4-2018

Criteria-based encryption

Tran Viet Xuan PHUONG

Guomin YANG
Singapore Management University, gmyang@smu.edu.sg

Willy SUSILO

Advance Access publication on 25 October 2017
doi:10.1093/comjnl/bxx088

Criteria-Based Encryption

Tran Viet Xuan Phuong*, Guomin Yang and Willy Susilo

Institute of Cybersecurity and Cryptology, School of Computing and Information Technology, Faculty of Engineering and Information Sciences, University of Wollongong, Australia
**Corresponding author: tvxp750@uowmail.edu.au*

We present a new type of public-key encryption called Criteria-based Encryption (or CE, for short). Different from Attribute-based Encryption, in CE, we consider the access policies as criteria carrying different weights. A user must hold some cases (or answers) satisfying the criteria and have sufficient weights in order to successfully decrypt a message. We then propose two CE Schemes under different settings: the first scheme requires a user to have at least one case for a criterion specified by the encryptor in the access structure, while the second scheme requires a user to have all the cases for each criterion. We prove that both schemes are secure under the Decisional q-Bilinear Diffie Hellman Exponent assumption without random oracles. In addition, we also present two special CE schemes for the above two settings without considering the weight requirement. We show that under this special case CE schemes can be constructed much more efficiently.

Keywords: criteria-based encryption; polynomial; root; Viète's formula; weight

Received 22 November 2016; revised 30 June 2017; editorial decision 28 August 2017
Handling editor: Bogdan Warinschi

1. INTRODUCTION

Fine-grained access control is very important to protect sensitive information in current and future information systems. Attribute-based encryption (ABE) [8, 19, 30] has provided an elegant solution for the problem by embedding the access control policy inside the encryption key. The access policy in an ABE is defined over the Universe of user attributes, and a user can decrypt the message if and only if his/her attributes can satisfy the access policy. In this paper, we consider a different way to formulate access policies via 'criteria'. A criterion is a principle or standard by which something may be judged or decided,[1] which makes it suitable for defining an access structure. We illustrate this idea via the following example.

Suppose the NSA is announcing an inaugural grant application to help them combatting cyber-terrorism in the US. However, in order to avoid the leakage of the project information to the general public, the NSA encrypts the details of the grant application based on some default selection criteria: 'the person must be born in the US' **AND** ('the person must be a faculty member in a US-based University' **OR** 'the person must hold a PhD that is awarded by one of the universities in the US'). In addition, the importance of each criterion could be different, e.g. the first criterion may have higher weight/credit (e.g. 3) than the remaining two (e.g. 2 and 1, respectively) in this scenario due to the sensitivity of the application. NSA may require an applicant to possess sufficient credits (e.g. 5) in order to become eligible to apply for the grant.

Now consider the following potential applicants (Table 1). Alice is a Professor in the University of Minnesota; she was born in the Massachusetts State, and acquired her PhD from Stanford University. Bob is an Associate Professor in Royal Holloway, UK; he was born in New York, and acquired his PhD from the New York University. Charlie is an Assistant Professor in MIT; he graduated from MIT, and was born in Canada. In this example, Professor Alice satisfies all the criteria and will be able to decrypt the call for application. On the other hand, neither Bob nor Charlie could decrypt the message. Although Bob satisfies the first and the third criteria, the total weight is below the threshold. Similar to ABE, a secure CE should also ensure that Bob cannot collude with Charlie to decrypt the message.

From the above example, we can see that a CE scheme differs from an ABE scheme in two aspects: first, ABE uses the same set of attributes to define access structures and to derive user secret keys, while in a CE scheme, a criterion and its satisfying cases are two different entities and they can have a one-to-many relationship; second, the existing ABE schemes

[1] Oxford dictionary.

TABLE 1. An example of CE (threshold = 5).

Name	Birth place weight = 3	Work place weight = 2	Alumni weight = 1	Decryption
Alice	Massachusetts	Minnesota University	Stanford University.	✓
Bob	New York	Royal Holloway	NYU	×
Charlie	Canada	MIT	MIT	×

have not considered the weight of an attribute when defining the access policy, and it is also an interesting problem to design an ABE supporting weight.

Our contributions: In this paper, we formalize the notion of Criteria-based Encryption (CE) and propose two concrete CE schemes under different settings. As demonstrated in the example, in a CE scheme, every criterion will have a weight and a set of cases (or answers) satisfying it. In our first CE scheme, we require the decryptor to have at least one case for each criterion specified in the access structure as well as sufficient accumulated weight in order to successfully decrypt the message, while in the second construction, we require the decryptor to possess all the cases for each criterion in addition to the weight requirement. To make our schemes expressive, we apply the Linear Secret Sharing Scheme (LSSS) over the criteria to define a monotone access structure. The main technical challenge is to allow successful decryption if the decryptor has only one case (or all the cases in the second scenario) for one criterion. Our solution for this problem is to represent a criterion as a polynomial and the corresponding cases satisfying the criteria as the roots of the polynomial. In the first scenario, we require that the root hold by the decryptor can satisfy the polynomial, while for the second scenario where the decryptor is required to have all the cases of a criterion, we apply the Viète's formula to ask the decryptor to reconstruct the polynomial if he/she possesses all the roots, and the decryption will be successful if and only if the reconstructed polynomial is identical to the one used in the encryption.

Another challenge in constructing CE schemes is to implement the weight condition. One option is to fix the weight for each criterion in the setup. This will simplify the construction. However, to make the CE scheme more useful, it is desirable to let the encryptor decide the weight of each criterion. In the example we give above, the second or the third criterion may have more weight than the first one if this is a grant application announced by the NSF instead of NSA. In our proposed CE schemes, we allow the encryptor to specify the weight of each criterion and the threshold for the accumulated weight. Since there are different combinations that can meet the threshold, we put all the valid cases in the ciphertext, and the decryptor can decrypt the message if he/she can meet one of these cases. In addition, we also consider the special setting of CE without weight. We show that under this special case CE schemes can be constructed much more efficiently.

1.1. Related work

Embedding policy-based access control into modern encryption schemes is an interesting but challenging task that has been intensively studied by the cryptologic research community in recent years. Attribute-based Encryption (ABE) [7, 8, 19, 22, 25, 30], which is an extension of Identity-based Encryption [9, 10, 28, 27], provided an elegant solution to achieve this task. The idea was proposed by Sahai and Waters [27] when they presented their Fuzzy IBE scheme which can be treated as the first Key Policy (KP) ABE based on a threshold access policy. The notion of ABE was later formalized by Goyal *et al*. [19]. There are two types of ABE schemes: in a KP ABE, the user attributes serve as the encryption key, and the access structure is embedded in the decryption key; in a Ciphertext Policy (CP) ABE, the access structure is used in the encryption process, and each user obtains secret keys based on his/her attributes. The Criteria-based Encryption schemes proposed in this paper are similar to CP-ABE schemes in the sense that we also put the access structure (defined over criteria rather than attributes) in the encryption process.

There are two main steams on the design of ABE schemes, those based on the LSSS (e.g. [8, 18, 19, 21, 22, 30]), and those based on the AND-gates with/without Wildcard [15, 23, 31]. LSSS-based ABE schemes are in general more expressive than other types of ABE schemes. In this paper, we will also focus on LSSS-based access structures.

It is worth noting that functional encryption [6, 11] is an emerging paradigm for public-key encryption that enables fine-grained control of access to encrypted data. We also stick to indistinguishability-based security due to the strong impossibility results from simulation-security for functional encryption [13, 14]. An interesting work of study indistinguishability obfuscation and functional encryption for general circuits [16] opens a new security approach for public-key encryption. Recently, a new notion of multi-input functional encryption has been proposed [2, 17], which is a generalization to the case of n-ary functions.

Criteria-based encryption is a special type of Functional Encryption. The later is a general term for a range of emerging public-key encryption systems including ABE, Predicate Encryption [12, 20, 29], Inner-Product Encryption [1, 3, 5, 4, 26, 24], etc. The functions underlying our criteria-based encryption are root verification for polynomials, and weight evaluation.

1.2. Organization

In Section 2, we present some basics related to polynomials and roots that play an important role in our schemes. Then we give the background and assumptions in Section 3 and present the definitions and security models of criteria-based encryption in Section 4. We present our CE schemes in Section 5, and prove their security in Section 6. Then we make a comparison between the two proposed schemes in Section 7. The paper is concluded in Section 8. We put the special CE without weight scheme and additional security proof in A.

2. POLYNOMIAL AND THE VIÈTE'S FORMULA

Consider a polynomial P_i with degree n:

$$P_i = a_n x^n + a_{n-1}x^{n-1} + \cdots + a_1 x + a_0. \quad (1)$$

We use a vector $\overrightarrow{v}$ to represent the coefficients of P_i

$$\overrightarrow{v} = (a_n, a_{n-1}, \ldots, a_1, a_0).$$

For any integer x, define

$$\overrightarrow{x} = (\underbrace{x \cdot x \cdots x}_{n}, \underbrace{x \cdots x}_{n-1}, \ldots, x, 1).$$

If $(\overrightarrow{v} \cdot \overrightarrow{x}) = 0$, then x is a root of P_i.

In our CE schemes, we represent a criterion by a polynomial P_i, and the cases satisfying the criterion as the roots of the polynomial. Figure 1 is an example of the above idea. The database system is encrypted under an access policy (*Criteria* P_1 AND *Criteria* P_2) OR *Criteria* P_3. Alice has the roots of P_1 and P_2, while Bob has a root of P_3. So both of them can satisfy the access policy. In the above example, we only require one root of a polynomial.

Another possible setting is that we require the user to have all the roots of a polynomial. Figure 2 shows an example under this setting where Alice have all the roots of P_2, which allow her to access the database encrypted under the access policy *Criteria* P_2 OR *Criteria* P_3.

To implement such a CE scheme, we will use the Viète's formula. Consider the polynomial P_i given above. Assume $a_n \neq 0$, we create a new coefficient vector $\overrightarrow{v}$ as

$$\overrightarrow{v} = \left(1, \frac{a_{n-1}}{a_n}, \ldots, \frac{a_0}{a_n}\right).$$

Let $\{x_i\}$ denote the roots of the P_i, then the Viète's formula describes a way to reconstruct $\overrightarrow{v}$ from $\{x_i\}$ as follows:

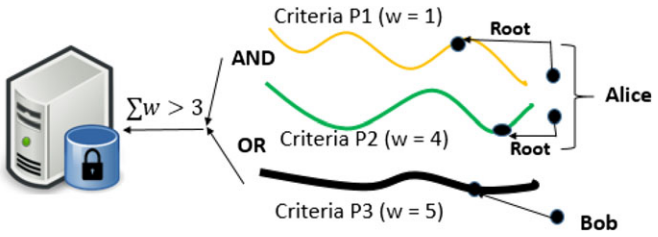

FIGURE 1. The single root example.

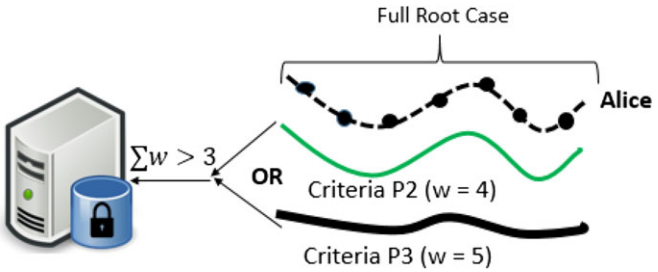

FIGURE 2. The full root set example.

$$\begin{cases} x_1 + x_2 + \cdots + x_n & = \left(-\dfrac{a_{n-1}}{a_n}\right) \\ (x_1x_2 + x_1x_3 + \cdots + x_1x_n) & \\ \quad + (x_2x_3 + x_3x_4 + \cdots + x_2x_n) & \\ \quad + \cdots + x_{n-1}x_n & = \left(\dfrac{a_{n-2}}{a_n}\right) \\ \cdots & \\ x_1x_2\cdots x_n & = (-1)^n \dfrac{a_0}{a_n} \end{cases}.$$

Equivalently, we can write

$$\sum_{1 \le i_1 < i_2 < \cdots < i_k \le n} x_{i_1}x_{i_2}\cdots x_{i_k} = (-1)^k \frac{a_{n-k}}{a_n}, \quad (2)$$

for $k = 1, 2, \ldots, n$.

3. BACKGROUND

3.1. Access structures

Let $\{P_1, P_2, \ldots, P_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^{\{P_1, P_2, \ldots, P_n\}}$ is monotone if $\forall B, C$: if $B \in \mathbb{A}$ and $B \subseteq C$, then $C \in \mathbb{A}$. An access structure is a collection $\mathbb{A}$ of non-empty subsets of $\{P_1, P_2, \ldots, P_n\}$, i.e. $\mathbb{A} \subseteq 2^{\{P_1, P_2, \ldots, P_n\}} \setminus \{\varnothing\}$. The sets in $\mathbb{A}$ are called the authorized sets, and the sets not in $\mathbb{A}$ are called the unauthorized sets.

3.2. Linear secret sharing scheme

A secret sharing scheme II over a set of parties $\mathcal{P}$ is called linear over $\mathbb{Z}_p$ if

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

Linear reconstruction: Suppose that II is an LSSS for the access structure $\mathbb{A}$. Let $S \in \mathbb{A}$ be any authorized set, and let $I \subset \{1, 2,\ldots,l\}$ be defined as $I = \{i: \rho(i) \in S\}$. Then there exist constants $\{\omega_i \in \mathbb{Z}_p\}_{i\in I}$ such that if $\{\lambda_i\}$ are valid shared of any secret s according to II. then $\sum_{i\in I} \omega_i \lambda_i = s$.

3.3. Bilinear map on prime order groups

Let $\mathbb{G}$ and $\mathbb{G}_\mathbb{T}$ be two multiplicative cyclic groups of same prime order p, and g a generator of $\mathbb{G}$. Let e: $\mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a bilinear map with the following properties:

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

Notice that the map e is symmetric since $e(g^a, g^b) = e(g, g)^{ab} = e(g^b, g^a)$.

3.4. Decision q-Bilinear Diffie–Hellman exponent (q-BDHE) assumption

The Decision q-BDHE problem in $\mathbb{G}$ is defined as follows: let $\mathbb{G}$ be a bilinear group of prime order p. Given a vector

$$(g, h, g_1, g_2,\ldots,g_q, g_{q+2},\ldots,g_{2q}, T),$$

where $g_i = g^{a^i} \in \mathbb{G}$ for shorthand. Let $\overrightarrow{y}_{g,a,q} = (g, g_1, g_2,\ldots, g_q, g_{q+2},\ldots,g_{2q})$. We say that the q-BDHE assumption holds in $\mathbb{G}$ if for any probabilistic polynomial-time algorithm A

$$|\Pr[A(g, h, \overrightarrow{y}_{g,a,q}, e(g_{q+1}, h)) = 1] - \Pr[A(g, h, \overrightarrow{y}_{g,a,q}, T) = 1]| \leq \epsilon(n),$$

where T is a random element of $\mathbb{G}_T$, and $\epsilon(n)$ is negligible in the security parameter n.

4. CRITERIA-BASED ENCRYPTION (CE)

In this section, we present the formal definition and security model for Criteria-based Encryption.

4.1. Functional definition

- Setup(1^n, d): The setup algorithm takes two inputs: the security parameter 1^n and degree d of the polynomials that represent the criteria. It outputs the public parameter PK and a master key MSK.
- Encryption(M, PK, Pol, $\overrightarrow{\nu}$, τ): The encryption algorithm takes as an input the public parameters PK, a message M, an access structure Pol over a set of criteria, a vector $\overrightarrow{\nu}$ indicating the weight of each criterion and a threshold value τ. It outputs a ciphertext CT.
- Key Gen(MSK, $\mathcal{C}$): The key generation algorithm takes input the master secret key MSK and the set of cases $\mathcal{C}$ a user possesses. It outputs a user secret key SK.
- Decrypt(CT, SK): The decryption algorithm takes as an input a ciphertext CT and a user secret SK, and outputs a message M or a special symbol $\perp$.

4.2. IND-CPA security of CE

Similar to other public-key encryption schemes, we define the IND-CPA security of a CE scheme via the following game:

- *Setup*: The challenger runs the Setup algorithm and gives the public parameters PK to the adversary.
- *Phase* 1: The adversary adaptively makes private key generation queries for any case set $\mathcal{C}$.
- *Challenge*: The adversary submits two equal length messages M_0 and M_1, a challenge access structure Pol^*, a weight vector $\overrightarrow{\nu^*}$, and a threshold value τ^*. The restriction is that Pol^* cannot be satisfied by any case set $\{\mathcal{C}\}$ appeared in Phase 1. The challenger then flips a random coin β, and encrypts M_β. The resulting ciphertext CT^* is given to the adversary.
- *Phase* 2: Phase 1 is repeated with the restriction that Pol^* cannot be satisfied by any case set appeared in the private key generation queries.
- *Guess*: The adversary outputs a guess β' of β.

We say a CE scheme is IND-CPA secure if for any probabilistic polynomial-time adversary $\mathcal{A}$

$$\mathbf{Adv}_{\mathcal{A}}^{\text{IND-CPA}}(k) = \left|\Pr[\beta' = \beta] - \frac{1}{2}\right|$$

is negligible in the security parameter k.

Selective security: In the selective IND-CPA model, the adversary has to provide Pol^*, $\overrightarrow{\nu^*}$, τ^* at the beginning of the game (i.e. before Setup).

5. CONSTRUCTIONS

In this section, we present two CE scheme based on the two scenarios (i.e. verify root of polynomial and check equal

coefficients). As general type as functional encryption, our CE has the similar generation as log N public-key size and R ciphertext size, where N is the size of Universe and R is the number of recipients. We illustrate our proposed scheme by the example in Table 1 by the criteria policy ('the person must be born in the US' **AND** 'the person must be a faculty member in a US-based University') **OR** ('the person must hold a PhD that is awarded by one of the universities in the US'). Suppose that P_1 = 'the person must be born in the US' weighted three, P_2 = 'the person must be a faculty member in a US-based University' weighted two, P_3 = 'the person must hold a PhDthat is awarded by one of the universities in the US' weighted one. Then the threshold τ is given by three. In this case, we consider the criterion set with accumulated weight larger than three as P_1P_2, P_1P_3, $P_1P_2P_3$.

Then the ciphertext is generated by the policy and the accumulated weight criterion set as

$$Pol = (P_1 \text{ AND } P_2) \text{ OR } P_3;\ T = \{12, 13, 123\}.$$

In the generation of each user's key, Alice is a Professor in the University of Minnesota; she was born in the Massachusetts State, and acquired her PhD from Stanford University. Then Alice's key is

$$C_{\text{Alice}} = (c_1, c_2, c_3);\ S_{\text{Alice}} = \{1, 2, 3, 12, 13, 123\}.$$

Alice can decrypt the message. Since $C_{\text{Alice}} \vDash Pol$, and $S_{C_{Alice} \vDash Pol} = \{1, 2, 3, 12, 13, 123\}$. Then $J = |T \cap S_{C_{Alice} \vDash Pol}| = \{12, 13, 123\}$.

In Bob's case, he is an Associate Professor in Royal Holloway, UK; he was born in New York, and acquired his PhD from the New York University. Then Bob's key is

$$C_{\text{Bob}} = (c_1', c_2', c_3');\ S_{\text{Bob}} = \{1, 2, 3, 12, 13, 123\}.$$

Bob cannot decrypt the messages, since the set of Bob's cases is not satisfied the Policy Pol as $C_{\text{Bob}} \nvDash Pol$.

Another case, Charlie is an Assistant Professor in MIT; he graduated from MIT, and was born in Canada. The key is generated by

$$C_{\text{Charlie}} = (c_1'', c_2'', c_3''); S_{\text{Charlie}} = \{1, 2, 3, 12, 13, 123\}.$$

Even though the case of Charlie is satisfied the selected criteria, the weighted case of criterion is less than give threshold *tau*. $C_{\text{Charlie}} \vDash Pol$, and $S_{C_{Charlie} \vDash Pol} = \{3\}$. Then $J = |T \cap S_{C_{Charlie} \vDash Pol}| = \varnothing$, then he cannot decrypt the message.

5.1. CE-Verify Root of Polynomial

- Setup(1^n, d): The key generation authority first chooses a group $\mathbb{G}$ of prime order p and a generator g. It also defines the criterion Universe U, which is expressed by a set of d-degree polynomials: $\{P_1, P_2, \ldots, P_n\}$. Each criterion is labeled by a random number $tag_i \in_R \mathbb{Z}_p$. Every polynomial has a set of coefficients $a_d, a_{d-1}, \ldots, a_0$, which are used to generate the following vectors:

$$\begin{cases} \overrightarrow{P_1} = \left(a_{1_d}, a_{1_{d-1}}, \ldots, a_{1_0}, \dfrac{1}{tag_1}\right) \\ \overrightarrow{P_1'} = (a_{1_d}, a_{1_{d-1}}, \ldots, a_{1_0}, 1) \\ \overrightarrow{P_2} = \left(a_{2_d}, a_{2_{d-1}}, \ldots, a_{2_0}, \dfrac{1}{tag_2}\right) \\ \overrightarrow{P_2'} = (a_{2_d}, a_{2_{d-1}}, \ldots, a_{2_0}, 1) \\ \ldots \\ \overrightarrow{P_n} = \left(a_{n_d}, a_{n_{d-1}}, \ldots, a_{n_0}, \dfrac{1}{tag_n}\right) \\ \overrightarrow{P_n'} = (a_{n_d}, a_{n_{d-1}}, \ldots, a_{n_0}, 1) \end{cases}$$

Then we choose $h_1, \ldots, h_n \in \mathbb{G}$ randomly, pick $y, a \in_R \mathbb{Z}_p$, and create the public parameters and master key as

$$PK = (g, e(g, g)^y, g^a, g^{tag_1\overrightarrow{P_1}}, \ldots, g^{tag_n\overrightarrow{P_n}}, g^{\overrightarrow{P_1'}}, \ldots, g^{\overrightarrow{P_n'}}, h_1, \ldots, h_n).$$
$$MSK = (g^y, g^{tag_1}, \ldots, g^{tag_n}).$$

- Encryption(M, PK, Pol, $\overrightarrow{\nu} = \{w_m\}_{m \in \{1, \ldots, n\}}$, τ): It takes as input a message $M \in \mathbb{G}_T$, an access structure $Pol = (\mathbb{A}, \rho)$, a vector $\overrightarrow{\nu}$ which indicates the weight of each criterion, and a given threshold τ. The function ρ associates each row of $\mathbb{A}$ to one criterion.
The algorithm then chooses a random vector $\overrightarrow{x} = (s, y_2, \ldots, y_n) \in \mathbb{Z}_p^n$. These values will be used to share the encryption exponent s. For $i = 1$ to l, where l denotes the number of rows in $\mathbb{A}$, it calculates $\lambda_i = \overrightarrow{x} \cdot \mathbb{A}_i$, where $\mathbb{A}_i$ is the vector corresponding to the ith row of $\mathbb{A}$.
Next, the encryptor identifies all the criterion set with accumulated weight larger than τ. Let $T = \{(k_1^i, k_2^i, \ldots, k_{\mu_i}^i)\}$ with μ_i is the maximum number combination of criteria i, which denotes such a set, where $k_j^i \in \{1, 2, \ldots, n\}$ denotes a position in the criterion Universe.
Finally, compute

$$C_1 = M \cdot e(g, g)^{ys},\ C_2 = g^s,$$
$$\{C_i = g^{a\lambda_i} g^{-tag_{\rho(i)}\overrightarrow{P}_{\rho(i)}s},\ C_i' = g^{a\lambda_i} g^{-\overrightarrow{P'}_{\rho(i)}s}\}_{i=1,\ldots,l},$$
$$\left\{\hat{C}_i = \prod_{j=1}^{\mu_i} h_{k_j^i}^s\right\}_{i=1,\ldots,len(T)}.$$

The ciphertext is set as

$$CT = (\tau, Pol, C_1, C_2, \{C_i, C_i'\}_{i=1,\ldots,l}, \{\hat{C}_i\}_{i=1,\ldots,len(T)}, T).$$

- KeyGen(MSK, $\mathcal{C}$): Let $\mathcal{C} \subseteq \{C_x\}_{x\in\{1,\ldots,n\}}$ denote a set of cases held by a user, and tag_x the tag for criterion x chosen in the setup. We use z_{tag_x} to denote a root of the polynomial corresponding to x. For each z_{tag_x} possessed by the user, the key generation authority first computes a vector

$$\overrightarrow{z_{tag_x}} = (z_{tag_x}^d, z_{tag_x}^{d-1}, \ldots, z_{tag_x}, 1).$$

For set $\mathcal{C}$, let $P = \{P_1, \ldots, P_{len(\mathcal{C})}\}$ denote the position of each corresponding criterion in the criterion Universe U. Also, let $S = \{(P_1), (P_2), \ldots, (P_1, P_2), \ldots, (P_1, P_2, \ldots, P_{len(\mathcal{C})})\} = \{(k_1^i, \ldots, k_{\nu_i}^i)\}(1 \le i \le 2^{len(\mathcal{C})})$ denote all the combinations of the elements in P. Next the authority chooses a random $t \in \mathbb{Z}_p$, and computes

$$L = g^t, \forall C_x \in \mathcal{C}\ K_x = (g^{tag_x})^{\overrightarrow{z_{tag_x}t}} \quad \text{and} \quad K_x' = g^{\overrightarrow{P'}_x t},$$
$$\left\{\hat{K}_i = g^y g^{at} \prod_{j=1}^{\nu_i} (h_{k_j^i})^t\right\}_{i=1,\ldots,len(S)}.$$

The user secret key is set as

$$SK = (L, \{K_x, K_x'\}_{C_x\in\{\mathcal{C}\}}, \{\hat{K}_i\}_{i=1,\ldots,len(S)}, S).$$

- Decryption(CT, SK): Given a ciphertext CT for an access structure($\mathbb{A}$, ρ), and a secret key SK for a set of cases $\mathcal{C}$, let $I \subset \{1, 2, \ldots, l\}$ be defined as $I = \{i: \mathcal{C} \text{ contains a case for } \rho(i)\}$. Then, let $\{\omega_i \in \mathbb{Z}_p\}_{i\in I}$ be a set of constants such that if $\{\lambda_i\}$ are valid shares of any secret s according to $\mathbb{A}$, then $\sum_{i\in I} \omega_i \lambda_i = s$.
Then it computes

$$\left(\prod_{i\in I} e(L, C_i') \cdot \frac{e(K_{\rho(i)}, C_i)\, e(K'_{\rho(i)}, C_2)}{e(K_{\rho(i)}, C_i')}\right)^{\omega_i}$$
$$= \left(\prod_{i\in I} e(g^t, g^{a\lambda_i} g^{-\overrightarrow{P'}_{\rho(i)} s}) \cdot \frac{e\left((g^{tag_{\rho(i)}})^{\overrightarrow{z}_{tag_{\rho(i)}} t}, g^{a\lambda_i} g^{-tag_{\rho(i)}\overrightarrow{P}_{\rho(i)} s}\right) e(g^{\overrightarrow{P'}_{\rho(i)} t}, g^s)}{e\left((g^{tag_{\rho(i)}})^{\overrightarrow{z}_{tag_{\rho(i)}} t}, g^{a\lambda_i} g^{-\overrightarrow{P'}_{\rho(i)} s}\right)}\right)^{\omega_i}$$
$$= \left(\prod_{i\in I} e(g^t, g^{a\lambda_i}) e(g^t, g^{-\overrightarrow{P'}_{\rho(i)} s}) \cdot e(g^{\overrightarrow{P'}_{\rho(i)} t}, g^s)\right)^{\omega_i}$$
$$= \prod_{i\in I} e(g, g)^{a\lambda_i t\omega_i} = e(g, g)^{ast}.$$

Let $J = \{T \cap S_{C \vDash Pol}\}$, for each $i \in J$, let i_T denote the position of i in T, i_S denote the position of i in S

$$\left(\prod_{i\in J} \frac{e(C_2, \hat{K_{i_S}})}{e(L, \hat{C_{i_T}})}\right)^{1/|J|}$$
$$= \left(\prod_{i\in J} \frac{e(g^s, g^y g^{at} \prod_{j=1}^{len(i)} (h_{i_j})^t)}{e(g^t, \prod_{j=1}^{len(i)} (h_{i_j})^s)}\right)^{1/|J|}$$
$$= \left(\prod_{i\in J} e(g^s, g^y g^{at})\right)^{1/|J|}$$
$$= ((e(g, g)^{ys} e(g, g)^{ast})^{|J|/|J|} = e(g, g)^{ys} e(g, g)^{ast}.$$

Finally, compute $e(g, g)^{ys} e(g, g)^{ast} / e(g, g)^{ast} = e(g, g)^{ys}$ and recover message M as $C_1 / e(g, g)^{ys}$.

5.2. CE-Equal Coefficients

In this section, we present another CE scheme which requires the decryptor to hold all the roots (or cases) for each polynomial (or criterion).

As mentioned earlier, the idea behind our construction is to apply the Viète's formula which allows us to reconstruct a polynomial given all the roots of that polynomial.

- Setup(1^n, d): The setup algorithm is similar to that of the first scheme, except that we new represent the polynomials using the following coefficient vectors:

$$\begin{cases} \overrightarrow{P_1} = \left(1, \frac{a_{1_{d-1}}}{a_{1_d}}, \ldots, \frac{a_{1_0}}{a_{1_d}}\right) \\ \overrightarrow{P_2} = \left(1, \frac{a_{2_{d-1}}}{a_{2_d}}, \ldots, \frac{a_{2_0}}{a_{2_d}}\right) \\ \ldots \\ \overrightarrow{P_n} = \left(1, \frac{a_{n_{d-1}}}{a_{n_d}}, \ldots, \frac{a_{n_0}}{a_{n_d}}\right). \end{cases}$$

Then we choose $h_1, \ldots, h_n \in \mathbb{G}$ randomly, pick y, $a \in_R \mathbb{Z}_p$ and create the public parameters and master key as

$$PK = (g, e(g, g)^y, g^a, g^{tag_1\overrightarrow{P_1}}, \ldots, g^{tag_n\overrightarrow{P_n}}, h_1, \ldots, h_n)$$
$$MSK = g^y, g^{tag_1}, \ldots, g^{tag_n}.$$

- Encryption (M, PK, Pol, $\overrightarrow{\nu} = \{w_m\}_{m\in\{1,\ldots,n\}} \subset U$, τ): Similar to the Encryption of the first scheme, for the access policy $Pol = (\mathbb{A}, \rho)$ and a vector $\overrightarrow{\nu}$ which indicates the weight of each criterion, the algorithm chooses a random vector $\overrightarrow{x} = (s, y_2, \ldots, y_n) \in \mathbb{Z}_p^n$, which share the encryption exponent s. For $i = 1$ to l, it calculates

$\lambda_i = \overrightarrow{x} \cdot \mathbb{A}_i$, where $\mathbb{A}_i$ is the vector corresponding to the ith row of $\mathbb{A}$.

Next, the encryptor will set the weight to every criteria in $\overrightarrow{\nu}$. Then the encryptor identifies all the criterion set with accumulated weight larger than τ. Let $T = \{(k_1^i, k_2^i, \ldots, k_{\mu_i}^i)\}$ with μ_i being the maximum number combination of criteria i, which denotes such a set, where $k_j^i \in \{1, 2, \ldots, n\}$ denotes a position in the criterion Universe and computes

$$C_1 = M \cdot e(g, g)^{ys},\ C_2 = g^s,$$
$$C_i = \{g^{a\lambda_i} (g^{tag_{\rho(i)}\ \overrightarrow{P_{\rho(i)}}})^{-s}\}_{i=1,\ldots,l},$$
$$\left\{\hat{C}_i = \prod_{j=1}^{\mu_i} h_{k_j^i}^s\right\}_{i=1,\ldots,len(T)}.$$

The ciphertext is set as

$$CT = (\tau, Pol_1, C_1, C_2, \{C_i\}_{i=1,\ldots,l}, \{\hat{C}_i\}_{i=1,\ldots,len(T)}, T).$$

- KeyGen(MSK, $\mathcal{C}$): Let $\mathcal{C} = \{C_x\}$, where C_x denotes the full case set for criterion x. Let $root_x = \{x_1, x_2, \ldots, x_d\}$ be the full root set for the polynomial corresponding to x. The key generation authority applies the Viète's formula on $root_x$ to produce the following vector:

$$\overrightarrow{z_{tag_x}} = (1, z_{x_{d-1}}, z_{x_{d-2}}, \ldots, z_{x_0}).$$

For set $\mathcal{C}$, let $P = \{P_1, \ldots, P_{len(\mathcal{C})}\}$ denote the position of each corresponding criterion in the criterion Universe U. Also, let $S = \{(P_1), (P_2), \ldots, (P_1, P_2), \ldots, (P_1, P_2, \ldots, P_{len(\mathcal{C})})\} = \{(k_1^i, \ldots, k_{\nu_i}^i)\}(1 \le i \le 2^{len(\mathcal{C})})$ denote all the combinations of the elements in P. Next the authority chooses a random $t \in \mathbb{Z}_p$, and computes

$$L = g^t,\ \{K_x = (g^{tag_x})^{\overrightarrow{z_{tag_x}}t}\}_{C_x \in \mathcal{C}},$$
$$\left\{\hat{K}_i = g^y g^{at} \prod_{j=1}^{\nu_i} (h_{k_j^i})^t\right\}_{i=1,\ldots,len(S)}$$

The user secret key is set as

$$SK = (L, \{K_x\}_{C_x \in \mathcal{C}}, \{\hat{K}_i\}_{i=1,\ldots,len(S)}, S).$$

- Decryption(CT, SK): Similar to the decryption algorithm of the first scheme, the user first identifies the set I and computes $\{\omega_i \in \mathbb{Z}_p\}_{i \in I}$ based on the access structure($\mathbb{A}$, ρ). Then the user computes

$$\prod_{i \in I} (e(L, C_i) \cdot (e(K_{\rho(i)}, C_2))^{\omega_i}$$
$$= \left(\prod_{i \in I} (e(g^t, g^{a\lambda_i} g^{-tag_{\rho(i)}\overrightarrow{P_{\rho(i)}}s}) \cdot e((g^{tag_{\rho(i)}})^{\overrightarrow{z_{tag_{\rho(i)}}}t}, g^s))^{\omega_i}\right.$$
$$= \left(\prod_{i \in I} (e(g^t, g^{a\lambda_i})^{\omega_i}\right.$$
$$= \prod_{i \in I} e(g, g)^{a\omega_i\lambda_i t} = e(g, g)^{ast}.$$

Let $J = \{T \cap S_{C \vDash Pol}\}$, for each $i \in J$, let i_T denote the position of i in T, and i_S denote the position of i in S. The user also computes

$$\left(\prod_{i \in J} \frac{e(C_2, \hat{K_{i_S}})}{e(L, \hat{C_{i_T}})}\right)^{\frac{1}{|J|}} = \left(\prod_{i \in J} \frac{e(g^s, g^y g^{at} \prod_{j=1}^{len(i)} (h_{i_j})^t)}{e(g^t, \prod_{j=1}^{len(i)} (h_{i_j})^s)}\right)^{1/|J|}$$
$$= \left(\prod_{i \in J} e(g^s, g^y g^{at})\right)^{1/|J|}$$
$$= (e(g, g)^{ys} e(g, g)^{ast})^{|J|/|J|}$$
$$= e(g, g)^{ys} e(g, g)^{ast}.$$

Finally, compute $e(g, g)^{ys} e(g, g)^{ast} / e(g, g)^{ast} = e(g, g)^{ys}$ and recover message M by $C_1 / e(g, g)^{ys}$.

6. SECURITY PROOF

THEOREM 6.1. *Assume that the decisional q-Bilinear Diffie–Hellman Exponent assumption holds in* $\mathbb{G}$, *then no poly-time adversary can break the selective IND-CPA security of our* CE-Verify *Root of Polynomial scheme with a non-negligible advantage.*

Proof of Theorem 1. Suppose that there exists an adversary $\mathcal{A}$ who can win the Selective IND-CPA game with a non-negligible advantage ϵ. We present another algorithm $\mathcal{B}$ which can solve the decisional q-BDHE problem.

- *Init*: $\mathcal{B}$ takes an instance $\overrightarrow{y}$, T of the q-BDHE problem as an input. As required in the selective model, $\mathcal{A}$ first submits a challenge access structure $Pol^* = (M^*, \rho^*)$, where M^* has n^* columns, a set of weight every chosen criteria $\overrightarrow{\nu^*} = \{w_1, w_2, \ldots, w_m\}_{m \in \{1,\ldots,n\}}$, and a threshold τ^* to $\mathcal{B}$.
- *Setup*: $\mathcal{B}$ simulates the public parameter

$$PK = (g, e(g, g)^y, g^a, \alpha_1, \ldots, \alpha_n, \beta_1, \ldots, \beta_n, h_1, \ldots, h_n)$$

and the master secret key $MSK = (g^y, \gamma_1, \ldots, \gamma_n)$ as follows:

$\mathcal{B}$ chooses random $y' \in \mathbb{Z}_p$, and sets $e(g, g)^y = e(g^a, g^{a^q}) \cdot e(g, g)^{y'}$, which implicitly sets $y = y' + a^{q+1}$. For each d-degree polynomial P_x (representing a criterion) for $1 \leq x \leq n$, $\mathcal{B}$ chooses a random tag tag_x. Let X denote the set of indices i, such that $\rho^*(i) = P_x$. $\mathcal{B}$ then sets

$$\alpha_x = g^{tag_x \overrightarrow{P_x}} \prod_{i \in X} g^{a\mathcal{M}^*_{i,1}} \cdot g^{a^2\mathcal{M}^*_{i,2}} \ldots g^{a^{n^*}\mathcal{M}^*_{i,n^*}},$$

$$\beta_x = g^{\overrightarrow{P'_x}} \prod_{i \in X} g^{a\mathcal{M}^*_{i,1}} \cdot g^{a^2\mathcal{M}^*_{i,2}} \ldots g^{a^{n^*}\mathcal{M}^*_{i,n^*}},$$

$$\gamma_x = g^{tag_x} \prod_{i \in X} g^{a\mathcal{M}^*_{i,1}} \cdot g^{a^2\mathcal{M}^*_{i,2}} \ldots g^{a^{n^*}\mathcal{M}^*_{i,n^*}}.$$

If $X = \varnothing$, then simply set $\gamma_x = g^{tag_x}$, $\alpha_x = g^{tag_x \overrightarrow{P_x}}$, $\beta_x = g^{\overrightarrow{P'_x}}$, $\mathcal{B}$ then picks randomly $u_1, \ldots, u_n \in \mathbb{Z}_p$ and sets

$$h_j = \begin{cases} g^{\frac{u_j}{\delta}} \text{ if } P_j \text{ has combination} \leq \tau^* \\ g^{u_j} \text{ otherwise,} \end{cases},$$

with $\delta = w_1 + w_2 + \cdots + w_m$.

- *Phase* 1: $\mathcal{A}$ submits a private key query for $\mathcal{C} = \{z_{tag_x}\}$, where $\mathcal{C}$ does not satisfy $\mathcal{M}^*$. For each z_{tag_x} denote $\overrightarrow{z_{tag_x}} = (z^d_{tag_x}, z^{d-1}_{tag_x}, \ldots, z_{tag_x}, 1)$. $\mathcal{B}$ answers the query as follows.
  First $\mathcal{B}$ chooses $r \in_R \mathbb{Z}_p$. Then by the definition of LSSS $\mathcal{B}$ can find a vector $\overrightarrow{w} = (w_1, \ldots, w_{n^*}) \in \mathbb{Z}_p^{n^*}$ such that $w_1 = -1$ and for all i, where $\rho^*(i)$ has a root in $\mathcal{C}$ we have $\overrightarrow{w} \cdot \mathcal{M}^*_i = 0$. $\mathcal{B}$ then sets

$$L = g^t = g^r \cdot g^{a^q w_1} \cdot g^{a^{q-1} w_2} \ldots g^{a^{q-n^*+1} w_{n^*}} = g^r \prod_{i=1,\ldots,n^*} (g^{a^{q+1-i}})^{w_i},$$

  which implicitly sets $t = r + w_1 a^q + w_2 a^{q-1} + \cdots + w_{n^*} a^{q-n^*+1}$.
  For each $z_{tag_x} \in \mathcal{C}$, if there is no i such that z_{tag_x} is a root of $\rho^*(i)$, then $\mathcal{B}$ can simply let $K_x = L^{tag_x \overrightarrow{z_{tag_x}}}$ and $K'_x = L^{\overrightarrow{P'_x}}$. Otherwise, $\mathcal{B}$ computes

$$K_x = (g^{tag_x} \prod_{i \in X} g^{a\mathcal{M}^*_{i,1}} \cdot g^{a^2\mathcal{M}^*_{i,2}} \ldots g^{a^{n^*}\mathcal{M}^*_{i,n^*}})^{\overrightarrow{z_{tag_x} t}},$$

  where $t = r + w_1 a^q + w_2 a^{q-1} + \cdots + w_{n^*} a^{q-n^*+1}$. Since $\mathcal{M}^*_i \cdot \overrightarrow{w} = 0$ meaning $\mathcal{M}^*_{i,1} \cdot w_1 + \mathcal{M}^*_{i,2} \cdot w_2 + \cdots + \mathcal{M}^*_{i,n^*} \cdot w_{n^*} = 0$, then we can cancel the term of $g^{a^{q+1}}$, and express K_x as

$$K_x = L^{tag_x \overrightarrow{z_{tag_x}}} \prod_{i \in X} \prod_{j=1,\ldots,n^*} \left(g^{(a^j/)r} \prod_{k=\{1,\ldots,n^*\}, k \neq j} (g^{a^{q+1+j-k}})^{w_k} \right)^{\mathcal{M}^*_{i,j} \overrightarrow{z_{tag_x}}}.$$

  To compute $\hat{K}_i$ from $\mathcal{C}$, let $P = \{P_1, \ldots, P_{len(\mathcal{C})}\}$ denote the position of each corresponding criterion in the criterion Universe U. Also, let $S = \{(P_1), (P_2), \ldots, (P_1, P_2), \ldots, (P_1, P_2, \ldots, P_{len(\mathcal{C})})\} = \{(k^i_1, \ldots, k^i_{\nu_i})\} (1 \leq i \leq 2^{len(\mathcal{C})})$ denote all the combinations of the elements in P.
  For $1 \leq i \leq 2^{len(\mathcal{C})}$

$$\begin{aligned} \hat{K}_i &= g^y g^{at} \prod_{j=1}^{\nu_i} g^{u_{kj} t} \\ &= g^{y'} \cdot g^{a^{q+1}} \cdot g^{ar} \cdot g^{a^{q+1} w_1} \cdot g^{a^q w_2} \ldots g^{a^{q-n^*+2} w_{n^*}} L^{\sum_{j=1}^{\nu_i} u_{kj}} \\ &= g^{y'} \cdot g^{ar} \cdot g^{a^q w_2} \ldots g^{a^{q-n^*+2} w_{n^*}} L^{\sum_{j=1}^{\nu_i} u_{kj}} \\ &= g^{y'} \cdot g^{ar} \prod_{i=2,\ldots,n^*} (g^{a^{q+2-i}})^{w_i} L^{\sum_{j=1}^{\nu_i} u_{kj}}. \end{aligned}$$

  Otherwise, set

$$\begin{aligned} \hat{K}_i &= g^y g^{at} \prod_{j=1}^{\nu_i} g^{\frac{u_{kj} t}{\delta}} \\ &= g^{y'} \cdot g^{a^{q+1}} \cdot g^{ar} \cdot g^{a^{q+1} w_1} \cdot g^{a^q w_2} \ldots g^{a^{q-n^*+2} w_{n^*}} L^{\frac{\sum_{j=1}^{\nu_i} u_{kj}}{\delta}} \\ &= g^{y'} \cdot g^{ar} \cdot g^{a^q w_2} \ldots g^{a^{q-n^*+2} w_{n^*}} L^{\frac{\sum_{j=1}^{\nu_i} u_{kj}}{\delta}} \\ &= g^{y'} \cdot g^{ar} \prod_{i=2,\ldots,n^*} (g^{a^{q+2-i}})^{w_i} L^{\frac{\sum_{j=1}^{\nu_i} u_{kj}}{\delta}}. \end{aligned}$$

- *Challenge*: The adversary gives two messages M_0 and M_1 to the simulator. Write $h = g^s$ for some unknown s, the simulator $\mathcal{B}$ flips a coin β, then it chooses random $y'_2, \ldots, y'_{n^*}$ and share the secret s using the vector

$$\overrightarrow{v} = (s, sa + y'_2, sa^2 + y'_3, \ldots, sa^{n-1} + y'_{n^*}) \in \mathbb{Z}_p^{n^*}.$$

  From $\overrightarrow{v}^*$, $\mathcal{B}$ chooses all the criterion set with accumulated weight larger than τ^*. Let $T = \{(k^i_1, k^i_2, \ldots, k^i_{\mu_i})\}$ denote such a set, where $k^i_j \in \{1, 2, \ldots, n\}$ denotes a position in the criterion Universe. Then $\mathcal{B}$ creates the challenge ciphertext as

$$C_1 = M_\beta T \cdot e(h, g^{y'}),\ C_2 = h$$

$$\begin{aligned} C_i &= g^{a\overrightarrow{v}\mathcal{M}_i^*} g^{-tag_i\overrightarrow{P_i}s} \\ &= g^{as\mathcal{M}_{j,1}^*} \cdot g^{(sa^2+y_2')\mathcal{M}_{j,2}^*} \cdots g^{(a^{n^*}s+y_{n^*}')\mathcal{M}_{j,n^*}^*} \\ &\quad \cdot (g^s)^{-tag_i\overrightarrow{P_i}} \cdot g^{a\mathcal{M}_{i,1}^*} \\ &\quad \cdot g^{a^2\mathcal{M}_{i,2}^*} \cdots g^{a^{n^*}\mathcal{M}_{i,n^*}^*})^{-s} \\ &= (g^s)^{-tag_i\overrightarrow{P_i}} \left(\prod_{j=1,\ldots,n^*} (g^a)^{\mathcal{M}_{i,j}^* y_j'} \right) \end{aligned}$$

$$\begin{aligned} C_i' &= g^{a\overrightarrow{v}\mathcal{M}_i^*} g^{-\overrightarrow{P'}_i s} \\ &= g^{as\mathcal{M}_{j,1}^*} \cdot g^{(sa^2+y_2')\mathcal{M}_{j,2}^*} \cdots g^{(a^{n^*}s+y_{n^*}')\mathcal{M}_{j,n^*}^*} \\ &\quad \cdot (g^s)^{-\overrightarrow{P'}_i} \cdot g^{a\mathcal{M}_{i,1}^*} \cdot g^{a^2\mathcal{M}_{i,2}^*} \cdots g^{a^{n^*}\mathcal{M}_{i,n^*}^*})^{-s} \\ &= (g^s)^{-\overrightarrow{P'}_i} \left(\prod_{j=1,\ldots,n^*} (g^a)^{\mathcal{M}_{i,j}^* y_j'} \right) \end{aligned}$$

$$\left\{ \hat{C}_i = \prod_{j=1}^{\mu_i} h_{k_j^i}^{s} \right\}_{i=1,\ldots,len(T)}.$$

If $T = e(g, g)^{\alpha^{q+1}s}$, the challenge ciphertext is a valid encryption of M_β. On the other hand, if T is uniformly distributed in $\mathbb{G}_T$, the challenge ciphertext is independent of β.

- *Phase* 2: This phase is simulated as in *Phase*1.
- *Guess*: $\mathcal{A}$ output $\beta' \in \{0, 1\}$. If $\beta' = \beta$ then B outputs 1, otherwise outputs 0.
  If $\beta' = 0$, then the simulation is the same as in the real game. Hence, $\mathcal{A}$ will have the probability $\frac{1}{2} + \epsilon$ to guess β correctly.
  If $\beta' = 1$, then T is random in $\mathbb{G}$, then $\mathcal{A}$ will have probability $\frac{1}{2}$ to guess β correctly. Therefore, $\mathcal{B}$ can solve the decision q-BDHE assumption also with advantage ϵ. □

THEOREM 6.2. *Assume that the decisional q-Bilinear Diffie–Hellman Exponent assumption holds in* $\mathbb{G}$, *then no poly-time adversary can break the selective IND-CPA security of our* CE-Equal Coefficients *scheme with a non-negligible advantage*.

Proof of Theorem 2. Suppose that there exists an adversary $\mathcal{A}$ who can win the Selective IND-CPA game with a non-negligible advantage ϵ. We present another algorithm $\mathcal{B}$ which can solve the decisional q-BDHE problem:

- *Init*: $\mathcal{B}$ takes an instance $\overrightarrow{y}$, T of the q-BDHE problem as an input. As required in the selective model, $\mathcal{A}$ first submits a challenge access structure $Pol^* = (M^*, \rho^*)$, where M^* has n^* columns, a set of weight every chosen criteria $\overrightarrow{\nu^*} = \{w_1, w_2, \ldots, w_m\}_{m\in\{1,\ldots,n\}}$, and a threshold τ^* to $\mathcal{B}$.
- *Setup*: $\mathcal{B}$ simulates the public parameter

$$PK = (g, e(g, g)^y, g^a, \alpha_1, \ldots, \alpha_n, h_1, \ldots, h_n)$$

and the master secret key $MSK = (g^y, \gamma_1, \ldots, \gamma_n)$ as follows:
$\mathcal{B}$ chooses random $y' \in \mathbb{Z}_p$, and sets $e(g, g)^y = e(g^a, g^{a^q}) \cdot e(g, g)^{y'}$, which implicitly sets $y = y' + a^{q+1}$. For each d-degree polynomial P_x (representing a criterion) for $1 \le x \le n$, $\mathcal{B}$ chooses a random tag tag_x. Let$X$ denote the set of indices i, such that $\rho^*(i) = P_x$. $\mathcal{B}$ then sets

$$\alpha_x = g^{tag_x\overrightarrow{P_x}} \prod_{i\in X} g^{a\mathcal{M}_{i,1}^*} \cdot g^{a^2\mathcal{M}_{i,2}^*} \cdots g^{a^{n^*}\mathcal{M}_{i,n^*}^*},$$

$$\gamma_x = g^{tag_x} \prod_{i\in X} g^{a\mathcal{M}_{i,1}^*} \cdot g^{a^2\mathcal{M}_{i,2}^*} \cdots g^{a^{n^*}\mathcal{M}_{i,n^*}^*}.$$

If $X = \varnothing$ then simply set $\gamma_x = g^{tag_x}$, $\alpha_x = g^{tag_x\overrightarrow{P_x}}$, $\beta_x = g^{\overrightarrow{P'}_x}$. $\mathcal{B}$ then picks randomly $u_1, \ldots, u_n \in \mathbb{Z}_p$ and sets

$$h_j = \begin{cases} g^{\frac{u_j}{\delta}} \text{ if } P_j \text{ has combination} \le \tau^* \\ g^{u_j} \text{ otherwise.} \end{cases},$$

with $\delta = w_1 + w_2 + \cdots + w_m$.

- *Phase* 1: $\mathcal{A}$ submits a private key query for $\mathcal{C} = \{z_{tag_x}\}$, where $\mathcal{C}$ does not satisfy $\mathcal{M}^*$. For each z_{tag_x} include a full set root as $\{x_1, x_2, \ldots, x_d\}$, then applying the Viète's formula on this set to produce the following vector $\overrightarrow{z_{tag_x}} = (1, z_{x_{d-1}}, z_{x_{d-2}}, \ldots, z_{x_0})$. $\mathcal{B}$ answers the query as follows:
  First $\mathcal{B}$ chooses $r \in_R \mathbb{Z}_p$. Then by the definition of LSSS $\mathcal{B}$ can find a vector $\overrightarrow{w} = (w_1, \ldots, w_{n^*}) \in \mathbb{Z}_p^{n^*}$ such that $w_1 = -1$ and for all i, where $\rho^*(i)$ has a root in $\mathcal{C}$ we have $\overrightarrow{w} \cdot \mathcal{M}_i^* = 0$. $\mathcal{B}$ then sets

$$\begin{aligned} L = g^t &= g^r \cdot g^{a^q w_1} \cdot g^{a^{q-1}w_2} \cdots g^{a^{q-n^*+1}w_{n^*}} \\ &= g^r \prod_{i=1,\ldots,n^*} (g^{a^{q+1-i}})^{w_i}, \end{aligned}$$

which implicitly sets $t = r + w_1 a^q + w_2 a^{q-1} + \cdots + w_{n^*} a^{q-n^*+1}$.
For each $z_{tag_x} \in \mathcal{C}$, if there is no i such that $\frac{z_{tag_x}}{z_{tag_x}}$ is a root of $\rho^*(i)$, then $\mathcal{B}$ can simply let $K_x = L^{tag_x \overrightarrow{z_{tag_x}}}$ and $K_x' = L^{\overrightarrow{P'}_x}$. Otherwise, $\mathcal{B}$ computes

$$K_x = \left(g^{tag_x} \prod_{i\in X} g^{a\mathcal{M}_{i,1}^*} \cdot g^{a^2\mathcal{M}_{i,2}^*} \cdots g^{a^{n^*}\mathcal{M}_{i,n^*}^*} \right)^{\overrightarrow{z_{tag_x}t}},$$

where $t = r + w_1 a^q + w_2 a^{q-1} + \cdots + w_{n^*} a^{q-n^*+1}$. Since $\mathcal{M}_i^* \cdot \overrightarrow{w} = 0$ meaning $\mathcal{M}_{i,1}^* \cdot w_1 + \mathcal{M}_{i,2}^* \cdot w_2 +$

TABLE 2. Computation cost of our CE schemes.

Scheme	No. of root	Ciphertext size	Key size	Dec cost
CE-1	1	$\lvert\mathbb{G}_T\rvert + (2l + len + 1)\lvert\mathbb{G}\rvert$	$(2n' + len + 1)\lvert\mathbb{G}\rvert$	$(3n' + 2j)$ **p**
CE-2	d	$\lvert\mathbb{G}_T\rvert + (l + len + 1)\lvert\mathbb{G}\rvert$	$(n' + len + 1)\lvert\mathbb{G}\rvert$	$(2n' + 2j)$ **p**

$\cdots + \mathcal{M}^*_{i,n^*} \cdot w_{n^*} = 0$, then we can cancel the term of $g^{a^{q+1}}$, and express K_x as

$$K_x = L^{tag_x \overrightarrow{z_{tag_x}}} \prod_{i \in X} \prod_{j=1,\ldots,n^*} \left(g^{(a^j/)r} \prod_{k=\{1,\ldots,n^*\}, k \neq j} (g^{a^{q+1+j-k}})^{w_k} \right)^{\mathcal{M}^*_{i,j} \overrightarrow{z_{tag_x}}}.$$

To compute $\hat{K}_i$ from $\mathcal{C}$, let $P = \{P_1, \ldots, P_{len(\mathcal{C})}\}$ denote the position of each corresponding criterion in the criterion Universe U. Also, let $S = \{(P_1), (P_2), \ldots, (P_1, P_2), \ldots, (P_1, P_2, \ldots, P_{len(\mathcal{C})})\} = \{(k^i_1, \ldots, k^i_{\nu_i})\} (1 \leq i \leq 2^{len(\mathcal{C})})$ denote all the combinations of the elements in P.
For $1 \leq i \leq 2^{len(\mathcal{C})}$

$$\begin{aligned}\hat{K}_i &= g^y g^{at} \prod_{j=1}^{\nu_i} g^{u_{kj} t} \\ &= g^{y'} \cdot g^{a^{q+1}} \cdot g^{ar} \cdot g^{a^{q+1} w_1} \\ &\quad \cdot g^{a^q w_2} \cdots g^{a^{q-n^*+2} w_{n^*}} L^{\sum_{j=1}^{\nu_i} u_{kj}} \\ &= g^{y'} \cdot g^{ar} \cdot g^{a^q w_2} \cdots g^{a^{q-n^*+2} w_{n^*}} L^{\sum_{j=1}^{\nu_i} u_{kj}} \\ &= g^{y'} \cdot g^{ar} \prod_{i=2,\ldots,n^*} (g^{a^{q+2-i}})^{w_i} L^{\sum_{j=1}^{\nu_i} u_{kj}}.\end{aligned}$$

Otherwise, set

$$\begin{aligned}\hat{K}_i &= g^y g^{at} \prod_{j=1}^{\nu_i} g^{\frac{u_{kj} t}{\delta}} \\ &= g^{y'} \cdot g^{a^{q+1}} \cdot g^{ar} \cdot g^{a^{q+1} w_1} \\ &\quad \cdot g^{a^q w_2} \cdots g^{a^{q-n^*+2} w_{n^*}} L^{\frac{\sum_{j=1}^{\nu_i} u_{kj}}{\delta}} \\ &= g^{y'} \cdot g^{ar} \cdot g^{a^q w_2} \cdots g^{a^{q-n^*+2} w_{n^*}} L^{\frac{\sum_{j=1}^{\nu_i} u_{kj}}{\delta}} \\ &= g^{y'} \cdot g^{ar} \prod_{i=2,\ldots,n^*} (g^{a^{q+2-i}})^{w_i} L^{\frac{\sum_{j=1}^{\nu_i} u_{kj}}{\delta}}.\end{aligned}$$

- *Challenge*: The adversary gives two messages M_0 and M_1 to the simulator. Write $h = g^s$ for some unknown s, the simulator $\mathcal{B}$ flips a coin β, then it chooses random $y'_2, \ldots, y'_{n^*}$ and share the secret s using the vector

$$\overrightarrow{v} = (s, sa + y'_2, sa^2 + y'_3, \ldots, sa^{n-1} + y'_{n^*}) \in \mathbb{Z}^{n^*}_p.$$

From $\overrightarrow{v}^*$, $\mathcal{B}$ chooses all the criterion set with accumulated weight larger than τ^*. Let $T = \{(k^i_1, k^i_2, \ldots, k^i_{\mu_i})\}$ denote such a set, where $k^i_j \in \{1, 2, \ldots, n\}$ denotes a position in the criterion Universe. Then $\mathcal{B}$ creates the challenge ciphertext as

$$\begin{aligned}C_1 &= M_\beta T \cdot e(h, g^{y'}), C_2 = h, \\ C_i &= g^{a \overrightarrow{v}} \mathcal{M}^*_i g^{-tag_i \overrightarrow{P_i} s} \\ &= g^{as \mathcal{M}^*_{j,1}} \cdot g^{(sa^2 + y'_2) \mathcal{M}^*_{j,2}} \cdots g^{(a^{n^*} s + y'_{n^*}) \mathcal{M}^*_{j,n^*}} \\ &\quad \cdot (g^s)^{-tag_i \overrightarrow{P_i}} \cdot g^{a \mathcal{M}^*_{i,1}} \cdot g^{a^2 \mathcal{M}^*_{i,2}} \cdots g^{a^{n^*} \mathcal{M}^*_{i,n^*}})^{-s} \\ &= (g^s)^{-tag_i \overrightarrow{P_i}} \left(\prod_{j=1,\ldots,n^*} (g^a)^{\mathcal{M}^*_{i,j} y'_j} \right), \\ &\quad \left\{ \hat{C}_i = \prod_{j=1}^{\mu_i} h^s_{k^i_j} \right\}_{i=1,\ldots,len(T)}.\end{aligned}$$

If $T = e(g, g)^{\alpha^{q+1} s}$, the challenge ciphertext is a valid encryption of M_β. On the other hand, if T is uniformly distributed in $\mathbb{G}_T$, the challenge ciphertext is independent of β.

- *Phase* 2: This phase is simulated as in *Phase* 1.
- *Guess*: $\mathcal{A}$ output $\beta' \in \{0, 1\}$. If $\beta' = \beta$ then B outputs 1, otherwise outputs 0.

If $\beta' = 0$, then the simulation is the same as in the real game. Hence, $\mathcal{A}$ will have the probability $\frac{1}{2} + \epsilon$ to guess β correctly.

If $\beta' = 1$, then T is random in $\mathbb{G}$, then $\mathcal{A}$ will have probability $\frac{1}{2}$ to guess β correctly. Therefore, $\mathcal{B}$ can solve the decision q-BDHE assumption also with advantage ϵ. □

7. SPECIAL CASE OF CE

In this section, we consider the special case of CE where we do not require the decryptor to satisfy the weight requirement. We show that under this new setting, we can construct much more efficient CE schemes. It is also worth noting that we may use an ABE to implement a CE where a criterion can be treated as a simple access structure containing only AND or OR operation.

7.1. SCE-Verify Root of Polynomial

- Setup $(1^n, d)$: The key generation authority first chooses a group $\mathbb{G}$ of prime order p and a generator g. Then it setups the system with the set of criteria, which is express by the set of d-degree polynomials: $\{P_1, P_2, \ldots, P_n\}$. Each criterion is labeled by choosing random numbers $tag_1, tag_2, \ldots, tag_n \in_R \mathbb{Z}_p$.
  Every polynomial has the set of coefficients $a_d, a_{d-1}, \ldots, a_0$, then it creates the vectors corresponding to all the polynomials based on the their coefficients:

$$\begin{cases} \overrightarrow{P_1} = \left(a_{1_d}, a_{1_{d-1}}, \ldots, a_{1_0}, \dfrac{1}{tag_1}\right) \\ \overrightarrow{P_1'} = (a_{1_d}, a_{1_{d-1}}, \ldots, a_{1_0}, 1) \\ \overrightarrow{P_2} = \left(a_{2_d}, a_{2_{d-1}}, \ldots, a_{2_0}, \dfrac{1}{tag_2}\right) \\ \overrightarrow{P_2'} = (a_{2_d}, a_{2_{d-1}}, \cdots, a_{2_0}, 1) \\ \ldots \\ \overrightarrow{P_n} = \left(a_{n_d}, a_{n_{d-1}}, \ldots, a_{n_0}, \dfrac{1}{tag_n}\right) \\ \overrightarrow{P_n'} = (a_{n_d}, a_{n_{d-1}}, \ldots, a_{n_0}, 1). \end{cases}$$

  Then we choose $y, a \in_R \mathbb{Z}_p$, and create the public parameters and master key as

$$PK = (g, e(g, g)^y, g^a, g^{tag_1 \overrightarrow{P_1}}, \ldots, g^{tag_n \overrightarrow{P_n}}, g^{\overrightarrow{P_1'}}, \ldots, g^{\overrightarrow{P_n'}})$$
$$MSK = (g^y, g^{tag_1}, \ldots, g^{tag_n}).$$

- Encryption(M, PK, Pol): Given a message $M \in \mathbb{G}_T$ and an access policy $Pol = (\mathbb{A}, \rho)$, the algorithm chooses a random vector $\overrightarrow{v} = (s, y_2, \ldots, y_n) \in \mathbb{Z}_p^n$. These values will be used to share the encryption exponent s. For $i = 1$ to l, it calculates $\lambda_i = \overrightarrow{v} \cdot \mathbb{A}_i$, where $\mathbb{A}_i$ is the vector corresponding to the ith row of $\mathbb{A}$.
  Then it computes

$$C_1 = M \cdot e(g, g)^{ys}, \; C_2 = g^s,$$
$$C_i = \{g^{a\lambda_i} g^{-tag_{\rho(i)} \overrightarrow{P_{\rho(i)}} s}\},$$
$$C_i' = \{g^{a\lambda_i} g^{-\overrightarrow{P'}_{\rho(i)} s}\},$$

  and output the ciphertext as

$$CT = (Pol, C_1, C_2, \{C_i, C_i'\}_{i=1,\ldots,l}).$$

- KeyGen$(MSK, \mathcal{C})$: Similar to the key Generation algorithm of our first CE scheme, we create a vector for criterion x as follows, where z_{tag_x} denotes a root of the polynomial corresponding to x

$$\overrightarrow{z_{tag_x}} = (z_{tag_x}^d, z_{tag_x}^{d-1}, \ldots, z_{tag_x}, 1).$$

  Next it chooses a random $t \in \mathbb{Z}_p$, and computes

$$K = g^y g^{at}, \; L = g^t, \; \forall C_x \in \mathcal{C},$$
$$\begin{cases} K_x = (g^{tag_x})^{\overrightarrow{z_{tag_x}} t}, \\ K_x' = g^{\overrightarrow{P'}_x t}. \end{cases}$$

  The secret key is set as $SK = (K, L, \{K_x, K_x'\}_{C_x \in \{\mathcal{C}\}})$.
- Decryption (CT, SK): Similar to the decryption algorithm of the first CE scheme, the user first identifies the set I and computes $\{\omega_i \in \mathbb{Z}_p\}_{i \in I}$ based on the access structure$(\mathbb{A}, \rho)$.
  Then it computes

$$\left(\prod_{i \in I} e(L, C_i') \cdot \frac{e(K_{\rho(i)}, C_i) e(K_{\rho(i)}', C_2)}{e(K_{\rho(i)}, C_i')}\right)^{\omega_i}$$
$$= \left(\prod_{i \in I} e(g^t, g^{a\lambda_i} g^{-\overrightarrow{P'}_{\rho(i)} s}) \cdot \frac{e((g^{tag_{\rho(i)}})^{\overrightarrow{z_{tag_{\rho(i)}}} t}, g^{a\lambda_i} g^{-tag_{\rho(i)} \overrightarrow{P}_{\rho(i)} s}) e(g^{\overrightarrow{P'}_{\rho(i)} t}, g^s)}{e((g^{tag_{\rho(i)}})^{\overrightarrow{z_{tag_{\rho(i)}}} t}, g^{a\lambda_i} g^{-\overrightarrow{P'}_{\rho(i)} s})}\right)^{\omega_i}$$
$$= \prod_{i \in I} e(g, g)^{a\lambda_i t \omega_i} = e(g, g)^{ast}.$$

  After that, compute $e(K, C_2)/e(g, g)^{ast} = e(g, g)^{ys}$ and recover message M by $C_1/e(g, g)^{ys}$.

It is easy to see that we can also modify the CE-Equal Coefficients scheme in a similar way. The security of these special CE schemes can be proved by following the proofs for the original CE schemes. In A, we provide the proof for the SCE-Verify Root of Polynomial scheme presented above.

7.2. Security model

Similar to other public-key encryption schemes, we define the IND-CPA security of a SCE scheme via the following game:

- *Setup*: The challenger runs the Setup algorithm and gives the public parameters PK to the adversary.
- *Phase* 1: The adversary adaptively makes private key generation queries for any cases criteria set $\mathcal{C}$ of its choice.
- *Challenge*: The adversary submits two equal length messages M_0 and M_1 and a challenge access structure Pol^* with the restriction that Pol^* cannot be satisfied by any cases criteria set $\mathcal{C}$ occurred in Phase 1. The

challenger then flips a random coin β, and encrypts M_β under Pol^*. The resulting ciphertext CT^* is given to the adversary.
- *Phase*2: Phase 1 is repeated with the restriction that Pol^* cannot be satisfied by any cases criteria set appeared in the private key generation queries.
- *Guess*: The adversary outputs a guess β' of β.

We say a SCE scheme is IND-CPA secure if for any probabilistic polynomial-time adversary $\mathcal{A}$

$$\mathbf{Adv}_{\mathcal{A}}^{\text{IND-CPA}}(k) = \left| \Pr[\beta' = \beta] - \frac{1}{2} \right|$$

is negligible in the security parameter k.

8. COMPARISONS

Table 2 gives a detailed comparison between the two proposed CE schemes. The schemes are compared in terms of the number of roots that must be held by the user for each polynomial, the ciphertext size, the secret key size and the decryption cost. In the table, **p** denotes the number of pairing operation, l is the number of rows in the access structure matrix, n' is the number of criteria a user satisfies, n is the total number of criteria in the Universe, $len \ll 2^n - 1$ as all the criterion set with accumulated weight larger than τ and j is the size of the set J.

9. DISCUSSION AND CONCLUSION

In this paper, we introduced a new type of public-key encryption named Criteria-based Encryption, which is a new cryptographic primitive that allows fine-grained access control over encrypted data. We also proposed two concrete criteria-based encryption schemes under two different settings, and proved their security under the decisional q-Bilinear Diffie–Hellman Exponent problem. One limitation of our schemes is that the criteria space cannot be too large due to the use of the frame vector in handling the weight. An interesting research problem is to find another more efficient way to implement the weight requirement, and we leave it as our future work.

APPENDIX A. SECURITY PROOF FOR SCE

THEOREM A.1. *Assume that the decisional q-bilinear Diffie–Hellman Exponent problem holds in* $\mathbb{G}$, *then no poly-time adversary can selectively break our* SCE-Verify Roots of Polynomial.

Proof of Theorem 3. We prove the security of our SCE scheme under the decisional q-bilinear Diffie–Hellman Exponent assumption. Suppose that an adversary $\mathcal{A}$ with non-negligible advantage ϵ chooses a challenge matrix $\mathcal{M}^*$ of size $l^* \times n^*$, where $n^* \leq q$ in our proposed scheme. We represent how to build a simulator $\mathcal{B}$ that solves the decisional q-BDHE problem.

- *Init*: $\mathcal{B}$ takes an instance $\overrightarrow{y}, T$ of the q-BDHE problem as an input. As required in the selective model, $\mathcal{A}$ first submits a challenge access structure $Pol^* = (M^*, \rho^*)$, where M^* has n^* columns to $\mathcal{B}$.
- *Setup:* $\mathcal{B}$ simulates the public parameter

$$PK = (g, e(g,g)^y, g^a, \alpha_1, \ldots, \alpha_n, \beta_1, \ldots, \beta_n, h_1, \ldots, h_{len}),$$

and the master secret key $MSK = (g^y, \gamma_1, \ldots, \gamma_n)$ as follows:
$\mathcal{B}$ chooses random $y'_{,} \in \mathbb{Z}_p$, and sets $e(g,g)^y = e(g^a, g^{a^q}) \cdot e(g,g)^{y'}$, which implicitly sets $y = y' + a^{q+1}$. For each d-degree polynomial P_x (representing a criterion) for $1 \leq x \leq n$, $\mathcal{B}$ chooses a random tag tag_x. Let X denote the set of indices i, such that $\rho^*(i) = P_x$. $\mathcal{B}$ then sets

$$\alpha_x = g^{tag_x \overrightarrow{P_x}} \prod_{i \in X} g^{a\mathcal{M}^*_{i,1}} \cdot g^{a^2 \mathcal{M}^*_{i,2}} \cdots g^{a^{n^*} \mathcal{M}^*_{i,n^*}},$$

$$\beta_x = g^{\overrightarrow{P'}_x} \prod_{i \in X} g^{a\mathcal{M}^*_{i,1}} \cdot g^{a^2 \mathcal{M}^*_{i,2}} \cdots g^{a^{n^*} \mathcal{M}^*_{i,n^*}}.$$

$$\gamma_x = g^{tag_x} \prod_{i \in X} g^{a\mathcal{M}^*_{i,1}} \cdot g^{a^2 \mathcal{M}^*_{i,2}} \cdots g^{a^{n^*} \mathcal{M}^*_{i,n^*}}.$$

If $X = \varnothing$, then simply set $\gamma_x = g^{tag_x}$, $\alpha_x = g^{tag_x \overrightarrow{P_x}}$, $\beta_x = g^{\overrightarrow{P'}_x}$.
- *Phase* 1: $\mathcal{A}$ submits a private key query for $\mathcal{C} = \{z_{tag_x}\}$, where $\mathcal{C}$ does not satisfy $\mathcal{M}^*$. For each z_{tag_x} denote $\overrightarrow{z_{tag_x}} = (z^d_{tag_x}, z^{d-1}_{tag_x}, \ldots, z_{tag_x}, 1)$. $\mathcal{B}$ answers the query as follows.
First $\mathcal{B}$ chooses $r \in_R \mathbb{Z}_p$. Then by the definition of LSSS $\mathcal{B}$ can find a vector $\overrightarrow{w} = (w_1, \ldots, w_{n^*}) \in \mathbb{Z}_p^{n^*}$ such that $w_1 = -1$ and for all i, where $\rho^*(i)$ has a root in $\mathcal{C}$ we have $\overrightarrow{w} \cdot \mathcal{M}^*_i = 0$. $\mathcal{B}$ then sets

$$L = g^t = g^r \cdot g^{a^q w_1} \cdot g^{a^{q-1} w_2} \cdots g^{a^{q-n^*+1} w_{n^*}} = g^r \prod_{i=1,\ldots,n^*} (g^{a^{q+1-i}})^{w_i},$$

which implicitly sets $t = r + w_1 a^q + w_2 a^{q-1} + \cdots + w_{n^*} a^{q-n^*+1}$.

To compute K and we show how to cancel the term $g^{a^{q+1}}$ as

$$\begin{aligned}K &= g^{y}g^{at}\\ &= g^{y'} \cdot g^{a^{q+1}} \cdot g^{ar} \cdot g^{a^{q+1}w_1} \cdot g^{a^q w_2} \cdots g^{a^{q-n^*+2}w_{n^*}}\\ &= g^{y'} \cdot g^{ar} \cdot g^{a^q w_2} \cdots g^{a^{q-n^*+2}w_{n^*}}\\ &= g^{y'} \cdot g^{ar} \prod_{i=2,\ldots,n^*} (g^{a^{q+2-i}})^{w_i},\end{aligned}$$

for $w_1 = -1$.

For each $z_{tag_x} \in \mathcal{C}$, if there is no i such that z_{tag_x} is a root of $\rho^*(i)$, then $\mathcal{B}$ can simply let $K_x = L^{tag_x \overrightarrow{z_{tag_x}}}$ and $K'_x = L^{\overrightarrow{P'}_x}$. Otherwise, $\mathcal{B}$ computes:

$$K_x = \left(g^{tag_x} \prod_{i \in X} g^{a\mathcal{M}^*_{i,1}} \cdot g^{a^2\mathcal{M}^*_{i,2}} \cdots g^{a^{n^*}\mathcal{M}^*_{i,n^*}} \right)^{\overrightarrow{z_{tag_x}t}},$$

where $t = r + w_1 a^q + w_2 a^{q-1} + \cdots + w_{n^*} a^{q-n^*+1}$. Since $\mathcal{M}^*_i \cdot \overrightarrow{w} = 0$ meaning $\mathcal{M}^*_{i,1} \cdot w_1 + \mathcal{M}^*_{i,2} \cdot w_2 + \cdots + \mathcal{M}^*_{i,n^*} \cdot w_{n^*} = 0$, then we can cancel the term of $g^{a^{q+1}}$, and express K_x as

$$\begin{aligned}K_x &= L^{tag_x \overrightarrow{z_{tag_x}}} \prod_{i \in X} \prod_{j=1,\ldots,n^*} \Big(g^{(a^j/)r}\\ &\qquad \prod_{k=\{1,\ldots,n^*\}, k \neq j} (g^{a^{q+1+j-k}})^{w_k} \Big)^{\mathcal{M}^*_{i,j} \overrightarrow{z_{tag_x}}},\\ K'_x &= L^{\overrightarrow{P'}_x}.\end{aligned}$$

- *Challenge*: The adversary gives two messages M_0 and M_1 to the simulator. Write $h = g^s$ for some unknown s, the simulator $\mathcal{B}$ flips a coin β, then it chooses random $y'_2, \ldots, y'_{n^*}$ and share the secret s using the vector:

$$\overrightarrow{v} = (s, sa + y'_2, sa^2 + y'_3, \ldots, sa^{n-1} + y'_{n^*}) \in \mathbb{Z}_p^{n^*}.$$

$\mathcal{B}$ creates the challenge ciphertext as

$$\begin{aligned}C_1 &= M_\beta T \cdot e(h, g^{y'}),\ C_2 = h,\\ C_i &= g^{a\overrightarrow{v}\mathcal{M}^*_i} g^{-tag_i \overrightarrow{P_i} s}\\ &= g^{as\mathcal{M}^*_{j,1}} \cdot g^{(sa^2+y'_2)\mathcal{M}^*_{j,2}} \cdots g^{(a^{n^*}s+y'_{n^*})\mathcal{M}^*_{j,n^*}} \cdot\\ &\quad (g^s)^{-tag_i \overrightarrow{P_i}} \cdot g^{a\mathcal{M}^*_{i,1}} \cdot\\ &\quad g^{a^2\mathcal{M}^*_{i,2}} \cdots g^{a^{n^*}\mathcal{M}^*_{i,n^*}})^{-s}\\ &= (g^s)^{-tag_i \overrightarrow{P_i}} \left(\prod_{j=1,\ldots,n^*} (g^a)^{\mathcal{M}^*_{i,j} y'_j} \right),\\ C'_i &= g^{a\overrightarrow{v}\mathcal{M}^*_i} g^{-\overrightarrow{P'}_i s}\\ &= g^{as\mathcal{M}^*_{j,1}} \cdot g^{(sa^2+y'_2)\mathcal{M}^*_{j,2}} \cdots g^{(a^{n^*}s+y'_{n^*})\mathcal{M}^*_{j,n^*}}.\\ &\quad (g^s)^{-\overrightarrow{P'}_i} \cdot (g^{a\mathcal{M}^*_{i,1}} \cdot g^{a^2\mathcal{M}^*_{i,2}} \cdots g^{a^{n^*}\mathcal{M}^*_{i,n^*}})^{-s}\\ &= (g^s)^{-\overrightarrow{P'}_i} \left(\prod_{j=1,\ldots,n^*} (g^a)^{\mathcal{M}^*_{i,j} y'_j} \right).\end{aligned}$$

If $T = e(g, g)^{\alpha^{q+1}s}$, the challenge ciphertext is a valid encryption of M_β. On the other hand, if T is uniformly distributed in $\mathbb{G}_T$, the challenge ciphertext is independent of β.

- *Phase* 2: This phase is simulated as in *Phase* 1.
- *Guess*: $\mathcal{A}$ output $\beta' \in \{0, 1\}$. If $\beta' = \beta$, then B outputs 1, otherwise outputs 0.

  If $\beta' = 0$, then the simulation is the same as in the real game. Hence, $\mathcal{A}$ will have the probability $\frac{1}{2} + \epsilon$ to guess β correctly.

  If $\beta' = 1$, then T is random in $\mathbb{G}$, then $\mathcal{A}$ will have probability $\frac{1}{2}$ to guess β correctly. Therefore, $\mathcal{B}$ can solve the decision q-BDHE assumption also with advantage ϵ. □

REFERENCES

[1] Abdalla, M., Bourse, F., De Caro, A. and Pointcheval, D. (2015) Simple Functional Encryption Schemes for Inner Products. In *Public-Key Cryptography – PKC 2015*, pp. 733–751.

[2] Abdalla, M., Gay, R., Raykova, M. and Wee, H. (2017) Multi-input Inner-product Functional Encryption from Pairings. In *Advances in Cryptology – EUROCRYPT 2017*, pp. 601–626.

[3] Agrawal, S., Agrawal, S., Badrinarayanan, S., Kumarasubramanian, A., Prabhakaran, M. and Sahai, A. (2015) On the Practical Security of Inner Product Functional Encryption. In *Public-Key Cryptography – PKC 2015*, pp. 777–798.

[4] Agrawal, S., Freeman, D. and Vaikuntanathan, V. (2011) Functional Encryption for Inner Product Predicates from Learning with Errors. In *Advances in Cryptology – ASIACRYPT 2011*, Volume 7073 of *Lecture Notes in Computer Science*, pp. 21–40.

[5] Agrawal, S., Freeman, D. M. and Vaikuntanathan, V. (2011) Functional Encryption for Inner Product Predicates from Learning with Errors. In *Proc. 17th Int. Conf. Theory and Application of Cryptology and Information Security, ASIACRYPT'11*, pp. 21–40, Springer-Verlag, Berlin, Heidelberg.

[6] Agrawal, S., Gorbunov, S., Vaikuntanathan, V. and Wee, H. (2013) Functional Encryption: New Perspectives and Lower Bounds, *Advances in Cryptology – Crypto 2013*. pp. 500–518.

[7] Attrapadung, N., Libert, B. and Panafieu, E. (2011) Expressive Key-Policy Attribute-Based Encryption with Constant-size Ciphertexts. In *Public Key Cryptography – PKC 2011*, Volume 6571 of *Lecture Notes in Computer Science*, pp. 90–108.

[8] Bethencourt, J., Sahai, A. and Waters, B. (2007) Ciphertext-policy Attribute-based Encryption. In *IEEE Symposium on Security and Privacy, 2007*. SP '07, pp. 321–334.

[9] Boneh, D. and Boyen, X. (2004) Efficient Selective-id Secure Identity-Based Encryption without Random Oracles. In *Advances in Cryptology – EUROCRYPT 2004*, Volume 3027 of *Lecture Notes in Computer Science*, pp. 223–238.

[10] Boneh, D. and Franklin, M. (2001) Identity-based Encryption from the Weil Pairing. In *Advances in Cryptology – CRYPTO 2001*, Volume 2139 of *Lecture Notes in Computer Science*, pp. 213–229.

[11] Boneh, D., Sahai, A. and Waters, B. (2011) Functional Encryption: Definitions and Challenges. In *Theory of Cryptography: 8th Theory of Cryptography Conf.*, *TCC*, pp. 253–273.

[12] Boneh, D. and Waters, B. (2007) Conjunctive, Subset, and Range Queries on Encrypted Data. In *Proc. 4th Conf. Theory of Cryptography, TCC'07*, pp. 535–554.

[13] Caro, A. D. and Iovino, V. (2013) On the power of rewinding simulators in functional encryption. *Design, Codes and Cryptography*, **84**, 373–399.

[14] Caro, A. D., Jain, V. I. A., O'Neill, A., Paneth, O. and Persiano, G. (2013) On the achievability of simulation-based security for functional encryption. In *Advances in Cryptology – {CRYPTO} 2013 – 33rd Annual Cryptology Conference – Part II*, pp. 519–535.

[15] Cheung, L. and Newport, C. (2007) Provably Secure Ciphertext Policy abe. In *Proc. 14th ACM Conf. Computer and Communications Security, CCS '07*, pp. 456–465, New York, NY, USA.

[16] Garg, S., Gentry, C., Halevi, S., Raykova, M., Sahai, A. and Waters, B. (2013) Candidate Indistinguishability Obfuscation and Functional Encryption for All Circuits. In *Proc. 2013 IEEE 54th Annual Symposium on Foundations of Computer Science, FOCS '13*, pp. 40–49, Washington, DC, USA, IEEE Computer Society.

[17] Goldwasser, S., Gordon, S. D., Goyal, V., Jain, A., Katz, J., Liu, F.-H., Sahai, A., Shi, E. and Zhou, H.-S. (2014) Multi-input Functional Encryption. In *Advances in Cryptology – EUROCRYPT 2014*, pp. 578–602.

[18] Goyal, V., Jain, A., Pandey, O. and Sahai, A. (2008) Bounded Ciphertext Policy Attribute based Encryption. In *Proc. 35th Int. Colloquium on Automata, Languages and Programming, Part II, ICALP '08*, pp. 579–591, Berlin, Heidelberg, Springer-Verlag.

[19] Goyal, V., Pandey, O., Sahai, A. and Waters, B. (2006) Attribute-based Encryption for Fine-grained Access Control of Encrypted Data. In *Proc. 13th ACM Conf. Computer and Communications Security, CCS '06*, pp. 89–98. ACM.

[20] Katz, J., Sahai, A. and Waters, B. (2008) Predicate Encryption Supporting Disjunctions, Polynomial Equations, and Inner Products. In *Proc. Theory and Applications of Cryptographic Techniques 27th Annual Int. Conf. Advances in Cryptology, EUROCRYPT'08*, pp. 146–162.

[21] Lewko, A. B., Okamoto, T., Sahai, A., Takashima, K. and Waters, B. (2010) Fully Secure Functional Encryption: Attribute-Based Encryption and (Hierarchical) Inner Product Encryption. In *EUROCRYPT*, pp. 62–91.

[22] Lewko, A. B. and Waters, B. (2012) New Proof Methods for Attribute-based Encryption: Achieving Full Security through Selective Techniques. In *CRYPTO*, pp. 180–198.

[23] Nishide, T., Yoneyama, K. and Ohta, K. (2008) Attribute-based Encryption with Partially Hidden Encryptor-specified Access Structures. In *Proc. 6th Int. Conf. Applied Cryptography and Network Security, ACNS'08*, pp. 111–129.

[24] Okamoto, T. and Takashima, K. (2012) Adaptively Attribute-hiding (hierarchical) Inner Product Encryption. In *Advances in Cryptology – EUROCRYPT 2012*, Vol. 7237. Lecture Notes in Computer Science, pp. 591–608.

[25] Ostrovsky, R., Sahai, A. and Waters, B. (2007) Attribute-based Encryption with Non-monotonic Access Structures. In *Proc. 14th ACM Conf. Computer and Communications Security, CCS '07*, pp. 195–203, New York, NY, USA. ACM.

[26] Park, J. (2011) Inner product encryption under standard assumption. *Designs, Codes and Cryptography*, **58**, 235–257.

[27] Sahai, A. and Waters, B. (2005) Fuzzy Identity-Based Encryption. In *Proc. 24th Annual Int. Conf. Theory Applications of Cryptographic Techniques, EUROCRYPT'05*, pp. 457–473. Springer-Verlag.

[28] Shamir, A. (1985) Identity-Based Cryptosystems and Signature Schemes. In *Proc. CRYPTO 84 on Advances in Cryptology*, pp. 47–53, New York, NY, USA, Springer-Verlag New York, Inc.

[29] Shi, E. and Waters, B. (2008) Delegating Capabilities in Predicate Encryption Systems. In *Proc. 35th Int. Colloquium on Automata, Languages and Programming, Part II, ICALP '08*, pp. 560–578.

[30] Waters, B. (2011) Ciphertext-Policy Attribute-based Encryption: An Expressive, Efficient, and Provably Secure Realization. In *Public Key Cryptography*, pp. 53–70.

[31] Zhou, Z. and Huang, D. (2010) On Efficient Ciphertext-Policy Attribute based Encryption and Broadcast Encryption: extended abstract. In *Proc. 17th ACM Conf. Computer and Communications Security, CCS '10*, pp. 753–755, New York, NY, USA. ACM.