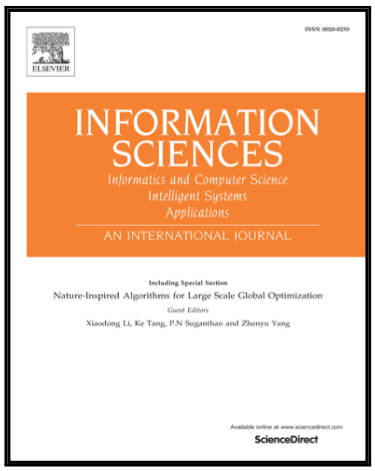

Security-Level Switchable Attribute-Based Encryption under The Strictly Weaker Assumption Family

Ge Song, Yuqiao Deng

PII: S0020-0255(16)31258-0
DOI: https://doi.org/10.1016/j.ins.2018.12.062
Reference: INS 14162

To appear in: *Information Sciences*

Received date: 14 October 2016
Revised date: 24 December 2018
Accepted date: 27 December 2018

Please cite this article as: Ge Song, Yuqiao Deng, Security-Level Switchable Attribute-Based Encryption under The Strictly Weaker Assumption Family, *Information Sciences* (2018), doi: https://doi.org/10.1016/j.ins.2018.12.062

Security-Level Switchable Attribute-Based Encryption under The Strictly Weaker Assumption Family

Ge Song[a], Yuqiao Deng[b,*]

[a] *College of Mathematics And Informatics, South China Agricultural University, Guangzhou, China.*
[b] *School of Mathematics And Statistics, Guangdong University of Finance and Economics, Guangzhou, China*

Abstract

Attribute-Based Encryption (ABE), a special type of public key encryption, efficiently shares sensitive data with fine-grained access control. ABE can be classified into two types: Ciphertext-Policy ABE (CP-ABE) and Key-Policy ABE (KP-ABE). However, the securities of most presented ABE systems were reduced to the q-type DBDH (Dicisional Diffie-Hellman Assumption) assumptions, which are stronger than the DBDH assumption. So, the above-mentioned ABE systems become insecure if DBDH is proved to be insecure. We propose a new ABE framework, called security-level switchable ABE (SLS-ABE). In SLS-ABE framework, a series of ABE systems can be generated and their securities are reduced to a k-BDH assumption family proposed by Benson et al. The k-BDH assumption family has the following properties: 1) any assumption in the k-BDH assumption family is associated with a parameter k, and the assumption becomes strictly weaker as the parameter k increases. 2) the 1-BDH assumption is proved to be equivalent to the DBDH assumption. So, all the k-BDH assumptions where $k > 1$ are weaker than DBDH assumption. We apply the technique of Benson et al. to construct ABE on k-BDH assumption, furthermore, we design a new framework to support the flexible switchable security-level for users. Concretely, the master public key, the master secret key and core keys issued by the system are constant. A User can generate different security-level public key/secret key pairs if it holds the core key. We propose a public key forgery attack model (PKFA) to capture the behaviors of adversary for generating a forged public

*Corresponding author. Email: gdufedyq@foxmail.com

key. We formally prove the selective-CPA security and PKFA security of our ABE systems. We compare the performances of our systems with Waters' ABE systems.

Keywords: KP-ABE, CP-ABE, k-BDH assumption family, selective security Model, PKFA

1. Introduction

Attribute Based Encryption (ABE), which has been presented by Sahai and Waters [17], is an influential paradigm for embedding complex access policy into the encrypted data. Key-Policy Attribute Based Encryption (KP-ABE) and Cipertext-Policy Attribute Based Encryption (CP-ABE) are two typical kinds of the ABE scheme [12]. In KP-ABE, the ciphertext is associated with the attributes set and the private key is associated with the access policy; In CP-ABE, the ciphertext is associated with the access policy and the private key is associated with the attributes set.

ABEs attract increasing concerns on new functionalities[7, 14] or better performance[10, 11, 13] in recent years, *however, most of them suffer from two indetectable secure problems described as follows*:

(1) *The q-type DBDH assumptions can not guarantee the security of ABE while encountering Cheon's attack [8].* Most of proposed ABEs are reduced to "q-type DBDH " assumptions [7, 15, 13, 19, 12]. Nevertheless, Cheon[8] claimed that q-type assumptions (and surely the ABEs associated with them) might meet a special attack. Recently, Sakemi et al. showed that Cheon's attack could be realized through executing a successful experiment. It means that the ABE system built on the q-type assumption might not be secure when encountering Cheon's attack.

(2) *Any single assumption may become insecure when new attacks against this assumption are found.* Almost all the frameworks of ABE are built to adapt to one assumption, the drawback of the above frameworks is that, they can not provide the property of "scalable", that is, when the current assumption upon which the framework built becomes insecure, the existing ABEs can not provide the "plug and play" mechanism to switch the old framework to a new one which based on a more secure assumption. In other words, the existing ABE framework is relatively "fixed" for *one assumption*.

We will simultaneously overcome the above two problems by employing a new *Security-Level Switchable* ABE framework. The assumption upon which ABE relies can be simply "switched" in a assumption series (we named the assumption series as Ł). The assumption series Ł satisfies the following three conditions: first, each assumption in Ł is weaker than q-type DBDH assumptions. Second, each assumption is associated with a parameter κ, we denote this assumption as AS_κ. Third, assumption AS_κ in Ł becomes progressively weaker as the value of the parameter κ increases.

We address the two secure problems mentioned above, if we can construct an ABE framework accommodating an arbitrary assumption in the assumption series Ł. First, since any assumption in Ł is weaker than the q-type DBDH assumption, our improved ABE is more secure than the existing schemes relied on the q-type assumptions. Second, when the current relied assumption AS_{κ_1} becomes insecure, we can switch to an *even weaker and remains secure assumption*, AS_{κ_2} where $\kappa_1 < \kappa_2$ (Notice that according to the property of Ł, assumption AS_{κ_2} is weaker than assumption AS_{κ_1}).

According to the description above, the remaining problem is that, first, we must find such a series of assumptions, Ł, and second, we must construct a suitable framework for Ł. Recently, Benson et al [3] proposed a proper assumption family: k-BDH Assumption Family. k-BDH Assumption Family satisfies all the properties of assumption series Ł. We give a brief introduction of the k-BDH Assumption Family as follows.

1.1. The k-BDH Assumption Family

The k-BDH Assumption Family, which firstly proposed in [3], is a decisional assumption family. We describe the k-BDH assumption as follows.

Let $\mathbb{G}$ be a multiplication group of prime order p, $x, y, r_1, ..., r_k \in Z_p$ be chosen at random and $g, v_1, ..., v_k$ be generators of $\mathbb{G}$. A vector

$$\vec{z} = (\mathbb{G}, p, g, g^x, g^y, v_1, ..., v_k, v_1^{r_1}, ..., v_k^{r_k})$$

is given to any probability polynomial time (PPT) algorithm $\mathcal{A}$.

The advantage ε for $\mathcal{A}$ to solve the k-BDH assumption is defined as

$$\varepsilon = |Pr[\mathcal{A}(\vec{z}, T = K) = 1] - Pr[\mathcal{A}(\vec{z}, T = R) = 1]| - \tfrac{1}{2},$$

where $K = e(g, g)^{xy(r_1+\cdots+r_k)}$ and R is a random element chosen from $\mathbb{G}$. The k-BDH assumption is solvable, if ε is non-negligible.

We note that each k-BDH assumption is associated with a parameter k. Especially, Benson et al. [3] gave the proof that the 1-BDH assumption is equivalent to the DBDH assumption, and the assumptions in the k-BDH assumption family become progressively weaker when the parameter k increases (section 4 in [3]).

We demonstrate that the k-BDH assumption family satisfies three properties of Ł. first of all, the k-BDH assumption family satisfies the first property of Ł: since the DBDH assumption is weaker than the q-type DBDH assumption, the 1-BDH assumption is equivalent to the DBDH assumption, and the l-BDH assumption is weaker than 1-BDH ($l > 1$), we can deduce that the l-BDH assumptions ($k \geq 1$) are weaker than the q-type DBDH assumption. Secondly, the k-BDH assumption family satisfies the second property of Ł as each k-BDH assumption is associated with a parameter k . Finally, it is obvious that the k-BDH assumption family satisfies the third property of Ł. Therefore, the k-BDH assumption family matches all the requirements of the assumption series Ł.

1.2. Our Technique

We mainly utilize the following two techniques in this paper.

Reduction from DBDH to k-BDH. The reduction from a general ABE to the k-BDH assumption is not straightforward, we briefly explain the difficulties of our technique. In the k-BDH assumption, we are given a vector

$$\vec{z} = (\mathbb{G}, p, g, g^x, g^y, v_1, \cdots, v_k, v_1^{r_1}, \cdots, v_k^{r_k}),$$

and to distinguish the term $T = e(g,g)^{xy(r_1+\cdots+r_k)}$ from a random element in $\mathbb{G}$. In the security reduction, we usually use the term T to randomize the message m and the vector $\vec{z}$ to generate the remaining ciphertext components and the secret keys. The difficulty arises because $\vec{z}$ includes bases $(v_1, v_2, \cdots, v_k)$, whereas T only contains one base g. So, we must "transform" the bases $(v_1, v_2, \cdots, v_k)$ into the base g in the security reduction phase.

Our work starts from considering a simplified transformation: we first transform the DBDH assumption to the 1-BDH assumption; then, we append the parameters of 1-BDH assumption to obtain the k-BDH assumption. It is given $\vec{z'} = (g, g^a, g^b, g^c)$ and to determine $T' = e(g,g)^{abc}$ in the DBDH assumption. We simply let $x = a$ and $y = b$ (x, y are parameters come from the 1-BDH assumption). However, there are no v_1 and r_1 in the DBDH assumption. We use the following trick proposed by [3]. We choose two

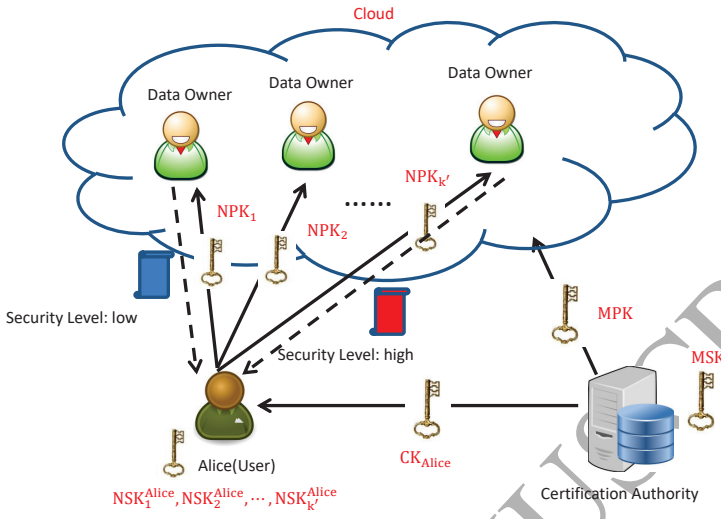

Figure 1: The Model of Our System

integers, then issue $\{v_1^{s_1}, v_1^{r'_1}\}$. The term $v_1^{s_1}$ is reduced to the term g and the term $v_1^{r'_1}$ is reduced to the term g^c. Concretely, suppose $v_1 = g^{k_1}$, where k_1 is some unknown term, we implicitly set $s_1 = l_1/k_1, r'_1 = c/k_1$, where l_1 is a chosen parameter to randomize the parameter. So, we have $v_1^{s_1} = (g^{k_1})^{l_1/k_1} = g^{l_1}, v_1^{r'_1} = (g^{k_1})^{c/k_1} = g^c$. We can then obtain the reduction methodology from DBDH framework to k-BDH assumption framework by appending the above parameters (i.e., append $\{v_1^{s_1}, v_1^{r'_1}\}$ to $\{v_i^{s_i}, v_i^{r'_i}\}_{i\in[k]}$).

Security Switchable Framework. Our ABE framework is motivated by the work of IBE proposed by [3]. However, that IBE cannot be flexibly switched among the k-BDH assumptions, because the parameter k is embedded in the public key. In other words, one needs to reset the system parameters when the IBE is switched to another assumption, and the reset operation makes the system unpractical.

We conquer the abovementioned obstacle by using the following technique. As shown in Fig.1, in our new framework, the PK/SK pair can be classified into three types, namely, the master public key (MPK)/ master secret key (MSK) pair, the normal public key (NPK)/ normal secret key (NSK_I) pair and a core key (CK_I), where I denotes the "Identity" of the user (the "identity" in the context of ABE means an attribute set in CP-ABE, or an access structure in KP-ABE). The MPK/MSK and the CK_I are

generated by the certification center (CA), whereas the NPK/NSK_I pair is issued by the user. The MPK/MSK and the CK_I are constant and the NPK/NSK_I pair is variable.

We achieve the following two advantages from this framework. First, the system reset is unnecessary, because the MPK/MSK pair and CK_I that generated by CA are constant. Second, our framework provides more flexible security choices for users. For example, suppose a user needs to receive sensitive information from multiple data owners in the cloud. Frequently, the security levels of this information are different, e.g., the disclosure of some trade secret documents is not allowed; and the disclosure of some paid movie videos is acceptable. The CA forms the MPK/MSK pair and generates one CK_I for each user. Every user is permitted to issue different "security level" NPK_k and corresponding NSK_k^I by itself if it holds the CK_I, where k denotes the security level (the larger the value of k, the higher the security level).

However, we must consider the following security issue: if a malicious user can generate other users' NPK/NSK pair, this will pose a security threat, we call this type of attack as public key forgery attack (PKFA). How to construct a security level switchable ABE framework that can resist PKFA is a challenge. The CK_I is indeed introduced to forbid the PKFA. We explain the role of CK_I below.

As shown in Fig. 2, in our framework, an adversary is allowed to issue $NPK_{k'}/NSK_{k'}^I$ pair, if it holds the core key CK_I issued by the CA, and it cannot form $NPK_{k'}/NSK_{k'}^{I'}$ pair where $I' \neq I$. In order to achieve the above goal, we first require that CK_I must be included in the $NSK_{k'}^I$, then one can modify the user's identity I to some I' if it can modify CK_I to $CK_{I'}$. Then, we utilize the following technique to forbid the modification of CK_I: we introduce a selective CPA-secure ABE scheme $ABE_s = (Setup_s, Enc_s, KeyGen_s, Dec_s)$. The CA initials the system through running the $Setup_s$ algorithm, then it executes the $KeyGen$ algorithm to generate secret key SK_I for user I, and CK_I is set to be SK_I in this phase. A selective CPA-secure ABE essentially prevents the user's identity from being modified (else, the CPA-secure can be broken). Therefore, an adversary can modify the identity of $NSK_{k'}^I$, if it can break the selective CPA-security of generic ABE. We will provide detailed proof about this below.

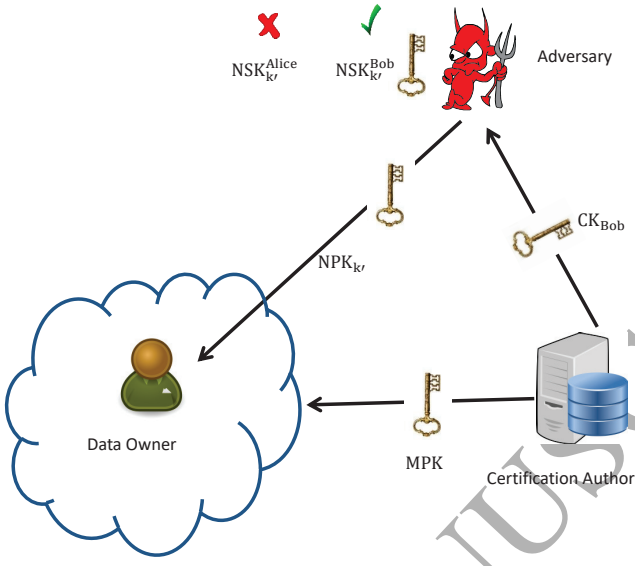

Figure 2: The PKFA Model

1.3. Our Contribution

The contribution of the paper is multifold. First, trivial ABE scheme will suffer from severe attack once the underlying assumption becomes insecure and must redesign the whole scheme in order to change the underlying assumption to a new one. We handle this problem by proposing a new ABE framework. The underlying assumption in our ABE framework can be flexibly "switched". Second, we demonstrate how to implement the ABE framework mentioned above. We propose two ABE schemes: KP-ABE and CP-ABE separately. Our two ABE schemes are both built to adapt to the assumption series: k-BDH assumption family. Third, the public key is constant in our ABE framework, so the system needs not to be reset when the underlying assumption changes.

1.4. Related Work

The first work related to Identity-Based Encryption (IBE) [5] was proposed by Boneh-Franklin a decade ago. After that, lots of works focused on all kinds of IBE schemes are presented [9][18][6].

Recently, Karyn Benson, Hovav Shacham and Brent Waters [3] proposed an IBE system based on an arbitrary assumption in k-BDH assumption.

They proved that the assumption can generalize DBDH assumption. Indeed, their works strengthen the security of IBE as one can create IBE scheme reduced to an weak enough assumption in k-BDH assumption family as it needs.

A new research trend about IBE has been put forward by Sahai and Waters [17]. They proposed an fuzzy IBE scheme providing the mechanism for the data provider to express how he will share data in the encryption algorithm. On this basis, Sahai and Waters [17] presented a new concept: Attribute-Based Encryption (ABE), in which a user's credentials is represented by a set of string called attributes and the predicate is represented by a formula over these attributes.

Most of existing ABE schemes are proposed based on the DBDH or q-type DBDH assumption. For example, in respect of KP-ABE, Goyal, Pandey, Sahai, and Waters [12] proposed an expressive KP-ABE scheme with fine-gained access control which is based on the DBDH assumption. Attrapadung et al [1] proposed a KP-ABE scheme with constant-size ciphertexts which is based on the q-DBDHE assumption (q-Decisional Bilinear Diffie-Hellman Exponent, one type of q-type DBDH). Ostrovsky et al. [15] proposed a KP-ABE scheme with a non-monotonic access structure where the secret keys are assocaited with a set of attributes including positive and negative attributes, their scheme is based on DBDH assumption. Rouselakis and Waters [16] proposed a KP-ABE with large universe using a new proving method, their scheme is under the assumption called "q-2" assumption which belongs to the q-type DBDH assumption.

In respect of CP-ABE, In 2007, using a monotonic access tree as access structure, Bethencourt et al. [4] proposed the first CP-ABE construction, however, the security of their scheme is limited that their system is proven secure in the generic group model. After that, Waters [19] proposed three CP-ABE schemes expressing the access structure using the tool of Linear Secret Sharing Scheme (LSSS), the three CP-ABEs are based on the q-parallel DBDHE assumption (q-parallel Bilinear Diffie-Hellman Exponent problem, one type of q-type DBDH), q-DBDHE assumption and DBDH assumption separately. There have been increasing concerns on CP-ABE, for example, Goyal et al proposed a bounded CP-ABE under the DBDH assumption, Chase [7] proposed Multi-authority CP-ABE under the DBDH assumption, Rouselakis and Waters [16] proposed a CP-ABE with large universe under the so-called "q-1" assumption, which is a type of q-type DBDH assumption, and so on.

2. Background

We first give the definition of bilinear maps, access structures and Linear Secret Sharing Schemes (LSSS). Finally, we give the security definitions of Ciphertext-Policy Attribute Based Encryption (CP-ABE) and Key-Policy Attribute Based Encryption (KP-ABE).

2.1. Bilinear Maps

Let $\mathbb{G}$ and $\mathbb{G}_T$ be two multiplicative cyclic groups of prime order p. Let g be a generator of $\mathbb{G}$ and e be a bilinear map, $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$. The bilinear map e has the following properties:

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

2.2. Access Structures

Definition 1 (Access Structure [2]) Let $\{P_1, P_2, ..., P_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^{\{P_1, P_2, ..., P_n\}}$ is monotone *if* $\forall B, C :$ *if* $B \in \mathbb{A}$ *and* $B \subseteq C$ *then* $C \in \mathbb{A}$.

In this paper, attributes are the equivalent of the parties. Thus, the access structure $\mathbb{A}$ will contain the authorized sets of attributes. And, in our context, the access structure is monotone [19].

2.3. Linear Secret-Sharing Schemes

Definition 2 (Linear Secret-Sharing Schemes (LSSS)) A secret-sharing scheme Π over a set of parties P is called linear (over $\mathbb{Z}_p$) if:

(1) A vector is formed by the shares for each party over $\mathbb{Z}_p$.
(2) Let M be a matrix of l rows and n columns, the function ρ be a map that maps row i to a party $\rho(i)$, where $i = 1, ..., l$. Assume a secret $s \in \mathbb{Z}_p$ is needed to be shared, choose a vector $v = (s, r_2, ..., r_n)$, where $r_2, ..., r_n \in \mathbb{Z}_p$ are randomly chosen, then $(Mv)_i$ for $i = 1, ..., l$ is the l shares of the secret s and the party $\rho(i)$ owns the share $(Mv)_i$.

It is pointed out in [2] that, every LSSS also enjoys the *linear reconstruction* property: Let $S \in \mathbb{A}$ be any authorized set, and let $I \subset (1, 2, ..., l)$ is the following set that $I = (i : \rho(i) \in \mathbb{Z}_p)$. Then, there must exist constant set $\{\omega_i \in \mathbb{Z}_p\}$ satisfying $\sum_{i \in I} \omega_i \lambda_i = s$, if λ_i are valid shares of any secret s.

2.4. Definitions of ABE

We briefly review the definitions of CP-ABE and KP-ABE in this section[17].

2.4.1. Definitions of CP-ABE

The CP-ABE is consisted of four algorithms: $Setup, Encrypt, KeyGen$ and $Decrypt$. The descriptions of these algorithms are demonstrated as follows.

Setup(U). This algorithm takes an attribute universe description U as input. It outputs the public key PK and the master secret key MSK.

Enc(PK,m,$\mathbb{A}$). This algorithm takes the PK, a message m, and an access structure $\mathbb{A}$ as intputs. It outputs the ciphertext CT.

KeyGen(MSK,S).This algorithm takes the MSK and an attribute set $\mathbb{A}$ as inputs. It outputs the secret key SK.

Dec(PK,CT,SK). This algorithm takes the CT and SK as inputs. It outputs the message m, if the attribute set embedded in the SK satisfies the access structure embedded in the CT.

Security Model for CP-ABE The security model of CP-ABE is a game played between the challenger and the adversary. The formal security game is shown as follows.

Init. The adversary publishes the challenge access structure $\mathbb{A}^*$.

Setup. The challenger generates the PK and sends it to the adversary.

Phase 1. The adversary queries any secret key associated with attribute set S with a limitation that, S can not satisfy the challenge access structure $\mathbb{A}^*$.

Challenge. The adversary submits two equal length messages m_0 and m_1. The challenger flips a random coin $b \in \{0, 1\}$, and forms the challenge ciphertext CT^* by encrypting the message m_b with the challenge access structure $\mathbb{A}^*$. The challenger gives CT^* to the adversary.

Phase 2. The same as the Phase 1.

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

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

A CP-ABE scheme is said to be selective secure, if any PPT algorithm has at most negligible advantage in winning the above game.

2.4.2. Definitions of KP-ABE

The KP-ABE is consisted of four algorithms: $Setup, Enc, KeyGen$ and Dec. The descriptions of these algorithms are demonstrated as follows.

Setup(U). This algorithm takes an attribute universe description U as input. It outputs the public key PK and the master secret key MSK.

Enc(PK,m,S). This algorithm takes the PK, a message m, and an attribute set S as inputs. It outputs the ciphertext CT.

KeyGen(MSK,$\mathbb{A}$).This algorithm takes the MSK and an access structure $\mathbb{A}$ as inputs. It outputs the secret key SK.

Dec(PK,CT,SK). This algorithm takes the CT and SK as inputs. It outputs the message m, if the attribute set embedded in the CT satisfies the access structure embedded in the SK.

Security Model for KP-ABE The security model of KP-ABE is a game played between the challenger and the adversary. The formal security game is shown as follows.

Init. The adversary publishes the challenge attribute set S^*.

Setup. The challenger generates the PK and sends it to the adversary.

Phase 1. The adversary queries any secret key associated with access structure $\mathbb{A}$ with a limitation that, the challenge attribute set S^* can not satisfy $\mathbb{A}$.

Challenge. The adversary submits two equal length messages m_0 and m_1. The challenger flips a random coin $b \in \{0, 1\}$, and forms the challenge ciphertext CT^* by encrypting the message m_b with the challenge attribute set S^*. The challenger gives CT^* to the adversary.

Phase 2. The same as the Phase 1.

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

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

A KP-ABE scheme is said to be selective secure, if any PPT algorithm has at most negligible advantage in winning the above game.

3. Security-Level Switchable ABE Framework and Security Models

We describe the Security-Level Switchable CP-ABE (SLS-CP-ABE) and Security-Level Switchable KP-ABE (SLS-KP-ABE) frameworks used to switch ABE among different assumptions in this section.

3.1. SLS-CP-ABE

A SLS-CP-ABE is consisted of five algorithms: *Setup, CoreKeyGen, NorKeyGen, Encrypt and Decrypt.* We formally describe these five algorithms as follows.

Setup(U). The setup algorithm takes the number U of attributes as inputs. It outputs the MPK and the MSK.

CoreKeyGen(MSK, S). The CoreKeyGen algorithm takes the MSK and the attribute set S as inputs. It outputs the core key CK^S.

NorKeyGen(S, CK^S, n_{max}, k').The NorKeyGen algorithm takes the attribute set S, the CK^S, the maximum number n_{max} of columns in the access structure, and a security parameter k as inputs. It outputs the "k' security level" normal public key $NPK_{k'}$ and normal secret key $NSK^S_{k'}$.

Encrypt(MPK,$NPK_{k'}$, $\mathbb{A}$, m). The Encrypt algorithm takes the MPK, the $NPK_{k'}$, a message m, and a LSSS access structure $\mathbb{A}$ as inputs. It outputs the "k' level security" ciphertext $CT^{\mathbb{A}}_{k'}$.

Decrypt$(CT^{\mathbb{A}}_{k'}, NSK^S_{k'})$. The Decrypt algorithm takes a $CT^{\mathbb{A}}_{k'}$ and a normal secret key $NSK^S_{k'}$ as inputs. It outputs the encrypted message m, if S satisfies $\mathbb{A}$.

We provide the rigorous PKFA model and selective security model of SLS-CP-ABE as follows.

PKFA Model for SLS-CP-ABE.

Init. $\mathcal{A}$ chooses and issues a challenge access structure $\mathbb{A}^*$.

Setup. $\mathcal{C}$ generates and issues the MPK to $\mathcal{A}$.

CoreKey Queries 1. $\mathcal{A}$ makes any core key query $CK^S_{k'}$ to $\mathcal{C}$ with a limitation that, the attribute set S cannot satisfy the challenge access structure $\mathbb{A}^*$.

Challenge. $\mathcal{A}$ chooses and submits two equal length messages m_0 and m_1. $\mathcal{C}$ flips a random coin b, and generates the challenge ciphertext $CT^{\mathbb{A}^*}_{k'}$ by encrypting the message m_b under the challenging access structure $\mathbb{A}^*$. $\mathcal{C}$ gives $CT^{\mathbb{A}^*}_{k'}$ to $\mathcal{A}$.

CoreKey Queries 2. The same as the CoreKey Queries 1 phase.

Guess. $\mathcal{A}$ outputs a guess b' of b.

The advantage of adversary $\mathcal{A}$ in winning the abovementioned game is defined as $Pr[b' = b] - \frac{1}{2}$.

A SLS-CP-ABE is said to be secure against PKFA, if any PPT algorithm can win the abovementioned game with at most negligible advantage.

Selective Security Model for SLS-CP-ABE. The selective security model of SLS-CP-ABE is a game played between the challenger $\mathcal{C}$ and the adversary $\mathcal{A}$.

Init. $\mathcal{A}$ chooses and issues a challenge access structure $\mathbb{A}^*$.

Setup. $\mathcal{C}$ generates and issues the MPK to $\mathcal{A}$.

CoreKey Queries 1. $\mathcal{A}$ makes any core key query CK^S to $\mathcal{C}$. $\mathcal{C}$ generates and sends the corresponding core key to $\mathcal{A}$.

NorKey Queries 1. $\mathcal{A}$ makes the normal public and secret key query $NPK_{k'}$, $NSK^S_{k'}$ to $\mathcal{C}$, where the $NPK_{k'}$ is the challenge NPK (namely, the challenge ciphertext is encrypted by $NPK_{k'}$). There is a limitation that,

the attribute set S included in $NSK_{k'}^{S}$ cannot satisfy the challenge access structure $\mathbb{A}^*$ (else, $\mathcal{A}$ can trivially win the game).

$\mathcal{C}$ generates and sends the corresponding normal key pair $NPK_{k'}$, $NSK_{k'}^{S}$ to $\mathcal{A}$.

Challenge. $\mathcal{A}$ chooses and submits two equal length messages m_0 and m_1. $\mathcal{C}$ flips a random coin b, and generates the challenge ciphertext $CT_{k'}^{\mathbb{A}^*}$ by encrypting the message m_b under the challenging access structure $\mathbb{A}^*$. $\mathcal{C}$ gives $CT_{k'}^{\mathbb{A}^*}$ to $\mathcal{A}$.

CoreKey Queries 2. The same as the CoreKey Queries 1 phase.

NorKey Queries 2. The same as the NorKey Queries 1 phase.

Guess. $\mathcal{A}$ outputs a guess b' of b.

The advantage of adversary $\mathcal{A}$ in winning the abovementioned game is defined as $Pr[b' = b] - \frac{1}{2}$.

A SLS-CP-ABE is said to be selective secure, if PPT algorithm can win the abovementioned game with at most negligible advantage.

Remarks. We provide the following observations of our CP-ABE framework.

1. We capture the public key forgery attack behavior of the adversary in the PKFA model. The adversary first issues a challenge access structure $\mathbb{A}^*$. Then, we provide Q core keys $\{CK^{S_i}\}_{i\in[Q]}$ to the adversary (Q is the permitted query times for the adversary), excepts for the key that the embedded attribute set S satisfies $\mathbb{A}^*$. The adversary can generate any NPK/NSK^{S_i} pair associated with attribute set S_i. However, because $\{S_i\}_{i\in[Q]}$ do not satisfy $\mathbb{A}^*$, the NSK set known by the adversary cannot decrypt the ciphertext formed by $\mathbb{A}^*$ (even the ciphertext is encrypted using the NPK formed by the adversary itself). The above model implies that, the adversary is granted Q "identity" (i.e., Q attribute sets $\{S_i\}_{i\in[Q]}$). It aims to derive a NSK associated with a new "identity" (i.e., a new attribute set), in order to satisfy the challenge access structure $\mathbb{A}^*$. However, the above attack fails, if the advantage of the adversary in winning the PKFA game is negligible.
2. Considering the selective security model. The adversary first issues a challenge access structure $\mathbb{A}^*$. Then, the adversary is allowed to hold any core key , the challenge NPK^*, and the NSK^S where S does not satisfy $\mathbb{A}^*$. The selective security definition shows that, if the selective security holds ,then the adversary is unable to create a $NSK^{*S'}$ where S' satisfies $\mathbb{A}^*$.

3.2. SLS-KP-ABE

The definition of SLS-KP-ABE is quite similar to the SLS-CP-ABE. We describe it as follows.

A SLS-KP-ABE is consisted of five algorithms: *Setup, CoreKeyGen, NorKeyGen, Encrypt and Decrypt.* We formally describe these five algorithms as follows.

Setup(U). The setup algorithm takes the number U of attributes as inputs. It outputs the MPK and the MSK.

CoreKeyGen$(MSK, \mathbb{A})$. The CoreKeyGen algorithm takes the MSK and the access structure $\mathbb{A}$ as inputs. It outputs the core key $CK^{\mathbb{A}}$.

NorKeyGen$(S, CK^{\mathbb{A}}, n_{max}, k')$.The NorKeyGen algorithm takes the access structure $\mathbb{A}$, the $CK^{\mathbb{A}}$, the maximum number n_{max} of columns in the access structure, and a security parameter k as inputs. It outputs the "k' security level" normal public key $NPK_{k'}$ and normal secret key $NSK^{\mathbb{A}}_{k'}$.

Encrypt$(MPK, NPK_{k'}, \mathbb{A}, m)$. The Encrypt algorithm takes the MPK, the $NPK_{k'}$, a message m, and a attribute set S as inputs. It outputs the "k' level security" ciphertext $CT^{S}_{k'}$.

Decrypt$(CT^{S}_{k'}, NSK^{\mathbb{A}}_{k'})$. The Decrypt algorithm takes a $CT^{S}_{k'}$ and a normal secret key $NSK^{\mathbb{A}}_{k'}$ as inputs. It outputs the encrypted message m, if S satisfies $\mathbb{A}$.

We provide the rigorous PKFA model and selective security model of SLS-KP-ABE as follows.

PKFA Model for SLS-KP-ABE.

Init. $\mathcal{A}$ chooses and issues a challenge attribute set S^*.

Setup. $\mathcal{C}$ generates and issues the MPK to $\mathcal{A}$.

CoreKey Queries 1. $\mathcal{A}$ makes any core key query $CK^{\mathbb{A}}_{k'}$ to $\mathcal{C}$ with a limitation that, the challenge attribute set S^* cannot satisfy the access structure $\mathbb{A}$.

Challenge. $\mathcal{A}$ chooses and submits two equal length messages m_0 and m_1. $\mathcal{C}$ flips a random coin b, and generates the challenge ciphertext $CT^{S^*}_{k'}$ by encrypting the message m_b under the challenging attribute set $\mathbb{S}^*$. $\mathcal{C}$ gives $CT^{S^*}_{k'}$ to $\mathcal{A}$.

CoreKey Queries 2. The same as the CoreKey Queries 1 phase.

Guess. $\mathcal{A}$ outputs a guess b' of b.

The advantage of adversary $\mathcal{A}$ in winning the abovementioned game is defined as $Pr[b' = b] - \frac{1}{2}$.

A SLS-KP-ABE is said to be secure against PKFA, if any PPT algorithm can win the abovementioned game with at most negligible advantage.

Selective Security Model for SLS-KP-ABE. The selective security model of SLS-KP-ABE is a game played between the challenger $\mathcal{C}$ and the adversary $\mathcal{A}$.

Init. $\mathcal{A}$ chooses and issues a challenge attribute set S^*.

Setup. $\mathcal{C}$ generates and issues the MPK to $\mathcal{A}$.

CoreKey Queries 1. $\mathcal{A}$ makes any core key query $CK^{\mathbb{A}}$ to $\mathcal{C}$. $\mathcal{C}$ generates and sends the corresponding core key to $\mathcal{A}$.

NorKey Queries 1. $\mathcal{A}$ makes the normal public and secret key query $NPK_{k'}$, $NSK_{k'}^{\mathbb{A}}$ to $\mathcal{C}$, where the $NPK_{k'}$ is the challenge NPK (namely, the challenge ciphertext is encrypted by $NPK_{k'}$). There is a limitation that, the challenge attribute set S^* cannot satisfy the access structure $\mathbb{A}$.

$\mathcal{C}$ generates and sends the corresponding normal key pair $NPK_{k'}$, $NSK_{k'}^{\mathbb{A}}$ to $\mathcal{A}$.

Challenge. $\mathcal{A}$ chooses and submits two equal length messages m_0 and m_1. $\mathcal{C}$ flips a random coin b, and generates the challenge ciphertext $CT_{k'}^{S^*}$ by encrypting the message m_b under the challenging attribute set S^*. $\mathcal{C}$ gives $CT_{k'}^{S^*}$ to $\mathcal{A}$.

CoreKey Queries 2. The same as the CoreKey Queries 1 phase.

NorKey Queries 2. The same as the NorKey Queries 1 phase.

Guess. $\mathcal{A}$ outputs a guess b' of b.

The advantage of adversary $\mathcal{A}$ in winning the abovementioned game is defined as $Pr[b' = b] - \frac{1}{2}$.

A SLS-KP-ABE is said to be selective secure, if any PPT algorithm can win the abovementioned game with at most negligible advantage.

4. Generic SLS-ABE

We first propose our SLS-ABEs (including SLS-CP-ABE and SLS-KP-ABE). Then, we give the security proofs of our schemes.

4.1. Generic SLS-CP-ABE

In this section, we first propose our construction; then, we prove the security of it.

4.1.1. Construction

Let $\rho(.)$ be a injective function that associates rows of LSSS matrix M to attributes, $[x]$ a positive integers set that $[x] = \{1, 2, ..., x\}$. The construction is described as follows.

Setup(U). The setup algorithm takes the number U of attributes as inputs. The algorithm chooses a selective-secure CP-ABE scheme (denoted as ABE_s)

$$ABE_s = (Setup_s, KeyGen_s, Enc_s, Dec_s).$$

Then, the algorithm executes the $Setup_s$ algorithm and generates the public key PK_s and the master secret key MSK_s. The MPK is issued as

$$MPK = \{PK_s\}.$$

The MSK is set as

$$MSK = \{MSK_s\}.$$

CoreKeyGen(MSK, S). The algorithm takes the MSK and the attributes set S as inputs, it calls the $KeyGen_s$ algorithm and achieves the secret key SK_s^S, it outputs the core key CK^S as

$$CK^S = \{SK_s^S\}.$$

NorKeyGen(S, CK^S, n_{max}, k'). The NorKeyGen algorithm takes the attributes set S, the MPK, the CK^S, the maximum number n_{max} of columns in the access structure, and a security parameter k' as inputs. The parameter k' represents the "security level" of these NPK/NSK pair. The algorithm generates the $NPK_{k'}$ and $NSK_{k'}^S$ as follows.

The algorithm chooses a group $\mathbb{G}$ of prime order p then generates $(g, \{v_i\}_{i\in[k']}) \in \mathbb{G}^{k'+1}$ and $(x, \{r_i\}_{i\in[k']}) \in Z_p^{k'+1}$, it chooses random elements $\{h_{t,i,j}\}_{t\in[k'],i\in[n_{max}],j\in[U]}$ and forms the $NPK_{k'}$ as

$$NPK_{k'} = \{g, g^x, \{v_t, v_t^{r_t}, h_{t,i,j}\}_{t\in[k'],i\in[n_{max}],j\in[U]}\}.$$

The algorithm also chooses random elements $p_{t,i} \in Z_p : (t \in [k'], i \in [n_{max}])$, then it computes

$$\begin{aligned} K_t &= g^{xr_t} g^{xp_{t,1}} : (t \in [k']), \\ L_{t,i} &= v_t^{p_{t,i}} : (t \in [k'], i \in [n_{max}]), \\ N_{t,\chi} &= \prod_{i\in[n_{max}]} (h_{t,i,\chi})^{p_{t,i}} : (t \in [k'], \chi \in S). \end{aligned}$$

The $NSK_{k'}^S$ is issued as follows

$$NSK_{k'}^S = \{CK^S, K_t, L_{t,i}, N_{t,\chi}\}_{t\in[k'],i\in[n_{max}],\chi\in S}.$$

Notice that, the core key CK^S is included in the $NSK^S_{k'}$, thus, the adversary cannot modify the attribute set S unless it can modify CK^S, we will formally prove this later.

Encrypt$(MPK, NPK_{k'}, \mathbb{A} = (M, \rho), m)$ The algorithm takes the MPK, the $NPK_{k'}$, a message m and a LSSS access structure (M, ρ) as inputs, it chooses $k'+1$ random vectors $\{\vec{r}, \{\vec{v_i} = (s_{i,1}, s_{i,2}, \cdots, s_{i,n_{max}})\}_{i\in[k']}\}$. The k' vectors $\vec{v_i}$ will be used to share the k' encryption exponents $s_{1,1}, \cdots, s_{k',1}$, and the vector $\vec{r}$ is a random vector used for the encryption of Enc_s algorithm .

The algorithm computes

$$\begin{aligned} C_0 &= Enc_s(MPK, \mathbb{A}, m \prod_{t\in[k']} e(g^x, v_t^{r_t})^{s_{t,1}}; \vec{r}), \\ C_t &= v_t^{s_{t,1}} : (t \in [k']), \\ C_{t,i,\tau} &= g^{xM_{\tau,i}\cdot s_{t,i}} h^{-s_{t,1}}_{t,i,\rho(\tau)} : (t \in [k'], i \in [n_{max}], \tau \in [\ell]) \end{aligned}$$

and issues the ciphertext

$$CT^{\mathbb{A}}_{k'} = \{C_0, \{C_t, C_{t,i,\tau}\}_{t\in[k'],i\in[n_{max}],\tau\in[\ell]}\}$$

along with a description of $\mathbb{A} = (M, \rho)$.

Decrypt$(CT^{\mathbb{A}}_{k'}, NSK^S_{k'})$ The decryption algorithm takes a ciphertext $CT^{\mathbb{A}}_{k'}$ and a private key $NSK^S_{k'}$ as inputs. Suppose the attributes set S satisfies the access structure $\mathbb{A}$, let set J be defined as $J = \{\tau : \rho(\tau) \in S\}$, then we can find a constants set $\{\omega_{t,\tau} \in \mathbb{Z}_p\}_{\tau\in J}$ satisfying:

$$\begin{aligned} \sum_{\tau\in J} \omega_{t,\tau} M_{\tau,1} &= 1, \\ \sum_{\tau\in J} \omega_{t,\tau} M_{\tau,2} &= 0, \\ &\vdots \\ \sum_{\tau\in J} \omega_{t,\tau} M_{\tau,n_{max}} &= 0, \end{aligned}$$

if terms $\{\lambda_{t,\tau} = M_{t,\tau} \cdot \overrightarrow{v_t}\}$ are valid shares of the secret $s_{t,1}$ where $t \in [k']$.

The above equations hold because if terms $\{\lambda_{t,\tau} = M_{t,\tau} \cdot \overrightarrow{v_t}\}$ are valid shares of the secret $s_{t,1}$, then the decryptor can efficiently find constants $\{\omega_{t,\tau} \in \mathbb{Z}_p\}_{\tau\in J}$ satisfying $\sum_{\tau\in J} \omega_{t,\tau}\lambda_{t,\tau} = s_{t,1}$ according to the description Sec. 2.3. Actually, the method for the decryptor to find such constants is that, it

finds constants satisfying $\sum_{\tau \in J} \omega_{t,\tau} M_{t,\tau} = (1, 0, \cdots, 0)$ (So we have:

$$\sum_{\tau \in J} \omega_{t,\tau} \lambda_{t,\tau} = \sum_{\tau \in J} \omega_{t,\tau} M_{t,\tau} \cdot \overrightarrow{v_t} = (1, 0, \cdots, 0) \cdot (s_{t,1}, s_{t,2}, \cdots, s_{t,n_{max}}) = s_{t,1}$$

).

The decryption algorithm works as follows, it first computes

$$\begin{aligned} CT_1 &= e(C_1, K_1) e(C_2, K_2) \cdots e(C_{k'}, K_{k'}) \\ &= \prod_{t \in [k']} e(g, v_t)^{x s_{t,1} r_t} \prod_{t \in [k']} e(g, v_t)^{x s_{t,1} p_{t,1}}, \end{aligned}$$

then, it computes

$$\begin{aligned} CT_2 &= \prod_{t \in [k']} \prod_{i \in [n_{max}]} e(L_{t,i}, \prod_{\tau \in J} C_{t,i,\tau}^{\omega_{t,\tau}}) \\ &= \prod_{t \in [k']} \prod_{i \in [n_{max}]} e(v_t^{p_{t,i}}, g^{\sum_{\tau \in J} x \omega_{t,\tau} M_{\tau,i} s_{t,i}}) \times \prod_{t \in [k']} \prod_{i \in [n_{max}]} e(v_t^{p_{t,i}}, \prod_{\tau \in J} h_{t,i,\rho(\tau)}^{-s_{t,1} \omega_{t,\tau}}) \\ &= \prod_{t \in [k']} e(g, v_t)^{x s_{t,1} p_{t,1}} \prod_{t \in [k']} \prod_{i \in [n_{max}]} e(v_t^{p_{t,i}}, \prod_{\tau \in J} h_{t,i,\rho(\tau)}^{-s_{t,1} \omega_{t,\tau}}), \end{aligned}$$

it also computes:

$$\begin{aligned} CT_3 &= \prod_{t \in [k']} \prod_{\tau \in J} e(N_{t,\rho(\tau)}^{\omega_{t,\tau}}, C_t) \\ &= \prod_{t \in [k']} \prod_{\tau \in J} e(\prod_{i \in [n_{max}]} h_{t,i,\rho(\tau)}^{p_{t,i} \omega_{t,\tau}}, v_t^{s_{t,1}}) \\ &= \prod_{t \in [k']} \prod_{i \in [n_{max}]} e(v_t^{p_{t,i}}, \prod_{\tau \in J} h_{t,i,\rho(\tau)}^{s_{t,1} \omega_{t,\tau}}). \end{aligned}$$

Finally, the algorithm can recover the message m through the following computation

$$m = C_0' \cdot (CT_1 / (CT_2 \cdot CT_3))^{-1},$$

where

$$\begin{aligned} C_0' &= Dec_s(CK_s^S, C_0) \\ &= Dec_s(NSK_s^S, Enc(PK_s, \mathbb{A}, m \prod_{t \in [k']} e(g^x, v_t^{r_t})^{s_{t,1}}; \vec{r})) \\ &= m \prod_{t \in [k']} e(g^x, v_t^{r_t})^{s_{t,1}}. \end{aligned}$$

The abovementioned C_0' can be derived because of the correctness of the ABE_s, namely, the message can be recovered by the Dec_s algorithm, if the attribute set S satisfies the access structure $\mathbb{A}$.

4.1.2. Proof

Theorem 1 *The advantage for any probability polynomial time (PPT) algorithm in winning the PKFA game is negligible, if ABE_s is a selective-secure CP-ABE scheme.*

Proof. Assume there exists an adversary $\mathcal{A}$ can break the PKFA model with non-negligible advantage, we construct a challenger $\mathcal{B}$ to break the selective-security of ABE_s scheme as follows.

Init. The adversary $\mathcal{A}$ chooses a challenge access structure $\mathbb{A}^* = (M^*, \rho^*)$ and sends $\mathbb{A}^*$ to the challenger $\mathcal{B}$, where M^* is a $\ell \times n_{max}$ matrix and ρ^* is an injective function that associates rows of M^* to attributes. $\mathcal{B}$ launches a simulator C and sends the access structure $\mathbb{A}^*$ to C as the challenge access structure it chooses.

Setup. $\mathcal{B}$ queries the public key of ABE_s to $\mathcal{C}$ and receives the PK_s from $\mathcal{C}$. $\mathcal{B}$ sets $MPK = PK_s$.

CoreKey Queries 1,2. The adversary $\mathcal{A}$ can query any core key CK^S to $\mathcal{B}$, with the only limitation that the attribute set S cannot satisfy the challenge access structure $\mathbb{A}^*$. The situation is similar when $\mathcal{B}$ queries the secret key SK_s^S to $\mathcal{C}$: $\mathcal{B}$ cannot query a secret key SK_s^S to $\mathcal{C}$ where the attribute set S satisfies the challenge access structure $\mathbb{A}^*$.

Once $\mathcal{B}$ receives a valid core key query CK^S from $\mathcal{A}$, it queries the secret key SK_s^S to $\mathcal{C}$ and returns to $\mathcal{A}$ the secret key what $\mathcal{C}$ returns to it.

Challenge. At some point, $\mathcal{A}$ generates a forged public key. Let this key be

$$NPK_{k'} = \{g, g^x, \{v_t, v_t^{r_t}, h_{t,i,j}\}_{t\in[k'],i\in[n_{max}],j\in[U]}\}.$$

$\mathcal{B}$ then alerts $\mathcal{C}$ to play the "challenge" game. $\mathcal{C}$ chooses and issues two messages (m_0, m_1) with the same length, then forms a challenge ciphertext

$$CT^* = Enc_s(PK_s, \mathbb{A}^*, m_\beta; \vec{r})$$

and sends this ciphertext to $\mathcal{B}$.

$\mathcal{B}$ chooses random $\{s_{t,i}\}_{i\in[k']}$ and generates two messages

$$(m_0' = \frac{m_0}{\prod\limits_{i\in[k']} e(g^x, v_i^{r_i})^{s_{t,i}}}, m_1' = \frac{m_1}{\prod\limits_{i\in[k']} e(g^x, v_i^{r_i})^{s_{t,i}}}),$$

then it forms the following ciphertext CT'^*

$$\begin{aligned} C_0 &= CT^*, \\ C_t &= v_t^{s_{t,1}} : (t \in [k']), \\ C_{t,i,\tau} &= g^{xM^*_{\tau,i}\cdot s_{t,i}} h_{t,i,\rho^*(\tau)}^{-s_{t,1}} : (t \in [k'], i \in [n_{max}], \tau \in [\ell]). \end{aligned}$$

$\mathcal{B}$ sends (m_0', m_1') and the challenge ciphertext CT'^* to $\mathcal{A}$.

Guess. $\mathcal{A}$ will eventually issue a guess β' of β to $\mathcal{B}$. $\mathcal{B}$ then returns to $\mathcal{C}$ the β' as its guess.

We demonstrate that if $\mathcal{A}$ guesses the correct β, then so does $\mathcal{B}$. It is obvious that $\mathcal{B}$ simulates the game with $\mathcal{A}$ perfectly, excepts for an ambiguous setting of the ciphertext component C_0. Actually, we have

$$\begin{aligned} C_0 &= CT^* \\ &= Enc_s(PK_s, \mathbb{A}^*, m_\beta; \vec{r}) \\ &= Enc_s(PK_s, \mathbb{A}^*, \frac{m_\beta}{\prod_{i\in[k']} e(g^x, v_i^{r_i})^{s_{t,i}}} \cdot \prod_{i\in[k']} e(g^x, v_i^{r_i})^{s_{t,i}}; \vec{r}) \\ &= Enc_s(PK_s, \mathbb{A}^*, m'_\beta \cdot \prod_{i\in[k']} e(g^x, v_i^{r_i})^{s_{t,i}}; \vec{r}). \end{aligned}$$

Thus, C_0 is proper distributed.

Therefore, $\mathcal{B}$ breaks the selective-security of ABE_s when $\mathcal{A}$ forges the NPK. Theorem 1 holds. □

Theorem 2 *The advantage of any PPT algorithm in winning the selective-security game is negligible, if the k'-BDH assumption holds.*

Proof. If there exists an adversary $\mathcal{A}$ can selectively break the security of our SLS-CP-ABE scheme with non-negligible advantage, then there exists a challenger $\mathcal{B}$ can resolve the k'-BDH assumption with non-negligible advantage.

Init. The adversary $\mathcal{A}$ chooses a challenge access structure $\mathbb{A}^* = (M^*, \rho^*)$ and sends $\mathbb{A}^*$ to the challenger $\mathcal{B}$, where M^* is a $\ell \times n_{max}$ matrix and ρ^* is an injective function that associates rows of M^* to attributes.

Setup. $\mathcal{B}$ takes the k'-BDH challenge vector

$$\vec{z} = (g, g^x, g^y, v_1, \cdots, v_{k'}, v_1^{\hat{r}_1}, \cdots, v_{k'}^{\hat{r}_{k'}}, T)$$

as inputs. The task of $\mathcal{B}$ is to determine whether

$$T = e(g,g)^{xy(\hat{r}_1+\cdots+\hat{r}_{k'})}.$$

$\mathcal{B}$ chooses a selective-secure CP-ABE scheme $ABE_s = (Setup_s, KeyGen_s, Enc_s, Dec_s)$ and launches the $Setup_s$ algorithm to obtain the public key and master secret key PK_s, MSK_s. $\mathcal{B}$ can form any secret key of ABE_s because it knows the master secret key MSK_s.

$\mathcal{B}$ issues

$$MPK = \{PK_s\}$$

and

$$MSK = \{MSK_s\}.$$

CoreKey Queries 1, 2. The adversary $\mathcal{A}$ can issue any core key query to $\mathcal{B}$. $\mathcal{B}$ can answer all theses queries because it knows the MSK_s.

NorKey Queries 1, 2. $\mathcal{A}$ queries a $NPK^*_{k'}/NSK^{*S}_{k'}$ pair to $\mathcal{B}$, where the $NPK^*_{k'}$ is the challenge NPK that used to generate the challenge ciphertext $CT^{\mathbb{A}^*}_{k'}$. $\mathcal{B}$ chooses random elements $\{z_{t,i,j} \in \mathbb{Z}_p\}_{t\in[k'],i\in[n_{max}],j\in[U]}$ and sets the public parameters $\{h_{t,i,j}\}_{t\in[k'],i\in[n_{max}],j\in[U]}$ as follows

$$h_{t,i,j} = \begin{cases} v_t^{z_{t,i,j}} g^{xM^*_{d,i}} : & (d \in [\ell]) \wedge (\rho^*(d) = j) \\ v_t^{z_{t,i,j}} : & else. \end{cases} \quad (1)$$

$\mathcal{B}$ sets the parameter $h_{t,i,j}$ using the following trick: if the attribute j is associated with a row x in the challenging matrix M^*, then $\mathcal{B}$ sets $h_{t,i,j} = v_t^{z_{t,i,j}} g^{xM^*_{d,i}}$; otherwise, $\mathcal{B}$ simply sets $h_{t,i,j} = v_t^{z_{t,i,j}}$. $\mathcal{B}$ also chooses $\{a_t\}_{t\in[k']}$ and issues $NPK^*_{k'}$ as follows

$$NPK^*_{k'} = \{g, g^x, \{v_t, (v_t^{\hat{r}_t})^{1/a_t}, h_{t,i,j}\}_{t\in[k'],i\in[n_{max}],j\in[U]}\}.$$

$\mathcal{B}$ implicitly sets $r_t = \hat{r}_t/a_t$ for $t \in [k']$ in the above setting.

$\mathcal{B}$ finds a vector $\vec{\omega} = (\omega_1, \cdots, \omega_{n_{max}})$ satisfying that $\omega_1 = -1$ and $M^*_i \cdot \vec{w} = 0$ for all i where $\rho^*(i) \in S$. This vector must exist according to the property of LSSS [2], as long as S does not satisfy $\mathbb{A}^*$. Then, $\mathcal{B}$ chooses $\{\theta_{t,i}\}_{t\in[k'],i\in[n_{max}]} \in Z_p$ and *implicitly* sets $p_{t,i}$ as

$$p_{t,i} = \theta_{t,i} + \omega_i \hat{r}_t/a_t \quad (t \in [k'], i \in [n_{max}]). \quad (2)$$

$\mathcal{B}$ sets $\{L_{t,i}\}_{t\in[k'],i\in[n_{max}]}$ as

$$\begin{aligned} L_{t,i} &= v_t^{p_{t,i}} \\ &= v_t^{\theta_{t,i}+\omega_i\hat{r}_t/a_t} \\ &= v_t^{\theta_{t,i}}((v_t^{\hat{r}_t})^{1/a_t})^{\omega_i} \end{aligned}$$

and constructs K_t as

$$\begin{aligned} K_t &= g^{xr_t} g^{xp_{t,1}} \\ &= g^{x\hat{r}_t/a_t + x\theta_{t,1} + x\omega_1\hat{r}_t/a_t} \\ &= g^{x\theta_{t,1}}. \end{aligned}$$

Finally, $\mathcal{B}$ sets $N_{t,\chi}$ as

$$N_{t,\chi} = \begin{cases} \prod_{i\in[n_{max}]} (v_t^{\hat{r}_t})^{z_{t,i,\chi}w_i/a_t} g^{x\theta_{t,i}M^*_{d,i}} v_t^{z_{t,i,\chi}\theta_{t,i}} & (\exists d : \rho^*(d) = \chi), \\ \prod_{i\in[n_{max}]} v_t^{z_{t,i,\chi}\theta_{t,i}} (v_t^{\hat{r}_t})^{\omega_i z_{t,i,\chi}/a_t} & (\neg\exists d : \rho^*(d) = \chi). \end{cases} \quad (3)$$

where $d \in [\ell]$ denotes a row in the challenge matrix M^*. $\mathcal{B}$ sets the parameter $N_{t,\chi}$ using the following trick: if the attribute χ is associated with a row d in the challenge matrix M^*, then

$$N_{t,\chi} = (\prod_{i\in[n_{max}]} (v_t^{\hat{r}_t})^{z_{t,i,\chi}w_i/a_t} g^{x\theta_{t,i}M^*_{d,i}} v_t^{z_{t,i,\chi}\theta_{t,i}}) \cdot (g^{x\hat{r}_t/a_t})^{\sum_{i\in[n_{max}]} M^*_{d,i}w_i}. \quad (4)$$

In the above equation (4), only the term $g^{x\hat{r}_t/a_t}$ is unknown by $\mathcal{B}$, however, this term is canceled out because we have $\sum_{i\in[n_{max}]} M^*_{d,i}\omega_i = 0$, where $\exists d : \rho^*(d) = \chi$. If there exists no such row d that associated with the attribute χ, then

$$N_{t,\chi} = \prod_{i\in[n_{max}]} (v_t^{\hat{r}_t})^{z_{t,i,\chi}w_i/a_t} v_t^{z_{t,i,\chi}\theta_{t,i}}. \quad (5)$$

Finally, $\mathcal{B}$ issues the secret key as follows

$$NSK^{*S}_{k'} = \{SK^S_s, K_t, L_{t,i}, N_{t,\chi}\}_{t\in[k'],i\in[n_{max}],\chi\in S}.$$

Challenge. $\mathcal{A}$ forms two messages m_0, m_1 with the same length, and issues them to $\mathcal{B}$. $\mathcal{B}$ flips a coin $\beta \in \{0,1\}$ and forms the challenge ciphertext by encrypting the message m_β with $NPK^*_{k'}$ as follows.

$\mathcal{B}$ sets the parameter $\{C_t\}_{t\in[k']}$ as

$$C_t \;=\; v_t^{s_{t,1}} = (g^y)^{a_t} : (t \in [k']).$$

$\mathcal{B}$ implicitly sets the value $s_{t,1} = a_t y/s_t$ where $v_t = g^{s_t} : (t \in [k'])$ and s_t is some integer unknown to $\mathcal{B}$ in the above setting. We have $C_t = v_t^{s_{t,1}} = (g^{s_t})^{a_t y/s_t} = (g^y)^{a_t}$. $\mathcal{B}$ chooses random vectors $\{\vec{y_t} = (0, y_{t,2}, \cdots, y_{t,n_{max}})\}_{t\in[k']}$ and *implicitly* sets the vector $\{\vec{v_t}\}_{t\in[k']}$ as follows

$$\begin{aligned} \vec{v_t} &= \underbrace{(a_t y/s_t, a_t y/s_t, \cdots, a_t y/s_t)}_{n_{max}} + \vec{y_t} \\ &= (a_t y/s_t, a_t y/s_t + y_{t,2}, \cdots, a_t y/s_t + y_{t,n_{max}}). \end{aligned}$$

The vector $\vec{v_t}$ is properly distributed, since the vector $\vec{y_t}$ randomizes the vector $(a_t y/s_t, a_t y/s_t, \cdots, a_t y/s_t)$. $\mathcal{B}$ creates the parameter $C_{t,i,\tau}$ as follows

$$C_{t,i,\tau} = (g^x)^{M^*_{\tau,i} y_{t,i}} (g^y)^{-z_{t,i,\tau} a_t}.$$

Notice that, every attribute τ must be associated with a row ϱ_τ in the challenge matrix M^*, namely, $\rho^*(\varrho_\tau) = \tau$. Therefore,

$$h_{t,i,\tau} = v_t^{z_{t,i,\tau}} g^{x M^*_{\varrho\tau,i}} : (t \in [k'], i \in [n_{max}], \rho^*(\varrho_\tau) = \tau).$$

$\mathcal{B}$ forms the ciphertext $C_{t,i,\tau}$ by the following computation

$$\begin{aligned} C_{t,i,\tau} &= g^{x M^*_{\varrho\tau,i} a_t y/s_t} g^{x M^*_{\varrho\tau,i} y_{t,i}} \times v_t^{-z_{t,i,\tau} a_t y/s_t} g^{-x M^*_{\varrho\tau,i} a_t y/s_t} \\ &= (g^x)^{M^*_{\varrho\tau,i} y_{t,i}} (g^y)^{-z_{t,i,\tau} a_t}. \end{aligned}$$

$\mathcal{B}$ chooses a random vector $\vec{r}^*$ to encrypt the challenge ciphertext, then forms the following ciphertext:

$$C_0 = Enc_s(PK_s, \mathbb{A}^*, m_\beta \cdot T; \vec{r}^*).$$

Guess. $\mathcal{A}$ will eventually output a guess β' of β. $\mathcal{B}$ outputs 0 when $\mathcal{A}$ outputs 0; otherwise, $\mathcal{B}$ outputs 1.

If $T = e(g,g)^{xy(\hat{r}_1 + \hat{r}_2 + \cdots + \hat{r}_{k'})}$, then $\mathcal{B}$ simulates the game perfectly with $\mathcal{A}$ because

$$\begin{aligned} \prod_{t \in [k']} e(g^x, v_t^{r_t})^{s_{t,1}} &= \prod_{t \in [k']} e(g^x, (g^{s_t})^{\hat{r}_t/a_t})^{a_t y/s_t} \\ &= e(g,g)^{xy(\hat{r}_1 + \hat{r}_2 + \cdots + \hat{r}_{k'})}. \end{aligned}$$

Therefore, $\mathcal{B}$ resolves the k'-BDH assumption when $\mathcal{A}$ breaks the SLS-CP-ABE scheme with non-negligible advantage. Otherwise, T is a random element in $\mathbb{G}_T$, then the message encrypted by $\mathcal{B}$ is a random message. $\mathcal{A}$ has to "guess" the coin β and gains no advantage in breaking our CP-ABE scheme. □

4.2. Generic Construction of SLS-KP-ABE

In this section, we present our SLS-KP-ABE scheme using the similar technique of the SLS-CP-ABE scheme.

4.2.1. Our Construction

Setup(U). The setup algorithm takes the number U of attributes as inputs, it chooses a selective-secure KP-ABE scheme (denoted as ABE_s)

$$ABE_s = (Setup_s, KeyGen_s, Enc_s, Dec_s)$$

and executes the $Setup_s$ algorithm to obtain the public key PK_s and master secret key MSK_s. The MPK is issued as

$$MPK = \{PK_s\}$$

and The MSK is

$$MSK = \{MSK_s\}.$$

CoreKeyGen$(MSK, \mathbb{A})$. The algorithm takes the MSK and the access structure $\mathbb{A}$ as inputs, it calls the $KeyGen_s$ algorithm and achieves the secret key $SK_s^{\mathbb{A}}$, it outputs the core key $CK^{\mathbb{A}}$ as

$$CK^{\mathbb{A}} = \{SK_s^{\mathbb{A}}\}.$$

NorKeyGen$(\mathbb{A}, CK^{\mathbb{A}}, k')$. The NorKeyGen algorithm takes the access structure $\mathbb{A} = (M, \rho)$, the $CK^{\mathbb{A}}$, and a security parameter k' as inputs, it chooses a group $\mathbb{G}$ of prime order p, generators $(g, \{v_t\}_{t\in[k']}) \in \mathbb{G}^{k'+1}$, $(x, \{r_t\}_{t\in[k']}) \in Z_p^{k'+1}$ and random elements $\{h_{t,j}\}_{t\in[k'],j\in[U]}$, then it generates the public key as

$$NPK_{k'} = \{g, g^x, \{v_t, v_t^{r_t}, h_{t,j}\}_{t\in[k'],j\in[U]}\}.$$

Let M be a $\ell \times n_{max}$ access structure matrix, the algorithm chooses k' random vectors $\{\vec{\varphi_t} = (\alpha_{t,1}, \alpha_{t,2}, \cdots, \alpha_{t,n_{max}})\}_{t\in[k']}$, where the elements $\{\alpha_{t,2}, \cdots, \alpha_{t,n_{max}}\}_{t\in[k']}$ are random chosen, and

$$\{\alpha_{t,1} = r_t\}_{t\in[k']}$$

are *implicitly* set by the algorithm. The above k' vectors will be used to share the k' exponents $(r_1, \cdots, r'_k)$. The algorithm chooses random elements $\{\eta_{t,\tau}\}_{t\in[k'],\tau\in[\ell]}$ and computes

$$\begin{aligned} K_{1,t,\tau} &= g^{x\langle M_\tau \cdot \vec{\varphi_t}\rangle} h_{t,\rho(\tau)}^{-\eta_{t,\tau}} : (t \in [k], \tau \in [\ell]) \\ K_{2,t,\tau} &= v_t^{\eta_{t,\tau}} : (t \in [k'], \tau \in [\ell]), \end{aligned}$$

where $\langle M_\tau \cdot \vec{\varphi_t} \rangle$ denotes the inner product of vector $\vec{M_\tau}$ (the τ^{th} row of matrix M) and vector $\vec{\varphi_t}$.

The secret key associated with the access structure $\mathbb{A}$ is issued as follows

$$SK_{k'}^{\mathbb{A}} = \{CK^{\mathbb{A}}, K_{1,t,\tau}, K_{2,t,\tau}\}_{t\in[k'],\tau\in[\ell]}.$$

Encrypt$(MPK, NPK_{k'}, S, m)$ The encrypt algorithm takes the MPK, the $NPK_{k'}$, the attribute set S and a message m as input. It chooses k' random elements $s_t \in Z_p, (t \in [k'])$ and a vector $\vec{r}$ used for the encryption of Enc_s algorithm. The algorithm computes

$$\begin{aligned} C_0 &= m \cdot Enc_s(MPK, S, m \prod_{t\in[k']} e(g^x, v_t^{r_t})^{s_t}; \vec{r}) \\ C_t &= v_t^{s_t} : (t \in [k']) \\ C_{t,\chi} &= h_{t,\chi}^{s_t} : (t \in [k'], \chi \in S). \end{aligned}$$

The ciphertext is published as:

$$CT_{k'}^{S} = (C_0, \{C_t, C_{t,\chi}\}_{t\in[k'],\chi\in S}).$$

Decrypt$(CT_{k'}^{S}, SK_{k'}^{\mathbb{A}})$ The decrypt algorithm takes a ciphertext $CT_{k'}^{S}$ and a private key $SK_{k'}^{\mathbb{A}}$ as inputs. Assume the attributes set S satisfies the access structure $\mathbb{A}$, we define a set J as $J = \{\tau : \rho(\tau) \in S\}$. If the terms $\{\lambda_{t,\tau}\}$ are valid shares of secret r_t for $t \in [k']$, the algorithm can find a constants set $\{\omega_{t,\tau} \in \mathbb{Z}_p\}_{\tau\in J}$ such that $\sum_{\tau\in J} \omega_{t,\tau}\lambda_{t,\tau} = r_t$.

The algorithm first computes

$$\begin{aligned} CT_1 &= \prod_{t\in[k]} e(C_t, \prod_{\tau\in J} K_{1,t,\tau}^{\omega_{t,\tau}}) \\ &= \prod_{t\in[k]} e(v_t^{s_t}, \prod_{\tau\in J} g^{x\omega_{t,\tau}\langle M_\tau\cdot\vec{\varphi_t}\rangle} h_{t,\rho(\tau)}^{-\omega_{t,\tau}\eta_{t,\tau}}) \\ &= \prod_{t\in[k]} e(v_t^{s_t}, g^{\sum_{\tau\in J} x\omega_{t,\tau}\langle M_\tau\cdot\vec{\varphi_t}\rangle}) \prod_{t\in[k]} e(v_t^{s_t}, \prod_{\tau\in J} h_{t,\rho(\tau)}^{-\eta_{t,\tau}\omega_{t,\tau}}) \\ &= \prod_{t\in[k]} e(v_t^{r_t}, g^x)^{s_t} \prod_{t\in[k]} \prod_{\tau\in J} e(v_t^{s_t}, h_{t,\rho(\tau)}^{\omega_{t,\tau}\eta_{t,\tau}})^{-1}. \end{aligned}$$

Then, the algorithm computes

$$\begin{aligned} CT_2 &= \prod_{t\in[k]} \prod_{\tau\in J} e(K_{2,t,\tau}, C_{t,\rho(\tau)}^{\omega_{t,\tau}}) \\ &= \prod_{t\in[k]} \prod_{\tau\in J} e(v_t^{\eta_{t,\tau}}, h_{t,\rho(\tau)}^{s_t\omega_{t,\tau}}) \\ &= \prod_{t\in[k]} \prod_{\tau\in J} e(v_t^{s_t}, h_{t,\rho(\tau)}^{\omega_{t,\tau}\eta_{t,\tau}}). \end{aligned}$$

Finally, the algorithm can recover the message m through the following computation

$$m = C'_0 \cdot (CT_1 \cdot CT_2)^{-1}.$$

where

$$\begin{aligned} C'_0 &= Dec_s(SK_s^{\mathbb{A}}, C_0) \\ &= Dec_s(SK_s^{\mathbb{A}}, Enc(PK_s, S, m \prod_{t\in[k']} e(g^x, v_t^{r_t})^{s_t}; \vec{r})) \\ &= m \prod_{t\in[k']} e(g^x, v_t^{r_t})^{s_t}. \end{aligned}$$

4.2.2. Proof

Theorem 3 *The advantage for any PPT algorithm in winning the PKFA game is negligible, if ABE_s is a selective-secure KP-ABE scheme.*

We omit the proof because it is quite similar to the proof of theorem 1.

Theorem 4 *The advantage for any PPT algorithm in winning the selective-security game is negligible, if the k'-BDH assumption holds.*

Proof. If there exists an adversary $\mathcal{A}$ can break the selective-security of our SLS-KP-ABE with non-negligible advantage, then there exists a challenger $\mathcal{B}$ can resolve the k'-BDH assumption with non-negligible advantage.

Init. $\mathcal{A}$ chooses a challenge attribute set S^* and sends it to $\mathcal{B}$.

Setup. $\mathcal{B}$ takes the k'-BDH challenge vector

$$\vec{z} = (g, g^x, g^y, v_1, \cdots, v_{k'}, v_1^{\hat{r}_1}, \cdots, v_{k'}^{\hat{r}_{k'}}, T)$$

as inputs. The task of $\mathcal{B}$ is to determine whether

$$T = e(g, g)^{xy(\hat{r}_1+\cdots+\hat{r}_{k'})}.$$

$\mathcal{B}$ chooses a selective-secure KP-ABE scheme $ABE_s = (Setup_s, KeyGen_s, Enc_s, Dec_s)$ and launches the $Setup_s$ algorithm to obtain the public key and master secret key PK_s, MSK_s. Notice that, $\mathcal{B}$ can form any secret key of ABE_s, since it knows the master secret key MSK_s.

$\mathcal{B}$ issues

$$MPK = \{PK_s\}$$

and

$$MSK = \{MSK_s\}.$$

CoreKey Queries 1, 2. The adversary $\mathcal{A}$ can query any core key to $\mathcal{B}$, and $\mathcal{B}$ can answer all theses queries because it knows the MSK_s.

NorKey Queries 1, 2. $\mathcal{B}$ must answer the normal key queries from $\mathcal{A}$. Assume $\mathcal{A}$ queries for a normal $NPK^*_{k'}/NSK^{*\mathbb{A}}_{k'}$ pair, where $NPK^*_{k'}$ is the challenge NPK used to encrypt the challenge message. $\mathcal{B}$ forms the $NPK^*_{k'}$ and the $NSK^{*\mathbb{A}}_{k'}$ as follows.

$\mathcal{B}$ chooses random elements $\{z_{t,j} \in \mathbb{Z}_p\}_{t\in[k'],j\in[U]}$ and sets the public parameters $\{h_{t,j}\}_{t\in[k'],j\in[U]}$ as follows

$$h_{t,j} = \begin{cases} v_t^{z_{t,j}} g^x & (j \notin S^*) \\ v_t^{z_{t,j}} & (j \in S^*). \end{cases} \quad (6)$$

Additionally, $\mathcal{B}$ chooses $\{a_t\}_{t\in[k']}$, and issues $NPK^*_{k'}$ as follows

$$NPK^*_{k'} = (g, g^x, \{v_t, (v_t^{\hat{r}_t})^{1/a_t}, h_{t,j}\}_{t\in[k'],j\in[U]}).$$

We note that $\mathcal{B}$ implicitly sets $r_t = \hat{r}_t/a_t$ for $t \in [k']$ in the above setting.

$\mathcal{B}$ sets the secret key $NSK^{*\mathbb{A}}_{k'}$ as follows (let $\mathbb{A} = (M, \rho)$). It finds a vector $\vec{\omega} = (\omega_1, \cdots, \omega_{n_{max}})$ satisfying that $\omega_1 = 1$ and $M_i \cdot \vec{w} = 0$ for all i where $\rho(i) \in S^*$. This vector must exist according to the property of LSSS [2], as long as the challenge attributes set S^* dose not satisfy the access structure $\mathbb{A}$. Define a vector $\vec{y_t}$ as $\vec{y_t} = (\omega_1 r_t/a_t, \omega_2 r_t/a_t, ..., \omega_{n_{max}} r_t/a_t)$, $\mathcal{B}$ chooses random vectors $\vec{y'_t} = (0, y_{t,2}, \cdots, y_{t,n_{max}})$ and *implicitly* sets the vector $\vec{\varphi_t}$ as follows

$$\begin{aligned} \vec{\varphi_t} &= \vec{y_t} + \vec{y'_t} \\ &= (\hat{r}_t/a_t, \omega_2 \hat{r}_t/a_t + y_{t,2}, \cdots, \omega_{n_{max}} \hat{r}_t/a_t + y_{t,n_{max}}) : (t \in [k']). \end{aligned}$$

The vector $\vec{\varphi_t}$ is properly distributed, since the vector $\vec{y'_t}$ randomizes the vector $(\omega_1 \hat{r}_t/a_t, \omega_2 \hat{r}_t/a_t, \cdots, \omega_{n_{max}} \hat{r}_t/a_t)$. The above setting sets $\alpha_{t,1} = r_t : (t \in [k'])$, because $\alpha_{t,i} = \omega_i \hat{r}_t/a_t + y'_{t,i}$ and $\omega_1 = 1, y'_{t,1} = 0, r_t = \hat{r}_t/a_t$.

The tricks of constructing $K_{t,\tau}$ can be classified into the following two categories: $\rho(\tau) \in S^*$ and $\rho(\tau) \notin S^*$.

(i) If $\rho(\tau) \in S^*$, $\mathcal{B}$ chooses random $m_{t,\tau} \in Z_p$ and sets $\eta_{t,\tau} = m_{t,\tau}$. The unknown term $g^{x\hat{r}_t/a_t}$ for $\mathcal{B}$ will not appear in $K_{t,\tau}$ because $\langle M_\tau \cdot \vec{w} \rangle = 0$. We provide the detailed construction of $K_{1,t,\tau}, K_{2,t,\tau}$ as follows

$$\begin{aligned} K_{1,t,\tau} &= g^{x\langle M_\tau \vec{\varphi_t} \rangle} h_{t,\rho(\tau)}^{-\eta_{t,\tau}} \\ &= g^{x(\langle M_\tau \vec{y_t} \rangle + \langle M_\tau \vec{y'_t} \rangle)} h_{t,\rho(\tau)}^{-\eta_{t,\tau}} \\ &= (g^{xr_t/a_t})^{\langle M_\tau \vec{\omega} \rangle} (g^x)^{\langle M_\tau \vec{y'_t} \rangle} h_{t,\rho(\tau)}^{-m_{t,\tau}} \\ &= (g^x)^{\langle M_\tau \vec{y'_t} \rangle} v_t^{-z_{t,\rho(\tau)} m_{t,\tau}}, \\ K_{2,t,\tau} &= v_t^{\eta_{t,\tau}} = v_t^{m_{t,\tau}}. \end{aligned}$$

(ii) If $\rho(\tau) \notin S^*$, $\mathcal{B}$ *implicitly* sets $\eta_{t,\tau} = \langle M_\tau \cdot \vec{\omega} \rangle r_t/a_t$. The unknown term $g^{x\hat{r}_t/a_t}$ for $\mathcal{B}$ may appear because $\langle M_\tau \cdot \vec{w} \rangle \neq 0$. However, this term can be canceled out by the term of $h_{t,\rho(\tau)}^{-\eta_{t,\tau}}$ as follows

$$\begin{aligned}
K_{1,t,\tau} &= g^{x\langle M_\tau \vec{\varphi_t} \rangle} h_{t,\rho(\tau)}^{-\eta_{t,\tau}} \\
&= g^{x(\langle M_\tau \vec{y_t} \rangle + \langle M_\tau \vec{y_t'} \rangle)} h_{t,\rho(\tau)}^{-\eta_{t,\tau}} \\
&= (g^{xr_t/a_t})^{\langle M_\tau \vec{\omega} \rangle} (g^x)^{\langle M_\tau \vec{y_t'} \rangle} (g^x v_t^{z_{t,\rho(\tau)}})^{-\langle M_\tau \cdot \vec{\omega} \rangle r_t/a_t} \\
&= (g^x)^{\langle M_\tau \vec{y_t'} \rangle} (v_t^{r_t/a_t})^{-\langle M_\tau \cdot \vec{\omega} \rangle z_{t,\rho(\tau)}}, \\
K_{2,t,\tau} &= v_t^{\eta_{t,\tau}} = (v_t^{r_t/a_t})^{\langle M_\tau \vec{\omega} \rangle z_{t,\rho(\tau)}}.
\end{aligned}$$

$\mathcal{B}$ issues the secret key as follows

$$NSK_{k'}^{*\mathbb{A}} = \{SK_s^{\mathbb{A}}, K_{1,t,\tau}, K_{2,t,\tau}\}_{t \in [k'], \tau \in [\ell]}.$$

Challenge. At some point, $\mathcal{A}$ outputs two messages (m_0, m_1) with the same lengths and gives them to $\mathcal{B}$. $\mathcal{B}$ sets $C_t : (t \in [k'])$ as

$$C_t = v_t^{s_t} = (g^y)^{a_t} : (t \in [k']).$$

Suppose $\{v_t = g^{\gamma_t}\}_{t \in [k']}$ for some γ_t unknown to $\mathcal{B}$, then $\mathcal{B}$ implicitly sets $s_t = a_t y/\gamma_t$ in the above equation.

$\mathcal{B}$ creates $C_{t,\chi}$ as follows

$$\begin{aligned}
C_{t,\chi} &= h_{t,\chi}^{s_t} \\
&= (g^{\gamma_t})^{z_{t,\chi} \cdot a_t y/\gamma_t} \\
&= (g^y)^{z_{t,\chi} a_t}.
\end{aligned}$$

We have $h_{t,\chi} = v_t^{z_{t,j}} = (g^{\gamma_t})^{z_{t,\chi}}$ in the above setting because $\chi \in S^*$.

$\mathcal{B}$ flips a coin $\beta \in \{0,1\}$, chooses a random vector $\vec{r}^*$ to encrypt the challenge ciphertext, then forms the following ciphertext:

$$C_0 = Enc_s(PK_s, S^*, m_\beta \cdot T; \vec{r}^*).$$

Guess. $\mathcal{A}$ will eventually output a guess β' of β. $\mathcal{B}$ outputs 0 when $\mathcal{A}$ outputs 0; otherwise, $\mathcal{B}$ outputs 1.

If $T = e(g,g)^{xy(\hat{r}_1 + \hat{r}_2 + \cdots + \hat{r}_{k'})}$, then $\mathcal{B}$ simulates the game perfectly with $\mathcal{A}$, because

$$\begin{aligned}
\prod_{t \in [k']} e(g^x, v_t^{r_t})^{s_t} &= \prod_{t \in [k']} e(g^x, (g^{\gamma_t})^{\hat{r}_t/a_t})^{a_t y/\gamma_t} \\
&= e(g,g)^{xy(\hat{r}_1 + \hat{r}_2 + \cdots + \hat{r}_{k'})}.
\end{aligned}$$

System	Ciphertext Size	Key Size	Dec. Time	Assumption
Sec. 4.2	$\mathcal{O}(k'n^2)$	$\mathcal{O}(k'k_{max}A)$	$\mathcal{O}(k'nT)$	k'-BDH
Sec. 4.3	$\mathcal{O}(k'A^2)$	$\mathcal{O}(k'n_{max}k_{max}A)$	$\mathcal{O}(k'nT)$	k'-BDH
Sec.3 in [19]	$\mathcal{O}(n)$	$\mathcal{O}(A)$	$\mathcal{O}(T)$	q-Parallel DBDHE
Sec.5 in[19]	$\mathcal{O}(n)$	$\mathcal{O}(k_{max}A)$	$\mathcal{O}(T)$	q-DBDHE
Sec.6 in [19]	$\mathcal{O}(n^2)$	$\mathcal{O}(k_{max}A + n_{max})$	$\mathcal{O}(nT)$	DBDH

Table 1: The Performance Comparisons of ABE Schemes

So, $\mathcal{B}$ resolves the k'-BDH assumption when $\mathcal{A}$ breaks our SLS-KP-ABE scheme with non-negligible advantage. Otherwise, T is a random element in $\mathbb{G}_T$, then the message encrypted by $\mathcal{B}$ is a random message. $\mathcal{A}$ has to "guess" the coin β and gains no advantage in breaking our SLS-KP-ABE scheme. □

5. Discussion

We analyse the performances of our two schemes with Waters' three CP-ABE schemes [19]. Let n be the size of access formula, A the number of attributes in the user's private key, k_{max} the maximum number of times a single attribute may appear in an access formula, n_{max} the number of columns, T the minimum number of satisfied nodes of the formula. The performances for the ABE schemes are shown in table 1.

We compare the performances of our ABE schemes to the Waters' CP-ABE schemes: our two schemes are both built upon the k'-BDH assumption, while the three CP-ABE schemes proposed by Waters are built upon the q-Parallel DBDHE assumption (q-type DBDH assumption), q-DBDHE assumption (q-type DBDH assumption) and DBDH assumption, separately.

First, we observe that ABEs based on q-type DBDH assumption are more efficient: the performances of scheme 4 and scheme 5 are better than any other scheme. However, we must construct ABEs (e.g., the schemes 1,2,3 in table 1) under weaker assumption, when the security needs to be strengthened .

We analyse the schemes 1,2,3 in table 1, we can observe from the table 1 that when $k' = 1$, our two schemes have the same storage performance and time performances with the Waters' scheme. When k' is increased, our schemes' performances becomes approximately k' times the Waters' scheme, however, our ABE systems are built upon the k'-BDH assumption, which is weaker than the DBDH assumption.

We provide a methodology to construct an ABE framework on generating different security-level ABE systems. The price paid for the security enhancement is reduced efficiency. However, the system security should take precedence over the efficiency, when the security is threatened.

6. Conclusion

We mainly do the following work in this article:

(1) The CP-ABE and KP-ABE schemes reduced upon the k-BDH assumption are constructed.

(2) Considering that the IBE scheme in [3] is not flexible enough when the security-level is switched, we first propose a new security-level switchable method: the master public parameters do not need to be changed when the security-level is switched. Users can freely publish their normal public keys with different security-levels. Our approach makes the k-BDH assumption more practical: users can define different security-levels all by themselves, so they can guarantee the confidentiality of different kinds of files.

(3) We propose a new security model, PKFA. PKFA captures the behavior of adversaries in the system to issue forged public keys. By using the core key technology, we can eliminate this type of attack from adversaries.

Acknowledgements

This work is supported by National Key R and D Program of China (No.2017YFB0802000), National Natural Science Foundation of China(61772147, 61300204, 61702197); Humanities and social science research project of Ministry of Education(15YJCZH029); Project of "the 13th Five-year Plan" for the Development of Philosophy and Social Sciences in Guangzhou(2017GZQN05); Guangdong Province Natural Science Foundation of major basic research and Cultivation project(2015A030308016); Natural Science Foundation of Guangdong Province(2017A030310261); Colleges And Universities Innovation Team Construction Project Guangdong province(2015KCXTD014); National Cryptography Development Fund(MMJJ20170117); Guangzhou City Bureau of cooperative innovation project(1201610005) and the Science and Technology Planning Project of Guangdong Province (2017A020208054).

References

References

[1] Nuttapong Attrapadung, Benoit Libert, and Elie de Panafieu. Expressive key-policy attribute based encryption with constant-size ciphertexts. *Public Key Cryptography*, 6571:90–108, 2011.

[2] Amos Beimel. *Secure Schemes for Secret Sharing and Key Distribution*. PhD thesis, Technion, Haifa, Israel, 1996.

[3] Karyn Benson, Hovav Shacham, and Brent Waters. The k-bdh assumption family: Bilinear map cryptography from progressively weaker assumptions. *Topics in Cryptology C CT-RSA 2013*, 7779:310–325, 2013.

[4] John Bethencourt, Amit Sahai, and Brent Waters. Ciphertext-policy attribute-based encryption. *IEEE Symposium on Security and Privacy*, pages 321–334, 2007.

[5] Dan Boneh and Xavier Boyen. Efficient selective-id secure identity-based encryption without random oracles. In *Proceedings of Eurocrypt 2004*, volume 3027, pages 223–238, 2004.

[6] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. *CRYPTO*, 2139:213–229, 2001.

[7] Melissa Chase. Multi-authority attribute based encryption. In Salil P. Vadhan, editor, *TCC*, volume 4392 of *Lecture Notes in Computer Science*, pages 515–534. Springer, 2007.

[8] Jung Hee Cheon. Security analysis of the strong diffie-hellman problem. In Serge Vaudenay, editor, *EUROCRYPT*, volume 4004 of *Lecture Notes in Computer Science*, pages 1–11. Springer, 2006.

[9] Clifford Cocks. An identity based encryption scheme based on quadratic residues. *IMA Int.Conf.*, 7:360–363, 2001.

[10] Hua Deng, Qianhong Wu, Bo Qin, Josep Domingo-Ferrer, Lei Zhang, Jianwei Liu, and Wenchang Shi. Ciphertext-policy hierarchical attribute-based encryption with short ciphertexts. *Information Sciences*, 275:370–384, 2014.

[11] V. Goyal, A. Jain, O. Pandey, and A. Sahai. Bounded ciphertext policy attribute based encryption. *Proceedings of the ICALP*, pages 579–591, 2008.

[12] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. *ACM Conference on Computer and Communications Security*, pages 89–98, 2006.

[13] Susan Hohenberger and Brent Waters. Attribute-based encryption with fast decryption. *IACR Cryptology ePrint Archive*, 2013:265, 2013.

[14] L. Ibraimi, M. Petkovic, S. Nikova, P. Hartel, and W. Jonker. Mediated ciphertext-policy attribute-based encryption and its application. *Information Security Applications*, 5932:309–323, 2009.

[15] Rafail Ostrovsky, Amit Sahai, and Brent Waters. Attribute-based encryption with non-monotonic access structures. *IACR Cryptology ePrint Archive*, 2007:323, 2007.

[16] Yannis Rouselakis and Brent Waters. New constructions and proof methods for large universe attribute-based encryption. *IACR Cryptology ePrint Archive*, 2012:583, 2012.

[17] Amit Sahai and Brent Waters. Fuzzy identity-based encryption. *EUROCRYPT*, 3494:457–473, 2005.

[18] Adi Shamir. Identity-based cryptosystems and signature schemes. *Advances in Cryptology: Proceedings of CRYPTO 84*, 7:47–53, 1984.

[19] Brent Waters. Ciphertext policy attribute based encryption : An expressive, efficient, and provably secure realization. *Cryptology ePrint report*, 2008/290.