing number of user data. In order to achieve the sharing of data and reduce the cost at the same time, using the third party, namely, cloud storage provider (CSP), will be an excellent priority. The cloud storage, which emerged as the extension and development of cloud computing, achieves the function that the users can access the data conveniently at any time and at any place by any networking equipment; therefore, it has been more and more extensively used. However, the users' data are stored in the CSP and got rid of the users' actual control; therefore, how to guarantee the users privacy and data security as much as possible without reducing the quality of service has become a key problem of secure cloud storage.

Sahai and Waters in 2005 proposed the notation of attribute-based encryption (ABE) [1] in which the ciphertext and key are, respectively, associated with a series of attributes, and an access structure is specified to define the attribute set that can be used to decrypt the ciphertext successfully. ABE can achieve fine-grained access control by using the flexible access structure, so it has been widely used in the cloud storage. The initial ABE schemes can only achieve the threshold operations so that the policy expression is not rich enough. To solve this problem, some scholars have proposed the ciphertext-policy ABE (CP-ABE) mechanism [2–4] and key-policy ABE (KP-ABE) mechanism [5, 6], which can realize rich attribute operations so as to support flexible access control policy.

However, the application of ABE in cloud storage also brings serious security challenges. There are a large number of users in the cloud storage environment, and different users may share the same attribute in the application of ABE. Therefore, if some attribute of a user is revoked, how to recall the user's corresponding access permissions without affecting the normal access of other legitimate users and posing a large load on the system has become an urgent problem to

be solved. Therefore, this paper mainly pursues the relative research on this issue.

Recently, individuals pay more and more attention to the problem of user revocation in the practical application of ABE. Ostrovsky et al. proposed an ABE scheme with system level user revocation [7]. In this scheme, the revocation is carried out by implementing the "NOT" operation on "AND" gates; however, the efficiency is rather low.

Subsequently, Staddon et al. proposed a KP-ABE scheme [8] which can achieve the revocation of users; however, this scheme is limited to be used if and only if the number of attributes associated with ciphertext is just half of the whole attributes in the system; therefore, the limit is too high which impedes its actual application. Liang et al. proposed a CP-ABE scheme [9] which achieved the revocation by using a binary tree. In this scheme, an attribute authority is responsible for generating the updating key for implementing the revocation; however, the efficiency is also very low. Moreover, it increases the computation and communication burden on the attribute authority greatly which may become the bottleneck. In addition, all the above schemes can only achieve the system level user revocation; namely, once some attribute of a user is revoked, he will lose not only the access permission corresponding to the revoked attribute but also the access permissions corresponding to the other legitimate attributes.

In the aspect of attribute revocation, individuals in the literatures [10–12] strove to achieve the revocation by setting the validity period for each attribute. This method is called coarse-grained revocation because it cannot realize the timely revocation. To solve this problem, Hur and Noh proposed a novel CP-ABE scheme in the literature [13] to realize the revocation by using a key encryption key tree, which can also achieve attribute level user revocation; namely, the revocation to some attribute of a user cannot affect the normal access of other legitimate attributes. In this scheme, if an attribute is revoked, then the CSP will generate a new key encryption key and reencrypts the ciphertext. However, each user needs to store $\log(n_u + 1)$ key encryption keys additionally, where n_u denotes the number of all the users in this scheme. Moreover, the scheme is proved to be secure in the generic group model which possesses heuristic security rather than provable security; therefore, some schemes proved secure in the generic group model are found to be unsafe in practical application. Subsequently, Yang et al. proposed a CP-ABE scheme [14] in the environment of cloud storage. In this scheme, the attribute authority generates two corresponding public parameters for each attribute, and once the revocation is implemented, the attribute authority needs to update the public parameters for the revoked attribute and the secret key for the user, which increases not only the computation load on the attribute authority but also the communication load between the attribute authority and the user.

In this paper, we propose a CP-ABE scheme that combines proxy reencryption methods to achieve the revocation. In this scheme, we achieve the revocation with the help of CSP, which offloads most of revocation operations for the attribute authority that has limited resources. If some attribute is revoked, then the ciphertext corresponding to this attribute will be updated by the CSP so that only the users whose attributes meet the access control policy and have not been revoked will be able to carry out the key updating and decrypt the ciphertext successfully. Additionally, in this scheme, we achieve the fine-grained attribute level user revocation; namely, the revocation to an attribute of some user cannot affect the normal access of this user's other legitimate attributes. Finally, we carry out the performance analysis and experimental verification to demonstrate the characteristics, which shows that, compared with the existing revocation schemes, although our scheme increases the computational load of CSP in order to achieve the attribute revocation, it does not need the participation of AA. Moreover, the user does not need any additional parameters to achieve the attribute revocation except of the private key, thus saving the storage space greatly.

2. Preliminaries

Before proposing the concrete scheme in this paper, we first introduce the related technologies that will be used including bilinear group, linear secret-sharing scheme (LSSS), and deterministic q-Parallel Bilinear Diffie-Hellman Exponent (BDHE) assumption.

2.1. Bilinear Map. In this part, we will briefly take a view to several facts related to the bilinear group as follows.

Definition 1 (bilinear map). The bilinear group has been widely used in various cryptographic systems after it was proposed for the first time. Let ψ be a group parameters generation algorithm which takes as input the security parameter λ and outputs the group parameters $(p, \mathbb{G}, \mathbb{G}_T, e)$. In these group parameters, p denotes a big prime whose size is determined by the security parameter λ, $\mathbb{G}$ and $\mathbb{G}_T$ are two multiplicative cyclic groups with order p, and $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is a bilinear map satisfying the following properties:

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

(2) Nondegeneracy: $\exists g \in \mathbb{G}$ satisfying that $e(g, g)$ has order p in $\mathbb{G}_T$.

(3) Computability: there exists an efficient algorithm to compute the bilinear pairing.

2.2. Linear Secret-Sharing Scheme

Definition 2 (linear secret-sharing scheme (LSSS) [15]). A secret-sharing scheme $\prod$ over a set of parties $\mathcal{P}$ is a LSSS (over $\mathbb{Z}_p$) if it satisfies the following properties:

(1) The secret share of each party constitutes a vector over $\mathbb{Z}_p$.

(2) For each secret-sharing scheme $\prod$, there exists a share-generation matrix $\mathbf{M}(l \times n)$ where, for each row $\mathbf{M}_i$ of the matrix $\mathbf{M}$, we define a function ρ: $\{1, \ldots, l\} \to \mathcal{P}$ that maps it to the corresponding party $\rho(i)$. Considering a vector $\vec{v} = (s, r_2, \ldots, r_n)$, where $s \in \mathbb{Z}_p$ is the sharing secret and parameters $r_2, \ldots, r_n \in \mathbb{Z}_p$ are chosen randomly to conceal the secret, then $\mathbf{M}\vec{v}$ is a vector that is composed of l shares of

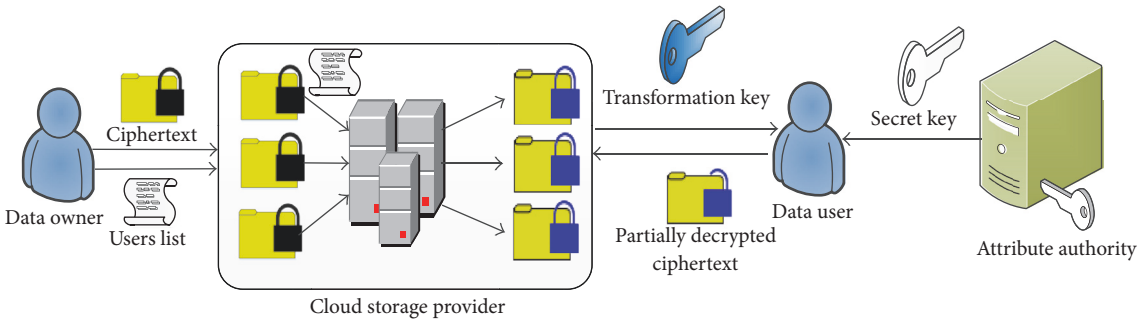

Figure 1: System model.

the secret s. Moreover, $\lambda_i = (\mathbf{M}\vec{v})_i$ denotes the secret share possessed by the party $\rho(i)$.

Suppose $\prod$ is a LSSS for the access structure $(\mathbf{M}, \rho)$ and S denotes any authorized set for $(\mathbf{M}, \rho)$. We define the set $\mathscr{D} \subset \{1, 2, \ldots, l\}$ as $\mathscr{D} = \{i: \rho(i) \in S\}$; then, the constants $\{w_i \in \mathbb{Z}_p\}_{i\in\mathscr{D}}$ can be computed in polynomial time such that if $\{\lambda_i\}$ are valid shares of any secret s according to $\prod$, then we have $\sum_{i\in\mathscr{D}} w_i\lambda_i = s$.

2.3. Decisional q-Parallel Bilinear Diffie-Hellman Exponent Assumption

Definition 3 (q-parallel BDHE assumption [16]). Let $\mathbb{G}$ denote the bilinear group with prime order p, the parameters $a, s, b_1, \ldots, b_q$ are chosen randomly in $\mathbb{Z}_p$, and g is a generator of $\mathbb{G}$. Then, the decisional q-Parallel BDHE assumption is that if there is an attacker $\mathscr{A}$ who is given the parameters

$$\begin{aligned}
\vec{y} &= g, g^s, g^a, \ldots, g^{a^q}, g^{a^{q+2}}, \ldots, g^{a^{2q}},\\
\vec{y} &= g^{sb_j}, g^{a/b_j}, \ldots, g^{a^q/b_j}, \ldots, g^{a^{2q}/b_j}, \quad \forall_{1\le j\le q},\\
\vec{y} &= g^{asb_k/b_j}, \ldots, g^{a^q sb_k/b_j} \quad \forall_{1\le k,j\le q},
\end{aligned} \tag{1}$$

then, it is hard for $\mathscr{A}$ to distinguish $e(g, g)^{a^{q+1}s}$ from a random element in $\mathbb{G}_T$. In addition, a polynomial time algorithm $\mathscr{B}$ will use the output of $\mathscr{A}$ to make a guess, and we define the advantage of $\mathscr{B}$ to solve the q-Parallel BDHE assumption in $\mathbb{G}$ and $\mathbb{G}_T$ as

$$\left|\Pr\left[\mathscr{B}\left(\vec{y}, e\left(g, g\right)^{a^{q+1}s}\right) = 0\right] - \Pr\left[\mathscr{B}\left(\vec{y}, R\right) = 0\right]\right|. \tag{2}$$

If there is no polynomial time algorithm to solve the q-Parallel BDHE assumption with a nonnegligible advantage, then we can say that the assumption holds in $\mathbb{G}$ and $\mathbb{G}_T$.

3. Attribute-Based Encryption

In this part, we will first give the system model for our proposed CP-ABE scheme with attribute level user revocation, and then we give a selectively secure model in terms of the ciphertext indistinguishability under a chosen plaintext attack (IND-CPA) [17] which is defined between a polynomial time attacker $\mathscr{A}$ and challenger $\mathscr{B}$. Finally, we will give the detailed construction.

3.1. System Model.

The concrete system model of our proposed CP-ABE scheme is shown as in Figure 1, which mainly consists of four entities as follows.

(1) Attribute Authority (AA). It is responsible for implementing the system setup algorithm to generate the system parameters and implementing the key generating algorithm to generate the secret key for the data user.

(2) Data Owner (DO). He is responsible for implementing the data encryption algorithm on the plaintext data and sends the generated ciphertext to the CSP. If the DO decides that some attribute needs to be revoked, he will first designate the responding revoked users list and then send the list to the CSP.

(3) Data User (DU). He is responsible for implementing the decryption algorithm. If the DU wants to access the data in the CSP, he will first send his transformation key to the CSP for partial decryption. Once the DU receives the partially decrypted ciphertext, he will use his secret key to implement the final decryption.

(4) Cloud Storage Provider (CSP). He is responsible for implementing the data reencryption algorithm to achieve the ciphertext updating and implementing the partial decryption algorithm for the DU. Here, we assume that the CSP is curious but honest; namely, he will honestly execute the tasks assigned by other legitimate entities in the system; however, he has the incentive to learn the contents of encrypted data as much as possible.

3.2. Selectively Secure Model.

This security model mainly draws lessons from the technique proposed by Tu et al. in the literature [18]. In this model, the attacker $\mathscr{A}$ firstly needs to submit a challenge access structure and a revocation list, and as a response he will obtain the corresponding public key parameters. Subsequently, $\mathscr{A}$ begins to make a series of secret key queries and ciphertext reencryption queries. In the challenge phase, $\mathscr{A}$ will give two messages with the equal length, and then the challenger $\mathscr{B}$ chooses to encrypt one of these two messages based on the random sampling. Next, $\mathscr{A}$ continues to make the secret key query and ciphertext reencryption query and finally outputs a random guess. If the guess is correct, then we can say $\mathscr{A}$ wins the game. The specific definition of this security model is given as follows.

Init. The attacker $\mathcal{A}$ initially chooses the challenge access control structure $\mathbb{A}^*$ and the revocation users list RL_{x^*} of attribute x^*.

Setup. The challenger $\mathcal{B}$ runs the algorithm *Setup* to obtain the public key PK and the master key MK. Finally, $\mathcal{B}$ gives PK to the attacker $\mathcal{A}$ and keeps MK private to itself.

Query Phase 1. The attacker $\mathcal{A}$ adaptively makes a series of secret key queries corresponding to the identity-attribute tuple, namely, $(\mathrm{ID}_1, S_1), \ldots, (\mathrm{ID}_{q_1}, S_{q_1})$; if $\mathrm{ID}_i \notin \mathrm{RL}_{x^*}$, then we set $S_i' = S_i$; otherwise, we set $S_i' = S_i/\{x^*\}$. Note that it must satisfy the restriction that any attributes set S_i' cannot satisfy the challenge access control structure $\mathbb{A}^*$ in this phase. In addition, $\mathcal{A}$ can also make a series of ciphertext reencryption queries associated with the revocation users list of some attribute and the ciphertext.

Challenge. The attacker $\mathcal{A}$ outputs two messages m_0 and m_1 with the equal length to the challenger $\mathcal{B}$. Then, $\mathcal{B}$ chooses a random bit $\beta \in \{0, 1\}$ and encrypts the message m_β under the access control structure $\mathbb{A}^*$ to generate the ciphertext CT^*. Finally, $\mathcal{B}$ sends CT^* to $\mathcal{A}$ as the challenge ciphertext.

Query Phase 2. The attacker $\mathcal{A}$ continues to make a series of secret key queries and ciphertext reencryption queries as in *Query Phase 1* with the same restriction.

Guess. The attacker $\mathcal{A}$ outputs its guess β' for β, and if $\beta' = \beta$, then $\mathcal{A}$ wins the game. In addition, the advantage of $\mathcal{A}$ in this game is defined as $\mathrm{Adv}_{\mathcal{A}} = |\Pr[\beta' = \beta] - 1/2|$.

If there is no polynomial time algorithm to break the security model above with a nonnegligible advantage, then we can say that our proposed CP-ABE scheme with attribute level user revocation is secure.

3.3. Construction. In this part, we will give the concrete construction of our proposed CP-ABE scheme. In our scheme, the attribute authority will first generate the system parameters that will be used in the subsequent algorithms. If the data owner DO wants to store his data on the CSP, he will first encrypt the data with some access control policy to generate the corresponding ciphertext, then he will send the ciphertext to the CSP. Once the DO decides that an attribute of some users list needs to be revoked, he will send the users list to the CSP. Then, the CSP will implement the reencryption on the ciphertext so that only the users whose attributes meet the access control policy associated with the ciphertext and have not been revoked will be able to carry out the key updating and decrypt the ciphertext successfully. In addition, we use the outsourcing decryption to improve the efficiency; namely, the data user (DU) can send his transformation key to the CSP for partial decryption, which makes full use of the computing resources in the CSP. Once the DU gets the partially decrypted ciphertext, he will implement the final decryption faster with less computing resources.

3.3.1. System Setup. In this phase, the attribute authority will generate the corresponding system parameters including the public key and the master key. The public key is accessible by all the entities in the system and the master key is kept private to the attribute authority.

(1) Setup ($\mathrm{setup}(\lambda, U, n) \rightarrow (\mathrm{PK}, \mathrm{MK})$). The setup algorithm takes as input the security parameter λ, the attributes set U, and the number n of users in the system; then, it runs the group parameters generation function ψ to obtain $(\mathbb{G}, \mathbb{G}_T, p, e)$, where p denotes a big prime, $\mathbb{G}$ and $\mathbb{G}_T$ are two cyclic groups with order p, and e is a bilinear map. Let g be the generator of $\mathbb{G}$. Then, the algorithm chooses random exponents $\alpha, \beta \in \mathbb{Z}_p$ and sets $g_i = g^{(\alpha^i)} \in \mathbb{G}$, where $i = 1, 2, \ldots, n, n+2, \ldots, 2n$. Next, it chooses a random exponent $\gamma \in \mathbb{Z}_p$ and sets $v = g^\gamma$. For each attribute $i \in U$, the algorithm chooses random parameters $h_i \in \mathbb{G}$. Finally, the system public key PK is set as $\mathrm{PK} = (p, g, g_1, \ldots, g_n, g_{n+2}, \ldots, g_{2n}, v, e(g, g)^\beta, h_1, \ldots, h_U)$ and the master key MK is set as $\mathrm{MK} = (\alpha, \gamma, g^\beta)$.

3.3.2. Data Encryption. If the data owner wants to store his data $m \in \mathbb{G}_T$ on the CSP, then he will first define an access control policy $(\mathbf{M}, \rho)$ where $\mathbf{M}$ is a $l \times n$ matrix, and the function ρ maps each row $\mathbf{M}_i$ of $\mathbf{M}$ to one corresponding attribute $\rho(i)$ with the restriction that ρ cannot map two distinct rows to one attribute just as in literature [19]. Next, the data encryption algorithm runs $\mathrm{Encrypt}(\mathrm{PK}, m, (\mathbf{M}, \rho))$ to encrypt the data m. Note that the encryption on the data m needs to multiply it with some group element in $\mathbb{G}_T$; therefore, m is also defined as an element in $\mathbb{G}_T$. If we want to encrypt some arbitrary data, then we can define a hash function: $H : \mathbb{Z}_p \rightarrow \mathbb{G}_T$ which maps the arbitrary data to an element in the group $\mathbb{G}_T$.

(2) Encrypt ($\mathrm{encrypt}(\mathrm{PK}, m, (\mathbf{M}, \rho)) \rightarrow \mathrm{CT}$). The encryption algorithm takes as input the public key PK, the plaintext message m, and an access control policy $(\mathbf{M}, \rho)$; then, it chooses random parameters $s, v_2, \ldots, v_n \in \mathbb{Z}_p$ and defines the vector $\mathbf{v} = (s, v_2, \ldots, v_n)$. For each row $\mathbf{M}_i$ of $\mathbf{M}$, the algorithm computes the inner product $\lambda_i = \mathbf{M}_i \cdot \mathbf{v}$, and then it chooses a random exponent $r_i \in \mathbb{Z}_p$ and outputs the ciphertext as follows:

$$\begin{aligned} \mathrm{CT} = \Big(& (\mathbf{M}, \rho),\ C = m \cdot e(g, g)^{\beta s},\ C_0 \\ & = g^s,\ \left\{ C_{i,1} = g_1^{\lambda_i} h_{\rho(i)}^{-r_i},\ C_{i,2} = g^{r_i} \right\}_{i=1}^{l} \Big). \end{aligned} \tag{3}$$

3.3.3. Data Reencryption. If the DO decides that the attribute x of users list RL_x needs to be revoked, then he will send (x, RL_x) to the CSP. Once the CSP receives (x, RL_x), he will use the broadcast encryption to update the ciphertext for the purpose of revoking the access permission corresponding to attribute x without affecting the normal access of other legitimate attributes for the users in RL_x.

(3) Re-Encrypt ($\text{Re-encrypt}(\mathrm{PK}, \mathrm{CT}, \mathrm{RL}_x) \rightarrow \mathrm{CT}''$). The reencryption algorithm takes as input the public key PK, the ciphertext $\mathrm{CT} = (C, C_0, \{C_{i,1}, C_{i,2}\}_{i=1}^{l})$, and the revocation

users list RL_x, and then it chooses a random exponent $v_x \in \mathbb{Z}_p^*$ and outputs the reencrypted ciphertext as follows:

$$\begin{aligned}\mathrm{CT}' = \Big(& (\mathbf{A}, \rho),\ C' = C,\ C_0' = C_0,\ \rho(i) \neq x\colon C_{i,1}' \\ & = C_{i,1},\ C_{i,2}' = C_{i,2},\ \rho(i) = x\colon C_{x,1}' = C_{x,1},\ C_{x,2}' \\ & = \left(C_{x,2}\right)^{1/v_x} = \left(g^{r_x}\right)^{1/v_x} \Big).\end{aligned} \tag{4}$$

Next, the algorithm chooses random parameters $\widetilde{s}, \widetilde{v}_2, \ldots, \widetilde{v}_n \in \mathbb{Z}_p$ and defines the vector $\widetilde{\mathbf{v}} = (\widetilde{s}, \widetilde{v}_2, \ldots, \widetilde{v}_n)$. Note that the reencryption algorithm will use the same access control policy $(\mathbf{M}, \rho)$ as in the *Encrypt* algorithm. For each row $\mathbf{M}_i$ of the matrix $\mathbf{M}$, it computes the inner product $\widetilde{\lambda}_i = \mathbf{M}_i \cdot \widetilde{\mathbf{v}}$ and chooses a random exponent $\widetilde{r}_i \in \mathbb{Z}_p$. Then, the algorithm defines a broadcast users set $N = n \setminus \{\mathrm{RL}_x\}$ and outputs the ciphertext header generated by encrypting the exponent v_x as follows:

$$\begin{aligned}\mathrm{Hdr}_x = \Bigg(& \mathrm{RL}_x,\ \widetilde{C} = v_x \cdot e\left(g_n, g_1\right)^{\widetilde{s}},\ \widetilde{C}_0 = g^{\widetilde{s}},\ \widetilde{C}_1 \\ & = \left(v \left(\prod_{j \in N} g_{n+1-j} \right)^{-1} \right)^{\widetilde{s}}, \\ & \left\{ \widetilde{C}_{i,1} = \left(g_1\right)^{\widetilde{\lambda}_i} h_{\rho(i)}^{-\widetilde{r}_i},\ \widetilde{C}_{i,2} = g^{\widetilde{r}_i} \right\}_{i=1}^{l} \Bigg).\end{aligned} \tag{5}$$

Finally, it returns the ciphertext as $\mathrm{CT}'' = (\mathrm{CT}', \mathrm{Hdr}_x)$.

3.3.4. Key Generation. In order to improve the decryption efficiency, we outsource the decryption of ciphertext to the CSP that has plenty of computing resources. The concrete key generation algorithm is given as follows.

(4) KeyGen ($\mathrm{keygen}_{\mathrm{out}}(\mathrm{PK}, \mathrm{MK}, \mathrm{ID}, \mathrm{S}) \rightarrow \mathrm{SK}$). The key generation algorithm takes as input the public key PK, the master key MK, a user's identity ID, and the attributes set S, and then it chooses a random exponent $r' \in \mathbb{Z}_p$ and generates the corresponding key $\mathrm{SK}' = (K', \widetilde{K}', L', \{K_i'\}_{i \in \mathrm{S}})$, where

$$\begin{aligned} K' &= g^{\alpha^{\mathrm{ID}} \gamma} g^{\alpha r'}, \\ \widetilde{K}' &= g^{\beta} g^{\alpha r'}, \\ L' &= g^{r'}, \\ \left\{ K_i' = h_i^{r'} \right\}_{i \in S}. & \end{aligned} \tag{6}$$

Next, the algorithm continues to choose a random exponent $z \in \mathbb{Z}_p^*$ and computes

$$\begin{aligned} K &= \left(K'\right)^{1/z} = \left(g^{\alpha^{\mathrm{ID}} \gamma}\right)^{1/z} \left(g^{\alpha r'}\right)^{1/z}, \\ \widetilde{K} &= \widetilde{K}'^{1/z} = \left(g^{\beta}\right)^{1/z} \left(g^{\alpha r'}\right)^{1/z}, \\ L &= \left(L'\right)^{1/z} = \left(g^{r'}\right)^{1/z}, \\ \left\{ K_i = \left(K_i'\right)^{1/z} = \left(h_i^{r'}\right)^{1/z} \right\}_{i \in S}. & \end{aligned} \tag{7}$$

Let $r = r'/z$; then, we have

$$\begin{aligned} K &= \left(g^{\alpha^{\mathrm{ID}} \gamma}\right)^{1/z} g^{\alpha r}, \\ \widetilde{K} &= \left(g^{\beta}\right)^{1/z} g^{\alpha r}, \\ L &= g^{r}, \\ \{K_i = h_i^{r}\}_{i \in S}. & \end{aligned} \tag{8}$$

Finally, we set the outsourced transformation key as $\mathrm{TK} = (K, \widetilde{K}, L, \{K_i = h_i^r\}_{i \in S})$ and the secret key as $\mathrm{SK} = (z, \mathrm{TK})$.

3.3.5. Partial Decryption. In order to achieve the outsourced decryption, the user needs to send his transformation key TK to the CSP. Note that the transformation key cannot leak any useful information associated with the secret key SK and the plaintext data m. The concrete partial decryption algorithm is given as follows.

(5) Transform ($\mathrm{transform}_{\mathrm{out}}(\mathrm{TK}, \mathrm{CT}'') \rightarrow \mathrm{TCT}$). The transformation algorithm takes as input the transformation key $\mathrm{TK} = (K, \widetilde{K}, L, \{K_i = h_i^r\}_{i \in \mathrm{S}})$ and the ciphertext $\mathrm{CT}'' = (\mathrm{CT}', \mathrm{Hdr}_x)$.

(1) If there is no attribute revoked, namely, $\mathrm{Hdr}_x = \Phi$, then we have the following.

Here, we have $\mathrm{CT}'' = ((\mathbf{M}, \rho), C, C_0, \{C_{i,1}, C_{i,2}\}_{i=1}^{l})$, and if the attributes set S associated with TK satisfies the access control policy $(\mathbf{M}, \rho)$ included in CT'', then the CSP computes the values $\{w_i \in \mathbb{Z}_p\}_{i \in I}$ satisfying $\sum_{i \in I} w_i \mathbf{M}_i = (1, 0, \ldots, 0)$ in polynomial time. Next, it computes

$$\begin{aligned} B &= \prod_{i \in I} e\left(C_{i,1}, L\right)^{w_i} e\left(C_{i,2}, K_{\rho(i)}\right)^{w_i} \\ &= \prod_{i \in I} e\left(g_1^{\lambda_i} h_{\rho(i)}^{-r_i}, g^{r}\right)^{w_i} e\left(g^{r_i}, h_{\rho(i)}^{r}\right)^{w_i} = e(g, g)^{\alpha r s}, \\ D &= e\left(C_0, \widetilde{K}\right) = e\left(g^{s}, g^{\beta/z} g^{\alpha r}\right) \\ &= e(g, g)^{\beta s / z} e(g, g)^{\alpha r s}, \\ E &= \frac{D}{B} = \frac{e(g, g)^{\beta s / z} e(g, g)^{\alpha r s}}{e(g, g)^{\alpha r s}} = e(g, g)^{\beta s / z}. \end{aligned} \tag{9}$$

Once the partial decryption is over, the CSP sends TCT = (C, E) to the corresponding user for the final decryption.

(2) If the attribute x of users list RL_x is revoked, namely, $\mathrm{Hdr}_x \neq \Phi$, then we have the following.

Here, we have $\mathrm{CT}' = ((\mathbf{M}, \rho), C', C_0', \{C_{i,1}', C_{i,2}'\}_{i=1}^{l})$ and $\mathrm{Hdr}_x = (\mathrm{RL}_x, \widetilde{C}, \widetilde{C}_0, \widetilde{C}_1, \{\widetilde{C}_{i,1}, \widetilde{C}_{i,2}\}_{i=1}^{l})$, and if the attributes set S satisfies the access control policy $(\mathbf{M}, \rho)$ and $\mathrm{ID} \notin \mathrm{RL}_x$, then the CSP implements the partial decryption on the ciphertext header Hdr_x. It also computes the values $\{\widetilde{w}_i \in \mathbb{Z}_p\}_{i\in\tilde{I}}$ satisfying $\sum_{i\in\tilde{I}} \widetilde{w}_i \mathbf{M}_i = (1, 0, \ldots, 0)$ and then continues to compute

$$\begin{aligned}
B_x &= \prod_{i\in\tilde{I}} e\left(\widetilde{C}_{i,1}, L\right)^{\widetilde{w}_i} e\left(\widetilde{C}_{i,2}, K_{\rho(i)}\right)^{\widetilde{w}_i} \\
&= \prod_{i\in\tilde{I}} e\left(g_1^{\tilde{\lambda}_i} h_{\rho(i)}^{-\tilde{r}_i}, g^r\right)^{\widetilde{w}_i} e\left(g^{-\tilde{r}_i}, h_{\rho(i)}^r\right)^{\widetilde{w}_i} \\
&= e(g,g)^{\alpha r s'}, \\
D_x &= e\left(\widetilde{C}_0, K\right) = e(g,g)^{\alpha^{\mathrm{ID}} \gamma \tilde{s}/z} e(g,g)^{\alpha r \tilde{s}}, \\
E_x &= \frac{D_x}{B_x} = e(g,g)^{\alpha^{\mathrm{ID}} \gamma \tilde{s}/z}, \\
F_x &= \frac{e\left(g_{\mathrm{ID}}, \widetilde{C}_1\right)}{e\left(\prod_{\substack{j\in N \\ j\neq \mathrm{ID}}} g_{n+1-j+\mathrm{ID}}, \widetilde{C}_0\right)^{-1}} \\
&= \frac{e\left(g_{\mathrm{ID}}, \left(v\left(\prod_{j\in N} g_{n+1-j}\right)^{-1}\right)^{\tilde{s}}\right)}{e\left(\prod_{\substack{j\in N \\ j\neq \mathrm{ID}}} g_{n+1-j+\mathrm{ID}}, g^{\tilde{s}}\right)^{-1}} \\
&= e\left(g_{\mathrm{ID}}, v\right)^{\tilde{s}} \cdot e\left(g_{n+1}, g\right)^{-\tilde{s}}.
\end{aligned} \tag{10}$$

Therefore, the partially decrypted ciphertext header is set as $\mathrm{Hdr}_x' = (\widetilde{C}, E_x, F_x)$.

Next, the CSP implements the partial decryption on the ciphertext CT' as follows:

$$\begin{aligned}
B_i &= e\left(C_{i,1}', L\right) e\left(C_{i,2}', K_{\rho(i)}\right) = e(g,g)^{\alpha r \lambda_i} \quad \rho(i) \neq x, \\
D &= e\left(C_0', \widetilde{K}\right) = e\left(g^s, g^{\beta/z} g^{\alpha r}\right) \\
&= e(g,g)^{\beta s/z} e(g,g)^{\alpha r s} \quad \rho(i) = x.
\end{aligned} \tag{11}$$

Therefore, the partially decrypted ciphertext is set as

$$\begin{aligned}
\mathrm{TCT}' = \Big((\mathbf{M}, \rho),\ C' = m \\
\cdot e(g,g)^{\beta s}, \{B_i\}_{\rho(i)\neq x}, C_{x,1}', C_{x,2}', D\Big).
\end{aligned} \tag{12}$$

Once the partial decryption is over, the CSP sends $\mathrm{TCT} = (\mathrm{TCT}', \mathrm{Hdr}_x')$ to the corresponding user for the final decryption.

3.3.6. Decryption. Once the user gets the partially decrypted ciphertext, he will use his secret key to implement the final decryption for obtaining the plaintext message as follows.

(6) Decrypt ($\mathrm{decrypt}(\mathrm{TCT}, \mathrm{SK}) \to m$). The decryption algorithm takes as input the partially decrypted ciphertext TCT and the user's secret key SK. Then, it decrypts the ciphertext as follows:

(1) If there is no attribute revoked, namely, $\mathrm{TCT} = (C, E)$, then the user computes

$$\frac{C}{E^z} = m \cdot \frac{e(g,g)^{\beta s}}{\left(e(g,g)^{\beta s/z}\right)^z} = m. \tag{13}$$

(2) If the attribute x of users list RL_x is revoked, namely, $\mathrm{TCT} = (\mathrm{TCT}', \mathrm{Hdr}_x')$, then we have the following.

Here, we have $\mathrm{TCT}' = (C', \{B_i\}_{\rho(i)\neq x}, C_{x,1}', C_{x,2}', D)$ and $\mathrm{Hdr}_x' = (\widetilde{C}, E_x, F_x)$, and then the user computes

$$\begin{aligned}
\widetilde{C} \cdot \frac{F_x}{(E_x)^z} &= v_x \cdot e\left(g_n, g_1\right)^{\tilde{s}} \cdot e\left(g_{\mathrm{ID}}, v\right)^{\tilde{s}} \\
&\quad \cdot \frac{e\left(g_{n+1}, g\right)^{-\tilde{s}}}{\left(e(g,g)^{\alpha^{\mathrm{ID}} \gamma \tilde{s}/z}\right)^z} = v_x.
\end{aligned} \tag{14}$$

If the attributes set S satisfies the access control policy $(\mathbf{M}, \rho)$, then the CSP computes the values $\{w_i \in \mathbb{Z}_p\}_{i\in I}$ satisfying $\sum_{i\in I} w_i \mathbf{M}_i = (1, 0, \ldots, 0)$ in polynomial time and continues to compute

$$\begin{aligned}
B_x &= e\left(C_{x,1}', L\right) e\left(C_{x,2}', \left(K_{\rho(x)}\right)^{v_x}\right) \\
&= e\left(g_1^{\lambda_x} h_{\rho(x)}^{-r_x}, g^r\right) \cdot e\left(\left(g^{r_x}\right)^{1/v_x}, \left(h_{\rho(x)}^r\right)^{v_x}\right) \\
&= e\left(g_1^{\lambda_x}, g^r\right) = e(g,g)^{\alpha r \lambda_x}, \\
B &= \prod_{i\in I} (B_i)^{w_i} = \prod_{i\in I} \left(e(g,g)^{\alpha r \lambda_i}\right)^{w_i} = e(g,g)^{\alpha r s}, \\
E &= \frac{D}{B} = \frac{e(g,g)^{\beta s/z} e(g,g)^{\alpha r s}}{e(g,g)^{\alpha r s}} = e(g,g)^{\beta s/z}, \\
\frac{C}{E^z} &= m \cdot \frac{e(g,g)^{\beta s}}{\left(e(g,g)^{\beta s/z}\right)^z} = m.
\end{aligned} \tag{15}$$

3.4. Security Proof

Theorem 4. *If the decisional q-Parallel BDHE assumption holds in $\mathbb{G}$ and $\mathbb{G}_T$, then there exists no polynomial time attacker to break our proposed CP-ABE scheme with attribute level user revocation selectively, where the challenge matrix is $\mathbf{M}^*(l^* \times n^*)$ with $l^*, n^* \leq q$.*

Proof. If there exists an attacker $\mathcal{A}$ who can selectively break our proposed CP-ABE scheme with a nonnegligible advantage $\varepsilon = \mathrm{Adv}_{\mathcal{A}}$, where the challenge matrix is $\mathbf{M}^*(l^* \times n^*)$ with

$l^*, n^* \le q$, then we can construct a challenger $\mathcal{B}$ to break the decisional q-Parallel BDHE assumption successfully. □

Init. The challenger $\mathcal{B}$ takes as input a q-Parallel BDHE challenge $\vec{y}, T$. In addition, the attacker $\mathcal{A}$ gives the challenge access control policy $(\mathbf{M}^*, \rho^*)$ and the revocation users list RL_{x^*} of attribute x^* where the matrix $\mathbf{M}^*$ has n^* columns.

Setup. The challenger $\mathcal{B}$ chooses a random exponent $\beta' \in \mathbb{Z}_p$ and computes $e(g,g)^{\beta} = e(g,g)^{\beta'} \cdot e(g^{\alpha}, g^{\alpha^q})$, where it implicitly sets $\beta = \beta' + \alpha^{q+1}$. In addition, it sets the broadcast users set as

$$\begin{aligned} \widehat{N} &= \mathrm{RL}_{x^*} \cap \{1, 2, \ldots, n\}, \\ N &= \{1, 2, \ldots, n\} \setminus \widehat{N}. \end{aligned} \tag{16}$$

Then, $\mathcal{B}$ selects a random exponent $u \in \mathbb{Z}_p$ and sets $v = g^u \prod_{k \in N} g_{q+1-k}$.

Next, $\mathcal{B}$ sets the group parameters $h_1, h_2, \ldots, h_U$, and for each x $(1 \le x \le U)$, $\mathcal{B}$ selects a random exponent $z_x \in \mathbb{Z}_p$. Let X denote the set of i satisfying $\rho^*(i) = x$; then, h_x is set as

$$h_x = g^{z_x} \prod_{i \in X} g^{a\mathbf{M}^*_{i,1}/b_i} \cdot g^{a^2 \mathbf{M}^*_{i,2}/b_i} \cdots g^{a^{n^*} \mathbf{M}^*_{i,n^*}/b_i}. \tag{17}$$

Note that if $X = \varnothing$, then we have $h_x = g^{z_x}$. In addition, we can say that h_x is distributed randomly because of the randomness of z_x.

Finally, $\mathcal{B}$ sends to $\mathcal{A}$ the public key PK as

$$\mathrm{PK} = \left(g, g_1, \ldots, g_q, g_{q+2}, \ldots, g_{2q}, v, e(g,g)^{\beta}, h_1, \ldots, h_U\right). \tag{18}$$

Query Phase 1. $\mathcal{A}$ makes to $\mathcal{B}$ a series of queries including the key generation query $\mathcal{O}_{\mathrm{kg}}$ and the ciphertext reencryption query $\mathcal{O}_{\mathrm{ree}}$.

(i) $\mathcal{A}$ makes to $\mathcal{B}$ a key generation query $\mathcal{O}_{\mathrm{kg}}$ associated with the identity ID_j and the attributes set S_j; if $\mathrm{ID}_j \notin \mathrm{RL}_{x^*}$, then we set the attributes set $S'_j = S_j$; otherwise, we set $S'_j = S_j \setminus \{x^*\}$. In addition, if S'_j satisfies the challenge access control policy $(\mathbf{M}^*, \rho^*)$, then $\mathcal{B}$ outputs $\perp$; otherwise, it generates the secret key as follows.

$\mathcal{B}$ first computes the vector $\vec{\mathbf{w}} = (w_1, \ldots, w_{n^*}) \in \mathbb{Z}_p^n$, where $w_1 = -1$, and for all $\rho^*(i) \in S'_j$, it satisfies $\mathbf{M}^*_i \vec{\mathbf{w}}^T = 0$. Note that the vector can be found in polynomial time according to the definition of LSSS.

Then, $\mathcal{B}$ chooses a random parameter $t \in \mathbb{Z}_p$ and defines the exponent r as

$$r = t + w_1 \alpha^q + w_2 \alpha^{q-1} + \cdots + w_{n^*} \alpha. \tag{19}$$

Next, $\mathcal{B}$ computes the key component L' as

$$L' = g^t \cdot \prod_{i=1,\ldots,n^*} \left(g^{\alpha^{q+1-i}}\right)^{w_i} = g^r. \tag{20}$$

According to the definition of r and $w_1 = -1$, we know that $g^{\alpha r}$ includes the item $g^{-\alpha^{q+1}}$. Although $g^{-\alpha^{q+1}}$ is not given in the assumption, it can be canceled by multiplying $g^{\alpha r}$ with $g^{\beta} = g^{\beta'} g^{\alpha^{q+1}}$, because we implicitly set $\beta = \beta' + \alpha^{q+1}$ when generating the key component $\widetilde{K}'$. In detail, it is constructed as follows:

$$\begin{aligned} \widetilde{K}' &= g^{\beta'} g^{\alpha^{q+1}} g^{\alpha t} g^{-\alpha^{q+1}} \prod_{i=2,\ldots,n^*} \left(g^{\alpha^{q+2-i}}\right)^{w_i} \\ &= g^{\beta'} g^{\alpha t} \prod_{i=2,\ldots,n^*} \left(g^{\alpha^{q+2-i}}\right)^{w_i}. \end{aligned} \tag{21}$$

Then, $\mathcal{B}$ will compute the key component K'_i, $\forall i \in S'_j$. For each attribute $i \in S'_j$, if there exists no row k satisfying $\rho^*(k) = i$, then we set $K'_i = (L')^{z_i}$; otherwise, let X denote the set of all the rows k satisfying $\rho^*(k) = i$, and then we set K'_i as

$$\begin{aligned} K'_i = \left(L'\right)^{z_i} \prod_{i \in X} \prod_{j=1,\ldots,n^*} \Bigg(& g^{(\alpha^j/b_i)t} \\ & \cdot \prod_{\substack{k=1,\ldots,n^*, \\ k \ne j}} \left(g^{(\alpha^{q+1+j-k}/b_i) w_k}\right) \Bigg)^{\mathbf{M}^*_{i,j}}. \end{aligned} \tag{22}$$

Next, $\mathcal{B}$ will set the key component K' for the user $\mathrm{ID}_j \notin \mathrm{RL}_{x^*}$. Similarly, $g^{\alpha r}$ includes the item $g^{-\alpha^{q+1}}$ that is not given in the assumption. However, we set the value v as $v = g^u \prod_{k \in N} g_{q+1-k}$ and we have $g^{\alpha^{\mathrm{ID}_j} \gamma} = (g^u \prod_{k \in \widehat{N}} g_{q+1-k})^{\alpha^{\mathrm{ID}_j}}$. Moreover, because $\mathrm{ID}_j \notin \mathrm{RL}_{x^*}$, namely, $\mathrm{ID}_j \in N$, $g^{\alpha^{\mathrm{ID}} \gamma}$ includes the term $g^{\alpha^{q+1}}$ that can be canceled by the term $g^{-\alpha^{q+1}}$ included in $g^{\alpha r}$:

$$\begin{aligned} K' &= g^{\alpha^{\mathrm{ID}} \gamma} g^{\alpha r} \\ &= \left(g^u \prod_{k \in N} g_{q+1-k}\right)^{\alpha^{\mathrm{ID}}} \cdot g^{\alpha t} g^{-\alpha^{q+1}} \prod_{i=2,\ldots,n^*} \left(g^{\alpha^{q+2-i}}\right)^{w_i} \\ &= \left(g^{\alpha^{\mathrm{ID}}}\right)^u \left(\prod_{k \in N \setminus \{\mathrm{ID}_j\}} g_{q+1-k+\mathrm{ID}_j}\right) \cdot g_{q+1-\mathrm{ID}_j+\mathrm{ID}_j} \\ &\quad \cdot g^{\alpha t} g^{-\alpha^{q+1}} \prod_{i=2,\ldots,n^*} \left(g^{\alpha^{q+2-i}}\right)^{w_i} \\ &= \left(g^{\alpha^{\mathrm{ID}}}\right)^u \left(\prod_{k \in N \setminus \{\mathrm{ID}_j\}} g_{q+1-k+\mathrm{ID}_j}\right) \\ &\quad \cdot g^{\alpha t} \prod_{i=2,\ldots,n^*} \left(g^{\alpha^{q+2-i}}\right)^{w_i}. \end{aligned} \tag{23}$$

Once the key components are all generated, the challenger $\mathscr{B}$ will select a random exponent $z \in \mathbb{Z}_p^*$ and set the outsourced transformation key TK as

$$\begin{aligned}\mathrm{TK} = \Big(& K = \left(K'\right)^{1/z},\ \widetilde{K} = \left(\widetilde{K}'\right)^{1/z},\ L \\ & = \left(L'\right)^{1/z},\ \{K_i\}_{i\in S_j'} = \left\{\left(K_i'\right)^{1/z}\right\}_{i\in S_j'}\Big).\end{aligned} \tag{24}$$

Therefore, the secret key is set as SK = (z, TK). Finally, $\mathscr{B}$ sends the transformation key TK to the attacker $\mathscr{A}$.

(ii) $\mathscr{A}$ makes to $\mathscr{B}$ a ciphertext reencryption query $\mathcal{O}_{\mathrm{ree}}$ associated with the revocation users list RL_x of attribute x and the ciphertext $\mathrm{CT} = (C, C_0, \{C_{i,1}, C_{i,2}\}_{i=1}^{l})$. Then, $\mathscr{B}$ generates the reencrypted ciphertext as follows.

$\mathscr{B}$ first selects a random exponent $v_x \in \mathbb{Z}_p^*$ and computes

$$\begin{aligned}\mathrm{CT}' = \Big\{ & C' = C,\ C_0' = C_0,\ \rho(i) \neq x\colon C_{i,1}' = C_{i,1},\ C_{i,2}' \\ & = C_{i,2},\ \rho\left(i = x\colon C_{i,1}' = C_{i,1},\ C_{i,2}' = \left(C_{i,2}\right)^{1/v_x}\right)\Big\}.\end{aligned} \tag{25}$$

Next, $\mathscr{B}$ selects random parameters $\widetilde{s}, \widetilde{v}_2, \ldots, \widetilde{v}_n \in \mathbb{Z}_p$ and defines the vector $\widetilde{\mathbf{v}} = (\widetilde{s}, \widetilde{v}_2, \ldots, \widetilde{v}_n)$. For each row $\mathbf{M}_i$ of the matrix $\mathbf{M}$, $\mathscr{B}$ computes the inner product $\widetilde{\lambda}_i = \mathbf{M}_i \cdot \widetilde{\mathbf{v}}$. Then, $\mathscr{B}$ selects a random exponent $\widetilde{r}_i \in \mathbb{Z}_p$ and defines the broadcast users set as $N = q \setminus \{\mathrm{RL}_x\}$. Finally, it encrypts the exponent v_x to generate the ciphertext header as follows:

$$\begin{aligned}\mathrm{Hdr}_{x^*} = \Big(& \mathrm{RL}_{x^*},\ \widetilde{C} = v_x \cdot e\left(g_n, g_1\right)^{\widetilde{s}},\ \widetilde{C}_0 = g^{\widetilde{s}},\ \widetilde{C}_1 \\ & = \left(g^u\right)^{\widetilde{s}},\ \left\{\widetilde{C}_{i,1} = \left(g_1\right)^{\widetilde{\lambda}_i} h_{\rho(i)}^{-\widetilde{r}_i},\ \widetilde{C}_{i,2} = g^{\widetilde{r}_i}\right\}_{i=1}^{l}\Big).\end{aligned} \tag{26}$$

Note that $\widetilde{C}_1$ is a correctly distributed ciphertext component which is demonstrated as follows:

$$\begin{aligned}\widetilde{C}_1 = \left(g^u\right)^{\widetilde{s}} &= \left(g^u \prod_{k\in N} g_{q+1-k} \cdot \left(\prod_{j\in N} g_{q+1-k}\right)^{-1}\right)^{\widetilde{s}} \\ &= \left(v\left(\prod_{j\in N} g_{q+1-k}\right)^{-1}\right)^{\widetilde{s}}.\end{aligned} \tag{27}$$

Therefore, the final reencrypted ciphertext is set as $\mathrm{CT}'' = (\mathrm{CT}', \mathrm{Hdr}_x)$.

Challenge. The attacker $\mathscr{A}$ submits to the challenger $\mathscr{B}$ two messages m_0 and m_1 with the equal length. Then, $\mathscr{B}$ selects a random coin $\beta \in \{0, 1\}$ and generates the challenge ciphertext components as

$$\begin{aligned}C^* &= m_\beta \cdot T \cdot e\left(g^s, g^{\beta'}\right), \\ C_0^* &= g^s.\end{aligned} \tag{28}$$

Next, $\mathscr{B}$ selects random parameters $y_2', \ldots, y_{n^*}' \in \mathbb{Z}_p$ and then sets the vector $\vec{\mathbf{v}} = (s, sa + y_2', sa^2 + y_3', \ldots, sa^{n-1} + y_{n^*}') \in \mathbb{Z}_p^{n^*}$ to implicitly share the key s. For $i = 1, 2, \ldots, n^*$, $\mathscr{B}$ defines R_i as the set of all $k \neq i$ satisfying $\rho^*(i) = \rho^*(k)$. Finally, $\mathscr{B}$ selects random exponents $r_1', r_2', \ldots, r_l' \in \mathbb{Z}_p$ and sets the challenge ciphertext components $C_{i,1}^*$ and $C_{i,2}^*$ as follows:

$$\begin{aligned}C_{i,1}^* &= g^{-r_i'} g^{-sb_i}, \\ C_{i,2}^* &= h_{\rho^*(i)}^{r'} \left(\prod_{j=2,\ldots,n^*} \left(g^\alpha\right)^{\mathbf{M}_{i,j}^* y_j'}\right) \cdot \left(g^{sb_i}\right)^{-z_{\rho^*(i)}} \\ &\quad \cdot \left(\prod_{k\in R_i} \prod_{j=1,\ldots,n^*} \left(g^{\alpha^j \cdot s \cdot (b_i/b_k)}\right)^{\mathbf{M}_{k,j}^*}\right).\end{aligned} \tag{29}$$

Query Phase 2. $\mathscr{A}$ continues to make to $\mathscr{B}$ a series of queries including the key generation query $\mathcal{O}_{\mathrm{kg}}$ and the ciphertext reencryption query $\mathcal{O}_{\mathrm{ree}}$ as in *Query Phase 1*.

Guess. The attacker $\mathscr{A}$ outputs its guess β' for β. If $\beta = \beta'$, then $\mathscr{A}$ outputs 0 denoting $T = e(g, g)^{\alpha^{q+1} s}$; otherwise, it outputs 1 denoting T is a random parameter in $\mathbb{G}_T$.

If $T = e(g, g)^{\alpha^{q+1} s}$, then $\mathscr{B}$ plays the proper security game, so we have

$$\Pr\left[\mathscr{B}\left(\vec{y},\ T = e\left(g, g\right)^{\alpha^{q+1} s}\right) = 0\right] = \frac{1}{2} + \mathrm{Adv}_{\mathscr{A}}. \tag{30}$$

Otherwise, T is a random element in $\mathbb{G}_T$; namely, m_β is completely random in the view of $\mathscr{A}$, so we have

$$\Pr\left[\mathscr{B}\left(\vec{y},\ T = R\right) = 0\right] = \frac{1}{2}. \tag{31}$$

4. Analysis

In this part, we will compare our proposed CP-ABE scheme with several existing revocation schemes in terms of functionality, storage cost, communication cost, and computation efficiency. The notations that will be used are described as follows: $|C_1|$ denotes the bit size of an element in $\mathbb{G}$; $|C_T|$ denotes the bit size of an element in $\mathbb{G}_T$; $|C_p|$ denotes the bit size of an element in $\mathbb{Z}_p^*$; $C_{\mathscr{T}}$ denotes the size of access control matrix associated with the ciphertext; $|C_k|$ denotes the bit size of the key encryption key in Hur's scheme [13]; t denotes the number of attributes associated with the ciphertext; k denotes the number of attributes associated with the secret key of a user; n_a denotes the number of all attributes in the system; n_u denotes the number of all users in the system.

4.1. Functionality. The functionality comparison is demonstrated in Table 1, from which we can see that Liang's scheme achieve the system level user revocation; namely, once an attribute of some user is revoked, he will lose all the access permissions in the system, which is impractical in the normal application. However, our scheme, Hur's scheme, and

Table 1: Comparison of functionalities.

Scheme	Access control granularity	Model	Assumption
Liang	System level user revocation	Standard	DBDH
Hur	Attribute level user revocation	Generic group	—
Yang	Attribute level user revocation	Random oracle	q-Parallel BDHE
Ours	Attribute level user revocation	Standard	q-Parallel BDHE

Yang's scheme achieve the attribute level user revocation; namely, the revocation of some attribute has no effect on the access permissions of other legitimate attributes. In addition, compared with the generic group model of Hur's scheme and the random oracle model of Yang's scheme, only our scheme is provably secure based on q-Parallel BDHE assumption in the standard model, which has stronger security.

4.2. Storage Cost . The storage cost comparison is demonstrated in Table 2. The storage cost of attribute authority (AA) is mainly generated by the master key MK. Our scheme and Hur's scheme have short and constant master key; however, the master key in Liang's scheme grows linearly with the number n_u of all users in the system and in Yang's scheme grows linearly with the number n_a of all attributes in the system. The storage cost of data owner (DO) is mainly generated by the public key PK. Hur's scheme has the shortest public key which is constant. The public key in Yang's scheme grows linearly with the number n_a of all attributes in the system and in Liang's scheme grows linearly with the number n_a of all attributes and the column vector $C_{\mathcal{T}}/t$ of access control matrix with each other as the slope and in our scheme grows linearly with the number n_a of all attributes and the number n_u of all users, however, with constant slope compared with Liang's scheme. The storage cost of cloud service provider (CSP) is mainly generated by the ciphertext and ciphertext header. Liang's scheme only achieves user revocation in which the key updating is implemented by using the method of subset cover and the ciphertext needs not to be updated; therefore, the ciphertext grows linearly with the size $C_{\mathcal{T}}$ of the access control matrix. Yang's scheme updates the key through the interaction between the AA and the data user (DU) and also updates the corresponding ciphertext associated with the revoked attribute; therefore, the ciphertext grows linearly with the number t of attributes associated with the ciphertext. In Hur's scheme, once the DO sends the ciphertext to the CSP, the CSP generates the corresponding ciphertext header for each attribute group. Therefore, the storage cost includes the ciphertext and ciphertext header; moreover, the ciphertext grows linearly with the number t of attributes associated with the ciphertext, and the ciphertext header grows linearly with the number t of attributes and the number n_u of all users in the system with each other as the slope. In our scheme, if some attribute is revoked, then the CSP selects a new exponent to update the ciphertext corresponding to the revoked attribute and then encrypts the exponent to generate the corresponding ciphertext header. Therefore, the storage cost also includes the ciphertext and ciphertext header; moreover, the ciphertext and ciphertext header both grow linearly with the number t of attributes associated with the ciphertext. The storage cost of the DU is mainly generated by the secret key. Our scheme and Yang's scheme have shorter secret key which grows linearly with the number k of attributes associated with the secret key. In Liang's scheme, the secret key is generated by using a binary tree; therefore, the size of secret key is associated with the number k of attributes, the column vector $C_{\mathcal{T}}/t$ of access control matrix, and the number n_u of all users in the system. In addition, in Liang's scheme, the key updating is implemented by using the method of subset cover, so the storage cost also includes the updating key that grows linearly with the smallest cover set. In Hur's scheme, every user needs to store a plenty of key encryption keys to decrypt the corresponding exponents for key updating; therefore, the size of secret key not only grows linearly with the number k of attributes but only grows logarithmically with the number n_u of all users in the system.

4.3. Communication Cost. The communication cost comparison is demonstrated in Table 3. The communication cost is mainly generated by the key and the ciphertext. The communication cost between the attribute authority (AA) and the data user (DU) is mainly generated by the secret key of user. In Liang's scheme, for every revocation, the AA needs to generate a new updating key which then is sent to the DU; therefore, it causes $2(n_u - n_m)\log(n_u/(n_u - n_m))|C_1|$ size communication cost additionally. In Yang's scheme, for every revocation, the AA needs to communicate with the DU for updating the key; therefore, it causes $2|C_1|$ size communication cost additionally between the AA and DU. In addition, the communication cost between the AA and data owner (DO) is mainly generated by the public key, and in Yang's scheme, the AA needs to update the public key for every attribute revocation; therefore, it generates $2|C_1|$ size communication cost also. The communication cost between the cloud service provider (CSP) and the DU is generated by the ciphertext, and in Hur's scheme, the CSP needs not only to send the ciphertext but also to generate the key encryption keys, which causes $(\log n_u + 1)|C_k|$ size communication cost; in addition, it also needs to send $((t \cdot n_u)/2)|C_p|$ size ciphertext header. In our proposed CP-ABE scheme, for every revoked attribute, the CSP selects a new exponent to implement the ciphertext updating and then encrypts the exponent to generate the ciphertext header, which causes $(2t + 2)|C_1| + |C_T|$ size communication size additionally. However, because we outsource the decryption to the CSP, the DU needs to send $(k + 3)|C_1|$ size transformation key to the CSP for partial decryption. If there is no attribute revoked, then the CSP generates only two elements in $\mathbb{G}_T$; otherwise, the CSP generates $t + 1$ elements in $\mathbb{G}_T$ and two elements in $\mathbb{G}$ corresponding to the ciphertext and three elements in $\mathbb{G}_T$ corresponding to the ciphertext header. In addition, the communication cost between the CSP and the DO is mainly generated by the ciphertext.

Table 2: Comparison of storage costs.

Entity	Liang	Hur	Yang	Ours
AA	$\lvert C_1\rvert + \left(2^{(\log n_u+1)} + 1\right)\lvert C_p\rvert$	$\lvert C_p\rvert + \lvert C_1\rvert$	$(4 + n_a)\lvert C_p\rvert$	$2\lvert C_p\rvert + \lvert C_1\rvert$
DO	$((C_{\mathcal{T}}/t)\cdot n_a + 6)\lvert C_1\rvert + \lvert C_T\rvert + \lvert C_p\rvert$	$2\lvert C_1\rvert + \lvert C_T\rvert$	$(2n_a + 4)\lvert C_1\rvert + \lvert C_T\rvert$	$(n_a + 2n_u + 1)\lvert C_1\rvert + \lvert C_T\rvert$
CSP	$(C_{\mathcal{T}} + 3)\lvert C_1\rvert + \lvert C_T\rvert$	$(2t + 1)\lvert C_1\rvert + \lvert C_T\rvert + ((t\cdot n_u)/2)\lvert C_p\rvert$	$(3t + 1)\lvert C_1\rvert + \lvert C_T\rvert$	$(4t + 3)\lvert C_1\rvert + 2\lvert C_T\rvert$
DU	$(k + 3 + C_{\mathcal{T}}/t)(\log n_u + 1)\lvert C_1\rvert + 2(n_u - n_m)\log(n_u/(n_u - n_m))\lvert C_1\rvert$	$(2k + 1)\lvert C_1\rvert + (\log n_u + 1)C_k$	$(k + 2)\lvert C_1\rvert$	$(k + 3)\lvert C_1\rvert + \lvert C_p\rvert$

Table 3: Comparison of communication costs.

Entity	Liang	Hur	Yang	Ours
AA & DU	$(k + 3 + C_{\mathcal{T}}/t)(\log n_u + 1)\lvert C_1\rvert + 2(n_u - n_m)\log(n_u/(n_u - n_m))\lvert C_1\rvert$	$(2k + 1)\lvert C_1\rvert$	$(k + 4)\lvert C_1\rvert$	$(k + 3)\lvert C_1\rvert + \lvert C_p\rvert$
AA & DO	$((C_{\mathcal{T}}/t)\cdot n_a + 6)\lvert C_1\rvert + \lvert C_T\rvert + \lvert C_p\rvert$	$2\lvert C_1\rvert + \lvert C_T\rvert$	$(2n_a + 6)\lvert C_1\rvert + \lvert C_T\rvert$	$(n_a + 2n_u + 1)\lvert C_1\rvert + \lvert C_T\rvert$
CSP & DU	$(C_{\mathcal{T}} + 3)\lvert C_1\rvert + \lvert C_T\rvert$	$(2t + 1)\lvert C_1\rvert + \lvert C_T\rvert + ((t\cdot n_u)/2)\lvert C_p\rvert + (\log n_u + 1)\lvert C_k\rvert$	$(3t + 1)\lvert C_1\rvert + \lvert C_T\rvert$	$(k + 3)\lvert C_1\rvert + 2\lvert C_T\rvert$ or $(k + 5)\lvert C_1\rvert + (t + 4)\lvert C_T\rvert$
CSP & DO	$(C_{\mathcal{T}} + 3)\lvert C_1\rvert + \lvert C_T\rvert$	$(2t + 1)\lvert C_1\rvert + \lvert C_T\rvert$	$(3t + 1)\lvert C_1\rvert + \lvert C_T\rvert$	$(2t + 1)\lvert C_1\rvert + \lvert C_T\rvert$

4.4. Computation Efficiency. In order to evaluate the computation efficiency of our proposed CP-ABE scheme with attribute level user revocation, we implement our scheme on a 3.4 GHZ processor PC with 64-bit Ubuntu 14.04 operating system, Intel® Core™ i7-3770CPU and 4 G memory. The public key is selected to provide a 128-bit security level. In addition, the experiment uses a 160-bit elliptic curve group based on the pairing-based cryptography library (PBC-0.5.14) [20] and cpabe-0.11 [21] which selects the supersingular curve $y^2 = x^3 + x$ over 512-bit finite field. The experimental data are obtained by computing the average value for 20 times. In this experiment, the time of PBC library computing a pairing operation is approximately 5.3 ms, and the time of computing an exponent operation in $\mathbb{G}$ and $\mathbb{G}_T$ is approximately 6.2 ms and 0.6 ms, respectively. In addition, the selection time of a random element in $\mathbb{G}$ and $\mathbb{G}_T$ is approximately 14 ms and 1.4 ms, respectively, by using the operation/dev/urandom in Ubuntu 14.04 operating system.

In this paper, we compare our scheme with several related schemes in terms of key generation time, encryption time, decryption time, and reencryption time; moreover, we set $C_{\mathcal{T}}/t = 6$, $n_u = 8$.

From Figure 2, we can see that the key generation time grows linearly with the number of attributes, and our key generation time is slightly higher than that of Yang's scheme; however, it is better than that of Hur's scheme and Liang's scheme. In particular, the key generation time in Liang's scheme is associated with not only the number of attributes but also the column vector $C_{\mathcal{T}}/t$ of access control matrix and the number n_u of all users in the system; therefore, its key generation time is much larger than the other three schemes.

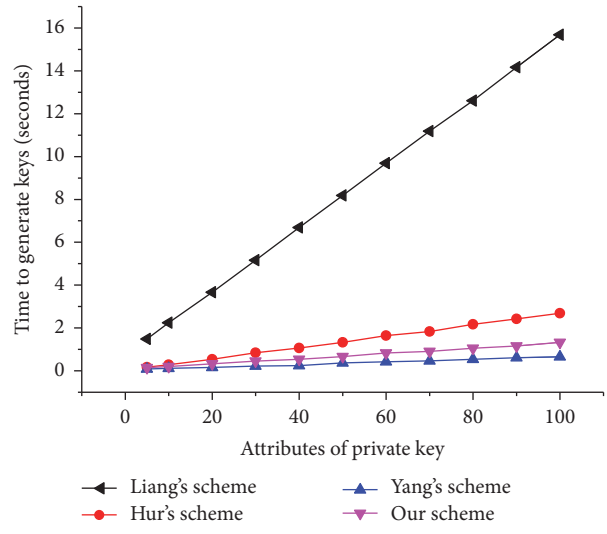

Figure 2: Key generation time.

From Figure 3, we can see that the encryption time grows linearly with the number of attributes associated with the access control policy. Our encryption time is slightly higher than that of Hur's scheme and, however, is better than that of Yang's scheme and Liang's scheme. Note that the encryption in Hur's scheme involves some polynomial operations; however, the running time is very short which is omitted here. The encryption time in Liang's scheme is not only associated with the number of attributes corresponding to the access control policy but also associated with the column vector $C_{\mathcal{T}}/t$ of access control matrix; therefore,

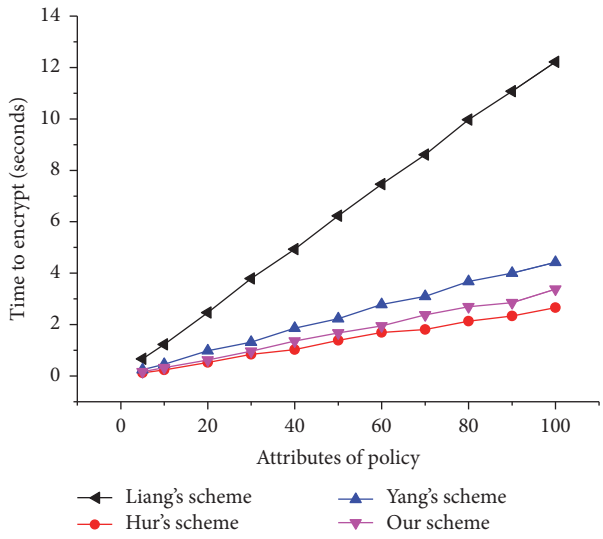

Figure 3: Encryption time.

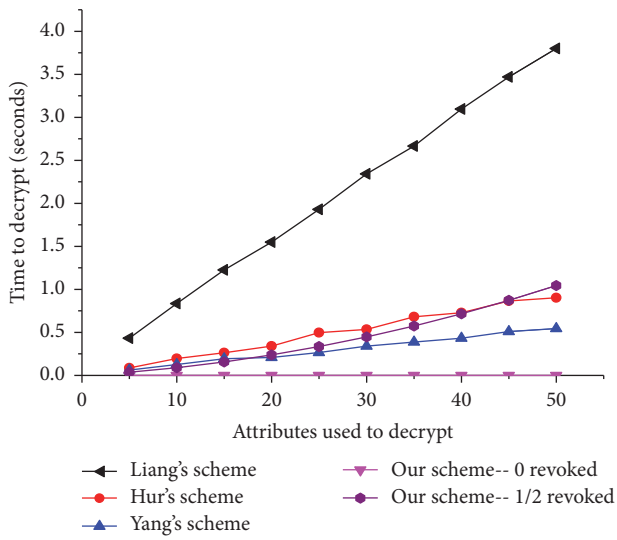

Figure 4: Decryption time.

the encryption time is much larger than the other three schemes.

In the decryption experiment, the computation time is mainly influenced by the number of attributes used in decryption. In order to demonstrate the experimental results better, we suppose that all the intermediate nodes in the binary tree use the (n, n)-threshold gates. In addition, our scheme is demonstrated under two circumstances; namely, no attribute is revoked and 50% attributes are revoked. From Figure 4, we can see that the decryption time in our scheme with 50% attributes revoked, Liang's scheme, Hur's scheme, and Yang's scheme grows linearly with the number of attributes used in decryption. Moreover, our scheme with no attribute revoked uses outsourced decryption, so the user needs only one exponent operation in $\mathbb{G}_T$. In addition, the decryption time of our scheme with 50% attributes revoked is a quadratic function for the attributes used in decryption; however, we also uses outsourced decryption which decreases the decryption time of user greatly. From Figure 4, we can see that when the number of attributes used in decryption locates in a certain range, the decryption time of our scheme with 50% attributes revoked is smaller than the other three schemes, and as the number of attributes used to decrypt increases, the decryption time goes over Yang's scheme and Hur's scheme successively, however, within acceptable range.

In addition, the comparison of reencryption times is shown in Figure 5. If there exists some attribute to be revoked, then the key or the ciphertext should be updated. Yang's scheme and Liang's scheme mainly implement the key updating while Hur's scheme and our scheme mainly implement the ciphertext updating. Therefore, from Figure 5, we can see that the reencryption time in Hur's scheme and our scheme is larger and grows linearly with the number of attributes associated with access control policy. However, all these computations are implemented by the CSP that has a plenty of computing resources. Although the reencryption time in Yang's scheme and Liang's scheme is shorter, it

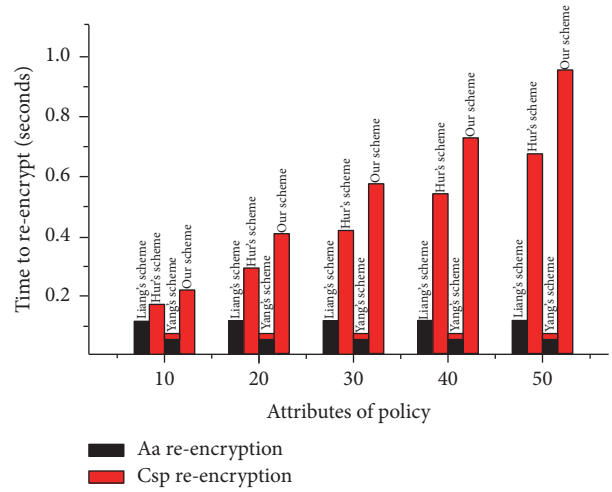

Figure 5: Reencryption time.

requires AA to implement the key updating. As we all know, the computation resources of AA are limited, which may be the bottleneck in the system.

5. Conclusion

In this paper, we propose a CP-ABE scheme which can achieve the attribute level user revocation. In this scheme, if some attribute of a user is revoked, then the ciphertext corresponding to the revoked attribute is updated so that only the user, whose attributes set satisfies the access control policy and has not been revoked, can carry out the key updating to decrypt the ciphertext successfully. The security of our scheme is proved secure based on the q-Parallel BDHE assumption in the standard model. Finally, the performance analysis and experimental verification are carried out, and the experimental results show that although our scheme

increases the computation cost of the CSP in order to achieve the attribute revocation, it does not require the participation of the AA, which decreases the computation cost of the AA. Moreover, the user does not need to store additional parameters to carry out the attribute revocation; thus, it greatly saves the storage space.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

The authors acknowledge the important comments given by the instructors and colleagues. This study acquired support from National Key Research Program of China "Collaborative Precision Position Project" (Grant no. 2016YFB0501900).

References

[1] A. Sahai and B. Waters, "Fuzzy identity-based Encryption," in *Advances in cryptology—EUROCRYPT 2005*, vol. 3494 of *Lecture Notes in Computer Sci.*, pp. 457–473, Springer, Berlin, Germany, 2005.

[2] U. C. Yadav, "Ciphertext-policy attribute-based encryption with hiding access structure," in *Proceedings of the 2015 5th IEEE International Advance Computing Conference, (IACC '15)*, pp. 6–10, India, June 2015.

[3] T. Naruse, M. Mohri, and Y. Shiraishi, "Provably secure attribute-based encryption with attribute revocation and grant function using proxy re-encryption and attribute key for updating," *Human-centric Computing and Information Sciences*, vol. 5, no. 1, pp. 1–13, 2015.

[4] H. Wang, B. Yang, and Y. Wang, "Server aided ciphertext-policy attribute-based encryption," in *proceedings of the IEEE International Conference on Advanced Information Networking Applications Workshops*, pp. 440–444, Gwangju, Korea, 2015.

[5] Q. Li, J. Ma, R. Li, J. Xiong, and X. Liu, "Large universe decentralized key-policy attribute-based encryption," *Security and Communication Networks*, vol. 8, no. 3, pp. 501–509, 2015.

[6] X. Wang, J. Zhang, E. M. Schooler, and M. Ion, "Performance evaluation of Attribute-Based Encryption: toward data privacy in the IoT," in *proceedings of the 2014 1st IEEE International Conference on Communications (ICC '14)*, pp. 725–730, Sydney, Australia, June 2014.

[7] R. Ostrovsky, A. Sahai, and B. Waters, "Attribute-based encryption with non-monotonic access structures," in *Proceedings of the 14th ACM Conference on Computer and Communications Security (CCS '07)*, pp. 195–203, November 2007.

[8] J. Staddon, P. Golle, M. Gagne, and P. Rasmussen, "A content-driven access control system," in *Proceedings of the 7th Symposium on Identity and Trust on the Internet (IDtrust '08)*, pp. 26–35, Gaithersburg, Maryland, USA, March 2008.

[9] X. Liang, R. Lu, and X. Lin, "Ciphertext policy attribute based encryption with efficient revocation," in *Proceedings of the IEEE Symposium on Security Privacy*, vol. 2008, pp. 321–334, 2010.

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

[11] A. Boldyreva, V. Goyal, and V. Kumart, "Identity-based encryption with efficient revocation," in *Proceedings of the 15th ACM conference on Computer and Communications Security (CCS '08)*, pp. 417–426, Alexandria, VA, USA, October 2008.

[12] M. Pirretti, P. Traynor, P. McDaniel, and B. Waters, "Secure attribute-based systems," in *Proceedings of the 13th ACM Conference on Computer and Communications Security (CCS '06)*, pp. 99–112, Alexandria, Va, USA, October-November 2006.

[13] J. Hur and D. K. 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] 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, May 2013.

[15] E. Zavattoni, L. J. Perez, S. Mitsunari et al., "Software implementation of an attribute-based encryption scheme," *IEEE Transactions on Computers*, vol. 64, no. 5, pp. 1429–1441, 2015.

[16] B. Waters, "Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization," *Lecture Notes in Computer Science*, vol. 2008, pp. 321–334, 2011.

[17] L. Cheung and C. Newport, "Provably secure ciphertext policy ABE," in *Proceedings of the 14th ACM Conference on Computer and Communications Security (CCS '07)*, pp. 456–465, NY, USA, November 2007.

[18] S. S. Tu, S. Z. Niu, and H. Li, "A fine-grained access control and revocation scheme on clouds," *Concurrency & Computation Practice & Experience*, vol. 28, no. 6, 2012.

[19] A. Lewko, T. Okamoto, A. Sahai, K. Takashima, and B. Waters, "Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption," in *Advances in cryptology—EUROCRYPT 2010*, vol. 6110 of *Lecture Notes in Comput. Sci.*, pp. 62–91, Springer, Berlin, Germany, 2010.

[20] B. Lynn, "The pairing-based cryptography (PBC) library[OL]," 2006, http://crypto.stanford.edu/pbc.

[21] J. Bethencourt, A. Sahai, and B. Waters, "Advanced crypto software collection: the cpabetoolkit[OL]," 2001, http://acsc.cs.utexas.edu/cpabe.