PAPER Special Section on Security, Privacy and Anonymity in Computation, Communication and Storage Systems

Private Similarity Searchable Encryption for Euclidean Distance*

Yuji UNAGAMI[†a)], ***Nonmember***, **Natsume MATSUZAKI**[††], ***Member***, **Shota YAMADA**[†††], **Nuttapong ATTRAPADUNG**[†††], **Takahiro MATSUDA**[†††], ***Nonmembers***, ***and*** **Goichiro HANAOKA**[†††], ***Member***

SUMMARY In this paper, we propose a similarity searchable encryption in the symmetric key setting for the weighted Euclidean distance, by extending the functional encryption scheme for inner product proposed by Bishop et al. [4]. Our scheme performs predetermined encoding independently of vectors **x** and **y**, and it obtains the weighted Euclidean distance between the two vectors while they remain encrypted.

key words: *searchable encryption, inner product encryption, the weighted Euclidean distance*

1. Introduction

Searchable encryption is a new paradigm, which allows similar data to be searched within an encrypted database. Most prior studies on searchable encryption [6], [8] have focused on searching exactly-matching data. We are, however, interested in searching the similar data of a certain distance.

Bishop et al. [4] recently proposed a new functional encryption (FE) scheme for inner product in the symmetric key setting. Their scheme uses asymmetric bilinear maps, and is secure against unbounded collusion under a simple assumption. They focused on function privacy in FE for inner product. Intuitively speaking, function privacy requires that given a decryption key K_f for a function f, one should not be able to learn any unnecessary information about f.

Besides inner product, the Euclidean distance is very commonly used to measure the distance. Oosawa et al. [17] proposed a system called a SYNAPSE Case Matching which is a content-based image retrieval system that supports lung cancer image diagnosis. In their system, Euclidean distance is used to measure the distance between a patient's lung image and each data in the medical case database, wherein the image has multiple parameters, such as color, figure, and size.

One scenario we consider is that a user searches similar symptom data of a certain Euclidean distance from an encrypted medical case database, providing his/her own encrypted user's information as a query to the Cloud. Using similar symptom data, he/she can receive useful advice for improving health. Another scenario is to analyze users' behavior using GPS information from their smartphone. In these scenarios, not only the user's information and the GPS information, it is necessary to protect but also the database for user's privacy.

1.1 Our Results

In this paper, we define the notion of similarity searchable encryption for the weighted Euclidean distance, where the weighted Euclidean distance is a slight extension of the Euclidean distance. In similarity searchable encryption for the weighted Euclidean distance, we consider two objects: queries and encrypted data. Our notion is considered in the symmetric key setting in the sense that both of queries and encrypted database cannot be generated without a master secret key. Using queries, we can search on the encrypted database for similar data. Here, we adopt the weighted Euclidean distance as an index of similarity. In our security notion, we require that both queries and encrypted database do not reveal any information more than necessary.

To obtain a scheme satisfying our requirements, we show a generic construction of similarity searchable encryption for the weighted Euclidean distance from any functional encryption for inner product (with function privacy). By starting from the functional encryption schemes for inner product in the literature [4], [9], we can obtain similarity searchable encryption schemes for the weighted Euclidean distance.

In Table 1, we compare our approach with existing approaches of securely computing the Euclidean distance. As shown in Table 1, general Fully Homomorphic Encryption [12] and Garbled Circuit [19] are known as being inefficient. Our proposal does not require the interaction for the

Table 1 Comparison of the secure Euclidean distance computation.

	Efficiency	Interaction	Input
D01 [10], LL04 [15]	High	Necessary	Revealed
G09 [12]	Low	Necessary	Revealed
Y86 [19]	Low	Necessary	Revealed
Our Scheme	High	Unnecessary	Encrypted

Manuscript received December 15, 2016.
Manuscript revised May 24, 2017.
Manuscript publicized July 21, 2017.

[†]The author is with Advanced Research Division, Panasonic Corporation, Kadoma-shi, 571–8501 Japan.
[††]The author is with the Faculty of Information Systems, University of Nagasaki, Nagasaki-ken, 851–2195 Japan.
[†††]The authors are with Information Technology Research Institute, National Institute of Advance Industrial Science and Technology, Tokyo, 135–0064 Japan.

*A preliminary version of this paper appears in Proceedings of the International Symposium on Information Theory and Its Applications (ISITA), 2016. See [1].

a) E-mail: unagami.yuji@jp.panasonic.com

DOI: 10.1587/transinf.2016INP0011

weighted Euclidean distance calculation. Additionally, it is a cyphertext for the input of the weighted Euclidean distance calculation in our proposal. On the other hand, against other works which leak no information, our proposal leak only the value of the weighted Euclidean distance.

1.2 Related Works

A similar line of study is searchable encryption, which allows one to search on encrypted database. Most prior studies on searchable encryption [6], [8] have focused mainly on searching the database to find an exact match to the query.

Subsequent works [3], [7] enabled searching on encrypted data with more complicated conditions. In our work, we focus on rather different form of queries. That is, we search on encrypted data to find data that is within certain distance from the query. Another difference from these works is that we consider the privacy of the queries.

In our conversion, we require certain form of functional encryption. The notion of functional encryption was proposed in the work of [5]. Later, Garg et al. proposed a construction of functional encryption based on indistinguishability obfuscation [11]. Since the current candidate constructions for indistinguishability obfuscation are extremely inefficient, their scheme is not practical. Subsequently, functional encryption for inner product, which is a special case of the more general notion of functional encryption, was proposed by Abdalla et al. [2]. Their construction is considered in the public key setting and they do not consider function privacy, which means that an attribute associated to a key can be leaked. Later, functional encryption for inner product in the symmetric key setting with function privacy was proposed [4]. Very recently, in the subsequent work [9], a scheme satisfying a stronger security notion was proposed.

Prior studies [10], [15] took a similar approach to our scheme, which uses inner product encryption to compute the Euclidean distances. However, their schemes require multi-round transactions, which is not needed in our scheme.

We note that essentially the same encoding as ours was used in the work of Guo et al. [13] who constructed predicate encryption that can deal with the Euclidean distances by incorporating a predicate encryption for inner product with the encoding. A crucial difference from their work is that in our work, we deal with *secret key functional encryption with anonymity and function privacy*, whereas they consider *public key predicate encryption without them*. Function privacy is necessary for our application to searchable encryption.

2. Preliminaries

2.1 Inner Product Encryption

Inner product encryption, which is also called functional encryption for inner product, is a special case of functional encryption [11]. In inner product encryption, secret keys and ciphertexts are associated with vectors. When one decrypts a ciphertext using a secret key, one obtains the inner product of these vectors. Inner product encryption is defined as follows.

Definition: Inner product Encryption
Let p and n be integers that depend on the security parameter. We note that p and n corresponds to the modulus and the dimension of the vector space on which we consider the computation of the inner product. Let $\mathbf{x}$ and $\mathbf{y}$ be vectors in $\mathbb{Z}_p^n$:

$$\mathbf{x} = (x_0, \ldots, x_{n-1}) \in \mathbb{Z}_p^n, \quad \mathbf{y} = (y_0, \ldots, y_{n-1}) \in \mathbb{Z}_p^n$$

Inner product encryption consists of the following four algorithms: IPE.Setup, IPE.Encrypt, IPE.KeyGen, and IPE.Decrypt. We note that the length of the bit representation of p is bounded by some polynomial of the security parameter λ.

$\mathsf{IPE.Setup}(1^\lambda, 1^n) \to (\mathsf{pp}, \mathsf{msk})$
The setup algorithm takes the security parameter λ and the length of vectors n as input, and outputs a public parameters pp and a master secret key msk.

$\mathsf{IPE.Encrypt}(\mathbf{x}, \mathsf{msk}, \mathsf{pp}) \to C_{\mathbf{x}}$
The encryption algorithm takes a vector $\mathbf{x} \in \mathbb{Z}_p^n$, the master secret key msk, and the public parameters pp as input, and outputs a ciphertext $C_{\mathbf{x}}$.

$\mathsf{IPE.KeyGen}(\mathbf{y}, \mathsf{msk}, \mathsf{pp}) \to K_{\mathbf{y}}$
The key generation algorithm takes the vector $\mathbf{y} \in \mathbb{Z}_p^n$, the master secret key msk, and the public parameters pp as input, and outputs a decryption key $K_{\mathbf{y}}$.

$\mathsf{IPE.Decrypt}(C_{\mathbf{x}}, K_{\mathbf{y}}, \mathsf{pp}) \to m$
The decryption algorithm takes a ciphertext $C_{\mathbf{x}}$, the decryption key $K_{\mathbf{y}}$, and the public parameters pp as input, and outputs m.

For Correctness, we require the following.

Correctness: We assume that $(\mathsf{pp}, \mathsf{msk})$ is the output of $\mathsf{IPE.Setup}(1^\lambda, 1^n)$, $C_{\mathbf{x}}$ is the output of $\mathsf{IPE.Encrypt}(\mathbf{x}, \mathsf{msk}, \mathsf{pp})$, and $K_{\mathbf{y}}$ is the output of $\mathsf{IPE.KeyGen}(\mathbf{y}, \mathsf{msk}, \mathsf{pp})$. We require the output m of $\mathsf{IPE.Decrypt}(C_{\mathbf{x}}, K_{\mathbf{y}}, \mathsf{pp})$ be the inner product of $\mathbf{x}$ and $\mathbf{y}$. Namely, we require $m = \langle \mathbf{x}, \mathbf{y} \rangle = \sum_{i=0}^{n-1} x_i y_i$.

Security Definition: We present here a summary of the security definition of inner product encryption. The definition of security states that a decryption key $K_{\mathbf{y}}$ and a ciphertext $C_{\mathbf{x}}$ do not reveal any information about $\mathbf{x}$, $\mathbf{y}$. We define security using the following game between a challenger C and an adversary $\mathcal{A}$.

Setup_game_IPE:
C runs IPE.Setup to generate msk and pp. It gives pp to $\mathcal{A}$. C also picks a random bit $b \in \{0, 1\}$.

Challenge*1_*IPE:
$\mathcal{A}$ sends C two vectors $\mathbf{x}_0, \mathbf{x}_1$ on which it wishes to be challenged. C picks $\mathbf{x}_b$ based on b selected in *Setup_game_IPE*. C runs $\mathsf{IPE.Encrypt}(\mathbf{x}_b, \mathsf{msk}, \mathsf{pp})$ to generate $C_{\mathbf{x}_b}$. It gives $C_{\mathbf{x}_b}$ to $\mathcal{A}$.

Challenge*2_*IPE:
$\mathcal{A}$ sends C two vectors $\mathbf{y}_0, \mathbf{y}_1$ on which it wishes to be challenged. C picks $\mathbf{y}_b$ based on the b selected in *Setup_game_IPE*. C runs $\mathsf{IPE.KeyGen}(\mathbf{y}_b, \mathsf{msk}, \mathsf{pp})$ algorithm to generate $K_{\mathbf{y}_b}$. It gives $K_{\mathbf{y}_b}$ to $\mathcal{A}$.

$\mathcal{A}$ can adaptively ask the challenger for above queries in arbitrary many times and an arbitrarily order. However, we require that $\langle \mathbf{x}_0^{(i)}, \mathbf{y}_0^{j} \rangle = \langle \mathbf{x}_1^{(i)}, \mathbf{y}_1^{(j)} \rangle$ for all i and j, where where $\mathbf{x}_0^{(i)}, \mathbf{x}_1^{(i)}$ (resp. $\mathbf{y}_0^{(j)}, \mathbf{y}_1^{(j)}$) are the vectors corresponding to the i-th (resp. j-th) query in the ***Challenge*1_*IPE*** phase (resp. ***Challenge*2_*IPE*** phase).

Guess_IPE:
$\mathcal{A}$ outputs b' and wins the game if $b = b'$. We define $\mathcal{A}$'s advantage in breaking the inner product encryption scheme as

$$Adv = \left| \Pr(b' = b) - \frac{1}{2} \right|.$$

We say that the inner product encryption scheme satisfies *full privacy* if the advantage of the adversary $\mathcal{A}$ is negligible in the security parameter λ.

Intuitively, the above security definition ensures that the information of the vectors corresponding to the ciphertexts and secret keys do not leak more than necessary. In this sense, the above definition captures both anonymity (i.e. the vectors $\mathbf{x}$s do not leak from the ciphertexts) and the function privacy (i.e. the vectors $\mathbf{y}$s do not leak from the decryption keys) at the same time.

Weaker Security Notion. We can consider a weaker security notion in which the queries of an adversary are restricted to satisfy

$$\langle \mathbf{x}_0^{(i)}, \mathbf{y}_0^{(j)} \rangle = \langle \mathbf{x}_0^{(i)}, \mathbf{y}_1^{(j)} \rangle = \langle \mathbf{x}_1^{(i)}, \mathbf{y}_1^{(j)} \rangle = \langle \mathbf{x}_1^{(i)}, \mathbf{y}_0^{(j)} \rangle$$

for all i and j. If the advantage of any adversary $\mathcal{A}$ is negligible in this (modified) game, we say that the scheme satisfies *weak privacy*.

2.2 Weighted Euclidean Distance

The weighted Euclidean distance is a generalization of the ordinary Euclidian distance and parametrized by $\{w_i\}_{i=1}^n$. The weighted Euclidean distance between the vector $\mathbf{x} \in \mathbb{Z}_p^n$ and $\mathbf{y} \in \mathbb{Z}_p^n$ is defined as the the square root of dist, which is defined as follows.

$$\mathsf{dist}(\mathbf{x}, \mathbf{y}) = \sum_{i=0}^{n-1} w_i (x_i - y_i)^2$$

3. Private Similarity Searchable Encryption Specifications and Security Definitions

3.1 Model

Let us consider the following scenario, which is a use case of our scheme proposed in this paper. We will consider a system that consists of a user, a server, and a database owner. The user generates a query and encrypts the query. The database owner encrypts a reference record in the database and sends the encrypted reference record to the server. The server extracts the similar data from the encrypted database using the encrypted query and the encrypted reference record. We adopt the weighted Euclidean distance as an index of similarity. The server obtains the weighted Euclidean distance between the query and the reference record while the query and the reference record remain encrypted. It is necessary to ensure the confidentiality of both of the query and of the reference record in the server. To capture this scenario, we propose the notion of similarity searchable encryption, which is defined as follows.

Definition: Private Similarity Searchable Encryption
The query $\mathbf{x}$ and the reference record $\mathbf{y}$ are n-length vectors over a finite field $\mathbb{Z}_p$. We note that the length of the bit representation of p is bounded by some polynomial of the security parameter λ.

$$\mathbf{x} = (x_0, \ldots, x_{n-1}) \in \mathbb{Z}_p^n, \quad \mathbf{y} = (y_0, \ldots, y_{n-1}) \in \mathbb{Z}_p^n$$

Let a vector $\mathbf{w} = (w_1, \ldots, w_n) \in \mathbb{Z}^n$ be the weight of the Euclidean distance. A similarity searchable encryption scheme consists of the following four algorithms, Setup, Query, EncDB, and Dist. In the following, we implicitly assume that all these algorithms take $\mathbf{w}$ as an additional input.

$\mathsf{Setup}(1^\lambda, 1^n) \to (\mathsf{pp}, \mathsf{msk})$
The setup algorithm takes the security parameter λ, and the length of vectors n as input, and outputs public parameters pp and a master secret key msk.

$\mathsf{Query}(\mathbf{x}, \mathsf{msk}, \mathsf{pp}) \to Q_{\mathbf{x}}$
The query algorithm takes a vector $\mathbf{x} \in \mathbb{Z}_p^n$, the master secret key msk, and the public parameters pp as input, and outputs a query $Q_{\mathbf{x}}$.

$\mathsf{EncDB}(\mathbf{y}, \mathsf{msk}, \mathsf{pp}) \to D_{\mathbf{y}}$
The database encryption algorithm takes a vector $\mathbf{y} \in \mathbb{Z}_p^n$, the master secret key msk, and the public parameters pp as input, and outputs an encrypted record $D_{\mathbf{y}}$.

$\mathsf{Dist}(Q_{\mathbf{x}}, D_{\mathbf{y}}, \mathsf{pp}) \to Z$
The distance measurement algorithm takes the query $Q_{\mathbf{x}}$, the encrypted record $D_{\mathbf{y}}$, and the public parameters pp as input, and outputs the weighted Euclidean distance Z.

For correctness, we require the following.

Correctness: We assume that $(\mathsf{pp}, \mathsf{msk})$ is the output of $\mathsf{Setup}(1^\lambda, 1^n)$, and $Q_{\mathbf{x}}$ and $D_{\mathbf{y}}$ are the output of $\mathsf{Query}(\mathbf{x}, \mathsf{msk}, \mathsf{pp})$ and $\mathsf{EncDB}(\mathbf{y}, \mathsf{msk}, \mathsf{pp})$, respectively. We require the output Z of $\mathsf{Dist}(Q_{\mathbf{x}}, D_{\mathbf{y}}, \mathsf{pp})$ be $\mathsf{dist}(\mathbf{x}, \mathbf{y})$.

3.2 Security Definitions

Here we define security for similarity searchable encryption. We need to guarantee that queries $\{Q_{\mathbf{x}}\}$ and encrypted records $\{D_{\mathbf{y}}\}$ do not reveal any information beyond their weighted Euclidian distances. We define the security using the following game between a challenger C and an adversary $\mathcal{A}$.

Weight_selection:
At the outset of the game, $\mathcal{A}$ is given 1^λ and 1^n as input. $\mathcal{A}$ then chooses a weight vector $\mathbf{w} = (w_1, \ldots, w_n)$ at his will and gives it to C. The vector $\mathbf{w}$ specifies the Euclidean distance and fixed throughout the game. In the following, the function Dist refers to the weighted Euclidean distance with respect to the weight.

Setup_game:
C runs Setup to generate msk and pp. Note that as mentioned above, C has to use the fixed $\mathbf{w}$ that was chosen by $\mathcal{A}$. C gives pp to $\mathcal{A}$. C also picks a random $b \in \{0, 1\}$.

***Challenge*1**:
$\mathcal{A}$ sends C two vectors $\mathbf{x}_0, \mathbf{x}_1 \in \mathbb{Z}_p^n$ on which it wishes to be challenged. C picks $\mathbf{x}_b$ based on the challenge bit b selected in the *Setup_game*. Then C runs $\mathsf{Query}(\mathbf{x}_b, \mathsf{msk}, \mathsf{pp})$ to generate $Q_{\mathbf{x}_b}$. C gives $Q_{\mathbf{x}_b}$ to $\mathcal{A}$. Notice that $\mathcal{A}$'s query only contains $\mathbf{x}_0$ and $\mathbf{x}_1$, but not $\mathbf{w}$ since $\mathbf{w}$ was already sent by $\mathcal{A}$ in the ***Weight_selection*** phase.

***Challenge*2**:
$\mathcal{A}$ sends C two vectors $\mathbf{y}_0, \mathbf{y}_1$ on which it wishes to be challenged. C picks $\mathbf{y}_b$ based on the b selected in the *Setup_game*. C runs $\mathsf{EncDB}(\mathbf{y}_b, \mathsf{msk}, \mathsf{pp})$ to generate $D_{\mathbf{y}_b}$. C gives $D_{\mathbf{y}_b}$ to $\mathcal{A}$.

$\mathcal{A}$ can adaptively ask C for the above queries with the following constraint:

$$\mathsf{dist}(\mathbf{x}_0^{(i)}, \mathbf{y}_0^{(j)}) = \mathsf{dist}(\mathbf{x}_1^{(i)}, \mathbf{y}_1^{(j)}),$$

for all i and j, where $\mathbf{x}_0^{(i)}, \mathbf{x}_1^{(i)}$ (resp. $\mathbf{y}_0^{(j)}, \mathbf{y}_1^{(j)}$) are the vectors corresponding to the i-th (resp. j-th) query in the *Challenge*1 phase (resp. *Challenge*2 phase).

Guess:
$\mathcal{A}$ outputs b' and wins the game if $b = b'$. We define $\mathcal{A}$'s advantage in breaking the security of the similarity searchable encryption scheme as

$$Adv = \left| \Pr(b' = b) - \frac{1}{2} \right|.$$

We say that the similarity searchable encryption scheme achieves *full privacy* if the advantage of the adversary $\mathcal{A}$ is negligible in the security parameter λ.

Weaker Security Notion. We can consider a weaker security notion in which the queries of the adversary are restricted to satisfy

$$\mathsf{dist}(\mathbf{x}_0^{(i)}, \mathbf{y}_0^{(j)}) = \mathsf{dist}(\mathbf{x}_0^{(i)}, \mathbf{y}_1^{(j)}) = \mathsf{dist}(\mathbf{x}_1^{(i)}, \mathbf{y}_1^{(j)}) = \mathsf{dist}(\mathbf{x}_1^{(i)}, \mathbf{y}_0^{(j)})$$

for all i and j. If the advantage of the adversary is negligible in this (modified) game, we say that the scheme satisfies *weak privacy*.

4. Construction

4.1 Construction Using Inner Product Encryption

In this section, we show a generic construction of a similarity searchable encryption scheme for the weighted Euclidean distance (Setup, Query, EncDB, Dist) from an inner product encryption scheme ($\mathsf{IPE.Setup}$, $\mathsf{IPE.KeyGen}$, $\mathsf{IPE.Encrypt}$, $\mathsf{IPE.Decrypt}$). The conversion is completely generic and based on the idea of encoding vectors so that their inner product corresponds to the (square of) distance between them. In the following, let $\mathbf{w} = (w_1, \ldots, w_n)$ be the weight vector.

$\mathsf{Setup}(1^\lambda, 1^n)$:
The setup algorithm takes the security parameter λ and the length of vectors n as input, and runs $\mathsf{IPE.Setup}(1^\lambda, 1^{n+2})$ to obtain $(\mathsf{pp}, \mathsf{msk})$. It outputs public parameters pp and a master secret key msk.

$\mathsf{Query}(\mathbf{x}, \mathsf{msk}, \mathsf{pp})$:
The query algorithm takes a vector $\mathbf{x} \in \mathbb{Z}_p^n$, the master secret key msk, and the public parameters pp as input. It first applies the encoding algorithm Encode_1, which is defined in the following, to $\mathbf{x}$.

$\mathsf{Encode}_1 : \mathbf{x} = (x_0, \ldots, x_{n-1}) \mapsto \mathbf{x}' = (x'_0, \ldots, x'_{n+1})$, where

$$\begin{aligned} x'_0 &= \sum_{i=0}^{n-1} w_i x_i^2, \\ x'_1 &= 1, \\ x'_2 &= -2w_0 x_0, \\ &\vdots \\ x'_{n+1} &= -2w_{n-1} x_{n-1}. \end{aligned}$$

Then, it runs $\mathsf{IPE.KeyGen}(\mathbf{x}', \mathsf{msk}, \mathsf{pp}) \to Q_{\mathbf{x}}$ and outputs $Q_{\mathbf{x}}$.

$\mathsf{EncDB}(\mathbf{y}, \mathsf{msk}, \mathsf{pp})$:
The database encryption algorithm takes a vector $\mathbf{y} \in \mathbb{Z}_p^n$, the master secret key msk, and the public parameters pp as input. It first applies the encoding algorithm Encode_2,

which is defined in the following, to the vector $\mathbf{y}$.

$\mathsf{Encode}_2 : \mathbf{y} = (y_0, \ldots, y_{n-1}) \mapsto \mathbf{y}' = (y'_0, \ldots, y'_{n+1})$, where

$$y'_0 = 1,$$
$$y'_1 = \sum_{i=0}^{n-1} w_i y_i^2$$
$$y'_2 = y_0,$$
$$\cdots$$
$$y'_{n+1} = y_{n-1}.$$

Then it runs $\mathsf{IPE.Encrypt}(\mathbf{y}', \mathsf{msk}, \mathsf{pp}) \to D_{\mathbf{y}}$ and outputs $D_{\mathbf{y}}$.

$\mathsf{Dist}(Q_{\mathbf{x}}, D_{\mathbf{y}}, \mathsf{pp})$:
The distance measurement algorithm takes $Q_{\mathbf{x}}, D_{\mathbf{y}}$, and pp as input. It runs $\mathsf{IPE.Decrypt}(C_{\mathbf{x}}, K_{\mathbf{y}}, \mathsf{pp}) = Z$ and outputs Z.

Correctness: The correctness of the resulting scheme (i.e., the similarity searchable encryption scheme) follows from the following claim and that of the underlying inner product encryption scheme.

Claim 1. *For any vectors $\mathbf{x}, \mathbf{y} \in \mathbb{Z}_p^n$, the following holds:*

$$\mathsf{dist}(\mathbf{x}, \mathbf{y}) = \langle \mathsf{Encode}_1(\mathbf{x}), \mathsf{Encode}_2(\mathbf{y}) \rangle.$$

Proof.

$$\begin{aligned}\mathsf{dist}(\mathbf{x}, \mathbf{y}) &= \sum_{i=0}^{n-1} w_i (x_i - y_i)^2 \\ &= \sum_{i=0}^{n-1} w_i x_i^2 + \sum_{i=0}^{n-1} w_i y_i^2 - 2 \sum_{i=0}^{n-1} w_i x_i y_i \\ &= \langle \mathsf{Encode}_1(\mathbf{x}), \mathsf{Encode}_2(\mathbf{y}) \rangle.\end{aligned}$$

□

4.2 Security Proof

In this section, we prove the following theorem, which addresses the security of our construction.

Theorem 1. *If the inner product encryption scheme* (IPE.Setup, IPE.KeyGen, IPE.Encrypt, IPE.Decrypt) *satisfies full privacy, so does the private similarity searchable encryption scheme constructed above. Similarly, if the inner product encryption scheme satisfies weak privacy, so does the private similarity searchable encryption scheme constructed above.*

Proof. We prove the former part of the theorem. The latter part can be proven similarly. Toward a contradiction, we assume an adversary $\mathcal{A}$ who breaks the full privacy of the private similarity searchable encryption scheme. From the adversary $\mathcal{A}$, we construct another adversary $\mathcal{B}$ against the underlying inner product encryption scheme.

Adversary $\mathcal{B}$:

1. Given $(1^\lambda, 1^n)$, $\mathcal{A}$ first chooses the weight vector $\mathbf{w} = (w_1, \ldots, w_n)$ and gives it to $\mathcal{B}$. The vector specifies the encoding functions Encode_1 and Encode_2.
2. $\mathcal{B}$ receives pp from its challenger C that has run $\mathsf{IPE.Setup}$. It gives pp to $\mathcal{A}$.
3. During the game, $\mathcal{A}$ chooses a query $(\mathbf{x}_0, \mathbf{x}_1) \in \mathbb{Z}_p^n \times \mathbb{Z}_p^n$. Then, $\mathcal{B}$ runs Encode_1 as follows and gives $(\mathbf{x}'_0, \mathbf{x}'_1)$ to the challenger C.

   $$\mathbf{x}'_0 = \mathsf{Encode}_1(\mathbf{x}_0),\ \mathbf{x}'_1 = \mathsf{Encode}_1(\mathbf{x}_1).$$

   C runs $\mathsf{IPE.Encrypt}$ on input $\mathbf{x}'_b$ to generate $C_{\mathbf{x}'_b}$ and gives it to $\mathcal{B}$. Then $\mathcal{B}$ gives $C_{\mathbf{x}'_b}$ to $\mathcal{A}$ as $Q_{\mathbf{x}_b}$.
4. $\mathcal{A}$ may choose $(\mathbf{y}_0, \mathbf{y}_1) \in \mathbb{Z}_p^n \times \mathbb{Z}_p^n$ as a query to the adversary $\mathcal{B}$. Then $\mathcal{B}$ runs Encode_2 as follows and gives $(\mathbf{y}'_0, \mathbf{y}'_1)$ to C.

   $$\mathbf{y}'_0 = \mathsf{Encode}_2(\mathbf{y}_0),\ \mathbf{y}'_1 = \mathsf{Encode}_2(\mathbf{y}_1).$$

   Then C runs $\mathsf{IPE.KeyGen}$ on input $\mathbf{y}'_b$ to generate $K_{\mathbf{y}'_b}$ and gives it to $\mathcal{B}$. Then $\mathcal{B}$ passes $K_{\mathbf{y}'_b}$ to $\mathcal{A}$ as $D_{\mathbf{y}_b}$. It should be noted that it must be true that $\langle \mathbf{x}'_0, \mathbf{y}'_0 \rangle = \langle \mathbf{x}'_1, \mathbf{y}'_1 \rangle$.
5. At the end of the game, $\mathcal{A}$ outputs b', which is the guess for b. $\mathcal{B}$ outputs the same bit b' as its guess.

Due to the constraints in the query made by $\mathcal{A}$, $\mathsf{dist}(\mathbf{x}_0, \mathbf{y}_0) = \mathsf{dist}(\mathbf{x}_1, \mathbf{y}_1)$ holds for all queried $(\mathbf{x}_0, \mathbf{x}_1)$ and $(\mathbf{y}_0, \mathbf{y}_1)$. Therefore, for all $(\mathbf{x}'_0, \mathbf{x}'_1)$ and $(\mathbf{y}'_0, \mathbf{y}'_1)$ defined above, it holds that

$$\langle \mathbf{x}'_0, \mathbf{y}'_0 \rangle = \mathsf{dist}(\mathbf{x}_0, \mathbf{y}_0) = \mathsf{dist}(\mathbf{x}_1, \mathbf{y}_1) = \langle \mathbf{x}'_1, \mathbf{y}'_1 \rangle.$$

Therefore, $\mathcal{B}$ only makes valid queries in the game. Furthermore, it can be easily seen that the advantage of $\mathcal{B}$ is the same as that of $\mathcal{A}$. By our assumption that the advantage of $\mathcal{A}$ is non-negligible, $\mathcal{B}$'s advantage is non-negligible as well. We conclude the proof of the theorem. □

5. Instantiations

5.1 Instantiation Based on [4]

Bishop et al. [4] constructed a (function private) inner product encryption scheme using Dual Pairing Vector Spaces [16]. The scheme satisfies weak privacy under the SXDH assumption. By applying our conversion in Sect. 4.1 to the scheme, we obtain a private similarity searchable encryption scheme with weak privacy. We write down the resulting scheme in the following. We note that there is a restriction on the scheme that the output of Dist be polynomial size in the security parameter. This restriction is inherited from [4].

In the scheme, we will use asymmetric bilinear groups consisting of G_1, G_2, G_T, all with prime order p. The groups are equipped with an efficiently computable map

$e : G_1 \times G_2 \to G_T$ that satisfies the following two properties: (1) $e(u^a, v^b) = e(u, v)^{ab}$ for all $u \in G_1$, $v \in G_2$, $a, b \in \mathbb{Z}$. (2) $e(u, v) \neq 1$ for all $u, v \neq 1$. In the following, for any vector $\mathbf{w} = (w_1, \ldots, w_n) \in \mathbb{Z}_p^n$ and a group element $g_i \in G_i$, we write $g_i^{\mathbf{w}} \in G_i^n$ to denote $(g_i^{w_1}, \ldots, g_i^{w_n}) \in G_i^n$ where $i \in \{1, 2\}$.

Setup$(1^\lambda, 1^n)$:
The setup algorithm takes the security parameter λ and positive integer n as input. It chooses an asymmetric bilinear groups (G_1, G_2, G_T) with prime order $p > 2^{\Theta(n)}$ equipped with bilinear map $e : G_1 \times G_2 \to G_T$. It fixes generators g_1, g_2 of G_1, G_2 respectively. It generates dual orthonormal bases $\mathbb{B} = \{\mathbf{b}_i\}$, $\mathbb{B}^* = \{\mathbf{b}_i^*\}$ $(i = 0, \ldots, 2n + 7)$ of $\mathbb{Z}_p^{2(n+4)}$ and dual orthonormal bases $\mathbb{D} = \{\mathbf{d}_i\}$, $\mathbb{D}^* = \{\mathbf{d}_i^*\}$ $(i = 0, \ldots, 3)$ of $\mathbb{Z}_p^2$. It defines the master secret key as $\mathsf{msk} = (\mathbb{B}, \mathbb{B}^*, \mathbb{D}, \mathbb{D}^*)$ and the public parameters $\mathsf{pp} = (G_1, G_2, G_T, g_1, g_2, p)$.

Query$(\mathbf{x}, \mathsf{msk}, \mathsf{pp})$:
The query algorithm takes a vector $\mathbf{x} = (x_0, \ldots, x_{n-1}) \in \mathbb{Z}_p^n$, the master secret key msk, and the public parameters pp as input. It first computes $\mathbf{x}' = (x'_0, \ldots, x'_{n+1}) = \mathsf{Encode}_1(\mathbf{x})$ (as in Sect. 4.1). Then, it defines $\mathbf{x}'' = (0, 1, x'_0, \ldots, x'_{n+1}) = (x''_0, \ldots, x''_{n+3})$[†] and computes the encrypted query $Q_{\mathbf{x}} = (Q_{\mathbf{x},1}, Q_{\mathbf{x},2})$ as follows.

$$Q_{\mathbf{x},1} = g_2^{\beta(x''_0\mathbf{b}_0 + \cdots + x''_{n+3}\mathbf{b}_{n+3}) + \beta^*(x''_0\mathbf{b}_{n+4} + \cdots + x''_{n+3}\mathbf{b}_{2n+7})},$$
$$Q_{\mathbf{x},2} = g_2^{\beta(\mathbf{d}_0 + \mathbf{d}_1) + \beta^*(\mathbf{d}_2 + \mathbf{d}_3)}$$

EncDB$(\mathbf{y}, \mathsf{msk}, \mathsf{pp})$:
The database encryption algorithm takes a vector $\mathbf{y} = (y_0, \ldots, y_{n-1}) \in \mathbb{Z}_p^n$, the master secret key msk, and the public parameters pp as input. It first computes $\mathbf{y}' = (y'_0, \ldots, y'_{n+1}) = \mathsf{Encode}_2(\mathbf{y})$ (as in Sect. 4.1). Then, it defines $\mathbf{y}'' = (1, 0, y'_0, \ldots, y'_{n+1}) = (y''_0, \ldots, y''_{n+3})$. It then picks random $\alpha, \alpha^* \in \mathbb{Z}_p$ and outputs the encrypted record $D_{\mathbf{y}} = (D_{\mathbf{y},1}, D_{\mathbf{y},2})$ computed as follows.

$$D_{\mathbf{y},1} = g_1^{\alpha(y''_0\mathbf{b}_0^* + \cdots + y''_{n+3}\mathbf{b}_{n+3}^*) + \alpha^*(y''_0\mathbf{b}_{n+4}^* + \cdots + y''_{n+3}\mathbf{b}_{2n+7}^*)},$$
$$D_{\mathbf{y},2} = g_1^{\alpha(\mathbf{d}_0^* + \mathbf{d}_1^*) + \alpha^*(\mathbf{d}_2^* + \mathbf{d}_3^*)}$$

Dist$(Q_{\mathbf{x}}, D_{\mathbf{y}}, \mathsf{pp})$:
The distance measurement algorithm takes $Q_{\mathbf{x}} = (Q_{\mathbf{x},1}, Q_{\mathbf{x},2})$, $D_{\mathbf{y}} = (D_{\mathbf{y},1}, D_{\mathbf{y},2})$, and the public parameters pp as input. Then it computes $Z = \mathsf{dist}(\mathbf{x}, \mathbf{y})$ as follows. It first computes

$$D_1 = e(Q_{\mathbf{x},1}, D_{\mathbf{y},1}),\ D_2 = e(Q_{\mathbf{x},2}, D_{\mathbf{y},2}).$$

It then computes a $Z \in \mathbb{Z}_p$ such that $D_2^Z = D_1$, and outputs Z.

[†]Although not mentioned explicitly in the paper, the security proof of Bishop et al. [4] implicitly assumes that an adversary is not allowed to query the zero vector in the security game. (In fact, if an adversary can query the zero vectors, there exists a simple attack to the scheme.) We can use simple padding technique to remove this restriction. Namely, we encode $\mathbf{x}$ and $\mathbf{y}$ as $\mathbf{x} \to (1, 0, \mathbf{x})$, and $\mathbf{y} \to (0, 1, \mathbf{y})$. Our scheme presented here is based on this modified version of the Bishop et al. scheme.

We note that we can guarantee that the Dist algorithm will run in polynomial time when the value of $\langle \mathbf{x}, \mathbf{y} \rangle$ is bounded by some fixed polynomial. It can be easily seen that Z is the weighted Euclidean distance between $\mathbf{x}$ and $\mathbf{y}$.

5.2 Instantiation Based on [9]

As we have seen, since the inner product encryption by [4] only achieves weak privacy, so does the resulting private similarity searchable encryption scheme obtained by the conversion in Sect. 4.1. Very recently, Datta et al. [9] proposed an inner product encryption scheme with full privacy (rather than weak privacy). Their scheme is similar to that of [4], but slightly more inefficient. By starting from their scheme, we obtain a private similarity searchable encryption scheme with full privacy.

Here, we estimate the efficiency of the resulting scheme. To estimate the efficiency, we count the number of scalar multiplications and the paring operations of our proposal based and use the implementation result of [20] to calculate the computational cost. In this evaluation, the number of dimensions of $\mathbf{x}$ and $\mathbf{y}$ were both 10. We calculated the 32 scalar multiplications of the query algorithm and the database encryption algorithm, and the execution times of these calculation were 1.84 seconds and 3.33 seconds respectively. And, we calculated the 32 paring operations of the distance measurement algorithm, and the execution time of this calculation was 10.94 seconds. From these results, it was shown that this instantiation is feasible by this evaluation.

Acknowledgement

A part of this work is supported by JST CREST grant number JPMJCR1688.

References

[1] Y. Unagami, N. Matsuzaki, S. Yamada, N. Attrapadung, T. Matsuda, and G. Hanaoka, "Private similarity searchable encryption for Euclidean distance," Proc. International Symposium on Information Theory and Its Applications, pp.754–758, Oct. 2016.
[2] M. Abdalla, F. Bourse, A.D. Caro, and D. Pointcheval, "Simple functional encryption schemes for inner products," Public-Key Cryptography, PKC 2015, Lecture Notes in Computer Science, vol.9020, pp.733–751, 2015.
[3] J. Baek, R. Safavi-Naini, and W. Susilo, "Public key encryption with keyword search revisited," Computational Science and Its Applications, ICCSA 2008, Lecture Notes in Computer Science, vol.5072, pp.1249–1259, 2008.
[4] A. Bishop, A. Jain, and L. Kowalczyk, "Function-hiding inner product encryption," Advances in Cryptology, ASIACRYPT 2015, Lecture Notes in Computer Science, vol.9452, pp.470–491, 2015.
[5] D. Boneh, A. Sahai, and B. Waters, "Functional encryption: Definitions and challenges," Theory of Cryptography, TCC 2011, Lecture Notes in Computer Science, vol.6597, pp.253–273, 2011.
[6] D. Boneh, G.D. Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," Advances in Cryptology, EUROCRYPT 2004, Lecture Notes in Computer Science, vol.3027, pp.506–522, 2004.
[7] D. Boneh and B. Waters, "Conjunctive, subset and range queries on

encrypted data," Theory of Cryptography, TCC 2007, Lecture Notes in Computer Science, vol.4392, pp.535–554, 2007.
[8] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky, "Searchable symmetric encryption: Improved definitions and efficient constructions," Proc. 13th ACM Conference on Computer and Communications Security, CCS 2006, pp.79–88, 2006.
[9] P. Datta, R. Dutta, and S. Mukhopadhyay, "Functional encryption for inner product with full function privacy," Public-Key Cryptography, PKC 2016, Lecture Notes in Computer Science, vol.9614, pp.164–195, 2016.
[10] W. Du and M.J. Atallah, "Protocols for secure remote database access with approximate matching," E-Commerce Security and Privacy, Advances in Information Security, vol.2, pp.87–111, 2001.
[11] S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters, "Candidate indistinguishability obfuscation and functional encryption for all circuits," Proc. 2013 IEEE 54th Annual Symposium on Foundations of Computer Science, FOCS 2013, pp.40–49, 2013.
[12] C. Gentry, "Fully homomorphic encryption using ideal lattices," Proc. Forty-First Annual ACM Symposium on Theory of Computing, STOC 2009, pp.169–178, 2009.
[13] F. Guo, W. Susilo, and Y. Mu, "POSTER: Euclidean distance based encryption: How to embed fuzziness in biometric based encryption," Proc. ACM Conference on Computer and Communications Security, CCS 2014, pp.1430–1432, 2014.
[14] 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.
[15] S. Laur and H. Lipmaa, "On private similarity search protocols," Proc. 9th Nordic Workshop on Secure IT Systems, pp.73–77, 2004.
[16] T. Okamoto and K. Takashima, "Fully secure functional encryption with general relations from the decisional linear assumption," Advances in Cryptology, CRYPTO 2010, Lecture Notes in Computer Science, vol.6223, pp.191–208, 2010.
[17] A. Oosawa, R. Hisanaga, T. Inoue, T. Hoshino, and K. Shimura, "Development of "SYNAPSE Case Match", content-based image retrieval system for supporting image diagnosis," Fuji Film research & development, no.58-2013.
[18] E. Shen, E. Shi, and B. Waters, "Predicate privacy in encryption systems," Theory of Cryptography, TCC 2009, Lecture Notes in Computer Science, vol.5444, pp.457–473, 2009.
[19] A.C.-C. Yao, "How to generate and exchange secrets," Proc. 27th FOCS, pp.162–167, 1986.
[20] E. Zavattoni, L.J.D. Perez, S. Mitsunari, A. Sánchez-Ramírez, T. Teruya, and F. Rodríguez-Henríquez, "Software implementation of an attribute-based encryption scheme," IEEE Trans. Comput., vol.64, no.5, pp.1429–1441, 2015.

Yuji Unagami received the B.E. and M.E. degrees in Industrial and Management Systems from Waseda University in 2004 and 2006, respectively. In 2006, he joined the Corporate R&D Division, Matsushita Electric Industrial (now Panasonic) Co., Ltd., Osaka Japan. His research interests are cyptography and information security.

Natsume Matsuzaki graduated from the Department of Mathematics, Nara Women's University in 1982 and received Ph.D. degree in Engineering from Yokohama National University in 2003. She worked for Panasonic corporation from 1982 to 2016. She joined University of Nagasaki in April in 2016. She is currently, a professor of department of information security, University of Nagasaki. She is interested in encryption and information security technologies.

Shota Yamada received his Ph.D. degree in Science from the University of Tokyo in 2014. From 2011 to 2014 and from 2014 to 2015, he had been a Research Fellow of Japan Society for the Promotion of Science (JSPS). From 2014, he has been with the National Institute of Advanced Industrial Science and Technology (AIST), Japan. He received Innovation Paper Award in 2012 and SCIS Paper Award in 2010, at Symposium on Cryptography and Information Security (SCIS).

Nuttapong Attrapadung received his bachelors degree (first-class honors) in Electrical engineering from Chulalongkorn university in Thailand in 2001, and received his masters degree and Ph.D. in Information and communication engineering from the University of Tokyo in 2004 and 2007, respectively. During 2007–2008, he is granted a JSPS post-doctoral fellowship. He has been with the National Institute of Advanced Industrial Science and Technology (AIST) since 2008, and is currently a senior researcher in the Advanced Cryptography Research Group. He received Best Paper Awards at SITA 2005 and SCIS 2006, and the Ericsson Young Scientist Award in 2010.

Takahiro Matsuda received his bachelors, masters, and Ph.D. degrees in Information and Communication Engineering from the University of Tokyo in 2006, 2008, and 2011, respectively. From 2009 to 2011 and from 2011 to 2013, he had been a Research Fellow of Japan Society for the Promotion of Science (JSPS). From 2011, he has been with the National Institute of Advanced Industrial Science and Technology (AIST), Japan. He received Innovation Paper Award in 2012 and 2014 and SCIS Paper Award in 2009, at Symposium on Cryptography and Information Security (SCIS), and Best Paper Award at Computer Security Symposium (CSS) 2009 and 2010. He also received IPSJ Yamashita SIG Research Award in 2012. He is a member of IPSJ and IACR.

Goichiro Hanaoka graduated from the Department of Engineering, The University of Tokyo in 1997 and received Ph.D. degree from the University of Tokyo in 2002. He joined AIST in 2005. He is currently, a leader, Advanced Cryptosystems Research Group, Information Technology Research Institute, AIST. He engages in the R&Ds for encryption and information security technologies including the efficient design and security evaluation of public key cryptosystem. He received the Wilkes Award from British Computer Society in 2007, Best Paper Award from from The Institute of Electronics, Information and Communication Engineers in 2008, Innovative Paper Award from Symposium on Cryptography and Information Security (SCIS) in 2012 and 2014, Award of Telecommunication Advancement Foundation in 2005, 20th Anniversary Award from SCIS in 2005, Best Paper Award from SCIS in 2006, Encouragement Award from Symposium on Information Theory and its Applications (SITA) in 2000, and others.