ICIC Express Letters
Volume **10**, Number **8**, August **2016**
ICIC International ©2016 ISSN 1881-803X
pp. **2019–2025**

NEW FULLY SECURE PREDICATE ONLY ENCRYPTION SCHEME SUPPORTING INNER PRODUCT

Yu Zhang and Songfeng Lu
School of Computer Science and Technology
Huazhong University of Science and Technology
No. 1037, Luoyu Road, Wuhan 430074, P. R. China
willow1223@126.com; lusongfeng@hotmail.com

Received December 2015; accepted March 2016

Abstract. *In a predicate encryption scheme, a secret key associated with a predicate f can decrypt a ciphertext corresponding to an attribute I when $f(I) = 1$. The predicate encryption scheme which supports inner product enables more complex evaluation on CNF/DNF formulae. Recently, a fully secure predicate encryption scheme supporting inner product was presented by Okamoto et al. When converting this scheme into a searchable encryption scheme, it should be found that the space and time complexity in this scheme can be improved. According to this issue, we propose a new fully secure predicate encryption scheme supporting inner product. Based on our scheme, a more efficient searchable encryption scheme can be created with better space and time complexity.*
Keywords: Public key system, Pairing-based cryptography, Predicate encryption, Fully security

1. **Introduction.** In the predicate encryption supporting inner product (IPE) scheme, each ciphertext associated with an attribute vector $\vec{x}$ can be decrypted by secret keys corresponding to predicate vector $\vec{v}$ if and only if $\vec{v} \cdot \vec{x} = 0$. The first IPE scheme has been proposed in [1]; however, they were proven in the selective security model. Recently, fully secure PE was proposed by Okamoto and Takashima [2]. The IPE scheme can be easily changed to be a searchable encryption scheme [5]. When we obtain a searchable encryption scheme by making use of an IPE scheme, the IPE scheme should be ensure that the attribute is hiding, and the message will be neglected since the message should be encrypted by applying standard public key encryption system, such as RSA. The IPE scheme which hides the attribute and sets message as 1 is called predicate-only IPE introduced in [1]. In this paper, we propose a new fully secure predicate-only IPE scheme. Compared with the searchable encryption scheme based on the previous scheme, the efficiency of the searchable encryption scheme based on our scheme is improved.

This paper is organized as follows. In Section 2, we give a brief introduction of bilinear groups and state the complexity assumptions. In Section 3, we will propose our scheme and the security proof of our scheme, and give a comparison to show the novelty of our scheme. The conclusion is presented in Section 4.

2. Preliminaries.

2.1. Composite order bilinear groups and complexity assumptions.
Composite order bilinear groups were first used in cryptographic construction in [3]. We use groups whose order N is a product of four (distinct) prime and a generator $\mathfrak{g}$ which takes as input a security parameter 1^n and outputs a description $I = (p_1, p_2, p_3, p_4, G, G_T, \hat{e})$, where p_1, p_2, p_3, p_4 are distinct primes, G and G_T are cyclic groups of order $N = p_1p_2p_3p_4$, and $\hat{e} : G \times G \to G_T$ is a non-degenerate bilinear map such that:
1) Bilinear: $\hat{e}\left(g^a, h^b\right) = \hat{e}(g, h)^{ab}$, where $g, h \in G$ and $a, b \in Z_N$;

2) Non-degenerate: If g is a generator of G then $\hat{e}(g,g)$ is a generator of G_T;
3) Computable: There is an efficient algorithm to compute $\hat{e}(g,h)$, for any $g,h \in G$.

We further require that the group operations in G and G_T, as well as the bilinear map $\hat{e}$, are computable in deterministic polynomial time with respect to n. Furthermore, we assume that the descriptions of G and G_T include generators of G and G_T, respectively. For $S \subseteq \{1,2,3,4\}$, we denote by $G_{\prod_{i\in S} p_i}$ the subgroup of order $\prod_{i\in S} p_i$. Suppose that $h_1 \in G_{\prod_{i\in S_1} p_i}$ and $h_2 \in G_{\prod_{i\in S_2} p_i}$, where $S_1, S_2 \subseteq \{1,2,3,4\}$. It is easy to verify that $\hat{e}(h_1,h_2) = 1$ if $gcd\left(\prod_{i\in S_1} p_i \times \prod_{i\in S_2} p_i | N^2, N\right) = N$. This is called the orthogonality property and is a crucial tool in our construction.

For proving the security of our construction, a General Subgroup Decision (GSD) complexity assumption presented in [3] is given as follows:

GSD Assumption. Let $S_0, S_1, S_2, \dots, S_k$ be non-empty subset of $\{1,2,3,4\}$ such that for each $j \in [2,k]$, either $S_j \cap S_0$, $S_j \cap S_1$ are both empty or $S_j \cap S_0$, $S_j \cap S_1$ are both not empty. Given a group generator $\mathfrak{g}$, we define the following distribution:

$$\mathbb{G} = (N = p_1p_2p_3p_4, G, G_T, \hat{e}) \overset{R}{\leftarrow} \mathfrak{g}, \quad T_0 \overset{R}{\leftarrow} G_{\prod_{i\in S_0} p_i}, \quad T_1 \overset{R}{\leftarrow} G_{\prod_{i\in S_1} p_i},$$

$$Z_2 \overset{R}{\leftarrow} G_{\prod_{i\in S_2} p_i}, \dots, Z_k \overset{R}{\leftarrow} G_{\prod_{i\in S_k} p_i}, \quad D = (\mathbb{G}, Z_2, Z_3, \dots, Z_k).$$

We define the advantage of an algorithm A in breaking GSD Assumption to be:

$$Adv1_{\mathfrak{g},A}(n) = \Pr[A(D,T_0) = 1] - \Pr[A(D,T_1) = 1]. \tag{1}$$

Definition 2.1. *For all probabilistic polynomial-time algorithms A, if $Adv1_{\mathfrak{g},A}(n)$ is a negligible function of n, then we can say that GSD Assumption holds for generator $\mathfrak{g}$.*

3. **Proposed IPE Scheme.**

3.1. **Construction.**

Based on the IPE model introduced in [1], our IPE scheme works as follows.

Setup. Choosing a bilinear group G of order $N = p_1p_2p_3p_4$ (where p_1, p_2, p_3, p_4 are distinct primes) and $\alpha_i, \beta_i \in Z_N$, $U_1, A_1 \in G_{p_1}$ and A_{4i}, B_{4i}, U_4 and $g_4 \in G_{p_4}$ where $i \in [1,n]$, the public key (pk) is published as:

$$pk = \left\{N, U_1U_4, {A_1}^{\beta_i}A_{4i}, {A_1}^{\alpha_i}B_{4i}, g_4\right\}.$$

The master secret keys is $msk = \{\alpha_i, \beta_i, U_1, A_1, g_3\}$ where g_3 is a generator of G_{p_3}.

Encrypt($\vec{x}, pk$). Choosing $n+2$ random elements s, $d_1, d_2, \dots, d_n, d_{n+1} \in Z_N$, for a vector $\vec{x} = \{x_1, x_2, \dots, x_n\}$, the ciphertext $C = (C_{11}, C_{12}, \dots, C_{1n}, C_2)$ is created as:

$$C_{1i} = \left({A_1}^{\beta_i}A_{4i}\right)^{sx_i} \times ({A_1}^{\alpha_i}B_{4i})^s \times {g_4}^{d_i} = {A_1}^{s(\beta_i x_i + \alpha_i)}C_{4i}, \;\; C_2 = (U_1U_4)^s \times g_4^{d_{n+1}} = U_1^s C_4.$$

In ciphertext C, $C_{4i} = A_{4i}^{sx_i}B_{4i}^s {g_4}^{d_i}$ for each $i \in [1,n]$ and $C_4 = g_4^{d_{n+1}}U_4^s$.

KeyGen($\vec{v}, msk$). The key generation algorithm chooses $r \in Z_N$ and generates random elements R_{3i}, R_3 by using g_{p_3} and raising it to the random exponents modulo N. For a vector $\vec{v} = \{v_1, v_2, \dots, v_n\}$, the key $sk_{\vec{v}} = (K_{11}, K_{12}, \dots, K_{1n}, K_2)$ is created as:

$$K_{1i} = {U_1}^{r\frac{v_i}{\beta_i}}R_{3i}, \quad K_2 = {A_1}^{r\sum_{i=1}^n \frac{\alpha_i v_i}{\beta_i}}R_3,$$

where $i \in [1,n]$.

Decryption. The algorithm computes $d = \frac{\hat{e}(C_2,K_2)}{\prod_{i=1}^n \hat{e}(C_{1i},K_{1i})}$. If $\vec{x}\cdot\vec{v} = 0$, then $d = 1$.

Correctness. Let C and $sk_{\vec{v}}$ be as the above, then

$$d = \frac{\hat{e}(C_2,K_2)}{\prod_{i=1}^n \hat{e}(C_{1i},K_{1i})} = \frac{\hat{e}\left({U_1}^s C_4, {A_1}^{r\sum_{i=1}^n \frac{\alpha_i v_i}{\beta_i}}R_3\right)}{\prod_{i=1}^n \hat{e}\left({A_1}^{s(\beta_i x_i+\alpha_i)}C_{4i}, {U_1}^{r\frac{v_i}{\beta_i}}R_{3i}\right)} = \frac{1}{\hat{e}(A_1,U_1)^{rs\sum_{i=1}^n x_i v_i}}.$$

Obviously, if $\vec{x}\cdot\vec{v} = 0$, then $d = 1$.

3.2. **Security.** To prove the security of our IPE system, according to the dual system encryption introduced in [6], we will introduce a special key which we call it *s-key* and a special ciphertext which we call it *s-ciphertext* in our scheme. These will not be applied in the real IPE system, but they will be used in the proof.

s-key: Let g_2 be a generator of the subgroup G_{p_2}. An *s-key* is created as follows. A normal key $K'_{11}, K'_{12}, \ldots, K'_{1n}, K'_2$ is constructed by the *keyGen* algorithm. Choosing a random element $\xi \in Z_N$ and setting $z_{ki} = u\frac{v_i}{\beta_i}$ for each $i \in [1,n]$ and $z_k = a\sum_{i=1}^{n}\frac{\alpha_i v_i}{\beta_i}$, K_{1i} is set to be $K'_{1i}g_2^{\xi z_{ki}}$ for each $i \in [1,n]$ and K_2 is set to be $K'_2 g_2^{\xi z_k}$, where $\vec{v} = \{v_1, v_2, \ldots, v_n\}$ is the vector used in constructing normal key and a, u, α_i, β_i are required from $msk = \{A_1 = g_1^a, U_1 = g_1^u, \alpha_i, \beta_i, g_3\}$. The *s-key* is $\{K_{11}, K_{12}, \ldots, K_{1n}, K_2\}$.

s-ciphertext: Let g_2 be a generator of the subgroup G_{p_2}. An *s-ciphertext* is created as follows. For a vector $\vec{x}$, a normal ciphertext $C'_0, C'_{11}, C'_{12}, \ldots, C'_{1n}, C'_2$ is constructed by the encryption algorithm. Given two random elements $s_0, s_1 \in Z_N$, for two distinct elements $\theta_0, \theta_1 \in \{s_0, s_1, 0\}$, suppose that $\vec{x}^{(0)}$ and $\vec{x}^{(1)}$ are two challenge vectors and $msk = \{A_1 = g_1^a, U_1 = g_1^u, \alpha_i, \beta_i, g_3\}$, we compute $z_{ci} = \theta_0 a\left(\beta_i x_i^{(0)} + \alpha_i\right) + \theta_1 a\left(\beta_i x_i^{(1)} + \alpha_i\right)$ for each $i \in [1,n]$ and $z_c = u(\theta_0 + \theta_1)$. Let $C_{1i} = C'_{1i}g_2^{z_{ci}}$ for each $i \in [1,n]$ and $C_2 = C'_2 g_2^{z_c}$. The *s-ciphertext* is $\{C_0, C_{11}, C_{12}, \ldots, C_{1n}, C_2\}$.

Obviously, for a vector $\vec{v}$ which is orthogonal to the two challenge vectors, it must have $\sum_{i=1}^{n} z_{ci}z_{ki} = z_c z_k \bmod N$ where $z_{k1}, z_{k2}, \ldots, z_{kn}, z_k$ are chosen from the *s-key* of the vector $\vec{v}$. Then the decryption algorithm can still work.

The security proof relies on GSD Assumption. We will prove security by using a hybrid method which uses a sequence of games. These games are described as follows.

1) $Game_{Real}$. This game is the real security game which is introduced in [1].
2) $Game_{Restricted}$. This game is the same as the real game except that the attacker can not ask for keys for vector $\vec{v} = (v_1, v_2, \ldots, v_n)$ satisfying $\vec{x}\cdot\vec{v} = \sum_{i=1}^{n} x_i v_i = 0 \bmod p_2$, where $\vec{x} = (x_1, x_2, \ldots, x_n)$ is one of the challenge vectors. We will retain this stronger restriction throughout the subsequent games.
3) $Game_k$. For each $k \in [0,q]$, we define $Game_k$ which is similar to $Game_{Restricted}$ except that the ciphertext given to A is *s-ciphertext* and the first k keys are *s-key*. The rest keys are normal. In $Game_0$, all the keys given to A are normal and the ciphertext is *s-ciphertext*. In $Game_q$, the ciphertext and all of the keys are in the special form.
4) $Game_{Final}$. This game is the same as $Game_q$ except the *s-ciphertext*. Given four random elements $m_0, m'_0, m_1, m'_1 \in Z_N$, $n+1$ random elements $R_{41}, R_{42}, \ldots, R_{4n}$, $R_4 \in G_{p4}$ and two challenge vectors $\vec{x}^{(0)}$ and $\vec{x}^{(1)}$, the *s-ciphertext* of challenge vector $\vec{x}^{(\beta)}$ is $C_{1i} = A_1^{m_0\left(\beta_i x_i^{(0)}+\alpha_i\right)+m_1\left(\beta_i x_i^{(1)}+\alpha_i\right)} g_2^{m'_0 a\left(\beta_i x_i^{(0)}+\alpha_i\right)+m'_1 a\left(\beta_i x_i^{(1)}+\alpha_i\right)} R_{4i}$ and $C_2 = U_1^{m_0+m_1} g_2^{u\left(m'_0+m'_1\right)} R_4$ where $i \in [1,n]$, $\beta \in \{0,1\}$ and a, u, α_i, β_i are required from $MSK = \{A_1 = g_1^a, U_1 = g_1^u, \alpha_i, \beta_i, g_3\}$.

Obviously, $Game_{Final}$ is a game such that the ciphertext holds no information of β. We will show these games are indistinguishable in the following lemmas.

Lemma 3.1. *Suppose that there exists a probabilistic polynomial time (PPT) algorithm A such that $Adv^{A}_{Game_{Real}} - Adv^{A}_{Game_{Restricted}} = \epsilon$. Then we can build a PPT algorithm B with advantage $\geq \frac{\epsilon}{3}$ in breaking GSD Assumption.*

Proof: Given $D = (N = p_1p_2p_3p_4, G, G_T, \hat{e}, g_1 \in G_{p1}, g_3 \in G_{p3}, g_4 \in G_{p4})$, B can simulate $Game_{Real}$ with A. With probability ϵ, A can generate vector $\vec{v}$ and $\vec{x}$ such that $\sum_{i=1}^{n} x_i v_i \bmod N \neq 0$ and $\sum_{i=1}^{n} x_i v_i \bmod p_2 = 0$. B uses these vectors to produce a nontrivial factor of N by computing $a = gcd\left(\sum_{i=1}^{n} x_i v_i, N\right)$. We set $b = \frac{N}{a}$. Notice that p_2 divides a and $N = ab = p_1p_2p_3p_4$, we consider three cases: Case 1: p_1 divides b; Case 2: p_1 cannot divide b and p_4 can divide b; Case 3: $a = p_1p_2p_4$ and $b = p_3$.

At least one of these cases must occur with probability $\geq \frac{\epsilon}{3}$. In Case 1, given D and T where $T \in G_{p_1}$ or $T \in G_{p_1p_2}$, B computes T^b. If T^b is the identity element of G_T, then $T \in G_{p_1}$. Otherwise, $T \in G_{p_1p_2}$. Therefore, B can break the GSD Assumption.

Case 2 is the same as Case 1 except that $T \in G_{p_4}$ or $T \in G_{p_2p_4}$. B computes T^b. If T^b is the identity element of G_T, then $T \in G_{p_4}$. Otherwise, $T \in G_{p_2p_4}$.

In Case 3, given $D = (N = p_1p_2p_3p_4, G, G_T, \hat{e}, g_1 \in G_{p1}, g_3 \in G_{p3}, g_4 \in G_{p4}, D_1D_2 \in G_{p_1p_2}, B_2B_3 \in G_{p_2p_3})$ and T where $T \in G_{p_1p_3}$ or $T \in G_{p_1p_2p_3}$, B computes $\hat{e}\left(T, (B_2B_3)^b\right)$. If $\hat{e}\left(T, (B_2B_3)^b\right)$ is the identity element of G_T, then $T \in G_{p_1p_3}$. Otherwise, $T \in G_{p_1p_2p_3}$.□

Lemma 3.2. *Suppose that there exists a PPT algorithm A such that $Adv^A_{Game_{Restricted}} - Adv^A_{Game_0} = \epsilon$. Then we can build a PPT algorithm B with advantage ϵ in breaking GSD Assumption.*

Proof: Given $D = (N = p_1p_2p_3p_4, G, G_T, \hat{e}, g_1 \in G_{p1}, g_3 \in G_{p3}, g_4 \in G_{p4})$, and T where $T \in G_{p_1}$ or $T \in G_{p_1p_2}$, B can simulate $Game_0$ or $Game_{Real}$ with A. To generate the public key, B chooses random exponents a, c, u, a_i, b_i, α_i and $\beta_i \in Z_N$ for each $i \in [1, n]$ and sets ${A_1}^{\beta_i}A_{4i} = {g_1}^{a\beta_i}{g_4}^{b_i}$, ${A_1}^{\alpha_i}B_{4i} = {g_1}^{a\alpha_i}{g_4}^{a_i}$ and $U_1U_4 = g_1^u g_4^c$ for each $i \in [1, n]$. Obviously, $U_1 = g_1^u$. B sends public key $\left\{N, U_1U_4, {A_1}^{\beta_i}A_{4i}, {A_1}^{\alpha_i}B_{4i}, g_4\right\}$ to A. Each time A asks B to provide a key for a vector $\vec{v}^{(j)} = \left(v_1^{(j)}, v_2^{(j)}, \ldots, v_n^{(j)}\right)$, B chooses $n+2$ random exponents r_j, $c_{j1}, c_{j2}, \ldots, c_{jn+1}$ and sets $K_{1i} = g_1^{ur_j\frac{v_i^{(j)}}{\beta_i}} g_3^{c_{ji}} = U_1^{r_j\frac{v_i^{(j)}}{\beta_i}} g_3^{c_{ji}}$, $K_2 = {A_1}^{r_j\sum_{i=1}^{n}\frac{\alpha_i v_i^{(j)}}{\beta_i}} g_3^{c_{jn+1}}$ for each $i \in [1, n]$.

After that, A sends B two challenge vectors, $\vec{x}^{(0)} = \left(x_1^{(0)}, x_2^{(0)}, \ldots, x_n^{(0)}\right)$ and $\vec{x}^{(1)} = \left(x_1^{(1)}, x_2^{(1)}, \ldots, x_n^{(1)}\right)$. B randomly chooses a $\beta \in \{0, 1\}$. Given $C_{41}, C_{42}, \ldots, C_{4n}$ and $C_4 \in G_{p_4}$ randomly (Random elements of G_{P_4} can be obtained by raising g_4 to random exponents modulo N), B generates the ciphertext as follows:

$$C_{1i} = T^{a\left(\beta_i x_i^{(\beta)} + \alpha_i\right)} C_{4i} \text{ and } C_2 = T^u C_4 \text{ where } i \in [1, n].$$

If $T \in G_{p_1p_2}$, suppose that $T = g_1^s g_2^{s'}$, then this is a *s-ciphertext* with $z_{ci} = s'a\left(\beta_i x_i^{(\beta)} + \alpha_i\right)$ and $z_c = s'u$. Note that $\theta_\beta = s'$ and $\theta_{1-\beta} = 0$ where $\beta \in \{0, 1\}$, this is distributed as in $Game_0$. If $T \in G_{p_1}$, this is a normal ciphertext. Therefore, if A can distinguish the $Game_{Real}$ from $Game_0$ with advantage ϵ, then B can use the output of A to break GSD Assumption with advantage ϵ. □

Lemma 3.3. *Suppose that there exists a PPT algorithm A such that $Adv^A_{Game_{k-1}} - Adv^A_{Game_k} = \epsilon$. Then we can build a PPT algorithm B with advantage ϵ in breaking GSD Assumption.*

Proof: Given $D = (N = p_1p_2p_3p_4, G, G_T, \hat{e}, g_1 \in G_{p1}, g_3 \in G_{p3}, g_4 \in G_{p4}, D_1D_2 \in G_{p_1p_2}, B_2B_3 \in G_{p_2p_3})$ and T where $T \in G_{p_1p_3}$ or $T \in G_{p_1p_2p_3}$, B can simulate $Game_{k-1}$ or $Game_k$ with A. Choosing random exponents a, u, c, a_i, b_i, α_i and $\beta_i \in Z_N$ for each $i \in [1, n]$, B sets ${A_1}^{\beta_i}A_{4i} = {g_1}^{a\beta_i}{g_4}^{b_i}$, ${A_1}^{\alpha_i}B_{4i} = {g_1}^{a\alpha_i}{g_4}^{a_i}$ and $U_1U_4 = g_1^u g_4^c$ for each $i \in [1, n]$, and sends public key $\left\{N, U_1U_4, {A_1}^{\beta_i}A_{4i}, {A_1}^{\alpha_i}B_{4i}, g_4\right\}$ to A. When A requests the j-th key for vector $\vec{v}^{(j)} = \left(v_1^{(j)}, v_2^{(j)}, \ldots, v_n^{(j)}\right)$, B generates the normal key or the semi-function key for vector $v^{(j)}$.

For $j < k$, B creates an *s-key*. Choosing random exponents r_j, r'_j, z_j and $t_{ji} \in Z_N$, for $i \in [1, n]$, B sets $K_{1i} = g_1^{\frac{ur_j v_i^{(j)}}{\beta_i}} (B_2B_3)^{\frac{ur'_j v_i^{(j)}}{\beta_i}} g_3^{t_{ji}}$ and $K_2 = g_1^{ar_j\sum_{i=1}^{n}\frac{\alpha_i v_i^{(j)}}{\beta_i}} (B_2B_3)^{ar'_j\sum_{i=1}^{n}\frac{\alpha_i v_i^{(j)}}{\beta_i}} g_3^{z_j}$.

For $j > k$, B generates normal keys by choosing random exponents r_j, w_j and $t_{ji} \in Z_N$ for $i \in [1, n]$ and setting $K_{1i} = g_1^{\frac{ur_j v_i^{(j)}}{\beta_i}} (g_3)^{t_{ji}}$, $K_2 = g_1^{ar_j\sum_{i=1}^{n}\frac{\alpha_i v_i^{(j)}}{\beta_i}} (g_3)^{w_j}$.

To create the k-th requested key, choosing $n+1$ random exponents $w_{k1}, w_{k2}, \ldots, w_{kn+1}$, B sets $K_{1i} = T^{\frac{uv_i^{(k)}}{\beta_i}} g_3^{w_{ki}}$ for each $i \in [1,n]$, $K_2 = T^{a\sum_{i=1}^{n} \frac{\alpha_i v_i^{(k)}}{\beta_i}} (g_3)^{w_{kn+1}}$. It can be found that $z_{ki} = \frac{uv_i^{(k)}}{\beta_i}$ and $z_k = a\sum_{i=1}^{n} \frac{\alpha_i v_i^{(k)}}{\beta_i}$.

After the key request phase, A sends B two challenge vectors, $\vec{x}^{(0)}$ and $\vec{x}^{(1)}$. B randomly chooses a $\beta \in \{0,1\}$ and generates the *s-ciphertext*. For the challenge vector $\vec{x}^{(\beta)} = \left(x_1^{(\beta)}, x_2^{(\beta)}, \ldots, x_n^{(\beta)}\right)$, B generates $n+1$ random elements $C_{41}, C_{42}, \ldots, C_{4n}, C_4 \in G_{P4}$ by taking g_4 and raising it to random exponents module N and sets $C_{1i} = (D_1D_2)^{a(\beta_i x_i^{(\beta)} + \alpha_i)} C_{4i}$ for each $i \in [1,n]$ and $C_2 = (D_1D_2)^u C_4$.

Suppose that $D_1D_2 = g_1^{d_1} g_2^{d_2}$, it sets $z_c = d_2 u$, $z_{ci} = d_2 a\left(\beta_i x_i^{(\beta)} + \alpha_i\right)$. Note that $\theta_\beta = d_2$ and $\theta_{1-\beta} = 0$ where $\beta \in \{0,1\}$.

Suppose that $\vec{x}^{(\beta)} \cdot \vec{v}^{(k)} = \sum_{i=1}^{n} x_i^{(\beta)} v_i^{(k)} \neq 0 \bmod N$ and $\vec{x}^{(\beta)} \cdot \vec{v}^{(k)} = \sum_{i=1}^{n} x_i^{(\beta)} v_i^{(k)} = 0 \bmod p_2$, it has $z_c z_k = \sum_{i=1}^{n} z_{ci} z_{ki} \bmod p_2$. It means that if $\vec{x}^{(\beta)} \cdot \vec{v}^{(k)} = 0 \bmod p_2$, then A has made an invalid key request. This is where we use our additional modular restriction. Therefore, according to the $Game_{Restricted}$, as long as $\vec{x}^{(\beta)} \cdot \vec{v}^{(k)} \neq 0 \bmod p_2$, we can find that z_k, z_c, z_{ki} and z_{ci} are correctly distributed to A for each $i \in [1,n]$.

Besides, we observe that, if B attempts to test whether k-th key is *s-key* by creating an *s-ciphertext* for a vector $\vec{x}^{(k)}$ such that $\vec{x}^{(k)}\vec{v}^{(k)} = 0$ and trying to decrypt, then B can find that decryption can still work whether k-th key is special or not since $z_c z_k = \sum_{i=1}^{n} z_{ci} z_{ki}$.

Therefore, we can conclude that, if $T \in G_{p_1p_3}$, then B has properly simulated $Game_{k-1}$. If $T \in G_{p_1p_2p_3}$, then B has properly simulated $Game_k$. So, we can find that if A can distinguish the $Game_{k-1}$ from $Game_k$ with advantage ϵ, then B can use the output of A to break GSD Assumption with advantage ϵ. □

Lemma 3.4. *Suppose that there exists a PPT algorithm A such that*

$$Adv_{Game_q}^{A} - Adv_{Game_{Final}}^{A} = \epsilon.$$

Then we can build a PPT algorithm B with advantage ϵ in breaking GSD Assumption.

Proof: Given $D = (N = p_1p_2p_3p_4, G, G_T, \hat{e}, g_2 \in G_{p_2}, g_3 \in G_{p_3}, g_4 \in G_{p_4}, W_1W_4 \in G_{p_1p_4}, E_1E_2 \in G_{p_1p_2})$ and T where $T \in G_{p_2p_4}$ or $T \in G_{p_1p_2p_4}$, B can simulate $Game_q$ or $Game_{Final}$ with A. B chooses random exponents a, c, u, a_i, b_i, α_i, $\beta_i \in Z_N$ for $i \in [1,n]$, and generates ${A_1}^{\beta_i} A_{4i} = (W_1W_4)^{a\beta_i} g_4^{a_i}$, ${A_1}^{\alpha_i} B_{4i} = (W_1W_4)^{a\alpha_i} g_4^{b_i}$, and $U_1U_4 = (W_1W_4)^u g_4^c$. Then, B sends public key $\left\{N, U_1U_4, {A_1}^{\beta_i} A_{4i}, {A_1}^{\alpha_i} B_{4i}, g_4\right\}$ to A. Each time B is asked to provide a key for a vector $\vec{v}^{(j)} = \left(v_1^{(j)}, v_2^{(j)}, \ldots, v_n^{(j)}\right)$, suppose that $E_1 = W_1^{e_1}$ where $e_1 \in Z_n$, B creates an *s-key* by choosing random exponents r_j, $z_{j1}, z_{j2}, \ldots, z_{jn+1} \in Z_N$ and setting $K_{1i} = (E_1E_2)^{ur_j \frac{v_i^{(j)}}{\beta_i}} g_3^{z_{ji}} = U_1^{e_1 r_j \frac{v_i^{(j)}}{\beta_i}} E_2^{ur_j \frac{v_i^{(j)}}{\beta_i}} g_3^{z_{ji}}$ and $K_2 = (E_1E_2)^{ar_j \sum_{i=1}^{n} \frac{\alpha_i v_i^{(j)}}{\beta_i}} g_3^{z_{jn+1}} = A_1^{e_1 r_j \sum_{i=1}^{n} \frac{\alpha_i v_i^{(j)}}{\beta_i}} E_2^{ar_j \sum_{i=1}^{n} \frac{\alpha_i v_i^{(j)}}{\beta_i}} g_3^{z_{jn+1}}$, where $i \in [1,n]$.

At some point, A sends B two challenge vectors, $\vec{x}^{(0)}$ and $\vec{x}^{(1)}$. Randomly choosing $\beta \in \{0,1\}$, $s, s' \in Z_N$, and $R_{4i}, R_4 \in G_{p_4}$ where $i \in [0,1]$, B creates the challenge ciphertext as follows: For each $i \in [1,n]$, $C_{1i} = (E_1E_2)^{sa\left(\beta_i x_i^{(\beta)} + \alpha_i\right)} T^{s'a\left(\beta_i x_i^{(\beta)} + \alpha_i\right) + s''a\left(\beta_i x_i^{(1-\beta)} + \alpha_i\right)} R_{4i}$, and $C_2 = (E_1E_2)^{su} T^{s'u + s''u} R_4$.

If $T \in G_{p_2p_4}$, suppose that $E_1 = W_1^{e_1}$, $E_2 = g_2^{e_2}$ and $T = g_2^{t_2} T_4$ where $e_1, e_2, t_2 \in Z_N$ and $T_4 \in G_{p_4}$, then the challenge ciphertext is:

$$C_{1i} = A_1^{se_1\left(\beta_i x_i^{(\beta)} + \alpha_i\right)} g_2^{(e_2 sa + t_2 s'a)\left(\beta_i x_i^{(\beta)} + \alpha_i\right) + t_2 s'' a\left(\beta_i x_i^{(1-\beta)} + \alpha_i\right)} R'_{4i}$$

and $C_2 = U_1^{se_1} g_2^{(e_2su+t_2s'u+t_2s''u)} R'_4$, where $R'_{4i} = T_4^{s'a\left(\beta_i x_i^{(\beta)}+\alpha_i\right)+s''a\left(\beta_i x_i^{(1-\beta)}+\alpha_i\right)} R_{4i}$, $R'_4 = T_4^{s'u+s''u} R_4$ and $i \in [1,n]$. Obviously, $\theta_\beta = e_2 s + t_2 s'$ and $\theta_{1-\beta} = t_2 s''$. Because e_2, t_2, s, s', s'' are chosen randomly in Z_N, in this case, B has properly simulated $Game_q$.

If $T \in G_{p_1p_2p_4}$, suppose that $E_1 = W_1^{e_1}$, $E_2 = g_2^{e_2}$ and $T = W_1^{t_1} g_2^{t_2} T_4$ where $e_1, e_2, t_1, t_2 \in Z_N$ and $T_4 \in G_{p_4}$, then the challenge ciphertext is:

$$C_{1i} = A_1^{(se_1+t_1s')\left(\beta_i x_i^{(\beta)}+\alpha_i\right)+t_1s''\left(\beta_i x_i^{(1-\beta)}+\alpha_i\right)} g_2^{(e_2sa+t_2s'a)\left(\beta_i x_i^{(\beta)}+\alpha_i\right)+t_2s''a\left(\beta_i x_i^{(1-\beta)}+\alpha_i\right)} R'_{4i}$$

and $C_2 = U_1^{(se_1+s't_1)+s''t_1} g_2^{(e_2su+t_2s'u+t_2s''u)} R'_4$, where $R'_{4i} = T_4^{s'a\left(\beta_i x_i^{(\beta)}+\alpha_i\right)+s''a\left(\beta_i x_i^{(1-\beta)}+\alpha_i\right)} R_{4i}$, $R'_4 = T_4^{s'u+s''u} R_4$ and $i \in [1,n]$. Obviously, $\theta_\beta = e_2 s + t_2 s'$ and $\theta_{1-\beta} = t_2 s''$. Because e_1, e_2, t_1, t_2, s, s', s'' are chosen randomly in Z_N, in this case, B has properly simulated $Game_{Final}$. Obviously, in $Game_{Final}$, given a key for vector $\vec{v}$ which is orthogonal to the two challenge vectors, the decryption algorithm still works. Therefore, we can find that if A can distinguish the $Game_q$ from $Game_{Final}$ with advantage ϵ, then B can use the output of A to break GSD Assumption with advantage ϵ. $\square$

Theorem 3.1. *If GSD Assumption holds, then our IPE scheme is secure.*

Proof: If GSD Assumption holds, the real security game is indistinguishable from $Game_{Final}$ based on the previous lemmas which have been proven. In $Game_{Final}$, the challenge ciphertext does not contain the value of β. Therefore, the value of β is information-theoretically hidden from the attacker. We can say that the attacker can attain no advantage in breaking our IPE scheme. $\square$

3.3. **Comparison.** Let $|G|$ and $|G_T|$ represent the size of an element of G and G_T, where both G and G_T are the group with prime order. Let $|G'|$ and $|G'_T|$ represent the size of an element of G and G'_T, where both $|G'|$ and $|G'_T|$ are the group with composite order. Note that $|G'| = c|G|$ and $|G'_T| = d|G_T|$, where c and d are a constant about 50 [4]. According to this description, the results of the comparison with the existing IPE scheme is shown in Table 1. Because the pk size, msk size, ciphertext size, sk size, encryption time and decryption time in an IPE scheme is linear with these sizes in searchable encryption scheme, we can say that the searchable encryption scheme based on the proposed scheme has less pk size, sk size, encryption time and decryption time.

TABLE 1. Comparison with the previous IPE scheme

	Previous Scheme [2]	Proposed
pk size	$O(n^2)\|G\|$	$O(n)\|G'\|$
msk size	$O(n^2)\|G\|$	$O(n)\|G'\|$
ciphertext size	$O(n)\|G\| + \|G_T\|$	$O(n)\|G'\| + \|G'_T\|$
sk size	$O(n)\|G\|$	$O(n)\|G'\|$
Encryption time	$O(n^2)$	$O(n)$
Decryption time	$O(n^2)$	$O(n)$

4. **Conclusion.** In this paper, we proposed a new fully secure predicate-only IPE scheme. The comparison shows that the searchable encryption scheme based on our scheme has better space and time complexity than the one based on the previous IPE schemes [1, 2]. The open problem is to construct a complete IPE scheme (not predicate only) which can be transformed to a more efficient searchable encryption scheme.

Acknowledgment. The authors gratefully acknowledge the NSAF of China under Grant No. 10876012.

REFERENCES

[1] J. Katz, A. Sahai and B. Waters, Predicate encryption supporting disjunctions, polynomial equations, and inner products, *Advances in Cryptology – EUROCRYPT 2008, Lecture Notes in Computer Science*, vol.4965, pp.146-162, 2008.

[2] T. Okamoto and K. Takashima, Adaptively attribute-hiding (hierarchical) inner product encryption, in *Advances in Cryptology – EUROCRYPT 2008, Lecture Notes in Computer Science*, D. Pointcheval and T. Johansson (eds.), Springer, Heidelberg, 2012.

[3] A. Lewko, *Tools for Simulating Features of Composite Order Bilinear Groups in the Prime Order Setting*, http://eprint.iacr.org/2011/490.pdf.

[4] D. M. Freeman, Converting pairing-based cryptosystems from composite-order groups to prime-order groups, *Lecture Notes in Computer Science*, vol.6110, pp.44-61, 2010.

[5] D. J. Park, K. Kim and P. J. Lee, Public key encryption with conjunctive field keyword search, in *WISA 2004, Lecture Notes in Computer Science*, C. H. Lim and M. Yung (eds.), Springer, 2004.

[6] A. B. Lewko and B. Waters, New techniques for dual system encryption and fully secure hibe with short ciphertexts, *TCC 2010, Lecture Notes in Computer Science*, Springer, vol.5978, pp.455-479, 2010.