Unbounded Non-Zero Inner Product Encryption

Bishnu Charan Behera and Somindu C. Ramanna

Department of Computer Science and Engineering,
Indian Institute of Technology Kharagpur, India
bishnu_charan_behera@iitkgp.ac.in, somindu@cse.iitkgp.ac.in

Abstract. In a non-zero inner product encryption (NIPE) scheme, ciphertexts and keys are associated with vectors from some inner-product space. Decryption of a ciphertext for $\vec{x}$ is allowed by a key for $\vec{y}$ if and only if the inner product $\langle \vec{x}, \vec{y} \rangle \neq 0$. Existing constructions of NIPE assume the length of the vectors are fixed apriori. We present the first constructions of *unbounded* non-zero inner product encryption (UNIPE) with constant sized keys. Unbounded here refers to the size of vectors not being pre-fixed during setup. Both constructions, based on bilinear maps, are proven selectively secure under the decisional bilinear Diffie-Hellman (DBDH) assumption.
Our constructions are obtained by transforming the unbounded inner product functional encryption (IPFE) schemes of Dufour-Sans and Pointcheval (ACNS 2019), one in the *strict domain* setting and the other in the *permissive domain* setting. Interestingly, in the latter case, we prove security from DBDH, a static assumption while the original IPE scheme relied on an interactive parameterised assumption. In terms of efficiency, features of the IPE constructions are retrained after transformation to NIPE. Notably, the public key and decryption keys have constant size.

Keywords: Unbounded vectors · Non-zero inner product encryption· Strict domain · Permissive domain

1 Introduction

Functional encryption (FE) [7] is a generalisation of public key encryption where each public key is associated with several secret keys, all of which have different decryption capabilities. More precisely, each secret key sk_f is associated with a function f, which allows to recover $f(m)$ from an encryption of a message m under the associated public key. Attribute-based encryption (ABE) [23,15] is a specific form of FE that allows fine-grained access to encrypted data. Here, a ciphertext is associated with an attribute $\vec{x}$ and a secret key for a user is associated to some attribute $\vec{y}$. Decryption succeeds i.e., the message can be fully recovered if and only if some relation (or predicate) R on $\vec{x}, \vec{y}$ holds true i.e., $R(\vec{x}, \vec{y}) = 1$. An ABE system is deemed secure if a colluding group of users holding secret keys cannot compromise the security of a ciphertext that their secret keys are not capable of decrypting.

In an inner product encryption (IPE) system (special case of ABE), attributes belong to some inner product space V and the relation is given by $R(\vec{x}, \vec{y}) = 1$ iff $\langle \vec{x}, \vec{y} \rangle = 0$, for $\vec{x}, \vec{y} \in V$. On the other hand, if R is defined as $R(\vec{x}, \vec{y}) = 1$ iff $\langle \vec{x}, \vec{y} \rangle \neq 0$, then the resulting ABE is called non-zero inner product encryption (NIPE) [3,27]. Applications of NIPE include identity-based revocation (IBR), an important primitive that allows to broadcast encrypted messages so that only a "small" subset of the recipients (known to the encryption algorithm) cannot decrypt i.e., their decryption capabilities are revoked.

In all existing NIPE constructions, the size of vectors are pre-determined and all public parameters of the sytem are chosen based on that. This makes them incapable of handling variable-length vectors. A layman approach to overcome this problem is to fix the size n to be arbitrarily large. This,

however, would lead to large parameters whose size typically grows linearly in n. A natural question is whether there exists a NIPE scheme with the parameters being completely unconstrained by the lengths of the vectors/attributes in keys and ciphertexts. We call such a scheme *unbounded* non-zero inner product encryption (UNIPE). Our main goal in this paper is to design a UNIPE scheme that achieves reasonable efficiency and can be proven secure under well-studied complexity assumptions.

Our Contribution. We propose the first constructions of non-zero inner product encryption using asymmetric pairings that support unbounded or variable-length vectors. The inner product of two variable length vectors $\vec{x} = (x_i)_{i \in D}$, $\vec{y} = (y_i)_{i \in D'}$ defined over domains D, D' respectively, is defined as $\langle \vec{x}, \vec{y} \rangle = \sum_{i \in D \cap D'} x_i y_i$. Our schemes are derived from the unbounded inner product FE schemes of Dufour-Sans and Pointcheval [14]. In an unbounded inner product FE scheme, decryption of a ciphertext for $\vec{x}$ by a key for $\vec{y}$ would recover $\langle \vec{x}, \vec{y} \rangle$. A transformation to NIPE must facilitate hiding another message, say M, in the ciphertext so that decryption recovers this message if and only if $\langle \vec{x}, \vec{y} \rangle \neq 0$.

Strict Domain. In the strict domain setting, decryption works only when the domains D, D' of the vectors corresponding to ciphertext and key respectively are identical. We first transform the IPFE construction of [14] in the strict domain setting to NIPE retaining the same efficiency. The resulting scheme is proven selectively secure from the decisional bilinear Diffie-Hellman (DBDH) assumption in the random oracle model.

Permissive Domain. In the permissive domain setting, decryption works even when D' is a subset of D. We apply our transformation to the permissive unbounded IPFE scheme of [14] and obtain a UNIPE scheme which we prove to be (selectively) secure under the decisional bilinear Diffie-Hellman (DBDH) assumption in the random oracle model. In contrast to the original scheme which relied on an interactive parameterised assumption called linearly extended DBDH, our proof relies on a well-studied static assumption.

Related Work. Katz, Sahai and Waters [16] were the first to propose the notion of inner product attribute encryption along with constructions and then a large number of works [18,24,21,2,17,19,20,21,22] followed, focussing on different aspects (such as efficiency, security) of the design of IPE and on extensions to generalised primitives. The first construction of NIPE [3] achieved security in the so-called *co-selective* model under the decision linear and decisional bilinear Diffie-Hellman assumptions in addition to constant-size ciphertexts. On the other hand, public parameters and secret keys have sizes linear in n, the pre-fixed length of vectors. Attrapadung *et al.* [4] and Yamada *et al.* [26] suggested more efficient constructions with security from the parameterised n-DBDHE assumption. The first construction of adaptively secure NIPE [22] (from decisional linear assumption) had linear sized parameters while either the ciphertext or the key is of constant-size. More efficient constructions appeared in [8] and [10]. Chen *et al.* [9] put forth the first NIPE scheme that simultaneously achieves short ciphertexts and secret keys with selective security from n-DBDHE assumption. Katsumata and Yamada [27] proposed the first NIPE without bilinear maps.

The first practical functional encryption systems for inner product (linear) functionality were built by Abdalla *et al.* [1] from simple assumptions, namely, the decision Diffie-Hellman and learning-with-errors assumptions. Subsequent works [5,6] showed how to achieve adaptive security and other security guarantees.

The notion of unbounded vectors in the inner product setting was first considered by Okamoto and Takashima in [20]. They proposed an unbounded zero IPE. Their scheme achieved adaptive security under DLIN assumption. Recently, Dutta et al. [13] UZIPE construction achieved adaptive security with reduced ciphertext and secret key size (in comparison to [20]). Unbounded vectors have also been considered in the context of designing FE schemes for inner product functionality. Tomida and Takashima [25] propose a construction that achieves adaptive security at the cost of ciphertext and key sizes being linear in the sizes of the associated domains. Dufour-Sans and Pointcheval [14] suggest a scheme that achieves short keys while obtaining security in a more restricted model. Unbounded vectors have also been considered in the context of multi-input FE [12] for the inner product functionality. A more recent work [11] considers simultaneously evaluating unbounded inner-product predicates (both zero and non-zero) and accordingly decrypt the inner-poduct of two separate vectors associated with the ciphertext and key respectively.

2 Technical Overview

As mentioned earlier, both our UNIPE schemes are derived from the unbounded inner product FE schemes of Dufour-Sans and Pointcheval [14]. We now provide a brief technical overview of our first construction in the strict domain setting.

Let $\vec{x} = (x_i)_{i \in D}$, $\vec{y} = (y_i)_{i \in D'}$ be 2 vectors defined over domains D, D' respectively. And let $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ be an asymmetric pairing of prime order p with P_1, P_2 generating $\mathbb{G}_1, \mathbb{G}_2$ respectively. In the strict domain FE scheme of [14], the ciphertext ct and key $\mathsf{sk}_{\vec{y}}$ corresponding to $\vec{x}$ and $\vec{y}$ are given by

$$\mathsf{ct} = (tP_1, (c_i)_{i \in D}) \text{ where } c_i = e(P_1, P_2)^{x_i} e(sP_1, tu_{i||D} P_2), \forall i \in D,$$

$$\mathsf{sk}_{\vec{y}} = \left(\vec{y}, -s(\sum_{i \in D'} y_i u_{i||D'}) P_2 \right),$$

where s is the master secret and its encoding in $\mathbb{G}_1$ is available in the public parameters, $u_{i||D}, u_{i||D'}$ are outputs of a hash function $\mathcal{H} : \{0,1\}^* \to \mathbb{G}_2$. The decryption algorithm recovers $e(P_1, P_2)^{\langle \vec{x}, \vec{y} \rangle}$ if $D = D'$ and obtains the inner product $\langle \vec{x}, \vec{y} \rangle$ via a discrete logarithm computation. In order to ensure efficiency of the discrete logarithm computation, the value of the inner product must be bounded. We carefully modify their scheme by introducing another element in the ciphertext that hides the message (or *payload*) in addition to randomising the x_i-component of each c_i. Let $\mathbb{G}_T$ be the message space and let M denote the message to be encrypted to vector $\vec{x}$. We mask the message with a $e(P_1, P_2)^{-zst}$ where $z \in \mathbb{Z}_p$ is the scalar used to randomise the x_i-component of c_i. The ciphertext and key have the following structure.

$$\mathsf{ct} = (tP_1, \vec{x}, (c_i)_{i \in D}, \hat{c}) \text{ where } c_i = e(sP_1, tP_2)^{z x_i} e(sP_1, tu_{i||D} P_2), \forall i \in D,$$

$$\hat{c} = M \cdot e([s]_1, [t]_2)^{-z}.$$

$$\mathsf{sk}_{\vec{y}} = \left(\vec{y}, -s(\sum_{i \in D'} y_i u_{i||D'}) P_2 \right).$$

Upon FE decryption we recover $e(P_1, P_2)^{zst \langle \vec{x}, \vec{y} \rangle}$ from which we can recover $e(P_1, P_2)^{-zst}$ using $\langle \vec{x}, \vec{y} \rangle$ which can in turn be used to unmask the message. We prove selective security of our scheme in the random oracle model from the DBDH assumption. The proof uses ideas from [1,14].

The construction of the permissive UNIPE scheme also proceeds in a similar manner. The main novelty lies in the fact that while the permissive unbounded IPFE of [14] relies on an interactive parameterised assumption, our transformed scheme can be proved secure relying only on DBDH, which is a static well-studied assumption.

3 Preliminaries

3.1 Notation

We write $x_1, \ldots, x_k \xleftarrow{\mathrm{R}} \mathcal{X}$ to indicate that $x_1, \ldots, x_k$ are sampled independently from a set $\mathcal{X}$ according to some distribution R (U denotes uniform distribution). For a (probabilicstic) algorithm $\mathcal{A}$, $y \longleftarrow \mathcal{A}(x)$ means that y is chosen according to the output distribution of $\mathcal{A}$ on input x.

Unbounded Vectors An unbounded vector is written as $\vec{x} = (x_i)_{i \in D}$ where D, a finite subset of $\mathbb{N}^*$ is called the domain of $\vec{x}$. In this paper, $x_i \in \mathbb{Z}_p$ for all $i \in D$, where p is defined by the bilinear map used in the construction of our encryption scheme.

Inner Products Given two vectors $\vec{x} = (x_i)_{i \in D}$ and $\vec{y} = (y_i)_{i \in D'}$, the inner product $\langle \vec{x}, \vec{y} \rangle$ is a function defined as:

$$\langle \vec{x}, \vec{y} \rangle = \sum_{i \in D \cap D'} x_i y_i$$

where the domains D and D' are non-empty finite subsets of $\mathbb{N}^*$.

3.2 Bilinear Groups and Related Assumptions

A bilinear map $\mathcal{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, P_1, P_2, e, p)$ consists of cyclic groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ of prime order p with the first two groups given by generators P_1, P_2 respectively and an *efficiently computable* map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, with the following two properties:

Bilinearity: $e(aQ_1, bQ_2) = e(Q_1, Q_2)^{ab}$, for all $Q_1 \in \mathbb{G}_1$, $Q_2 \in \mathbb{G}_2$ and $a, b \in \mathbb{Z}_p$.
Non-degeneracy: $e(P_1, P_2)$ is a generator for $\mathbb{G}_T$ unless $P_1 = 0$ or $P_2 = 0$ where $P_1 \in \mathbb{G}_1$, $P_2 \in \mathbb{G}_2$.

The bilinear group generator $\mathsf{GroupGen}(\vartheta)$ takes a security parameter ϑ as input and returns a bilinear map $\mathcal{G}$ over a ϑ-bit prime p.

We represent an element $aP_\iota \in \mathbb{G}_\iota$ for $\iota \in \{1, 2\}$ as $[a]_\iota$ and an element $e(P_1, P_2)^a \in \mathbb{G}_T$ as $[a]_T$, where P_ι is a generator of G_ι. Given $[a]_\iota$ it is generally hard to obtain a. Observe that for $a, b \in \mathbb{Z}_p$, given $[a]_\iota, [b]_\iota$, one can compute $[a + b]_\iota$ as $[a]_\iota + [b]_\iota$. Furthermore, given $[a]_1, [b]_2$, one can compute $[ab]_T$ as $e([a]_1, [b]_2)$.

Decisional Bilinear Diffie-Hellman (DBDH) Assumption. Given an asymmetric bilinear map $\mathcal{G} \leftarrow \mathsf{GroupGen}(\vartheta)$ with the following distributions:

$$([k]_1, [l]_1, [k]_2, [m]_2, [klm]_T) \text{ and } ([k]_1, [l]_1, [k]_2, [m]_2, [r]_T)$$

where $k, l, m, r \xleftarrow{\mathrm{U}} \mathbb{Z}_p$, the DBDH problem asks to distinguish between the above distributions. For a probabilistic polynnomial time adversary $\mathscr{A}$, define

$$\mathsf{Adv}^{\mathrm{DBDH}}_{\mathscr{A},\mathcal{G}}(\vartheta) = \Big| \Pr[\mathscr{A}(\mathcal{G}, [k]_1, [l]_1, [k]_2, [m]_2, [klm]_T) = 1] - \Pr[\mathscr{A}(\mathcal{G}, [k]_1, [l]_1, [k]_2, [m]_2, [r]_T) = 1] \Big|.$$

The Decisional Bilinear Diffie-Hellman(DBDH) assumption holds if for all PPT adversaries $\mathscr{A}$, $\mathsf{Adv}^{\mathrm{DBDH}}_{\mathscr{A},\mathcal{G}}(\vartheta) \leq \mathsf{negl}(\vartheta)$.

3.3 Unbounded Non-Zero Inner Product Encryption(UNIPE)

Let $\mathcal{M}$ be a message space, and $\mathcal{V}$ be an inner product space. A non-zero inner product encryption scheme over $\mathcal{V}$ is defined by the following probabilistic algorithms.

Setup(ϑ)**:** Takes as input a security parameter ϑ and returns the public parameter pp with the master secret key msk as output.

KeyGen$(\mathsf{msk}, \vec{y}, D)$**:** Inputs the master secret key msk, a vector $\vec{y} = (y_i)_{i \in D}$ with a non-empty domain set $D \subseteq s[\vartheta]$, where $s[\vartheta]$ is a polynomial, and returns a secret key $\mathsf{sk}_{\vec{y}}$.

Encrypt$(\mathsf{pp}, M, \vec{x})$**:** Inputs the public parameters pp, vector $\vec{x} = (x_i)_{i \in D'}$ with a message M, where D' is a non-empty domain set with $D' \subseteq m[\vartheta]$ for some polynomial $m[\vartheta]$ and returns a ciphertext ct.

Decrypt$(\mathsf{pp}, \mathsf{sk}_{\vec{y}}, \mathsf{ct})$**:** Recovers and returns the message M if $\langle \vec{x}, \vec{y} \rangle \neq 0$; otherwise returns $\perp$.

Correctness. The scheme above is said to be correct if for all $(\mathsf{pp}, \mathsf{msk}) \longleftarrow \mathsf{Setup}(\vartheta)$, for any message M and any vector $\vec{x}$ over any domain D', for any vector $\vec{y}$ over any domain D, if $\mathsf{ct} \longleftarrow \mathsf{Encrypt}(\mathsf{pp}, M, \vec{x})$ and $\mathsf{sk}_{\vec{y}} \longleftarrow \mathsf{KeyGen}(\mathsf{msk}, \vec{y}, D)$, then $\mathsf{Decrypt}(\mathsf{pp}, \mathsf{sk}_{\vec{y}}, \mathsf{ct}) = M$ if and only if $D = D'$ and $\langle \vec{x}, \vec{y} \rangle \neq 0$.

Selective Security. The notion of selective security of a UNIPE scheme is formally defined in terms of a game $\mathsf{sel\text{-}IND}$ between an adversary $\mathscr{A}$ and a challenger $\mathscr{C}$ described below.

Initialization: The game starts by adversary $\mathscr{A}$ declaring a vector $\vec{x}^* = (\vec{x}^*_i)_{i \in D^*}$ for some domain set D^*.

Setup: The challenger $\mathscr{C}$ executes the Setup algorithm and passes the public parameters to $\mathscr{A}$.

Key Query Phase 1: A number of key extraction queries are made by $\mathscr{A}$. Each query vector $\vec{y} = (y_i)_{i \in D'}$ has the restriction that $\langle \vec{x}^*, \vec{y} \rangle = 0$, and the challenger $\mathscr{C}$ responds to this with the secret key $\mathsf{sk}_{\vec{y}}$.

Challenge Ciphertext: $\mathscr{A}$ requests the challenge ciphertext by submitting two challenge messages M^0, M^1, to be encrypted under $\vec{x}^*$. A bit $\zeta \xleftarrow{\mathrm{U}} \{0, 1\}$ is uniformly chosen by the challenger which then encrypts M^ζ under $\vec{x}^*$ and returns the challenge ciphertext ct^* to $\mathscr{A}$.

Key Query Phase 2: Identical to Key Query Phase 1.

Guess: The adversary concludes the game with a guess ζ' of ζ.

In the $\mathsf{sel\text{-}IND}$ game, if $\zeta' = \zeta$, then $\mathscr{A}$ wins the game. The advantage of the adversary $\mathscr{A}$ in winning the $\mathsf{sel\text{-}IND}$ game is defined as:

$$\mathsf{Adv}^{\mathsf{sel\text{-}IND}}_{\mathscr{A},\mathrm{UNIPE}}(\vartheta) = \left| \Pr[\zeta' = \zeta] - \frac{1}{2} \right|$$

The UNIPE scheme is said to be selectively secure if no probabilistic polynomial time adversary has a non-negligible advantage in winning the preceding game.

3.4 Restrictions on Domains of Vectors

We consider unbounded NIPE in two settings with some restrictions placed on the inner product functionality leading to successful decryption.

Strict Unbounded NIPE. In the strict domain setting, given ciphertext and key for two related vectors $\vec{x} = (x_i)_{i \in D} \in \mathbb{Z}_p^{|D|}$ and $\vec{y} = (y_i)_{i \in D'} \in \mathbb{Z}_p^{|D'|}$ respectively, the decryption algorithm will work when $D = D'$. In the context of NIPE, decryption of the message M is possible if and only if $D = D'$ and $\langle \vec{x}, \vec{y} \rangle \neq 0$.

Permissive Unbounded NIPE. A NIPE scheme in the permissive domain setting allows decryption when $D' \subseteq D$ and $\langle \vec{x}, \vec{y} \rangle \neq 0$. We emphasise that D' is the domain of the vector corresponding to the secret key used for decryption and D corresponds to the vector associated with ciphertext.

4 A Strict Domain UNIPE Scheme

4.1 Construction

We first present our construction of UNIPE. As mentioned earlier, our construction closely follows that in [14]. Below are the algorithms defining our scheme.

Setup(ϑ)**:** Choose a pairing $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, P_1, P_2, e) \longleftarrow \mathsf{GroupGen}(\vartheta)$. Choose $s \xleftarrow{\text{U}} \mathbb{Z}_p$ and a hash function $\mathcal{H} : \{0,1\}^* \to \mathbb{G}_2$. The public parameter is defined as $\mathsf{pp} = [s]_1$, and the master secret key is set as $\mathsf{msk} = (s, \mathsf{pp})$.

Encrypt$(\mathsf{pp}, M, \vec{x} = (x_i)_{i \in D})$**:** Takes the public parameter pp, the message M and an unbounded vector $\vec{x} = (x_i)_{i \in D}$ over some domain set D as input. Elements $z, t \xleftarrow{\text{U}} \mathbb{Z}_p$ are chosen uniformly. Set

$$c_i = e([s]_1, [t]_2)^{z x_i} e([s]_1, t[u_{i||D}]_2), \forall i \in D$$
$$\hat{c} = M \cdot e([s]_1, [t]_2)^{-z}$$

where $[u_{i||D}]_2 = \mathcal{H}(i||D)$. Finally, return the ciphertext $\mathsf{ct} = ([t]_1, \vec{x}, (c_i)_{i \in D}, \hat{c})$.

KeyGen$(\mathsf{msk}, \vec{y} = (y_i)_{i \in D'})$**:** Computes and returns

$$\mathsf{sk}_{\vec{y}} = \left(\vec{y}, -s \sum_{i \in D'} y_i [u_{i||D'}]_2 \right)$$

Decrypt$(\mathsf{pp}, \mathsf{ct}, \mathsf{sk}_{\vec{y}})$**:** Takes pp, a ciphertext $\mathsf{ct} = ([t]_1, \vec{x} = (x_i)_{i \in D}, (c_i)_{i \in D}, \hat{c})$, and a secret key $sk_{\vec{y}} = (\vec{y}, f)$ as input. If the inner product of the associated vectors $\vec{x}$ and $\vec{y}$ is zero, i.e., $\langle \vec{x}, \vec{y} \rangle = 0$ then return $\perp$; otherwise, compute and output

$$M = \hat{c} \cdot \left(e([t]_1, f) \prod_{i \in D \cap D'} c_i^{y_i} \right)^{\frac{1}{\langle \vec{x}, \vec{y} \rangle}}.$$

Correctness: When $D = D'$ and $\langle \vec{x}, \vec{y} \rangle \neq 0$, we have

$$\begin{aligned}
&\hat{c} \cdot \left(e([t]_1, f) \prod_{i \in D} c_i^{y_i} \right)^{\frac{1}{\langle \vec{x}, \vec{y} \rangle}} \\
&= \hat{c} \cdot \left(e\left([t]_1, -s \sum_{i \in D} y_i [u_{i||D}]_2 \right) \prod_{i \in D} e(P_1, P_2)^{stzy_i x_i + sty_i u_{i||D}} \right)^{\frac{1}{\langle \vec{x}, \vec{y} \rangle}} \\
&= \hat{c} \cdot \left(e(P_1, P_2)^{-st \sum_{i \in D} y_i u_{i||D}} e(P_1, P_2)^{\sum_{i \in D} stzy_i x_i + sty_i u_{i||D}} \right)^{\frac{1}{\langle \vec{x}, \vec{y} \rangle}} \\
&= M \cdot e(P_1, P_2)^{-stz} \left(e(P_1, P_2)^{stz \langle \vec{x}, \vec{y} \rangle} \right)^{\frac{1}{\langle \vec{x}, \vec{y} \rangle}} \\
&= M.
\end{aligned}$$

4.2 Proof of Security

We work in the selective model. For the proof, we need an additional assumption about the challenge vector $\vec{x}^* = (x_i^*)_{i \in D^*}$. In order to avoid $\langle \vec{x}^*, \vec{x}^* \rangle = 0$, we assume that $x_i^* \in \{0, 1, \ldots, L-1\}$ for a suitable value of L ensuring that the prime p is larger than $|D^*|.L^2$.

Theorem 1. *The proposed strict domain UNIPE scheme achieves selective security under the DBDH assumption in the random oracle model.*

Proof. Let $\mathscr{A}$ be an adversary that breaks the selective security of our scheme. We construct an algorithm $\mathscr{B}$ that breaks the DBDH assumption.

First, $\mathscr{B}$ receives a DBDH instance $([k]_1, [l]_1, [k]_2, [m]_2, [d]_T)$, $\mathscr{B}$'s objective is to guess whether $d = klm$ or d is randomly distributed in $\mathbb{Z}_p$. The adversary $\mathscr{A}$ sets the target attribute vector $\vec{x}^* = (x_i^*)_{i \in D^*}$ with target domain D^* and sends it to $\mathscr{B}$. Note that the $\mathscr{A}$ is restricted to make key queries on vector $\vec{y} = (y_i)_{i \in D'}$ such that $\langle \vec{x}^*, \vec{y} \rangle = 0$.

Let the cardinality of the target domain set be n, i.e., $|D^*| = n$ and $\Psi : D^* \to [n]$ be a function that maps the original indices to $\{1, 2, \ldots, n\}$. That is, we now work in the (isomorphic) space $\mathbb{Z}_p^n$ instead of $\{(w_i)_{i \in D^*} | w_i \in \mathbb{Z}_p\}$. Assume that $\left\{ (\vec{b}_j)_{j=1}^{n-1} \right\}$ to be a basis for $(\vec{x}^*)^{\perp}$. So, the family $\left\{ \vec{x}^*, (\vec{b}_j)_{j=1}^{n-1} \right\}$ is a basis for $\mathbb{Z}_p^n$, and each canonical vector can be expressed in the following form:

$$e_i = \alpha_i \cdot \vec{x}^* + \sum_{j \in [n-1]} \lambda_{i,j} \cdot \vec{b}_j$$

where $\alpha_i, \lambda_{i,j} \in \mathbb{Z}_p$. Also, $(n-1)$ random scalars $(v_1, \ldots v_{n-1}) \in \mathbb{Z}_p^{n-1}$ are chosen uniformly. $\mathscr{B}$ can now simulate the game in the following way:

Public Parameters: $\mathscr{B}$ fixes the public parameter $\mathsf{pp} = [k]_1$ and passes it to $\mathscr{A}$, implicitly setting the master secret key as k.

Random Oracle Calls: On any input $i||D$, if $D \neq D'$ or $i \notin D^*$, $\mathscr{B}$ picks a random group element $u_{i||D} \in \mathbb{Z}_p$, stores $i||D, u_{i||D}$ and sends $u_{i||D} P_2 \in \mathbb{G}_2$ to $\mathscr{A}$. Otherwise, it returns

$$[u_{i||D^*}]_2 = \alpha_{\Psi(i)} [m]_2 + \sum_{j \in [n-1]} \lambda_{\Psi(i),j} [v_j]_2$$

Challenge Ciphertext: $\mathscr{A}$ sends two challenge messages M^0 and M^1, to be encrypted under $\vec{x}^*$. A bit $\zeta \xleftarrow{\text{U}} \{0,1\}$ is chosen uniformly by $\mathscr{B}$ and the ciphertext for M^ζ is generated with the vector $\vec{x}^* = (x_i^*)_{i \in D^*}$. It sets $[t]_1 = [l]_1$ and

$$c_i^* = [d]_T^{x_i^* + \alpha_{\Psi(i)}} e([l]_1, [k]_2)^{\sum_{j \in [n-1]} \lambda_{\Psi(i),j} v_j}, \forall i \in D^*$$
$$\hat{c}^* = M^\zeta \cdot [d]_T^{-1}$$

$\mathscr{B}$ sends the ciphertext $C^* = ([l]_1, (c_i^*)_{i \in D^*}, \hat{c}^*)$ to $\mathscr{A}$.

Key Queries: On any input $y = (y_i)_{i \in D'}$, if $D' \neq D^*$, $\mathscr{B}$ returns

$$\left(y, -\sum_{i \in D'} u_{i||D'} y_i [k]_2 \right)$$

Otherwise $\mathscr{B}$ returns

$$sk_y = \left(y, -\left(\sum_{i \in D^*} y_i \left(\sum_{j \in [n-1]} \lambda_{\Psi(i),j} v_j \right) \right) [k]_2 \right)$$

Finally, $\mathscr{A}$ makes a guess $\zeta' \in \{0,1\}$ and if $\zeta' = \zeta$, $\mathscr{B}$ returns 1; else it returns 0.

$\mathscr{B}$ correctly simulates the game for $\mathscr{A}$ when given a true DBDH tuple. The public parameter is a uniform random element from group $\mathbb{G}_2$. The random oracle calls are responded with random elements of $\mathbb{G}_2$. We now show that the key queries are also perfectly simulated. Clearly, for the case when $D' \neq D^*$, the keys have the correct distribution. Otherwise, observe that $\alpha_i = \frac{x^*_{\Psi(i)}}{\langle \vec{x}^*, \vec{x}^* \rangle}$ and hence $\sum_{i \in D^*} y_i \alpha_{\Psi(i)} = 0$.

$$\begin{aligned}
-s \sum_{i \in D^*} y_i [u_{i||D^*}]_2 &= -k \sum_{i \in D^*} y_i \left(\alpha_{\Psi(i)} [m]_2 + \sum_{j \in [n-1]} \lambda_{\Psi(i),j} [v_j]_2 \right) \\
&= -\left(\sum_{i \in D^*} y_i \alpha_{\Psi(i)} m + \sum_{i \in D^*} y_i \sum_{j \in [n-1]} \lambda_{\Psi(i),j} v_j \right) [k]_2 \\
&= -\left(\sum_{i \in D^*} y_i \sum_{j \in [n-1]} \lambda_{\Psi(i),j} v_j \right) [k]_2
\end{aligned}$$

which is precisely what $\mathscr{B}$ computes.

Now for the challenge ciphertext, when $d = klm$, a legitimate encryption of M^ζ under $\vec{x}^* = (x_i^*)_{i \in D^*}$ is generated, implicitly setting $z = m$. On the other hand, if $\mathscr{B}$ is given a random tuple i.e., d is uniformly distributed in $\mathbb{Z}_p$, then the bit ζ is information-theoretically hidden from $\mathscr{A}$ in which case, $\mathscr{A}$'s probability of winning the game is exactly $1/2$. Therefore, we have

$$\begin{aligned}
\mathsf{Adv}^{\text{DBDH}}_{\mathscr{B},\mathcal{G}}(\vartheta) &= \big| \Pr[\mathscr{B}(\mathcal{G}, [k]_1, [l]_1, [k]_2, [m]_2, [klm]_T) = 1] \\
&\qquad - \Pr[\mathscr{B}(\mathcal{G}, [k]_1, [l]_1, [k]_2, [m]_2, [r]_T) = 1] \big| \\
&= |\Pr[\mathscr{A} \text{ wins} \mid d = klm] - \Pr[\mathscr{A} \text{ wins} \mid d = r]| \\
&= \left| \Pr[\zeta = \zeta' \text{ in the real game}] - \frac{1}{2} \right| \\
&= \mathsf{Adv}^{\mathsf{sel\text{-}IND}}_{\mathscr{A},\text{UNIPE}}(\vartheta)
\end{aligned}$$

5 Permissive UNIPE

5.1 Construction

Our UNIPE scheme in the permissive domain setting is defined by the following algorithms. We prove selective security from the DBDH assumption in the next subsection.

Setup(ϑ)**:** Choose a pairing $\mathcal{G} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, P_1, P_2, e) \longleftarrow \mathsf{GroupGen}(\vartheta)$. Choose $s \xleftarrow{\text{U}} \mathbb{Z}_p$ and a hash function $\mathcal{H} : \{0,1\}^* \to \mathbb{G}_2$. The public parameter is defined as $\mathsf{pp} = [s]_1$, and the master secret key is set as $\mathsf{msk} = (s, \mathsf{pp})$.

Encrypt$(\mathsf{pp}, M, \vec{x} = (x_i)_{i \in D})$**:** Takes the public parameter pp, the message M and an unbounded vector $\vec{x} = (x_i)_{i \in D}$ over some domain set D as input. Elements $z, t \xleftarrow{\text{U}} \mathbb{Z}_p$ are chosen uniformly. Set

$$c_i = e([s]_1, [t]_2)^{z x_i} e([s]_1, t[u_i]_2), \forall i \in D$$
$$\hat{c} = M \cdot e([s]_1, [t]_2)^{-z}$$

where $[u_i]_2 = \mathcal{H}(i)$. Finally, return the ciphertext $\mathsf{ct} = ([t]_1, \vec{x}, (c_i)_{i \in D}, \hat{c})$.

KeyGen$(\mathsf{msk}, \vec{y} = (y_i)_{i \in D'})$**:** Computes and returns

$$\mathsf{sk}_{\vec{y}} = \left(\vec{y}, -s \sum_{i \in D'} y_i [u_i]_2 \right)$$

Decrypt$(\mathsf{pp}, \mathsf{ct}, \mathsf{sk}_{\vec{y}})$**:** Takes pp, a ciphertext $\mathsf{ct} = ([t]_1, \vec{x} = (x_i)_{i \in D}, (c_i)_{i \in D}, \hat{c})$, and a secret key $sk_{\vec{y}} = (\vec{y}, f)$ as input. If the inner product of the associated vectors $\vec{x}$ and $\vec{y}$ is zero, i.e., $\langle \vec{x}, \vec{y} \rangle = 0$ then return $\perp$; otherwise, compute and output

$$M = \hat{c} \cdot \left(e([t]_1, f) \prod_{i \in D'} c_i^{y_i} \right)^{\frac{1}{\langle \vec{x}, \vec{y} \rangle}}.$$

Correctness: When $D' \subseteq D$ and $\langle \vec{x}, \vec{y} \rangle \neq 0$, we have

$$\begin{aligned}
&\hat{c} \cdot \left(e([t]_1, f) \prod_{i \in D'} c_i^{y_i} \right)^{\frac{1}{\langle \vec{x}, \vec{y} \rangle}} \\
&= \hat{c} \cdot \left(e\left([t]_1, -s \sum_{i \in D'} y_i [u_i]_2 \right) \prod_{i \in D'} e(P_1, P_2)^{stzy_i x_i + sty_i u_i} \right)^{\frac{1}{\langle \vec{x}, \vec{y} \rangle}} \\
&= \hat{c} \cdot \left(e(P_1, P_2)^{-st \sum_{i \in D'} y_i u_i} e(P_1, P_2)^{\sum_{i \in D'} stzy_i x_i + sty_i u_i} \right)^{\frac{1}{\langle \vec{x}, \vec{y} \rangle}} \\
&= M \cdot e(P_1, P_2)^{-stz} \left(e(P_1, P_2)^{stz \langle \vec{x}, \vec{y} \rangle} \right)^{\frac{1}{\langle \vec{x}, \vec{y} \rangle}} \\
&= M.
\end{aligned}$$

5.2 Security Proof

Again, we assume that for challenge vector $\vec{x}^* = (x_i^*)_{i \in D^*}$, $x_i^* \in \{0, 1, \ldots, L-1\}$ for a suitable L.

Theorem 2. *The proposed permissive domain UNIPE scheme achieves selective security under the DBDH assumption in the random oracle model.*

Proof. Let $\mathscr{A}$ be an adversary that breaks the selective security of our scheme. We construct an algorithm $\mathscr{B}$ that breaks the DBDH assumption.

First, $\mathscr{B}$ receives a DBDH instance $([k]_1, [l]_1, [k]_2, [m]_2, [d]_T)$, $\mathscr{B}$'s objective is to guess whether $d = klm$ or d is randomly distributed in $\mathbb{Z}_p$. The adversary $\mathscr{A}$ sets the target attribute vector $\vec{x}^* = (x_i^*)_{i \in D^*}$ with target domain D^* and sends it to $\mathscr{B}$. Note that the $\mathscr{A}$ is restricted to make key queries on vector $\vec{y} = (y_i)_{i \in D'}$ such that $\langle \vec{x}^*, \vec{y} \rangle = 0$.

Let the cardinality of the target domain set be n, i.e., $|D^*| = n$ and $\Psi : D^* \to [n]$ be a function that maps the original indices to $\{1, 2, \ldots, n\}$. That is, we now work in the (isomorphic) space $\mathbb{Z}_p^n$ instead of $\{(w_i)_{i \in D^*} | w_i \in \mathbb{Z}_p\}$. Assume that $\left\{(\vec{b}_j)_{j=1}^{n-1}\right\}$ to be a basis for $(\vec{x}^*)^{\perp}$. So, the family $\left\{\vec{x}^*, (\vec{b}_j)_{j=1}^{n-1}\right\}$ is a basis for $\mathbb{Z}_p^n$, and each canonical vector can be expressed in the following form:

$$e_i = \alpha_i \cdot \vec{x}^* + \sum_{j \in [n-1]} \lambda_{i,j} \cdot \vec{b}_j$$

where $\alpha_i, \lambda_{i,j} \in \mathbb{Z}_p$. Also, $(n-1)$ random scalars $(v_1, ...v_{n-1}) \in \mathbb{Z}_p^{n-1}$ are chosen uniformly. $\mathscr{B}$ can now simulate the game in the following way:

Public Parameters: $\mathscr{B}$ fixes the public parameter $\mathsf{pp} = [k]_1$ and passes it to $\mathscr{A}$, implicitly setting the master secret key as k.

Random Oracle Calls: On any input i, if $i \notin D^*$, $\mathscr{B}$ picks a random group element $u_i \in \mathbb{Z}_p$, stores i, u_i and sends $u_i P_2 \in \mathbb{G}_2$ to $\mathscr{A}$. Otherwise, it returns

$$[u_i]_2 = \alpha_{\Psi(i)}[m]_2 + \sum_{j \in [n-1]} \lambda_{\Psi(i),j}[v_j]_2$$

Challenge Ciphertext: $\mathscr{A}$ sends two challenge messages M^0 and M^1, to be encrypted under $\vec{x}^*$. A bit $\zeta \xleftarrow{\text{U}} \{0,1\}$ is chosen uniformly by $\mathscr{B}$ and the ciphertext for M^ζ is generated with the vector $\vec{x}^* = (x_i^*)_{i \in D^*}$. It sets $[t]_1 = [l]_1$ and

$$c_i^* = [d]_T^{x_i^* + \alpha_{\Psi(i)}} e([l]_1, [k]_2)^{\sum_{j \in [n-1]} \lambda_{\Psi(i),j} v_j}, \forall i \in D^*$$
$$\hat{c}^* = M^\zeta \cdot [d]_T^{-1}$$

$\mathscr{B}$ sends the ciphertext $C^* = ([l]_1, (c_i^*)_{i \in D^*}, \hat{c}^*)$ to $\mathscr{A}$.

Key Queries: On any input $y = (y_i)_{i \in D'}$, if $D' \cap D^* \neq D^*$ and $D' \cap D^* \neq D'$, $\mathscr{B}$ returns

$$\left(y, -\sum_{i \in D'} u_i y_i [k]_2\right)$$

If $D' \supset D^*$, $\mathscr{B}$ returns

$$\left(y, -\left(\sum_{i\in D^*} y_i \left(\sum_{j\in[n-1]} \lambda_{\Psi(i),j} v_j\right) + \sum_{i\in D'\setminus D^*} y_i u_i\right)[k]_2\right)$$

Otherwise $\mathscr{B}$ returns

$$sk_y = \left(y, -\left(\sum_{i\in D'} y_i \left(\sum_{j\in D'} \lambda_{\Psi(i),j} v_j\right)\right)[k]_2\right)$$

Finally, $\mathscr{A}$ makes a guess $\zeta' \in \{0,1\}$ and if $\zeta' = \zeta$, $\mathscr{B}$ returns 1; else it returns 0.

$\mathscr{B}$ correctly simulates the game for $\mathscr{A}$ when given a true DBDH tuple. The public parameter is a uniform random element from group $\mathbb{G}_2$. The random oracle calls are responded with random elements of $\mathbb{G}_2$. We now show that the key queries are also perfectly simulated. Clearly, for the case when $D' \cap D^* \neq D^*$ and $D' \cap D^* \neq D'$, the keys have the correct distribution. Now for the case $D' \supset D^*$, observe that $\alpha_i = \frac{x^*_{\Psi(i)}}{\langle \vec{x}^*, \vec{x}^* \rangle}$ and hence $\sum_{i\in D^*} y_i \alpha_{\Psi(i)} = 0$.

$$\begin{aligned}
-s\sum_{i\in D'} y_i [u_i]_2 &= -s\left(\sum_{i\in D^*} y_i [u_i]_2 + \sum_{i\in D'\setminus D^*} y_i [u_i]_2\right) \\
&= -k\left(\sum_{i\in D^*} y_i \left(\alpha_{\Psi(i)}[m]_2 + \sum_{j\in[n-1]} \lambda_{\Psi(i),j}[v_j]_2\right) + \sum_{i\in D'\setminus D^*} y_i [u_i]_2\right) \\
&= -\left(\sum_{i\in D^*} y_i \alpha_{\Psi(i)} m + \sum_{i\in D^*} y_i \sum_{j\in[n-1]} \lambda_{\Psi(i),j} v_j + \sum_{i\in D'\setminus D^*} y_i u_i\right)[k]_2 \\
&= -\left(\sum_{i\in D^*} y_i \sum_{j\in[n-1]} \lambda_{\Psi(i),j} v_j + \sum_{i\in D'\setminus D^*} y_i u_i\right)[k]_2
\end{aligned}$$

Otherwise,

$$\begin{aligned}
-s\sum_{i\in D'} y_i [u_{i||D'}]_2 &= -k\sum_{i\in D'} y_i \left(\alpha_{\Psi(i)}[m]_2 + \sum_{j\in D'} \lambda_{\Psi(i),j}[v_j]_2\right) \\
&= -\left(\sum_{i\in D'} y_i \alpha_{\Psi(i)} m + \sum_{i\in D'} y_i \sum_{j\in D'} \lambda_{\Psi(i),j} v_j\right)[k]_2 \\
&= -\left(\sum_{i\in D'} y_i \sum_{j\in D'} \lambda_{\Psi(i),j} v_j\right)[k]_2
\end{aligned}$$

which is precisely what $\mathscr{B}$ computes.

Now for the challenge ciphertext, when $d = klm$, a legitimate encryption of M^ζ under $\vec{x}^* = (x^*_i)_{i\in D^*}$ is generated, implicitly setting $z = m$. On the other hand, if $\mathscr{B}$ is given a random tuple

i.e., d is uniformly distributed in $\mathbb{Z}_p$, then the bit ζ is information-theoretically hidden from $\mathscr{A}$ in which case, $\mathscr{A}$'s probability of winning the game is exactly $1/2$. Therefore, we have

$$\begin{aligned}
\mathsf{Adv}^{\mathrm{DBDH}}_{\mathscr{B},\mathcal{G}}(\vartheta) &= \big|\Pr[\mathscr{B}(\mathcal{G}, [k]_1, [l]_1, [k]_2, [m]_2, [klm]_T) = 1] \\
&\qquad - \Pr[\mathscr{B}(\mathcal{G}, [k]_1, [l]_1, [k]_2, [m]_2, [r]_T) = 1]\big| \\
&= |\Pr[\mathscr{A} \text{ wins} \mid d = klm] - \Pr[\mathscr{A} \text{ wins} \mid d = r]| \\
&= \left|\Pr[\zeta = \zeta' \text{ in the real game}] - \frac{1}{2}\right| \\
&= \mathsf{Adv}^{\mathsf{sel\text{-}IND}}_{\mathscr{A},\mathrm{UNIPE}}(\vartheta)
\end{aligned}$$

6 Comparison with Existing Constructions

A comparison of our schemes with prior pairing-based NIPE schemes is provided in Table 1. The works [22], [8] each present two NIPE constructions, one with constant-size ciphertexts and the other with constant-size keys named 1 and 2 respectively. [9] describes two NIPE schemes, one using asymmetric prime-order pairings and the other in the setting of composite-order pairings, named 1 and 2 respectively.

Note that none of the prior works consider unbounded vectors. Among the ones that achieve succinct keys, [8]-2 improves on [22]-2 in terms of efficiency. We highlight the constants in the running time of encryption and decryption algorithms to indicate that the constructions in [8] are more efficient. Both have public parameters and ciphertexts of size linear in the vector-length. On the other hand they achieve adaptive security. The constructions in [9] are comparable to ours in terms of secret key size and security guarantees though in addition they have succint ciphertexts as well. Since they support only fixed length vectors, it is but natural that public parameters are linear sized. However, the prime-order construction relies on a parameterized assumption while security of our scheme relies on a static assumption albeit in the random oracle model. The composite-order construction, on the other hand, relies on the static subgroup decision assumptions. It would be interesting to see if prior techniques are useful in constructing unbounded NIPE without domain restrictions or random oracles and with enhanced security guarantees.

Scheme	Pairing	Unbounded	#pp	#cpr	#key	#enc	#dec	Security	Assumptions
[3]	Symmetric	No	$(n+11)S_1 + S_T$	$9S_1 + S_T$	$(n+6)S_1$	$O(n)[M_1] + [E_T]$	$O(n)[M_1] + 9[P]$	Co-Selective	D-Lin, DBDH
[22]-1	Asymmetric	No	$(8n+23)S_1$	$13S_1 + S_T$	$(4n+5)S_2$	$(2n+17)[M_1] + [E_T]$	$(4n-4)[M_2] + 13[P]$	Adaptive	D-Lin
[22]-2	Asymmetric	No	$(8n+23)S_1$	$(4n+5)S_1 + S_T$	$13S_2$	$(8n^2+15)[M_1] + [E_T]$	$(4n-4)[M_1] + 13[P]$	Adaptive	D-Lin
[26]	Symmetric	No	$(n+2)S_1 + S_T$	$2S_1 + S_T$	$(n+2)S_1$	$O(n)[M_1] + [P] + [E_T]$	$O(n)[M_1] + 3[P] + [E_T]$	Selective	n-DBDHE
[8]-1	Asymmetric	No	$(6n+2)S_1 + 2S_T$	$6S_1 + S_T$	$(3n+3)S_2$	$3(n+1)[M_1] + [E_T] + [P]$	$[E_T] + 3n[M_2] + 6[P]$	Adaptive	D-Lin
[8]-2	Asymmetric	No	$(6n+8)S_1 + 2S_T$	$(3n+3)S_1 + S_T$	$9S_2$	$6n[M_1] + [E_T] + [P]$	$[E_T] + 3n[M_1] + 6[P]$	Adaptive	D-Lin
[9]-1	Asymmetric	No	$(4n+1)S_1$	$2S_1 + S_T$	S_2	$O(n)[M_1] + [E_T] + [P]$	$O(n)[M_2] + [E_T] + 2[P]$	Selective	n-DBDHE
[9]-2	Composite-Order	No	$(3n+1)S_1$	$2S_1 + S_T$	S_1	$O(n)[M_1] + [E_T] + [P]$	$O(n)[M_1] + [E_T] + 2[P]$	Selective	Subgroup
This work	Asymmetric	Yes	S_1	$(n+1)S_T$	S_2	$(2n_c+1)[P] + [E_T]$	$(n_k+1)[E_T] + [P]$	Selective	DBDH

Table 1: Comparison of our scheme with previously known pairing-based NIPE schemes.

We use some additional notation for the comparison. #pp, #cpr, #key refer to sizes of public parameters, ciphertext and key repectively. #enc, #dec refer to running time of encryption, decryption algorithms respectively. n denotes the lengths of vectors in the bounded length setting. In

the unbounded case, n_c, n_k denote the lengths of vectors associated to the ciphertext, key respectively. S_ι denotes the size of representation of elements from $\mathbb{G}_\iota$ for $\iota \in \{1, 2, T\}$. $[M_\iota]$ denotes the cost of scalar multiplication in group $\mathbb{G}_\iota$ for $\iota \in \{1, 2\}$ and $[E_T]$ is the cost of exponentiation in $\mathbb{G}_T$. Let $[P]$ denote the cost of pairing computation. Note that in case of symmertic pairings (and composite order symmetric pairings), $\mathbb{G}_1 = \mathbb{G}_2$. In the table, we write symmetric/asymmetric for the prime-order pairing setting; it is assumed that composite-order pairings are symmetric.

7 Conclusion

This work proposes the first construction for unbounded NIPE in both strict and permissive domain settings. Both schemes achieve efficiency in terms of size of the secret key, specified by a single element from $\mathbb{G}_2$. Many interesting problems remain open, such as constructing unbounded NIPE with short ciphertexts and/or adaptive security without random oracles. Another direction for future research is to remove the domain restrictions altogether.

Acknowledgments

The first author expresses thanks to University Grants Commission (UGC), India for their support.

References

1. M. Abdalla, F. Bourse, A. De Caro, D. Pointcheval. Simple Functional Encryption Schemes for Inner Products. In *PKC 2015*, *LNCS* 6056, pp. 733–751. Springer, 2015.
2. S. Agrawal, D. Freeman, V. Vaikuntanathan. Functional Encryption for Inner Product Predicates from Learning with Errors. In *Asiacrypt 2011*, *LNCS* 7073, pp. 21–40. Springer, 2011.
3. N. Attrapadung, B. Libert. Functional encryption for inner product: Achieving constant-size ciphertexts with adaptive security or support for negation. In *PKC 2010*, *LNCS* 6056, pp. 384–402. Springer Berlin Heidelberg, 2010.
4. N. Attrapadung, B. Libert, E. De Panafieu. Expressive key-policy attribute-based encryption with constant-size ciphertexts. In *PKC 2011*, *LNCS* 6571, pp. 90–108
5. S. Agrawal, B. Libert, D. Stehlé. Fully Secure Functional Encryption for Inner Products, from Standard Assumptions. Cryptology ePrint Archive: Report 2015/608, 2015.
6. A. Bishop, A. Jain, L. Kowalczyk. Function-Hiding Inner Product Encryption. In *Asiacrypt'15*, *LNCS* 9452, pp. 470–491, 2015.
7. D. Boneh, A. Sahai and B. Waters. Functional encryption: Definitions and challenges. In *TCC 2011 LNCS* 6597, pp. 253-273, 2011.
8. J. Chen, R. Gay, H. Wee. Improved Dual System ABE in Prime-Order Groups via Predicate Encodings. In *Eurocrypt 2015 (2)*, *LNCS* 9057, pp. 595–624
9. J. Chen, B. Libert, S. Ramanna. Non-Zero Inner Product Encryption with Short Ciphertexts and Private Keys. In *SCN 2016*, *LNCS* 9841, pp. 23–41, 2016.
10. J. Chen, H. Wee. Doubly spatial encryption from DBDH. In *Theor. Comput. Sci.*543:79–89, 2014.
11. U. Dowerah, S. Dutta., A. Mitrokotsa, S. Mukherjee, T. Pal. Unbounded Predicate Inner Product Functional Encryption from Pairings. In *Journal of Cryptology* 36(3), pp. 29, 2023.
12. P. Datta, T. Okamoto, J. Tomida. Full-Hiding (Unbounded) Multi-input Inner Product Functional Encryption from the k-Linear Assumption. In *PKC 2018* part II, *LNCS* 10770, pp. 245–277, 2018.
13. S. Dutta, T. Tapas Pal, and R. Dutta. Fully Secure Unbounded Zero Inner Product Encryption with Short Ciphertexts and Keys. In *ProvSec 2021*, *LNCS* 13059, pp. 241–258, 2021.
14. E. Dufour-Sans and D. Pointcheval. Unbounded Inner-Product Functional Encryption with Succinct Keys. In *ACNS 2019*, *LNCS* 12726, pp. 426–441, 2019.
15. V. Goyal, O. Pandey, A. Sahai, B. Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *ACM CCS'06*, pp. 89–98, 2006.

16. J. Katz, A. Sahai, B. Waters. Predicate Encryption Supporting Disjunctions, Polynomial Equations, and Inner Products. In *Eurocrypt'08*, *LNCS* 4965, pp. 146-162.
17. A. Lewko, T. Okamoto, A. Sahai, K. Takashima, B. Waters. Fully Secure Functional Encryption: Attribute-Based Encryption and (Hierarchical) Inner Product Encryption. In *Eurocrypt 2010*, *LNCS* 6110, pp. 62–91, 2010.
18. T. Okamoto, K. Takashima. Hierarchical Predicate Encryption for Inner-Products.In *Asiacrypt'09*, *LNCS* 5912, pp. 214–231, 2009.
19. T. Okamoto, K. Takashima. Fully secure functional encryption with general relations from the decisional linear assumption. In *Crypto'10*, *LNCS* 6223, pp. 191–208, 2010.
20. T. Okamoto, K. Takashima. Adaptively Attribute-Hiding (Hierarchical) Inner Product Encryption. In *Eurocrypt'12*, *LNCS* 7237, pp. 591–608, 2012.
21. T. Okamoto, K. Takashima. Fully Secure Unbounded Inner-Product and Attribute-Based Encryption. In *Asiacrypt'12*, *LNCS* 7658, pp. 349–366, 2012.
22. T. Okamoto, K. Takashima. Achieving short ciphertexts or short secret-keys for adaptively secure general inner-product encryption. Designs, Codes and Cryptography 77.2-3 (2015): 725–771.
23. A. Sahai, B. Waters. Fuzzy Identity-Based Encryption In *Eurocrypt'05*, *LNCS* 3494, pp. 457–473, 2005.
24. E. Shen, E. Shi, B. Waters. Predicate Privacy in Encryption Systems. In *TCC'09*, *LNCS* 5444, pp. 457–473, 2009.
25. J. Tomida and K. Takashima. Unbounded Inner Product Functional Encryption from Bilinear Maps. In *ASIACRYPT 2018*, *LNCS* 11274, pp. 723-779, 2018.
26. S. Yamada, N. Attrapadung, G. Hanaoka, N. Kunihiro. A Framework and Compact Constructions for Non-monotonic Attribute-Based Encryption. In *PKC 2014*, *LNCS* 8383, pp. 275–292, 2014.
27. Katsumata, S., Yamada, S. (2019). Non-zero Inner Product Encryption Schemes from Various Assumptions: LWE, DDH and DCR. In: Lin, D., Sako, K. (eds) Public-Key Cryptography â€“ PKC 2019.