Large Universe Ciphertext-Policy Attribute-Based Encryption with Attribute Level User Revocation in Cloud Storage

Huijie Lian[1], Qingxian Wang[2], and Guangbo Wang[1]
[1]Zhengzhou Information Science and Technology Institute, Zhengzhou
[2]31008 army, Beijing

Abstract: *Ciphertext-Policy Attribute-Based Encryption (CP-ABE), especially large universe CP-ABE that is not bounded with the attribute set, is getting more and more extensive application in the cloud storage. However, there exists an important challenge in original large universe CP-ABE, namely dynamic user and attribute revocation. In this paper, we propose a large universe CP-ABE with efficient attribute level user revocation, namely the revocation to an attribute of some user cannot influence the common access of other legitimate attributes. To achieve the revocation, we divide the master key into two parts: delegation key and secret key, which are sent to the cloud provider and user separately. Note that, our scheme is proved selectively secure in the standard model under "q-type" assumption. 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 of the private key, thus saving the storage space greatly.*

Keywords: *Ciphertext-policy attribute-based encryption, outsourced decryption, large universe, attribute level user revocation.*

Received February 12, 2017; accepted May 10, 2017
https://doi.org/10.34028/iajit/17/1/13

1. Introduction

Nowadays, the cloud storage as a novel technology of network storage, is enjoying more and more popular application for flexible data storing and sharing on demand. However, there also exist some security concerns, such as data confidentiality and information leakage. Once the users upload their data to the Cloud Storage Provider (CSP), they will have no choice but to trust the CSP and lose the direct control of their data. To solve this problem, users tend to encrypt their sensitive data using some cryptographic scheme before sending them to the CSP.

Sahai and Waters [17] in 2005 proposed the notation of Attribute Based Encryption (ABE). ABE can achieve fine-grained access control by using the flexible access structure, so it has been widely used in the cloud storage. Since then, some scholars have further proposed the Ciphertext-Policy ABE (CP-ABE) mechanism [6, 7, 8] and Key-Policy ABE (KP-ABE) mechanism [1, 10], which can realize rich attribute operations so as to support flexible access control policy. However, all these ABE schemes have a common limitation, namely the system parameters must be chosen at the setup phase, which cannot offer complex flexibility. Lewko and waters [11] first solved this problem by introducing the classification of ABE scheme: small universe and large universe. In the small universe scheme, the size of attribute size is polynomial to the system parameter and must be set at the initial phase. More importantly, the public parameters increase linearly with the size of the universe. In the large universe scheme, the attribute universe can be arbitrarily large and the public parameters can keep constant. Afterwards, Rouselakis and Waters [16] proposed two large universe ABE schemes (one CP-ABE and one KP-ABE) on prime order bilinear groups. However, it does not involve the dynamic user and attribute revocation which is critical to cloud storage environment. Therefore, this paper mainly pursues the relative research on this issue.

Recently, individuals pay more and more attention on the problem of user and attribute revocation in the practical application of ABE. Ostrovsky *et al*. [14] in 2007 proposed the first revocable ABE scheme, however, the efficiency is rather low. Subsequently, Staddon et al. proposed a KP-ABE scheme [18] 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. Liang *et al*. [12] proposed a CP-ABE scheme which achieved the

revocation by using a binary tree, 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.

Note that, 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 this revoked attribute, but also the access permissions corresponding to the other legitimate attributes. In the aspect of attribute level user revocation, individuals in the literatures [2, 4, 15] 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 proposed a novel CP-ABE scheme in the literature [9] to realize the revocation by using a key encryption key tree, however, each user needs to store $\log(n_u+1)$ key encryption keys additionally. Moreover, the scheme is proved to be secure in the generic group model. Subsequently, Yang *et al*. [21] proposed a CP-ABE scheme 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. Note that, all these revocation schemes are only applicable to small universe environment that is not practical.

In this paper, we propose the first large universe CP-ABE scheme that combines proxy re-encryption method to achieve the attribute revocation. In this scheme, we achieve the revocation with the help of CSP, which offloads most of revocation operations for the authority. The keys are divided into two forms: the secret key for user and the delegation key for CSP, and the delegation key is used to re-encrypt the ciphertext. Only the users who are not revoked can can update the secret keys successfully and further decrypt the re-encrypted ciphertext.

2. Preliminaries

Before proposing the concrete scheme in this paper, we first introduce the related technologies that will be used including bilinear maps and q-type assumption.

2.1. Bilinear Map

- *Definition* 1. (Bilinear Map [20]) 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, G, G_T, e). In these group parameters, p denotes a big prime whose size is determined by the security parameter λ, G and G_T are two multiplicative cyclic groups with order p, e: $G\times G\rightarrow G_T$ is a bilinear map satisfying the following properties:

1. *Bilinearity*: $\forall\, u,v\in G,\ a,b\in Z_p$, we have $e(u^a,v^b)=e(u,v)^{ab}$.
2. *Non-degeneracy*: $\exists g\in G$ satisfying that $e(g,g)$ has order p in G_T.
3. *Computability*: There exists an efficient algorithm to compute the bilinear pairing.

2.2. Q-type Assumption

- *Definition* 2. (q-type Assumption) Let G denote the bilinear group with prime order p, and the parameters a, s, $b_1,...,b_q$ are chosen randomly in Zp, g is a generator of G. Then the q-type assumption is that if there is an attacker $\mathcal{A}$ who is given the parameters:

$$\vec{y}=\begin{cases} g^{a_i}, g^{b_j}, g^{sb_j}, g^{a^i b_j}, g^{a^i/b_j^2} & \forall (i,j)\in[q,q] \\ g^{a^i b_j/b_{j'}^2} & \forall (i,j,j')\in[2q,q,q] \ \text{with}\ j\neq j' \\ g^{a^i/b_j} & \forall (i,j)\in[2q,q] \ \text{with}\ i\neq q+1 \\ g^{sa^i b_j/b_{j'}}, g^{sa^i b_j/b_{j'}^2} & \forall (i,j,j')\in[2q,q,q] \ \text{with}\ j\neq j' \end{cases} \quad (1)$$

Then, it is hard for $\mathcal{A}$ to distinguish $e(g,g)^{a^{q+1}s}$ from a random element in G_T. In addition, we define the advantage of $\mathcal{B}$ to solve the q-type assumption in G and G_T as

$$|\Pr[\mathcal{B}(\vec{y},e(g,g)^{a^{q+1}s})=0]-\Pr[\mathcal{B}(\vec{y},R)=0]| \quad (2)$$

3. Large Universe CP-ABE with Attribute Level User Revocation

In this part, we will first give the system model for our proposed CP-ABE scheme with attribute level user revocation, then we give a selectively secure model in terms of the ciphertext Indistinguishability Under a Chosen Plaintext Attack (IND-CPA) which is defined between a polynomial time attacker $\mathcal{A}$ and challenger $\mathcal{B}$. Finally, we will give the detailed construction.

3.1. System Model

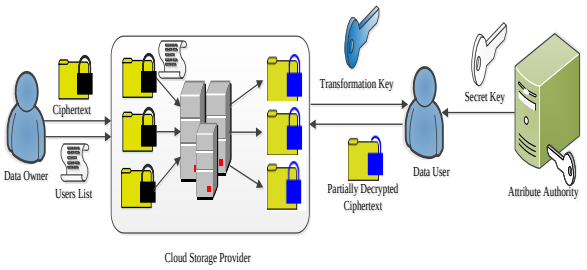

Figure 1. System model.

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)*: It is responsible for implementing the data encryption algorithm on the plaintext data, and sends the generated cipher text 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)*: It is responding 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 partially decryption. Once the DU receives the partially decrypted cipher text, he will use his secret key to implement the final decryption.
4. *Cloud Storage Provider (CSP)*: It is responsible for implementing the data re-encryption 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*. [19] in the literature. The specific definition of this security model is given as follows:

- *Init*: The attacker $\mathcal{A}$ first declares the challenge access structure $\mathbb{A}^*(M^*, \rho^*)$ and attribute revocation list RL_{x^*} which denotes the set of members whose attribute x^* is revoked.
- *Setup*: The challenger $\mathcal{B}$ runs the Setup algorithm by taking as input a security parameter 1^λ and the attribute universe *u*, to generate the public parameters *PK* and master key *MK*. $\mathcal{B}$ begins the interaction with $\mathcal{A}$ by giving the public parameters *PK*.
- *Query Chase* 1: $\mathcal{A}$ adaptively makes a series of secret key query corresponding to the identity-attribute tuple, namely $(ID_1, S_1), \ldots, (ID_{q_1}, S_{q_1})$, if $ID_i \notin RL_{x^*}$, 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 re-encryption query associated with the revocation users list of some attribute and the ciphertext.
- *Challenge*: The attacker $\mathcal{A}$ declares two equal length messages m_0, m_1, and submits them to the challenger $\mathcal{B}$. Then $\mathcal{B}$ randomly chooses a random $\beta \in \{0,1\}$ and encrypts the message m_B using the access structure $\mathbb{A}^*$ with the revocation list RL_{x^*}. Finally, $\mathcal{B}$ gives the challenge ciphertext CT^* to $\mathcal{A}$.
- *Query* 1: The attacker $\mathcal{A}$ continues to make a series of secret key query and ciphertext re-encryption query as in Query 1 with the same restriction.
- *Guess*: $\mathcal{A}$ outputs a guess β' for β.

The advantage of $\mathcal{A}$ in the above game is defined as $Adv_A = |Pr[\beta' = \beta] - 1/2|$.

3.3. Construction

Our proposed large universe CP-ABE scheme with attribute level user revocation based on Rouselakis *et al*. [16] construction is mainly composed of six polynomial time algorithms given as follows:

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.

- *Setup*: $Setup(1^\lambda) \to (PK, MK)$ The setup algorithm takes the security parameter 1^λ as input, then it first runs the group generator ψ to obtain $D = (G, G_T, p, e)$ where G and G_T are two cyclic groups of prime order p, and e is a bilinear map. The attribute universe is $\mathcal{U} = \mathbb{Z}_p$.

Then the algorithm randomly chooses parameters $g, u, h, w, v \in \mathbb{G}$ and $\alpha_1, \alpha_2 \in Z_p$ such that $\alpha_1 + \alpha_2 = \alpha \ mod \ p$. Finally, the public key *PK* is set as

$$PK = (D, g, u, h, w, v, e(g,g)^\alpha)$$

The master key *MK* is set as

$$MK = (\alpha_1, \alpha_2)$$

3.3.2. Key Generation

In order to improve the efficiency by outsourcing the decryption of cipher text, we give the concrete key generation algorithm as follows:

- KeyGen $KeyGen_{out}(PK, MK, S) \rightarrow (SK_1, SK_2)$ The key generation algorithm takes as input the public key PK, the master key MK and a set of attributes $S = \{s_1, s_2, ..., s_k\}$, then it first randomly chooses k+1 exponents $r', r_1', r_2', ..., r_k' \in Z_p$ and generates the corresponding key $SK_1' = (K_0', K_1', \{K_{\sigma,2}', K_{\sigma,3}'\}_{\sigma=1}^{k})$ for the user where

$$K_0' = g^{\alpha_1} w^{r'}, K_1' = g^{r'}, \\ \{K_{\sigma,2}' = g^{r_\sigma'}, K_{\sigma,3}' = (u^{s_\sigma} h)^{r_\sigma'} v^{-r'}\}_{\sigma=1}^{k} \quad (3)$$

Next, this algorithm uses the other part of the master key a_2 to generate the delegation key as $SK_2=g^{a}{}_2$ for the CSP.

Once the user receives the key SK_1', it will choose a random exponent $z \in Z_p^*$ and compute:

$$K_0 = (K_0')^{1/z} = (g^{\alpha_1} w^{r'})^{1/z}, K_1 = (K_1')^{1/z} = (g^{r'})^{1/z} \\ \{K_{\sigma,2} = (K_{\sigma,2}')^{1/z} = (g^{r_\sigma'})^{1/z}, \\ K_{\sigma,3} = (K_{\sigma,3}')^{1/z} = ((u^{s_\sigma} h)^{r_\sigma'} v^{-r'})^{1/z}\}_{\sigma=1}^{k} \quad (4)$$

Let $r = r'/z, r_1 = r_1'/z, r_2 = r_2'/z, ..., r_k = r_k'/z$, then we have:

$$K_0 = g^{\alpha_1/z} w^{r}, K_1 = g^{r}, \\ \{K_{\sigma,2} = g^{r_\sigma}, K_{\sigma,3} = (u^{s_\sigma} h)^{r_\sigma} v^{-r}\}_{\sigma=1}^{k} \quad (5)$$

Therefore, it sets the outsourced transformation key as $TK = (K_0, K_1, \{K_{\sigma,2}, K_{\sigma,3}\}_{\sigma=1}^{k})$ and the secret key as $SK_1=(TK)$.

3.3.3. Data Encryption

If a user wants to store his data m on the CSP, then he first define an access control policy $(\mathcal{M}, \rho)$ where $\mathcal{M}$ is a $l \times n$ matrix, and the function ρ maps each row M_i of M to one corresponding attribute $\rho(i)$ with the restriction that ρ cannot map two distinct rows to one attribute just as in literature [10]. Next, the data encryption algorithm runs $Encrypt(PK, m, (\mathcal{M}, \rho))$ to encrypt the message m.

- *Encrypt*: $Encrypt(PK, m, (\mathcal{M}, \rho)) \rightarrow CT$ The encryption algorithm takes as input the public key PK, the plaintext message m and the access structure $\mathcal{A}$ encoded as an LSSS policy with access matrix $\mathcal{M} \in Z_p^{l \times n}$ and map function $\rho: [l] \rightarrow Z_p$. Then the algorithm chooses random exponents $t_1, t_2, ..., t_l$ and a random vector $\vec{v} = (s, y_2, ..., y_n) \in Z_p^n$, where s is the secret to be shared. Next, for $i = 1$ to l, the algorithm computes $\lambda_i = \mathcal{M}_i \vec{v}$. Finally, the cipher text is published as $CT = (\mathcal{A}, C, C_0, \{C_{i,1}, C_{i,2}, C_{i,3}\}_{i \in [l]})$, where $C = m \cdot e(g,g)^{\alpha s}$, $C_0 = g^s$, and for each attribute $i \in [l]$:

$$C_{i,1} = w^{\lambda_i} v^{t_i}, C_{i,2} = (u^{\rho(i)} h)^{-t_i}, C_{i,3} = g^{t_i} \quad (6)$$

3.3.4. Data Re-Encryption

If the attribute x of users list RL_w is revoked, then we will use the broadcast encryption to update the cipher text 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_w.

- Reencry $Re-encrypt(PK, CT, SK_2, RL_x) \rightarrow (RCT, SK_2')$ The re-encryption algorithm takes as input the public key PK, the cipher text CT, the delegation key SK_2 and the revocation list RL_x of attribute x. In addition, we denote ID_i as the identity of user i. Then the algorithm processes as follows:

1. If there is no attribute revoked, namely $RL_x = \Phi$, then the CSP chooses a random $k \in Z_p$ and re-encrypts the cipher text CT as follows:

$$CT' = (C' = C = m \cdot e(g,g)^{\alpha s}, C_0' = C_0 = g^s, C_1' = g^{s/k} \\ \forall i = 1, 2, ..., l: \\ C_{i,1}' = w^{\lambda_i} v^{t_i} v^k, C_{i,2}' = (u^{\rho(i)} h)^{-t_i} (u^{\rho(i)} h)^{-k}, C_{i,3}' = g^{t_i} g^k) \quad (7)$$

Therefore, the re-encrypted cipher text is set as $RCT = CT'$. In addition, the re-encryption algorithm will generate the updated delegation key as $SK_2' = (g^{\alpha_2})^k$.

2. If there is an attribute x revoked from a user ID_j, namely $RL_x \neq \Phi$, then the CSP will choose a random exponent $v_x \in Z_p$ and encrypt it as the cipher text header CH_x using the broadcast encryption scheme [5] for those users $ID_i, i \neq j$ who possess the revoked attribute and have not been revoked.

Then the CSP also chooses a random $k \in Z_p$ and re-encrypts the cipher text CT to output:

$$CT' = (C' = C = m \cdot e(g,g)^{\alpha s}, C_0' = C_0 = g^s, C_1' = g^{s/k} \\ \forall i = 1, 2, ..., l: \\ C_{i,1}' = w^{\lambda_i} v^{t_i} v^k, C_{i,2}' = (u^{\rho(i)} h)^{-t_i} (u^{\rho(i)} h)^{-k} \\ for\ \rho(i) \neq x: C_{i,3}' = g^{t_i} g^k \\ for\ \rho(i) = x: C_{i,3}' = (g^{t_i} g^k)^{1/v_x}) \quad (8)$$

Therefore, the re-encrypted cipher text is set as $RCT = (CH_x, CT')$. Also the re-encryption algorithm will generate the updated delegation key as $SK_2' = (g^{\alpha_2})^k$.

3.3.5. Partially Decryption

In order to achieve the outsourced decryption, the user needs to send his transformation key TK to the CSP,

and then the CSP decrypts the cipher text partially as follows:

- *Transform*: $Transform_{out}(TK, SK_2', RCT) \to TCT$ The transformation algorithm takes as input the transformation key $TK = (K_0, K_1, \{K_{\sigma,2}, K_{\sigma,3}\}_{\sigma=1}^{k})$, the delegation key SK_2' and the re-encrypted cipher text *RCT*.

  1. If there is no attribute revoked, namely $CH_x = \Phi$.

Here, we have $RCT = (C', C_0', C_1', \{C_{i,1}', C_{i,2}', C_{i,3}'\}_{i=1}^{l})$, and if the attributes set S associated with *TK* satisfies the access control policy (M, ρ) included in *RCT*, then the CSP can compute the values $\{w_i \in Z_p\}_{i \in I}$ satisfying $\sum_{i \in I} w_i M_i = (1, 0, ..., 0)$ in polynomial time. Next, it computes

$$\begin{aligned} B &= \prod_{i \in I} (e(C_{i,1}', K_1) e(C_{i,2}', K_{i,2}) e(C_{i,3}', K_{i,3}))^{w_i} \\ &= \prod_{i \in I} (e(w^{\lambda_i} v^{t_i} v^{k}, g^{r}) e((u^{\rho(i)} h)^{-t_i} (u^{\rho(i)} h)^{-k}, g^{r_i}) \cdot \\ &\quad e(g^{t_i} g^{k}, (u^{s_i} h)^{r_i} v^{-r}))^{w_i} \\ &= e(g, w)^{rs} \\ D &= e(C_0', K_0) = e(g^{s}, g^{\alpha_1 / z} w^{r}) = e(g, g)^{\alpha_1 s / z} e(g, w)^{rs} \\ E &= e(SK_2', C_1') = e((g^{\alpha_2})^{k}, g^{s/k}) = e(g, g)^{\alpha_2 s} \\ F &= D / B = e(g, g)^{\alpha_1 s / z} e(g, w)^{rs} / e(g, w)^{rs} = e(g, g)^{\alpha_1 s / z} \end{aligned} \tag{9}$$

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

  2. If the attribute x of users list RL_x is revoked, namely $CH_x \neq \Phi$.

Here, we have $RCT = (CH_x, CT')$ and $CT' = (C', C_0', C_1', \{C_{i,1}', C_{i,2}', C_{i,3}'\}_{i=1}^{l})$, then the CSP implements the partial decryption on the cip her text CT' as follows:

$\rho(i) \neq x$:

$$B_i = e(C_{i,1}', K_1) e(C_{i,2}', K_{i,2}) e(C_{i,3}', K_{i,3}) = e(g, w)^{r\lambda} \tag{10}$$

$\rho(i) = x$: $C_{i,1}', C_{i,2}', C_{i,3}'$ are kept unchanged.

$$\begin{aligned} D &= e(C_0', K_0) = e(g^{s}, g^{\alpha_1 / z} w^{r}) = e(g, g)^{\alpha_1 s / z} e(g, w)^{rs} \\ E &= e(SK_2', C_1') = e((g^{\alpha_2})^{k}, g^{s/k}) = e(g, g)^{\alpha_2 s} \end{aligned} \tag{11}$$

Therefore, the partially decrypted cipher text is set as:

$$TCT' = (C', \{B_i\}_{\rho(i) \neq x}, \{C_{i,1}', C_{i,2}', C_{i,3}'\}_{\rho(i) = x}, D, E)$$

Once the partial decryption is over, the CSP sends $TCT = (CH_x, TCT')$ to the corresponding user for the final decryption.

3.3.6. Decryption

Once the user gets the partially decrypted cipher text, he will implement the final decryption to obtain the plaintext message as follows:

- *Decrypt*: $Decrypt(TCT, SK_1) \to m$ The decryption algorithm takes as input the partially decrypted cipher text *TCT* and the user's secret key SK_1. Then it decrypts the cipher text as follows:

  1. If there is no attribute revoked, then $TCT = (C', E, F)$. Then the user computes:

$$\begin{aligned} C' / (E \cdot F^{z}) &= m \cdot e(g, g)^{\alpha s} / (e(g, g)^{\alpha_2 s} \cdot (e(g, g)^{\alpha_1 s / z})^{z}) \\ &= m \cdot e(g, g)^{\alpha s} / (e(g, g)^{\alpha_2 s} \cdot e(g, g)^{\alpha_1 s}) \\ &= m \cdot e(g, g)^{\alpha s} / e(g, g)^{\alpha s} \\ &= m \end{aligned} \tag{12}$$

  2. If the attribute x of users list RL_x is revoked, then $TCT = (CH_x, C', \{B_i\}_{\rho(i) \neq x}, \{C_{i,1}', C_{i,2}', C_{i,3}'\}_{\rho(i) = x}, D, E)$.

If it satisfies that $ID \notin RL_x$, then the user can decrypt the broadcast encryption ciphertext successfully to obtain the corresponding exponent v_x, and then continues to compute:

$$\begin{aligned} B_i &= e(C_{i,1}', K_1) e(C_{i,2}', K_{i,2}) e(C_{i,3}', (K_{i,3})^{v_x}) \\ &= e(w^{\lambda_i} v^{t_i} v^{k}, g^{r}) e((u^{\rho(i)} h)^{-t_i} (u^{\rho(i)} h)^{-k}, g^{r_i}) \cdot \\ &\quad e((g^{t_i} g^{k})^{1/v_x}, ((u^{A_i} h)^{r_i} v^{-r})^{v_x}) \\ &= e(g, w)^{r\lambda_i} \end{aligned} \tag{13}$$

Moreover, if the attributes set S satisfies the access control policy (M, ρ), then the CSP computes the values $\{w_i \in Z_p\}_{i \in I}$ satisfying $\sum_{i \in I} w_i M_i = (1, 0, ..., 0)$ in polynomial time and continues to compute:

$$B = \prod_{i \in I} (B_i)^{w_i} = \prod_{i \in I} (e(g, w)^{r\lambda_i})^{w_i} = e(g, w)^{r} \tag{14}$$

$$\begin{aligned} F &= (D / B)^{z} = (e(g, g)^{\alpha_1 s / z} e(g, w)^{rs} / e(g, w)^{rs})^{z} \\ &= (e(g, g)^{\alpha_1 s / z})^{z} = e(g, g)^{\alpha_1 s} \end{aligned} \tag{15}$$

$$\begin{aligned} C' / (E \cdot F) &= m \cdot e(g, g)^{\alpha s} / (e(g, g)^{\alpha_2 s} \cdot e(g, g)^{\alpha_1 s}) \\ &= m \cdot e(g, g)^{\alpha s} / e(g, g)^{\alpha s} \\ &= m \end{aligned} \tag{16}$$

3.4. Security Proof

- *Theorem* 1: If the decisional q-type assumption holds in G and G_T, then there exists no polynomial time attacker to break our proposed large universe CP-ABE scheme with attribute level user revocation selectively, where the challenge matrix is $M^{*}(l^{*} \times n^{*})$ with $l^{*}, n^{*} \leq q$.
- *Init*: The challenger $\mathcal{B}$ takes as input a q-type challenge $\vec{y}, T$. In addition, the attacker $\mathcal{A}$ gives the challenge access control policy (M^{*}, ρ^{*}) and the revocation users list $RL_{x^{*}}$ of attribute x^{*} where M^{*} is an $l^{*} \times n^{*}$ matrix with $l^{*}, n^{*} \leq q$ and $\rho^{*}: [l] \to Z_p$ is a map function.

- *Setup*: The challenger $\mathcal{B}$ first chooses random exponents $\alpha', \alpha'' \in Z_p$ and implicitly sets $\alpha_1 = \alpha' + a^{q+1}$, $\alpha_2 = \alpha''$, $\alpha = \alpha' + a^{q+1} + \alpha''$ by setting $e(g,g)^{\alpha} = e(g^a, g^{a^q}) \cdot e(g,g)^{\alpha'} \cdot e(g,g)^{\alpha''}$. Note that this way α is correctly distributed and a is information-theoretically hidden from the attacker $\mathcal{A}$. Then $\mathcal{B}$ chooses random exponents $u', v', h' \in Z_p$ and uses the assumption instance to construct the following public keys:

$$\begin{aligned} g &= g \\ u &= g^{u'} \cdot \prod_{(j,k)\in[l,n]} (g^{a^k/b_j^2})^{\boldsymbol{M}^*_{j,k}} \\ h &= g^{h'} \cdot \prod_{(j,k)\in[l,n]} (g^{a^k/b_j^2})^{-\rho^*(j)\boldsymbol{M}^*_{j,k}} \\ w &= g^a \\ v &= g^{v'} \cdot \prod_{(j,k)\in[l,n]} (g^{a^k/b_j})^{\boldsymbol{M}^*_{j,k}} \\ e(g,g)^{\alpha} &= e(g^a, g^{a^q}) \cdot e(g,g)^{\alpha'} \cdot e(g,g)^{\alpha''} \end{aligned} \tag{17}$$

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

$$PK = (g, u, h, w, v, e(g,g)^{\alpha})$$

- *Query* 1: $\mathcal{A}$ makes to $\mathcal{B}$ a series of queries including the key generation query $\mathcal{O}_{kg}$ and re-encryption query $\mathcal{O}_{ree}$.
- $\mathcal{A}$ makes to $\mathcal{B}$ a key generation query $\mathcal{O}_{kg}$ associated with the identity ID_j and the attributes set S_j, if $ID_j \notin RL_{x^*}$, 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 $(\boldsymbol{M}^*, \rho^*)$, then $\mathcal{B}$ outputs $\perp$, otherwise it generates the secret key as follows:

$\mathcal{B}$ first computes the vector $\vec{w} = (w_1, ..., w_{n^*}) \in Z_p^n$ where $w_1 = -1$, and for all $\rho^*(i) \in S'_j$, it satisfies $\boldsymbol{M}^*_i \vec{w}^T = 0$. Then $\mathcal{B}$ chooses a random parameter $t \in Z_p$ and defines the exponent r as:

$$\begin{aligned} r &= t + w_1 a^q + w_2 a^{q-1} + ... + w_{n^*} a^{q+1-n^*} \\ &= r + \sum_{i\in[n^*]} w_i a^{q+1-i}. \end{aligned} \tag{18}$$

Next, $\mathcal{B}$ computes the key component K'_1 as:

$$\begin{aligned} K'_1 &= g^r = g^{(t + w_1 a^q + w_2 a^{q-1} + ... + w_{n^*} a^{q+1-n^*})} \\ &= g^t \prod_{i\in[n^*]} (g^{a^{q+1-i}})^{w_i} \end{aligned} \tag{19}$$

According to the definition of r and $w_1 = -1$, we know that w^r includes the item $g^{-a^{q+1}}$ that can be canceled by multiplying w^r with $g^{\alpha_1} = g^{\alpha'} g^{a^{q+1}}$, because we implicitly set $\alpha_1 = \alpha' + a^{q+1}$ when generating K'_0. In detail, it is constructed as follows:

$$\begin{aligned} K'_0 &= g^{\alpha_1} w^r = g^{\alpha'} g^{a^{q+1}} g^{at} \prod_{i\in[n^*]} (g^{a^{q+2-i}})^{w_i} \\ &= g^{\alpha'} (g^a)^t \prod_{i=2}^{n^*} (g^{a^{q+2-i}})^{w_i} \end{aligned} \tag{20}$$

Next, $\mathcal{B}$ will compute the key components $K'_{\sigma,2}, K'_{\sigma,3}, \forall \sigma \in S'_j$. Before this, it will first set the common part v^{-r} as follows:

$$\begin{aligned} v^{-r} &= v^{-t} (g^{v'} \prod_{(j,k)\in[l^*,n^*]} g^{a^k \boldsymbol{M}^*_{j,k}/b_j})^{-\sum_{i\in[n^*]} w_i a^{q+1-i}} \\ &= v^{-t} \prod_{i\in[n^*]} (g^{a^{q+1-i}})^{-v' w_i} \cdot \\ &\quad \prod_{(i,j,k)\in[n^*,l^*,n^*], i\neq k} (g^{a^{q+1+k-i}/b_j})^{-w_i \boldsymbol{M}^*_{j,k}} \cdot \\ &\quad \prod_{(i,j)\in[n^*,l^*]} (g^{a^{q+1}/b_j})^{-w_i \boldsymbol{M}^*_{j,i}} \end{aligned} \tag{21}$$

Let

$$v^{-t} \prod_{i\in[n^*]} (g^{a^{q+1-i}})^{-v' w_i} \cdot \prod_{(i,j,k)\in[n^*,l^*,n^*], i\neq k} (g^{a^{q+1+k-i}/b_j})^{-w_i \boldsymbol{M}^*_{j,k}} = \varphi$$

then we have

$$\begin{aligned} v^{-r} &= \varphi \cdot \prod_{(i,j)\in[n,l]} (g^{a^{q+1}/b_j})^{-w_i \boldsymbol{M}^*_{j,i}} \\ &= \varphi \cdot \prod_{j\in[l]} (g^{-<w, \boldsymbol{M}^*_j>})^{a^{q+1}/b_j} \\ &= \varphi \cdot \prod_{j\in[l], \rho^*(j)\notin S'_j} (g^{-<w, \boldsymbol{M}^*_j>})^{a^{q+1}/b_j} \end{aligned} \tag{22}$$

Note that, $\mathcal{B}$ can compute the part φ by using the parameters given in the assumption, while the remaining part has to be canceled by the term $(u^{s_\sigma} h)^{r_\sigma}$. Therefore, for each attribute $s_\sigma \in S'_j$, $\mathcal{B}$ chooses a random parameter $r'_\sigma \in Z_p$ and implicitly sets

$$\begin{aligned} r_\sigma &= r'_\sigma + r \cdot \sum_{i'\in[l],\, \rho^*(i')\notin S'_j} b_{i'} / (s_\sigma - \rho^*(i')) \\ &= r'_\sigma + t \cdot \sum_{i'\in[l],\, \rho^*(i')\notin S'_j} b_{i'} / (s_\sigma - \rho^*(i')) + \\ &\quad \sum_{(i,i')\in[n,l],\, \rho^*(i')\notin S'_j} w_i a^{q+1-i} b_{i'} / (s_\sigma - \rho^*(i')) \end{aligned} \tag{23}$$

Then $\mathcal{B}$ can compute the term $(u^{s_\sigma} h)^{r_\sigma}$ of key component $K'_{\sigma,3}$ as

$$\begin{aligned} &(u^{s_\sigma} h)^{r_\sigma} \\ &= (u^{s_\sigma} h)^{r'_\sigma} \cdot (K_{\sigma,2} / g^{r'_\sigma})^{u' s_\sigma + h'} \cdot \\ &\quad \prod_{(i',j,k)\in[n,l,n],\rho^*(i')\notin S'_j} (g^{t(s_\sigma - \rho^*(j))\boldsymbol{M}^*_{j,k} b_{i'} a^k / (s_\sigma - \rho^*(i') b_j^2)}) \cdot \\ &\quad \prod_{(i,i',j,k)\in[n,l,l,n],\rho^*(i')\notin S'_j} (g^{(s_\sigma - \rho^*(j)) w_i \boldsymbol{M}^*_{j,k} b_{i'} a^{q+1+k-i} / (s_\sigma - \rho^*(i') b_j^2}) \\ &= \varphi \cdot \prod_{(i,j)\in[n,l],\rho^*(j)\notin S'_j} (g^{(s_\sigma - \rho^*(j)) w_i \boldsymbol{M}^*_{j,i} b_j a^{q+1+i-j} / (s_\sigma - \rho^*(j) b_j^2)}) \\ &= \varphi \cdot \prod_{j\in[l],\rho^*(j)\notin S'_j} g^{<w, \boldsymbol{M}^*_j> a^{q+1}/b_j} \end{aligned} \tag{24}$$

where φ includes the remaining terms of the product. The terms φ and $K'_{\sigma,2}$ can be computed by using the parameters given in the assumption. The second term of $(u^{s_\sigma} h)^{r_\sigma}$ cancels exactly with the problematic term of v^{-r}. Therefore, $\mathcal{B}$ can compute the key components $K'_{\sigma,2}$ and $K'_{\sigma,3}$.

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

$$TK = (K_0 = (K_0')^{1/z},\ K_1 = (K_1')^{1/z},\ \{K_{\sigma,2} = (K_{\sigma,2}')^{1/z}, K_{\sigma,3} = (K_{\sigma,3}')^{1/z}\}_{\sigma \in S_j'} \tag{25}$$

Therefore, the secret key is set as: $SK_1 = (z, TK)$. Finally, $\mathcal{B}$ sends the transformation key *TK* to the attacker $\mathcal{A}$.

- $\mathcal{A}$ makes to $\mathcal{B}$ a re-encryption query $\mathcal{O}_{ree}$ associated with the revocation users list RL_x of attribute x and the cipher text $CT = (\mathbb{A}, C, C_0, \{C_{i,1}, C_{i,2}, C_{i,3}\}_{i \in [l]})$. Then $\mathcal{B}$ generates the re-encrypted cipher text as follows:

  1. If there is no attribute revoked, namely $RL_x = \Phi$, then the CSP chooses a random $k \in Z_p$ and re-encrypts the cipher text CT as follows:

$$\begin{aligned} &C' = C = m \cdot e(g,g)^{\alpha s}, C_0' = C_0 = g^s, C_1' = (C_0)^{1/k} = g^{s/k} \\ &\forall i = 1,2,...,l: \\ &C_{i,1}' = C_{i,1} \cdot v^k = w^{\lambda_i} v^{t_i} v^k, C_{i,2}' = C_{i,2} \cdot (u^{\rho(i)} h)^{-k} \\ &= (u^{\rho(i)} h)^{-t_i} (u^{\rho(i)} h)^{-k}, C_{i,3}' = C_{i,3} \cdot g^k = g^{t_i} g^k \end{aligned} \tag{26}$$

Therefore, the re-encrypted cipher text is set as $\perp\ RCT = (C', C_0', C_1', \{C_{i,1}', C_{i,2}', C_{i,3}'\}_{i=1}^l)$.

  2. If there is an attribute x revoked from a user ID_j, namely $RL_x \neq \Phi$, then the CSP will choose a random exponent $v_x \in Z_p$ and encrypt it as the cipher text header CH_x using the broadcast encryption scheme [22] for those users $ID_i, i \neq j$. Then the CSP also chooses a random $k \in Z_p$ and re-encrypts the ciphertext *CT* as follows:

$$\begin{aligned} &C' = C = m \cdot e(g,g)^{\alpha s}, C_0' = C_0 = g^s, C_1' = (C_0)^{1/k} = g^{s/k} \\ &\forall i = 1,2,...,l: \\ &\quad C_{i,1}' = C_{i,1} \cdot v^k = w^{\lambda_i} v^{t_i} v^k, \\ &\quad C_{i,2}' = C_{i,2} \cdot (u^{\rho(i)} h)^{-k} = (u^{\rho(i)} h)^{-t_i} (u^{\rho(i)} h)^{-k}, \\ &\quad for\ \rho(i) \neq x: C_{i,3}' = C_{i,3} \cdot g^k = g^{t_i} g^k \\ &\quad for\ \rho(i) = x: C_{i,3}' = (C_{i,3} \cdot g^k)^{1/v_x} = (g^{t_i} g^k)^{1/v_x} \end{aligned} \tag{27}$$

Therefore, the re-encrypted ciphertext is set as $RCT = (CH_x, C', C_0', C_1', \{C_{i,1}', C_{i,2}', C_{i,3}'\}_{i=1}^l)$. Finally, $\mathcal{B}$ sends *RCT* to the attacker $\mathcal{A}$.

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

$$C^* = m_\beta \cdot T \cdot e(g^s, g^{\alpha'}) \cdot e(g^s, g^{\alpha''}), C_0^* = g^s \tag{28}$$

Next, $\mathcal{B}$ selects random parameters $y_2', ..., y_{n^*}' \in Z_p$, and then sets the vector $\vec{v} = (s, sa + y_2', sa^2 + y_3', ..., sa^{n-1} + y_{n^*}') \in Z_p^{n^*}$ to implicitly share the key s. Since $\vec{\lambda} = \boldsymbol{M}^* \vec{v}$, we have

$$\lambda_\tau = \sum_{i \in [n]} \boldsymbol{M}_{\tau,i}^* sa^{i-1} + \sum_{i=2}^n \boldsymbol{M}_{\tau,i}^* y_i' \tag{29}$$

Let $\lambda_\tau' = \sum_{i=2}^n \boldsymbol{M}_{\tau,i}^* y_i'$ and λ_τ' is known to $\mathcal{B}$. For each row, $\mathcal{B}$ implicitly sets $t_\tau = -sb_\tau$. Next, it continues to compute:

$$\begin{aligned} &C_{\tau,1} = w^{\lambda_\tau} v^{t_\tau} \\ &= w^{\lambda_\tau'} \cdot \prod_{i \in [n]} g^{\boldsymbol{M}_{\tau,i}^* sa^{i-1}} \cdot (g^{sb_\tau})^{-v'} \cdot \prod_{(j,k) \in [l,n]} g^{-\boldsymbol{M}_{j,k}^* a^k sb_\tau / b_j} \\ &= w^{\lambda_\tau'} \cdot (g^{sb_\tau})^{-v'} \cdot \prod_{(j,k) \in [l,n], j \neq \tau} (g^{a^k sb_\tau / b_j})^{-\boldsymbol{M}_{j,k}^*} \\ &C_{\tau,2} = (u^{\rho^*(\tau)} h)^{-t_\tau} \\ &= (g^{sb_\tau})^{-(u'\rho^*(\tau) + h')} \cdot (\prod_{(j,k) \in [l,n]} g^{(\rho^*(\tau) - \rho^*(j)) \boldsymbol{M}_{j,k}^* a^k / b_j^2})^{-sb_\tau} \\ &= (g^{sb_\tau})^{-(u'\rho^*(\tau) + h')} \cdot \prod_{(j,k) \in [l,n], j \neq \tau} (g^{sb_\tau a^k / b_j^2})^{-(\rho^*(\tau) - \rho^*(j)) \boldsymbol{M}_{j,k}^*} \\ &C_{\tau,3} = g^{t_\tau} = (g^{sb_\tau})^{-1} \end{aligned} \tag{30}$$

Finally, $\mathcal{B}$ sends the challenge cipher text $CT^* = ((\boldsymbol{M}^*, \rho^*), C, C_0, \{C_{\tau,1}, C_{\tau,2}, C_{\tau,3}\}_{\tau \in [l]})$ to the attacker $\mathcal{A}$.

- *Query* 2: $\mathcal{A}$ continues to make to $\mathcal{B}$ a series of queries including the key generation query $\mathcal{O}_{kg}$ and the cipher text re-encryption query $\mathcal{O}_{ree}$ as in Query 1.
- *Guess*: The attacker $\mathcal{A}$ outputs its guess β' for β. If $\beta = \beta'$, then $\mathcal{A}$ outputs 0 denoting $T = e(g,g)^{\alpha^{q+1} s}$, otherwise outputs 1 denoting T is a random parameter in $\mathbb{G}_T$.

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

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

Otherwise, T is a random element in G_T, namely m_β is completely random in the view of $\mathcal{A}$, so we have:

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

4. Analysis

In this part, we will compare our proposed large universe 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 G; $|C_T|$ denotes the bit size of an element in G_T; $|C_p|$ denotes the bit size of an element in Z_p^*; C_T denotes the number of elements in the access control matrix associated with the cipher text; $|C_k|$ denotes the bit size of the key encryption key in Hur's scheme [9]; t denotes the number of attributes associated with the cipher text; 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; n_m denotes the number of revoked users.

Before the comparison, we will first carry out the concrete performance analysis on the broadcast encryption scheme [15] with constant length key and constant length ciphertext that will be used in our paper: the public key size is $(2n_u+1)|C_1|$, the master key size is $|C_p|$, the user's secret key size is $|C_1|$ and the ciphertext size is $2|C_1|$。

4.1. Functionality

The functionality comparison is demonstrated in Table 1, from which we can see that Liang's scheme only achieve the system level user revocation, which is impractical in the normal application. However, our scheme、Hur's scheme and Yang's scheme achieve the attribute level user revocation. 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-type assumption in the standard model, which has stronger security. Moreover, our scheme is applicable to large universe environment while the other three ones are limited to small universe environment.

Table 1. Comparison of functionality.

Scheme	Access control granularity	Universe	Model	Assumption
Liang *et al.* [12]	system level user revocation	Small	standard	DBDH
Hur *et al.* [9]	attribute level user revocation	Small	generic group	-
Yang *et al.* [21]	attribute level user revocation	Small	random oracle	q-parallel BDHE
Ours	attribute level user revocation	Large	standard	q-type

Table 2. Comparison of storage cost.

Entity	Liang	Hur	Yang	Ours
AA	$\|C_1\|+(2^{(\log n_u+1)}+1)\|C_p\|$	$\|C_p\|+\|C_1\|$	$(4+n_a)\|C_p\|$	$3\|C_p\|$
O	$((C_T/t)\cdot n_a+6)\|C_1\|+\|C_T\|+\|C_p\|$	$2\|C_1\|+\|C_T\|$	$(2n_a+4)\|C_1\|+\|C_T\|$	$(2n_u+6)\|C_1\|+\|C_T\|$
CSP	$(C_T+3)\|C_1\|+\|C_T\|$	$(2t+1)\|C_1\|+\|C_T\|+\frac{t\cdot n_u}{2}\|C_p\|$	$(3t+1)\|C_1\|+\|C_T\|$	$(3t+5)\|C_1\|+\|C_T\|$
U	$(k+3+C_T/t)(\log n_u+1)\|C_1\|+2(n_u-n_m)\log(n_u/(n_u-n_m))\|C_1\|$	$(2k+1)\|C_1\|+(\log n_u+1)C_k$	$(k+2)\|C_1\|$	$(2k+3)\|C_1\|+\|C_p\|$

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 n_u and in Yang's scheme grows linearly with n_u; The storage cost of data owner O is mainly generated by the public key *PK*. Hur's scheme has the shortest public key. The public key in Yang's scheme grows linearly with n_u, in Liang's scheme grows linearly with n_u and the column vector C_T/t of access control matrix with each other as the slope and in our scheme grows linearly with n_a and n_u, however, with constant slope; The storage cost of cloud service provider CSP is mainly generated by the ciphertext and ciphertext header. In Liang's scheme, the revocation is implemented by updating the key other than the ciphertext, therefore, the ciphertext grows linearly with G_T. In Yang's scheme not only updates the key but also updates the corresponding ciphertext, therefore, the ciphertext grows linearly with t. In Hur's scheme, the storage cost includes the ciphertext and ciphertext header, moreover, the ciphertext grows linearly with t, and the ciphertext header grows linearly with t and n_u with each other as the slope. In our scheme, the storage cost also includes the ciphertext and ciphertext header, moreover, the ciphertext and ciphertext header both grow linearly with t. The storage cost of the data user U is mainly generated by the secret key. Our scheme and Yang's scheme have shorter secret key which grows linearly with k. In Liang's scheme, the secret key is generated by using a binary tree, therefore, the size of secret key is associated with k、C_T/t and n_u. 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 k, but only grows logarithmically with n_u.

4.3. Computation Efficiency

In order to evaluate the computation efficiency of our proposed large universe 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 4G memory. The experiment uses a 160-bit elliptic curve group based on the pairing-based cryptography library (PBC-0.5.14) [13] and cpabe-0.11 [3] which selects the super singular 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 G and G_T is approximately 6.2 ms and 0.6 ms respectively. In addition, the selection time of a random element in G and 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 re-encryption time, moreover, we set $C_T/t=6$ and $n_u=8$.

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, is better than that of Hur's scheme and Liang's scheme. In particular, the key generation time in Liang's scheme is not only associated with the number of attributes, but also associated with the number C_T/t of columns in the 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.

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, 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 number C_T/t of columns in the access control matrix, therefore, 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 to decrypt. 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 to decrypt. Moreover, our scheme with no attribute revoked uses outsourced decryption, so the user needs only one exponent operation in G_T. In addition, the decryption time of our scheme with 50% attributes revoked is a quadratic function for the attributes used to decrypt, 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 to decrypt 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 re-encryption time 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 re-encryption 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 re-encryption time in Yang's scheme and Liang's scheme is shorter, it 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.

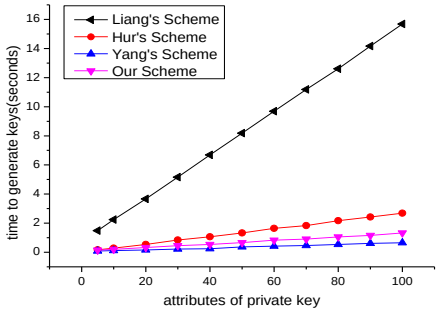

Figure 2. Time to generate keys.

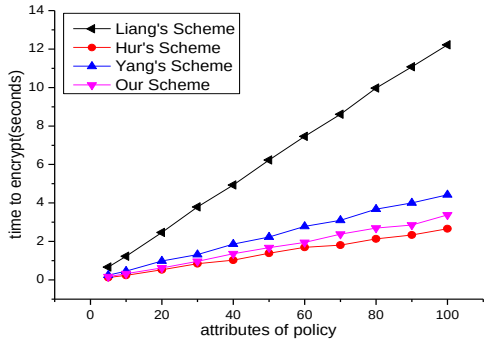

Figure 3. Time to encrypt.

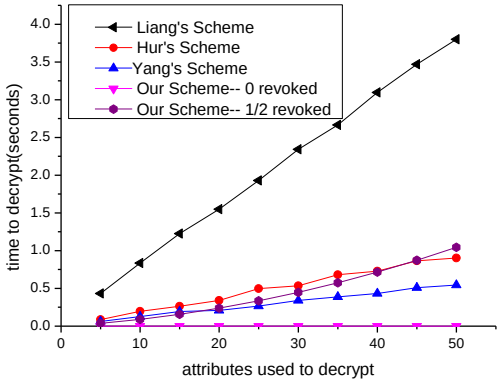

Figure 4. Time to decrypt.

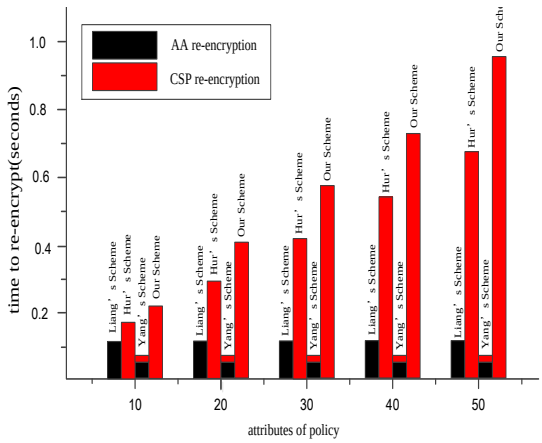

Figure 5. Time to re-encrypt.

5. Conclusions

In this paper, we propose a large universe CP-ABE scheme which can achieve the attribute level user revocation. Moreover, 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 AA, which decreases the computation cost of AA. Moreover, the user does not need to store additional parameters to carry out the attribute revocation, thus it greatly saves the storage space.

References

[1] Attrapadung N., Libert B., and de Panafieu D., "Expressive Key-Policy Attribute-Based Encryption with Constant-Size Ciphertexts," *in Proceedings of International Conference on Practice and Theory in Public Key Cryptography*, Taormina, pp. 90-108, 2011.

[2] Bethencourt J., Sahai A., and Waters B., "Ciphertext-Policy Attribute-Based Encryption," *in Proceedings of IEEE Symposium on Security and Privacy*, Berkeley, pp. 321-334, 2007.

[3] Bethencourt J., Sahai A., and Waters B., "Advanced Crypto Software Collection: the cpabetoolkit," http://acsc.cs.utexas.edu/cpabe/, Last Visited, 2017.

[4] Boldyreva A., Goyal V., and Kumar V., "Identity-based Encryption with Efficient Revocation," *in Proceedings of the 15th ACM Conference on Computer and Communications Security*, Alexandria, pp. 417-426, 2008.

[5] Boneh D., Gentry C., and Waters B., "Collusion Resistant Broadcast Encryption with Short Ciphertexts and Private Keys," *in Proceedings of the 25th Annual International Conference on Advances in Cryptology*, Santa Barbara, pp. 258-275, 2005.

[6] Cheung L. and Newport C., "Provably Secure Ciphertext-Policy ABE," *in Proceedings of the ACM Conference on Computer and Communication Security*, Alexandria, pp. 456-465, 2007.

[7] Goyal V., Pandey O., Sahai A., and Waters B., "Attribute-based Encryption for Fine-Grained Access Control of Encrypted Data," *in Proceedings of the 13th ACM Conference on Computer and Communications Security*, Alexandria, pp. 89-98, 2010.

[8] Goyal V., Jain A., Pandey O., and Sahai A., "Bounded Ciphertext Policy Attribute Based Encryption," *in Proceedings of International Colloquium on Automata, Languages, and Programming*, Reykjavik, pp. 579-591, 2015.

[9] Hur J. and Noh D., "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.

[10] Lewko A., Okamoto T., Sahai A., Takashima K., and Waters B., "Fully Secure Functional Encryption: Attribute-Based Encryption and (Hierarchical) Inner Product Encryption," *in Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques*, French Riviera, pp. 62-91, 2010.

[11] Lewko A. and Waters B., "Unbounded HIBE and Attribute-Based Encryption," *in Proceedings of International Conference on Theory and*

Applications of Cryptographic Techniques: Advances in Cryptology, Tallinn, pp. 547-567, 2011.

[12] Liang X., Lu R., and Lin X., "Ciphertext Policy Attribute Based Encryption with Efficient Revocation," *in Proceedings of the IEEE Symposium on Security and Privacy*, Berlin, pp. 321-334, 2008.

[13] Lynn B., "The Pairing-Based Cryptography," http://crypto.stanford.edu/pbc, Last Visited, 2017.

[14] Ostrovsky R., Sahai A., and Waters B., "Attribute-Based Encryption With Non-Monotonic Access Structures," *in Proceedings of the 14th ACM Conference on Computer and Communications Security*, Alexandria, pp. 195-203, 2007.

[15] Pirretti M., Traynor P., McDaniel P., and Waters B., "Secure Attribute-Based Systems," *in Proceedings of the 13th ACM Conference on Computer and Communications Security*, The Netherlands, pp. 99-112, 2006.

[16] Rouselakis Y. and Waters B., "Practical Constructions and New Proof Methods for Large Universe Attribute-Based Encryption," *in Proceedings of ACM Sigsac Conference on Computer & Communications Security*, Berlin, pp. 463-474, 2013.

[17] Sahai A. and Waters B., "Fuzzy Identity-Based Encryption," *in Proceedings of International Conference on Theory and Applications of Cryptographic Techniques*, Aarhus, pp. 457-473, 2005.

[18] Staddon J., Golle P., Gagn M., and Rasmussen P., "A Content-Driven Access Control System," *in Proceedings of the, Symposium on Identity and Trust on the Internet*, Gaithersburg, pp. 26-35, 2008.

[19] Tu S., Niu S., and Li H., "A Fine-Grained Access Control and Revocation Scheme on Clouds," *Concurrency and Computation Practice and Experience*, vol. 28, no. 6, pp. 1697-1714, 2016.

[20] Waters B., "Ciphertext-Policy Attribute-Based Encryption: An Expressive, Efficient, and Provably Secure Realization," *in Proceedings of the 14th International Conference on Practice and Theory in Public Key Cryptography Conference on Public Key Cryptography*, Taormina, pp. 53-70, 2011.

[21] Yang K., Jia X., and Ren K., "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*, Hangzhou, pp. 523-528, 2013.

Huijie Lian he is currently pursuing the Ph.D. degree in Institute of Information Science and Technology, Zhengzhou, China. His research interests include the big data security and privacy.

Qingxian Wang he was born in 1960. Now he is a PhD supervisor. His main research interests include network security and the big data security.

Guangbo Wang he was born in 1987. His research interests include cryptograph theory especially attribute-based encryption.