

# Flexible Attribute-Based Encryption, Revisited

Seiko ARITA<sup>†</sup>

## Abstract

Flexible attribute-based encryption, which is a variant of ciphertext-policy attribute-based encryption, allows one to *loosen* a decryption policy underlying a given ciphertext, if one knows some system-wide trapdoor information, without knowing its underlying plaintext message. We give new security definition for flexible attribute-based encryption. Our new definition is stronger and more natural in the sense that it allows adversaries to issue the challenge ciphertext to the loosening oracle. Second, we construct a concrete construction of flexible attribute-based encryption using composite-order pairing map. We prove its security (in our new definition) in the random oracle model against static adversaries employing the dual encryption method.

**Keywords:** Attribute-based encryption, Re-encryption, Loosening operation.

## 1 Introduction

Attribute-based encryption (ABE) was first proposed by Sahai and Waters [22], in which, a message  $m$  is encrypted to a ciphertext  $c$  under some predicate  $f$ , and a user with credential  $X$  can decrypt the ciphertext  $c$  if and only if the predicate  $f$  is satisfied by the user's credential  $X$ :  $f(X) = 1$ . Goyal, Pandey, Sahai, and Waters [7] proposed two complementary forms of ABE: Key-Policy ABE and Ciphertext-Policy ABE. In this paper, we focus on Ciphertext-Policy ABE, in which attributes are used to describe users' credentials and formulas over these attributes are attached to the ciphertext by the encrypting party.

The first construction of Ciphertext-Policy ABE was given by Bethencourt, Sahai, and Waters [5]. Its security is proved under the generic bilinear group model. Waters [25] gives an ABE construction which can be proven secure in the standard model against selective adversaries. Lewko, Okamoto, Sahai, Takashima, and Waters [16] and Okamoto and Takashima [20] give fully secure constructions of ABE in the standard model.

Many systems are proposed to secure storage services by using such ABE constructions [21, 23, 3, 2, 11, 24, 12, 9]. In those systems, users can encrypt their documents by describing decryption policies using some set of attributes, without knowing explicit

---

<sup>†</sup>Graduate School of Information Security, Institute of Information Security

identities of entities that satisfy those policies. This would enable fine access control over documents in storage without heavy load of administrations of key managements.

Ordinal ABE schemes assume static policies. If a document is once encrypted to some ciphertext under a policy  $f$ , there is no means to loosen the underlying policy  $f$  of the ciphertext without decrypting it, of course. However, in real systems, decryption policies are not always static: Yesterday's secret is not necessarily secret of today. Arita [1] proposed a notion of flexible attribute-based encryption (fABE), motivated by problems of static decryption policies. Flexible attribute-based encryption, which is a variant of ciphertext-policy ABE, allows one to *loosen* a decryption policy underlying a given ciphertext, if one knows some system-wide trapdoor information, without knowing its underlying plaintext message.

**Our contributions.** First, we reconsider the security condition required for such flexible ABE. In the security definition of *indistinguishability under loosening operation* (IND-LSO) for fABE in [1], the challenge ciphertext is required to be hiding its plaintext even if adversaries can use loosen-oracle services. (An adversary can issue some ciphertext  $c_f$  and policy difference  $\Delta f$  to loosen-oracle that replies to the adversary a loosened ciphertext  $c_{\text{or}(f, \Delta f)}$  that can be decrypted under the loosened policy  $\text{or}(f, \Delta f)$ ). The problem of the definition is that an adversary is banned to issue the challenge ciphertext itself to the loosen-oracle. This is somewhat artificial and problematic since the security of challenge ciphertext should be protected as long as the adversary does not know any secret key that satisfies the loosened underlying policy of the challenge ciphertext. We will give a new, more natural definition of IND-LSO that allows adversaries to issue the challenge ciphertext to the loosening oracles.

Second, we construct a concrete construction of fABE that is secure under our new stronger security definition, based on the scheme of Waters[25]. Although the scheme in [1] is proved only in the generic bilinear group model, we can prove the security of our construction depending only on the random oracle model against static adversaries. For the sake we use the dual encryption approach [18].

**Related works.** The concept of the flexible attribute-based encryption is similar to the attribute-based proxy re-encryption [8, 14, 10, 15, 19, 6, 13]. In the attribute-base proxy re-encryption, one can generate re-encryption key  $rk_{f_1 \rightarrow f_2}$ , and by using the key  $rk_{f_1 \rightarrow f_2}$ , a ciphertext  $c_{f_1}$  for policy  $f_1$  can be re-encrypted into a ciphertext  $c_{f_2}$  for policy  $f_2$ . To generate such re-encryption key  $rk_{f_1 \rightarrow f_2}$ , the secret key  $sk_{f_1}$  for policy  $f_1$  is required. On a while, in our flexible ABE, all ciphertexts can be “loosened” using the single (system-wide) trapdoor information (which is independent of individual policies).

The ABE scheme of [6] is more similar to flexible attribute-based encryption in a sense. A user in the scheme of [6] can provide the proxy with a single transformation key that allows the proxy to translate any ABE ciphertext into an ElGamal-style ciphertext, which the user is able to decrypt under the user's secret key (with one simple exponentiation).

## 2 Preliminaries

Our construction will use a pairing map  $e : G \times G \rightarrow G_T$  over bilinear groups of composite order  $N = p_1 p_2$ . We denote its parameter generation algorithm as  $\mathbb{G} = (N = p_1 p_2, G = G_{p_1} G_{p_2}, G_T = G_{T,p_1} G_{T,p_2}, e) \leftarrow \text{GenG}(1^k)$ . Here,  $G_i$  denotes the subgroup of the source group  $G$  with prime order  $p_i$  and  $G_{T,i}$  denotes the subgroup of the target group  $G_T$  with prime order  $p_i$  ( $i = 1, 2$ ).

### 2.1 Assumptions

Regarding the composite-order pairing map, we use the following three well-known assumptions ([18]), with some adaptation to our situation.

#### 2.1.1 The decisional BDH assumption

Given a composite-order parameter  $\mathbb{G} = (N = p_1 p_2, G = G_{p_1} G_{p_2}, G_T = G_{T,p_1} G_{T,p_2}, e) \leftarrow \text{GenG}(1^k)$ , the Bilinear Diffie-Hellman (BDH) problem over  $\mathbb{G}$  asks to find  $e(g, g)^{abc}$  given  $g, g^a, g^b, g^c$ , where  $g \xleftarrow{\$} G_{p_1}$  and  $a, b, c \xleftarrow{\$} \mathbb{Z}_N$ .

The *decisional BDH assumption* over  $\mathbb{G}$  means that the (even) decisional version of the BDH problem is difficult. That is, for any PPT algorithm  $A$ , its advantage  $\mathbf{Adv}_A^{\text{dBDH}} \stackrel{\text{def}}{=} |\Pr[A(g, g^a, g^b, g^c, T_0) = 1 \mid T_0 = e(g, g)^{abc}] - \Pr[A(g, g^a, g^b, g^c, T_1) = 1 \mid T_1 \xleftarrow{\$} G_{T,p_1}]|$  must be negligible in  $k$ .

#### 2.1.2 The decisional parallel BDHE assumption

Given a composite-order parameter  $\mathbb{G} = (N = p_1 p_2, G = G_{p_1} G_{p_2}, G_T = G_{T,p_1} G_{T,p_2}, e) \leftarrow \text{GenG}(1^k)$ , the  $q$ -parallel Bilinear Diffie-Hellman Exponent (BDHE) problem asks to find  $e(g_2, g_2)^{a(q+1)^s}$  given

$$y = \begin{aligned} & (p_1, g_1, p_2, g_2, g_2^s, g_2^a, \dots, g_2^{a^q}, g_2^{a^{q+2}}, \dots, g_2^{a^{2q}}, \\ & (g_2^{sb_j}, g_2^{a/b_j}, \dots, g_2^{a^q/b_j}, g_2^{a^{q+2}/b_j}, \dots, g_2^{a^{2q}/b_j})_{j \in [q]}, \\ & (g_2^{asb_k/b_j}, \dots, g_2^{a^q sb_k/b_j})_{j, k \in [q], j \neq k}, \end{aligned}$$

where  $g_1 \xleftarrow{\$} G_{p_1}$ ,  $g_2 \xleftarrow{\$} G_{p_2}$  and  $a, s, b_1, \dots, b_q \xleftarrow{\$} \mathbb{Z}_{p_2}$ .

The *decisional parallel BDHE assumption* means that the (even) decisional version of parallel BDHE problem is difficult. I.e., for any PPT algorithm  $A$ , its advantage  $\mathbf{Adv}_A^{q\text{-dpBDH}} \stackrel{\text{def}}{=} |\Pr[A(y, T_0) = 1 \mid T_0 = e(g_2, g_2)^{a(q+1)^s}] - \Pr[A(y, T_1) = 1 \mid T_1 \xleftarrow{\$} G_{T,p_2}]|$  must be negligible in  $k$ .

### 2.1.3 The subgroup assumption

Given a composite order parameter  $\mathbb{G} = (N = p_1 p_2, G, G_T, e) \leftarrow \text{GenG}(1^k)$ , the subgroup problem between  $G_{p_1}$  and  $G_{p_2}$  asks to distinguish the two distributions  $(g_1, X_1 X_2, T_0)$  and  $(g_1, X_1 X_2, T_1)$  where  $g_1 \leftarrow G_{p_1}, X_1 X_2 \leftarrow G = G_{p_1} G_{p_2}, T_0 \leftarrow G_{p_1}$  and  $T_1 \leftarrow G = G_{p_1} G_{p_2}$ .

The *subgroup assumption* between  $G_{p_1}$  and  $G_{p_2}$  means that the subgroup problem between  $G_{p_1}$  and  $G_{p_2}$  is difficult. That is, for any PPT algorithm  $A$ , its advantage  $\text{Adv}_A^{\text{subgr}} \stackrel{\text{def}}{=} |\Pr[A(g_1, X_1 X_2, T_0) = 1] - \Pr[A(g_1, X_1 X_2, T_1) = 1]|$  must be negligible in  $k$ .

## 3 Flexible Attribute-Based Encryption

A *flexible attribute-based encryption* (fABE) is a tuple of five PPT algorithms: **Setup**, **Enc**, **Keygen**, **Dec** and **Loosen**.

- $(par, mk, lk) \leftarrow \text{Setup}(1^k)$ .  
Given a security parameter  $1^k$ , **Setup** generates a public parameter  $par$ , a master secret  $mk$  and a trapdoor information  $lk$  for loosening operation.
- $c_f \leftarrow \text{Enc}(par, m, f)$ .  
**Enc** encrypts a given message  $m$  into a ciphertext  $c_f$  under a given decryption policy  $f$ . The policy is supposed to be explicit in the ciphertext  $c_f$ .
- $d_{as} \leftarrow \text{Keygen}(par, mk, as)$ .  
**Keygen** generates a secret key  $d_{as}$  for a given attribute set  $as$ , using the master secret  $mk$ . The attribute set  $as$  is supposed to be explicit in the secret key  $d_{as}$ .
- $m \leftarrow \text{Dec}(par, c_f, d_{as})$ .  
**Dec** decrypts a given ciphertext  $c_f$  with a given secret key  $d_{as}$  and outputs a resulting message  $m$ . The output  $m$  may be an error symbol  $\perp$  when the given ciphertext  $c_f$  is not valid.
- $c'_{\text{or}(f, \Delta f)} \leftarrow \text{Loosen}(par, lk, c_f, \Delta f)$ .  
Using the trapdoor information  $lk$ , **Loosen** re-encrypts a given ciphertext  $c_f$  under some policy  $f$  to a new “loosened” ciphertext  $c'_{\text{or}(f, \Delta f)}$ , which can be decrypted by the more loosened condition  $\text{or}(f, \Delta f)$ .

**Correctness.** For any valid setup information  $(par, mk, lk) \leftarrow \text{Setup}(1^k)$  and for any valid ciphertext  $c_f \leftarrow \text{Enc}(par, f, m)$ , it must be that  $\text{Dec}(par, c_f, d_{as}) = m$  if the secret key  $d_{as}$  is generated by **Keygen** for some attribute set  $as$  that satisfies the decryption policy  $f$  underlying  $c_f$ .

If the ciphertext  $c_f$  is loosened by a policy  $\Delta f$  to a new ciphertext  $c_{\text{or}(f,\Delta f)}$  as  $c_{\text{or}(f,\Delta f)} \leftarrow \text{Loosen}(par, lk, c_f, \Delta f)$ , then we must have  $\text{Dec}(par, c_{\text{or}(f,\Delta f)}, d_{as'}) = m$  if the corresponding attribute set  $as'$  satisfies the appended policy  $\Delta f$  (or  $f$ ).

### 3.1 Security of Flexible Attribute-Based Encryption

To define security of flexible attribute-based encryption  $\text{fABE} = (\text{Setup}, \text{Enc}, \text{Keygen}, \text{Dec}, \text{Loosen})$ , we describe games using the framework of code-based games [4]. A game has an **Initialize** procedure, procedures to respond to adversary oracle queries, and a **Finalize** procedure. A game  $\text{Game}_A$  is executed with an adversary  $A$  as follows. First, **Initialize** executes, and its outputs are the inputs to  $A$ . Then  $A$  executes, its oracle queries being answered by the corresponding procedures of  $\text{Game}_A$ . When  $A$  terminates, its output becomes the input to the **Finalize** procedure. The output of the latter is called the output of the game, and we let  $y \leftarrow \text{Game}_A$  denote the event that this game output takes value  $y$ .

#### 3.1.1 Indistinguishability under Loosening Operation (IND-LSO)

Let  $A$  be an arbitrary PPT adversary against  $\text{fABE}$ . In our game  $\text{Game}_{A,\text{fABE}}^{\text{ind-lso}}(k)$ , procedures **Initialize** and **Finalize** are defined as follows.

<p><b>procedure Initialize:</b></p> <p><math>b \xleftarrow{\\$} \{0, 1\}</math></p> <p><math>(par, mk, lk) \leftarrow \text{Setup}(1^k)</math></p> <p>return <math>par</math>.</p>	<p><b>procedure Finalize</b> (<math>b'</math>):</p> <p>return <math>b' \stackrel{?}{=} b</math>.</p>
--	--

Procedures **Keygen**, **LR** and **Loosen** are used in  $\text{Game}_{A,\text{fABE}}^{\text{ind-lso}}(k)$  to answer oracle queries from  $A$ :

<p><b>procedure Keygen</b> (<math>as</math>):</p> <p>assert(<math>f^*(as) = \text{false}</math>)</p> <p><math>d_{as} \leftarrow \text{Keygen}(par, mk, as)</math></p> <p>return <math>d_{as}</math>.</p>	<p><b>procedure LR</b> (<math>f^*, m_0, m_1</math>):</p> <p>assert(<math>f^*(as) = \text{false}</math>) for <math>as</math>'s submitted to <b>Keygen</b>-oracle.</p> <p><math>c_{f^*} \leftarrow \text{Enc}(par, f^*, m_b)</math></p> <p>return <math>c_{f^*}</math>.</p>
<p><b>procedure Loosen</b> (<math>c_f, \Delta f</math>):</p> <p>If <math>c_f = c_{f^*}</math>, then assert(<math>\text{or}(f^*, \Delta f)(as) = \text{false}</math>) for <math>as</math>'s submitted to <b>Keygen</b>-oracle.</p> <p><math>c_{\text{or}(f,\Delta f)} \leftarrow \text{Loosen}(par, lk, c_f, \Delta f)</math></p> <p>If <math>c_f = c_{f^*}</math>, then update <math>c_{f^*}</math> as <math>c_{f^*} = c_{\text{or}(f,\Delta f)}</math>.</p> <p>return <math>c_{\text{or}(f,\Delta f)}</math>.</p>	

In the above, “assert( $f^*(as) = \text{false}$ )” means that one must assert that the condition  $f^*(as) = \text{false}$  holds if  $f^*$  defined. Abort if does not hold, or else continue.

In the definition of IND-LSO in [1], adversaries cannot send the target ciphertext

$c^*$  with any policy  $\Delta f$  to the Loosen oracle. We admit it as long as the minimum requirement is satisfied, i.e.,  $\text{or}(f^*, \Delta f)(as) = \text{false}$  for  $as$ 's submitted to **Keygen**-oracle.

**Definition 1** A flexible attribute-based encryption fABE is said to be *indistinguishable under loosening operation (IND-LSO)* if for an arbitrary PPT adversary  $A$  its advantage  $\text{Adv}_{A, \text{fABE}}^{\text{ind-lso}}(k) := |\Pr[\text{Game}_{A, \text{fABE}}^{\text{ind-lso}}(k) = 1] - 1/2|$  is a negligible function in  $k$ .

If adversaries  $A$  are constrained to output its challenge policy  $f^*$  as well as its loosening's  $\Delta f_1, \dots, \Delta f_m$  in advance before receiving  $par$ , we call such  $A$  *selective adversaries*. A flexible attribute-based encryption fABE is said to be *selectively indistinguishable under loosening operation (sIND-LSO)* if it is IND-LSO against all selective adversaries.

### 3.1.2 Indistinguishability under Loosening Key

In another type of game  $\text{Game}_{A, \text{fABE}}^{\text{ind-lsk}}(k)$ , following procedures are defined:

<p><b>procedure Initialize:</b></p> <p><math>b \xleftarrow{\\$} \{0, 1\}</math></p> <p><math>(par, mk, lk) \leftarrow \text{Setup}(1^k)</math></p> <p>return <math>(par, lk)</math>.</p>	<p><b>procedure Finalize</b> (<math>b'</math>):</p> <p>return <math>b' \stackrel{?}{=} b</math>.</p>	<p><b>procedure LR</b> (<math>f^*, m_0, m_1</math>):</p> <p><math>c_{f^*} \leftarrow \text{Enc}(par, f^*, m_b)</math></p> <p>return <math>c_{f^*}</math>.</p>
--	--	---

We note that **Initialize** returns a trapdoor information  $lk$  as well as a parameter  $par$  and adversaries will know the trapdoor  $lk$ .

**Definition 2** A flexible attribute-based encryption fABE is said to be *indistinguishable under loosening key (IND-LSK)* if for an arbitrary PPT adversary  $A$  its advantage  $\text{Adv}_{A, \text{fABE}}^{\text{ind-lsk}}(k) := |\Pr[\text{Game}_{A, \text{fABE}}^{\text{ind-lsk}}(k) = 1] - 1/2|$  is negligible in  $k$ .

## 4 A Concrete Construction

### 4.1 Scheme

In the followings,  $\sharp\text{Leaf}(f)$  denotes a number of leaf nodes of a given binary formula  $f$ .  $(\rho, M) \leftarrow \text{LSS}(p, f)$  denotes a transformation which converts a Boolean formula  $f$  into a linear secret sharing scheme defined by a share-generating matrix  $M$  over prime  $p$  (with its corresponding secret-restoring coefficients  $(\omega_i)_i$ ) and an assignment function  $\rho$  from rows of matrix  $M$  to the universe of attributes. For details of the conversion we refer to [17]. Predicate  $\text{IsDH}(g, g_1, g_2, g_3)$  means the tuple  $(g, g_1, g_2, g_3)$  is a Diffie-Hellman tuple, i.e.,  $g_3 = g_2^a$  and  $g_1 = g^a$  for some  $a$ . For two vectors  $a = (a_1, \dots, a_n)$  and  $b = (b_1, \dots, b_n)$ ,  $a \cdot b$  denotes their inner product:  $a \cdot b = \sum_{i=1, \dots, n} a_i b_i$ .

- Setup ( $1^k, B(k)$ ):

- Generate a group parameter:  $(N = p_1 p_2, G, G_T, e) \leftarrow \text{GenG}$ .
  - Choose  $g, w_1 (= g^{\delta_1}), w_2 (= g^{\delta_2}) \xleftarrow{\$} G_{p_1}$  and  $\alpha, a \xleftarrow{\$} \mathbb{Z}_N$ .
  - For  $i = 1$  to  $B$  choose  $\gamma_i \xleftarrow{\$} \mathbb{Z}_N$  and compute  $u_i = g^{\gamma_i}$ . Set  $\gamma = (\gamma_1, \dots, \gamma_B)$ .
  - Choose two hash functions:  $F : \{0, 1\}^* \rightarrow \{0, 1\}^B$  and  $G : \{0, 1\}^* \rightarrow \mathbb{Z}_N$ . Let  $H(S) = u_1^{F_1(S)} u_2^{F_2(S)} \dots u_B^{F_B(S)} = g^{\gamma \cdot F(S)}$  and  $I(S) = w_1 w_2^{G(S)} = g^{\delta_1 + G(S)\delta_2}$ .
  - Choose a randomness extractor  $K : \{0, 1\}^* \rightarrow \{0, 1\}^k$ .
  - Choose a symmetric encryption  $(E, D)$ .
  - Return a public parameter  $pp = (N, g, g^\alpha, e(g, g)^\alpha, w_1, w_2, u_1, u_2, \dots, u_B)$ , a master key  $mk = (pp, g^\alpha)$  and a loosening key  $lk = (pp, \gamma_1, \gamma_2, \dots, \gamma_B)$ .
- Enc  $(par, f, M)$ :
    - Assert that  $\#\text{Leaf}(f) \leq B$  and compute  $(\rho, A) \leftarrow \text{LSS}(f)$ . ( $A$ 's size be  $l \times n$ .)
    - Choose  $v = (s, v_2, \dots, v_n) \xleftarrow{\$} \mathbb{Z}_N^n$  and do the following:
      - \* Compute  $\kappa = K(e(g, g)^{\alpha s}), P = E_\kappa(M)$  and  $C = g^s$ .
      - \* For  $j = 1$  to  $l$ , choose  $r_j \xleftarrow{\$} \mathbb{Z}_N$  and compute  $C_j = (g^\alpha)^{A_j \cdot v} H(\rho(j))^{-r_j}, D_j = g^{r_j}$ .
      - \* For  $j = 1$  to  $l$ , compute  $E_j = I(f, P, C, (C_j, D_j)_{j=1, \dots, l})^{r_j}$ .
    - Return ciphertext  $c_f = (f, P, C, (C_j, D_j)_{j=1, \dots, l}, (E_j)_{j=1, \dots, l})$ .
  - Keygen  $(mk, as)$ :
    - With  $t \xleftarrow{\$} \mathbb{Z}_N$  compute  $K = g^\alpha g^{at}, L = g^t$  and for  $i \in as$  compute  $K_i = H(i)^t$ .
    - Return  $d_{as} = (as, K, L, (K_i)_{i \in as})$ .
  - Dec  $(par, c_f = (f, P, C, (C_j, D_j)_{j=1, \dots, l}, (E_j)_{j=1, \dots, l}), d_{as} = (as, K, L, (K_i)_{i \in as})$ :
    - Assert that the attribute set  $as$  satisfies the policy  $f$  (if not return  $\perp$ ) and compute  $(\rho, A) \leftarrow \text{LSS}(f)$ . ( $A$ 's size be  $l \times n$ .)
    - Compute  $(\omega_j)_{j=1, \dots, l}$  satisfying  $\sum_{\rho(j) \in as} \omega_j A_j = (1, 0, \dots, 0)$ .
    - Compute  $\chi = e(C, K) / \prod_{\rho(j) \in as} (e(C_j, L) e(D_j, K_{\rho(j)}))^{\omega_j}$  and  $\kappa = K(\chi)$ .
    - Return  $D_\kappa(P)$ .
  - Loosen  $(par, lk = \gamma, c_f = (f, P, C, (C_j, D_j)_{j=1, \dots, l}, (E_j)_{j=1, \dots, l}), \Delta f)$ :
    - Let  $f = \text{or}(f, \Delta f)$  and  $(\rho, A) \leftarrow \text{LSS}(f)$ . (the size of  $A$  be  $l \times n$ .)
    - Assert that  $\#\text{Leaf}(f) \leq B$  and for  $j = 1$  to  $l$  assert that  $\text{DH}(g, I(f, P, C, (C_j, D_j)_{j=1, \dots, l}), D_j, E_j)$ .
    - Restore  $g^{as}$  from  $(C_j D_j^{\gamma \cdot F(\rho(j))}) = g^{A_j \cdot v})_{j=1, \dots, l}$ .
    - Choose  $y_2, \dots, y_l \xleftarrow{\$} \mathbb{Z}_N$  and compute  $g^{a\lambda_j} = g^{A_j \cdot (s, y_2, \dots, y_l)}$ .

- For  $j = 1$  to  $l$ , choose  $r_j \xleftarrow{\$} \mathbb{Z}_N$  and compute  $C_j = g^{a\lambda_j} H(\rho(j))^{-r_j}$ ,  $D_j = g^{r_j}$ .
- For  $j = 1$  to  $l$ , compute  $E_j = I(f, P, C, (C_j, D_j)_{j=1, \dots, l})^{r_j}$ .
- Return the loosened ciphertext  $c_f = (f, P, C, (C_j, D_j)_{j=1, \dots, l}, (E_j)_{j=1, \dots, l})$ .

**Correctness** It is direct to see that, if computed honestly, a ciphertext  $c_f$  of a message  $M$  will be decrypted to the original message  $M$  using the secret key  $d_{as}$  for attribute set  $as$  satisfying the underlying policy  $f$  of  $c_f$ .

The Loosen algorithm computes encodings  $g^{a\lambda_j}$  of the original shares  $\lambda_j = A_j \cdot v$  included in the ciphertext  $c_f$  by using the loosening trapdoor  $\gamma$  for any  $j = 1, \dots, l$  as  $g^{a\lambda_j} = C_j D_j^{\gamma \cdot F(\rho(j))}$ . Then it can restore the encoding  $g^{as}$  of the secret  $as$  in a multiplicative way from those encodings  $g^{a\lambda_j}$  and then can compute new shares  $g^{a\lambda_j}$  of  $as$  in a multiplicative way as  $g^{a\lambda_j} = g^{aA_j \cdot (s, y_2, \dots, y_l)}$ . Thus, the resulting ciphertext  $c_f$  has quite the same distribution as ordinary ciphertexts directly produced by **Enc** with respect to the loosened-policy  $f = \mathbf{or}(f, \Delta f)$ .

## 4.2 Security

**Theorem 1** *Under the subgroup assumption between  $G_{p_1}$  and  $G_{p_1}G_{p_2}$ , the decisional parallel BDHE assumption on  $G_{p_1}G_{p_2}$  and the assumption that the symmetric encryption  $\text{SKE} = (\text{E}, \text{D})$  is one-time CPA-secure, the proposed fAB-KEM scheme is selectively IND-LSO in the random oracle model. More precisely, for any PPT adversary  $A$  against the scheme in the IND-LSO game in the random oracle model with respect to the function  $G(\cdot)$ , there exists algorithms  $B_1, B_2, B_3$  for the subgroup assumption between  $G_{p_1}$  and  $G_{p_1}G_{p_2}$ , an algorithm  $B_4$  for the decisional parallel BDHE assumption in  $G_{p_1}G_{p_2}$  and a one-time CPA adversary  $B_5$  against SKE that satisfy*

$$\text{Adv}_{A, \text{fAB-KEM}}^{\text{ind-lso}} \leq \text{Adv}_{B_1}^{\text{subgr}} + \text{Adv}_{B_2}^{\text{subgr}} + q_{\text{loose}} \cdot B \cdot \text{Adv}_{B_3}^{\text{subgr}} + \text{Adv}_{B_4}^{\text{g-dpBDH}} + \text{Adv}_{\text{SKE}, B_5}^{\text{cpa1}} + O(q_{\text{loose}} B / p_2).$$

To prove Theorem 1 we employ the dual encryption method in the form of [18]. Major difficulty in proving Theorem 1 is in the treatment of trapdoor information behind the hash function  $H$ . Because we are using the trapdoor behind  $H$  as the loosening key, we cannot use  $G_{p_1}$  components of the hash function  $H$  as the room for the simulation in proof. Instead we change ciphertexts and secret keys into semi-functional ones respectively and do the simulation by using  $G_{p_2}$  components of  $H$ .

### 4.2.1 Proof of Theorem 1

- **Game<sub>0</sub>**:

This is the original game  $\text{Game}_{A, \text{fABE}}^{\text{ind-lso}}(k)$  between an adversary  $A$  and the challenger with respect to the proposed construction of fABE.



- **Game<sub>1</sub>**:

This game differs from **Game<sub>0</sub>** only in that the challenger now answers a “semi-functional” ciphertext (instead of the normal ciphertext) in response to **LR**-query. More precisely, (only) **LR** is re-written as **LR<sub>1</sub>**:

- **procedure LR<sub>1</sub> (f<sup>\*</sup>)**:

- \*  $\text{assert}(f^*(as) = \text{false}), \text{assert}(\#\text{Leaf}(f^*) \leq B), (\rho, A) \leftarrow \text{LSS}(f^*)$  ( $A : l \times n$ )
- \*  $v = (s, v_2, \dots, v_n) \xleftarrow{\$} \mathbb{Z}_N^n, \kappa = K(e(g, g)^{\alpha s}), P = E_\kappa(M_b), C = g^s g_2^s$
- \*  $\alpha', a', \gamma', \gamma'_1, \dots, \gamma'_B, \delta'_1, \delta'_2 \xleftarrow{\$} \mathbb{Z}_N$  (Define  $H'(S) = g_2^{\gamma' \cdot F(S)}$  and  $I'(S) = g_2^{\delta'_1 + G(S)\delta'_2}$ )
- \*  $r_j \xleftarrow{\$} \mathbb{Z}_N, C_j = (g^a)^{A_j \cdot v} H(\rho(j))^{-r_j} (g_2^{a'})^{A_j \cdot v} H'(\rho(j))^{-r_j}, D_j = g^{r_j} g_2^{r_j}$  ( $j = 1, \dots, l$ )
- \*  $E_j = I(f^*, P, C, (C_j, D_j)_{j \in [l]})^{r_j} I'(f^*, P, C, (C_j, D_j)_{j \in [l]})^{r_j}$  ( $j = 1, \dots, l$ ).
- \* Return  $c_{f^*} = (f^*, P, C, (C_j, D_j)_{j=1, \dots, l}, (E_j)_{j=1, \dots, l})$ .
- \* (**Remark**) We note that  $s$  is randomly selected from  $\mathbb{Z}_N = \mathbb{Z}_{p_1} \times \mathbb{Z}_{p_2}$  so that, for example, two elements  $g^s$  and  $g_2^s$  are independently uniformly distributed in  $G_{p_1}$  and  $G_{p_2}$ , respectively. Similar for pairs  $(g^{A_j v}, g_2^{A_j v}), (g^{r_j}, g_2^{r_j})$ , and  $(g^{\delta_i}, g_2^{\delta_i})$ .

**Claim 1** *Under the subgroup assumption between  $G_{p_1}$  and  $G_{p_1}G_{p_2}$ , the outputs of **Game<sub>0</sub>** and **Game<sub>1</sub>** are indistinguishable. More precisely, for any adversary  $A$  there exists a distinguisher  $B$  for the subgroup assumption that satisfies  $\text{Adv}_B^{\text{subgr}} = |\text{Pr}[\text{Game}_0^A = 1] - \text{Pr}[\text{Game}_1^A = 1]|$ .*

**Proof** Given any adversary  $A$  in **Game<sub>0</sub>** or **Game<sub>1</sub>**, we construct a following distinguisher  $B$  for the subgroup assumption between  $G_{p_1}$  and  $G_{p_1}G_{p_2}$ :

- **B** ( $g_1, T$ ): //  $T$  is either  $g_1^s$  or  $g_1^s g_2^s$  with  $s \xleftarrow{\$} \mathbb{Z}_N$  (Note that  $g_1^s$  and  $g_2^s$  are independently uniformly distributed in  $G_{p_1}$  and  $G_{p_2}$ , respectively.)
  - (Initialize) Choose  $\alpha, a, \gamma_1, \dots, \gamma_B, \delta_1, \delta_2 \xleftarrow{\$} \mathbb{Z}_N$ , and set  $pp = (N, g = g_1, g_1^a, e(g, g)^\alpha, F, G, u_1 = g^{\gamma_1}, \dots, u_B = g^{\gamma_B}, w_1 = g^{\delta_1}, w_2 = g^{\delta_2}), mk = (pp, g^\alpha)$  and  $lk = (pp, \gamma_1, \dots, \gamma_B)$ . Invoke a copy of  $A$  on input  $pp$ .
  - (Oracle simulation) Simulate **Keygen**-oracle and **Loosen**-oracle honestly using  $mk$  and  $lk$ , respectively. For **LR**-oracle, do as follows.
    - \* Given a challenge query  $(f^*, M_0, M_1)$ , choose  $b \xleftarrow{\$} \{0, 1\}$  and let  $(\rho, A) \leftarrow \text{LSS}(f^*)$ . ( $A : l \times n$ )
    - \* Choose  $w \xleftarrow{\$} \mathbb{Z}_N^n$  with  $w_1 = 1$  and  $\tilde{r}_j \xleftarrow{\$} \mathbb{Z}_N$  for  $j = 1$  to  $l$ .
    - \* Compute  $\kappa = K(e(g_1, T)^\alpha)$  and  $P = E_\kappa(M_b)$ .

- \* Set  $C = T$  and compute  $C_j = T^{a_{A_j} \cdot w} T^{-\tilde{r}_j \gamma \cdot F(\rho(j))}$ ,  $D_j = T^{\tilde{r}_j}$  for  $j = 1$  to  $l$  and  $E_j = D_j^{\delta_1 + G(f^*, P, C, (C_j, D_j)) \delta_2}$  for  $j = 1$  to  $l$ .
  - \* Return  $c_f^* = (f^*, P, C, (C_j, D_j)_{j=1, \dots, l}, (E_j)_{j=1, \dots, l})$  to  $A$ .
- (Output) Given the output  $b'$  of  $A$ , output 1 if  $b = b'$  or 0 otherwise.

**Correctness of the simulation.** It is obvious that the above simulated view of  $A$  by  $B$  is the same as the corresponding view of  $A$  in **Game<sub>0</sub>** or **Game<sub>1</sub>**, except that the distributions of the challenge ciphertext  $c_f^*$ .

When  $T = g_1^s$ , components of the simulated  $c_f$  are distributed as:

- $\kappa = K(e(g_1, T)^\alpha) = K(e(g_1, g_1^s)^\alpha)$ ,  $P = E_\kappa(M_b)$ ,  $C = g_1^s$ .
- $C_j = (g_1^s)^{a_{A_j} \cdot w} (g_1^s)^{-\tilde{r}_j \gamma \cdot F(\rho(j))} = (g_1)^{a_{A_j} \cdot sw} (g_1^{\gamma \cdot F(\rho(j))})^{-\tilde{r}_j s} = (g_1)^{a_{A_j} \cdot v} (g_1^{\gamma \cdot F(\rho(j))})^{-r_j}$   
(Let  $v = sw$  and  $r_j = \tilde{r}_j s$ , which are independently uniform in  $\mathbb{Z}_N$ ).
- $D_j = (g_1^s)^{\tilde{r}_j} = g_1^{r_j}$ .
- $E_j = (g_1^{r_j})^{\delta_1 + G(f^*, P, C, (C_j, D_j)) \delta_2} = (g_1^{\delta_1 + G(f^*, P, C, (C_j, D_j)) \delta_2})^{r_j} = I(f^*, P, C, (C_j, D_j))^{r_j}$ .

Thus, the distribution of  $c_f^*$  is the same as the normal ciphertexts in **Game<sub>0</sub>**.

When  $T = g_1^s g_2^s$ , components of the simulated  $c_f^*$  are distributed as:

- $\kappa = K(e(g_1, g_1^s g_2^s)^\alpha) = K(e(g_1, g_1^s)^\alpha)$ ,  $P = E_\kappa(M_b)$ ,  $C = g_1^s g_2^s$ .
- $C_j = (g_1^s g_2^s)^{a_{A_j} \cdot w} (g_1^s g_2^s)^{-\tilde{r}_j \gamma \cdot F(\rho(j))} = (g_1)^{a_{A_j} \cdot sw} (g_2)^{a_{A_j} \cdot sw} (g_1^{\gamma \cdot F(\rho(j))})^{-\tilde{r}_j s} (g_2^{\gamma \cdot F(\rho(j))})^{-\tilde{r}_j s}$   
 $= (g_1)^{a_{A_j} \cdot v} (g_2)^{a_{A_j} \cdot v} (g_1^{\gamma \cdot F(\rho(j))})^{-r_j} (g_2^{\gamma \cdot F(\rho(j))})^{-r_j}$ .
- $D_j = (g_1^s g_2^s)^{\tilde{r}_j} = g_1^{r_j} g_2^{r_j}$ .
- $E_j = (g_1^{r_j} g_2^{r_j})^{\delta_1 + G(f^*, P, C, (C_j, D_j)) \delta_2} = (g_1^{\delta_1 + G(f^*, P, C, (C_j, D_j)) \delta_2})^{r_j} (g_2^{\delta_1 + G(f^*, P, C, (C_j, D_j)) \delta_2})^{r_j}$   
 $= I(f^*, P, C, (C_j, D_j))^{r_j} I'(f^*, P, C, (C_j, D_j))^{r_j}$ .

Thus, the distribution of  $c_f^*$  is the same as the semi-functional ciphertexts in **Game<sub>1</sub>**.

**The advantage.** By the above observation we see that

$$\begin{aligned}
 \text{Adv}_B^{\text{subgr}} &= |\Pr[B(g_1, T) = 1 \mid T = g_1^s] - \Pr[B = (g_1, T) \mid T = g_1^s g_2^s]| \\
 &= |\Pr[b' = b \mid \mathbf{Game}_0] - \Pr[b' = b \mid \mathbf{Game}_1]| \\
 &= |\Pr[\mathbf{Game}_0^A = 1] - \Pr[\mathbf{Game}_1^A = 1]|.
 \end{aligned}$$

□

• **Game<sub>2</sub>:**

This game differs from **Game<sub>1</sub>** only in the following two points:

- **Keygen**-oracle now answers “semi-functional” secret keys (instead of the normal secret keys) in response to *all of* the **Keygen**-queries.
- The challenge ciphertext  $c_{f^*}$  is “more-semi-functional”.

More precisely, **Keygen** and **LR<sub>1</sub>** are re-written as the following **Keygen<sub>2</sub>** and **LR<sub>2</sub>**, respectively:

- **procedure Keygen<sub>2</sub>** ( $as$ ):
  - \* assert( $f^*(as) = \text{false}$ )
  - \*  $t \xleftarrow{\$} \mathbb{Z}_N$ .
  - \*  $\alpha', a', \gamma', \gamma'_1, \dots, \gamma'_B, \delta'_1, \delta'_2 \xleftarrow{\$} \mathbb{Z}_N$  (Define  $H'(S) = g_2^{\gamma' \cdot F(S)}$ ,  $I'(S) = g_2^{\delta'_1 + G(S)\delta'_2}$ )
  - \*  $K = g^\alpha g^{at} g_2^{\alpha'} g_2^{a't}$ ,  $L = g^t g_2^t$ ,  $K_i = H(i)^t H'(i)^t$  ( $i \in as$ )
  - \* Return  $d_{as} = (as, K, L, (K_i)_{i \in as})$ .
  - \* (**Remark**) We note that  $t$  was randomly selected from  $\mathbb{Z}_N = \mathbb{Z}_{p_1} \times \mathbb{Z}_{p_2}$  so that  $g^t$  and  $g_2^t$  are independently uniformly distributed in  $G_{p_1}$  and  $G_{p_2}$ , respectively.
- **procedure LR<sub>2</sub>** ( $f^*$ ):
  - \* Only difference from **LR<sub>1</sub>** is in method of computing  $\kappa$ :  

$$\kappa = K(e(g, g)^{\alpha s} e(g_2, g_2)^{\alpha' s})$$
- (**Remark**) The random strings  $\alpha', a', \gamma', \gamma'_1, \dots, \gamma'_B, \delta'_1, \delta'_2$  are shared among procedures **Keygen<sub>2</sub>** and **LR<sub>2</sub>**.

**Claim 2** *Under the subgroup assumption between  $G_{p_1}$  and  $G_{p_1}G_{p_2}$ , the outputs of **Game<sub>1</sub>** and **Game<sub>2</sub>** are indistinguishable. More precisely, for any adversary  $A$  there exists a distinguisher  $B$  for the subgroup assumption that satisfies  $\text{Adv}_B^{\text{subgr}} = |\Pr[\text{Game}_1^A = 1] - \Pr[\text{Game}_2^A = 1]|$ .*

**Proof** Given any adversary  $A$  in **Game<sub>0</sub>** or **Game<sub>1</sub>**, we construct a following distinguisher  $B$  for the subgroup assumption between  $G_{p_1}$  and  $G_{p_1}G_{p_2}$ :

- **B** ( $g_1, X_1 X_2 = g_1^s g_2^s, T$ ): //  $T$  is either  $g_1^\alpha$  or  $g_1^\alpha g_2^\alpha$  with  $\alpha \xleftarrow{\$} \mathbb{Z}_N$  (Note that  $g_1^\alpha$  and  $g_2^\alpha$  are independently uniform in  $G_{p_1}$  and  $G_{p_2}$ , respectively.)
  - (Initialize) Choose  $a, \gamma_1, \dots, \gamma_B, \delta_1, \delta_2 \xleftarrow{\$} \mathbb{Z}_N$  and set  $pp = (N, g = g_1, g_1^\alpha, e(g, g)^\alpha = e(g_1, T), F, G, u_1 = g^{\gamma_1}, \dots, u_B = g^{\gamma_B}, w_1 = g^{\delta_1}, w_2 = g^{\delta_2})$  and  $lk = (pp, \gamma_1, \dots, \gamma_B)$ . Invoke  $A$  on input  $pp$ .
  - (Oracle simulation) Simulate **Loosen**-oracle honestly using  $lk$ . For **Keygen**-oracle query  $as$ , do as follows:
    - \* Choose  $\tilde{t} \xleftarrow{\$} \mathbb{Z}_N$ , compute  $K = T T^{a\tilde{t}}, L = T^{\tilde{t}}, (K_i = T^{\tilde{t}\gamma \cdot F(i)})_{i \in as}$  and send  $d_{as} = (K, L, (K_i)_{i \in as})$  to  $A$ .

As **LR**-oracle simulation, process challenge query  $f^*$  as follows:

- \* Choose  $b \xleftarrow{\$} \{0, 1\}$  and let  $(\rho, A) \leftarrow \text{LSS}(f^*)$ . ( $A : l \times n$ )
  - \* Choose  $w \xleftarrow{\$} \mathbb{Z}_N^n$  with  $w_1 = 1$  and  $\tilde{r}_j \xleftarrow{\$} \mathbb{Z}_N$  for  $j = 1$  to  $l$ .
  - \* Compute  $\kappa = K(e(X_1 X_2, T))$  and  $P = \mathbf{E}_\kappa(M_b)$ .
  - \* Set  $C = X_1 X_2$  and compute  $C_j = (X_1 X_2)^{a_{A_j} \cdot w} (X_1 X_2)^{-\tilde{r}_j \gamma \cdot F(\rho(j))}$ ,  $D_j = (X_1 X_2)^{\tilde{r}_j}$  for  $j = 1$  to  $l$  and  $E_j = D_j^{\delta_1 + G(f^*, P, C, (C_j, D_j)) \delta_2}$  for  $j = 1$  to  $l$ .
  - \* Send  $c_{f^*} = (f^*, P, C, (C_j, D_j)_{j=1, \dots, l}, (E_j)_{j=1, \dots, l})$  to  $A$ .
- (Output) Given the output  $b'$  of  $A$ , output 1 if  $b = b'$  or 0 otherwise.

**Correctness of the simulation by  $B$ .** The above simulated view of  $A$  inside  $B$  is the same as the corresponding view of  $A$  in **Game<sub>1</sub>** or **Game<sub>2</sub>**, except distributions of the secret keys  $sk_{as}$  and the challenge ciphertext  $c_{f^*}$ .

When  $T = g_1^\alpha$ , the simulated secret keys  $sk_{as}$  are distributed as:

- $K = g_1^\alpha (g_1^\alpha)^{a\tilde{t}} = g_1^\alpha g_1^{a\tilde{t}}$ . (Let  $t = \alpha\tilde{t}$ , uniform in  $\mathbb{Z}_N$ .)
- $L = (g_1^\alpha)^{\tilde{t}} = g_1^t$ ,  $K_i = (g_1^\alpha)^{\tilde{t}\gamma \cdot F(i)} = (g_1^{\gamma \cdot F(i)})^t = H(i)^t$ .

Thus, the distribution of  $sk_{as}$  is quite the same as normal secret keys in **Game<sub>1</sub>**. At the same time the simulated  $c_{f^*}$  is distributed as:

- $\kappa = K(e(g_1^s g_2^s, g_1^\alpha)) = K(e(g_1, g_1)^{\alpha s})$ ,  $P = \mathbf{E}_\kappa(M_b)$ .
- $C = g_1^s g_2^s$ ,  $C_j = (g_1^s g_2^s)^{a_{A_j} \cdot w} (g_1^s g_2^s)^{-\tilde{r}_j \gamma \cdot F(\rho(j))} = (g_1^a)^{A_j \cdot v} (g_1^{\gamma \cdot F(\rho(j))})^{-r_j} (g_2^a)^{A_j \cdot v} (g_2^{\gamma \cdot F(\rho(j))})^{-r_j}$ . (Let  $v = sw$  and  $r_j = \tilde{r}_j s$ , independently uniform in  $\mathbb{Z}_N$ ).
- $D_j = (g_1^s g_2^s)^{\tilde{r}_j} = g_1^{r_j} g_2^{r_j}$ .
- $E_j = (g_1^{r_j} g_2^{r_j})^{\delta_1 + G(f^*, P, C, (C_j, D_j)) \delta_2} = (g_1^{\delta_1 + G(f^*, P, C, (C_j, D_j)) \delta_2})^{r_j} (g_2^{\delta_1 + G(f^*, P, C, (C_j, D_j)) \delta_2})^{r_j} = I(f^*, P, C, (C_j, D_j))^{r_j} I'(f^*, P, C, (C_j, D_j))^{r_j}$

Thus, the distribution of  $c_{f^*}$  is identical to the corresponding distribution of the semi-functional ciphertext in **Game<sub>1</sub>**.

When  $T = g_1^\alpha g_2^\alpha$ , the simulated secret keys  $d_{as}$  are distributed as:

- $K = g_1^\alpha g_2^\alpha (g_1^\alpha g_2^\alpha)^{a\tilde{t}} = g_1^\alpha g_1^{a\tilde{t}} \cdot g_2^\alpha g_2^{a\tilde{t}}$ . (Let  $t = \alpha\tilde{t}$ , uniform in  $\mathbb{Z}_N$ .)
- $L = (g_1^\alpha g_2^\alpha)^{\tilde{t}} = g_1^t g_2^t$ ,  $K_i = (g_1^\alpha g_2^\alpha)^{\tilde{t}\gamma \cdot F(i)} = H(i)^t H'(i)^t$ .

Thus, the distribution of  $d_{as}$  is identical to the semi-functional secret keys in **Game<sub>2</sub>**. The simulated challenge ciphertext  $c_{f^*}$  is distributed as:

- $\kappa = K(e(g_1^s g_2^s, g_1^\alpha g_2^\alpha)) = K(e(g_1, g_1)^{\alpha s} e(g_2, g_2)^{\alpha s})$ ,  $P = \mathbf{E}_\kappa(M_b)$ .
- Other components of  $c_{f^*}$  are computed in the same way as the  $T = g_1^\alpha$  case using the above  $P$ .

Thus, the distribution of  $c_{f^*}$  is identical to the corresponding distribution of more-semi-functional ciphertexts in **Game<sub>2</sub>**.

**The advantage of  $B$ .** By the above observation we see that

$$\begin{aligned} \mathbf{Adv}_B^{\text{subgr}} &= |\Pr[B(g_1, X_1 X_2, T) = 1 \mid T = g_1^\alpha] - \Pr[B(g_1, X_1 X_2, T) = 1 \mid T = g_1^\alpha g_2^\alpha]| \\ &= |\Pr[b' = b \mid \mathbf{Game}_1] - \Pr[b' = b \mid \mathbf{Game}_2]| \\ &= |\Pr[\mathbf{Game}_1^A = 1] - \Pr[\mathbf{Game}_2^A = 1]|. \end{aligned}$$

□

• **Game<sub>3</sub>:**

This game differs from **Game<sub>2</sub>** only in the simulation of **Loosen**-oracle. The procedure **Loosen<sub>2</sub>** in **Game<sub>2</sub>** (that is the same as the original **Loosen**) is modified to the following **Loosen<sub>3</sub>**.

- **procedure Loosen<sub>3</sub>** ( $c_f, \Delta_f$ ):
  - \* If  $c_f \neq c_{f^*}$ , assert all of  $D_j$  components of  $c_f$  are inside the subgroup  $G_{p_1}$ . (If not, abort)
  - \* Call **Loosen<sub>2</sub>**( $c_f, \Delta_f$ ) and return its output.

**Claim 3** *Under the subgroup assumption between  $G_{p_1}$  and  $G_{p_1}G_{p_2}$ , the outputs of **Game<sub>2</sub>** and **Game<sub>3</sub>** are indistinguishable in the random oracle model w.r.t. the function  $G$ . More precisely, for any adversary  $A$  in the random oracle model w.r.t. the function  $G$ , there exists a distinguisher  $B$  for the subgroup assumption that satisfies  $\mathbf{Adv}_B^{\text{subgr}} \geq \frac{1}{q_{\text{loosen}}^B} |\Pr[\mathbf{Game}_2^A = 1] - \Pr[\mathbf{Game}_3^A = 1]| - O(1/p_2)$  where  $q_{\text{loosen}}$  denotes an upper bound of the number of loosen queries issued by the adversary  $A$ .*

To prove Claim 3, first we prepare an experiment **Exp<sub>1</sub>**:

• **Exp<sub>1</sub>:**

- Generate a group parameter:  $pp = (N = p_1 p_2, G, G_T, e) \leftarrow \mathbf{GenG}$ .
- Choose  $g_1 \xleftarrow{\$} G_{p_1}$ ,  $g_1^s g_2^{s'}$ ,  $g_1^\alpha g_2^{\alpha'}$   $\xleftarrow{\$} G_{p_1} G_{p_2}$ ,  $a \xleftarrow{\$} \mathbb{Z}_N$  and invoke  $A$  on input  $pp, g_1, g_1^s g_2^{s'}, g_1^\alpha g_2^{\alpha'}$  and  $A = g_1^a$
- If  $A$  halts with output  $(C, D)$ , return 1 if  $e(g_1, D) = e(A, C)$  holds and at the same time  $C$  is not in  $G_{p_1}$ , otherwise return 0.

Define  $\mathbf{Adv}_A^{\text{exp1}} = \Pr[A = 1 \text{ in } \mathbf{Exp}_1]$ . We have

**Lemma 1** *For any PPT adversary  $A$ , there exists a PPT algorithm  $B$  for the subgroup assumption that satisfies  $\mathbf{Adv}_B^{\text{subgr}} \geq \mathbf{Adv}_A^{\text{exp1}} - O(1/p_2)$ .*

**Proof** Given any PPT adversary  $A$  in **Exp<sub>1</sub>**, we construct the following algorithm  $B$  for the subgroup assumption:

- **B** ( $g_1, T$ ) //  $T$  is either  $g_1^s$  or  $g_1^s g_2^{s'}$

- Choose  $\alpha, a \xleftarrow{\$} \mathbb{Z}_N$  and invoke  $A$  on input  $(g_1, T, T^\alpha, A = g_1^a)$ .
- When  $A$  halts with output  $(C, D)$ , do the following:
  - \* Compute  $E = D/C^a$ .
  - \* If  $e(g_1, D) = e(A, C)$ , then output 1 if  $e(E, T) \neq 1$  and output 0 otherwise.
  - \* Otherwise output 0.

First note that if  $e(g_1, D) = e(A, C)$  then  $E = D/C^a \in G_{p_2}$ . So, when  $T = g_1^s$ ,  $\Pr[\mathbf{B} = 1] = 0$ .

We consider the case of  $T = g_1^s g_2^{s'}$ . In the case the simulation by  $B$  is perfect for  $A$ . Since  $a \bmod p_2$  is not in the view of  $A$ , if  $C \notin G_{p_1}$ ,  $E$  would have a random  $G_{p_2}$  component and  $e(E, T)$  would be equal to 1 only with a negligible probability  $O(1/p_2)$ . Thus,

$$\begin{aligned} \Pr[\mathbf{B} = 1] &= \Pr[e(g_1, D) = e(A, C) \text{ and } e(E, T) \neq 1] \\ &\geq \Pr[e(g_1, D) = e(A, C) \text{ and } C \notin G_{p_1}] - O(1/p_2) \\ &\geq \mathbf{Adv}_A - O(1/p_2). \end{aligned}$$

Summing up,

$$\begin{aligned} \mathbf{Adv}_B &= |\Pr[\mathbf{B} = 1 \mid T = g_1^s] - \Pr[\mathbf{B} = 1 \mid T = g_1^s g_2^{s'}]| \\ &\geq \mathbf{Adv}_A - O(1/p_2) \quad \square \end{aligned}$$

**Proof (Claim 3)** Define an event **Outside** be the event that an adversary  $A$  issues some loosen query  $(c_f, \Delta f)$  such that  $c_f \neq c_{f^*}$  and its some  $D_{j^*}$  component is outside from  $G_{p_1}$  in the game **Game<sub>2</sub>** (or **Game<sub>3</sub>**). It is clear that  $|\Pr[\mathbf{Game}_2 = 1] - \Pr[\mathbf{Game}_3 = 1]| \leq \Pr[\mathbf{Outside}]$ .

First, given any adversary  $A$  for the game **Game<sub>2</sub>** (or **Game<sub>3</sub>**) in the random oracle model w.r.t. the function  $G(\cdot)$ , we construct an adversary  $B$  for the experiment **Exp<sub>1</sub>** as described below.  $B$  will succeed in **Exp<sub>1</sub>** (almost) whenever its simulating  $A$  occurs the **Outside** event.

- **B**  $(N, g_1, X_1 X_2 = g_1^s g_2^s, Y_1 Y_2 = g_1^\alpha g_2^\alpha, A = g_1^\eta)$ :

– **(Initialize)**

- \* Choose  $a, \gamma_1, \dots, \gamma_B \xleftarrow{\$} \mathbb{Z}_N$  and set  $u_1 = g_1^{\gamma_1}, \dots, u_B = g_1^{\gamma_B}$ ,  $\gamma = (\gamma_1, \dots, \gamma_B)$ .
- \* Choose  $\eta_1, \eta_2 \xleftarrow{\$} \mathbb{Z}_N$ , set  $w_1 = g^{\eta_1} A, w_2 = A^{\eta_2}$  and choose  $i^* \xleftarrow{\$} [q_{\text{loose}}]$ .
- \* Invoke  $A$  on input  $pp = (N, g_1, g_1^a, e(g_1, g_1)^\alpha = e(g_1, Y_1 Y_2), u_1, \dots, u_B, w_1, w_2)$  and simulate responses to oracle queries from  $A$  as follows:

- **(KeyGen)** When receiving a Keygen query for attribute set  $as$ , do as follows:
  - \* Choose  $\tilde{t} \xleftarrow{\$} \mathbb{Z}_N$ , compute  $K = Y_1 Y_2 (Y_1 Y_2)^{a\tilde{t}}, L = (Y_1 Y_2)^{\tilde{t}}, (K_i = (Y_1 Y_2)^{\tilde{t}\gamma \cdot F(i)})_{i \in as}$  and send  $d_{as} = (K, L, (K_i)_{i \in as})$  to  $A$ .
- **(Challenge)** When receiving a challenge query  $(f^*, M_0, M_1)$ , respond as follows:
  - \* Choose  $b \xleftarrow{\$} \{0, 1\}$  and let  $(\rho, A) \leftarrow \text{LSS}(f^*)$ . ( $A : l \times n$ )
  - \* Choose  $w \xleftarrow{\$} \mathbb{Z}_N^n$  with  $w_1 = 1$  and  $\tilde{r}_j \xleftarrow{\$} \mathbb{Z}_N$  for  $j = 1$  to  $l$ .
  - \* Compute  $\kappa = K(e(X_1 X_2, Y_1 Y_2))$  and  $P = \mathbf{E}_\kappa(M_b)$ .
  - \* Set  $C = X_1 X_2$  and compute  $C_j = (X_1 X_2)^{a A_j \cdot w} (X_1 X_2)^{-\tilde{r}_j \gamma \cdot F(\rho(j))}, D_j = (X_1 X_2)^{\tilde{r}_j}$  for  $j = 1$  to  $l$ .
  - \* Define  $G(f^*, P, C, (C_j, D_j)_{j \in [l]}) = -\eta_2^{-1} \bmod N$  and compute  $E_j = D_j^{\eta_1}$  for  $j = 1$  to  $l$ .
  - \* Send  $c_{f^*} = (f^*, P, C, (C_j, D_j)_{j \in [l]}, (E_j)_{j=1, \dots, l})$  to  $A$ .
- **(Loosen)** When receiving a loosen query  $(c_f, \Delta f)$ , do as follows:
  - \* If this loosen query is not the  $i^*$ -th loosen query, simulate the response honestly using the trapdoor  $\gamma$ .
  - \* If  $c_f = (f, P, C, (C_j, D_j), (E_j))$  coincides to the target ciphertext  $c_{f^*}$ , abort.
  - \* Let  $f = \mathbf{or}(f, \Delta f)$  and let  $(\rho, A) \leftarrow \text{LSS}(f)$ . ( $A : l \times n$ )
  - \* Assert that  $\#\text{Leaf}(f) \leq B$  and assert that  $\text{DH}(g, I(f, P, C, (C_j, D_j)_j), D_j, E_j)$  for  $j = 1$  to  $l$ .
  - \* Choose  $j^* \xleftarrow{\$} [B]$  and output  $(D_{j^*}, (D_{j^*}^{-\eta_1} E_{j^*})^{1/(1+\eta_2 G(f, P, C, (C_j, D_j)_j))})$  and halt.

**Correctness of the simulation by  $B$ .** It is direct to see the parameter  $pp$  given to  $A$  is perfectly simulated. We check the distributions of responses of the simulated oracles.

The  $i^*$  chosen by  $B$  in its initialization step indicates the guess of the first index of loosen query  $(c_f, \Delta f)$  that occurs the **Outside** event, that is, in which some  $D_{j^*}$  components of  $c_f$  is outside from  $G_{p_1}$ . From now on, we concentrate on the case that this guess  $i^*$  as well as  $j^*$  was right.

First we examine the response of Keygen oracle. The simulated secret key  $d_{as}$  is distributed as:

- $K = g_1^\alpha g_2^\alpha (g_1^\alpha g_2^\alpha)^{a\tilde{t}} = g_1^\alpha g_1^{a\tilde{t}} \cdot g_2^\alpha g_2^{a\tilde{t}}$ . (Let  $t = \alpha\tilde{t}$ )
- $L = (g_1^\alpha g_2^\alpha)^{\tilde{t}} = g_1^t g_2^t, K_i = (g_1^\alpha g_2^\alpha)^{\tilde{t}\gamma \cdot F(i)} = H(i)^t H'(i)^t$ .

Thus, the distribution of  $d_{as}$  is quite the same as semi-functional secret keys in **Game<sub>2</sub>** or **Game<sub>3</sub>**.

Next we examine the response of challenge oracle. Components of the simulated  $c_f^*$  are distributed as:

- $\kappa = K(e(g_1^s g_2^s, g_1^\alpha g_2^\alpha)) = K(e(g_1, g_1)^{\alpha s} e(g_2, g_2)^{\alpha s}), P = \mathbf{E}_\kappa(M_b)$
- $C = g_1^s g_2^s, C_j = (g_1^s g_2^s)^{a_{A_j \cdot w}} (g_1^s g_2^s)^{-\tilde{r}_j \gamma \cdot F(\rho(j))} = (g_1)^{a_{A_j \cdot sw}} (g_2)^{a_{A_j \cdot sw}} (g_1^{\gamma \cdot F(\rho(j))})^{-\tilde{r}_j s} (g_2^{\gamma \cdot F(\rho(j))})^{-\tilde{r}_j s} = (g_1)^{a_{A_j \cdot v}} (g_2)^{a_{A_j \cdot v}} (g_1^{\gamma \cdot F(\rho(j))})^{-r_j} (g_2^{\gamma \cdot F(\rho(j))})^{-r_j}$ . (Let  $v = sw$  and  $r_j = \tilde{r}_j s$ , uniform in  $\mathbb{Z}_N$ .)
- $D_j = (g_1^s g_2^s)^{\tilde{r}_j} = g_1^{r_j} g_2^{r_j}$ .
- $E_j = (g_1^{r_j} g_2^{r_j})^{\eta_1}$ . Its  $G_{p_1}$  component is  $(g_1^{r_j})^{\eta_1} = (g_1^{\eta_1})^{r_j} = (g_1^{\eta_1} A^{1+G(c_{f^*} \setminus (E_j)) \eta_2})^{r_j} = (w_1 w_2^{G(c_{f^*} \setminus (E_j))})^{r_j} = I(G(c_{f^*} \setminus (E_j)))^{r_j}$ . (Here note that  $G(c_{f^*} \setminus (E_j))$  was programmed to be  $-\eta_2^{-1}$ .) For the  $G_{p_2}$  component, by definition of the function  $I'(\cdot)$ , we have  $(g_2^{r_j})^{\eta_1} = I'(G(c_{f^*} \setminus (E_j)))^{r_j}$ . So,  $E_j = I(G(c_{f^*} \setminus (E_j)))^{r_j} I'(G(c_{f^*} \setminus (E_j)))^{r_j}$ .

Thus, we know that the distribution of  $c_f^*$  is identical to the more-semi-functional ciphertext in **Game<sub>2</sub>** (or **Game<sub>3</sub>**).

For loosen oracle queries  $B$ 's simulation is trivially perfect since  $B$  uses the right trapdoor  $\gamma$ , before reaching the  $i^*$ -th loosen query. Summing up, we know that  $B$ 's simulation is perfect until the adversary  $A$  issues the  $i^*$ -th loosen query, in which the submitted ciphertext  $c_f$  by  $A$  is supposed to involve some non-trivial  $G_{p_2}$  element in the  $D_{j^*}$  component.

**The advantage of  $B$ .** At the  $i^*$ -the loosen query, by the DH check, we can suppose  $(g, I(f, P, C, (C_j, D_j)), D_j, E_j)$  constitutes a DH-tuple. Here,  $I(f, P, C, (C_j, D_j)) = w_1 w_2^{G(c_f \setminus (E_j))} = g_1^{\eta_1} A^{1+\eta_2 G(c_f \setminus (E_j))}$ . Hence,  $(g, g_1^{\eta_1} A^{1+\eta_2 G(c_f \setminus (E_j))}, D_j, E_j)$  is a DH-tuple. Then,  $(g, A, D_j, (D_j^{-\eta_1} E_j)^{1/(1+\eta_2 G(c_f \setminus (E_j)))})$  also constitutes a DH-tuple and the output  $(D_j, (D_j^{-\eta_1} E_j)^{1/(1+\eta_2 G(c_f \setminus (E_j)))})$  by  $B$  means its success in the **Exp<sub>1</sub>**.

Here we note that by  $c_f \neq c_{f^*}$ , we have  $c_f \setminus (E_j) \neq c_{f^*} \setminus (E_j^*)$ . (If  $c_f \setminus (E_j) = c_{f^*} \setminus (E_j^*)$  and  $c_f \neq c_{f^*}$ , then  $E_j/E_j^*$  becomes a non-trivial  $G_{p_2}$  element and breaks the subgroup assumption, even without making a detour via **Exp<sub>1</sub>**.) Then  $G(c_f \setminus (E_j)) \neq G(c_{f^*} \setminus (E_j^*)) = -\eta_2^{-1}$  (except a negligible probability) and the division-by-zero error does not occur in the above.

After all,  $\mathbf{Adv}_B^{\text{exp1}} \geq \frac{1}{q_{\text{loosen}B}} \Pr[\text{Outside}] - O(1/p_2) \geq \frac{1}{q_{\text{loosen}B}} |\Pr[\mathbf{Game}_2^A = 1] - \Pr[\mathbf{Game}_3^A = 1]| - O(1/p_2)$ . By Lemma 1 there exists a PPT algorithm  $B'$  for the subgroup assumption such that  $\mathbf{Adv}_{B'}^{\text{subgr}} \geq \mathbf{Adv}_B^{\text{exp1}} - O(1/p_2)$ . Hence,  $\mathbf{Adv}_{B'}^{\text{subgr}} \geq \frac{1}{q_{\text{loosen}B}} |\Pr[\mathbf{Game}_2^A = 1] - \Pr[\mathbf{Game}_3^A = 1]| - O(1/p_2) \square$

- **Game<sub>4</sub>:**

This game differs from **Game<sub>3</sub>** only in the simulation of **LR**-oracle. The **LR**-oracle procedure **LR<sub>3</sub>** in **Game<sub>3</sub>** is changed to the following new **LR<sub>4</sub>**.

– procedure **LR<sub>4</sub>** ( $f^*$ ):



- \* Only difference from **LR<sub>3</sub>** is in the method of computing  $\kappa$ :  
Choose a fresh  $x \xleftarrow{\$} \mathbb{Z}_{\mathbb{N}}$  and compute  $\kappa = K(e(g, g)^{\alpha s} e(g_2, g_2)^x)$ .

**Claim 4** *Under the decisional parallel BDHE assumption on  $\mathbb{G}$ , the outputs of **Game<sub>3</sub>** and **Game<sub>4</sub>** are indistinguishable for all selective adversaries. More precisely, for any selective adversary  $A$  that issues queries around decryption policies with matrix of size at most  $q$ , there exists a distinguisher  $B$  for the decisional  $q$ -parallel BDHE assumption on  $\mathbb{G}$  that satisfies  $\text{Adv}_B^{\text{q-dpBDHE}} = |\Pr[\mathbf{Game}_2^A = 1] - \Pr[\mathbf{Game}_3^A = 1]|$ .*

**Proof** Given any selective adversary  $A$  that issues queries around decryption policies  $(M, \rho)$  with size  $l \times n$  both of which are at most  $q$ , we construct a following distinguisher  $B$  for the decisional  $q$ -parallel BDHE assumption on  $\mathbb{G} = (N = p_1 p_2, G = G_{p_1} G_{p_2}, G_T = G_{T, p_1} G_{T, p_2}, e_2)$ . Let

$$y = \begin{aligned} & (p_1, g_1, p_2, g_2, g_2^s, g_2^a, \dots, g_2^{a^q}, g_2^{a^{q+2}}, \dots, g_2^{a^{2q}}, \\ & (g_2^{sb_j}, g_2^{a/b_j}, \dots, g_2^{a^q/b_j}, g_2^{a^{q+2}/b_j}, \dots, g_2^{a^{2q}/b_j})_{j \in [q]}, \\ & (g_2^{asb_k/b_j}, \dots, g_2^{a^q sb_k/b_j})_{j, k \in [q], j \neq k}, \end{aligned}$$

be an instance of  $q$ -parallel BDHE problem on  $\mathbb{G}$  (Section 2.1.2).

- **B**  $(y, T)$ : //  $T$  is either  $e(g_2, g_2)^{a^{q+1}s}$  or  $e(g_2, g_2)^x$ .
  - **(Initialize)** Invoke a copy of adversary  $A$  and do as follows:
    - \* Receive a target policy  $f^*(= \Delta f_0)$  and its loosening 's  $\Delta f_1, \dots, \Delta f_\tau$  from adversary  $A$  and then let  $(M^*, \rho^*) \leftarrow (M_0^* | M_1^* | \dots | M_\tau^*, \rho^*)$ , where  $(M_i^*, \rho^*) \leftarrow \text{LSS}(\Delta f_i)$  ( $i = 0, \dots, \tau$ ) and  $M_i^* | M_j^*$  denotes the matrix concatenating rows of two matrices  $M_i^*$  and  $M_j^*$ . (This corresponds to taking **or**-operation **or** $(\Delta f_i, \Delta f_j)$  of the two formulas  $\Delta f_i$  and  $\Delta f_j$ . See for example [17] for details of the conversion.) Let the size of  $M^*$  be  $l \times n$ . Set the initial challenge policy  $f^* = \mathbf{false}$ .
    - \* Choose  $\alpha' \xleftarrow{\$} \mathbb{Z}_{p_2}$  and compute  $e(g_2, g_2)^{\alpha_2} = e(g_2^a, g_2^{a^q}) e(g_2, g_2)^{\alpha'}$ . (We set  $\alpha_2 = \alpha' + a^{q+1}$  that is uniform in  $\mathbb{Z}_{p_2}$ .)
    - \* For all attributes  $x = x_1, \dots, x_B$  that appear in  $(M^*, \rho^*)$ , choose  $z_x \xleftarrow{\$} \mathbb{Z}_{p_2}$  and compute  $H'_x = g_2^{z_x} \prod_{\rho^*(i)=x} \prod_{j \in [n]} g_2^{a^j M_{i,j}^* / b_i}$ .
    - \* Choose  $v_1, \dots, v_B \in G_{p_2}$  randomly so that  $H'(x) = v_1^{F_1(x)} \dots v_B^{F_B(x)} = H'_x$  for  $x = x_1, \dots, x_B$ . That is, set a  $B \times B$  binary invertible matrix  $\Omega = (\Omega_{ij} = F_j(x_i))$  and set  $(v_1, \dots, v_B)^T = \Omega^{-1} \cdot (H'_{x_1}, \dots, H'_{x_B})^T$ . (Here, for matrix  $M = (m_{ij})$  and vector  $(g^{c_1}, \dots, g^{c_B})^T$ , we set  $M \cdot (g^{c_1}, \dots, g^{c_B})^T \stackrel{\text{def}}{=} (g^{\sum_i m_{1i} c_i}, \dots, g^{\sum_i m_{Bi} c_i})^T$ .
    - \* For  $G_{p_1}$  parameters, do honest simulation. That is, choose  $\alpha_1, a_1, \gamma_1, \dots, \gamma_B \xleftarrow{\$} \mathbb{Z}_{p_1}$  and compute  $u_1 = g_1^{\alpha_1}, \dots, u_B = g_B^{\gamma_B}$ .

- \* Choose  $\delta_1, \delta_2 \xleftarrow{\$} \mathbb{Z}_N$  and compute  $w_1 = g_1^{\delta_1}, w_2 = g_1^{\delta_2}$ . Choose  $\beta \xleftarrow{\$} \{0, 1\}$ .
  - \* Return  $pp = (N, g_1, g_1^{\alpha_1}, e(g_1, g_1)^{\alpha_1}, u_1, \dots, u_B, w_1, w_2)$  to  $A$ .
- **(Challenge or Loosen for  $c_{f^*}$ )** When receiving the challenge query  $(\Delta f_0 = f^*, M_0, M_1)$  or the  $i$ -th loosen query  $\Delta f_i$  against the current challenge ciphertext  $c_{f^*} = (f^*, P, C, (C_j, D_j)_j, (E_j)_j)$  ( $i \in [i..\tau]$ ), do as follows:
- \* Assert that the  $i$ -th query  $\Delta f_i$  coincides to the one received in advance before giving  $pp$  to  $A$ .
  - \* Update  $f^* = \mathbf{OR}(f^*, \Delta f_i)$ . Assert( $\#\text{Leaf}(f) \leq B$ ).
  - \* In the case of loosen query, Assert( $\text{DH}(g, I(f^*, P, C, (C_j, D_j)_j), D_j, E_j)$ ) ( $j = 1, \dots, l$ ).
  - \* Set  $M = M_0^* | M_1^* | \dots | M_i^*$  of size  $l \times n$ .
  - \* Choose  $v = (s_1, y_2, \dots, y_n) \xleftarrow{\$} \mathbb{Z}_{p_1}^n$  and  $(y'_2, \dots, y'_n) \xleftarrow{\$} \mathbb{Z}_{p_2}^{n-1}$ , and do the following:
    - Compute  $\kappa = K(e(g_1, g_1)^{\alpha_1 s_1} Te(g_2^s, g_2^{s'}))$  and  $P = \mathbf{E}_\kappa(M_\beta)$ .
    - Set  $C = g_1^{s_1} g_2^s$ . Choose  $r_{11}, \dots, r_{1l} \xleftarrow{\$} \mathbb{Z}_{p_1}, r'_1, \dots, r'_l \xleftarrow{\$} \mathbb{Z}_{p_2}$ . For  $j = 1$  to  $l$ , set  $C_j = C_j^{(1)} C_j^{(2)}$  where  $C_j^{(1)} = g_1^{s_1 M_{j,1} + y_2 M_{j,2} + \dots + y_n M_{j,n}} H(x)^{-r_{1j}}$  and  $C_j^{(2)} = (g_2^a)^{y'_2 M_{j,2} + \dots + y'_n M_{j,n}} H'(x)^{-r'_j} (g_2^{sb_j})^{-z_x} \prod_{\rho(k)=x, k \neq j} \prod_{i \in [n]} (g^{sa^i b_j / b_k})^{-M_{k,i}}$  with  $x = \rho(j)$ .
    - Compute  $D_j = g_1^{r_{1j}} g_2^{r'_j} g_2^{sb_j}$  ( $j \in [l]$ ) and  $E_j = D_j^{\delta_1 + G(f^*, P, C, (C_j, D_j)) \delta_2}$  ( $j \in [l]$ )
  - \* Return  $c_{f^*} = (f^*, P, C, (C_j, D_j)_j, (E_j)_j)$ .
- **(KeyGen)** When receiving a Keygen query for attribute set  $as$ , do as follows:
- \* Asserting that the received  $as$  does not satisfy the current challenge policy  $f^*$ , find  $w = (w_1, \dots, w_n)$  such that  $w_1 = -1, w \cdot M_i = 0$  ( $i \in \rho^{-1}(as)$ ).
  - \* Choose  $t_1 \xleftarrow{\$} \mathbb{Z}_{p_1}$  and  $r \xleftarrow{\$} \mathbb{Z}_{p_2}$ .
  - \* Compute  $K = g_1^{\alpha_1} g^{a_1 t_1} \cdot g_2^{a'} g_2^{ar} \prod_{i \in [2..n]} (g_2^{a^{q+2-i}})^{w_i}$ .
  - \* Compute  $L^{(1)} = g_1^{t_1}$  and  $L^{(2)} = g_2^r \prod_{i \in [n]} (g_2^{a^{q+1-i}})^{w_i}$ .
  - \* For all attributes  $x = x_1, \dots, x_B$  that appear in the challenge policy  $(M^*, \rho^*)$ ,
    - Compute  $\xi_x = (L^{(2)})^{z_x} \prod_{\rho(i)=x} \prod_{j \in [n]} ((g_2^{a^j / b_i})^r \prod_{k \in [n], k \neq j} (g_2^{a^{j+q+1-k} / b_i})^{w_k})^{M_{i,j}}$
  - \* For each  $y \in as$ , do the following:
    - Set  $K_y^{(1)} = H(y)^{t_1}$  and  $K_y^{(2)} = (\Omega^{-1} \cdot (\xi_x)_{x \in \{x_1, \dots, x_B\}})^{F(y)}$ .
  - \* Return  $K, L = L^{(1)} L^{(2)}, (K_y = K_y^{(1)} K_y^{(2)})_{y \in as}$ .
- **(Loosen for  $c_f \neq c_{f^*}$ )** When receiving a loosen query  $(c_f, \Delta f)$  with  $c_f \neq c_{f^*}$ , simulate the loosen-oracle behavior using the trapdoor  $\gamma = (\gamma_1, \dots, \gamma_B)$  generated in the initialization step, following the instructions of honest loosen procedure.

- **(Output)** Given the output  $\beta'$  of  $A$ , output 1 if  $\beta = \beta'$  or 0 otherwise.

**Correctness of the simulation.** It is direct to see the parameter  $pp$  given to  $A$  is perfectly simulated. We check the distributions of responses of the simulated oracles.

First we examine the response of the challenge oracle. When  $T = e(g_2, g_2)^x$  is a random element of  $G_{p_2}$ , the simulated  $\kappa = K(e(g_1, g_1)^{\alpha_1 s_1} e(g_2, g_2)^{x+s\alpha'})$  has the same distribution as in **Game<sub>3</sub>**. When  $T = e(g_2, g_2)^{a^{q+1}s}$ , the  $\kappa$  is distributed as:

- $\kappa = K(e(g_1, g_1)^{\alpha_1 s_1} e(g_2, g_2)^{a^{q+1}s} e(g_2^s, g_2^{\alpha'})) = K(e(g_1, g_1)^{\alpha_1 s_1} e(g_2, g_2)^{\alpha_2 s})$ . (We set  $\alpha_2 = \alpha' + a^{q+1}$  that is uniform in  $\mathbb{Z}_{p_2}$ .)

This is identical to  $\kappa$  in **Game<sub>2</sub>**. The simulated ciphertext  $c_{f^*}$  is distributed as:

- $C = g_1^{s_1} g_2^s, C_j^{(1)} = g_1^{s_1 M_{j,1} + y_2 M_{j,2} + \dots + y_n M_{j,n}} H(x)^{-r_{1j}}$
- $C_j^{(2)} = (g_2^a)^{y_2' M_{j,2} + \dots + y_n' M_{j,n}} H'(x)^{-r_j'} (g_2^{sb_j})^{-z_x} \prod_{\rho(k)=x, k \neq j} \prod_{i \in [n]} (g^{sa^i b_j / b_k})^{-M_{k,i}}$   
 $= (g_2^a)^{y_2' M_{j,2} + \dots + y_n' M_{j,n}} \cdot \prod_{i \in [n]} (g_2^{sa^i})^{M_{j,i}} \cdot H'(x)^{-r_j'} (g_2^{sb_j})^{-z_x} \cdot \prod_{i \in [n]} (g_2^{sa^i})^{-M_{j,i}}$   
 $\prod_{\rho(k)=x, k \neq j} \prod_{i \in [n]} (g^{sa^i b_j / b_k})^{-M_{k,i}}$   
 $= (g_2^a)^{y_2' M_{j,2} + \dots + y_n' M_{j,n}} \cdot \prod_{i \in [n]} (g_2^{sa^i})^{M_{j,i}} \cdot H'(x)^{-r_j'} \{g_2^{z_x} \cdot \prod_{i \in [n]} (g_2^{a^i / b_j})^{M_{j,i}} \cdot$   
 $\prod_{\rho(k)=x, k \neq j} \prod_{i \in [n]} (g^{a^i / b_k})^{M_{k,i}}\}^{-sb_j} = (g_2^a)^{y_2' M_{j,2} + \dots + y_n' M_{j,n}} \cdot \prod_{i \in [n]} (g_2^{sa^i})^{M_{j,i}} \cdot$   
 $H'(x)^{-r_j'} \{g_2^{z_x} \cdot \prod_{i \in [n]} \prod_{\rho(k)=x} (g^{a^i M_{k,i} / b_k})\}^{-sb_j}$   
 $= (g_2^a)^{s M_{j,1} + (as + y_2') M_{j,2} + \dots + (a^{n-1}s + y_n') M_{j,n}} H'(x)^{-r_j' - sb_j}$   
 $= (g_2^a)^{s M_{j,1} + y_2 M_{j,2} + \dots + y_n M_{j,n}} H'(x)^{-r_j}$  (We set  $y_2 = as + y_2', \dots, y_n = a^{n-1}s + y_n'$   
 and  $r_j = r_j' + sb_j$ , that are all independently uniform in  $\mathbb{Z}_{p_2}$ .)
- $D_j = g_1^{r_{1j}} g_2^{r_j'} g_2^{sb_j} = g_1^{r_{1j}} g_2^{r_j}$
- $E_j = D_j^{\delta_1 + G(f, P, C, (C_j, D_j)) \delta_2} = I(f, P, C, (C_j, D_j))^{r_{1j}} I'(f, P, C, (C_j, D_j))^{r_j}$

Thus,  $c_{f^*}$  is distributed in the same way as **Game<sub>2</sub>** or **Game<sub>3</sub>**.

Next we examine the response of the Keygen oracle. The simulated secret key  $d_{as}$  is distributed as:

- $K = g_1^{\alpha_1} g^{a_1 t_1} \cdot g_2^{\alpha'} g_2^{ar} \prod_{i \in [2..n]} (g_2^{a^{q+2-i}})^{w_i}$   
 $= g_1^{\alpha_1} g^{a_1 t_1} \cdot g_2^{\alpha_2} g_2^{-a^{q+1}} \cdot g_2^{ar} \prod_{i \in [2..n]} (g_2^{a^{q+2-i}})^{w_i} = g_1^{\alpha_1} g^{a_1 t_1} \cdot g_2^{\alpha_2} g_2^{ar} \prod_{i \in [1..n]} (g_2^{a^{q+2-i}})^{w_i}$   
 $= g_1^{\alpha_1} g^{a_1 t_1} \cdot g_2^{\alpha_2} g_2^{at}$ . (We set  $t = r + \sum_{i \in [n]} a^{q+1-i} w_i$ , uniform in  $\mathbb{Z}_{p_2}$ .)
- $L^{(1)} = g_1^{t_1}, L^{(2)} = g_2^r \prod_{i \in [n]} (g_2^{a^{q+1-i}})^{w_i} = g_2^t$ .
- $\xi_x = (L^{(2)})^{z_x} \prod_{\rho(i)=x} \prod_{j \in [n]} ((g_2^{a^j / b_i})^r \prod_{k \in [n], k \neq j} (g_2^{a^{j+q+1-k} / b_i})^{w_k})^{M_{i,j}}$   
 $= (L^{(2)})^{z_x} \prod_{\rho(i)=x} \prod_{j \in [n]} ((g_2^{a^j / b_i})^r \prod_{k \in [n]} (g_2^{a^{j+q+1-k} / b_i})^{w_k})^{M_{i,j}}$   
 (Note that  $\prod_{j \in [n]} g_2^{a^{q+1} w_j M_{i,j} / b_i} = g_2^{a^{q+1} \sum_{j \in [n]} w_j M_{i,j} / b_i} = 1$ )  
 $= (L^{(2)})^{z_x} (\prod_{\rho(i)=x} \prod_{j \in [n]} g_2^{a^j M_{i,j} / b_i})^{r + \sum_{k \in [n]} w_k a^{q+1-k}} =$   
 $g_2^{t z_x} (\prod_{\rho(i)=x} \prod_{j \in [n]} g_2^{a^j M_{i,j} / b_i})^t$   
 $= (H'_x)^t$ .

- $K_y^{(1)} = H(y)^{t_1}$ ,  $K_y^{(2)} = (\Omega^{-1} \cdot (\xi_x)_{x \in \{x_1, \dots, x_B\}})^{F(y)} = (\Omega^{-1} \cdot ((H'_x)^t)_{x \in \{x_1, \dots, x_B\}})^{F(y)} = H(y)^t$ .

Finally we examine the response of the loosen oracle for query  $(\Delta f, c_f (\neq c_{f^*}))$ . By modification of the rule in **Game<sub>3</sub>**, we can suppose  $D_j$  components of the submitted ciphertext  $c_f$  is in  $G_{p_1}$ . So the faked trapdoor  $\gamma$ , that is distributed in the right way only in  $\mathbb{Z}_{p_1}$  components, suffices to loosen the  $c_f$  (with respect to  $\Delta f$ ).

**The advantage.** By the above observation we see that

$$\begin{aligned} \text{Adv}_B^{\text{q-dpBDH}} &= |\Pr[B(y, T) = 1 \mid T = e(g_2, g_2)^{a^{q+1}s}] - \Pr[B(y, T) = 1 \mid T = e(g_2, g_2)^x]| \\ &= |\Pr[b' = b \mid \mathbf{Game}_3] - \Pr[b' = b \mid \mathbf{Game}_4]| \\ &= |\Pr[\mathbf{Game}_3^A = 1] - \Pr[\mathbf{Game}_4^A = 1]|. \end{aligned}$$

□

In the final game **Game<sub>4</sub>**, since the challenge ciphertext  $c_f^*$  is informationally independent from  $\kappa$  that encrypts the payload  $P = E_\kappa(M_b)$ , it is easy to see that

$$|\Pr[\mathbf{Game}_4^A = 1] - 1/2| \leq \text{Adv}_{\text{SKE}, B_5}^{\text{cpa1}} \quad (1)$$

for some one-time CPA adversary  $B_5$  against SKE.

Putting Claim 1, Claim 2, Claim 3, Claim 4 and Equation (1) together, we finish the proof of Theorem 1 □

**Theorem 2** *Under the decisional BDH assumption on  $\mathbf{G}$  and the assumption that the symmetric encryption  $\text{SKE} = (\text{E}, \text{D})$  is one-time CPA-secure, the proposed fABE scheme is IND-LSK. More precisely, for any PPT adversary  $A$  against the scheme in the IND-LSK game in the standard model, there exists an algorithm  $B_1$  for the decisional BDH assumption on  $\mathbf{G}$  and a one-time CPA adversary  $B_2$  against SKE that satisfies*

$$\text{Adv}_{A, \text{fABE}}^{\text{ind-lsk}} \leq \text{Adv}_{B_1}^{\text{dB DH}} + \text{Adv}_{\text{SKE}, B_2}^{\text{cpa1}}.$$

**Proof** For ease of presentation, here we assume the used symmetric encryption SKE is perfectly one-time CPA-secure. Given any PPT adversary  $A$  in the IND-LSK game, we construct the following algorithm  $B$  for the decisional BDH assumption on  $\mathbf{G}$ :

- **B**  $(g, g^s, g^\alpha, g^\beta, T)$  //  $T$  is either  $e(g, g)^{s\alpha\beta}$  or random in  $G_{T, p_1}$ .
  - (Initialize) Choose  $a, \gamma_1, \dots, \gamma_B, \delta_1, \delta_2 \xleftarrow{\$} \mathbb{Z}_N$ , and set  $pp = (N, g, g^\alpha, e(g^\alpha, g^\beta), u_1 = g^{\gamma_1}, \dots, u_B = g^{\gamma_B}, w_1 = g^{\delta_1}, w_2 = g^{\delta_2})$ , and invoke a copy of  $A$  on input  $pp$ .
  - (Oracle simulation) For **LR**-oracle simulation, do as follows.
    - \* Given challenge query  $(f^*, M_0, M_1)$ , choose  $b \xleftarrow{\$} \{0, 1\}$  and let  $(\rho, A) \leftarrow \text{LSS}(f^*)$ . ( $A : l \times n$ )

- \* Choose  $w \xleftarrow{\$} \mathbb{Z}_N^n$  with  $w_1 = 1$  and  $\tilde{r}_j \xleftarrow{\$} \mathbb{Z}_N$  for  $j = 1$  to  $l$ .
  - \* Compute  $\kappa = K(T)$  and  $P = \mathbf{E}_\kappa(M_b)$ .
  - \* Set  $C = g^s$  and compute  $C_j = (g^s)^{a_{A_j} \cdot w} (g^s)^{-\tilde{r}_j \gamma \cdot F(\rho(j))}$ ,  $D_j = (g^s)^{\tilde{r}_j}$  for  $j = 1$  to  $l$  and  $E_j = D_j^{\delta_1 + G(f^*, P, C, (C_j, D_j)) \delta_2}$  for  $j = 1$  to  $l$ .
  - \* Return  $c_f^* = (f^*, P, C, (C_j, D_j)_{j=1, \dots, l}, (E_j)_{j=1, \dots, l})$  to  $A$ .
- (Output) Given the output  $b'$  of  $A$ , output 1 if  $b = b'$  or 0 otherwise.

It is direct to see that the simulation by  $B$  is perfect for the invoked adversary  $A$  when  $T = e(g, g)^{s\alpha\beta}$ , and that the simulated view of  $A$  is independent from the choice of  $b$  when  $T$  is random in  $G_{T, p_1}$ . So, we have

$$\begin{aligned}
 \mathbf{Adv}_B^{\text{dB DH}} &= |\Pr[B(g, g^s, g^\alpha, g^\beta, T) = 1 \mid T = e(g, g)^{s\alpha\beta}] - \Pr[B(g, g^s, g^\alpha, g^\beta, T) = 1 \mid T \xleftarrow{\$} G_{T, p_1}]| \\
 &= |\Pr[b' = b \text{ in Game}_{A, \text{fABE}}^{\text{ind-lsk}}] - 1/2| \\
 &= \mathbf{Adv}_{A, \text{fABE}}^{\text{ind-lsk}} \quad \square
 \end{aligned}$$

## 5 Conclusion

We gave new security definition for flexible attribute-based encryption. Our new definition of IND-LSO is stronger and more natural in the sense that it allows adversaries to issue the challenge ciphertext to the loosening oracles. Second, we constructed a concrete construction of flexible attribute-based encryption using composite-order pairing map. We proved its security (in our new definition) in the random oracle model against selective adversaries, employing the dual encryption method.

### 参考文献

- [1] Seiko Arita, “Flexible Attribute-Based Encryption”, ICICS 2012, LNCS 7618, pp. 471-478, 2012.
- [2] Randy Badinand, Adam Benderand, Neil Springand, Bobby Bhattacharjee, and Daniel Starin. “Persona: An online social network with user defined privacy”, In ACM SIGCOMM, 2009.
- [3] Rakesh Bobba, Omid Fatemieh, Fariba Khan, Arindam Khanand Carl A. Gunter, Himanshu Khurana, and Manoj Prabhakaran, “Attribute-based messaging: Access control and confidentiality”, ACM Transactions on Information and System Security (TISSEC), Volume 13 (4).
- [4] M. Bellare and P. Rogaway, “The Security of Triple Encryption and a Framework for Code-Based Game-Playing Proofs”, EUROCRYPT 2006, LNCS 4004, pp. 409426, Springer-Verlag, 2006.
- [5] John Bethencourt, Amit Sahai, Brent Waters, “Ciphertext-Policy Attribute-Based Encryption”, SP '07, pp. 321–334, IEEE Computer Society, 2007.
- [6] M. Green, S. Hohenberger, B. Waters, “Outsourcing the Decryption of ABE Ciphertexts”. In Usenix Security 2011.
- [7] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters, “Attribute-based encryption for fine-grained access control of encrypted data”, In ACM Conference on Computer and Communications Security, pages 89–98, 2006.

- [8] S. Guo, Y. Zeng, J. Wei, and Q. Xu, “Attribute-based re-encryption scheme in the standard model”, *Wuhan University Journal of Natural Sciences*, 13(5):621–625, 2008.
- [9] Jyun-Yao Huang, Chen-Kang Chiang, and I-En Liao, “An Efficient Attribute-Based Encryption and Access Control Scheme for Cloud Storage Environment”, *GPC 2013, LNCS 7861*, pp. 453–463, 2013.
- [10] Luan Ibraimi, Muhammad Asim, Milan Petkovic, “An Encryption Scheme for a Secure Policy Updating”, *SECURITY 2010*, pp. 399–408, 2010.
- [11] Seny Kamara and Kristin Lauter, “Cryptographic Cloud Storage”, *FC ’10, LNCS 6054*, pp. 136–149, 2010.
- [12] Hyoseung Kim, Seunghwan Park, Jong Hwan Park, Dong Hoon Lee, “Attribute-Based Encryption for Commercial Content Distribution”, *IT Convergence and Security 2012, Lecture Notes in Electrical Engineering Volume 215*, 2013, pp 205–213
- [13] Yutaka Kawai, and Katsuyuki Takashima, “Functional Proxy-Re-Encryption” *SCIS 2013, The 30th Symposium on Cryptography and Information Security*, 1A1-(3).
- [14] X. Liang, Z. Cao, H. Lin, J. Shao, “Attribute Based Proxy Re-encryption with Delegating Capabilities”, *ASIACCS ’09*, pp. 276–286, 2009
- [15] Song Luo, Jianbin Hu, and Zhong Chen, “Ciphertext Policy Attribute-Based Proxy Re-encryption”, *ICICS 2010, LNCS 6476*, pp. 401–415, 2010.
- [16] Allison Lewko, Tatsuaki Okamoto, Amit Sahai, Katsuyuki Takashima, and Brent Waters, “Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption”, *EUROCRYPT ’10*, pp. 62–91, 2010.
- [17] Allison Lewko and Brent Waters, “Decentralizing Attribute-Based Encryption”, *Eurocrypt 2011, LNCS 6632*, pp. 568–588, 2011.
- [18] Allison Lewko and Brent Waters, “New Proof Methods for Attribute-Based Encryption: Achieving Full Security through Selective Techniques”, *CRYPTO 2012, LNCS 7417*, pp. 180–198, 2012.
- [19] Takeo Mizuno and Hiroshi Doi, “Hybrid Proxy Re-encryption Scheme for Attribute-Based Encryption”, *Inscrypt 2009, LNCS 6151*, pp. 288–302, 2010.
- [20] Tatsuaki Okamoto and Katsuyuki Takashima, “Fully Secure Functional Encryption with General Relations from the Decisional Linear Assumption”, *CRYPTO 2010, LNCS 6223*, pp. 191–208, 2010.
- [21] Matthew Pirretti, Patrick Traynor, Patrick McDaniel, and Brent Waters, “Secure attribute-based systems”, In *ACM Conference on Computer and Communications Security*, pp. 99–112, 2006.
- [22] Amit Sahai and Brent Waters, “Fuzzy identity-based encryption”, *EUROCRYPT ’05*, pp. 457–473, 2005.
- [23] Patrick Traynor, Kevin R. B. Butler, William Enck, and Patrick McDaniel, “Realizing massive-scale conditional access systems through attribute-based cryptosystems”, In *NDSS*, 2008.
- [24] Yar-Ling Tan, Bok-Min Goi, Ryoichi Komiya, Syh-Yuan Tan, “A Study of Attribute-Based Encryption for Body Sensor Networks”, *Informatics Engineering and Information Science, Communications in Computer and Information Science Volume 251*, 2011, pp 238–247.
- [25] Brent Waters, “Ciphertext-Policy Attribute-Based Encryption: An Expressive, Efficient, and Provably Secure Realization”, *PKC 2011, LNCS 6571*, pp. 53–70, 2011.