Hidden Ciphertext Policy Attribute-Based Encryption Under Standard Assumptions

Tran Viet Xuan Phuong, Guomin Yang, *Member, IEEE*, and Willy Susilo, *Senior Member, IEEE*

***Abstract*— We propose two new ciphertext policy attribute-based encryption (CP-ABE) schemes where the access policy is defined by AND-gate with wildcard. In the first scheme, we present a new technique that uses only one group element to represent an attribute, while the existing ABE schemes of the same type need to use three different group elements to represent an attribute for the three possible values (namely, positive, negative, and wildcard). Our new technique leads to a new CP-ABE scheme with constant ciphertext size, which, however, cannot hide the access policy used for encryption. The main contribution of this paper is to propose a new CP-ABE scheme with the property of hidden access policy by extending the technique we used in the construction of our first scheme. In particular, we show a way to bridge ABE based on AND-gate with wildcard with inner product encryption and then use the latter to achieve the goal of hidden access policy. We prove that our second scheme is secure under the standard decisional linear and decisional bilinear Diffie–Hellman assumptions.**

***Index Terms*— Attribute based encryption, hidden policy, inner product encryption, Viète's formula.**

I. Introduction

ACCESS control (i.e., authentication and authorisation) plays an important role in many information systems. Among all the existing cryptographic tools, Attribute Based Encryption (ABE) has provided an effective way for fine-grained access control. ABE, which is an extension of identity-based encryption (IBE) [4], [23], allows an access structure/policy to be embedded into the ciphertext (this is referred to as ciphertext-policy ABE, or CP-ABE) or user secret key (this is referred to as key-policy ABE, or KP-ABE). In a CP-ABE, the user's attributes used for key generation must satisfy the access policy used for encryption in order to decrypt the ciphertext, while in a KP-ABE, the user can only decrypt ciphertexts whose attributes satisfy the policy embedded in the key. We can see that access control is an inherent feature of ABE, and by using some expressive access structures, we can effectively achieve fine-grained access control. Since its introduction in the seminal work of Sahai and Waters [21], ABE has been extensively studied in recent years (e.g., [2], [3], [7], [8], [11], [12], [17], [26]). There are different ways to define an access structure/policy for ABE. The fuzzy IBE given by Sahai and Waters [21], which can be treated as the first KP-ABE, used a specific threshold access policy. Later, the Linear Secret Sharing Scheme (LSSS) realizable (or monotone) access structure has been adopted by many subsequent ABE schemes [3], [11], [12], [26]. In [7], Cheung and Newport proposed another way to define access structure using AND-Gate with wildcard. To be more precise, for each attribute in the universe, there are two possible values: positive and negative. A user's attributes are then defined by a sequence of positive and negative symbols w.r.t. each attribute in the universe (assuming that the attributes are placed in order in the universe). An access structure is also defined by a sequence of positive and negative symbols, plus a special wildcard (i.e., "don't care") symbol. Cheung and Newport showed that by using this simple access structure, which is sufficient for many applications, CP-ABE schemes can be constructed based on standard complexity assumptions. Subsequently, several ABE schemes [6], [9], [20], [28] were proposed following this specific access structure.

A. This Work

In this work, we explore new techniques for the construction of CP-ABE schemes based on the AND-gate with wildcard access structure. The existing schemes of this type need to use three different elements to represent the three possible values – positive, negative, and wildcard – of an attribute in the access structure. In this paper, we propose a new construction which uses only one element to represent one attribute. The main idea behind our construction is to use the "positions" of different symbols to perform the matching between the access policy and user attributes. Specifically, We put the indices of all the positive, negative and wildcard attributes defined in an access structure into three sets, and by using the technique of Viète's formulas [22], we allow the decryptor to remove all the wildcard positions, and perform the decryption correctly if and only if the remaining user attributes match those defined in the access structure. Our new technique leads to a new CP-ABE scheme with constant ciphertext size. Although a secure ABE can well protect the secrecy of the encrypted data against unauthorised access, it does not protect the privacy of the receivers/decryptors by default. That is, given the ciphertext, an unauthorised user may still be able to obtain some information of the data recipients. For example, a health organization wants to send a message to all the

Manuscript received April 15, 2015; revised July 6, 2015; accepted August 15, 2015. Date of publication September 2, 2015; date of current version October 30, 2015. Part of the content (Scheme 1) presented in this paper has been published in the *Proceedings of the 21st ACM Conference on Computer and Communications Security*, Arizona, USA, 2014. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Shouhuai Xu.

The authors are with the Centre for Computer and Information Security Research, School of Computing and Information Technology, University of Wollongong, Wollongong NSW 2500, Australia (e-mail: tvxp750@uowmail.edu.au; gyang@uow.edu.au; wsusilo@uow.edu.au).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIFS.2015.2475723

TABLE I

COMPARIONS AMONG CP-ABE SCHEMES

Scheme	Order Groups	Ciphertext Size	Dec. Cost	Security Models	Assumption	Access Structures	Wildcard	Hidden Policy
CN[7]	p	$\lvert\mathbb{G}_T\rvert + (n+1)\lvert\mathbb{G}\rvert$	$(n+1)\mathbf{p}$	Selective	DBDH	AND-gates on +/-	√	X
Nishide et al.[20]	p	$\lvert\mathbb{G}_T\rvert + (2mn+1)\lvert\mathbb{G}\rvert$	$(3n+1)\mathbf{p}$	Selective	DBDH + DLIN	AND-gates on multi-valued attributes	√	√
Emura et al.[9]	p	$\lvert\mathbb{G}_T\rvert + 2\lvert\mathbb{G}\rvert$	$2\mathbf{p}$	Selective	DBDH	AND-gates on multi-valued attributes	X	X
Li et al.[18]	p	$\lvert\mathbb{G}_T\rvert + 4mn\lvert\mathbb{G}\rvert$	$4n\mathbf{p}$	Selective	DBDH + DLIN	AND-gates on multi-valued attributes	√	√
ZH[28]	p	$\lvert\mathbb{G}_T\rvert + 2\lvert\mathbb{G}\rvert$	$(2n+1)\mathbf{p}$	Selective	n-BDHE	AND-gates on multi-valued attributes	√	X
Herranz et al.[13]	p	$\lvert\mathbb{G}_T\rvert + 2\lvert\mathbb{G}\rvert$	$3\mathbf{p}$	Selective	aMSE-DDH	Threshold Gates	X	X
Lai et al.[15]	pqr	$\lvert\mathbb{G}_T\rvert + (2mn+2)\lvert\mathbb{G}\rvert$	$(n+1)\mathbf{p}$	Fully	Subgroup Assumption	AND-gates on multi-valued attributes	√	√
Li et al[19]	pq	$\lvert\mathbb{G}_T\rvert + 2\lvert\mathbb{G}\rvert$	$2\mathbf{p}$	Fully	DBDH	AND-gates on multi-valued attributes	X	√
Chen et al.[6]	p	$\lvert\mathbb{G}_T\rvert + 2\lvert\mathbb{G}\rvert$	$2\mathbf{p}$	Selective	n-BDHE	AND-gates on +/-	X	X
Ge et al.[10]	p	$\lvert\mathbb{G}_T\rvert + 2\lvert\mathbb{G}\rvert$	$2\mathbf{p}$	Selective	n-BDHE	Threshold Gates	X	X
Chen et al.[5]	pqr	$\lvert\mathbb{G}_T\rvert + 2\lvert\mathbb{G}\rvert$	$2\mathbf{p}$	Fully	Subgroup Assumption	Threshold Gates	X	X
Zhang et al.[27]	p	$\lvert\mathbb{G}_T\rvert + 2\lvert\mathbb{G}\rvert$	$2\mathbf{p}$	Selective	n-BDHE	AND-gates on multi-valued attributes	X	X
Scheme 1 [25]	p	$\lvert\mathbb{G}_T\rvert + 4\lvert\mathbb{G}\rvert$	$6\mathbf{p}$	Selective	DLIN	AND-gates on +/-	√	X
Scheme 2	p	$\lvert\mathbb{G}_T\rvert + (4w+2)\lvert\mathbb{G}\rvert$	$(4w+2)\mathbf{p}$	Selective	DBDH + DLIN	AND-gates on +/-	√	√

Notations. **p**: pairing operation; n: number of attributes in an access structure/list; m: number of possible values for an attribute; w: number of wildcard in an access structure.

patients that carry certain diseases. Then the attribute universe will contain all the diseases, and an access policy will have the format "$++-**+\ldots$" where "+" ("−") indicates positive (negative) for a particular disease. If a CP-ABE cannot hide the access policy, then from the fact whether a person can decrypt the message or not, people can directly learn some sensitive information of the user. Therefore, it is also very important to hide the access policy in such applications. However, most of the existing ABE schemes based on AND-Gate with wildcard cannot achieve this property. To address this problem, we further study the problem of hiding the access policy for CP-ABE based on AND-Gate with wildcard. As the main contribution of this work, we extend the technique we have used in the first construction to bridge ABE based on AND-Gate with wildcard with Inner Product Encryption (IPE) [1], [14], [24]. Specifically, we present a way to convert an access policy containing positive, negative, and wildcard symbols into a vector $\vec{X}$ which is used for encryption, and the user's attributes containing positive and negative symbols into another vector $\vec{Y}$ which is used in key generation, and then apply the technique of IPE to do the encryption. Again, we use the positions of different symbols and the Viète's formulas [22] to perform the conversion. The details are provided in Section IV-A. In Table I, we give a comparison among CP-ABE schemes that are based on the AND-Gate access structure or have constant-size ciphertext. We use **p** to denote the pairing operation, n the number of attributes in an access structure or attribute list, m the number of all possible values for each attribute, and w the number of wildcard in an access structure. We can see that among all the schemes that can support wildcard and provide hidden access policy, our Scheme 2 gives the best performance since the ciphertext size and the decryption cost depend only on the number of wildcard in an access structure.

B. Paper Organization

We present the preliminaries and security definitions in Section II, which is followed by our first scheme in Section III. We then present the second scheme with security proof in Section IV. The paper is concluded in Section V.

II. PRELIMINARIES

A. Bilinear Map and Its Related Assumptions

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

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

Definition 1: The Decisional Bilinear Diffie-Hellman (DBDH) problem in $\mathbb{G}$ is defined as follows: given a tuple $(g, g^a, g^b, g^c, T) \in \mathbb{G}^4 \times \mathbb{G}_T$, decide whether $T = e(g, g)^{abc}$ or $T = e(g, g)^r$ where a, b, c, r are randomly selected from $\mathbb{Z}_p$. An algorithm A has advantage ϵ in solving the DBDH problem in $\mathbb{G}$ if

$$\begin{aligned}\mathbf{Adv}_A^{\mathrm{DBDH}}(k) = {} & \Pr[A(1^k, g, g^a, g^b, g^c, Z) = 1 | Z = e(g, g)^{abc}] \\ & - \Pr[A(1^k, g, g^a, g^b, g^c, Z) = 1 | Z = g^r] \leq \epsilon.\end{aligned}$$

We say that the DBDH assumptions holds in $\mathbb{G}$ if ϵ is negligible for any PPT algorithm A.

Definition 2: The Decisional Linear (DLIN) problem in $\mathbb{G}$ defined as follows: given a tuple $(g, g^a, g^b, g^{ac}, g^d, Z) \in \mathbb{G}^5 \times \mathbb{G}_T$, decide whether $T = g^{b(c+d)}$ or Z in random in $\mathbb{G}$. An algorithm A has advantage ϵ in solving the DLIN problem in $\mathbb{G}$ if

$$\begin{aligned}&\mathbf{Adv}_A^{\mathrm{DLIN}}(k) \\ &\quad = \Pr[A(1^k, g, g^a, g^b, g^{ac}, g^d, Z) = 1 | Z = g^{b(c+d)}] \\ &\qquad - \Pr[A(1^k, g, g^a, g^b, g^{ac}, g^d, Z) = 1 | Z = g^r] \leq \epsilon\end{aligned}$$

where $a, b, c, d, r \in_R \mathbb{Z}_p$. We say that the DLIN assumptions holds in $\mathbb{G}$ if ϵ is negligible for any PPT algorithm A.

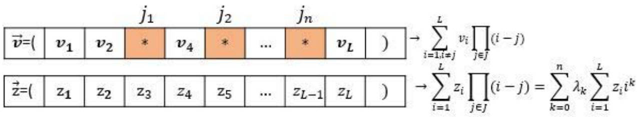

j_1
j_2
j_n
$\vec{v}$=(
v_1
v_2
*
v_4
*
...
*
v_L
)
$\rightarrow \sum_{i=1, i\neq j}^{L} v_i \prod_{j\in J}(i-j)$
$\vec{z}$=(
z_1
z_2
z_3
z_4
z_5
...
z_{L-1}
z_L
)
$\rightarrow \sum_{i=1}^{L} z_i \prod_{j\in J}(i-j) = \sum_{k=0}^{n} \lambda_k \sum_{i=1}^{L} z_i i^k$

Fig. 1. Expressing two vectors $\vec{v}$ and $\vec{z}$.

B. The Viète's Formulas

Consider two vectors $\overrightarrow{v} = (v_1, v_2, \ldots, v_L)$ and $\overrightarrow{z} = (z_1, z_2, \ldots, z_L)$ (Fig. 1). Vector $\overrightarrow{v}$ contains both alphabets and wildcards, and vector $\overrightarrow{z}$ only contains alphabets. Let $J = \{j_1, \ldots, j_n\} \subset \{1, \ldots, L\}$ denote the positions of the wildcards in vector $\overrightarrow{v}$.

TABLE II

LIST OF ATTRIBUTES AND POLICIES

Attributes	Att_1	Att_2	Att_3	Att_4
Description	CS	EE	Faculty	Student
Alice	+	−	−	+
Bob	−	+	+	−
Carol	+	+	+	−
W_1	+	−	−	+
W_2	+	−	*	*

Let $\prod_{j\in J}(i-j) = \sum_{k=0}^{n}\lambda_k i^k$, where λ_k are the coefficients dependent on J, then we have

$$\sum_{i=1,i\notin J}^{L} v_i \prod_{j\in J}(i-j) = \sum_{k=0}^{n}\lambda_k \sum_{i=1}^{L} z_i i^k. \qquad (1)$$

if $v_i = z_i \vee v_i = *$ for $i = 1\dots L$.

To hide the computation, we can choose a random group element H_i and put v_i, z_i as the exponents of H_i. Then (1) becomes

$$\prod_{i=1,i\notin J}^{L} H_i^{v_i \prod_{j\in J}(i-j)} = \prod_{k=0}^{n}(\prod_{i=1}^{L} H_i^{z_i i^k})^{\lambda_k}.$$

Using the Viète's formulas [22] we can construct the coefficient λ_k in (1) by:

$$\lambda_{n-k} = (-1)^k \sum_{1\le i_1<i_2<\ldots<i_k\le n} j_{i_1} j_{i_2}\ldots j_{i_k},\ 0\le k\le n,$$

where $n = |J|$.

Take as an example, if we have $J = \{j_1, j_2, j_3\}$, then the polynomial is $(x-j_1)(x-j_2)(x-j_3)$, and we have $\lambda_3 = 1$, $\lambda_2 = -(j_1+j_2+j_3)$, $\lambda_1 = (j_1j_2+j_1j_3+j_2j_3)$, $\lambda_0 = -j_1j_2j_3$.

C. Access Structure

Let $U = \{Att_1, Att_2, \ldots, Att_L\}$ be the universe of the attributes in the system. Each Att_i is represented by a unique value A_i. When a user joins the system, the user is tagged with an attribute list defined as $S = \{S_1, S_2, \ldots, S_L\}$ where each symbol S_i has two possible values: '+' and '−'. Let $W = \{S'_1, S'_2, \ldots, S'_L\}$ denote an AND-gate with wildcard access policy where each symbol S'_i has three possible values: '+', '−', and '$*$'. The wildcard '$*$' means "don't care" (i.e., both positive and negative attributes are accepted). We use the notation $S \models W$ to denote that the attribute list S of a user satisfies W.

For example, suppose $U = \{Att_1 = \text{CS}, Att_2 = \text{EE}, Att_3 = \text{Faculty}, Att_4 = \text{Student}\}$. Alice is a student in the CS department; Bob is a faculty in the EE department; Carol is a faculty holding a joint position in the EE and CS departments. Their attribute lists are illustrated in Table II. The access structure W_1 can be satisfied by all the CS students without being in the EE department, while W_2 can be satisfied by all CS students and faculties excluding those in EE.

D. CP-ABE Definition

A ciphertext-policy attribute based encryption scheme consists of four algorithms: Setup, Encrypt, KeyGen, and Decrypt.

- *Setup(λ, U):* The setup algorithm takes security parameters and attribute universe description as input. It outputs the public parameters PK and a master key MSK.
- *Encrypt(PK, M, W):* The encryption algorithm takes as input the public parameters PK, a message M, and access structure W over the universe of attributes, and outputs a ciphertext CT.
- *Key Generation(MSK, L):* The key generation algorithm takes as input the master key MSK and a set of attributes $L \subset U$, and outputs a private key SK.
- *Decrypt(PK, CT, SK):* The decryption algorithm takes as input the public parameters PK, a ciphertext CT, and a private key SK, and outputs a message M or a special symbol '$\perp$'.

E. Security Definition for CP-ABE With Hidden Access Policy

We define the Selective IND-CPA security for CP-ABE with hidden access policy via the following game.

- *Init:* The adversary commits to the challenge access policies W_0, W_1.
- *Setup:* The challenger runs the Setup algorithm and gives PK to the adversary.
- *Phase 1:* The adversary submits the attribute list L for a KeyGen query. If $(L \models W_0 \wedge L \models W_1)$ or $(L \not\models W_0 \wedge L \not\models W_1)$, the challenger gives the adversary the secret key SK_L. The adversary can repeat this polynomially many times.
- *Challenge:* The adversary submits messages M_0, M_1 to the challenger. If the adversary obtained the SK_L whose associated attribute list L satisfies both W_0 and W_1 in Phase 1, then it is required that $M_0 = M_1$. The challenger flips a random coin β and passes the ciphertext $Encrypt(PK, M_\beta, W_\beta)$ to the adversary.
- *Phase 2:* Phase 1 is repeated. If $M_0 \neq M_1$, the adversary cannot submit L such that $L \models W_0 \wedge L \models W_1$.
- *Guess:* The adversary outputs a guess β' of β.

We say a CP-ABE scheme with hidden access policy is secure if for any probabilistic polynomial-time adversary $\mathcal{A}$,

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

is negligible in the security parameter k.

Full Security: In the above selective security model, the adversary is required to commit the challenge policy before seeing the system parameters. In the full security model, the adversary can choose the challenge policy in the Challenge phase, which makes the model stronger. However, similar to many other CP-ABE schemes given in Table I, we cannot directly prove the security of our schemes in the full security model. We should note that there are transformations from the selective security to full security [15], and we can apply the same transformation to our schemes presented in this paper. However, the transformed schemes will be based on composite order pairing groups, and hence less efficient.

III. OUR FIRST CONSTRUCTION

In this section, we present our first scheme based on the AND-Gate with wildcard access policy. Below is the main idea of our construction.

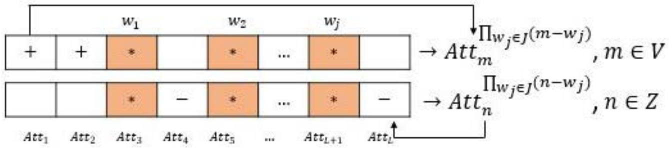

w_1
w_2
w_j
+
+
*
*
…
*
*
−
*
…
*
−
$\rightarrow Att_m^{\prod_{w_j \in J}(m-w_j)}, m \in V$
$\rightarrow Att_n^{\prod_{w_j \in J}(n-w_j)}, n \in Z$
Att_1
Att_2
Att_3
Att_4
Att_5
…
Att_{L+1}
Att_L

Fig. 2. Access Policy.

We represent each attribute in the universe by an element A_i. Given an access structure W, we first define three sets J, V, and Z where J contains the positions of all the wildcard positions, and V and Z contain the positions of all the positive and negative attributes, respectively. We then represent each positive/negative attribute in an access structure as shown in Fig. 2.

The set J is attached to the ciphertext and sent to the decryptor. In the decryption process, based on J, the decryptor can reconstruct the coefficients λ_{w_j}, and generate

$$\prod_{j \in J} (A_i)^{i^j \lambda_{w_j}} = (A_i)^{\prod_{w_j \in J} (i - w_j)}$$

according to the Viète's formulas, for each positive or negative attribute Att_i associated with the secret key. In this way, all the wildcard positions will take no effect during decryption. Below are the details of our construction.

Setup(1^k): Let N_1, N_2, N_3 be three upper bounds defined as $N_1 \leq L$: the maximum number of wildcard in an access structure; $N_2 \leq L$: the maximum number of positive attribute in an attribute set S; $N_3 \leq L$: the maximum number of negative attribute in an attribute set S.

The setup algorithm first generates bilinear groups $\mathbb{G}$, $\mathbb{G}_T$ with order p, and selects two random generators $V_0, V_1, g \in \mathbb{G}$. Then randomly choose $\alpha, \beta_1, \beta_2, a_1, \ldots, a_L \in_R \mathbb{Z}_p$, and set $\Omega_1 = e(g, V_0)^{\alpha\beta_1} e(g, V_1)^{\alpha\beta_1}$, $\Omega_2 = e(g, V_0)^{\alpha\beta_2} e(g, V_1)^{\alpha\beta_2}$. Let $A_i = g^{a_i}$ for $1 = 1, \ldots, L$.

The Public Key and Master Secret Key are defined as:

$$PK = (e, g, \Omega_1, \Omega_2, g^\alpha, V_0, V_1, A_1, \ldots, A_L),$$
$$MSK = (\alpha, \beta_1, \beta_2, a_1, \ldots, a_L).$$

Encrypt(W, M, PK): Suppose that the access structure W contains: $n_1 \leq N_1$ wildcards which occur at positions $J = \{w_1, \ldots, w_{n_1}\}$; $n_2 \leq N_2$ positive attributes which occur at positions $V = \{v_1, \ldots, v_{n_2}\}$; $n_3 \leq N_3$ negative attributes which occur at positions $Z = \{z_1, \ldots, z_{n_3}\}$. Compute for the wildcard positions $\{w_j\}$ $(j = 0, 1, 2, \cdots, n_1)$ $\{\lambda_{w_j}\}$ and set $t_w = \sum_{j=0}^{n_1} \lambda_{w_j}$. The encryption algorithm then chooses $r_1, r_2 \in_R \mathbb{Z}_p$, and creates the ciphertext as:

$$C_0 = M\Omega_1^{r_1}\Omega_2^{r_2}, \quad C_1 = g^{\frac{\alpha r_1}{t_w}}, \quad C_2 = g^{\frac{r_2}{t_w}},$$
$$C_3 = (V_0 \prod_{i \in V} (A_i^{\frac{\prod_{j=0}^{n_1} (i - w_j)}{t_w}})^{r_1 + r_2},$$
$$C_4 = (V_1 \prod_{i \in Z} (A_i^{\frac{\prod_{j=0}^{n_1} (i - w_j)}{t_w}})^{r_1 + r_2},$$

The ciphertext is set as:

$$CT = (C_0, C_1, C_2, C_3, J = \{w_1, w_2, \ldots, w_{n_1}\}).$$

KeyGen(MSK, S): Suppose that a user joins the system with the attribute list S, which contains: $n_2' \leq N_2$ positive attributes which occur at positions $V' = \{v_1', \ldots, v_{n_2'}'\}$; $n_3' \leq N_3$ negative attributes which occur at positions $Z' = \{z_1', \ldots, z_{n_3'}'\}$.

By means of the Viète's formulas, for all the positive positions $\{v_k'\}$ $(k = 0, 1, 2, \cdots, n_2')$, calculate $\{\lambda_{v_k'}\}$ and set $t_v' = \sum_{k=0}^{n_2} \lambda_{v_k'}$; and for all the negative positions $\{z_\tau'\}$ $(\tau = 0, 1, 2, \cdots, n_3')$, calculate $\{\lambda_{z_\tau'}\}$ and set $t_z' = \sum_{\tau=0}^{n_3'} \lambda_{z_\tau'}$. The algorithm then chooses $s \in_R \mathbb{Z}_p$ and computes $s_1 = \beta_1 + s, s_2 = \beta_2 + s$ and creates the secret key as:

$$\begin{aligned}
L_1 &= g^{\frac{\alpha s}{t_v'}}, L_2 = g^{\frac{\alpha s}{t_z'}}, \\
K_1 &= \{K_{1,0}, K_{1,1}, \ldots, K_{1,N_1}\} \\
&= \{V_0^{s_1} \prod_{i \in V'} g^{s a_i}, V_0^{s_1} \prod_{i \in V'} g^{s a_i i}, \ldots, V_0^{s_1} \prod_{i \in V'} g^{s a_i i^{N_1}}\}, \\
K_1' &= \{K_{1,0}', K_{1,1}', \ldots, K_{1,N_1}'\} \\
&= \{V_0^{\alpha s_2} \prod_{i \in V'} g^{s \alpha a_i}, V_0^{\alpha s_2} \prod_{i \in V'} g^{s \alpha a_i i}, \ldots, V_0^{\alpha s_2} \prod_{i \in V'} g^{s \alpha a_i i^{N_1}}\}. \\
K_2 &= \{K_{2,0}, K_{2,1}, \ldots, K_{2,N_1}\} \\
&= \{V_1^{s_1} \prod_{i \in Z'} g^{s a_i}, V_1^{s_1} \prod_{i \in Z'} g^{s a_i i}, \ldots, V_1^{s_1} \prod_{i \in Z'} g^{s a_i i^{N_1}}\}, \\
K_2' &= \{K_{2,0}', K_{2,1}', \ldots, K_{2,N_1}'\} \\
&= \{V_1^{\alpha s_2} \prod_{i \in Z'} g^{s \alpha a_i}, V_1^{\alpha s_2} \prod_{i \in Z'} g^{s \alpha a_i i}, \ldots, V_1^{\alpha s_2} \prod_{i \in Z'} g^{s \alpha a_i i^{N_1}}\}.
\end{aligned}$$

The user secret key is set as:

$$SK = (L_1, L_2, K_1, K_1', K_2, K_2').$$

Decrypt(CT, SK): The algorithm first identifies the wildcard positions in $J = \{w_1, \ldots, w_{n_1}\}$ and computes $\{\lambda_{w_j}\}$. Then we have the equation shown at the bottom of this page, and M can be recovered by $\Omega_1^{-r_1}\Omega_2^{-r_2} \cdot C_0$.

$$\begin{aligned}
&\frac{e(L_1, C_3)^{t_v'} \cdot e(L_2, C_4)^{t_z'}}{e(\prod_{j=0}^{n_1} K_{1,j}^{\lambda_{w_j}}, C_1) \cdot e(\prod_{j=0}^{n_1} (K_{1,j}')^{\lambda_{w_j}}, C_2) \cdot e(\prod_{j=0}^{n_1} K_{2,j}^{\lambda_{w_j}}, C_1) \cdot e(\prod_{j=0}^{n_1} (K_{2,j}')^{\lambda_{w_j}}, C_2)} \\
&\quad = e(g, V_0)^{-\alpha\beta_1 r_1} e(g, V_0)^{-\alpha\beta_2 r_2} e(g, V_1)^{-\alpha\beta_1 r_1} e(g, V_1)^{-\alpha\beta_2 r_2} \\
&\quad = \Omega_1^{-r_1}\Omega_2^{-r_2}
\end{aligned}$$

IV. CP-ABE With Hidden Access Policy

Although the CP-ABE scheme presented in the previous section can achieve constant ciphertext size, it cannot hide the access policy since the wildcard positions need to be included in the ciphertext. In this section, we extend the technique used in our first construction to build another CPA-ABE which can hide the access policy. One way to achieve the attribute hiding property is to apply the Inner Product Encryption technique in the construction of CP-ABE. Such an approach has been used in previous works on policy hiding CP-ABE [5], [15], [16]. However, since our CP-ABE scheme is based on the Viète's formula, we cannot directly use the previous approach. In this paper, we propose a new transformation technique which can deal with the Viète's formula.

A. Our Idea

Our main idea is to convert the access policy and user attributes into two vectors, and then apply the technique of Inner Product Encryption to hide the access policy. Similar to the first scheme, we separate the positive, negative, and wildcard symbols in an access structure into three sets: V, Z, and J. Based on the set J, by applying the Viète's formulas, we can construct a polynomial $\sum_{k=0}^{n} a_k i^k$ with coefficients $(a_0, a_1, \ldots, a_n)$.

Then we combine the set of positive positions V as:

$$\Pi_V = +\sum_{i \in V} \prod_{w_j \in J} (i - w_j)$$

and the set of negative positions Z as:

$$\Pi_Z = -\sum_{i \in Z} \prod_{w_j \in J} (i - w_j).$$

We then produce a vector

$$\overrightarrow{v} = (a_0, a_1, \ldots, a_n, 0_{n+1}, \ldots, 0_{N_1}, \Pi_V, \Pi_Z)$$

which will be used for encryption.

In user key generation, we also separate the positive and negative attributes into two sets and construct two vectors

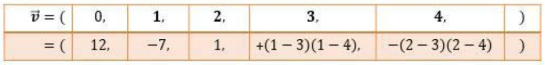
$$\vec{x_{V'}} = (v'_0, v'_1, v'_2, \ldots, v'_{N_1}, 1, 0),$$
$$\vec{x_{Z'}} = (z'_0, z'_1, z'_2, \ldots, z'_{N_1}, 0, 1),$$

in which:

$$v'_k = -\sum_{i \in V'} i^k, k = 0, \ldots, N_1,$$
$$z'_k = +\sum_{i \in Z'} i^k, k = 0, \ldots, N_1.$$

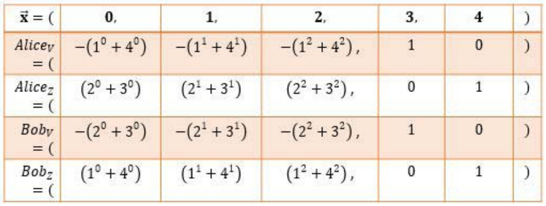
Notice that we assume there are at most N_1 wildcard positions in an access policy. The decryption will be based on the inner products of $(\vec{v}, \vec{x_{V'}})$ and $(\vec{v}, \vec{x_{Z'}})$, which should both return 0 in order to have a successful decryption.

Below we give a simple example based on Table II to illustrate our idea. Let $L = 4$, $N_1 = 2$ and $W_2 = (+, -, *, *)$ be the access policy. Then we create three sets for wildcard positions $J = \{3, 4\}$, positive positions $V = \{1\}$, and negative positions $Z = \{2\}$. Based on Viète's formulas, we can calculate



$\vec{v}$ = (	0,	1,	2,	3,	4,	)
= (	12,	−7,	1,	+(1 − 3)(1 − 4),	−(2 − 3)(2 − 4)	)

Fig. 3. The vector $\vec{v}$ for access policy W_2.



$\vec{x}$ = (	0,	1,	2,	3,	4	)
$Alice_V$ = (	$-(1^0 + 4^0)$	$-(1^1 + 4^1)$	$-(1^2 + 4^2)$,	1	0	)
$Alice_Z$ = (	$(2^0 + 3^0)$	$(2^1 + 3^1)$	$(2^2 + 3^2)$,	0	1	)
Bob_V = (	$-(2^0 + 3^0)$	$-(2^1 + 3^1)$	$-(2^2 + 3^2)$,	1	0	)
Bob_Z = (	$(1^0 + 4^0)$	$(1^1 + 4^1)$	$(1^2 + 4^2)$,	0	1	)

Fig. 4. The vector $\vec{z}$ for Alice and Bob.

$$a_2 = 1; \quad a_1 = -7, \quad a_0 = 12$$

and obtain the vector $\vec{v}$ for the access policy (Fig. 3) and the vectors for Alice and Bob as shown in Fig 4.

If we calculate the inner product of $\vec{v}$ and the two vectors of Alice, the product will return 0, i.e., Alice's attributes satisfy the access policy W_2. On the other hand, the inner product of $(\vec{v}, \vec{Bob_v}) = 8$ and $(\vec{v}, \vec{Bob_z}) = 4$, which means Bob's attributes cannot satisfy W_2.

B. Our Second Construction

Setup(1^k): Assume that we have L attributes in the universe, and each attribute has two possible values: positive and negative. In addition, we also consider wildcard (meaning "don't care") in access structures. Let N_1, N_2, N_3 be three upper bounds defined as:

$N_1 \leq L$: the maximum number of wildcard in an access structure;

$N_2 \leq L$: the maximum number of positive attribute in an attribute set S;

$N_3 \leq L$: the maximum number of negative attribute in an attribute set S.

The setup algorithm first randomly generates $(g, \mathbb{G}, \mathbb{G}_T, p, e)$ and set $n = N_1 + 3$. It then chooses randomly γ_1, γ_2, θ_1, θ_2, $\{u_{1,i}\}_{i=1}^n$, t_1, $\{t_{1,i}\}_{i=1}^n$, $\{t_{2,i}\}_{i=1}^n$, $\{w_{1,i}\}_{i=1}^n$, $\{z_{1,i}\}_{i=1}^n$, $\{z_{2,i}\}_{i=1}^n$ in $\mathbb{Z}_p$ and g_2 in $\mathbb{G}$. Then it selects a random $\Delta \in \mathbb{Z}_p$ and obtains $\{u_{2,i}\}_{i=1}^n, \{w_{2,i}\}_{i=1}^n, w_2, u_2$ under the condition:

$$\Delta = \gamma_1 u_{2,i} - \gamma_2 u_{1,i} \;\; \Delta = \theta_1 w_{2,i} - \theta_2 w_{1,i}.$$

For i from 1 to n, it creates:

$$U_{1,i} = g^{u_{1,i}}, \quad U_{2,i} = g^{u_{2,i}}, \quad W_{1,i} = g^{w_{1,i}}, \quad W_{2,i} = g^{w_{2,i}},$$
$$T_{1,i} = g^{t_{1,i}}, \quad T_{2,i} = g^{t_{2,i}}, \quad Z_{1,i} = g^{z_{1,i}},$$
$$V_1 = g^{\gamma_1}, \quad V_2 = g^{\gamma_2}, \quad X_1 = g^{\theta_1}, \quad V_2 = g^{\theta_2}.$$

Next it sets $g_1 = g^{\Delta}, Y = e(g, g_2)$, and the public key PK and master key MSK as

$$PK = (g, \mathbb{G}, \mathbb{G}_T, p, e, g_1, Y, \{U_{1,i}, U_{2,i}, T_{1,i}, T_{2.i}, W_{1.i}, W_{2,i}, Z_{1,i}, Z_{2,i}\}_{i=1}^n, \{V_i, X_i\}_{i=1}^2)$$
$$MSK = (g_2, \{u_{1,i}, u_{2,i}, t_{1,i}, t_{2,i}, w_{1,i}, w_{2,i}, z_{1,i}, z_{2,i}\}_{i=1}^n, \{v_i, x_i\}_{i=1}^2).$$

Encrypt(W, M, PK): Suppose that the access structure W contains: $n_1 \leq N_1$ wildcards which occur at positions $J = \{w_1, \ldots, w_{n_1}\}$; $n_2 \leq N_2$ positive attributes which occur at positions $V = \{v_1, \ldots, v_{n_2}\}$; $n_3 \leq N_3$ negative attributes which occur at positions $Z = \{z_1, \ldots, z_{n_3}\}$. Based on Viète's formulas, compute for the wildcard positions $\{w_j\}$ $(j = 0, 1, 2, \cdots, n_1)$

$$\begin{aligned} a_{n_1} &= 1 \\ a_{n_1-1} &= -(w_1 + w_2 + \ldots + w_{n_1}) \\ a_{n_1-2} &= (w_1 w_2 + w_1 w_3 + \ldots + w_{n_1-1} w_{n_1}) \\ &\cdots \\ a_0 &= -(w_1 \cdot w_2 \cdots w_{n_1}) \end{aligned}$$

Next it computes:

$$\begin{aligned} \Pi_V &= +\sum_{i \in V} \prod_{w_j \in J} (i - w_j) \\ \Pi_Z &= -\sum_{i \in Z} \prod_{w_j \in J} (i - w_j) \end{aligned}$$

It creates a vector $\vec{v} = (v_1, v_2, \ldots, v_n)$ as:

$$\vec{v} = (a_0, a_1, \ldots, a_{n_1}, 0_{n_1+1}, \ldots, 0_{N_1}, \Pi_V, \Pi_Z).$$

The encryption algorithm chooses random $s_1, s_2, \alpha, \beta \in \mathbb{Z}_p$ and creates the ciphertext as follows:

$$\begin{aligned} C_m = M \cdot Y^{s_2}, \quad C_A &= g^{s_2}, \quad C_B = g_1^{s_1}, \\ \{C_{1,i}, C_{2,i}\} &= \{U_{1,i}^{s_1} T_{1,i}^{s_2} V_1^{v_i \alpha}, U_{2,i}^{s_1} T_{2,i}^{s_2} V_2^{v_i \alpha}\}, \\ \{C_{3,i}, C_{4,i}\} &= \{W_{1,i}^{s_1} Z_{1,i}^{s_2} X_1^{v_i \beta}, W_{2,i}^{s_1} Z_{2,i}^{s_2} X_2^{v_i \beta}\}, \end{aligned}$$

Then ciphertext CT is set as:

$$CT = (C_m, C_A, C_B, \{C_{1,i}, C_{2,i}, C_{3,i}, C_{4,i}\}_{i=1}^n).$$

KeyGen(MSK, S): Suppose that a user joins the system with the attribute list S, which contains: $n'_2 \leq N_2$ positive attributes which occur at positions $V' = \{v'_1, \ldots, v'_{n'_2}\}$; $n'_3 \leq N_3$ negative attributes which occur at positions $Z' = \{z'_1, \ldots, z'_{n'_3}\}$. By means of the Viète's formulas, for all the positive positions $\{v'_k\}$ $(k = 0, 1, 2, \cdots, n'_2)$, for all the negative positions $\{z'_\tau\}$ $(\tau = 0, 1, 2, \cdots, n'_3)$, it sets:

$$\begin{aligned} v'_k &= -\sum_{i \in V'} i^k, k = 0, \ldots, N_1 \\ z'_k &= +\sum_{i \in Z'} i^k, k = 0, \ldots, N_1 \end{aligned}$$

It creates vectors $\overrightarrow{x_V}$ and $\overrightarrow{x_Z}$ as:

$$\begin{aligned} \overrightarrow{x_V} &= (v'_0, v'_1, \ldots, v'_{N_1}, 1, 0). \\ \overrightarrow{x_Z} &= (z'_0, z'_1, +\ldots, z'_{N_1}, 0, 1). \end{aligned}$$

The key generation algorithm chooses randomly $r_{i,1}, r_{i,2}$ for $i = 1$ to n, and $f_1, f_2, r_1, r_2 \in \mathbb{Z}_p$, and then creates the secret key as follows:

$$\begin{aligned} \{K_{1,i}, K_{2,i}\} &= \{g^{-\gamma_2 r_{1,i}} g^{f_1 x_{V_i} u_{2,i}}, g^{\gamma_1 r_{1,i}} g^{-f_1 x_{V_i} u_{1,i}}\}, \\ \{K_{3,i}, K_{4,i}\} &= \{g^{-\theta_2 r_{2,i}} g^{f_2 x_{Z_i} w_{2,i}}, g^{\theta_1 r_{2,i}} g^{-f_2 x_{Z_i} w_{1,i}}\}, \\ K_A &= g_2 \cdot \prod_{i=1}^n K_{1,i}^{-t_{1,i}} K_{2,i}^{-t_{2,i}} K_{3,i}^{-z_{1,i}} K_{4,i}^{-z_{2,i}}, \\ K_B &= \prod_{i=1}^n g^{-(r_{1,i}+r_{2,i})}. \end{aligned}$$

The secret key is set as:

$$SK = (K_A, K_B, \{K_{1,i}, K_{2,i}, K_{3,i}, K_{4,i}\}_{i=1}^n).$$

Decrypt(SK, CT): The decryption algorithm returns

$$\frac{C_m}{e(C_A, K_A) \cdot e(C_B, K_B) \prod_{j=1}^4 \prod_{i=1}^n e(C_{j,i}, K_{j,i})}.$$

Correctness:

$$\begin{aligned} &e(C_{1,i}, K_{1,i}) \\ &\quad = e(U_{1,i}^{s_1} T_{1,i}^{s_2} V_1^{v_i \alpha}, g^{-\gamma_2 r_{1,i}} g^{f_1 x_{V_i} u_{2,i}}) \\ &\quad = e(g,g)^{r_{1,i} s_1 (-u_{1,i} \gamma_2)} \cdot e(g,g)^{-r_{1,i} v_i \alpha \gamma_1 \gamma_2} \cdot e(g, K_{1,i})^{t_{1,i} s_2} \\ &\qquad \cdot e(g,g)^{f_1 x_{V_i} u_{1,i} u_{2,i} s_1} \cdot e(g,g)^{f_1 v_i x_{V_i} \alpha \gamma_1 u_{2,i}}. \\ &e(C_{2,i}, K_{2,i}) \\ &\quad = e(U_{2,i}^{s_1} T_{2,i}^{s_2} V_2^{v_i \alpha}, g^{\gamma_1 r_{1,i}} g^{-f_1 x_{V_i} u_{1,i}}) \\ &\quad = e(g,g)^{r_{1,i} s_1 u_{2,i} \gamma_1} \cdot e(g,g)^{r_{1,i} v_i \alpha \gamma_1 \gamma_2} \cdot e(g, K_{2,i})^{t_{2,i} s_2} \\ &\qquad \cdot e(g,g)^{-f_1 x_{V_i} u_{1,i} u_{2,i} s_1} \cdot e(g,g)^{-f_1 v_i x_{V_i} \alpha \gamma_2 u_{1,i}}. \\ &\prod_{j=1}^2 \prod_{i=1}^n e(C_{j,i}, K_{j,i}) \\ &\quad = \prod_{i=1}^n e(g,g)^{r_{1,i} s_1 \Delta} \cdot e(g,g)^{f_1 v_i x_{V_i} \alpha \Delta} \\ &\qquad \cdot e(g, K_{1,i})^{t_{1,i} s_2} e(g, K_{2,i})^{t_{2,i} s_2}. \\ &\prod_{j=3}^4 \prod_{i=1}^n e(C_{j,i}, K_{j,i}) \\ &\quad = \prod_{i=1}^n e(g,g)^{r_{2,i} s_1 \Delta} \cdot e(g,g)^{f_2 v_i x_{Z_i} \beta \Delta} \\ &\qquad \cdot e(g, K_{3,i})^{z_{1,i} s_2} e(g, K_{4,i})^{z_{2,i} s_2}. \end{aligned}$$

Then we have:

$$\begin{aligned} &\prod_{j=1}^4 \prod_{i=1}^n e(C_{j,i}, K_{j,i}) \\ &\quad = e(g,g)^{(\sum v_i x_{V_i}) f_1 \alpha \Delta} e(g,g)^{(\sum v_i x_{Z_i}) f_2 \beta \Delta} \\ &\qquad \times \prod_{i=1}^n e(g, K_{1,i})^{t_{1,i} s_2} e(g, K_{2,i})^{t_{2,i} s_2} e(g, K_{3,i})^{z_{1,i} s_2} \\ &\qquad \times e(g, K_{4,i})^{z_{2,i} s_2} e(g,g)^{r_{1,i} s_1 \Delta} e(g,g)^{r_{2,i} s_1 \Delta}. \end{aligned}$$

Also, since

$$\begin{aligned} e(C_A, K_A) &= e(g^{s_2}, g_2 \cdot \prod_{i=1}^n K_{1,i}^{-t_{1,i}} K_{2,i}^{-t_{2,i}} K_{3,i}^{-z_{1,i}} K_{4,i}^{-z_{2,i}}) \\ e(C_B, K_B) &= e(g^{s_1 \Delta}, \prod_{i=1}^n g^{-(r_{1,i}+r_{2,i})}) \end{aligned}$$

we have

$$\begin{aligned} &\frac{C_m}{e(C_A, K_A) \cdot e(C_B, K_B) \cdot \prod_{j=1}^4 \prod_{i=1}^n e(C_{j,i}, K_{j,i})} \\ &\qquad = \frac{M}{e(g,g)^{((\sum v_i x_{V_i}) f_1 \alpha \Delta) + ((\sum v_i x_{Z_i}) f_2 \beta \Delta)}}. \end{aligned}$$

Therefore, the message M will be returned iff $(\vec{v}, \overrightarrow{x_V}) = 0$ and $(\vec{v}, \overrightarrow{x_Z}) = 0$, meaning the attributes list in user key SK satisfies the access policy in the ciphertext CT.

C. Security Proof for Our Second Construction

Theorem 1: Assume the Decision Bilinear Diffie-Hellman assumption and Decisional Linear Assumption hold in group $\mathbb{G}$, then our CP-ABE scheme is selective IND-CPA secure and policy hiding.

Since our scheme actually uses the vector corresponding to an access policy to do the encryption. In order to prove that our scheme is policy hiding, we only need to prove that the adversary cannot tell which vector, among the two vectors $\vec{v}$ and $\vec{x}$ corresponding to W_0 and W_1 respectively, has been used to generate the ciphertext. In our proof we will consider two cases $M_0 = M_1$ and $M_0 \neq M_1$.

In the case $M_0 = M_1$, we only consider the following game sequence from **Game**$_1$ to **Game**$_5$. In this case, we only prove the property of attribute hiding. For the other case $M_0 \neq M_1$, we need to consider the whole proof from **Game**$_0$ to **Game**$_6$. Below we first give a high level description of each game. In each game, we separate the vector used to generate $(C_A, C_B, C_{1,i}, C_{2,i})$ from the vector for $(C_A, C_B, C_{3,i}, C_{4,i})$. However, the same vector is used for both parts in **Game**$_0$ and **Game**$_6$.

Game$_0$: The challenge ciphertext $\mathbf{CT}_0$ is generated under $(\vec{v}, \vec{v})$ and M_0. The ciphertext $\mathbf{CT}_0$ is computed as follows:

$$(M_0 \cdot Y^{-s_2}, g^{s_2}, g_1^{s_1}, \{U_{1,i}^{s_1} T_{1,i}^{s_2} V_1^{v_i \alpha}, U_{2,i}^{s_1} T_{2,i}^{s_2} V_2^{v_i \alpha}\}_{i=1}^{n},$$
$$\{W_{1,i}^{s_1} Z_{1,i}^{s_2} X_1^{v_i \beta}, W_{2,i}^{s_1} Z_{2,i}^{s_2} X_2^{v_i \beta}\}_{i=1}^{n})$$

Game$_1$: The challenge ciphertext $\mathbf{CT}_1$ is generated under $(\vec{v}, \vec{v})$ and a random message $R \in \mathbb{G}_T$. The ciphertext $\mathbf{CT}_1$ is computed as follows:

$$(R', g^{s_2}, g_1^{s_1}, \{U_{1,i}^{s_1} T_{1,i}^{s_2} V_1^{v_i \alpha}, U_{2,i}^{s_1} T_{2,i}^{s_2} V_2^{v_i \alpha}\}_{i=1}^{n},$$
$$\{W_{1,i}^{s_1} Z_{1,i}^{s_2} X_1^{v_i \beta} W_{2,i}^{s_1} Z_{2,i}^{s_2} X_2^{v_i \beta}\}_{i=1}^{n})$$

Game$_2$: The challenge ciphertext $\mathbf{CT}_2$ is generated under $(\vec{v}, \vec{0})$ and a random message $R \in \mathbb{G}_T$. The ciphertext $\mathbf{CT}_2$ is computed as follows:

$$(R', g^{s_2}, g_1^{s_1}, \{U_{1,i}^{s_1} T_{1,i}^{s_2} V_1^{v_i \alpha}, U_{2,i}^{s_1} T_{2,i}^{s_2} V_2^{v_i \alpha}\}_{i=1}^{n},$$
$$\{W_{1,i}^{s_1} Z_{1,i}^{s_2}, W_{2,i}^{s_1} Z_{2,i}^{s_2}\}_{i=1}^{n},)$$

Game$_3$: The challenge ciphertext $\mathbf{CT}_3$ is generated under $(\vec{v}, \vec{x})$ and a random message $R \in \mathbb{G}_T$. The ciphertext $\mathbf{CT}_3$ is computed as follows:

$$(R', g^{s_2}, g_1^{s_1}, \{U_{1,i}^{s_1} T_{1,i}^{s_2} V_1^{v_i \alpha}, U_{2,i}^{s_1} T_{2,i}^{s_2} V_2^{v_i \alpha}\}_{i=1}^{n},$$
$$\{W_{1,i}^{s_1} Z_{1,i}^{s_2} X_1^{x_i \beta}, W_{2,i}^{s_1} Z_{2,i}^{s_2} X_2^{x_i \beta}\}_{i=1}^{n})$$

Game$_4$: The challenge ciphertext $\mathbf{CT}_4$ is generated under $(\vec{0}, \vec{x})$ and a random message $R \in \mathbb{G}_T$. The ciphertext $\mathbf{CT}_4$ is computed as follows:

$$(R', g^{s_2}, g_1^{s_1}, \{U_{1,i}^{s_1} T_{1,i}^{s_2}, U_{2,i}^{s_1} T_{2,i}^{s_2}\}_{i=1}^{n},$$
$$\{W_{1,i}^{s_1} Z_{1,i}^{s_2} X_1^{x_i \beta}, W_{2,i}^{s_1} Z_{2,i}^{s_2} X_2^{x_i \beta}\}_{i=1}^{n})$$

Game$_5$: The challenge ciphertext $\mathbf{CT}_5$ is generated under $(\vec{x}, \vec{x})$ and a random message $R \in \mathbb{G}_T$. The ciphertext $\mathbf{CT}_5$ is computed as follows:

$$(R', g^{s_2}, g_1^{s_1}, \{U_{1,i}^{s_1} T_{1,i}^{s_2} V_1^{x_i \alpha}, U_{2,i}^{s_1} T_{2,i}^{s_2} V_2^{x_i \alpha}\}_{i=1}^{n},$$
$$\{W_{1,i}^{s_1} Z_{1,i}^{s_2} X_1^{x_i \beta}, W_{2,i}^{s_1} Z_{2,i}^{s_2} X_2^{x_i \beta}\}_{i=1}^{n},)$$

Game$_6$: The challenge ciphertext $\mathbf{CT}_6$ is generated under $(\vec{x}, \vec{x})$ and message $M_1 \in \mathbb{G}_T$. The ciphertext $\mathbf{CT}_6$ is computed as follows:

$$(M_1 \cdot Y^{-s_2}, g^{s_2}, g_1^{s_1}, \{U_{1,i}^{s_1} T_{1,i}^{s_2} V_1^{x_i \alpha}, U_{2,i}^{s_1} T_{2,i}^{s_2} V_2^{x_i \alpha}\}_{i=1}^{n},$$
$$\{W_{1,i}^{s_1} Z_{1,i}^{s_2} X_1^{x_i \beta}, W_{2,i}^{s_1} Z_{2,i}^{s_2} X_2^{x_i \beta}\}_{i=1}^{n}).$$

D. Indistinguishability Between Game$_0$ and Game$_1$

Suppose that there exists an adversary $\mathcal{A}$ which can distinguish the two games with a non-negligible advantage ϵ, we construct another algorithm $\mathcal{B}$ which uses $\mathcal{A}$ to solve the Decision Bilinear Diffie-Hellman problem also with advantage ϵ. On input $(g, A = g^a, B = g^b, C = g^c, Z) \in \mathbb{G}_4$, $\mathcal{B}$ simulates the game for $\mathcal{A}$ as follows.

- *Setup:* $\mathcal{B}$ selects random elements γ_1, γ_2, θ_1, θ_2, λ, $\{u_{1,i}\}_{i=1}^{n}$, $\{t_{1,i}\}_{i=1}^{n}$, $\{t_{2,i}\}_{i=1}^{n}$, $\{w_{1,i}\}_{i=1}^{n}$, $\{z_{1,i}\}_{i=1}^{n}$, $\{z_{2,i}\}_{i=1}^{n}$, in $\mathbb{Z}_p$.

  Then it selects a random $\Delta \in \mathbb{Z}_p$ to obtain $\{u_{2,i}\}_{i=1}^{n}, \{w_{2,i}\}_{i=1}^{n}$ under the condition:
  $$\Delta = \gamma_1 u_{2,i} - \gamma_2 u_{1,i} \quad \Delta = \theta_1 w_{2,i} - \theta_2 w_{1,i}.$$
  Then for $i = 1$ to n, $\mathcal{B}$ sets:
  $$U_{1,i} = g^{u_{1,i}}, \quad U_{2,i} = g^{u_{2,i}},$$
  $$T_{1,i} = (g^b)^{v_i \gamma_1} g^{t_{1,i}}, \quad T_{2,i} = (g^b)^{v_i \gamma_2} g^{t_{2,i}},$$
  $$W_{1,i} = g^{w_{1,i}}, \quad W_{2,i} = g^{w_{2,i}},$$
  $$Z_{1,i} = (g^b)^{v_i \theta_1} g^{z_{1,i}}, \quad Z_{2,i} = (g^b)^{v_i \theta_1} g^{z_{2,i}}.$$
  and
  $$V_1 = g^{\gamma_1}, \quad V_2 = g^{\gamma_2}, \quad X_1 = g^{\theta_1}, \quad X_2 = g^{\theta_2}$$
  $$g_1 = g^{\Delta}, \quad Y = e(g^a, g^b)^{-\Delta} \cdot e(g, g)^{\lambda}.$$
  Each public key component is distributed properly following the random exponents:
  $$\overline{t_{1,i}} = v_i \gamma_1 b + t_{1,i}, \quad \overline{t_{2,i}} = v_i \gamma_2 b + t_{2,i},$$
  $$\overline{z_{1,i}} = v_i \theta_1 b + z_{1,i}, \quad \overline{z_{2,i}} = v_i \theta_2 b + z_{2,i},$$
  $$g_2 = g^{-ab\Delta} g^{\lambda}.$$
- *Key Generation Phase 1 & 2:* $\mathcal{A}$ issues private key queries for the attribute list L. Consider a query with two vectors $\vec{y_V} = (y_{V_1}, \ldots, y_{V_n})$ and $\vec{y_Z} = (y_{Z_1}, \ldots, y_{Z_n})$. $\mathcal{A}$ can request the private key query as long as $(\overrightarrow{v}, \overrightarrow{y}_V) = (\overrightarrow{v}, \overrightarrow{y}_Z) = c_y \neq 0$.

  $\mathcal{B}$ picks random exponents $\{r_{1,i}\}_{i=1}^{n}, \{r_{2,i}\}_{i=1}^{n}$, and $f_1', f_2', r_1, r_2,$. Then $\mathcal{B}$ computes:
  $$\begin{aligned} K_{1,i} &= g^{-\gamma_2 r_{1,i}} g^{(\frac{a}{2c_y} + f_1') y_{V_i} u_{2,i}} \\ &= g^{\frac{a}{2c_y} y_{V_i} u_{2,i}} g^{-\gamma_2 r_{1,i}} g^{f_1' y_{V_i} u_{2,i}} \\ &= g^{\frac{a}{2c_y} y_{V_i} u_{2,i}} \cdot K_{1,i}'. \\ K_{2,i} &= g^{\gamma_1 r_{1,i}} g^{-(\frac{a}{2c_y} + f_1') y_{V_i} u_{1,i}} \\ &= g^{-\frac{a}{2c_y} y_{V_i} u_{1,i}} g^{\gamma_1 r_{1,i}} g^{-f_1' y_{V_i} u_{1,i}} \\ &= g^{-\frac{a}{2c_y} y_{V_i} u_{1,i}} \cdot K_{2,i}'. \end{aligned}$$

which implicitly sets: $f_1 = \frac{a}{2c_y} + f_1'$. Next $\mathcal{B}$ computes:

$$\begin{aligned} K_{3,i} &= g^{-\theta_2 r_{2,i}} g^{(\frac{a}{2cy}+f_2')y_{Z_i} w_{2,i}} \\ &= g^{\frac{a}{2cy} y_{Z_i} w_{2,i}} g^{-\theta_2 r_{2,i}} g^{f_2' y_{Z_i} w_{2,i}} \\ &= g^{\frac{a}{2cy} y_{Z_i} w_{2,i}} \cdot K_{3,i}'. \\ K_{4,i} &= g^{\theta_1 r_{2,i}} g^{-(\frac{a}{2cy}+f_2')y_{Z_i} w_{1,i}} \\ &= g^{-\frac{a}{2cy} y_{Z_i} w_{1,i}} g^{\theta_1 r_{1,i}} g^{-f_2' y_{Z_i} w_{1,i}} \\ &= g^{-\frac{a}{2cy} y_{Z_i} w_{1,i}} \cdot K_{4,i}'. \end{aligned}$$

which implicitly sets: $f_2 = \frac{a}{2c_y} + f_2'$. Then K_B and K_A are computed as:

$$\begin{aligned} K_B &= \prod\nolimits_{i=1}^{n} g^{-(r_{1,i}+r_{2,i})} \\ K_A &= g_2 \prod\nolimits_{i=1}^{n} K_{1,i}^{-\overline{t_{1,i}}} K_{2,i}^{-\overline{t_{2,i}}} K_{3,i}^{-\overline{z_{1,i}}} K_{4,i}^{-\overline{z_{2,i}}}. \end{aligned}$$

For K_A, we can compute:

$$\begin{aligned} &K_{1,i}^{-\overline{t_{1,i}}} K_{2,i}^{-\overline{t_{2,i}}} \\ &= (g^{\frac{a}{2cy} y_{V_i} u_{2,i}} \cdot K_{1,i}')^{-\overline{t_{1,i}}} \cdot (g^{-\frac{a}{2cy} y_{V_i} u_{1,i}} \cdot K_{2,i}')^{-\overline{t_{1,i}}} \\ &= (g^{\frac{a}{2cy} y_{V_i} u_{2,i}})^{-(v_i \gamma_1 b + t_{1,i})} \cdot (K_{1,i}')^{-\overline{t_{1,i}}} \cdot (K_{2,i}')^{-\overline{t_{2,i}}} \\ &\quad \cdot (g^{-\frac{a}{2cy} y_{V_i} u_{1,i}})^{-(v_i \gamma_2 b + t_{1,i})} \\ &= g^{-\frac{ab}{2cy} v_i y_{V_i} (\gamma_1 u_{2,i} - \gamma_2 u_{1,i})} \cdot (K_{1,i}')^{-\overline{t_{1,i}}} \cdot (K_{2,i}')^{-\overline{t_{2,i}}} \\ &\quad \cdot g^{\frac{a}{2cy} y_{V_i} (u_{1,i} t_{2,i} - u_{2,i} t_{1,i})} \\ &= g^{-\frac{ab\Delta}{2cy} v_i y_{V_i}} g^{\frac{a}{2cy} y_{V_i} (u_{1,i} t_{2,i} - u_{2,i} t_{1,i})} \cdot (K_{1,i}')^{-\overline{t_{1,i}}} \cdot (K_{2,i}')^{-\overline{t_{2,i}}}. \end{aligned}$$

Similarly, we can compute:

$$\begin{aligned} K_{3,i}^{-\overline{z_{1,i}}} K_{4,i}^{-\overline{z_{2,i}}} &= g^{-\frac{ab\Delta}{2cy} v_i y_{Z_i}} g^{\frac{a}{2cy} y_{Z_i} (w_{1,i} z_{2,i} - w_{2,i} z_{1,i})} \\ &\quad \cdot (K_{3,i}')^{-\overline{z_{1,i}}} \cdot (K_{4,i}')^{-\overline{z_{2,i}}}. \end{aligned}$$

Since $g_2 = g^{ab\Delta} g^{\lambda}$ then K_A can be computed as:

$$\begin{aligned} K_A = g^{\lambda} \prod\nolimits_{i=1}^{n} & \\ &\times g^{\frac{a}{2cy} y_{V_i} (u_{1,i} t_{2,i} - u_{2,i} t_{1,i})} g^{\frac{a}{2cy} y_{Z_i} (w_{1,i} z_{2,i} - w_{2,i} z_{1,i})} \\ &\cdot (K_{1,i}')^{-\overline{t_{1,i}}} \cdot (K_{2,i}')^{-\overline{t_{2,i}}} \cdot (K_{3,i}')^{-\overline{z_{1,i}}} \cdot (K_{4,i}')^{-\overline{z_{2,i}}}. \end{aligned}$$

$\mathcal{B}$ gives $\mathcal{A}$ the private key: $SK = (K_A, K_B, \{K_{1,i}, K_{2,i}, K_{3,i}, K_{4,i}\}_{i=1}^{n})$ for the queried vector $\vec{y}$.

- *Challenge Ciphertext:* To generate a challenge ciphertext, $\mathcal{B}$ picks random $s_1', \alpha', \beta' \in \mathbb{Z}_p$. $\mathcal{B}$ implicitly sets:

$$s_1 = s_1', s_2 = c, \quad \alpha = -bc + \alpha', \quad \beta = -bc + \beta'.$$

Then $\mathcal{B}$ sets $A = g^c = g^{s_2}$, $B = g^{\Delta s_1} = g_1^{s_1}$. For i from 1 to n, $\mathcal{B}$ computes:

$$\begin{aligned} C_{1,i} &= (g^{u_{1,i}})^{s_1} ((g^b)^{v_i \gamma_1} g^{t_{1,i}})^c g^{v_i \gamma_1 (-bc+\alpha')} = U_{1,i}^{s_1} T_{1,i}^{s_2} V_1^{v_i \alpha} \\ C_{2,i} &= (g^{u_{2,i}})^{s_1} ((g^b)^{v_i \gamma_2} g^{t_{2,i}})^c g^{v_i \gamma_2 (-bc+\alpha')} = U_{2,i}^{s_1} T_{2,i}^{s_2} V_2^{v_i \alpha} \\ C_{3,i} &= (g^{w_{1,i}})^{s_1} ((g^b)^{v_i \theta_1} g^{z_{1,i}})^c g^{v_i \theta_1 (-bc+\beta')} \\ &= W_{1,i}^{s_1} Z_{1,i}^{s_2} X_1^{v_i \beta} \\ C_{4,i} &= (g^{w_{2,i}})^{s_1} ((g^b)^{v_i \theta_2} g^{z_{2,i}})^c g^{v_i \theta_2 (-bc+\beta')} \\ &= W_{2,i}^{s_1} Z_{2,i}^{s_2} X_2^{v_i \beta} \end{aligned}$$

Next $\mathcal{B}$ computes $C_m = Z^{\Delta} \cdot e(g, g^c)^{\lambda} \cdot M_0$. If $Z = e(g, g)^{abc}$ the challenge ciphertext is distributed in **Game**$_0$, otherwise if Z is randomly chosen in $\mathbb{G}_T$, then the challenge ciphertext is distributed in **Game**$_1$. Hence, if $\mathcal{A}$ can distinguish these two games, $\mathcal{B}$ can solve the DBDH problem.

E. Indistinguishability Between Game$_1$ and Game$_2$

Suppose that there exists an adversary $\mathcal{A}$ which can distinguish these two games with non-negligible advantage ϵ, we construct another algorithm $\mathcal{B}$ which uses $\mathcal{A}$ to solve the Decision Linear problem with advantage ϵ. On input $(g, g^a, g^b, g^{ac}, g^d, Z) \in \mathbb{G}_6$, $\mathcal{B}$ simulates the game for $\mathcal{A}$ as follows.

- *Setup:* $\mathcal{B}$ selects random elements γ_1, γ_2, θ_1, θ_2, λ, $\{u_{1,i}\}_{i=1}^{n}, \{t_{1,i}\}_{i=1}^{n}, \{t_{2,i}\}_{i=1}^{n}, \{w_{1,i}\}_{i=1}^{n}, \{z_{1,i}\}_{i=1}^{n}, \{z_{2,i}\}_{i=1}^{n}$ in $\mathbb{Z}_p$. Then it selects a random $\Delta \in \mathbb{Z}_p$ to obtain $\{u_{2,i}\}_{i=1}^{n}, \{w_{2,i}\}_{i=1}^{n}, w_2, u_2$ under the condition:

$$\Delta = \gamma_1 u_{2,i} - \gamma_2 u_{1,i}, \quad \Delta = \theta_1 w_{2,i} - \theta_2 w_{1,i},$$

Then for $i = 1$ to n, $\mathcal{B}$ sets:

$$\begin{aligned} U_{1,i} &= (g^a)^{u_{1,i}}, \quad U_{2,i} = (g^a)^{u_{2,i}}, \\ T_{1,i} &= g^{t_{1,i}}, \quad T_{2,i} = g^{t_{2,i}}, \\ W_{1,i} &= (g^a)^{w_{1,i}} (g^b)^{\theta_1 v_i}, \quad W_{2,i} = (g^a)^{w_{2,i}} (g^b)^{\theta_2 v_i}, \\ Z_{1,i} &= g^{z_{1,i}} (g^b)^{\theta_1 v_i}, \quad Z_{2,i} = g^{z_{2,i}} (g^b)^{\theta_2 v_i},. \\ V_1 &= g^{\gamma_1}, \quad V_2 = g^{\gamma_2}, \quad X_1 = g^{\theta_1}, \\ X_2 &= g^{\theta_2}, \quad g_1 (g^a)^{\Delta}, \quad g_2 = g^{\lambda}. \end{aligned}$$

Each public key component is distributed properly following the random exponents:

$$\begin{aligned} \overline{u_{1,i}} &= a u_{1,i}, \quad \overline{u_{2,i}} = a u_{2,i}, \\ \overline{w_{1,i}} &= a w_{1,i} + \theta_1 b v_i, \quad \overline{w_{2,i}} = a w_{2,i} + \theta_2 b v_i, \\ \overline{z_{1,i}} &= v_i \theta_1 b + z_{1,i}, \quad \overline{z_{2,i}} = v_i \theta_2 b + z_{2,i}. \end{aligned}$$

- *Key Generation Phase 1 & 2:* $\mathcal{A}$ issues private key queries for the attribute list L. Consider a query will be created two vectors $\vec{y_V} = (y_{V_1}, \ldots, y_{V_n})$ and $\vec{y_Z} = (y_{Z_1}, \ldots, y_{Z_n})$ following (5). $\mathcal{B}$ picks random exponents $\{r_{1,i}'\}_{i=1}^{n}, \{r_{2,i}'\}_{i=1}^{n}$, and f_1, f_2. Then $\mathcal{B}$ computes:

$$\begin{aligned} K_{1,i} &= g^{-\gamma_2(-v_i y_{V_i} b + r_{1,i}')} g^{f_1 y_{V_i} u_{2,i}} \\ &= g^{\gamma_2 v_i y_{V_i} b} g^{-\gamma_2 r_{1,i}'} g^{f_1 y_{V_i} u_{2,i}} \\ &= g^{\gamma_2 v_i y_{V_i} b} \cdot K_{1,i}'. \\ K_{2,i} &= g^{\gamma_1(-v_i y_{V_i} b + r_{1,i}')} g^{-f_1 y_{V_i} u_{1,i}} \\ &= g^{-\gamma_1 v_i y_{V_i} b} g^{\gamma_2 r_{1,i}'} g^{-f_1 y_{V_i} u_{1,i}} \\ &= g^{-\gamma_1 v_i y_{V_i} b} \cdot K_{2,i}'. \end{aligned}$$

which implicitly sets: $r_{1,i} = -y_{V_i} v_i b + r_{1,i}'$. Next $\mathcal{B}$ computes:

$$\begin{aligned} K_{3,i} &= g^{-\theta_2(v_i y_{Z_i} b + a r_{2,i}')} g^{f_2 y_{Z_i} w_{2,i}} \\ &= g^{-\theta_2 v_i y_{Z_i} b} g^{-\gamma_2 r_{2,i}' a} g^{f_2 y_{Z_i} u_{2,i}} \\ &= g^{-\theta_2 v_i y_{Z_i} b} \cdot K_{3,i}'. \\ K_{4,i} &= g^{\theta_1(v_i y_{Z_i} b + a r_{2,i}')} g^{-f_2 y_{Z_i} w_{1,i}} \\ &= g^{\theta_1 v_i y_{Z_i} b} g^{\theta_2 a r_{2,i}'} g^{-f_2 y_{Z_i} w_{1,i}} \\ &= g^{\theta_1 v_i y_{Z_i} b} \cdot K_{4,i}'. \end{aligned}$$

which implicitly sets: $r_{2,i} = y_{Z_i}\upsilon_i b + ar'_{2,i}$.
Then K_B and K_A are computed as:

$$\begin{aligned} K_B &= \prod_{i=1}^{n} g^{-(r_{1,i}+r_{2,i})} \\ &\quad \times \prod_{i=1}^{n} g^{-(-y_{V_i}\upsilon_i b + r'_{1,i} + y_{Z_i}\upsilon_i b + ar'_{2,i})} \\ &= g^{-(r'_1+ar'_2)} \prod_{i=1}^{n} g^{-(r'_{1,i}+ar'_{2,i})}. \\ K_A &= g_2 \prod_{i=1}^{n} K_{1,i}^{-t_{1,i}} K_{2,i}^{t_{2,i}} K_{3,i}^{-\overline{z_{1,i}}} K_{4,i}^{-\overline{z_{2,i}}}. \end{aligned}$$

For K_A, we can compute:

$$\begin{aligned} &K_{1,i}^{-t_{1,i}} K_{2,i}^{t_{2,i}} \\ &\quad = g^{-\gamma_2 \upsilon_i y_{V_i} b t_{1,i}} g^{\gamma_1 \upsilon_i y_{V_i} b t_{2,i}} \cdot (K'_{1,i})^{-t_{1,i}} \cdot (K'_{2,i})^{-t_{2,i}}. \\ &K_{3,i}^{-\overline{z_{1,i}}} K_{4,i}^{-\overline{z_{2,i}}} \\ &\quad = g^{-\theta_2(\upsilon_i y_{Z_i} b)(-z_{1,i}-\theta_1 b\upsilon_i)} g^{(-\theta_2 ar'_{2,i})(-z_{1,i}-\theta_1 b\upsilon_i)} \\ &\qquad \times g^{(f_2 y_{Z_i} w_{2,i})(-z_{1,i}-\theta_1 b\upsilon_i)} g^{(-f_2 y_{Z_i} w_{1,i})(-z_{2,i}-\theta_2 b\upsilon_i)} \\ &\qquad \times g^{\theta_1(\upsilon_i y_{Z_i} b)(-z_{2,i}-\theta_2 b\upsilon_i)} g^{(\theta_1 ar'_{2,i})(-z_{2,i}-\theta_2 b\upsilon_i)} \\ &\quad = g^{-(\upsilon_i y_{Z_i} b + ar'_{2,i})\Delta} g^{(f_2 y_{Z_i} w_{2,i})(-z_{1,i}-\theta_1 b\upsilon_i)} \\ &\qquad \times g^{(-f_2 y_{Z_i} w_{1,i})(-z_{2,i}-\theta_2 b\upsilon_i)}. \end{aligned}$$

Since $g_2 = g^\lambda$ then K_A is computed as:

$$\begin{aligned} K_A &= g^\lambda \prod_{i=1}^{n} g^{-\gamma_2 \upsilon_i y_{V_i} b t_{1,i}} g^{\gamma_1 \upsilon_i y_{V_i} b t_{2,i}} \\ &\qquad \cdot (K'_{1,i})^{-t_{1,i}} \cdot (K'_{2,i})^{-t_{2,i}} \\ &\qquad \times g^{-(\upsilon_i y_{Z_i} b + ar'_{2,i})\Delta} g^{(f_2 y_{Z_i} w_{2,i})(-z_{1,i}-\theta_1 b\upsilon_i)} \\ &\qquad \times g^{(-f_2 y_{Z_i} w_{1,i})(-z_{2,i}-\theta_2 b\upsilon_i)}. \end{aligned}$$

$\mathcal{B}$ gives $\mathcal{A}$ the private key $SK = (K_A, K_B, \{K_{1,i}, K_{2,i}, K_{3,i}, K_{4,i}\}_{i=1}^{n},)$ for the queried vector $\vec{y}$.

- *Challenge Ciphertext:* To generate a challenge ciphertext, $\mathcal{B}$ picks random $s'_1, \alpha' \in \mathbb{Z}_p$. $\mathcal{B}$ implicitly sets:

$$s_1 = c, \quad s_2 = d, \quad \alpha = \alpha'$$

Then $\mathcal{B}$ sets: $A = g^d = g^{s_2}$, $B = (g^{ac})^\Delta = g_1^{s_1}$. For i from 1 to n, $\mathcal{B}$ computes:

$$\begin{aligned} C_{1,i} &= (g^{au_{1,i}})^c (g^d)^{t_{1,i}} g^{\upsilon_i \gamma_1(\alpha')} = U_{1,i}^{s_1} T_{1,i}^{s_2} V_1^{\upsilon_i \alpha} \\ C_{2,i} &= (g^{au_{2,i}})^c (g^d)^{t_{2,i}} g^{\upsilon_i \gamma_2(\alpha')} = U_{2,i}^{s_1} T_{2,i}^{s_2} V_2^{\upsilon_i \alpha} \end{aligned}$$

Next $\mathcal{B}$ computes for i from 1 to n:

$$\begin{aligned} C_{3,i} &= (g^{aw_{1,i}})^c (g^d)^{z_{1,i}} Z^{\theta_1 \upsilon_i}, \\ C_{4,i} &= (g^{aw_{2,i}})^c (g^d)^{z_{2,i}} Z^{\theta_2 \upsilon_i} \end{aligned}$$

If $Z = g^{b(c+d)g^r}$ for r chosen randomly in $\mathbb{Z}_p$, then $\mathcal{B}$ is simulating **Game**$_1$ with $\beta = r$:

$$\begin{aligned} C_{3,i} &= (g^{aw_{1,i}})^c (g^d)^{z_{1,i}} (g^{b(c+d)g^r})^{\theta_1 \upsilon_i} = W_{1,i}^{s_1} Z_{1,i}^{s_2} X_1^{\upsilon_i \beta} \\ C_{4,i} &= (g^{aw_{2,i}})^c (g^d)^{z_{2,i}} (g^{b(c+d)} g^r)^{\theta_2 \upsilon_i} = W_{2,i}^{s_1} Z_{2,i}^{s_2} X_2^{\upsilon_i \beta} \end{aligned}$$

If $Z = g^{b(c+d)}$, then $\mathcal{B}$ is simulating **Game**$_2$

$$\begin{aligned} C_{3,i} &= (g^{aw_{1,i}})^c (g^d)^{z_{1,i}} (g^{b(c+d)})^{\theta_1 \upsilon_i} = W_{1,i}^{s_1} Z_{1,i}^{s_2} \\ C_{4,i} &= (g^{aw_{2,i}})^c (g^d)^{z_{2,i}} (g^{b(c+d)})^{\theta_2 \upsilon_i} = W_{2,i}^{s_1} Z_{2,i}^{s_2}. \end{aligned}$$

Therefore, if $\mathcal{A}$ can distinguish the two games, $\mathcal{B}$ can solve the DLIN problem.

F. Indistinguishability of Game$_2$ and Game$_3$

Suppose that there exists an adversary $\mathcal{A}$ which can distinguish these two games with a non-negligible advantage ϵ, we construct another algorithm $\mathcal{B}$ that uses $\mathcal{A}$ to solve the Decision Linear problem with advantage ϵ. On input $(g, g^a, g^b, g^{ac}, g^d, Z) \in \mathbb{G}_6$, $\mathcal{B}$ simulates the game for $\mathcal{A}$ as follows.

- *Setup:* $\mathcal{B}$ selects random elements γ_1, γ_2, θ_1, θ_2, λ, $\{u_{1,i}\}_{i=1}^{n}$, $\{t_{1,i}\}_{i=1}^{n}$, $\{t_{2,i}\}_{i=1}^{n}$, $\{w_{1,i}\}_{i=1}^{n}$, $\{z_{1,i}\}_{i=1}^{n}$, $\{z_{2,i}\}_{i=1}^{n}$, in $\mathbb{Z}_p$. Then it selects a random $\Delta \in \mathbb{Z}_p$ to obtain $\{u_{2,i}\}_{i=1}^{n}, \{w_{2,i}\}_{i=1}^{n}, w_2, u_2$ under the condition:

$$\Delta = \gamma_1 u_{2,i} - \gamma_2 u_{1,i}, \quad \Delta = \theta_1 w_{2,i} - \theta_2 w_{1,i}.$$

Then for $i = 1$ to n, $\mathcal{B}$ sets:

$$\begin{aligned} U_{1,i} &= (g^a)^{u_{1,i}}, \quad U_{2,i} = (g^a)^{u_{2,i}}, \\ T_{1,i} &= g^{t_{1,i}}, \quad T_{2,i} = g^{t_{2,i}}, \\ W_{1,i} &= (g^a)^{w_{1,i}} (g^b)^{\theta_1 \upsilon_i}, \quad W_{2,i} = (g^a)^{w_{2,i}} (g^b)^{\theta_2 \upsilon_i}, \\ Z_{1,i} &= g^{z_{1,i}} (g^b)^{\theta_1 \upsilon_i}, \quad Z_{2,i} = g^{z_{2,i}} (g^b)^{\theta_2 x_i}, \\ V_1 &= g^{\gamma_1}, \quad V_2 = g^{\gamma_2}, \quad X_1 = g^{\theta_1}, \quad X_2 = g^{\theta_2}, \\ g_1 &= (g^a)^\Delta, \quad g_2 = g^\lambda. \end{aligned}$$

Each public key component is distributed properly following the random exponents:

$$\begin{aligned} \overline{u_{1,i}} &= au_{1,i}, \quad \overline{u_{2,i}} = au_{2,i} \\ \overline{w_{1,i}} &= aw_{1,i} + \theta_1 b\upsilon_i, \quad \overline{w_{2,i}} = aw_{2,i} + \theta_2 bx_i, \\ \overline{z_{1,i}} &= \upsilon_i \theta_1 b + z_{1,i}, \quad \overline{z_{2,i}} = \upsilon_i \theta_2 b + z_{2,i}. \end{aligned}$$

- *Key Generation Phase 1 & 2:* $\mathcal{A}$ issues private key queries for the attribute list L. Consider a query will be created two vectors $\vec{y_V} = (y_{V_1}, \ldots, y_{V_n})$ and $\vec{y_Z} = (y_{Z_1}, \ldots, y_{Z_n})$ following (5). Notice that $\mathcal{A}$ obey the restrictions defined in the model. That is $(\vec{\upsilon}, \vec{y}_V) = (\vec{\upsilon}, \vec{y}_Z) = 0 \bmod p$ if and only if $(\vec{x}, \vec{y}_V) \bmod p$ and $(\vec{x}, \vec{y}_Z) \bmod p$. There are two cases we need to consider.
  - *Case 1:* $(\vec{\upsilon}, \vec{y}_V) = 0 = (\vec{x}, \vec{y}_Z) \bmod p$. In this case, $\mathcal{B}$ picks random exponents $\{r'_{1,i}\}_{i=1}^{n}, \{r'_{2,i}\}_{i=1}^{n}$, and f_1, f_2. Then $\mathcal{B}$ computes:

$$\begin{aligned} K_{1,i} &= g^{-\gamma_2(-\upsilon_i(y_{V_i})b + r'_{1,i})} g^{f_1(y_{V_i})u_{2,i}} \\ &= g^{\gamma_2 \upsilon_i (y_{V_i})b} g^{-\gamma_2 r'_{1,i}} g^{f_1(y_{V_i})u_{2,i}} \\ &= g^{\gamma_2 \upsilon_i (y_{V_i})b} \cdot K'_{1,i}. \\ K_{2,i} &= g^{\gamma_1(-\upsilon_i(y_{V_i})b + r'_{1,i})} g^{f_1(y_{V_i})u_{1,i}} \\ &= g^{-\gamma_1 \upsilon_i (y_{V_i})b} g^{\gamma_2 r'_{1,i}} g^{f_1(y_{V_i})u_{1,i}} \\ &= g^{-\gamma_1 \upsilon_i (y_{V_i})b} \cdot K'_{2,i}. \end{aligned}$$

which implicitly sets: $r_{1,i} = -y_{V_i}\upsilon_i b + r'_{1,i}$. Next $\mathcal{B}$ computes:

$$\begin{aligned} K_{3,i} &= g^{-\theta_2(x_i(y_{Z_i})b + ar'_{2,i})} g^{f_2(y_{Z_i})w_{2,i}} \\ &= g^{-\theta_2 x_i (y_{Z_i})b} g^{-\gamma_2 r'_{2,i} a} g^{f_2(y_{Z_i})u_{2,i}} \\ &= g^{-\theta_2 x_i (y_{Z_i})b} \cdot K'_{3,i}. \\ K_{4,i} &= g^{\theta_1(x_i(y_{Z_i})b + ar'_{2,i})} g^{f_2(y_{Z_i})w_{1,i}} \\ &= g^{\theta_1 x_i (y_{Z_i})b} g^{\theta_2 ar'_{2,i}} g^{f_2(y_{Z_i})w_{1,i}} \\ &= g^{\theta_1 x_i (y_{Z_i})b} \cdot K'_{4,i}. \end{aligned}$$

which implicitly sets: $r_{2,i} = x_i y_{Z_i} b + ar'_{2,i}$ $\mathcal{B}$ also compute K_A and K_B as follows.

$$K_B = \prod_{i=1}^{n} g^{-(r_{1,i}+r_{2,i})} = \prod_{i=1}^{n} g^{-(r'_{1,i}+ar'_{2,i})}.$$
$$K_A = g_2 \prod_{i=1}^{n} K_{1,i}^{-t_{1,i}} K_{2,i}^{t_{2,i}} K_{3,i}^{-\overline{z_{1,i}}} K_{4,i}^{-\overline{z_{2,i}}}.$$

For K_A, its components are computed as follows:

$$\begin{aligned}
&K_{1,i}^{-t_{1,i}} K_{2,i}^{-t_{2,i}}\\
&\quad = g^{-\gamma_2 \upsilon_i y_{V_i} b t_{1,i}} g^{-\gamma_1 \upsilon_i y_{V_i} b t_{2,i}} \cdot (K'_{1,i})^{-t_{1,i}} \cdot (K'_{2,i})^{-t_{2,i}}.\\
&K_{3,i}^{-\overline{z_{1,i}}} K_{4,i}^{-\overline{z_{2,i}}}\\
&\quad = g^{-\theta_2 (x_i y_{Z_i} b)(-z_{1,i}-\theta_1 b x_i)} g^{(-\theta_2 a r'_{2,i})(-z_{1,i}-\theta_1 b x_i)}\\
&\qquad \times g^{(f_2 y_{Z_i} w_{2,i})(-z_{1,i}-\theta_1 b x_i)} g^{(-f_2 y_{Z_i} w_{1,i})(-z_{2,i}-\theta_2 b x_i)}\\
&\qquad \times g^{\theta_1 (x_i y_{Z_i} b)(-z_{2,i}-\theta_2 b x_i)} g^{(\theta_1 a r'_{2,i})(-z_{2,i}-\theta_2 b x_i)}\\
&\quad = g^{-(\upsilon_i y_{Z_i} b + a r'_{2,i})\Delta} g^{(f_2 y_{Z_i} w_{2,i})(-z_{1,i}-\theta_1 b x_i)}\\
&\qquad \times g^{(-f_2 y_{Z_i} w_{1,i})(-z_{2,i}-\theta_2 b x_i)}.
\end{aligned}$$

Since $g_2 = g^{\lambda}$ then K_A can be computed as:

$$\begin{aligned}
K_A = g^{\lambda} \prod_{i=1}^{n} & g^{-\gamma_2 \upsilon_i y_{V_i} b t_{1,i}} g^{\gamma_1 \upsilon_i y_{V_i} b t_{2,i}}\\
& \cdot (K'_{1,i})^{-t1,i} \cdot (K'_{2,i})^{-t_{2,i}}\\
& \cdot g^{-(x_i y_{Z_i} b + a r'_{2,i})\Delta} g^{(f_2 y_{Z_i} w_{2,i})(-z_{1,i}-\theta_1 b x_i)}\\
& \cdot g^{(-f_2 y_{Z_i} w_{1,i})(-z_{2,i}-\theta_2 b x_i)}.
\end{aligned}$$

$\mathcal{B}$ gives $\mathcal{A}$ the private key $SK = (K_A, K_B, \{K_{1,i}, K_{2,i}, K_{3,i}, K_{4,i}\}_{i=1}^{n}$ for the queried vector $\vec{y}$.

○ *Case 2:* $(\vec{\upsilon}, \vec{y}_V) = c_\upsilon \neq 0$ and $(\vec{x}, \vec{y}_Z) = c_x \neq 0$. In this case, $\mathcal{B}$ picks random exponents $\{r'_{1,i}\}_{i=1}^{n}, \{r'_{2,i}\}_{i=1}^{n}$, and f_1, f_2. Then $\mathcal{B}$ computes:

$$\begin{aligned}
K_{1,i} &= g^{-\gamma_2(-c_x \upsilon_i y_{V_i} b + r'_{1,i})} g^{f_1 y_{V_i} u_{2,i}}\\
&= g^{\gamma_2 c_x \upsilon_i y_{V_i} b} g^{-\gamma_2 r'_{1,i}} g^{f_1 y_{V_i} u_{2,i}}\\
&= g^{\gamma_2 c_x \upsilon_i y_{V_i} b} \cdot K'_{1,i}.\\
K_{2,i} &= g^{\gamma_1(c_x - \upsilon_i y_{V_i} b + r'_{1,i})} g^{-f_1 y_{V_i} u_{1,i}}\\
&= g^{-\gamma_1 c_x \upsilon_i y_{V_i} b} g^{\gamma_2 r'_{1,i}} g^{-f_1 y_{V_i} u_{1,i}}\\
&= g^{-\gamma_1 c_x \upsilon_i y_{V_i} b} \cdot K'_{2,i}.
\end{aligned}$$

which implicitly sets: $r_{1,i} = -c_x x_i y_{V_i} b + r'_{1,i}$. Next $\mathcal{B}$ computes:

$$\begin{aligned}
K_{3,i} &= g^{-\theta_2 c_\upsilon (x_i y_{Z_i} b + a r'_{2,i})} g^{f_2 y_{Z_i} w_{2,i}}\\
&= g^{-\theta_2 c_\upsilon x_i y_{Z_i} b} g^{-\gamma_2 r'_{2,i} a} g^{f_2 y_{Z_i} u_{2,i}}\\
&= g^{-\theta_2 c_\upsilon x_i y_{Z_i} b} \cdot K'_{3,i}.\\
K_{4,i} &= g^{\theta_1 c_\upsilon (x_i y_{Z_i} b + a r'_{2,i})} g^{-f_2 y_{Z_i} w_{1,i}}\\
&= g^{\theta_1 c_\upsilon x_i y_{Z_i} b} g^{\theta_2 a r'_{2,i}} g^{-f_2 y_{Z_i} w_{1,i}}\\
&= g^{\theta_1 c_\upsilon x_i y_{Z_i} b} \cdot K'_{4,i}.
\end{aligned}$$

which implicitly sets: $r_{2,i} = c_\upsilon x_i y_{Z_i} b + a r'_{2,i}, r_2$. Then K_B and K_A are computed as follows:

$$\begin{aligned}
K_B &= g^{-(r_1+r_2)} \prod_{i=1}^{n} g^{-(r_{1,i}+r_{2,i})}\\
&= \prod_{i=1}^{n} g^{-(r'_{1,i}+a r'_{2,i})}.\\
K_A &= g_2 \prod_{i=1}^{n} K_{1,i}^{-t_{1,i}} K_{2,i}^{t_{2,i}} K_{3,i}^{-\overline{z_{1,i}}} K_{4,i}^{-\overline{z_{2,i}}}.
\end{aligned}$$

For K_A, the components are computed as follows:

$$\begin{aligned}
&K_{1,i}^{-t_{1,i}} K_{2,i}^{-t_{2,i}}\\
&\quad = g^{-\gamma_2 c_x \upsilon_i y_{V_i} b t_{1,i}} g^{\gamma_1 c_x \upsilon_i y_{V_i} b t_{2,i}}\\
&\qquad \cdot (K'_{1,i})^{-t_{1,i}} \cdot (K'_{2,i})^{-t_{2,i}}.\\
&K_{3,i}^{-\overline{z_{1,i}}} K_{4,i}^{-\overline{z_{2,i}}}\\
&\quad = g^{-\theta_2 c_\upsilon (x_i y_{Z_i} b)(-z_{1,i}-\theta_1 b x_i)} g^{(-\theta_2 a r'_{2,i})(-z_{1,i}-\theta_1 b x_i)}\\
&\qquad \times g^{(f_2 y_{Z_i} w_{2,i})(-z_{1,i}-\theta_1 b x_i)}\\
&\qquad \times g^{\theta_1 c_\upsilon (x_i y_{Z_i} b)(-z_{2,i}-\theta_2 b x_i)} g^{(\theta_1 a r'_{2,i})(-z_{2,i}-\theta_2 b x_i)}\\
&\qquad \times g^{(-f_2 y_{Z_i} w_{1,i})(-z_{2,i}-\theta_2 b x_i)}\\
&\quad = g^{(-c_\upsilon x_i y_{Z_i} b + a r'_{2,i})\Delta} g^{(f_2 y_{Z_i} w_{2,i})(-z_{1,i}-\theta_1 b x_i)}\\
&\qquad \times g^{(-f_2 y_{Z_i} w_{1,i})(-z_{2,i}-\theta_2 b x_i)}
\end{aligned}$$

Since $g_2 = g^{\lambda}$ then K_A is computed as:

$$\begin{aligned}
K_A = g^{\lambda} \prod_{i=1}^{n} & g^{-\gamma_2 c_x \upsilon_i y_{V_i} b t_{1,i}} g^{\gamma_1 c_x \upsilon_i y_{V_i} b t_{2,i}}\\
& \cdot (K'_{1,i})^{-t_{1,i}} \cdot (K'_{2,i})^{-t_{2,i}}\\
& \cdot g^{-(c_\upsilon x_i y_{Z_i} b + a r'_{2,i})\Delta} g^{(f_2 y_{Z_i} w_{2,i})(-z_{1,i}-\theta_1 b x_i)}\\
& \cdot g^{(-f_2 y_{Z_i} w_{1,i})(-z_{2,i}-\theta_2 b x_i)}
\end{aligned}$$

$\mathcal{B}$ gives $\mathcal{A}$ the private key $SK = (K_A, K_B, \{K_{1,i}, K_{2,i}, K_{3,i}, K_{4,i}\}_{i=1}^{n})$ for the queried vector $\vec{y}$.

- *Challenge Ciphertext*: To generate a challenge ciphertext, $\mathcal{B}$ picks random $s'_1, \alpha' \in \mathbb{Z}_p$. $\mathcal{B}$ implicitly sets:

$$s_1 = c, \quad s_2 = d, \quad \alpha = \alpha'$$

Then $\mathcal{B}$ sets: $A = g^d = g^{s_2}, B = (g^{ac})^{\Delta} = g_1^{s_1}$. For i from 1 to n, $\mathcal{B}$ computes:

$$\begin{aligned}
C_{1,i} &= (g^{a u_{1,i}})^c (g^d)^{t_{1,i}} g^{\upsilon_i \gamma_1 (\alpha')} = U_{1,i}^{s_1} T_{1,i}^{s_2} V_1^{\upsilon_i \alpha}\\
C_{2,i} &= (g^{a u_{2,i}})^c (g^d)^{t_{2,i}} g^{\upsilon_i \gamma_2 (\alpha')} = U_{2,i}^{s_1} T_{2,i}^{s_2} V_2^{\upsilon_i \alpha}.
\end{aligned}$$

Next $\mathcal{B}$ computes for i from 1 to n:

$$\begin{aligned}
C_{3,i} &= (g^{a w_{1,i}})^c (g^d)^{z_{1,i}} Z^{\theta_1 x_i}\\
C_{4,i} &= (g^{a w_{2,i}})^c (g^d)^{z_{2,i}} Z^{\theta_2 x_i}.
\end{aligned}$$

If $Z = g^{b(c+d)}$ then, $\mathcal{B}$ is playing **Game**$_2$ with $\mathcal{A}$

$$\begin{aligned}
C_{3,i} &= (g^{a w_{1,i}})^c (g^d)^{z_{1,i}} (g^{b(c+d) g^r})^{\theta_1 \upsilon_i} = W_{1,i}^{s_1} Z_{1,i}^{s_2} X_1^{x_i \beta}\\
C_{4,i} &= (g^{a w_{2,i}})^c (g^d)^{z_{2,i}} (g^{b(c+d)} g^r)^{\theta_2 \upsilon_i} = W_{2,i}^{s_1} Z_{2,i}^{s_2} X_2^{x_i \beta}.
\end{aligned}$$

Otherwise, if $Z = g^{b(c+d) g^r}$ for r chosen randomly in $\mathbb{Z}_p$, then $\mathcal{B}$ is playing **Game**$_3$ with $\mathcal{A}$ by setting $\beta = r$

$$\begin{aligned}
C_{3,i} &= (g^{a w_{1,i}})^c (g^d)^{z_{1,i}} (g^{b(c+d)})^{\theta_1 \upsilon_i} = W_{1,i}^{s_1} Z_{1,i}^{s_2}\\
C_{4,i} &= (g^{a w_{2,i}})^c (g^d)^{z_{2,i}} (g^{b(c+d)})^{\theta_2 \upsilon_i} = W_{2,i}^{s_1} Z_{2,i}^{s_2}.
\end{aligned}$$

Therefore, if $\mathcal{A}$ can distinguish $Game_2$ from $Game_3$, then $\mathcal{B}$ can solve the DLIN problem.

The rest of the proof is similar to the above proofs:

- the indistinguishability between $Game_3$ and $Game_4$ can be proved in the same way as for $Game_2$ and $Game_3$;
- the indistinguishability between $Game_4$ and $Game_5$ can be proved in the same way as for $Game_1$ and $Game_2$;
- the indistinguishability of $Game_5$ and $Game_6$ can be proved in the same way as for $Game_0$ and $Game_1$.

V. Conclusion

In this paper, we presented two new constructions of Ciphertext Policy Attribute Based Encryption for the AND-Gate with wildcard access policy. Our first scheme achieves constant ciphertext size, but cannot hide the access policy. On the other hand, our second scheme can even hide the access policy against the legitimate decryptors. We proved that our second construction is secure under the Decisional Bilinear Diffie-Hellman and the Decision Linear assumptions. One shortcoming of our second construction is that its ciphertext size is no longer constant, then proving this construction in fully secure. We leave the solution for this problem as our future work.

Acknowledgement

The authors thank the anonymous reviewers for their invaluable comments and suggestions.

References

[1] M. Abdalla, A. De Caro, and D. H. Phan, “Generalized key delegation for wildcarded identity-based and inner-product encryption,” *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 6, pp. 1695–1706, Dec. 2012.

[2] N. Attrapadung, B. Libert, and E. de Panafieu, “Expressive key-policy attribute-based encryption with constant-size ciphertexts,” in *Public Key Cryptography* (Lecture Notes in Computer Science), vol. 6571. Berlin, Germany: Springer-Verlag, 2011, pp. 90–108.

[3] J. Bethencourt, A. Sahai, and B. Waters, “Ciphertext-policy attribute-based encryption,” in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2007, pp. 321–334.

[4] D. Boneh and M. K. Franklin, “Identity-based encryption from the Weil pairing,” in *Proc. 21st Annu. Int. CRYPTO*, 2001, pp. 213–229.

[5] C. Chen *et al.*, “Fully secure attribute-based systems with short ciphertexts/signatures and threshold access structures,” in *Topics in Cryptology* (Lecture Notes in Computer Science), vol. 7779, E. Dawson, Ed. Berlin, Germany: Springer-Verlag, 2013, pp. 50–67.

[6] C. Chen, Z. Zhang, and D. Feng, “Efficient ciphertext policy attribute-based encryption with constant-size ciphertext and constant computation-cost,” in *Proc. 5th Int. Conf. Provable Secur. (ProvSec)*, 2011, pp. 84–101.

[7] L. Cheung and C. Newport, “Provably secure ciphertext policy ABE,” in *Proc. 14th ACM Conf. Comput. Commun. Secur. (CCS)*, New York, NY, USA, 2007, pp. 456–465.

[8] N. Doshi and D. Jinwala, “Hidden access structure ciphertext policy attribute based encryption with constant length ciphertext,” in *Proc. Int. Conf. Adv. Comput., Netw. Secur.*, 2012, pp. 515–523.

[9] K. Emura, A. Miyaji, A. Nomura, K. Omote, and M. Soshi, “A ciphertext-policy attribute-based encryption scheme with constant ciphertext length,” in *Proc. 5th Int. Conf. ISPEC*, 2009, pp. 13–23.

[10] A. Ge, R. Zhang, C. Chen, C. Ma, and Z. Zhang, “Threshold ciphertext policy attribute-based encryption with constant size ciphertexts,” in *Proc. 17th Austral. Conf. Inf. Secur. Privacy*, 2012, pp. 336–349.

[11] V. Goyal, A. Jain, O. Pandey, and A. Sahai, “Bounded ciphertext policy attribute based encryption,” in *Proc. 35th Int. Colloq. Auto., Lang. Program. (ICALP)*, 2008, pp. 579–591.

[12] V. Goyal, O. Pandey, A. Sahai, and B. Waters, “Attribute-based encryption for fine-grained access control of encrypted data,” in *Proc. 13th ACM Conf. Comput. Commun. Secur. (CCS)*, 2006, pp. 89–98.

[13] J. Herranz, F. Laguillaumie, and C. Ràfols, “Constant size ciphertexts in threshold attribute-based encryption,” in *13th PKC*, 2010, pp. 19–34.

[14] J. Katz, A. Sahai, and B. Waters, “Predicate encryption supporting disjunctions, polynomial equations, and inner products,” in *Proc. Theory Appl. Cryptogr. Techn. 27th Annu. Int. Conf. Adv. Cryptol. (EUROCRYPT)*, 2008, pp. 146–162.

[15] J. Lai, R. H. Deng, and Y. Li, “Fully secure cipertext-policy hiding CP-ABE,” in *Proc. 7th Int. Conf. Inf. Secur. Pract. Exper. (ISPEC)*, 2011, pp. 24–39.

[16] A. Lewko, T. Okamoto, A. Sahai, K. Takashima, and B. Waters, “Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption,” in *Proc. 24th Annu. Int. EUROCRYPT*, 2010, pp. 62–91.

[17] A. Lewko and B. Waters, “New proof methods for attribute-based encryption: Achieving full security through selective techniques,” in *Proc. 32nd Annu. Conf. CRYPTO*, 2012, pp. 180–198.

[18] J. Li, K. Ren, B. Zhu, and Z. Wan, “Privacy-aware attribute-based encryption with user accountability,” in *Proc. 12th Int. Conf. Inf. Secur. (ISC)*, 2009, pp. 347–362.

[19] X. Li, D. Gu, Y. Ren, N. Ding, and K. Yuan, “Efficient ciphertext-policy attribute based encryption with hidden policy,” in *Internet and Distributed Computing Systems*, vol. 7646. Berlin, Germany: Springer-Verlag, 2012, pp. 146–159.

[20] T. Nishide, K. Yoneyama, and K. Ohta, “Attribute-based encryption with partially hidden encryptor-specified access structures,” in *Proc. 6th Int. Conf. Appl. Cryptogr. Netw. Secur. (ACNS)*, 2008, pp. 111–129.

[21] A. Sahai and B. Waters, “Fuzzy identity-based encryption,” in *Proc. 24th Annu. Int. Conf. Theory Appl. Cryptogr. Techn. (EUROCRYPT)*, 2005, pp. 457–473.

[22] S. Sedghi, P. van Liesdonk, S. Nikova, P. Hartel, and W. Jonker, “Searching keywords with wildcards on encrypted data,” in *Security and Cryptography for Networks* (Lecture Notes in Computer Science), vol. 6280. Berlin, Germany: Springer-Verlag, 2010, pp. 138–153.

[23] A. Shamir, “Identity-based cryptosystems and signature schemes,” in *Proc. CRYPTO*, 1984, pp. 47–53.

[24] E. Shi and B. Waters, “Delegating capabilities in predicate encryption systems,” in *Proc. 35th Int. Colloq. Auto., Lang. Program. (ICALP)*, 2008, pp. 560–578.

[25] T. V. X. Phuong, G. Yang, and W. Susilo, “Poster: Efficient ciphertext policy attribute based encryption under decisional linear assumption,” in *Proc. 21st ACM Conf. Comput. Commun. Secur. (CCS)*, Arizona City, AZ, USA, 2014.

[26] B. Waters, “Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization,” in *Proc. 14th Int. Conf. Public Key Cryptogr.*, 2011, pp. 53–70.

[27] Y. Zhang, D. Zheng, X. Chen, J. Li, and H. Li, “Computationally efficient ciphertext-policy attribute-based encryption with constant-size ciphertexts,” in *Provable Security*. New York, NY, USA: Springer-Verlag, 2014, pp. 259–273.

[28] Z. Zhou and D. Huang, “On efficient ciphertext-policy attribute based encryption and broadcast encryption: Extended abstract,” in *Proc. 17th ACM Conf. Comput. Commun. Secur. (CCS)*, New York, NY, USA, 2010, pp. 753–755.

Tran Viet Xuan Phuong received the bachelor’s degree from the Vietnam National University of Science, in 2010, and the M.S. degree from the Japan Advanced Institute of Science Technology, in 2012. She is currently pursuing the Ph.D. degree with the School of Computer and Information Technology, University of Wollongong, Wollongong, NSW, Australia. Her main research interest is applied cryptography. She is also a member of the Centre for Computer and Information Security Research.

Guomin Yang (M’13) received the Ph.D. degree in computer science from the City University of Hong Kong, Hong Kong, in 2009.

He is currently a Senior Lecturer and a DECRA Fellow with the School of Computing and Information Technology, University of Wollongong, Wollongong, NSW, Australia. His current research interests include applied cryptography and network security.

Willy Susilo (SM’08) received the Ph.D. degree in computer science from the University of Wollongong, Wollongong, NSW, Australia.

He is currently a Professor and the Head of the School of Computing and Information Technology with the University of Wollongong. He is also the Director of Centre for Computer and Information Security Research, University of Wollongong. His current research interests include cloud security, cryptography, and information security.