Ciphertext Policy Attribute Based Encryption with Efficient Revocation

Xiaohui Liang[†], Rongxing Lu[†], Xiaodong Lin[‡], and Xuemin (Sherman) Shen[†]
[†]Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, Canada
[‡]Faculty of Business and Information Technology, University of Ontario Institute of Technology, Canada
{x27liang, rxlu, xshen}@bbcr.uwaterloo.ca; xiaodong.lin@uoit.ca

ABSTRACT

Revocation is a vital open problem in almost every cryptosystem dealing with malicious behaviors. In ciphertext policy attribute based encryption, unlike traditional public key cryptosystem, different users may hold the same functional secret keys related with the same attribute set leading to additional difficulties in designing revocation mechanism. In this paper, we propose the ciphertext policy attribute based encryption scheme with efficient revocation which can be proved secure in the standard model. Our construction uses linear secret sharing and binary tree techniques as the underlying tools. In addition to assigned attribute set, each user is also assigned with a unique identifier. Therefore, a user can be easily revoked by using his/her unique identifier; on the other hand, the encryption and decryption algorithms of ABE (Attribute Based Encryption) can be done without any involvement of these unique identifiers. Then, we prove the chosen plaintext security of our construction based on Decisional Bilinear Diffie-Hellman (DBDH) assumption in the standard model. Finally, we provide some discussion on the efficiency of our scheme and the extensions including delegation capability and chosen ciphertext security.

Categories and Subject Descriptors

E.4 [**Data Encryption**]: Public Key Cryptosystems

General Terms

Security, Theory

Keywords

Efficient Revocation, Ciphertext Policy Attribute Based Encryption, Standard Model

1. INTRODUCTION

Ciphertext Policy Attribute based Encryption (CP-ABE), similar with role-based access control system, can be widely applied to realize access control in many applications including medical systems and education systems. For example, the sensitive medical records, tightly related to patients' privacy, must be accessed only if the users are authorized with patients' consent; solutions of exams in the education online system also should be only read by professors or specified teaching assistants. The CP-ABE scheme deals with those situations, by encrypting the target information with expressive access policies, such as "Medicine" and "Physician", "Professor" or ("Computer Science" and "Teaching Assistant"). In fact, CP-ABE can provide a perfect solution to an access control system by considering, efficient distributing, expressive access control and data confidentiality.

In the traditional CP-ABE scheme, once users obtain the credentials from a system manager at the beginning of setup phase, the access ability is always valid for those who may even break the confidential rules by abusing these private information. Upon detecting those malicious adversaries, without any revocation mechanism embedded, the system manager has to rebuild up the whole system. Therefore, revocation mechanism should be designed into the system from the beginning rather than being added after the other issues are addressed, as it requires careful planning on where functionality should be placed and how to reduce the computational and communication costs. In this paper, we aim at developing the CP-ABE scheme with efficient revocation.

Designing a revocation mechanism for CP-ABE is not a simple task while considering the following aspects: first, system manager only associates user secret keys with different sets of attributes instead of individual characteristics. The fuzzy identities therefore encumber the system's revocation on one specified user; second, users' individuality are taken place by several common attributes, and thus revocation on attributes or attribute sets can not accurately exclude the users with misbehaviors; third, the system must be secure against collusion attack from revoked users even though they share some common attributes with non-revoked users.

To consider the revocation problem in a traditional CP-ABE scheme, limited choices are available. One is the revocation of a single attribute, which is not in connection with users' behaviors but more likely to be periodical update of universal attribute set of the whole system. Another possible solution is to revoke one attribute set corresponding to one specific set of users. In this way, all the users' access abilities will be revoked if they share the same attribute set with the malicious user, which is inappropriate in the real application.

Contribution. As a solution to accurately and efficiently revoke the users' access abilities, we modify the traditional CP-ABE model to CP-ABE-R (Ciphertext Policy Attribute Based Encryption with Revocation) model in which each user is identified by a unique identifier. However, the encryption and decryption algorithms are completed without the involvement of these unique identifiers. The system manager assigns user secret keys along different paths in the revocation tree according to different unique identifiers, and publishes the revocation information according to a time stamp, such as "Sep. 21th, 2009 to Sep. 28th, 2009". A sender, without any knowledge of the receivers' unique identifiers, encrypts the data with an access policy and a time stamp initialized by the system manager. The update information is the primary trigger controlling the user's access ability. If the user's unique identifier is not in the revocation list during "Sep. 9th, 2009" to "Sep. 19th, 2009", he/she possesses the access ability corresponding to his/her attribute set; otherwise, he/she would be deprived of all access abilities. Therefore, this CP-ABE-R model leverages the expressive access control ability with accurate and efficient revocation.

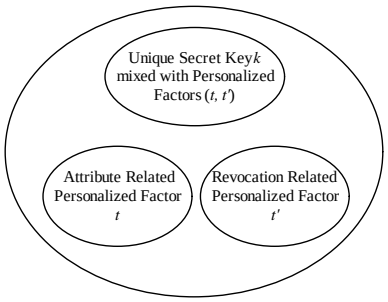

Figure 1: The components of secret keys

In our construction, the binary tree technique is adopted to reduce communication and computational costs during the update phase. To avoid the need for maintaining secure channels between the system manager and the non-revoked users, the update information is generated corresponding to the key update nodes, a minimum set providing enough information to the non-revoked users and no useful information to the revoked users. In our design, a user secret key consists of three components: a unique secret key k mixed with personalized factors including attribute related personalized factor t and revocation related personalized factor t', as shown to Figure 1. t and t' are two independent random variables among different users, and those personalized factors are key components to resist collusion attack, which will be elaborated in Section 4. In our decryption algorithm, non-revoked users select the available update information to depersonalize the factor t', extract the access structure components in ciphertext to depersonalize the factor t, and eventually use unique secret key k to decrypt the message. To the best of our knowledge, this work is the first attempt to address the revocation issue in ciphertext policy attribute based encryption.

Related works. The revocation problem in public key encryption scheme has been well studied [17, 19]. Efficient revocation of certificates has been an active topic in the past several years [10, 18]. Gentry [9] also discusses the certificate revocation problem in certificate-based encryption scheme.

Several research works [3, 4, 15] related to revocation in identity based encryption (IBE) setting are as follows. The schemes in [4, 15] accommodate a special semi-trusted third party called mediator who is able to provide help for the non-revoked users on decryption. Boldyreva et.al. [3] adopted the techniques of fuzzy IBE and binary tree to implement a revocation scheme in the IBE setting which reduces the amounts of update information in comparison with previous works. They also presented an intuitional way to apply the same technique to KP-ABE scheme [12] and Fuzzy IBE scheme [20]. Yu et. al. [22] proposed a tailored key policy attribute based encryption with revocation especially for fined-grained distribute data access control in wireless sensor networks. However, the periodical change of system public parameters in [22] introduces extra computational and communication costs. In this paper, we construct a ciphertext policy attribute based encryption scheme with efficient revocation by using binary tree technique.

For the research on solving revocation problem in broadcast encryption, Boneh and Waters 2006, [6] introduced a new primitive called augmented broadcast encryption scheme which can be constructed for broadcast encryption with trace-and-revoke function. Lewko et.al. [13] proposed a revocation scheme with very small secret keys but revocation list is controlled by the encrypter itself. In our scheme, the revocation list is controlled by the system manager, which will be more realistic for the practical scenario suitable for the ABE setting.

CP-ABE allows a sender to disseminate messages according to an access policy which can be expressed as a boolean function consisting of (OR, AND) gates between attributes. A receiver whose secret key is associated with those attributes could only decrypt a ciphertext successfully if and only if its attributes satisfy the ciphertext's access policy. Bethencourt et. al. [2] proposed the first CP-ABE scheme which can not be proved secure in the standard model. Then, many elegant works [8, 11, 14] proved secure in standard model have been presented later, where tradeoff between expressiveness of access policy and security assumption is made. Waters [21] developed a general method to construct a ciphertext policy attribute based encryption scheme using linear secret sharing technique and his schemes are the most efficient so far. The construction proposed in this paper follows Waters' work [21].

Organization. Section 2 gives a brief review on definition of linear secret sharing scheme, a new definition on algorithms in CP-ABE-R model and the corresponding security model for CP-ABE-R. In Sections 3 and 4, we propose a ciphertext policy attribute based encryption scheme with efficient revocation mechanism according to CP-ABE-R model and present a complete proof in the standard model. The discussion on efficiency, delegating capability and chosen ciphertext security of our scheme are given in Section 5 and we conclude our paper in Section 6.

2. PRELIMINARIES

2.1 Linear Secret Sharing Schemes

Linear Secret Sharing Scheme (LSSS)[1] is a useful technique in constructing attribute based crypto-systems [16, 21]. It can be summarized as follows:

DEFINITION 1 (LINEAR SECRET SHARING SCHEME[1]). *A secret-sharing scheme* Π *over a set of parties* $\mathcal{P}$ *is called*

linear (over $\mathbb{Z}_p$) if

1. *The shares for each party form a vector over $\mathbb{Z}_p$.*
2. *There exists a matrix M called the share-generating matrix for π. The matrix M has l rows and n columns. For all $i = 1, \cdots, l$, the i'th row of M we let the function ρ defined the party labeling row i as $\rho(i)$. When we consider the column vector $v = (s, r_2, \cdots, r_n)$, where $s \in \mathbb{Z}_p$ is the secret to be shared, and $r_2, \cdots, r_n \in \mathbb{Z}_p$ are randomly chosen, then Mv is the vector of l shares of the secret s according to π. The share $(Mv)_i$ belongs to party $\rho(i)$.*

Suppose that π is an LSSS for access structure $\mathbb{A}$[1]. Let $\mathcal{S} \in \mathbb{A}$[2] be any authorized set, and let $\mathcal{I} \subset \{1, 2, \cdots, l\}$ be defined as $\mathcal{I} = \{i : \rho(i) \in \mathcal{S}\}$. Then, there exist constants $\{\omega \in \mathbb{Z}_p\}_{i \in \mathcal{I}}$ such that, if $\{\lambda_i\}$ are valid shares of any secret s according to π, then $\sum_{i \in \mathcal{I}} \omega_i \lambda_i = s$. Also, these constants ω_i can be found in time polynomial in the size of the share-generating matrix M (refer to [1]).

2.2 CP-ABE-R

A CP-ABE-R scheme includes a tuple of probabilistic polynomial-time algorithms as follows.

- **Setup**$(\mathcal{U}, n_{max})$ The setup algorithm takes the universal attribute set $\mathcal{U}$ and the maximum size n_{max} of columns in an access structure as input. It outputs the public parameters $\mathcal{PP}$ and a master key $\mathcal{MK}$.
- **KGen**$(u_{id}, \mathcal{S}, \mathcal{MK})$ The key generation algorithm takes a unique identifier u_{id}, an attribute set $\mathcal{S} \subseteq \mathcal{U}$ and the master key $\mathcal{MK}$ as input. It outputs a secret key $(u_{id}, \mathcal{SK})$ to the user.
- **KUpd**$(rl, t, \mathcal{MK})$ The key update algorithm takes a revocation list rl, a time stamp t and the master key $\mathcal{MK}$ as input. It outputs the update information $\mathcal{UI}$.
- **Enc**$(\mathcal{PP}, (M, \rho), \mathcal{M}, t)$ The encryption algorithm takes the public parameters $\mathcal{PP}$, an access structure (M, ρ), a message $\mathcal{M}$ and a time stamp t as input. It outputs the ciphertext $\mathcal{C}$ with (M, ρ).
- **Dec**$(\mathcal{C}, (M, \rho), \mathcal{SK}, \mathcal{UI})$ The decryption algorithm takes the ciphertext $\mathcal{C}$ with (M, ρ), the secret key $\mathcal{SK}$ and the update information $\mathcal{UI}$. If the attribute set related with $\mathcal{SK}$ satisfies the access structure (M, ρ) and the unique identifier associated with $\mathcal{SK}$ has not been revoked in update information $\mathcal{UI}$, it decrypts the ciphertext and returns a message $\mathcal{M}$; else, it returns $\perp$.

2.3 Security Model for CP-ABE-R

Selective Access Structure Model for CP-ABE-R Selective Access Structure (SAS) Model is widely used in analyzing ciphertext policy attribute based encryption schemes [8, 11, 14, 21]. In this paper, we propose the SAS model for a ciphertext policy attribute based encryption with revocation. The CP-ABE-R scheme is secure in the SAS model if no probabilistic polynomial time adversary $\mathcal{A}$ has a non-negligible advantage in winning the following game.

[1]$\mathbb{A}$ can be related to an access structure consisting of (OR, AND) gates between attributes.

[2]The attribute set $\mathcal{S}$ satisfies the access structure $\mathbb{A}$.

INIT $\mathcal{A}$ chooses an access structure (M^*, ρ^*), a set of unique identifier u^* and a time stamp t^* that he wishes to be challenged upon, where the column of M^* is no larger than n_{max}. The queries of key generation oracle for $\mathcal{S} \in (M^*, \rho^*)$ are associated with the unique identifiers in u^*. The challenger runs **Setup** algorithm and gives $\mathcal{A}$ the resulting public parameters $\mathcal{PP}$. It keeps the corresponding master key $\mathcal{MK}$ for itself.

PHASE 1 $\mathcal{A}$ issues several queries to key generation oracle, revoke oracle and key update oracle.

- Key generation oracle: $\mathcal{A}$ issues queries for secret keys related to several tuples $(u_{id}, \mathcal{S})$.
- Revoke oracle: $\mathcal{A}$ inputs several revoked unique identifiers u_{id} and a time stamp t, the simulator adds u_{id} to the revocation list rl at time t.
- Key update oracle: for any time stamp t, the key update information is generated according to the revocation list collected in revoke oracle.

CHALLENGE Once $\mathcal{A}$ decides that **PHASE 1** is over, it generates two messages $\mathcal{M}_0$ and $\mathcal{M}_1$ from the message space of equal length. The challenger chooses $\mu \in \{0, 1\}$ at random and encrypts $\mathcal{M}_\mu$ with (M^*, ρ^*). Then, the ciphertext $\mathcal{C}^*$ is given to $\mathcal{A}$.

PHASE 2 The same as **PHASE 1**.

GUESS $\mathcal{A}$ outputs a guess $\mu' \in \{0, 1\}$ and wins the game if $\mu' = \mu$.

The following **conditions** must always hold:

1. In the key generation oracle, once $(\mathcal{S}, u_{id})$ is queried, the adversary will not query any other tuples $(\mathcal{S}', u_{id})$, where $\mathcal{S}' \neq \mathcal{S}$.
2. In the key update oracle, at time t, the key update oracle outputs $\mathcal{UI}$ based on the information collected from revocation oracle before t.
3. In the key generation oracle, if the adversary queries on $u_{id} \in u^*$, then for all $u_{id} \in u^*$, revocation oracle must be queried on (u_{id}, t^*).

We define $\mathcal{A}$'s advantage in this game as $|\Pr[\mu' = \mu] - \frac{1}{2}|$.

3. THE CP-ABE-R SCHEME

The CP-ABE-R scheme supports the access structure (M, ρ), where the column of M is no larger than n_{max} and ρ is an injective function.

Let $\mathbb{G}$ and $\mathbb{G}_T$ be two (multiplicative) cyclic groups of the same order p, where p is a large prime. Suppose $\mathbb{G}$ and $\mathbb{G}_T$ are equipped with a pairing, i.e., a non-degenerated and efficiently computable bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ such that $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab} \in \mathbb{G}_T$ for all $a, b \in \mathbb{Z}_p^*$ and any $(g_1, g_2) \in \mathbb{G}^2$ [5].

Setup$(\mathcal{U}, n_{max})$ The setup algorithm takes as input the universal attribute set $\mathcal{U}$ in the system and the maximum size n_{max} of columns in one access structure. It first chooses a

group $\mathbb{G}$ of prime order p and a generator g. In addition, it chooses random exponents $\alpha, a, b, d \in \mathbb{Z}_p$, random elements $h_{j,x} \in \mathbb{G}$ and a function $H : \mathbb{Z}_p \to \mathbb{G}$.

Define $H(x) = g^{bx^2} \prod_{i=1}^{3} h_i^{\triangle_{i,3}(x)}$, where $(h_i)_{1\le i\le 3} \in \mathbb{G}$ and the lagrange coefficient $\triangle_{i,3}(x) \stackrel{def}{=} \prod_{j=1,j\neq i}^{3} \left(\frac{x-j}{i-j}\right)$.

The system public parameters are

$$\mathcal{PP} = \langle g, e(g,g)^{\alpha}, A = g^a, B = g^b, (h_{j,x})_{1\le j\le n_{max}, x\in\mathcal{U}}, d, H\rangle$$

For revoking users' access abilities after key generation

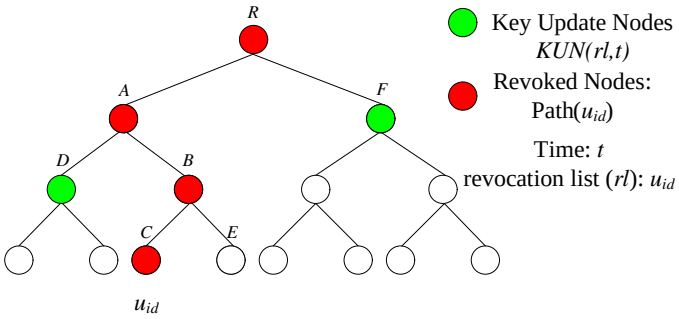

Figure 2: Revocation Tree T

phase, the system manager builds up a revocation tree first, denoted as T in Figure 2. The figure shows an example of revocation tree with height 3. The revocation tree corresponds to time t and the identifier of revoked user is u_{id} which is associated with one leaf node. $Path(u_{id})$ (red nodes) records all the nodes along the path from the leaf node C to the root R. $KUN(rl,t)$ (green nodes) records all the nodes that covers the leaf nodes associated with non-revoked users. In reality, the system manager adopts a revocation tree with height n and the maximum users is up to 2^n. For each node y, it selects random number $a_y \in \mathbb{Z}_p^*$ and stores this value to the node. We let the parameter T represent all the values of a_y.

The system master key is

$$\mathcal{MK} = \langle g^{\alpha}, a, b, T\rangle$$

KGen$(u_{id}, \mathcal{S}, \mathcal{MK})$

The key generation algorithm takes a unique identifier u_{id}, an attribute set $\mathcal{S}$ and the master key $\mathcal{MK}$ as input.

Firstly, the algorithm checks the unique identifier u_{id} to see whether it has been queried before. If the answer is yes, $\mathcal{S}$ must be the same with the one in the previous query and the algorithm outputs the same secret key; if not, the algorithm chooses a vacant leaf node and bind it with u_{id}. $Path(u_{id})$ can be generated as the red nodes shown in Figure 2.

Then, the algorithm chooses $t_y, (t_{j,y})_{1\le j\le n_{max}}, r_{d,y} \in \mathbb{Z}_p$ for $y \in Path(u_{id}), 1 \le j \le n_{max}$, and outputs the secret key $\mathcal{SK} =$

$$\langle \{(L_{j,y})_{1\le j\le n_{max}} (K_{x,y})_{x\in\mathcal{S}}, K_y, D_y, d_y\}_{y\in Path(u_{id})}\rangle =$$

$$\langle \{(g^{t_{j,y}})_{1\le j\le n_{max}}, (\prod_{1\le j\le n_{max}} h_{j,x}^{t_{j,y}})_{x\in\mathcal{S}},$$

$$g^{\alpha} g^{at_{1,y}} g^{bt_y}, B^{a_y d + t_y} H(d)^{r_{d,y}}, g^{r_{d,y}}\}_{y\in Path(u_{id})}\rangle$$

Notice that for different users, a_y remains the same while $t_y, t_{j,y}, r_{d,y}$ are with different values.

KUpd(rl, t, T)

The key update algorithm takes revocation list rl, current time t and a revocation tree T as input.

Firstly, the algorithm marks all the nodes red along $Path(u_{id})$, where $u_{id} \in rl$.

$KUN(rl,t)$ can be generated as the minimum cover set (green) of rest uncolored nodes.

Then, the algorithm chooses $t, r_{t,y} \in \mathbb{Z}_p$, and outputs the updated information $\mathcal{UI} = \langle \{E_y, e_y\}_{y\in KUN(rl,t)}\rangle =$

$$\langle \{B^{a_y t} H(t)^{r_{t,y}}, g^{r_{t,y}}\}_{y\in KUN(rl,t)}\rangle$$

where $r_{t,y} \in \mathbb{Z}_p$ are random numbers.

Notice that for different time stamp t, a_y are the same while $r_{t,y}$ are with different values.

Encrypt$(\mathcal{PP}, (M,\rho), \mathcal{M}, t)$

The encryption algorithm takes system public parameters $\mathcal{PP}$, an access structure (M,ρ), a message $\mathcal{M}$ and current time t as input.

If M is a matrix with size $l \times n$, we expand M to a $l \times n_{max}$ matrix by filling element 0 into the columns from $(n+1)$-th to n_{max}-th. Note that such conversion does not affect the satisfying logic of an access structure. The algorithm first chooses a random vector $\bar{v} = (s, y_2, \cdots, y_{n_{max}}) \in \mathbb{Z}_p^{n_{max}}$. Denote $M_{i,j}$ as the ith row, jth column element of M. Then, the algorithm outputs the ciphertext

$$\mathcal{C} = \langle C_{\mathcal{M}}, C_s, (C_{i,j})_{1\le i\le l, 1\le j\le n_{max}}, C_d, C_t\rangle =$$

$$\langle \mathcal{M}\cdot e(g,g)^{\alpha s}, g^s, (g^{aM_{i,j}v_j} h_{j,\rho(i)}^{-s})_{1\le i\le l, 1\le j\le n_{max}},$$

$$H(d)^s, H(t)^s\rangle$$

with (M,ρ).

Decrypt$(\mathcal{C}, \mathcal{SK}, \mathcal{UI})$

The decryption algorithm takes the ciphertext $\mathcal{C}$, a user secret key $\mathcal{SK}$ and the published update information $\mathcal{UI}$ as input.

If the user is non-revoked, according to the definition of $Path(u_{id})$ and $KUN(rl,t)$, it is able to find $\bar{y} \in Path(u_{id}) \cap KUN(rl,t)$,

Extract two sets of elements from $\mathcal{SK}$ and $\mathcal{UI}$:

$$\langle (L_{j,\bar{y}})_{1\le j\le n_{max}}, (K_{x,\bar{y}})_{x\in\mathcal{S}}, K_{\bar{y}}, D_{\bar{y}}, d_{\bar{y}}\rangle \text{ and } \langle E_{\bar{y}}, e_{\bar{y}}\rangle$$

1. It obtains $e(g,g)^{at_{1,\bar{y}}s}$ as follows:

   If the attribute set $\mathcal{S}$ of $\mathcal{SK}$ satisfies the access structure (M,ρ) in the ciphertext $\mathcal{C}$, it is able to find set $I = \{i|\rho(i) \in \mathcal{S}\}$, and then calculates its weight $(\omega_j)_{j\in\mathcal{I}}$ so that $\prod_{i\in\mathcal{I}} M_{i,1}\omega_i = 1$ and for $2 \le j \le n_{max}$, $\prod_{i\in\mathcal{I}} M_{i,j}\omega_i = 0$. Thus,

$$
\begin{aligned}
& \prod_{1\le j\le n_{max}} e(\prod_{i\in\mathcal{I}} C_{i,j}^{\omega_i}, L_{j,\bar{y}}) \cdot e(\prod_{i\in\mathcal{I}} K_{\rho(i),\bar{y}}^{\omega_i}, C_s) \\
= & \prod_{1\le j\le n_{max}} \prod_{i\in\mathcal{I}} (e(C_{i,j}^{\omega_i}, L_{j,\bar{y}}) \cdot e(K_{\rho(i),\bar{y}}^{\omega_i}, C_s)) \\
= & \prod_{1\le j\le n_{max}} \prod_{i\in\mathcal{I}} e(g^{aM_{i,j}\omega_i v_1}, g^{t_{j,\bar{y}}}) \\
= & \prod_{i\in\mathcal{I}} e(g^{aM_{i,1}\omega_i v_1}, g^{t_{1,\bar{y}}}) \\
= & e(g,g)^{at_{1,\bar{y}}s}
\end{aligned}
$$

2. It obtains $e(g,g)^{t_{\bar{y}}bs}$ as follows:

$$e(D_{\bar{y}}, C_s)/e(d_{\bar{y}}, C_d) = e(g,g)^{(a_{\bar{y}}d+t_{\bar{y}})bs}$$

$$e(E_{\bar{y}}, C_s)/e(e_{\bar{y}}, C_t) = e(g,g)^{(a_{\bar{y}}t+b_{\bar{y}})bs}$$

$$\text{using lagrange interpolation} \Rightarrow e(g,g)^{t_{\bar{y}}bs}$$

3. Finally, it decrypts as follows:

$$e(C_s, K_{\bar{y}}) = e(g,g)^{\alpha s} e(g,g)^{at_{1,\bar{y}}s} e(g,g)^{t_{\bar{y}}bs}$$

$$\text{using steps 1\&2's results to obtain} \Rightarrow e(g,g)^{\alpha s} \Rightarrow \mathcal{M}$$

4. SECURITY ANALYSIS OF CP-ABE-R

4.1 Complexity Assumption

The security of our construction is reduced to Decisional Bilinear Diffie-Hellman (DBDH) assumption. In the following, we introduce the DBDH problem and its corresponding assumption.

Decisional Bilinear Diffie-Hellman Problem. An algorithm $\mathcal{S}$ is a ε'-solver of the DBDH problem if it distinguishes with probability at least $\frac{1}{2}+\varepsilon'$ between the two following probability distributions:

$\mathcal{D}_{bdh} = (g, g^a, g^b, g^c, e(g,g)^{abc})$, where a,b,c are chosen randomly in $\mathbb{Z}_p$, g is a generator of group $\mathbb{G}$ and e is a bilinear mapping from $\mathbb{G}\times\mathbb{G}\to\mathbb{G}_T$,

$\mathcal{D}_{rand} = (g, g^a, g^b, g^c, Z)$, where a,b,c are chosen randomly in $\mathbb{Z}_p$ and Z is chosen randomly in $\mathbb{G}_T$.

DEFINITION 2. *The DBDH assumption holds in G and G_T if no any probabilistic polynomial-time ε'-solver of the DBDH problem for non-negligible values of ε'.*

4.2 Security Analysis

THEOREM 1. *If the DBDH assumption holds in $(\mathbb{G},\mathbb{G}_T)$, then the proposed CP-ABE-R scheme is chosen plaintext secure in the SAS model.*

PROOF. Suppose there exists a polynomial-time adversary $\mathcal{A}$ who can win the game described in Section 2.3 with non-negligible advantage ϵ. We construct a simulator who can distinguish the DBDH tuple from a random tuple with non-negligible advantage $\frac{1}{2}\epsilon$.

We first let the challenger set the groups $\mathbb{G}$ and $\mathbb{G}_T$ with an efficient bilinear map e and two generators g,h of $\mathbb{G}$. The challenger flips a fair binary coin ν, outside of $\mathcal{S}$'s view. If $\nu=1$, the challenger sets $(g,A,B,C,Z)\in D_{bdh}$; otherwise, $(g,A,B,C,Z)\in D_{rand}$.

INIT The simulator runs $\mathcal{A}$. $\mathcal{A}$ chooses a challenge access structure (M^*,ρ^*), a set of unique identifier u^* and a time stamp t^* to be challenged upon, where M^* is $l^*\times n^*$ matrix. M^* is a $l^*\times n_{max}$ matrix filled with element 0 from n^*+1-th to n_{max}-th columns.

The simulator executes the following steps:

- It randomly selects $\alpha'\in\mathbb{Z}_p^*$, and implicitly sets $\alpha = ab+\alpha'$. Therefore, $e(g,g)^{\alpha} = e(g,g)^{\alpha'}\cdot e(A,B)$.
- For each (j,x) pair where $x\in\mathcal{U}$ and $1\le j\le n_{max}$, it chooses a random value $z_{x,j}\in\mathbb{Z}_p$. The group elements $h_{j,x}$ with $(x\in\mathcal{U}, 1\le j\le n_{max})$ can be generated: for $1\le j\le n^*$ and $x=\rho^*(i)$ for some $1\le i\le l^*$, $h_{j,x} = g^{z_{x,j}} g^{aM^*_{i,j}}$; otherwise, $h_{j,x} = g^{z_{x,j}}$.
- It randomly selects $d\in\mathbb{Z}_p$.
- It picks random second-degree polynomials $f(x), u(x)$ with coefficients in $\mathbb{Z}_p$, s.t. $u(x) = -x^2$ for $x = d, t^*$, o.w. $u(x)\neq -x^2$. For $1\le i\le 3$, $h_i = B^{u(i)}g^{f(i)}$. Thus, $H(x) = B^{x^2+u(x)}g^{f(x)}$.

Finally, the simulator publishes the public parameters

$$\mathcal{PP} = \langle g, e(g,g)^{\alpha}, A, B, (h_{j,x})_{1\le j\le n_{max}, x\in\mathcal{U}}, d, H\rangle$$

The simulator randomly selects $\mathbf{tp}\in\{0,1\}$. If $[\mathbf{tp}=0]$, the simulator doesn't output the secret keys for $(\mathcal{S}\in(M^*,\rho^*), u_{id}\in u^*)$ but outputs the key update information for any (rl,t); otherwise $[\mathbf{tp}=1]$, the simulator outputs the secret keys for $(\mathcal{S}\in(M^*,\rho^*), u_{id}\in u^*)$ and outputs the key update information only for the tuple (rl,t^*) with $u^*\subseteq rl$.

The simulator randomly associates several leaf nodes of T to each $u_{id}\in u^*$ and denotes $Path(u^*) = \bigcup_{u_{id}\in u^*} Path(u^*_{id})$. Depending on $\mathbf{tp}$, the simulator also sets values of the revocation tree T in two separate ways.

If $[\mathbf{tp}=\mathbf{0}]$ The simulator randomly chooses $a_y\in\mathbb{Z}_p$ for all the nodes in T.

If $[\mathbf{tp}=\mathbf{1}]$ The simulator randomly chooses $l_y\in\mathbb{Z}_p$ for $y\in Path(u^*)$ by implicitly setting $l_y = a_y d - a$. The simulator also randomly chooses $a_y\in\mathbb{Z}_p$ for the rest nodes $y\notin Path(u^*)$.

Notice that the master key is kept by system manager only in the real environment and thus it is unnecessary for the simulator to control them.

PHASE 1 $\mathcal{A}$ is allowed to adaptively makes queries to the following oracles.

$\diamond$ Key generation oracle: $\mathcal{A}$ issues queries for secret keys related with several tuples $(u_{id},\mathcal{S})$. The simulator does follows depending on the value $\mathbf{tp}$:

If $[\mathbf{tp}=\mathbf{0}]$, if $(u_{id}\in u^*, \mathcal{S}\in(M^*,\rho^*))$, the simulator outputs $\perp$; otherwise, $(u_{id}\notin u^*, \mathcal{S}\notin(M^*,\rho^*))$.

According to the setting of revocation tree in the **INIT** phase, for $y\in Path(u_{id})$, a_y is known. Then, the simulator executes procedure **KG1** with input a_y. **KG1** outputs a partial secret key related to one node y, as shown in Figure 3.

It randomly selects $r_i\in\mathbb{Z}_p$ for $1\le i\le n_{max}$. Then, as $\mathcal{S}\notin(M^*,\rho^*)$, it is able to find a vector $\vec{\omega} = (\omega_1,\cdots,\omega_{n_{max}})\in\mathbb{Z}_p^{n_{max}}$ so that $\omega_1=-1$ and for all i where $\rho^*(i)\in\mathcal{S}$ we have that $\vec{\omega}\cdot M_i^* = 0$. Due to the definition of LSSS, such a vector $\vec{\omega}$ exists. (for $n^*<j\le n_{max}$, since $M^*_{i,j}=0$, we could simply let $\omega_j=0$).

randomly selects $r_i \in \mathbb{Z}_p$ for $1 \le i \le n_{max}$.
Find $\vec{\omega} = (\omega_1, \cdots, \omega_{n_{max}}) \in \mathbb{Z}_p^{n_{max}}$, so that
1. $\omega_1 = -1$;
2. for all i where $\rho^*(i) \in \mathcal{S}$ we have that $\vec{\omega} \cdot M_i^* = 0$.
For $1 \le j \le n_{max}$
$L_{j,y} = g^{t_{j,y}} = g^{t'_{j,y}} B^{\omega_j}$
$K_y = g^{\alpha'} A^{t'_{1,y}} B^{t_y}$
For $x \in \mathcal{S}$
if no such i, $\rho^*(i) = x$,
$K_{x,y} = \prod_{j=1}^{n_{max}} L_{j,y}^{z_{x,j}}$
otherwise,
$K_{x,y} = \prod_{j=1}^{n_{max}} g^{z_{x,j} t'_{j,y}} B^{z_{x,j}\omega_j} A^{M^*_{i,j} t'_{j,y}}$
randomly selects $r_{d,y} \in \mathbb{Z}_p$,
$D_y = B^{a_y d + t_y} H(d)^{r_{d,y}}, d_y = g^{r_{d,y}}$
Output $(L_{j,y})_{1\le j\le n_{max}} (K_{x,y})_{x\in\mathcal{S}}, K_y, D_y, d_y$.

Figure 3: KG1(a_y)

For $1 \le j \le n_{max}$, the simulator implicitly sets $t_{j,y} = t'_{j,y} + \omega_j \cdot b$ and then calculate $L_{j,y} = g^{t_{j,y}} = g^{t'_{j,y}} B^{\omega_j}$.

Notice that, $t_{1,y} = t'_{1,y} - b$. Therefore, $g^{at_{1,y}}$ includes the component g^{-ab}. The simulator also sets $\alpha = \alpha' + ab$ and randomly selects $t_y \in \mathbb{Z}_p$, so g^{ab} can be canceled by:

$$K_y = g^{\alpha} g^{at_{1,y}} g^{bt_y} = g^{\alpha'} A^{t'_{1,y}} B^{t_y}$$

If there is no i such that $\rho^*(i) = x \in \mathcal{S}$, we have

$$K_{x,y} = \prod_{j=1}^{n_{max}} L_{j,y}^{z_{x,j}}$$

otherwise, there exists i such that $\rho^*(i) = x \in \mathcal{S}$, we have

$$\begin{aligned} K_{x,y} &= \prod_{j=1}^{n_{max}} (g^{z_{x,j}} g^{aM^*_{i,j}})^{t_{j,y}} \\ &= \prod_{j=1}^{n_{max}} (g^{z_{x,j}} g^{aM^*_{i,j}})^{t'_{j,y}+\omega_j \cdot b} \\ &= \prod_{j=1}^{n_{max}} g^{z_{x,j} t'_{j,y}} B^{z_{x,j}\omega_j} A^{M^*_{i,j} t'_{j,y}} \end{aligned}$$

Finally, it randomly selects $r_{d,y} \in \mathbb{Z}_p$ and calculates:

$$(D_y, d_y) = (B^{a_y d + t_y} H(d)^{r_{d,y}}, g^{r_{d,y}})$$

The procedure **KG1** outputs

$$(L_{j,y})_{1\le j\le n_{max}} (K_{x,y})_{x\in\mathcal{S}}, K_y, D_y, d_y$$

The simulator repeats the above procedure **KG1** with input a_y for all $y \in Path(u_{id})$ and outputs the secret key as

$$\langle \{(L_{j,y})_{1\le j\le n_{max}} (K_{x,y})_{x\in\mathcal{S}}, K_y, D_y, d_y\}_{y\in Path(u_{id})} \rangle$$

[tp = 1] The simulator outputs the secret key for any $(u_{id}, \mathcal{S})$. It separately takes two cases into consideration, i.e. $(u_{id} \in u^*, \mathcal{S} \in (M^*, \rho^*))$, $(u_{id} \notin u^*, \mathcal{S} \notin (M^*, \rho^*))$.

1. If $(u_{id} \in u^*, \mathcal{S} \in (M^*, \rho^*))$, for $y \in Path(u_{id}) \subseteq Path(u^*)$, where a_y is unknown and $l_y = a_y d - a$ is known, the simulator executes procedure **KG2** with input l_y, shown in Figure 4.

   It randomly selects $t'_y, r_{d,y} \in \mathbb{Z}_p$ and implicitly set $t_y = -a + t'_y$, where random number $t'_y \in \mathbb{Z}_p$. Therefore, g^{bt_y} includes the component g^{-ab}. The simulator also

randomly selects $t'_y, r_{d,y} \in \mathbb{Z}_p$, $t_{j,y} \in \mathbb{Z}_p$ for $1 \le j \le n_{max}$.
$K_y = g^{\alpha'} A^{t_{1,y}} B^{t'_y}$
For $1 \le j \le n_{max}$, $L_{j,y} = g^{t_{j,y}}$
For $x \in \mathcal{S}$
if no such i, $\rho^*(i) = x$,
$K_{x,y} = \prod_{j=1}^{n_{max}} L_{j,y}^{z_{x,j}}$
otherwise,
$K_{x,y} = \prod_{j=1}^{n_{max}} (g^{z_{x,j}} A^{M^*_{i,j}})^{t_{j,y}}$
randomly selects $r_{d,y} \in \mathbb{Z}_p$
$D_y = B^{l_y} B^{t'_y} H(d)^{r_{d,y}}, d_y = g^{r_{d,y}}$
Output $(L_{j,y})_{1\le j\le n_{max}} (K_{x,y})_{x\in\mathcal{S}}, K_y, D_y, d_y$.

Figure 4: KG2(l_y)

sets $\alpha = \alpha' + ab$ and randomly selects $t_{j,y} \in \mathbb{Z}_p$ for $1 \le j \le n_{max}$, now g^{ab} can be canceled as

$$K_y = g^{\alpha} g^{at_{1,y}} g^{bt_y} = g^{\alpha'} A^{t_{1,y}} B^{t'_y}$$

For $1 \le j \le n_{max}$, the simulator calculates

$$L_{j,y} = g^{t_{j,y}}$$

If there is no i such that $\rho^*(i) = x \in \mathcal{S}$, we have

$$K_{x,y} = \prod_{j=1}^{n_{max}} L_{j,y}^{z_{x,j}}$$

Otherwise, there exists i such that $\rho^*(i) = x \in \mathcal{S}$, we have

$$\begin{aligned} K_{x,y} &= \prod_{j=1}^{n_{max}} (g^{z_{x,j}} g^{aM^*_{i,j}})^{t_{j,y}} \\ &= \prod_{j=1}^{n_{max}} (g^{z_{x,j}} A^{M^*_{i,j}})^{t_{j,y}} \end{aligned}$$

Finally, it randomly selects $r_{d,y} \in \mathbb{Z}_p$ and calculates:

$$(D_y, d_y) = (B^{l_y} B^{t'_y} H(d)^{r_{d,y}}, g^{r_{d,y}})$$

The procedure **KG2** outputs

$$(L_{j,y})_{1\le j\le n_{max}} (K_{x,y})_{x\in\mathcal{S}}, K_y, D_y, d_y$$

The simulator repeats the above procedure **KG2** with input l_y for all $y \in Path(u_{id})$ and outputs the secret key as

$$\{(L_{j,y})_{1\le j\le n_{max}} (K_{x,y})_{x\in\mathcal{S}}, K_y, D_y, d_y\}_{y\in Path(u_{id})}$$

2. If $(u_{id} \notin u^*, \mathcal{S} \notin (M^*, \rho^*))$, for $y \in Path(u_{id})$,

   (a) If $y \in Path(u^*)$, where l_y is known, the simulator repeatedly executes the procedure **KG2**(l_y) and obtains part of the secret key for $y \in Path(u_{id}) \cap Path(u^*)$

   $$\{(L_{j,y})_{1\le j\le n_{max}} (K_{x,y})_{x\in\mathcal{S}}, K_y, D_y, d_y\}$$

   (b) If $y \notin Path(u^*)$, where a_y is known, the simulator repeatedly executes the procedure **KG1**(a_y) and obtains part of the secret key for $y \in Path(u_{id}) \setminus Path(u^*)$

   $$\{(L_{j,y})_{1\le j\le n_{max}} (K_{x,y})_{x\in\mathcal{S}}, K_y, D_y, d_y\}$$

(c) Finally, it integrates all the parts and outputs:

$$\{(L_{j,y})_{1\le j\le n_{max}}(K_{x,y})_{x\in\mathcal{S}}, K_y, D_y, d_y\}_{y\in Path(u_{id})}$$

$\diamond$ Revoke oracle: $\mathcal{A}$ inputs several revoked unique identifiers u_{id} and a time stamp t, and the simulator adds u_{id} to the revocation list at time t.

$\diamond$ Key update oracle: $\mathcal{A}$ issues queries for key update information with several tuples (rl, t). Depending on the value **tp**, the simulator separates the simulation into two cases.
If $[\mathbf{tp} = \mathbf{0}]$, the simulator outputs key update information for any (rl, t).

In this case, a_y for all the nodes $y \in T$ is known to the simulator. Therefore, the simulator calculates the key update information as follows:

$$\{E_y, e_y\}_{y\in KUN(rl,t)} = \{B^{a_y t}H(t)^{r_{t,y}}, g^{r_{t,y}}\}_{y\in KUN(rl,t)}$$

If $[\mathbf{tp} = \mathbf{1}]$, the simulator calculates key update information for any (rl, t) except the case $(t = t^*, u^* \cap rl \neq \emptyset)$.

1. For the case $t \neq t^*$, revocation list can be any subset of queried unique identifiers.

   (a) For $y \in KUN(rl, t) \cap Path(u^*)$, $l_y = a_y d + a$ is known to the simulator. It randomly selects $r'_y \in \mathbb{Z}_p$ and implicitly sets

   $$r_y = (r'_y + \frac{a}{t^2 + u(t)}) \cdot td^{-1}$$

   Now, we have:

   $$\begin{aligned} E_y &= B^{a_y t}H(t)^{r_y} \\ &= B^{l_y td^{-1} - atd^{-1}}H(t)^{(r'_y + \frac{a}{t^2+u(t)})td^{-1}} \\ &= B^{l_y td^{-1}}(B^{t^2+u(t)}g^{f(t)})^{r'_y}A^{\frac{f(t)}{t^2+u(t)}td^{-1}} \\ e_y &= g^{r_y} \\ &= g^{(r'_y + \frac{a}{t^2+u(t)})td^{-1}} \\ &= g^{r'_y}A^{\frac{td^{-1}}{t^2+u(t)}} \end{aligned}$$

   The simulator calculates

   $$\{E_y, e_y\}_{y\in KUN(rl,t)\cap Path(u^*)}$$

   .

   (b) For $y \in KUN(rl, t) \setminus Path(u^*)$, a_y is known to the simulator. The simulator calculates

   $$\{E_y = B^{a_y t}H(t)^{r_{t,y}}, e_y = g^{r_{t,y}}\}_{y\in KUN(rl,t)\setminus Path(u^*)}$$

   (c) Finally, the simulator integrates all the key update information $\{E_y, e_y\}_{y\in KUN(rl,t)}$

2. For the case $t = t^*$, if $KUN(rl, t) \cap Path(u^*) \neq \emptyset$, the simulator outputs $\perp$; otherwise, the simulator knows a_y for $y \in KUN(rl, t)$. Therefore, the simulator outputs the key update information as follows:

   $$\{E_y, e_y\}_{y\in KUN(rl,t)} = \{B^{a_y t}H(t)^{r_{t,y}}, g^{r_{t,y}}\}_{y\in KUN(rl,t)}$$

CHALLENGE In this phase, the simulator builds the challenge ciphertext. $\mathcal{A}$ generates two messages $\mathcal{M}_0$ and $\mathcal{M}_1$ from the message space of the equal length. Then, the challenger chooses $\mu \in \{0, 1\}$ randomly and outputs the challenge ciphertext $\mathcal{M}_\mu$ with the access structure (M^*, ρ^*) and the time stamp t^* as follows:

1. It implicitly sets $s = c$, we have $C^*_{\mathcal{M}} = \mathcal{M} \cdot e(g, g)^{\alpha s} = \mathcal{M} \cdot e(g, C)^{\alpha'} \cdot Z$ and $C^*_s = C$.

2. It randomly selects $y_j \in \mathbb{Z}_p$ for $2 \le j \le n_{max}$ and implicitly sets $\vec{v} = (s, s + y'_2, s + y'_3, \cdots, s + y'_{n_{max}})$.

3. For $1 \le i \le l$ and $1 \le j \le n_{max}$,

   $$\begin{aligned} C^*_{i,j} &= g^{aM^*_{i,j}v_j}h^{-s}_{j,\rho^*(i)} \\ &= g^{aM^*_{i,j}(s+y'_j)}(g^{z_{\rho^*(i),j}}g^{aM^*_{i,j}})^{-s} \\ &= A^{M^*_{i,j}(y'_j)}C^{-z_{\rho^*(i),j}} \end{aligned}$$

   We also have

   $$\begin{aligned} C^*_d &= H(d)^s \\ &= (B^{d^2+u(d)}g^{f(d)})^s \\ &= C^{f(d)} \end{aligned}$$

   and

   $$\begin{aligned} C^*_{t^*} &= H(t^*)^s \\ &= (B^{t^{*2}+u(t^*)}g^{f(t^*)})^s \\ &= C^{f(t^*)} \end{aligned}$$

Finally, it outputs the challenged ciphertext

$$\mathcal{C}^* = \langle C^*_{\mathcal{M}}, C^*_s, (C^*_{i,j})_{1\le i\le l, 1\le j\le n_{max}}, C^*_d, C^*_{t^*}\rangle$$

PHASE 2 The simulator acts exactly the same as in **PHASE 1**

GUESS The adversary will output a guess μ' of μ. If any abort happens, the simulator outputs 0; otherwise, it then outputs 1 to guess $Z = e(g, g)^{abc}$ if $\mu' = \mu$ and outputs 0 to guess that Z is a random group element in $\mathbb{G}_T$ if $\mu' \neq \mu$.

Probability Analysis.
Denote **Exp** as the experiment describe above, "abort" as the case of $\perp$ output by in **Exp**, "real" as the case that input of **Exp** is a tuple selected from D_{bdh}, "rand" as the case that input of **Exp** is a tuple selected from D_{rand}. We assume $\Pr[\mathbf{tp} = 0] = \Pr[\mathbf{tp} = 1] = \frac{1}{2}$.
From the **GUESS** phase, we have the follows:

1. If $\nu = 1$, the challenger sets $(g, A, B, C, Z) \in D_{bdh}$, the simulator gives a perfect simulation if no "abort" happens,

   $$Adv_{\mathcal{A}} = \Pr[\mathbf{Exp} = 1|\text{real} \wedge \overline{\text{abort}}] - \frac{1}{2}$$

2. If $\nu = 0$, the challenger sets $(g, A, B, C, Z) \in D_{rand}$, so $\mathcal{M}_\mu$ is completely hidden from the adversary,

   $$\Pr[\mathbf{Exp} = 1|\text{rand} \wedge \overline{\text{abort}}] = \frac{1}{2}$$

Notice that the probability for "abort" cases depends on the behavior of adversary and **tp** which are both irrelevant with the tuple chosen from D_{bdh} or D_{rand}. Therefore, we have:

$$\Pr[\mathbf{Exp} = 1|\text{real} \wedge \text{abort}] = 0$$

$$\Pr[\mathbf{Exp} = 1|\text{rand} \wedge \text{abort}] = 0$$

From the key generation oracle, when $\mathbf{tp} = 0 \wedge u_{id} \in u^* \wedge \mathcal{S} \in (M^*, \rho^*)$, **Exp** outputs $\perp$. When $\mathbf{tp} = 1 \wedge t = t^* \wedge (u^* \setminus rl \neq \emptyset)$, **Exp** outputs $\perp$.
According to the restrictions mentioned in Section 2.3, the $u_{id} \in u^*$ is only allowed to be queried when u^* is added to the revocation list rl at time t^*.

Therefore,

$$\Pr[u_{id} \in u^*] \leq \Pr[u^* \setminus rl = \emptyset | t = t^*]$$

$$\begin{aligned}
\Pr[\text{abort}] &= \Pr[\mathbf{tp} = 0 \wedge u_{id} \in u^*] \\
&\quad + \Pr[\mathbf{tp} = 1 \wedge u^* \setminus rl \neq \emptyset | t = t^*] \\
&= \tfrac{1}{2}\Pr[u_{id} \in u^*] + \tfrac{1}{2}\Pr[u^* \setminus rl \neq \emptyset | t = t^*] \\
&\leq \tfrac{1}{2}\Pr[u_{id} \in u^*] + \tfrac{1}{2}(1 - \Pr[u_{id} \in u^*]) \\
&= \tfrac{1}{2}
\end{aligned}$$

Thus, $\Pr[\overline{\text{abort}}] \geq \frac{1}{2}$.

$$\begin{aligned}
Adv_{\mathcal{C}} &= \Pr[\mathbf{Exp} = 1|\text{real}] - \Pr[\mathbf{Exp} = 1|\text{rand}] \\
&= \Pr[\text{abort}] \cdot \Pr[\mathbf{Exp} = 1|\text{real} \wedge \text{abort}] \\
&\quad + \Pr[\overline{\text{abort}}] \cdot \Pr[\mathbf{Exp} = 1|\text{rand} \wedge \overline{\text{abort}}] \\
&\quad - \Pr[\text{abort}] \cdot \Pr[\mathbf{Exp} = 1|\text{real} \wedge \text{abort}] \\
&\quad - \Pr[\overline{\text{abort}}] \cdot \Pr[\mathbf{Exp} = 1|\text{rand} \wedge \overline{\text{abort}}] \\
&= \Pr[\overline{\text{abort}}] \cdot (\Pr[\mathbf{Exp} = 1|\text{rand} \wedge \overline{\text{abort}}] \\
&\quad - \Pr[\mathbf{Exp} = 1|\text{rand} \wedge \overline{\text{abort}}]) \\
&\geq \tfrac{1}{2} \cdot Adv_{\mathcal{A}}
\end{aligned}$$

□

5. DISCUSSION

5.1 Efficiency

In the proposed CP-ABE-R scheme, the size of each user secret key is $(n_{max} + |\mathcal{S}| + 3) \cdot \log n$ group elements, where n_{max} is the maximum size of columns in the access structure (M, ρ), $\mathcal{S}$ is the attribute set corresponding to the user secret key and n is the total user number. In comparison with the traditional ciphertext policy attribute based encryption, the size of user secret key is increased by multiplying $\log n$. The $\log n$ factor here is the number of the nodes on the path from the root of the revocation tree to the leaf node representing each user. Intuitionally, since users are associated with different unique paths in the revocation tree, it is necessary for such an expansion to uniform the difference. Once the non-revoked user extracts the useful update information with available node, it will use one piece of secret key related to this node for decrypting the ciphertext. Thus, the computational cost of encryption and decryption algorithms are acceptable.

Waters [21] presents several variants of ciphertext policy attribute based encryption by finding tradeoff between efficiency and security assumption. Besides the one based on DBDH assumption, the rest constructions can be also extended to the one in CP-ABE-R model with improved efficiency by adopting the same technique introduced in this paper, however the complexity assumption becomes more complicated.

In Table 1, we give the comparisons between Waters' scheme [21] and CP-ABE-R scheme in terms of public parameters size (PP), secret key size (SK), ciphertext size (CT), encryption time (EN), decryption time (DE) and revocation (RE). Denote $\mathbf{T_{exp}}, \mathbf{T_{pair}}$ as the time for one modular exponentiation and one bilinear pairing computation, respectively.

5.2 Delegating capability and Chosen Ciphertext Security

To derive the delegating capability in traditional CP-ABE scheme is straightforward since user secret keys related with fuzzy identities are not personalized according to different users. Therefore, as long as the secret keys can be re-randomized, users are with the delegating capability to enroll new valid users whose secret key is associated with the same attribute set. Furthermore, the users can delegate part of his access ability to the others, e.g., the professor with attribute set "Professor" and "Computer Science" may delegate a users with the access ability corresponding to a subset "Professor". In the CP-ABE-R model, we implicitly assign each user with a unique identifier which can be represented as a unique path in the revocation tree. Though each components of user secret keys can still be re-randomized, user's unique identifier does not change, i.e., the delegatees' access abilities would be revoked once the original delegator's identifier is not involved in the update information. Notice that the delegated secret keys are available during the simulation, the security proof of the CP-ABE-R scheme with delegation algorithm is essentially unaffected. In addition, chosen ciphertext security can be realized in the standard model by using the technique of one-time signature [7]. Some similar conversion methods can be found in the paper [3, 8].

	Waters[21]	CP-ABE-R
PP	$(n_{max}\lvert\mathcal{U}\rvert + 2) \times \lvert\mathbb{G}\rvert$ $+\lvert\mathbb{G}_T\rvert$	$(n_{max}\lvert\mathcal{U}\rvert + 6) \times \lvert\mathbb{G}\rvert$ $+\lvert\mathbb{G}_T\rvert + \lvert\mathbb{Z}_p\rvert$
SK	$(n_{max} + \lvert\mathcal{S}\rvert + 1) \times \lvert\mathbb{G}\rvert$	$(n_{max} + \lvert\mathcal{S}\rvert + 3) \times \lvert\mathbb{G}\rvert \times \log n$
CT	$(ln_{max} + 1) \times \lvert\mathbb{G}\rvert + \lvert\mathbb{G}_T\rvert$	$(ln_{max} + 3) \times \lvert\mathbb{G}\rvert + \lvert\mathbb{G}_T\rvert$
EN	$(ln_{max} + 2) \times \mathbf{T_{exp}}$	$(ln_{max} + 4) \times \mathbf{T_{exp}}$
DE	$(n_{max} + 2) \times \mathbf{T_{pair}}+$ $(n_{max}\lvert\mathcal{I}\rvert + \lvert\mathcal{I}\rvert) \times \mathbf{T_{exp}}$	$(n_{max} + 6) \times \mathbf{T_{pair}}+$ $(n_{max}\lvert\mathcal{I}\rvert + \lvert\mathcal{I}\rvert + 2) \times \mathbf{T_{exp}}$
RE	No	Yes

Table 1: Comparison of Schemes

6. CONCLUSION

In this paper, we firstly studied the feasible revocation operations in CP-ABE scheme: single attribute revocation, attribute set revocation and unique identifier revocation. Then, based on unique identifier revocation technique, we proposed the CP-ABE-R scheme in which malicious users can be efficiently revoked. We presented the ciphertext policy attribute based encryption scheme with efficient revocation by using linear secret sharing scheme and binary tree technique as the underlying tools. We have shown that the delegating capability can be easily provided in the proposed scheme, but all the delegatees are associated with their original delegator's unique identifier.

For our future work, we will improve the efficiency of CP-ABE-R scheme, such as shortening the size of user secret key, reducing the amount of published update information, and developing faster encryption/decryption algorithms.

7. REFERENCES

[1] Beimel, A. Secure schemes for secret sharing and key distribution. In *PhD thesis* (1996).

[2] Bethencourt, J., Sahai, A., and Waters, B. Ciphertext-policy attribute-based encryption. In *IEEE Symposium on Security and Privacy* (2007), pp. 321–334.

[3] Boldyreva, A., Goyal, V., and Kumar, V. Identity-based encryption with efficient revocation. In *ACM Conference on Computer and Communications Security* (2008), pp. 417–426.

[4] Boneh, D., Ding, X., Tsudik, G., and Wong, M. A method for fast revocation of public key certificates and security capabilities. In *USENIX Security Symposium* (2001), pp. 22–22.

[5] Boneh, D., and Franklin, M. K. Identity-based encryption from the weil pairing. In *CRYPTO* (2001), pp. 213–229.

[6] Boneh, D., and Waters, B. A fully collusion resistant broadcast, trace, and revoke system. In *ACM Conference on Computer and Communications Security* (2006), pp. 211–220.

[7] Canetti, R., Halevi, S., and Katz, J. Chosen-ciphertext security from identity-based encryption. In *EUROCRYPT* (2004), pp. 207–222.

[8] Cheung, L., and Newport, C. Provably secure ciphertext policy abe. In *ACM Conference on Computer and Communications Security* (2007), pp. 456–465.

[9] Gentry, C. Certificate-based encryption and the certificate revocation problem. In *EUROCRYPT* (2003), pp. 272–293.

[10] Goyal, V. Certificate revocation using fine grained certificate space partitioning. In *Financial Cryptography* (2007), pp. 247–259.

[11] Goyal, V., Jain, A., Pandey, O., and Sahai, A. Bounded ciphertext policy attribute based encryption. In *ICALP (2)* (2008), pp. 579–591.

[12] Goyal, V., Pandey, O., Sahai, A., and Waters, B. Attribute-based encryption for fine-grained access control of encrypted data. In *ACM Conference on Computer and Communications Security* (2006), pp. 89–98.

[13] Lewko, A., Sahai, A., and Waters, B. Revocation systems with very small private keys. In *Cryptology ePrint Archive: Report 2008/309* (2008).

[14] Liang, X., Cao, Z., Lin, H., and Xing, D. Provably secure and efficient bounded ciphertext policy attribute based encryption. In *ASIACCS* (2009), pp. 343–352.

[15] Libert, B., and Quisquater, J.-J. Efficient revocation and threshold pairing based cryptosystems. In *PODC* (2003), pp. 163–171.

[16] Maji, H., Prabhakaran, M., and Rosulek, M. Attribute-based signatures: Achieving attribute-privacy and collusion-resistance. In *Cryptology ePrint Archive: Report 2008/328* (2008).

[17] Naor, D., Naor, M., and Lotspiech, J. Revocation and tracing schemes for stateless receivers. In *CRYPTO* (2001), pp. 41–62.

[18] Naor, M., and Nissim, K. Certificate revocation and certificate update. In *USENIX Security Symposium* (1998).

[19] Naor, M., and Pinkas, B. Efficient trace and revoke schemes. In *Financial Cryptography* (2000), pp. 1–20.

[20] Sahai, A., and Waters, B. Fuzzy identity-based encryption. In *EUROCRYPT* (2005), pp. 457–473.

[21] Waters, B. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In *Cryptology ePrint Archive: Report 2008/290* (2008).

[22] Yu, S., Ren, K., and Lou, W. Fdac: Toward fine-grained distributed data access control in wireless sensor networks. In *IEEE INFOCOM* (2009).