OUTSOURCED KP-ABE WITH CHOSEN-CIPHERTEXT SECURITY

Chao Li, Bo Lang and Jinmiao Wang

State Key Laboratory of Software Development Environment,
Beihang University, Beijing, China
lichao@nlsde.buaa.edu.cn
langbo@buaa.edu.cn
wangjinmiao@nlsde.buaa.edu.cn

ABSTRACT

Key-Policy Attribute Based Encryption (KP-ABE) has always been criticized for its inefficiency drawbacks. Based on the cloud computing technology, computation outsourcing is one of the effective solution to this problem. Some papers have proposed their schemes; however, adversaries in their attack models were divided into two categories and they are assumed not to communicate with each other, which is obviously unrealistic. In this paper, we first proved there exist severe security vulnerabilities in these schemes for such an assumption, and then proposed a security enhanced Chosen Ciphertext Attack (SE-CCA) model, which eliminates the improper limitations. By utilizing Proxy Re-Encryption (PRE) and one-time signature technology, we also constructed a concrete KP-ABE outsourcing scheme (O-KP-ABE) and proved its security under SE-CCA model. Comparisons with existing schemes show that our constructions have obvious comprehensive advantages in security and efficiency.

KEYWORDS

KP-ABE, computation outsourcing, CCA, security model, proxy re-encryption, one-time signature

1. INTRODUCTION

With the development of Internet, the data storing service of third-party is increasingly popular. However, the data, in such a case, will be out of its owner's control and is managed by Cloud Storage Providers (CSPs). Then the confidentiality of the data becomes a problem. At present, encryption is the primary mechanism to implement data protection. However, traditional encryption schemes are not suitable in such a situation for their lack of ability in access control and huge overhead of key management. Sahai and Waters [1] addressed this issue by introducing the notion of attribute-based encryption (ABE), a new kind of public key based one-to-many encryption scheme which can achieve fine-grained access control on ciphertexts. In such a cryptosystem, private keys and ciphertexts are associated with an attribute group or an access policy respectively. A user is able to decrypt a ciphertext if and only if the attribute group satisfies the access policy. ABE can be classified as KP-ABE [2] and Ciphertext-Policy Attribute Based Encryption (CP-ABE) [3]-[5]. In KP-ABE, user's private key is associated with an access policy and ciphertexts are associated with a group of attributes. CP-ABE is just the opposite. They are suitable for different application scenarios. The former is data-centred with data attributes; the latter is user-centred with user attributes.

Natarajan Meghanathan et al. (Eds) : WiMONe, NCS, SPM, CSEIT - 2014
pp. 147–160, 2014. © CS & IT-CSCP 2014 DOI : 10.5121/csit.2014.41211

Although ABE is promising in implementing fine-grained access control on ciphertexts, it is being criticized for its inefficiency, which is first reflected in the process of decryption. The decryption of ABE is based on time-consuming bilinear pairings of which the number is in proportion to the complexity of the access policy. While conventional desktop computers would be able to handle such a task, it presents a significant challenge for users that manage and view private data on mobile devices. The inefficiency is also reflected in key-issuing. In applications that use ABE, the user groups are dynamically changed and the attribute universes of the user are usually very large. Hence, the heavy work of key generating during the initialization and user revocation will make Public Key Generator (PKG) a bo/ttleneck of the whole system. As the key is bundled with the policy in KP-ABE, more tasks will be needed in key generating a key and it will be more difficult to recognize the users affected by one revocation. Thus the inefficiency problem of key-issuing is more serious to KP-ABE.

The main solution to solve the problems above is outsourcing, by which we can outsource the heavy computation tasks to a third party who has strong computing power. And, the rise of cloud computing has provided techniques and application fundamentals for this. Cloud Service Providers (CSPs) could provide users the pay-on-demand computing services, such as Amazon's EC2 and Microsoft's Windows Azure. Based on this idea, Green et al. [6] firstly proposed the concrete ABE outsourcing schemes. In these schemes, all of the pairings in decryption are outsourced to a third party and cleartexts can be recovered by a simple ElGamal decryption without leaking any useful information of cleartexts and user private keys. We henceforth refer to this paper as Green11 [6].

However, Green11 cannot solve the inefficiency problem of key-issuing. Li et al. [7] extended the outsourcing idea to key-issuing of KP-ABE and proposed a new scheme model called Outsourced ABE (OABE). In OABE, there are three CSPs: S-CSP, D-CSP and KG-CSP, and they provide services of ciphertexts storing, decrypting and key generating respectively. We henceforth refer to this paper as LCLJ13. LCLJ13 can outsource both decryption and key-issuing. However, two pairings are still needed in the decryption phase. The authors of another paper [8] whose target is the checkability of outsourced results have improved the decryption efficiency by adopting the key-blinding technique. But the key-blinding work is done by PKG, thus it hasn't eased the PKG's burden compared to the traditional non-outsourcing KP-ABE. We refer to this paper as LHLC13.

Although the aforementioned schemes have alleviated the inefficiency problem of ABE to some degree, all of them fall short on the security, especially for LCLJ13 and LHLC13. Green11 can resist the Replayable Chosen Ciphertext Attack (RCCA) [9], which lies between Chosen Plaintext Attack and Chosen Ciphertext Attack. And we will explain RCCA in more detail in Section 6. LCLJ13 and LHLC13 share the same attack model, in which the adversaries are classified into two types: a curious user colluding with D-CSP and a curious KG-CSP. And they both assumed that the two types of adversaries cannot collude. However, in the real OABE system, all CSPs cannot be completely trusted. Thus, such an assumption is unrealistic. Actually, in their schemes, a curious user can decrypt any ciphertext by colluding with KG-CSP. We will give the proof in Section 4.1. Therefore, it is very important to construct a more secure KP-ABE outsourcing scheme which can outsource both key-issuing and decryption.

Our contributions. This paper focus on the KP-ABE outsourcing scheme which can outsource both key-issuing and decryption. Although LCLJ13 and LHLC13 can do this, they still have much space in improving efficiency and security, especially the security. We will firstly prove the security vulnerabilities in LCLJ13 and LHLC13 for their assumption of no collusion between curious users and KP-CSP. We further propose a security enhanced CCA (SE-CCA) model based on the analysis of environment with Chosen Ciphertext Attack, all CSPs are suspect and curious users may collude with any CSP. Then by utilizing the technique of PRE and one-time signature,

we construct a new concrete KP-ABE outsourcing scheme (O-KP-ABE) with proved security under SE-CCA.

Organization. The rest of the paper is organized as follows. In Section 2, we introduce the related work. Next, we give necessary background information in Section 3. In Section 4, we first give a detailed security analysis of existing schemes and then we describe our new KP-ABE outsourcing model and security enhanced CCA (SE-CCA) model. We present a concrete construction of a new KP-ABE outsourcing scheme (O-KP-ABE) and prove its security under SE-CCA in Section 5. In Section 6, we compare our scheme with all relative schemes in security and efficiency and analyse the results. Finally, we conclude our work.

2. RELATED WORK

ABE Outsourcing. This idea was firstly proposed by Green et al. [6] in their work. They have also constructed concrete schemes which can outsource the decryption based on this idea. Later, Zhou et al. [10] proposed a different ABE scheme with outsourced encryption and decryption. Zhou11 and Green11 both utilized the key-blinding technique to outsource decryption, in which the user firstly chooses a value randomly as the blind factor, and then runs exponentiations on the original key components with the blind factor. However, both of them haven't considered the computation overhead at PKG. Li et al. [7] firstly constructed a KP-ABE outsourcing scheme which can outsource both key-issuing and decryption. And the technique they adopted was different, of which the core idea is using a default attribute. This default attribute will be appended to each data's attribute group and each user's access policy. Besides, there are also papers [8], [11] researching on the verification of outsourcing results. And the main means is to append a redundancy to the ciphertext.

Proxy Re-Encryption (PRE). The notion of PRE was firstly proposed by Blaze et al. [12]. They also constructed a simple concrete scheme. PRE can be represented by the formula $D(\prod(E(m,e_A),\pi_{A\to B}),d_B)=m$, which means the ciphertext encrypted by A's public key e_A after being re-encrypted by proxy key $\pi_{A\to B}$ can be decrypted by B's secret key d_B. $\pi_{A\to B}$ is public and the re-encryption work can be done by an untrusted proxy server without fearing the leakage of the message *m*, and user secret keys d_A, d_B. Then Ateniese et al. [13] have made a further research on PRE, and they have concluded the features of a PRE scheme. Besides, they have also put forward an improved PRE scheme. There are also some papers [14] make research on the more secure PRE schemes.

One-Time Signature. There are several techniques to construct a CCA secure scheme from a CPA secure one. One-time signature is one of them. Canetti et al. [15] proposed this technique and utilized it to construct a CCA secure public key encryption scheme based on Identity-Based Encryption (IBE) [15]. Then Cheung and Newport [16] applied the similar technique to CP-ABE and constructed a CCA secure CP-ABE scheme from the CPA secure one. One-time signature contains a pair of keys (*sk, vk*) in which the former is used for signing and the latter is used for verifying and its length in binary bits is constant. In the scheme of Cheung et al., every bit in *vk* is defined as an attribute, thus there are two attributes corresponding to each bit. Each user secret key contains two components for both occurrences of each bit except for ABE components. For encryption, the encryptor chooses a pair (*sk, vk*) and encrypts the message with *vk* in addition to other attributes. The whole ciphertext is then signed with *sk*. And the decryptor will first verify the signature before decryption. We will take advantage of this technique to construct the scheme of O-KP-ABE and prove its CCA security under SE-CCA model.

3. PRELIMINARIES

3.1. Bilinear Maps

Let G and G_T be two multiplicative cyclic groups of prime order p and g is a generator of G. $e: G \times G \to G_T$ is a bilinear map with the properties:

- Bilinearity: for all $u, v \in G$ and $a, b \in Z_p$, we have $e(u^a, v^b) = e(u, v)^{ab}$.
- Non-degeneracy: $e(g, g) \neq 1$.

We say that G is a bilinear group if the group operation in G and the bilinear map $e: G \times G \to G_T$ are both efficiently computable. Notice that the map e is symmetric since $e(u^a, v^b) = e(u, v)^{ab} = e(u^b, v^a)$.

3.2. DBDH Assumption

We define the decisional Bilinear Diffie-Hellman problem as follows. A challenger chooses a group G of prime order p according to the security parameter. Let $a, b, c \in Z_p$ be chosen at random and g be a generator of G. Given (g, g^a, g^b, g^c), the adversary must distinguish a valid tuple $e(g, g)^{abc} \in G_T$ from a random element R in G_T.

Then we can get the definition of DBDH assumption:

Definition 1 (DBDH Assumption) *We say that the DBDH assumption holds if no polytime algorithm has a non-negligible advantage in solving the DBDH problem.*

3.3. Access Structure

Definition 2 (Access Structure [17]) *Let* $\{P_1, P_2, ..., P_n\}$ *be a set of parties. A collection* $A \subseteq 2^{\{P_1, P_2, ..., P_n\}}$ *is monotone if* $\forall B, C$ *if* $B \in A$ *and* $B \subseteq C$ *then* $C \in A$. *An access structure (respectively, monotone access structure) is a collection (respectively, monotone collection)* A *of non-empty subsets of* $\{P_1, P_2, ..., P_n\}$, *i.e.,* $A \subseteq 2^{\{P_1, P_2, ..., P_n\}} \setminus \varnothing$. *The sets in* A *are called the authorized sets, and the sets not in* A *are called the unauthorized sets.*

In the context of ABE, the role of the parties is taken by the attributes. Thus, the access structure A will contain the authorized sets of attributes.

4. NEW MODELS FOR KP-ABE WITH OUTSOURCING

4.1. Security Analysis of Existing Schemes

We will take LCLJ13 for example to prove it in this section, that is, if curious users collude with KG-CSP, they can decrypt any ciphertext CT. LHLC13 suffers from the same problem. The full proof is shown as follows.

Assume a curious user will collude with KG-CSP, and his secret key is SK = (SK_1, SK_2), in which $SK_2 = (d_{\theta 0} = g_2^{x_2} (g_1 h)^{r_\theta}, d_{\theta 1} = g^{r_\theta})$. As KG-CSP computes all delegated key-issuing work,

it may store the copies of all OKs, including the curious user's, and assume it is OK = x_1. Given a ciphertext CT= $(C_0 = m.e(g_1, g_2)^s, C_1 = g^s, E_\theta = (g_1 h)^s)$, the curious user performs the following steps:

(1) With public parameter g_2, OK and C_1 in CT, he can calculate $e(g, g_2)^{x_1 s}$; with SK2 and E_θ in CT, he can calculate $e(C_1, d_{\theta 0}) / e(d_{\theta 1}, E_\theta) = e(g, g_2)^{x_2 s}$.

(2) As the master key is $x = x_1 + x_2$, he is able to get $e(g, g_2)^{xs}$ through calculation.

(3) As $C_0 = m.e(g, g_2)^{xs}$, the curious user can recover *m*.

As CT in the above process can be any ciphertext, the curious user can decrypt all ciphertexts by colluding with KG-CSP. This is obviously incorrect. Thus there exist severe security vulnerabilities in LCLJ13 and LHLC13.

4.2. Model of KP-ABE with Outsourcing

In this section, we give our KP-ABE outsourcing model by modifying the model of KP-ABE with outsourced decryption in Green11. Our model supports the outsourcing of decryption and key-issuing simultaneously. The model is similar to LCLJ13, but it needs no subsequent processing of outsourced key-issuing, i.e. the PKG in our model need not do any further computations after receiving TK from KG-CSP. By distinguishing OK and TK, on one hand the user could decrypt the ciphertext himself when the network is unavailable; on the other hand the D-CSP need not to generate TK from OK whenever it translates ciphertexts. Our new KP-ABE outsourcing scheme consists of six algorithms, rather than seven in LCLJ13.

Setup. The setup algorithm takes no input other than the implicit security parameter. It is used to initialize the system and output the public parameter PK and master key MK. This algorithm is performed by PKG.

Encrypt (PK, M, S). The encryption algorithm takes as input the public parameters PK, a message M and a set of attributes S. It will encrypt M and produce a ciphertext CT. This algorithm is performed by Data Owner (DO).

Keygen_IN (A , MK, PK). This algorithm is the first step of key-issuing and is performed by PKG. It takes as input the access structure A , the master key MK and the public parameters PK. It outputs the outsourcing key OK and user private key SK.

Keygen_OUT (OK, PK). This algorithm is the second step of key-issuing and is performed by KG-CSP. It takes as input the outsourcing key OK and public parameters PK. It will output the transformation key TK and return it to PKG.

Transform_OUT (TK, CT). This algorithm completes the preprocessing of ciphertext and is performed by D-CSP. It firstly checks whether the attribute set S in CT satisfies the access structure A in TK. It outputs the partially decrypted ciphertext CT' if $S \in A$ otherwise it outputs $\perp$.

Decrypt (CT, CT', SK). This algorithm takes the ciphertext CT, partially decrypted ciphertext CT', and user private key SK as input. It outputs the message M if $S \in A$, otherwise $\perp$.

4.3. Enhanced Security Model

This section analyzes all possible attacks to the KP-ABE outsourcing model given in section 4.2 under CCA and proposes a new Security Enhanced CCA model SE-CCA.

As the above outsourcing model outsources the majority of work during key-issuing and decrypting to a third party who is not completely trusted, more information may be leaked. Even though the computations of key-issuing and decryption are outsourced to different parties, they may collude with each other. Thus, the outsourcing model above will face attacks different from any previous ones, which results in a different attack model.

Through careful analysis of the new outsourcing model, we find attackers under CCA may get the following information or services:

- Like the basic ABE schemes, the attacker is able to achieve the service of key-issuing, and thus get the key pair (SK, TK) corresponding to the specific access policy A.
- Since KG-CSP is not trusted, it may save the copies of all OKs sent from PKG and the corresponding TKs. So the attacker may get all of the key pairs (OK, TK).
- Combining the above two points, the attacker can get the tuple of keys (OK, TK, SK) corresponding to the specific policy A.
- As S-CSP and D-CSP are both untrusted and the TK corresponding to any access policy A can be achieved, the attacker is able to get the transforming service to all ciphertexts.
- The adversary can get specific decrypting services under CCA model.

Based on these observations, we propose the new CCA model SE-CCA, and the model is defined as follows:

Init. The adversary A declares the set of attributes S^* and submits it to the challenger C.
Setup. The challenger C runs the **Setup** algorithm of KP-ABE outsourcing scheme and sends the public parameters PK to adversary A.

Phase 1. The adversary A is allowed to make any of the following queries repeatedly:

i. Query for (SK, OK, TK) corresponding to the access structure A with the restriction that for all $x \in Y_A$, $x \notin S^*$, in which Y_A is the collection of the attributes in A.
ii. Query for (OK, TK) corresponding to the access structure A, with the restriction that for all $x \in Y_A$, $x \in S^*$, in which Y_A is the collection of the attributes in A.
iii. Query for the transforming ciphertext CT' corresponding to CT encrypted with S^*.
iv. A submits a ciphertext CT encrypted with S^* and gets the corresponding message m.

Challenge. A sends to C two equal length messages m_0, m_1. Then C flips a random coin b, and encrypts m_b with S^*. The ciphertext CT^* will be sent to A.

Phase 2. The adversary repeats **Phase 1** with the restriction that the ciphertexts submitted for decryption are not equal to CT^*.
Guess. A outputs a guess b' of b.

Definition 3 (SE-CCA Secure KP-ABE with Outsourcing) *An KP-ABE outsourcing scheme is SE-CCA secure if all polynomial time adversaries have at most a negligible advantage in the game of SE-CCA.*

5. O-KP-ABE

5.1. Access Trees

Our construction uses the tree-based access structure which is represented by *T*. Each interior node of the tree is a threshold gate and the leaves are associated with attributes. This structure is very expressive. For example, we can represent a tree with "AND" and "OR" gates by using respectively 2 of 2 and 1 of 2 threshold gates. A user is able to decrypt a ciphertext if and only if the attributes in ciphertext satisfies the access structure in the user's private key. The definitions of *T* and the relative functions are identical to paper [2].

5.2. Construction of O-KP-ABE

Let G_1 be a bilinear group of prime order p and let g be a generator of G_1. In addition, let $e: G_1 \times G_1 \to G_T$ denote the bilinear map. We also define the Lagrange coefficient $D_{i,S}$ for $i \in Z_p$ and a subset, S, of Z_p: $D_{i,S}(x) = \prod_{j \in S, j \neq i} \frac{x-j}{i-j}$. Assume the length of *vk* in one-time signature is *l* and vk_j is the value of j_{th} bit in *vk*.

Our construction consists of 6 algorithms.

Setup. First, choose a bilinear group G_1 of prime order p with a generator g and a bilinear map $e: G_1 \times G_1 \to G_T$. Next, determine the universe of attributes according to the actual situation $U = \{a_1, a_2, ..., a_n\}$, and let i represents the index of attribute a_i in *U*. Then, choose $\alpha, t_i, \beta_i, \omega, u_j \in Z_p, 1 \le i \le n, 1 \le j \le 2l$ and $g_2 \in G_1$, in which t_i and β_i correspond to a_i. Then the master key is $MK = (\alpha, \omega, t_i, \beta_i, u_j), 1 \le i \le n, 1 \le j \le 2l$, and the public parameters PK are $PK = \{U, g, g_1 = g^{\alpha}, g_2, T_i = g^{t_i}, P_i = g_2^{t_i^{-1}\beta_i}, U_j = g^{u_j}\}, 1 \le i \le n, 1 \le j \le 2l$.

Encrypt (PK, M, S). Run the key generating algorithm of one-time-signature to get a pair of key (*sk, vk*) and randomly choose a value $s \in Z_p$. For j_{th} bit in *vk*, if $vk_j = 0$, calculate $E_j = U_j^s$; if $vk_j = 1$, calculate $E_j = U_{j+l}^{\ s}$. Thus, we can get $C = \{S, C_0 = M.e(g_1, g_2)^s, \{C_y = T_i^s\}_{y \in S}, E\}$, in which $E = \{E_j\}, 1 \le j \le l$, *y* represents an attribute and *i* is the index of *y* in the universe of attributes U.

Then sign on *C* with *sk* and obtain a signature σ. The final ciphertext is $CT = (C, \sigma, vk)$.

Keygen_IN (T, MK, PK). The algorithm proceeds as follows. First choose a random value $z \in Z_p$ and calculate $\delta = (\alpha - \omega)/z$. Then, choose a polynomial q_x for each node x (including the leaves) in the tree T. These polynomials are chosen in the following way in a top-down manner, starting from the root node *r*.

For each node x in the tree, set the degree η_x of the polynomial q_x to be one less than the threshold value k_x of that node, that is $\eta_x = k_x - 1$. Then, for the root node *r*, set $q_r(0) = \delta$ and η_r other points of the polynomial q_r randomly to define it completely. For any other node x, set $q_x(0) = q_{parent(x)}(index(x))$ and choose η_x other points randomly to completely define q_x.

Once the polynomials have been decided, for each leaf node *x*, we can get the value of $q_x(0)$, and then calculate $d_x = q_x(0) / \beta_i$, in which *i* is the index of *x* in the universe of attributes.

Then, randomly choose $\omega_j \in Z_p, 1 \le j \le l-1$, and calculate $\omega_l = \omega - \sum_{j=1}^{l-1} \omega_j$. Afterwards, calculate $\varphi_j = \omega_j / u_j$, $\xi_j = \omega_j / u_{j+l}$, $1 \le j \le l$. Thus, we have $G = (G_{j0} = g_2^{\varphi_j}, G_{j1} = g_2^{\xi_j})$.

Finally, user's private key is SK = *z* and outsourcing key is $OK = \{T, \{d_x\}_{x \in Y_T}, G\}$.

Keygen_OUT (OK, PK). For each element dx in OK calculate $D_x = P_i^{d_x} = g_2^{t_i^{-1} \cdot q_x(0)}$, in which *i* is the index of attribute *x* in the universe of attributes U. The transformation key is:

$TK = \{T, \{D_x\}_{x \in Y_T}, \mathrm{G}\}$, in which Y_T is the attributes set of leaves in T.

Transform_OUT (TK, CT). First verify the signature σ. On failure, return $\perp$. Otherwise, proceed the transformation procedure which is defined as a recursive algorithm TransformNode (*x*, TK, CT). This recursive algorithm outputs a group element of G_T or $\perp$.

If the node *x* is a leaf node then:

$$TransformNode(x, TK, CT) = \begin{cases} e(D_x, C_x) = e(g_2^{t_i^{-1} \cdot q_x(0)}, g^{t_i \cdot s}) \\ = e(g, g_2)^{q_x(0).s} \quad \text{if } x \in S \\ \\ \perp \quad \text{otherwise} \end{cases}$$

If *x* is not a leaf node, the algorithm TransformNode (*x*, TK, CT) proceeds as follows: for all nodes *z* that are children of *x*, it calls TransformNode (*z*, TK, CT) and stores the output as F_z. Let S_x be an arbitrary k_x sized set of child nodes *z* such that $F_z \neq \perp$. If no such set exists then the node was not satisfied and the function returns $\perp$.

Otherwise, we compute:

$$\begin{aligned} & TransformNode(x, TK, CT) \\ &= \prod_{z \in S_x} F_z^{\mathrm{D}_{i.S_x'}(0)}, \quad \text{in which } \begin{matrix} i = index(z) \\ S_x' = \{index(z) : z \in S_x\} \end{matrix} \\ &= \prod_{z \in S_x} (e(g, g_2)^{s.q_{z(0)}})^{\mathrm{D}_{i.S_x'}(0)} \\ &= \prod_{z \in S_x} (e(g, g_2)^{s.q_{parent(z)}(index(z))})^{\mathrm{D}_{i.S_x'}(0)} \\ &= \prod_{z \in S_x} e(g, g_2)^{s.q_x(i).\mathrm{D}_{i.S_x'}(0)} \\ &= e(g, g_2)^{s.q_x(0)} \end{aligned}$$

If CT cannot satisfy TK, the algorithm returns $\perp$, otherwise we can get $CT_0' = e(g, g_2)^{s\delta}$. Then, for j_{th} bit in *vk*, if $vk_j = 0$, calculate $e(E_j, G_{j0}) = e(g^{u_j s}, g_2^{\omega_j / u_j}) = e(g, g_2)^{\omega_j s}$; if $vk_j = 1$, calculate

$e(E_j, G_{j1}) = e(g^{u_{j+l}s}, g_2^{\omega_j/u_{j+l}}) = e(g, g_2)^{\omega_j s}$. Then we can get $W' = \prod_{j=1}^{l} e(g, g_2)^{\omega_j s} = e(g, g_2)^{\omega s}$. Thus, we can get the final result $CT' = \{CT_0', W'\}$.

Decrypt (CT, CT', SK). If the user has the privilege to access the data, then upon receiving CT' from D-CSP, the user completes the decryption and gets a message $M = C_0 / (CT_0'^{SK}, W')$.

5.3. Proof of Security under SE-CCA

We prove the following theorem:

THEOREM 1. *If an adversary can break the scheme of O-KP-ABE under the SE-CCA model, then a simulator can be constructed to solve the DBDH problem with a non-negligible advantage.*

PROOF: Suppose there exists a polynomial adversary A who can attack our scheme under the SE-CCA model with advantage ε and the probability to forge an legal signature is Pr[forge], then we can build a simulator S who can win DBDH problem with a non-negligible advantage $\frac{\varepsilon}{2} - \frac{1}{2}\Pr[\text{forge}]$. The process of simulation is as follows:

The challenger C generates the tuple (A, B, C, Z). Then, the simulator S chooses a signature key pair (*sk*, vk**). The length of *vk** is *l* and vk_j* is the value of j_{th} bit in *vk**. We assume the universe of attributes, U, is defined.

Init. A chooses the set of attributes *S** it wishes to be challenged upon and sents it to S.

Setup. The simulator S sets $g_1 = A = g^{\alpha}$ (thus, $a = \alpha$) and $g_2 = B$. Then choose a random value $\omega \in Z_p$. For each $a_i \in$ U, S chooses random values $t_i', \beta_i' \in Z_p$. If $a_i \in S^*$, the simulator sets $T_i = g^{t_i'}$ and $P_i = B^{t_i'^{-1}.\beta_i'} = g_2^{t_i'^{-1}.\beta_i'}$ (thus, $t_i = t_i', \beta_i = \beta_i'$); if $a_i \notin S^*$, S sets $T_i = A^{t_i'}.g^{-\omega t_i'} = g^{(\alpha-\omega)t_i'}$ and $P_i = g^{\beta_i'/t_i'} = g^{(\alpha-\omega)\beta_i'/(\alpha-\omega)t_i'}$, (thus, $t_i = (\alpha-\omega)t_i', \beta_i = (\alpha-\omega)\beta_i'$).

Next, randomly choose $u_j' \in Z_p, 1 \le j \le 2l$. If $vk_j^* = 0$, set $u_j = u_j'$ and $u_{j+l} = bu_{j+l}'$; otherwise, set $u_j = bu_j'$ and $u_{j+l} = u_{j+l}'$. Then calculate $U_j = g^{u_j}$.

So the public parameters are $PK = (U, g, g_1, g_2, T_i, P_i, U_j), 1 \le i \le n, 1 \le j \le 2l$, and they will be sent to A.

Phase 1. The adversary A is allowed to make any of the following four queries repeatedly:

i. A submits an access tree *T* with the restriction that for all $x \in Y_T$, $x \notin S^*$, in which Y_T is the attributes set of leaves in *T*. And S must construct the corresponding key tuple (SK, OK, TK). S firstly chooses a random value $z \in Z_p$ and sets SK = *z*.

Then, set $q_r(0) = 1/z$ and calculate the value of $q_x(0)$ of each leaf node *x* in tree *T* following the steps of **Keygen_IN**. Next, let $Q_x(0) = (\alpha-\omega).q_x(0)$, thus $Q_r(0) = (\alpha-\omega)/z$. Since the simulator sets $\beta_i = (\alpha-\omega)\beta_i'$ for all $a_i \notin S^*$, we can calculate $d_x = Q_x(0)/\beta_i = q_x(0)/\beta_i'$. Afterwards, randomly choose $\omega_j \in Z_p$, $1 \le j \le l-1$, and calculate $\omega_l = \omega - \sum_{j=1}^{l-1}\omega_j$. If $vk_j^* = 0$, calculate $G_{j0} = g_2^{\varphi_j/u_j} = B^{\varphi_j/u_j}$ and $G_{j1} = g_2^{\varphi_j/u_{j+l}} = g_2^{\varphi_j/bu_{j+l}'} = g^{\varphi_j/u_{j+l}'}$; otherwise,

$G_{j0} = g_2^{\varphi_j/u_j} = g^{\varphi_j/u_j'}$ and $G_{j1} = g_2^{\varphi_j/u_{j+l}} = g_2^{\varphi_j/u_{j+l}'} = B^{\varphi_j/u_{j+l}'}$. Thus, we get $G = \{G_{j0}, G_{j1}\}$. The outsourcing key is OK=$\{T, \{d_x\}_{x \in Y_T}, G\}$.

For each element d_x in OK calculate $D_x = P_i^{d_x}$, and the transformation key is TK=$\{T, \{D_x\}_{x \in Y_T}, G\}$.

Finally, send (SK, OK, TK) to the adversary A.

ii. A submits an access tree *T* with the restriction that for all $x \in Y_T$, $x \in S^*$, in which Y_T is the attributes set of leaves in *T*. And S must construct the corresponding key tuple (OK, TK).

S firstly chooses a random value $z' \in Z_p$, and sets $\delta = 1/z'$ (thus $z = (\alpha - \omega).z'$). Then, set $q_r(0) = 1/z'$ and calculate the value of $q_x(0)$ of each leaf node *x* in tree *T* following the steps of **Keygen_IN**. Since the simulator sets $\beta_i = \beta_i'$ for all $x \in S^*$, we can calculate $d_x = q_x(0)/\beta_i = q_x(0)/\beta_i'$. Afterwards, generate *G* using the same approach in i. Thus, the outsourcing key is OK=$\{T, \{d_x\}_{x \in Y_T}, G\}$.

For each element d_x in OK calculate $D_x = P_i^{d_x}$, and the transformation key is TK=$\{T, \{D_x\}_{x \in Y_T}, G\}$.

iii. A submits a ciphertext CT encrypted by *S**, the simulator must transform it to CT'.

First, S verifies the correctness of σ. On failure, the simulator will terminate the game and return $\perp$, which is called the "Exist Event". Otherwise, generate an access tree and revoke ii to get the corresponding TK. Then with TK, S can transform CT to CT'.

iv. A submits a ciphertext CT=$\{C, \sigma, vk\}$ encrypted by *S**, and the simulator will decrypt it.

First, revoke query iii, if the result is not $\perp$, then proceed as follows:

- ✧ If *vk* = *vk**, we call the "Forgery" happens and the simulator will terminate the game and output the guess *u*' of *u* randomly.
- ✧ If $vk \neq vk^*$, we can assume the j_{th} bit of them are different. Without loss of generality, we assume $vk_j = 1$ and thus $vk_j^* = 0$. Therefore, $E_j = g^{u_{j+l}s} = g^{bu_{j+l}s}$. Then the simulator can calculate $e(A, E_j) = e(g^a, g^{bu_{j+l}'s} = e(g,g)^{abs.u_{j+l}'}$. Since u_{j+l}' is known to S, he calculates $C_0 / e(A, E_j)^{1/u_{j+l}'s}$ as the message m and sends it to the adversary.

Challenge. The adversary A submits two challenge messages m_0, m_1 with equal length to S. The simulator S will flip a fair binary coin *b*, and returns an encryption of m_b. The ciphertext is outputted as $C^* = \{S^*, C_0 = m_b.z, \{C_y = C^{t_i'}\}_{y \in S^*}, E\}$. E is calculated as follows: if $vk_j^* = 0$, $E_j = U_j^s = g^{u_j's} = C^{u_j'}$; otherwise, $E_j = U_{j+l}^s = g^{u_{j+l}'s} = C^{u_{j+l}'}$.

If $u = 0$ then $Z = e(g,g)^{abc}$. If we let $s = c$, then we have $C_0 = m_b.e(g,g)^{abc} = m_b.e(g^a, g^b)^c = m_b.e(g_1, g_2)^s$, $C_y = C^{t_i'} = g^{t_i's} = T_i^s$. Therefore, the ciphertext is a valid random encryption of message m_b.

If $u = 1$, then $Z = e(g,g)^z$. Thus, $C_0 = m_b.e(g,g)^z$. Since *z* is random, C_0 will be a random element of G_T from adversary's view and the message contains no information about m_b.

Then, sign *C** with *sk** to get the signature σ. The final ciphertext is $CT^* = \{C^*, \sigma, vk^*\}$ and it will be sent to the adversary.

Phase 2. Repeat the process of **Phase 1** with the restriction that the ciphertexts submitted for decryption are not equal to *CT**.

Guess. A will submit a guess *b'* of *b*. If *b'*=*b*, the simulator will output *u'*=0 to indicate it was given a valid BDH-tuple, otherwise it will output *u'*=1 to indicate it was given a random 4-tuple.

First of all, "forge" represents that the event of forgery happens and "¬forge" represents the opposite.

If "Forgery" happens, S will not wait for A 's guess of *b* and output the guess of *u* randomly, thus $\Pr(u' = u \mid \text{forge}) = \frac{1}{2}$.

If *u*=1 and "Forgery" doesn't happen, the adversary gains no information about *b*. Therefore, we have $\Pr(b \neq b' \mid \neg\text{forge} \mid u = 1) = \frac{1}{2}$. Since the simulator guess *u'*=1 when *u'*=*u*, we have $\Pr(u' = u \mid \neg\text{forge} \mid u = 1) = \frac{1}{2}$. Thus, the probability to solve DBDH problem for S when *u*=1 is:

$$\begin{aligned}
&\Pr(u' = u \mid u = 1) \\
&= \Pr(u' = u, \text{forge} \mid u = 1) + \Pr(u' = u, \neg\text{forge} \mid u = 1) \\
&= \Pr(u' = u \mid \text{forge} \mid u = 1).\Pr(\text{forge}) + \\
&\quad \Pr(u' = u \mid \neg\text{forge} \mid u = 1).\Pr(\neg\text{forge}) \\
&= \tfrac{1}{2}.\Pr(\text{forge}) + \tfrac{1}{2}.(1 - \Pr(\text{forge})) \\
&= \tfrac{1}{2}
\end{aligned}$$

If *u*=0, the adversary sees an valid encryption of m_b. The adversary's advantage in this situation is ε by definition. Therefore, we have $\Pr(b' = b \mid \neg\text{forge} \mid u = 0) = \frac{1}{2} + \varepsilon$. Since the simulator guess *u'*=0 when *b'*=*b*, we have $\Pr(u' = u \mid \neg\text{forge} \mid u = 0) = \frac{1}{2} + \varepsilon$. Thus, the probability to solve DBDH problem for S when *u*=0 is:

$$\begin{aligned}
&\Pr(u' = u \mid u = 0) \\
&= \Pr(u' = u, \text{forge} \mid u = 0) + \Pr(u' = u, \neg\text{forge} \mid u = 0) \\
&= \Pr(u' = u \mid \text{forge} \mid u = 0).\Pr(\text{forge}) + \\
&\quad \Pr(u' = u \mid \neg\text{forge} \mid u = 0).\Pr(\neg\text{forge}) \\
&= \tfrac{1}{2}.\Pr(\text{forge}) + (\tfrac{1}{2} + \varepsilon).(1 - \Pr(\text{forge})) \\
&= \tfrac{1}{2} + \varepsilon - \varepsilon.\Pr(\text{forge})
\end{aligned}$$

Thus, the overall advantage of simulator in the DBDH game is:

$$\begin{aligned}
&\tfrac{1}{2}\Pr(u' = u \mid u = 0) + \tfrac{1}{2}\Pr(u' = u \mid u = 1) - \tfrac{1}{2} \\
&= \tfrac{1}{2}.(\tfrac{1}{2} + \varepsilon - \varepsilon.\Pr(\text{forge})) + \tfrac{1}{2}.\tfrac{1}{2} - \tfrac{1}{2} \\
&= \tfrac{1}{2}.\varepsilon - \tfrac{1}{2}.\varepsilon.\Pr(\text{forge}) \\
&\geq \tfrac{1}{2}.\varepsilon - \tfrac{1}{2}.\Pr(\text{forge})
\end{aligned}$$

6. ANALYSIS AND DISCUSSION

6.1. Analysis

This section compares our scheme with other existing KP-ABE outsourcing schemes in efficiency and security. The results are shown in Table 1.

Table 1. Comparisons in efficiency and security between our schemes and others.

Scheme	KG Ops	Dec Ops	Security Level
Green11	SS+5|Y|G	1G	RCCA
LCLJ13	3G	2P	CPA
LHLC13	(2|Y|+5)G	1G	CPA
O-KP-ABE(ours)	SS+2*l*G	1G	SE-CCA

G and P stand for the maximum time to compute an exponentiation in G and a pairing respectively. |Y| denotes the number of leaves in access tree. l is the length of vk. SS represents the time to share a secret in key-issuing phase.

As Green11 has not considered the outsourcing of key-issuing, the number of exponentiations that it must accomplish is proportional to the size of the access tree. Thus, its efficiency of key-issuing is relatively low. LCLJ13 can outsource both decrypting and key-issuing and PKG only needs to complete three exponentiations during the key-issuing phase. However, the user still has to complete two pairings when decrypting ciphertexts. LHLC13 improves that, and the user only needs to complete one exponentiation in decryption. But its efficiency in key-issuing decreases sharply, even no better than the original scheme [2] without outsourcing.

Our O-KP-ABE scheme has the highest efficiency in decryption, where the user only needs one exponentiation. We have utilized the technique of one-time signature, and PKG must do extra 2l exponentiations. Thus, its efficiency in key-issuing is relatively low. That is the cost of security.

In the aspect of security, Green11 can resist the Replayable Chosen Ciphertext Attack (RCCA) [9]. The traditional notion of security against CCA is a bit too strong, since it does not allow any bit of the ciphertext to be altered. However, there exist encryption schemes that are not CCA secure, but seem sufficiently secure “for most practical purposes”. For these reasons, Canetti et al. proposed the notion of RCCA. On one hand, RCCA security accepts as more secure than some non-CCA schemes; on the other hand, it suffices for most of existing applications of CCA security. Thus, the security of RCCA lies between CPA and CCA. Although the authors of LCLJ13 and LHLC13 declared that their schemes are CPA secure, we have proved that they have severe security vulnerability when collusion is considered. Our scheme has been proved secure under the SE-CCA model, which means O-KP-ABE has removed the security vulnerability in LCLJ13 and LHLC13 and can be CCA secure. Thus our scheme has a relatively higher security compared with the existing schemes.

6.2. Discussions

Verifiability. Although in our scheme the proxy servers, namely CSPs, cannot learn anything useful, there is no guarantee on the correctness of the outsourcing results. In some applications users or PKGs often request to check whether the outsourcing work is indeed done correctly. This is another important issue in outsourcing KP-ABE, and some approaches have been proposed. For example, Lai et al. [11] and Li et al. [8] addressed this problem by appending a redundancy to the ciphertext. However, both of them only considered the verification of outsourced decrypting, they

have not considered the same request for outsourced key-issuing. In our future work, we will consider the verification issue of our O-KP-ABE scheme.

Similar Problems for CP-ABE. Despite the work like this paper that focus on the KP-ABE outsourcing scheme, there are also many papers [10] engaged in the outsourcing of encryption and decryption of CP-ABE. However, we find that some of them have the similar security vulnerabilities to LCLJ13 and LHLC13. For example, in Zhou et al.'s scheme [10], the ciphertext embedded with the access policy $T_{ESP} \wedge T_{DO}$ can be decrypted by any user that satisfies T_{DO} by colluding with ESP. Firstly, the user gets s_1 from ESP. Then he chooses a pair of key components $d = (D_j, D_j')$ from his secret key, and we assume that the corresponding attribute is y. After that, the user can compute the ciphertext components pair corresponding to y as $c = (C_y = g^{s_1}, C_y' = H(y)^{s_1})$. And then he can calculate $e(g,g)^{rs_1}$ with d and c. In addition, because the user satisfies T_{DO}, he can compute $e(g,g)^{rs_2}$. At last, with the public parameter h and the key component D, the user can restore the message in the ciphertext. Thus we can see that this scheme has severe security problems. However, if we replace the hash function H with the attributes universe, then the similar technique in our scheme can be used to solve this problem. We leave this problem to our future work.

7. CONCLUSION

It is unrealistic for the existing KP-ABE outsourcing schemes to assume that curious users will not collude with KG-CSP. Thus, in this paper we first proposed a new security enhanced model SE-CCA. All CSPs in SE-CCA are curious and allowed to collude with each other. Besides, the attackers can get decryption service for specific ciphertexts in SE-CCA. Then, we constructed a concrete outsourcing scheme O-KP-ABE and proved its security under SE-CCA. Our scheme has the highest security compared with existing schemes. Except for that, O-KP-ABE also has relative higher efficiency. Hence, our construction has a comprehensive advantage over existing schemes in security and efficiency.

ACKNOWLEDGEMENTS

This work was supported by the National Natural Science Foundation of China (Grant No.61170088) and Foundation of the State Key Laboratory of Software Development Environment (Grant No. SKLSDE-2013ZX-05).

REFERENCES

[1] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in Advances in Cryptology-EUROCRYPT 2005. Springer, 2005, pp. 457–473.

[2] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in Proceedings of the 13th ACM conference on Computer and communications security. ACM, 2006, pp. 89–98.

[3] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in Security and Privacy, 2007. SP'07. IEEE Symposium on. IEEE, 2007, pp. 321–334.

[4] L. Ibraimi, Q. Tang, P. Hartel, and W. Jonker, "Efficient and provable secure ciphertext-policy attribute-based encryption schemes," in Information Security Practice and Experience. Springer, 2009, pp. 1–12.

[5] B. Waters, "Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization," in Public Key Cryptography–PKC 2011.Springer, 2011, pp. 53–70.

[6] M. Green, S. Hohenberger, and B. Waters, "Outsourcing the decryption of abe ciphertexts." in USENIX Security Symposium, 2011, p. 3.

[7] J. Li, X. Chen, J. Li, C. Jia, J. Ma, and W. Lou, "Fine-grained access control system based on outsourced attribute-based encryption," in Computer Security–ESORICS 2013. Springer, 2013, pp. 592–609.

[8] J. Li, X. Huang, J. Li, X. Chen, and Y. Xiang, "Securely outsourcing attribute-based encryption with checkability," IEEE Transactions on Parallel and Distributed Systems, pp. 2201–2210, 2013.

[9] R. Canetti, H. Krawczyk, and J. B. Nielsen, "Relaxing chosen-ciphertext security," in Advances in Cryptology-CRYPTO 2003. Springer, 2003, pp. 565–582.

[10] Z. Zhou and D. Huang, "Efficient and secure data storage operations for mobile cloud computing," in Proceedings of the 8th International Conference on Network and Service Management. International Federation for Information Processing, 2012, pp. 37–45.

[11] J. Lai, R. H. Deng, C. Guan, and J. Weng, "Attribute-based encryption with verifiable outsourced decryption," Information Forensics and Security, IEEE Transactions on, vol. 8, no. 8, pp. 1343–1354, 2013.

[12] M. Blaze, G. Bleumer, and M. Strauss, "Divertible protocols and atomic proxy cryptography," in Advances in CryptologyEUROCRYPT'98. Springer, 1998, pp. 127–144.

[13] G. Ateniese, K. Fu, M. Green, and S. Hohenberger, "Improved proxy re-encryption schemes with applications to secure distributed storage," ACM Transactions on Information and System Security (TISSEC), vol. 9, no. 1, pp. 1–30, 2006.

[14] R. Canetti and S. Hohenberger, "Chosen-ciphertext secure proxy re-encryption," in Proceedings of the 14th ACM conference on Computer and communications security. ACM, 2007, pp. 185–194.

[15] R. Canetti, S. Halevi, and J. Katz, "Chosen-ciphertext security from identity-based encryption," in Advances in Cryptology-Eurocrypt 2004. Springer, 2004, pp. 207–222.

[16] L. Cheung and C. Newport, "Provably secure ciphertext policy abe," in Proceedings of the 14th ACM conference on Computer and communications security. ACM, 2007, pp. 456–465.

[17] A. Beimel, "Secure schemes for secret sharing and key distribution," Ph.D. dissertation, Technion-Israel Institute of technology, Faculty of computer science, 1996.