2014년 1월 전자공학회 논문지 제 51권 제 1 호
Journal of The Institute of Electronics and Information Engineers Vol. 51, NO. 1, January 2014 http://dx.doi.org/10.5573/ieie.2014.51.1.057

논문 2014-51-1-9

Bilinear Group에서 속성 은닉을 가지는 안전한 내적 암호화 방식

(Secure Inner Product Encryption Scheme with Attribute Hiding in Bilinear Groups)

리프키 사디킨*, 박 영 호**

(Rifki Sadikin and YoungHo Park©)

요 약

내적 암호화 방식은 비밀키와 암호문 사이에 파인 그레인 관계를 제공하는 암호학적 프리미티브이다. 본 논문은 완전한 속성 은닉 보호를 수행하는 새로운 IPE 방식을 제안한다. 제안한 IPE 방식은 합성 위수의 bilinear groups에 기반한다. 본 논문에서는 이중 암호화 시스템 체계를 사용하여 제안한 IPE의 완전한 속성 은닉 보호를 증명한다. 성능 분석에서 기존의 IPE 방식들과 제안한 IPE 방식의 연산량과 메모리 할당량을 비교한다.

Abstract

Inner product encryption (IPE) scheme is a cryptographic primitive that provides fine grained relations between secret keys and ciphertexts. This paper proposes a new IPE scheme which achieves fully attribute hiding security. Our IPE scheme is based on bilinear groups of a composite order. We prove the fully attribute hiding security of our IPE by using dual encryption system framework. In performance analysis, we compare the computation cost and memory requirement of our proposed IPE to other existing IPE schemes.

Keywords : functional encryption, inner product encryption, attribute based encryption

Ⅰ. Introduction

Nowadays the needs of privacy and security of data distribution over a public network becomes eminent by emerging popularity of cloud systems. In a public cloud computing model, users can upload their sensitive data or query for private data by relying on a provider which is assumed to be honest or at least semi-honest[1]. Traditional approaches such as relying authentication and authorization to one trusted server are not comply to the characteristics of cloud system. Distributing security services across all parties in the system can overcome the problem in traditional approach. However, a traditional public key

* 정회원, 경북대학교 전자전기컴퓨터학부
(Kyungpook National University)
** 정회원, 경북대학교 산업전자공학과
(Kyungpook National University)
© Corresponding Author(E-mail: parkyh@knu.ac.kr)
※ 이 논문은 2012년도 정부(교육과학기술부)의 재원으로 한국연구재단의 기초연구사업 지원을 받아 수행된 것임(NRF-2012R1A1A4A01002603)
접수일자: 2013년8월23일, 수정완료일: 2013년12월24일

crypto system can not easily adapted to provide such security since a public key only related to one private key. We need an encryption scheme that can express complex relation between secret keys and ciphertexts.

Predicate encryption allows complex relation between secret keys and ciphertexts. The secret keys in a predicate encryption scheme are associated to a predicate p_X with a parameter X and the ciphertexts are associated by a parameter Y. The decryption only work only if $p_X(Y)=1$. There are 3 subclasses of predicate encryption: anonymous identity based encryption which supports a identity equality predicate[2~3], hidden vector encryption scheme which supports a conjunctive combination of equality predicate[4~6] and inner product encryption (IPE) scheme which supports polynomial evaluation, conjunctions, and disjunctions for attribute based encryption[7~14].

The IPE scheme uses vector of attributes $\vec{x},\vec{y}\in \boldsymbol{Z_n}$ and an inner product function to define its predicate. There are two security definitions for IPE scheme: *payload hiding* and *attribute hiding*[7]. An IPE scheme is called *payload hiding* if the attacker learn nothing about the message m_b from the challenge ciphertext $CT_{m_\beta,\vec{y}*}$ where $\beta\in\{0,1\}$ in an indistinguishable game. An IPE is called *attribute hiding* if the attacker learn nothing about the challenge vector attributes $\overrightarrow{y_\beta}*$ from the challenge ciphertext $CT_{m,\vec{y_\beta}*}$.

The security of an IPE scheme is proven under indistinguishability game with a presence of a chosen plaintext (ciphertext) attacker under full or selective security definition. If the challenge plaintexts and vectors of attributes are chosen before the game is started then the IPE scheme is proven under *selective security*[15]. Otherwise, the IPE is proven under *full security* definition[16].

The first inner product encryption was proposed by [7~8] proven under selective security and [9~13] achieved full security IPE scheme by using dual encryption technique from [14]. Based on *payload/attribute hiding,* [10] achieves *payload hiding* IPE scheme, [9] achieves weakly *attribute hiding* and [11~14] achieves *attribute-hiding*.

One of limitations of many existing IPE schemes is attribute vectors bounded by public parameters. The length of attributes vectors are limited to a constant. This limitation comes from the fact that the parameters for secret key and encryption are settled once the public parameters have been set. The first attribute based encryption that present unbounded version of an attribute based encryption scheme is proposed in [17]. While the only unbounded IPE known so far presented in [13].

In this paper, we propose an inner product encryption scheme that achieves full and attribute hiding security definition where the length of vectors in ciphertexts and secret keys are not bounded to a constant length in public parameters. Our construction achieves *attribute hiding* and adaptive security by proving the scheme under indistinguishable game by using complexity assumption from sub group problem and we achieve unboundedness by using Lagrange interpolation of prechosen random points. The proposed IPE scheme has different structure with existing unbounded fully secure and attribute hiding IP scheme [13]. Moreover, the proposed IPE scheme has advantages from [13] that the length of ciphertext/secret key is shorter than that in [13]. Implicitly, the number of pairing computing for decryption in ours is lesser than that in [13].

II. Definitions

In this section, we present definitions of an inner product encryption and its security, composite bilinear groups and complexity assumptions for building our scheme.

2.1 Inner Product Encryption Scheme

Let $\vec{x},\vec{y}\in\Sigma^l$ be attributes vector in secret key and ciphertext space. We define a predicate:

$$p_{\vec{x}}(\vec{y})=\begin{cases}1, \text{if} \langle \vec{x},\vec{y}\rangle=0\\ 0, \text{otherwise}\end{cases} \quad (1)$$

Where $\vec{x}=(x_1,\cdots,x_l)$, $\vec{y}=(y_1,\cdots,y_l)$, and $\langle \vec{x},\vec{y}\rangle=x_1y_1+\cdots+x_ly_l$ is a notation for inner product between $\vec{x}$ and $\vec{y}$. An inner product encryption scheme consists the following algorithms:

$(PK,MSK)\leftarrow$ ***Setup***(λ,d) with λ is a security parameter and $d$contains the information of minimum vector attributes length. The setup algorithm returns a set of public parameters PK and a master key MSK.

$SK_{\vec{x}}\leftarrow$ ***Keygen***$(PK,MSK,\vec{x})$ returns a secret key $SK_{\vec{x}}$corresponds to attribute $\vec{x}$.

$CT_{m,\vec{y}}\leftarrow$ ***Encrypt***$(PK,m,\vec{y})$ where m is a plaintext from plaintext space. The encryption algorithm returns a ciphertext $CT_{m,\vec{y}}$ associated with a vector attribute $\vec{y}$.

$m|\perp\leftarrow$ ***Decrypt***$(SK_{\vec{x}},CT_{m,\vec{y}})$. The decryption algorithm returns the plaintext m or an indistinguished symbol $\perp$.

An IPE system should satisfy the following correctness requirement: for all (PK,MSK) generated by ***Setup***(λ,d), for any key $SK_{\vec{x}}\leftarrow$ ***Keygen***$(PK,MSK,\vec{x})$ and any ciphertext $CT_{m,\vec{y}}\leftarrow$ ***Encrypt***$(PK,m,\vec{y})$ we have:

$$\textbf{\textit{Decrypt}}(SK_{\vec{x}},CT_{m,\vec{y}})=m \quad \text{if} \langle \vec{x},\vec{y}\rangle=0 \quad (2)$$

2.2 Security of IPE scheme

An inner product encryption scheme over vector attributes space $\vec{x},\vec{y}\in\Sigma^l$ is an *attribute hiding* and full security with the presence of a chosen plaintext adversary $\boldsymbol{A}$. if for all PPT adversaries $\boldsymbol{A}$, the advantage of $\boldsymbol{A}$ in the following game is negligible in the security parameter λ:

Setup phase. In setup phase, the challenger $\boldsymbol{C}$ runs ***Setup***(λ,d) and gives public parameter PK to adversary $\boldsymbol{A}$ and keep the master key MSK for itself.

Phase 1. In phase 1, the adversary $\boldsymbol{A}$ makes q queries for secret key associated to q vector attributes $\vec{x_1},\cdots,\vec{x_q}$ for the challenger $\boldsymbol{C}$. For each k -th query, $\boldsymbol{C}$ gives $SK_{\vec{x_k}}\leftarrow$ ***Keygen***$(PK,MSK,\vec{x_k})$ to $\boldsymbol{A}$.

Challenge. In challenge phase, the adversary $\boldsymbol{A}$ gives the challenger $\boldsymbol{C}$ two tuples $(\vec{y_0}^*,m_0)$ and $(\vec{y_1}^*,m_1)$ with $m_0,m_1\in\boldsymbol{M}$ (plaintext space) with the restrictions:

- *Payload hiding.* If $m_0\neq m_1$ None of vector attributes $\vec{x_k}$ in $\vec{x_1},\cdots,\vec{x_q}$ queried in phase 1 satisfy $\langle \vec{x_k},\vec{y_0}^*\rangle=\langle \vec{x_k},\vec{y_1}^*\rangle=0$.
- *Attribute hiding.*. If $m_0=m_1$ then for any key query $\vec{x_k}$ in $\vec{x_1},\cdots,\vec{x_q}$ satisfied $\langle \vec{x_k},\vec{y_0}^*\rangle=\langle \vec{x_k},\vec{y_1}^*\rangle$.

The challenger $\boldsymbol{C}$ throws a binary dice $\beta\leftarrow\{0,1\}$ and then sends $CT_{m_\beta,\vec{y_\beta}^*}\leftarrow$ ***Encrypt***$(PK,m_\beta,\vec{y_\beta}^*)$ to the adversary $\boldsymbol{A}$

Phase 2. Repeat phase 1, to query $\vec{x_{q+1}},\cdots,\vec{x_{2q}}$ with the same restriction with step 3. The challenger give the corresponding key $SK_{\vec{x_k}}\leftarrow$ ***Keygen***$(PK,MSK,\vec{x_k})$ to $\boldsymbol{A}$.

Guess. At the end of game, the adversary $\boldsymbol{A}$ submit a guess b' for b. The adversary $\boldsymbol{A}$ wins the game if $b=b'$.

The advantage for the adversary $\boldsymbol{A}$ in above game is defined as:

$$\text{Adv}_{\boldsymbol{A}}^{Game}=\left|\Pr[b=b']-\frac{1}{2}\right| \quad (3)$$

Relaxed version of *attribute hiding* that restrict only to $\langle \overrightarrow{x_k}, \overrightarrow{y_0}^* \rangle = \langle \overrightarrow{x_k}, \overrightarrow{y_1}^* \rangle \neq 0$ (the attacker can only queried the vector that does not satisfy inner product predicate) is called *weakly attribute hiding*.

2.3 Composite Order Bilinear Groups

Let a group generator algorithm $\mathbf{gen}(\lambda)$ produces the following set $(N, \mathbf{G}, \mathbf{G_T}, \hat{\mathbf{e}})$ with N is a composite number produces by four distinct primes $N = p_1 p_2 p_3 p_4$, $\mathbf{G}$ is an additive group, $\mathbf{G_T}$ is a multiplicative group with order N, and $\hat{\mathbf{e}}$ is a bilinear map $\mathbf{G} \times \mathbf{G} \rightarrow \mathbf{G_T}$ that satisfied the following characteristics:

- Bilinearity, $\forall g, h \in \mathbf{G}$ and $\forall a, b \in \mathbf{Z_N}$ we have: $\hat{\mathbf{e}}(g^a, h^b) = \hat{\mathbf{e}}(g, h)^{ab}$.
- Non-degeneration, $\exists\ g \in \mathbf{G}$ such that $\hat{\mathbf{e}}(g, g) \neq 1$.

We assume operation in $\mathbf{G}$, $\mathbf{G_T}$ and bilinear map $\hat{\mathbf{e}}$ are all computable in time polynomial of λ. We point out that elements from different subgroup of $\mathbf{G}$ are orthogonal each other. That is, let denotes sub group of $\mathbf{G}$ order of p_i as $\mathbf{G}_{p_i}$, if $g \in \mathbf{G}_{p_i}$ and $h \in \mathbf{G}_{p_j}$ then $\mathrm{e}(g, h) = 1$ if $i \neq j$.

2.4 General Subgroup Decision Problem

We use variants of subgroup decision problem assumptions from [18]. The assumptions are listed in Table 1.

표 1. 가정들
Table 1. Assumptions.

No	Set Given (D)	Decide
1	$\{I, g_1, g_3, g_4\}$	$T_0 \in \mathbf{G}_{p_1}$ $T_1 \in \mathbf{G}_{p_1 p_2}$
2	$\{I, g_1, g_3, g_4, A_1 A_2, B_2 B_3\}$	$T_0 \in \mathbf{G}_{p_1 p_3}$ $T_1 \in \mathbf{G}_{p_1 p_2 p_3}$
3	$\{I, g_1, g_2, g_3, g_4,$ $g_1^{\alpha} A_2, g_1^{s} B_2\}$	$T_0 = \hat{\mathbf{e}}(g_1, g_1)^{\alpha s}$ $T_1 \in \mathbf{G}_T$
4	$\{I, g_1, g_2, g_3, g_4, U,$ $U^s A_{24}, U^{\hat{r}}, A_1 A_4,$ $A_1^{\hat{r}} A_2, g_1^{\hat{r}} B_2, g_1^{s} B_{24}\}$	$T_0 = A_1^{s} D_{24}$ $T_1 \in \mathbf{G}_{p_1 p_2 p_4}$

We define the advantage for an polynomial time algorithm $\boldsymbol{A}$ in decisional game for breaking Assumption 1,2,3 or 4: given a set D from k-th row in Table 1 and T_β, where $\beta \in \{0,1\}$ $\boldsymbol{A}$ to be:

$$\mathrm{Adv}_{\boldsymbol{A}}^{Assumption-k} = \left| \Pr\left[\boldsymbol{A}(D, T_0)\right] - \Pr\left[\boldsymbol{A}(D, T_1)\right] = 1 \right| \quad (4)$$

Ⅲ. The Proposed IPE Scheme

Our attribute hiding inner product encryption scheme consists of 4 algorithms:

Setup(λ, d). Using the security parameter , Setup runs a group generator $\mathbf{gen}(\lambda)$ and receives $(N = p_1 p_2 p_3 p_4, \mathbf{G}, \mathbf{G_T}, \hat{\mathbf{e}})$. The Setup algorithm implicitly declares that $\mathbf{Z}_{\boldsymbol{N}}$ as the attributes space.

The Setup algorithm chooses randomly $X_1, Y_1 \in \mathbf{G}_{p_1}$. $X_3 \in \mathbf{G}_{p_3}$ and $Y_4, X_4, U_{4,1}, \cdots, U_{4,d}, V_{4,1}, \cdots, V_{4,d}, W_{4,1}, \cdots, W_{4,d} \in \mathbf{G}_{p_4}$ where $U_{4,1}, \cdots, U_{4,d}, V_{4,1}, \cdots, V_{4,d}, W_{4,1}, \cdots, W_{4,d}$ are computed through exponentiation of Y_4. Next, it chooses randomly $\alpha, a, \alpha_0, \alpha_1, \cdots, \alpha_d, \beta_0, \beta_1, \cdots, \beta_d, \gamma_0, \cdots, \gamma_d \in \mathbf{Z}_{\boldsymbol{N}}$, Then it creates 3 polynomial functions:

$$u(x) = \alpha_0 + \alpha_1 x + \cdots + x^d \alpha_d \ . \quad (5)$$

$$v(x) = \beta_0 + \beta_1 x + \cdots + x^d \beta_d \quad (6)$$

$$w(x) = \gamma_0 + \gamma_1 x + \cdots + \gamma^d \beta_d \quad (7)$$

Then, it computes $\mathbf{Y} = \hat{\mathbf{e}}(Y_1, Y_1)^{\alpha}$ and publishes public parameters PK as follows:

$$PK = \left\{ \begin{array}{l} Y_1, Y_4, Q = X_1 X_4, Y_1^{a}, \\ Y_1^{u(0)} U_{4,0}, \cdots, Y_1^{u(d)} U_{4,d}, \\ Y_1^{v(0)} V_{4,0}, \cdots, Y_1^{v(d)} V_{4,d}, \\ Y_1^{w(0)} W_{4,0}, \cdots, Y_1^{w(d)} W_{4,d}, \\ \mathbf{Y} \end{array} \right\} \quad (8)$$

with three public functions $\mathbf{U}(x), \mathbf{V}(x), \mathbf{W}(x) : \mathbf{Z_N} \rightarrow \mathbf{G}_{p_1 p_4}$ defined as:

$$\mathbf{U}(k)=\begin{cases} Y^{u(k)}U_{4,k}, & \text{,if } k\leqq d \\ \prod_{i=0}^{d}\mathbf{U}(i)^{\triangle_{i,S}(k)} & \text{,otherwise} \end{cases} \quad (9)$$

$$\mathbf{V}(k)=\begin{cases} Y^{v(k)}V_{4,k}, & \text{,if } k\leqq d \\ \prod_{i=0}^{d}\mathbf{V}(i)^{\triangle_{i,S}(k)} & \text{,otherwise} \end{cases} \quad (10)$$

$$\mathbf{W}(k)=\begin{cases} Y^{w(k)}W_{4,k}, & \text{,if } k\leqq d \\ \prod_{i=0}^{d}\mathbf{W}(i)^{\triangle_{i,S}(k)} & \text{,otherwise} \end{cases} \quad (11)$$

where $S=\{0,1,\cdots,d\}$ and $\triangle_{i,S}(x)$ is called Lagrange coefficient defined as follows:

$$\triangle_{i,S}(x)=\prod_{\forall j\in S, i\neq j}\frac{x-j}{i-j} \quad (12)$$

By using Lagrange interpolation, the public functions in Equation 9, 10 or 11 implicitly computes $\mathbf{U}(x)=Y_1^{u(x)}$, $\mathbf{V}(x)=Y_1^{v(x)}$ or $\mathbf{U}(x)=Y_1^{u(x)}$ respectively. Note that $Y_1^{u(0)},\cdots,Y_1^{u(d)},Y_1^{v(0)},\cdots,Y_1^{v(d)},Y_1^{w(0)},\cdots,Y_1^{w(d)}$ are exposed in the public parameter. $\mathbf{U}(x),\mathbf{V}(x),\mathbf{W}(x)$

At the end the Setup algorithm keeps the master key MSK for it self. The master key is computed as follows:

$$MSK=\alpha,u(x),X_1,X_3 \quad (13)$$

Keygen$(PK,MSK,\vec{x}=[x_1,\cdots,x_l])$. To generate a secret key $SK_{\vec{x}}$ associated with a vector attributes $\vec{x}$, the ***Keygen*** algorithm chooses randomly $r\in\mathbf{Z_N}$, $R_3,R_3',R_{3,1},\cdots,R_{3,l}\in\ \mathbf{G_{p_3}}$ using exponentiation of X_3. Next, it computes:

$$K_1=Y_1^{\alpha}(Y_1^{a}Y_1^{\sum_{i=1}^{l}\frac{w(i)}{u(i)}(x_i+v(i))}X_1)^{r}R_3 \quad (14)$$

$$K_2=Y_1^{r}R_3' \quad (15)$$

For $i=1,\cdots,l$ the KeyGen algorithm computes:

$$K_{3,i}=Y_1^{\left(\frac{1}{u(i)}(x_i+v(i))\right)r}R_{3,i} \quad (16)$$

At the end, the ***Keygen*** algorithm returns $SK_{\vec{x}}=\left\{K_1,K_2,\{K_{3,i}\}_{i=1}^{l}\right\}$.

Encrypt$(PK,m,\vec{y}=[y_1,\cdots,y_l])$. To encrypt a plaintext $m\in\mathbf{G}_T$, associated with a vector attributes $\vec{y}$. First, the ***Encrypt*** algorithm chooses randomly $s\in\mathbf{Z_N}$ and $R_4,R_4',R_{4,1},\cdots,R_{4,l}\in\ \mathbf{G_{p_4}}$ using exponentiation of X_4. Next, it computes:

$$C_0=m\mathbf{Y}^{s} \quad (17)$$

$$C_1=Y_1^{s}R_4 \quad (18)$$

$$C_2=\left(Y_1^{u}\left(\prod_{i=1}^{l}\mathbf{V}(i)^{-y_i}\right)Q\right)^{s}R_4' \quad (19)$$

For $i=1,\cdots,l$ the ***Encrypt*** algorithm computes:

$$C_{3,i}=\left(\mathbf{U}(i)^{y_i}\mathbf{W}(i)\right)^{s}R_{4,i} \quad (20)$$

At the end, the ***Encrypt*** algorithm returns $CT_{m,\vec{y}}=\left\{C_0,C_1,C_2,\{C_{3,i}\}_{i=1}^{l}\right\}$.

Decrypt$(SK_{\vec{x}},CT_{m,\vec{y}})$. The decryption algorithm returns m if $\langle\vec{x},\vec{y}\rangle=0$ Otherwise, it returns $\perp$.

The correctness of decryption algorithm is shown as follows:

First the decryption algorithm computes $A_1=\hat{\mathbf{e}}(K_1,C_1)$ which yields:

$$A_1=\hat{\mathbf{e}}(Y_1,Y_1)^{\alpha s+rs\left(a+\sum_{i=1}^{d}\frac{w(i)}{u(i)}(x_i+v(i))\right)}\hat{\mathbf{e}}(Y_1,X_1)^{rs} \quad (21)$$

then it computes $A_2=\hat{\mathbf{e}}(K_2,C_2)$ which yields:

$$A_2=\hat{\mathbf{e}}(Y_1,Y_1)^{rs(a-\sum_{i=1}^{d}v(i)y_i)}\hat{\mathbf{e}}(Y_1,X_1)^{rs} \quad (22)$$

and also computes $A_3 = \prod_{i=1}^{l} \hat{e}(K_{3,i}, C_{3,i})$ which yields:

$$A_3 = \hat{e}(Y_1, Y_1)^{rs\left(\langle \vec{x}, \vec{y} \rangle + \sum_{i=1}^{l} v(i)y(i) + \sum_{i=1}^{l}\left(\frac{w(i)}{u(i)}(x(i)+v(i))\right)\right)} \quad (23)$$

At the end the decryption computes:

$$A_0 = \frac{C_0 A_2 A_3}{A_1} = m\,\hat{e}(Y_1, Y_1)^{rs\langle \vec{x}, \vec{y} \rangle} \quad (24)$$

The decryption can recover m from A_0 in Equation 24 only if $\langle \vec{x}, \vec{y} \rangle = 0$ as required for satisfying correctness of an IPE scheme.

Ⅳ. Security of the Proposed IPE Scheme

In this section, we prove the security of our proposed IPE that satisfying full and *attribute hiding* security. Dual encryption system from Security of the proposed IPE scheme used dual encryption system in [14] to prove that our scheme achieves full security definition in indistinguishable game with the presence of chosen-plaintext attacker. While for proving that the proposed IPE scheme achieves *attribute-hiding* the simulation includes that the attacker can not distinguish between the challenge ciphertext associated with the real challenge vector attributes $\vec{y}^*$ and random elements.

4.1 Semifunctional Secret Key and Ciphertext

Semifunctional version of a secret key and a ciphertext are used in proof only to achieve full security definition. A semifunctional secret key and ciphertext behaves similar to a normal version, however the decryption between a semifunctional secret key and a semifunctional ciphertext should be failed. Our proof exploits elements from subgroup G_{p_2} as it is orthogonal to elements from other subgroups (which are used in normal version of secret key and ciphertext).

(1) Semifunctional secret key

A generator Y_2 of G_{p_2} is used to generate semifunctional secret key $SK_{x,sf}^{\rightarrow}$. A semifunctional secret key for a vector attributes $\vec{x} = [x_1, \cdots, x_l]$ is generated as follows:

(1) Call ***Keygen*** algorithm to generate a normal secret key for $\vec{x} = [x_1, \cdots, x_l]$: $SK_{x_k}^{\rightarrow} \leftarrow$ ***Keygen***$(PK, MSK, \vec{x_k})$, where $SK_{\vec{x}} = \left\{K_1', K_2', \{K_{3,i}'\}_{i=1}^{l}\right\}$

(2) Choose $d, \gamma, z_k, z_{k,1} \cdots, z_{k,l}, \omega_1, \cdots, \omega_l \in Z_N$ randomly.

(3) Generate a semifunctional secret key type:

$$SK_{x,sf,1}^{\rightarrow} = \left\{\begin{matrix} K_1 = K_1' Y_2^{d + z_k \gamma}, \\ K_2 = K_2' Y_2^{\gamma}, \\ \left\{K_{3,i} = K_{3,i}' Y_2^{(z_{k,i} + \omega_i)\gamma}\right\}_{i=0}^{l} \end{matrix}\right\} \quad (25)$$

(2) Semifunctional ciphertext

A semifunctional ciphertext $CT_{m,\vec{y},sf}$ for a vector attributes $\vec{y} = [y_1, \cdots, y_l]$ is generated as follows:

(1) Call ***Encrypt*** algorithm to generate a normal ciphertext for $\vec{y} = [y_1, \cdots, y_l]$: $CT_{m,\vec{y}} \leftarrow$ ***Encrypt***$(PK, m, \vec{y})$, where $CT_{m,\vec{y}} = \left\{C_0', C_1', C_2', \{C_{3,i}'\}_{i=1}^{l}\right\}$

(2) Choose $c, v, z_c, z_{c,1}, \cdots, z_{c,l}, \tau_{c,1}, \cdots, \tau_{c,l} \in Z_N$ randomly.

(3) Generate a semifunctional secret key:

$$CT_{m,\vec{y},sf} = \left\{\begin{matrix} C_0', C_1 = C_1' Y_2^{v}, \\ C_2 = C_2' Y_2^{c - z_c v}, \\ \left\{C_{3,i} = C_{3,i}' Y_2^{(z_{c,i} + \tau_i)v}\right\}_{i=1}^{l} \end{matrix}\right\} \quad (26)$$

The decryption between a semifunctional secret key type 1 $SK_{x,sf,1}^{\rightarrow}$ and a ciphertext $CT_{m,\vec{y},sf}$ where $\langle \vec{x}, \vec{y} \rangle = 0$ produces a blinding factor to the plaintext with an element from G_{p_2}:

$$e(Y_2, Y_2)^{(c\gamma - dv) + (\sum_{i=1}^{l}(z_{c,i}z_{k,i} + z_{c,i}\omega_i + z_{k,i}\tau_i + \omega_i\tau_i) - z_c - z_k)\gamma v} \quad (27)$$

If $c\gamma = d\nu$, $z_k = \sum_{i=1}^{l}(z_{c,i}\omega_i + z_{k,i}\tau_i + \omega_i\tau_i) - z_c$ and $\sum_{i=1}^{l} z_{c,i}z_{k,i} = 0$ then we have a nominal semifunctional secret key that can decrypt semi functional ciphertext.

4.2 Security Games

The full security proof used reduction of indistinguishable games. The games are defined as follow:

(1) $Game_{Real}$: this game is the same as security definition given above. The challenger $\boldsymbol{C}$ returns normal secret keys in secret key queries phase and a normal challenge ciphertext in challenge phase.

(2) $Game_0$: this game is the same as $Game_{Real}$ except at challenge phase the challenger $\boldsymbol{C}$ returns a semifunctional ciphertext.

(3) $Game_k$: this game is the same as $Game_0$ except for the first k secret key queries the challenger $\boldsymbol{C}$ returns semifunctional secret keys. While for the rest of secret key queries the challenger returns normal secret keys

(4) $Game_{final_0}$: this game is the same as $Game_{2q}$ where all secret key queries is answered by semifunctional secret key type 2 but in challenge phase this game return a random element form $\mathbb{G}_T$ for C_0 in the challenged ciphertext $CT_{m_b, \vec{y_b}*}$. The $Game_{final,0}$ is used to show *payload hiding*. security of the scheme.

(5) $Game_{final_1}$: this game is the same as $Game_{final_0}$ with addition in challenge phase this game return a random element form $\mathbb{G}_{p_1p_2p_4}$ for C_2 in the challenged ciphertext $CT_{m_b, \vec{y_b}*}$.

(6) $Game_{final_2}$: this game is the same as $Game_{final_1}$ with addition in challenge phase this game return a random element form $\mathbb{G}_{p_1p_2p_4}$ for all $C_{3,i}$ in the challenged ciphertext $CT_{m_b, \vec{y_b}*}$. $Game_{final_2}$ is used to show *attribute hiding* security of the scheme.

4.3 Security Game Reductions

Our proof used a reduction method to show that $Game_{Real}$ is indistinguishable with $Game_{final_1,l}$ in a step-by-step reduction manner. We have to prove the following reductions:

(1) $Game_{Real} \approx Game_0$.
(2) $Game_{k-1} \approx Game_k$
(3) $Game_{2q} \approx Game_{final_0}$
(4) $Game_{final_0} \approx Game_{final_1}$
(5) $Game_{final_1} \approx Game_{final_2}$

Where $Game_A \approx Game_B$ is defined as for any polynomial time attacker $\boldsymbol{A}$, it can not distinguish whether it interacts with $Game_A$ or $Game_B$.

Lemma 1. Suppose there exists a PPT algorithm $\boldsymbol{A}$ that have $\left|\mathrm{Adv}_{\boldsymbol{A}}^{Game_{Real}} - \mathrm{Adv}_{\boldsymbol{A}}^{Game_0}\right| = \epsilon$ then there exists an algorithm $\boldsymbol{B}$ which have advantage ϵ in breaking Assumption 1.

Proof. Given $\{(N = p_1p_2p_3p_4, \mathbb{G}, \mathbb{G}_T, \hat{e}), g_1, g_3, g_4\}$ and T from Assumption 1. We can build algorithm $\boldsymbol{B}$ simulates indistinguishable game with algorithm $\boldsymbol{A}$ as an adversary. The game runs as follows:

- **Setup** phase, algorithm $\boldsymbol{B}$ do as $\boldsymbol{Setup}(\lambda, d)$ algorithm in Section 3 by setting $Y_1 = g_1, X_1 = Y_1^b, X_3 = g_3, X_4 = g_4$ where $b \in \mathbb{Z}_N$ and gives public parameters PK to algorithm $\boldsymbol{A}$.

- **Phase 1 and 2**, algorithm $\boldsymbol{B}$ can answer all secret key queries from algorithm $\boldsymbol{A}$ since it knows the master key PK.
- **Challenge** phase, algorithm $\boldsymbol{B}$ receives $\{(m_0, \overrightarrow{y_0}^*), (m_1, \overrightarrow{y_1}^*)\}$ from algorithm $\boldsymbol{A}$. Then, algorithm $\boldsymbol{B}$ tosses a binary coin $\beta \leftarrow \{0,1\}$ and set the challenge ciphertext $CT_{m_\beta \overrightarrow{y_\beta}^*} = \{C_0, C_1, C_2, \{C_{3,i}\}_{i=1}^{l}\}$ for the challenge vector attributes $\overrightarrow{y_\beta}^* = [y_1, \cdots, y_l]$ as follows :

$$C_0 = m_\beta \hat{e}(Y_1, T)^\alpha \tag{28}$$

$$C_1 = TR_4 \tag{29}$$

$$C_2 = T^{(a - \sum_{i=1}^{l} v(i) y_i + b)} R_4' \tag{30}$$

For $i = 1, \cdots, l$ computes:

$$C_{3,i} = T^{u(i) y_i + w(i)} R_{4,i} \tag{31}$$

where $\alpha, a \in \mathbb{Z}_N$ and polynomials $u(x), v(x), w(x)$ are from the setup phase.

Analysing the game, when $T \in \mathbb{G}_{p_1}$, we set $T = Y_1^s$ then the challenge ciphertext has the same distribution with normal ciphertext. When $T \in \mathbb{G}_{p_1 p_2}$, The algorithm $\boldsymbol{B}$ sets $T = Y_1^s Y_2^v$ and $\tau_i = w(i) \bmod p_2$ which implicitly set $c = av \bmod p_2$, $z_c = \left(\left(\sum_{i=1}^{l} v(i) y_i\right) + b\right) \bmod p_2$ and $z_{c,i} = (u(i) y_i + \tau_i) \bmod p_2$. Even though reuse algorithm $\boldsymbol{B}$ reuses $u(1), \cdots, u(l), v(1), \cdots, v(l), w(1), \cdots, w(l)$ their values in $\bmod p_2$ are uncorrelated with their values in $\bmod p_1$ according to Chinese remainder theorem. Hence when $T \in \mathbb{G}_{p_1 p_2}$ the ciphertext in challenge phase has the same distribution with semifunctional ciphertext.

Therefore algorithm $\boldsymbol{B}$ properly simulates $Game_{Real}$ when $T \in \mathbb{G}_{p_1}$ and $Game_0$ when $T \in \mathbb{G}_{p_1 p_2}$. This completes our proof.

Lemma 2. Suppose there exists a PPT algorithm $\boldsymbol{A}$ that have $\left|\mathrm{Adv}_{\boldsymbol{A}}^{Game_{k-1}} - \mathrm{Adv}_{\boldsymbol{A}}^{Game_k}\right| = \epsilon$ then there exists an algorithm $\boldsymbol{B}$ which have advantage ϵ in breaking Assumption 2.

Proof. Given $\{(N = p_1 p_2 p_3 p_4, \mathbb{G}, \mathbb{G}_T, \hat{e}), g_1, g_3, g_4, A_1 A_2, B_2 B_3\}$ and T from Assumption 2, we can build algorithm $\boldsymbol{B}$ simulates indistinguishable game with algorithm $\boldsymbol{A}$ as an adversary. The game runs as follows:

- **Setup** phase, algorithm $\boldsymbol{B}$ do as $\boldsymbol{Setup}(\lambda, d)$ algorithm in Section 3 by setting $Y_1 = g_1, X_1 = Y_1^b, X_3 = g_3, X_4 = g_4$ where $b \in \mathbb{Z}_N$ and gives public parameters PK to algorithm $\boldsymbol{A}$.
- **Phase 1 and 2**, Algorithm $\boldsymbol{A}$ makes $2q$ key queries: $\overrightarrow{x_1}, \cdots, \overrightarrow{x_{2q}}$. There are 3 cases on how algorithm $\boldsymbol{B}$ answers the key query for i-th query:

  (1) For $i < k$. First, algorithm $\boldsymbol{B}$ chooses $r', w, f, z_1, \cdots, z_l \in \mathbb{Z}_N$, then algorithm $\boldsymbol{B}$ answers the i-th key query for $\overrightarrow{x_i} = [x_1, \cdots, x_l]$ with $SK_{\overrightarrow{x_i}} = \{K_1, K_2, \{K_{3,j}\}_{j=1}^{l}\}$ where

$$K_1 = Y_1^\alpha Y_1^{\left(a + \sum_{i=1}^{l} \frac{w(i)}{u(i)} (x_i + v(i)) + b\right) r'} (B_2 B_3)^{b+w} \tag{32}$$

$$K_2 = Y_1^{r'} X_3^{r_2'} (B_2 B_3)^f \tag{33}$$

$$K_{3,i} = Y_1^{\left(\frac{1}{u(i)} (x_i + v(i))\right) r'} (B_2 B_3)^{(z_i + \frac{v(i)}{u(i)}) f} \tag{34}$$

In this case, algorithm $\boldsymbol{B}$ returns a semifunctional secret key for each query. Let us define $B_2 = Y_2^\phi$, then algorithm $\boldsymbol{B}$ creates a semifunctional secret key by setting $r = r'$, $\gamma = \phi.f \bmod p_2$, $d = \phi.b \bmod p_2$, $z_k = \phi.w/f \bmod p_2$,

$z_{k,i} = \phi.z_i \bmod p_2$. and $\omega_i = \phi.u(i)/v(i) \bmod p_2$.

(2) For $i = k$, algorithm $\boldsymbol{B}$ uses T from assumption to create secret key $SK_{\overrightarrow{x_k}} = \left\{K_1, K_2, \left\{K_{3,j}\right\}_{j=1}^{l}\right\}$ for $\overrightarrow{x_k} = [x_1, \cdots, x_l]$ where

$$K_1 = Y_1^{\alpha} T^{\left(a + \sum_{i=1}^{l} \frac{1}{u(i)}(x_i + v(i)) + b\right)} (B_2 B_3)^{h_1'} \quad (35)$$

$$K_2 = T X_3^{r_2'} \quad (36)$$

$$K_{3,i} = T^{\left(\frac{1}{u(i)}(x_i + v(i))\right)} X_3^{r_{3,j}'} \quad (37)$$

(3) For $i > k$, creates normal secret key for $\overrightarrow{x_k} = [x_1, \cdots, x_l]$. algorithm $\boldsymbol{B}$ can produce normal secret key since it knows the master key *MSK*.

- **Challenge** phase, algorithm $\boldsymbol{B}$ sets the challenge ciphertext $CT_{m_\beta, \overrightarrow{y_\beta}^*} = \left\{C_0, C_1, C_2, \left\{C_{3,i}\right\}_{i=1}^{l}\right\}$ for $\overrightarrow{y_\beta}^* = [y_1, \cdots, y_l]$ as follows :

$$C_0 = m_\beta \hat{\mathbf{e}}(Y_1, A_1 A_2)^{\alpha} \quad (38)$$

$$C_1 = (A_1 A_2) R_4 \quad (39)$$

$$C_2 = (A_1 A_2)^{\left(a - \sum_{i=1}^{l} v(i) y_i + b\right)} R_4' \quad (40)$$

For $i = 1, \cdots, l$ computes:

$$C_{3,i} = (A_1 A_2)^{u(i) y_i + w(i)} R_{4,i} \quad (41)$$

where $\alpha, a \in \mathbf{Z}_N$ and polynomials $u(x), v(x), w(x)$ are from the setup phase.

In this phase, algorithm $\boldsymbol{B}$ creates a semifunctional ciphertext for the challenge vector attributes by setting $Y_1^s Y_2^v = A_1 A_2$, and implicitly sets $Y_2^c = A_2^a$,

$$z_c = -(v(1) y_1 + \cdots + v(l) y_l) + b \bmod p_2,$$

$$z_c = u(i) y_i \bmod p_2 \text{ and } \tau_i = w(i) \bmod p_2$$

- **Guess** phase, at the end $\boldsymbol{A}$ outputs β' as a guess for β. Algorithm $\boldsymbol{B}$ then outputs whatever $\boldsymbol{A}$ outputs.

Analyzing the game, let us consider in secret key query phase in case $i = k$. We have the following two observations: (1) When $T \in \mathrm{G}_{p_1 p_3}$, the algorithm $\boldsymbol{B}$ sets $T = Y_1^r R_3''$ then the secret key has the same distribution with a normal secret key. (2) When $T \in \mathrm{G}_{p_1 p_2 p_3}$, the algorithm $\boldsymbol{B}$ sets $T = Y_1^r Y_2^{\gamma} R_3''$ and implicitly sets $z_k = \sum_{i=1}^{l} \frac{w(i)}{u(i)}(x_i + v(i)) + b \bmod p_2$, $z_{k,i} = \frac{x_i}{u(i)} \bmod p_2$, and $\omega_i = \frac{v(i)}{u(i)} \bmod p_2$. In this case, the k-th secret key has the same distribution with a semifunctional secret key.

Since the restriction for *payload hiding* is none of vector attributes $\overrightarrow{x_k}$ in $\overrightarrow{x_1}, \cdots, \overrightarrow{x_q}$ queried in phase 1 and 2 satisfy $\langle \overrightarrow{x_k}, \overrightarrow{y_0}^* \rangle = \langle \overrightarrow{x_k}, \overrightarrow{y_1}^* \rangle = 0$. then $z_k, z_{k,1}, \cdots, z_{k,l}, z_c, z_{c,1}, \cdots, z_{c,l}$ are independent and randomly distributed. Furthermore, algorithm $\boldsymbol{B}$ can not test itself whether the k-th secret key is a semifunctional or a normal key by doing decryption test using $\overrightarrow{x_k}$ which satisfied $\langle \overrightarrow{x_k}, \overrightarrow{y_0}^* \rangle = \langle \overrightarrow{x_k}, \overrightarrow{y_1}^* \rangle = 0$ because in this case the secret key is a nominal semifunctional or a normal key which both can decrypt the challenge ciphertext.

Thus algorithm $\boldsymbol{B}$ properly simulates $Game_{k-1}$ when $T \in \mathbf{G}_{p_1 p_3}$ and $Game_k$ when $T \in \mathbf{G}_{p_1 p_2 p_3}$. This completes our proof.

Lemma 3. Suppose there exists a PPT algorithm $\boldsymbol{A}$ that have $\left|\mathrm{Adv}_{\boldsymbol{A}}^{Game_{2q}} - \mathrm{Adv}_{\boldsymbol{A}}^{Game_{final_0}}\right| = \epsilon$ then there exists an algorithm $\boldsymbol{B}$ which have advantage ϵ in breaking Assumption 3.

Proof. Algorithm $\boldsymbol{B}$ is given $\{I, g_1, g_2, g_3, g_4, g_1^{\alpha} A_2, g_1^s B_2\}$ where $I = \left(N = p_1 p_2 p_3 p_4, \mathbf{G}, \mathbf{G_T}, \hat{\mathbf{e}}\right)$ and T from

assumption 3. Then, algorithm $\boldsymbol{B}$ plays an indistinguishable game with $\boldsymbol{A}$ as adversary.. The game runs as follows:

- **Setup** phase, algorithm $\boldsymbol{B}$ sets public parameters as follows: First, it chooses randomly $a, \alpha_0, \alpha_1, \cdots, \alpha_d, \beta_0, \beta_1, \cdots, \beta_d, \gamma_0, \cdots, \gamma_d \in \mathbb{Z}_N$, then it creates 3 polynomial functions as in Equation 5, 6 and 7. It sets $Y_1 = g_1, Y_3 = g_3, Y_4 = g_4, X_1 = Y_1^b$ and compute all element in $\boldsymbol{G}_{p_4}$ by exponentiation of Y_4. Algorithm $\boldsymbol{B}$ computes $\mathrm{Y} = \hat{\mathrm{e}}(Y_1, g_1^{\alpha} A_2) = \hat{\mathrm{e}}(Y_1, Y_1)^{\alpha}$, sets public parameters PK as Equation 8 and send public parameters PK to algorithm $\boldsymbol{A}$.
- **Phase 1 and 2**, Algorithm $\boldsymbol{B}$ always answer secret key query with a semifunctional secret key. After receiving i-th vector attributes $\overrightarrow{x_i} = [x_1, \cdots, x_l]$, algorithm $\boldsymbol{B}$ chooses $r', f, w, z_1, \cdots, z_l, r_1, r_2, r_{3,1}, \cdots, r_{3,l} \in \mathbb{Z}_N$ and sets

$$K_1 = \left(g_1^{\alpha} A_2\right) g_2^{w} Y_1^{\left(a + \sum_{i=1}^{l} \frac{w(i)}{u(i)}(x_i + v(i)) + b\right) r'} X_3^{r_1} \quad (42)$$

$$K_2 = Y_1^{r'} (g_2)^{f} X^{r_2} \quad (43)$$

$$K_{3,i} = Y_1^{\left(\frac{1}{u(i)}(x_i + v(i))\right) r'} g_2^{(z_i + \frac{v(i)}{u(i)}) f} (X_3)^{r_{3,i}} \quad (44)$$

Algorithm $\boldsymbol{B}$ returns for all query with a semifunctional secret key by setting $A_2 = Y_2^d$, $\gamma = f \bmod p_2$. $z_k = w/f \bmod p_2$, $z_{k,i} = z_i \bmod p_2$. and $\omega_i = u(i)/v(i) \bmod p_2$.

- **Challenge** phase, algorithm $\boldsymbol{B}$ sets the challenge ciphertext $CT_{m_\beta, \overrightarrow{y_\beta}*} = \left\{C_0, C_1, C_2, \{C_{3,i}\}_{i=1}^{l}\right\}$ for $\overrightarrow{y_\beta}* = [y_1, \cdots, y_l]$ as follows:

$$C_0 = m_\beta T \quad (45)$$

$$C_1 = (g_1^s B_2) R_4 \quad (46)$$

$$C_2 = \left(g_1^s B_2\right)^{\left(a - \sum_{i=1}^{l} v(i) y_i + b\right)} R_4' \quad (47)$$

For $i = 1, \cdots, l$ computes

$$C_{3,i} = (g_1^s B_2)^{u(i) y_i + w(i)} R_{4,i} \quad (48)$$

where $a \in \mathbb{Z}_N$ and polynomials $u(x), v(x), w(x)$ are from the setup phase.

By writing $B_2 = g_2^v$, algorithm $\boldsymbol{B}$ returns a ciphertext that elements $C_1, C_2, \{C_{3,i}\}_{i=1}^{l}$ has the same distribution as a semifunctional ciphertext. Algorithm $\boldsymbol{B}$ implicitly sets $c = v.a \bmod p_2$, $z_c = -(v(1) y_1 + \cdots + v(l) y_l) + b \bmod p_2$, $z_{c,i} = u(i) y_i \bmod p_2$, and $\tau_i = w(i) \bmod p_2$.

- **Guess** phase, at the end $\boldsymbol{A}$ outputs β' as a guess for β. Algorithm $\boldsymbol{B}$ then outputs whatever $\boldsymbol{A}$ outputs.

We have two observations: (1) when $T = \hat{\mathrm{e}}(g_1, g_1)^{\alpha s}$ then the challenge ciphertext has the same distribution with a semifunctional ciphertext. (2) when $T \in \mathrm{G}_T$ then the challenge ciphertext has the same distribution with a semifunctional ciphertext except C_0 is a random element from $\boldsymbol{G}_T$.

Thus algorithm $\boldsymbol{B}$ properly simulates $Game_{2q}$ when $T = \hat{\mathrm{e}}(g_1, g_1)^{\alpha s}$ and $Game_{final_0}$ when $T \in \mathrm{G}_T$. This completes our proof. □

Lemma 4. Suppose there exists a PPT algorithm $\boldsymbol{A}$ that have $\left|\mathrm{Adv}_{\mathbf{A}}^{Game_{final0}} - \mathrm{Adv}_{\mathbf{A}}^{Game_{final1}}\right| = \epsilon$ then there exists an algorithm $\boldsymbol{B}$ which have advantage ϵ in breaking Assumption 4.

Proof. Algorithm $\boldsymbol{B}$ is given $\left\{I, g_1, g_2, g_3, g_4, U, U^s A_{24}, U^r, A_1 A_4, A_1^r A_2, g_1^r B_2, g_1^s B_{24}\right\}$ where $I = \left(N = p_1 p_2 p_3 p_4, \mathrm{G}, \mathrm{G}_T, \hat{\mathrm{e}}\right)$ and T from Assumption 4. Then, algorithm $\boldsymbol{B}$ plays an indistinguishable game with $\boldsymbol{A}$ as adversary.. The game runs as follows:

- **Setup** phase, algorithm $\boldsymbol{B}$ sets public parameters as follows: First, it chooses randomly $a, \alpha, \alpha_0, \alpha_1, \cdots, \alpha_d, \beta_0, \beta_1, \cdots, \beta_d, \gamma_0, \cdots, \gamma_d \in \mathbb{Z}_N$, then it creates 3 polynomial functions as in Equation 5, 6 and 7. It sets $Y_1 = g_1, Y_3 = g_3, Y_4 = g_4$ and compute

all element in G_{p_4} by exponentiation of Y_4. Algorithm B creates three polynomials and sets three public functions . At the end, algorithm B sets public parameters PK as

$$PK = \left\{ \begin{array}{l} Y_1, Y_4, Q = A_1 A_4, Y_1^a, \\ U^{u(0)} U_{4,0}, \cdots, U^{u(d)} U_{4,d}, \\ U^{v(0)} V_{4,0}, \cdots, U^{v(d)} V_{4,d}, \\ U^{w(0)} W_{4,0}, \cdots, U^{w(d)} W_{4,d}, \\ \mathrm{Y} = \hat{\mathrm{e}}(\mathrm{Y}_1, \mathrm{Y}_1)^{\alpha} \end{array} \right\} \quad (49)$$

and send public parameters PK to algorithm A.

- **Phase 1 and 2**, Algorithm B should create a semifunctional secret key for each key query for vector attributes $\overrightarrow{x_i} = [x_1, \cdots, x_l]$. For each query algorithm B chooses $r', z_1', \cdots, z_l', r_1, r_2, r_{3,1}, \cdots, r_{3,l} \in \mathbf{Z}_N$ and sets the secret key as $SK_{\overrightarrow{x_i}} = \left\{K_1, K_2, \{K_{3,j}\}_{j=1}^{l}\right\}$ where

$$K_1 = Y_1^{\alpha} \left(g_1^{\hat{r}} B_2\right)^{ar'} \left(U^{\hat{r}}\right)^{\left(\sum_{i=1}^{l} \frac{w(i)}{u(i)} (x_i + v(i))\right) r'} \left(A_1^{\hat{r}} A_2\right)^{r'} X_3^{r_1} \quad (50)$$

$$K_2 = \left(g_1^{\hat{r}} B_2\right)^{r'} X_3^{r_2} \quad (51)$$

$$K_{3,j} = \left(U^{\hat{r}}\right)^{\left(\frac{1}{u(j)} (x_j + v(j))\right) r'} Y_2^{z_j'} (X_3)^{r_{3,i}} \quad (52)$$

This implicitly sets the randomness $r = \hat{r} r'$. According to Equation 50, 51 and 52 , all elements in $SK_{\overrightarrow{x_i}} = \left\{K_1, K_2, \{K_{3,j}\}_{j=1}^{l}\right\}$ contains an element in G_{p_2} it show that the queried secret key has the same distribution with a semifunctional secret key.

- **Challenge** phase, at some points, algorithm A sends B two tuples $\left(\overrightarrow{y_0}^*, m_0\right)$ and $\left(\overrightarrow{y_1}^*, m_1\right)$. Algorithm B throws a binary dice $\beta \leftarrow \{0,1\}$ and sets the challenge ciphertext $CT_{m_\beta, \overrightarrow{y_\beta}^*} = \left\{C_0, C_1, C_2, \{C_{3,i}\}_{i=1}^{l}\right\}$ for $\overrightarrow{y_\beta}^* = [y_1, \cdots, y_l]$ where C_0 is a random from G_T and compute the other elements as follows:

$$C_1 = g_1^s B_{24} \quad (53)$$

$$C_2 = \left(g_1^s B_{24}\right)^a \left(U^s A_{24}\right)^{-(v(i) y_i)} T \quad (54)$$

For $i = 1, \cdots, l$ computes:

$$C_{3,i} = \left(U^s A_{24}\right)^{u(i) y_i + w(i)} \quad (55)$$

where $a \in \mathbf{Z}_N$ and polynomials $u(x), v(x), w(x)$ are from the setup phase. The algorithm B implicitly sets $c, v, z_c, z_{c,1}, \cdots, z_{c,l}, \tau_1, \cdots, \tau_l$ to random values.

- **Guess.** Algorithm B then outputs whatever A outputs.

We have the following analysis: when $T = A_1^s D_{24}$ then C_2 in $CT_{m_\beta, \overrightarrow{y_\beta}^*} = \left\{C_0, C_1, C_2, \{C_{3,i}\}_{i=1}^{l}\right\}$ has the same distribution with C_2 element in a semifunctional ciphertext, since the algorithm B implicitly sets $X_1 = A_1$. Otherwise, when $T \in \mathbf{G}_{p_1 p_2 p_4}$ is a random element then C_2 in $CT_{m_\beta, \overrightarrow{y_\beta}^*} = \left\{C_0, C_1, C_2, \{C_{3,i}\}_{i=1}^{l}\right\}$ is a random element from $\mathbf{G}_{p_1 p_2 p_4}$.

Therefore, when $T = A_1^s D_{24}$, algorithm B properly simulated $Game_{final_0}$ and when $T \in \mathbf{G}_{p_1 p_2 p_4}$ is a random element from $\mathbf{G}_{p_1 p_2 p_4}$, algorithm B properly simulated $Game_{final_1}$. This completes our proof.

Lemma 5. Suppose there exists a PPT algorithm A that have $\left|\mathrm{Adv}_{\mathbf{A}}^{Game_{final1,k-1}} - \mathrm{Adv}_{\mathbf{A}}^{Game_{final1,k}}\right| = \epsilon$ then there exists an algorithm B which have advantage ϵ in breaking Assumption 4.

Proof. The proof is similar to previous proof except when computing PK and $CT_{m_\beta, \overrightarrow{y_\beta}^*} = \left\{C_0, C_1, C_2, \{C_{3,i}\}_{i=1}^{l}\right\}$. In setup phase the algorithm B selects randomly $w_c \in \mathbf{Z}_N$, creates three polynomials $u'(x), v'(x), w'(x)$ as in Equation 5, 6, and 7 then it sets PK as

표 2. IPE 방식들의 비교
Table 2. Comparison of IPE schemes.

	KSW08 [7]	LOS10 [10]	AL10 [9]	OT10 [11]	OT12a [12]	OT12b [13]	Proposed IPE
$\|PK\|$	$O(n)\|\mathbb{G}\|$	$O(n^2)\|\mathbb{G}\|$	$O(n)\|\mathbb{G}\|$	$O(n^2)\|\mathbb{G}\|$	$O(n)\|\mathbb{G}\|$	$O(1)\|\mathbb{G}\|$	$O(1)\|\mathbb{G}\|$
$\|SK\|$	$(2n+1)$ $\|\mathbb{G}\|$	$(2n+3)$ $\|\mathbb{G}\|$	$(n+7)$ $\|\mathbb{G}\|$	$(3n+2)$ $\|\mathbb{G}\|$	$11\|\mathbb{G}\|$	$(15n+5)$ $\|\mathbb{G}\|$	$(n+2)$ $\|\mathbb{G}\|$
$\|CT\|$	$(2n+1)$ $\|\mathbb{G}\|$ $+\|\mathbb{G}_T\|$	$(2n+3)$ $\|\mathbb{G}\|$ $+\|\mathbb{G}_T\|$	$7\|\mathbb{G}\|$ $+\|\mathbb{G}_T\|$	$(3n+2)$ $\|\mathbb{G}\|$ $+\|\mathbb{G}_T\|$	$(5n+1)$ $\|\mathbb{G}\|$ $+\|\mathbb{G}_T\|$	$(15n+5)$ $\|\mathbb{G}\|$ $+\|\mathbb{G}_T\|$	$(n+2)$ $\|\mathbb{G}\|$ $+\|\mathbb{G}_T\|$
Unbounded/Bounded	bounded	bounded	bounded	bounded	bounded	unbounded	unbounded
Fully/Selective	selective	fully	fully	fully	fully	fully	fully
Fully(Weakly)-AH/ PH	fully-AH	weakly-AH	PH	weakly-AH	fully-AH	fully-AH	fully-AH
Assumption	GSD	n-eDDH	DLIN/ DDH	DLIN	DLIN	DLIN	GSD

$$PK=\left\{\begin{array}{l} Y_1, Y_4, Q=A_1A_4, Y_1^a, \\ U^{u'(0)}U_{4,0},\cdots,U^{u'(d)}U_{4,d}, \\ U^{v'(0)}V_{4,0},\cdots,U^{v'(d)}V_{4,d}, \\ W_0=((A_1A_4)^{w_c})^{w'(0)},\cdots, \\ W_d=((A_1A_4)^{w_c})^{w'(d)}W_{4,d}, \\ \mathrm{Y}=\hat{\mathrm{e}}(\mathrm{Y}_1,\mathrm{Y}_1)^{\alpha} \end{array}\right\} \quad (56)$$

In challenge phase the algorithm $\boldsymbol{B}$ sets the challenge ciphertext $CT_{m_\beta,\overrightarrow{y_\beta}*}=\left\{C_0,C_1,C_2,\{C_{3,i}\}_{i=1}^{l}\right\}$ for $\overrightarrow{y_\beta}*=[y_1,\cdots,y_l]$ where C_0 is a random from $\boldsymbol{G_T}$ and

$$C_1=g_1^sB_{24} \quad (57)$$

$$C_2=\left(g_1^sB_{24}\right)^a\left(U^sA_{24}\right)^{-(v'(i)y_i)}Z_1Z_{24} \quad (58)$$

Where $Z_1Z_{24}\in G_{p_1p_2p_4}$ is a random element from $T\in\mathbb{G}_{p_1p_2p_4}$. and For $i=1,\cdots,l$:

$$C_{3,i}=(U^sA_{24})^{u'(i)y_i}T^{w_cw'(i)} \quad (59)$$

We have the following analysis: when $T=A_1^sD_{24}$ then all $C_{3,i}$ in $CT_{m_\beta,\overrightarrow{y_\beta}*}=\left\{C_0,C_1,C_2,\{C_{3,i}\}_{i=1}^{l}\right\}$ has the same distribution with $C_{3,i}$ element in a semifunctional ciphertext. By writing $A_1=Y_1^\phi$ then the algorithm $\boldsymbol{B}$ implicitly sets the polynomial $w(x)=\phi w_cw'(x)$. Otherwise, when $T\in\mathbb{G}_{p_1p_2p_4}$ is a random element then all $C_{3,i}$ in $CT_{m_\beta,\overrightarrow{y_\beta}*}=\left\{C_0,C_1,C_2,\{C_{3,i}\}_{i=1}^{l}\right\}$ is a random element from $\mathbb{G}_{p_1p_2p_4}$.

Therefore, when $T=A_1^sD_{24}$, algorithm $\boldsymbol{B}$ properly simulated $Game_{final_1}$ and when $T\in\mathbb{G}_{p_1p_2p_4}$ is a random element from $\mathbb{G}_{p_1p_2p_4}$, algorithm $\boldsymbol{B}$ properly simulated $Game_{final_2}$. This completes our proof.

Theorem 1. If assumption 1,2,3 and 4 are hold then our unbounded inner product encryption scheme is fully secure and attribute hiding.

Proof. We have proved by **Lemma** 1, 2, 3, 4, and 5 that $Game_{real}$ is indistinguishable from $Game_{final_2}$ if only assumption 1,2,3 and 4 are hold. Since in challenge phase is information theoretically hidden from the attacker $\boldsymbol{A}$ then $\boldsymbol{A}$ can obtain no advantage in breaking our unbounded IPE scheme in full security sense.

V. Performance

We compare the proposed IPE scheme to other existing IPE schemes(KSW08 in [7], LOS10 in [9], AL10 [10], OT10 in [11], OT12a in [12], and OT12b in [13]). Table 2 summarizes the performance comparison between the proposed IPE scheme and

existing schemes.

The proposed IPE scheme achieves fully attribute hiding security. To the best our knowledge, only OT12a[12] and OT12b[13] achieve the same security level. Furthermore, the proposed IPE does not bound the attribute vector in a secret key or ciphertext (in sense of the size of the attribute vector) where most existing IPE schemes except OT12b[13] bounds the attribute vector with the size of public parameters. The size of public parameters of the proposed IPE scheme is constant like OT12b[13] while others grow linearly/polynomially with the size of attributes vector (that bound the size of attributes vector in a secret key/ciphertext). Therefore only the proposed IPE and OT12b that presents unbounded and fully attribute hiding security IPE scheme.

It is interesting to compare the secret key/ciphertext size of the proposed IPE scheme and OT12b. The size of a secret key/ciphertext in the proposed IPE scheme is smaller than in OT12b: $(n+2)|\mathbb{G}|$ compare to $(15n+5)|\mathbb{G}|$. Therefore, the memory requirement for storing a secret key/ciphertext of the proposed IPE scheme is more efficient than in that OT12b[13]. In decryption, the number of pairing is the same as the size of a secret key/ciphertext. Thus, the number of pairing in decryption of the proposed IPE scheme is smaller than in that OT12b[13]. One downside of the proposed IPE scheme is that it uses composite order bilinear groups which known has more complexity in pairing computation than prime order bilinear groups. However, we can use transformation technique in [19] to transform the proposed IPE scheme to prime order bilinear groups.

VI. Conclusion

The proposed IPE scheme achieves full and attribute hiding definition based on variant of decisional sub group problems assumptions using dual encryption framework. Furthermore, we relax the boundedness between public parameters and the size of attribute vectors used in key generation/encryption. The proposed IPE scheme allows arbitrary vector attributes length to be used in key generation or encryption algorithm. Compared to the existing IPE schemes, the proposed IPE scheme achieves more advanced security definition than most existing IPE schemes and also has smaller size for a secret key/ciphertext than the same security existing IPE scheme.

REFERENCES

[1] Dongyoung Koo, Junbeom Hur, and Hyunsoo Yoon, “Secure and efficient data retrieval over encrypted data using attribute-based encryption in cloud storage,” *Computers & Electrical Engineering*, vol. 39, no. 1, pp. 34-46, 2013.

[2] Boneh, D. and Franklin, M, “Identity-Based Encryption from the Weil Pairing,” *SIAM Journal on Computing*, vol. 32, no. 3, pp. 586-615, 2003.

[3] Boyen, Xavier and Brent Waters, “Anonymous hierarchical identity-based encryption (without random oracles).” *CRYPTO2006*, pp. 290-307, 2006.

[4] Park, J. H., and Lee D. H.. “A hidden vector encryption scheme with constant-size tokens and pairing computations,” *IEICE transactions on fundamentals of electronics, communications and computer sciences*, vol. 93, no. 9, pp. 1620-1631, 2010.

[5] Park, J. H., Lee, K., Susilo, W., and Lee, D. H., “Fully secure hidden vector encryption under standard assumptions,” *Information Sciences,* vol. 232, pp. 188-207, 2013.

[6] De Caro, Angelo, Vincenzo Iovino, and Giuseppe Persiano. “Fully secure hidden vector encryption,” *Pairing-Based Cryptography - Pairing 2012*. pp. 102-121, 2013.

[7] Katz, Jonathan, Amit Sahai, and Brent Waters. “Predicate encryption supporting disjunctions, polynomial equations, and inner products,” *EUROCRYPT2008*, pp. 146-162, 2008.

[8] Agrawal, Shweta, David Mandell Freeman, and Vinod Vaikuntanathan. “Functional encryption for inner product predicates from learning with

errors," *ASIACRYPT2011*, pp. 21-40, 2011.
[9] Lewko, Allison, et al. "Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption," *EUROCRYPT2010*, pp. 62-91, 2010.
[10] Attrapadung, Nuttapong, and Benoît Libert. "Functional encryption for inner product: Achieving constant-size ciphertexts with adaptive security or support for negation," *Public Key Cryptography - PKC 2010*, pp 384-402, 2010.
[11] Okamoto, Tatsuaki, and Katsuyuki Takashima. "Fully secure functional encryption with general relations from the decisional linear assumption," *CRYPTO2010*, pp. 191-208, 2010.
[12] Okamoto, Tatsuaki, and Katsuyuki Takashima. "Adaptively attribute-hiding (hierarchical) inner product encryption," *EUROCRYPT2012*, pp. 591-608, 2012.
[13] Okamoto, Tatsuaki, and Katsuyuki Takashima. "Fully secure unbounded inner-product and attribute-based encryption," *ASIACRYPT2012*, pp. 349-366, 2012.
[14] Okamoto, Tatsuaki and, Katsuyuki Takashima, "Efficient (Hierarchical) Inner-Product Encryption Tightly Reduced from the Decisional Linear Assumption," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E96-A, no. 1, pp. 42-52, 2013.
[15] Boneh, Dan, et al. "Chosen-ciphertext security from identity-based encryption," SIAM Journal on Computing, vol. 36, no. 5, pp. 1301-1328, 2006.
[16] Waters, Brent. "Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions," *CRYPTO2009*, pp. 619-636, 2009.
[17] Lewko, Allison, and Brent Waters. "Unbounded HIBE and attribute-based encryption," *EUROCRYPT2011*, pp. 547-567, 2011.
[18] De Caro, Angelo, Vincenzo Iovino, and Giuseppe Persiano. "Fully secure anonymous hibe and secret-key anonymous ibe with short ciphertexts," *Pairing-Based Cryptography-Pairing 2010*, pp. 347-366, 2010.
[19] Freeman, David Mandell. "Converting pairing-based cryptosystems from composite-order groups to prime-order groups," *EUROCRYPT2010*, pp. 44-61, 2010.

저 자 소 개

리프키 사디킨(정회원)
1999년 Gadjah Mada 대학교 전기공학과 학사
2004년 Indonesia 대학교 컴퓨터공학과 석사
2009년～현재 경북대학교 전자전기컴퓨터학부 박사과정
<주관심분야 : 정보보호, 네트워크보안>

박 영 호(정회원)-교신저자
1989년 경북대학교 전자공학과 학사
1991년 경북대학교 전자공학과 석사
1995년 경북대학교 전자공학과 박사
1996년～2008년 상주대학교 전자전기공학부 교수
2003년～2004년 Oregon State Univ. 방문교수
2008년～현재 경북대학교 산업전자공학과 교수
<주관심분야 : 정보보호, 네트워크보안, 모바일 컴퓨팅>