A New Approach for Practical Function-Private Inner Product Encryption

Sungwook Kim[a], Jinsu Kim[a], Jae Hong Seo[b,*]

[a] *Security Lab., Software R&D Center, Samsung Electronics, Republic of Korea*
[b] *Department of Mathematics, Myongji University, Republic of Korea*

Abstract

Functional Encryption (FE) is a new paradigm supporting restricted decryption keys of function f that allows one to learn $f(x_j)$ from encryptions of messages x_j. A natural and practical security requirements for FE is to keep not only messages $x_1, \ldots, x_q$ but also functions $f_1, \ldots f_q$ confidential from encryptions and decryptions keys, except inevitable information $\{f_i(x_j)\}_{i,j\in[q]}$, for any polynomial a-priori unknown number q, where f_i's and x_j's are adaptively chosen by adversaries. Such the security requirement is called *full function privacy*. In this paper, we particularly focus on function-private FE for inner product functionality in the *private key setting* (simply called Inner Product Encryption (IPE)). To the best of our knowledge, there are two approaches for fully function-private IPE schemes in the private key setting. One of which is to employ a general transformation from (non-function-private) FE for general circuits (Brakerski and Segev, TCC 2015). This approach requires heavy crypto tools such as indistinguishability obfuscation (for non-function-private FE for general circuits) and therefore inefficient. The other approach is relatively practical; it directly constructs IPE scheme by using *dual pairing vector spaces (DPVS)* (Bishop et al. ASIACRYPT 2015, Datta et al. PKC 2016, and Tomida et al. ISC 2016).

We present a new approach for practical function-private IPE schemes that does not employ DPVS but generalizations of Brakerski-Segev transformation. Our generalizations of Brakerski-Segev transformation are easily combinable with existing (non-function-private) IPE schemes as well as (non-function-private) FE schemes for general circuits in several levels of security. Our resulting IPE schemes achieve better performance in comparison with Bishop et al. IPE scheme as well as Datta et al. IPE scheme while preserving the same security notion under the same complexity assumption. In comparison with Tomida et al. IPE scheme, ours have comparable performance in the size of both ciphertext and decryption key, but better performance in the size of master key.

Keywords: Functional Encryption, Function Privacy, Inner Product

*Corresponding author
Email addresses: sw14.kim@samsung.com (Sungwook Kim), jinsu86.kim@samsung.com (Jinsu Kim), jaehongseo@mju.ac.kr (Jae Hong Seo)

1. Introduction

The usage of classical encryption schemes and its security requirement are simple; one who has the decryption key can recover all encrypted messages and all the other users including adversaries cannot obtain any information from the encrypted messages. That is, the decryption policy in classical encryption schemes is *all-or-nothing*. In Functional Encryption (FE), it provides more fine-grained handling of decryption policy by supporting restricted decryption keys associated with specific function f that allow one to learn $f(x_j)$ from encryption of message x_j.

There are numerous realizations of FE schemes for either general or specific functionality. The first FE scheme for simple functionality (beyond classical encryption scheme) is identity-based encryption scheme [1, 2, 3]. Subsequently, several attribute-based encryption schemes [4, 5, 6] and predicate encryption schemes are proposed [7, 6, 8]. The general concept and the security notion (in terms of message privacy) of FE for general functionality is later formalized by Boneh, Sahai, Waters and O'Neill [9, 10]. Until recently, there has been extensively researched on the security of FE schemes and the methodology to construct FE schemes from a variety of assumptions [9, 10, 11, 12, 13, 14, 15, 16, 17, 18].

Function Privacy in Private Key Setting. Contrary to classical encryption scheme, there are several decryption keys in FE scheme and each decryption key is empowered to obtain restricted information from encrypted messages. Due to this difference, the decryption keys as well as the encrypted messages may be subject to keep confidential and we call such a security notion *function privacy* [19].

In this paper, we focus on the function privacy in the private key setting like many previous works [19, 20, 21, 22, 23] since the function privacy has much advantage in the private key setting. In the public key setting, given a decryption key associated with function f, the adversary can always learn $f(x_j)$ for any chosen x_j by herself.[1] Therefore, to define meaningful function privacy in the public key setting, it seems necessary to assume the functions are sampled from somewhat unpredictable distributions [24, 25]. In the private key setting, we do not need such the restriction and we can consider the privacy of encrypted messages and the privacy of functional keys in a completely symmetric manner.

Full Function Privacy. There are several levels of function privacy in private key functional encryption. The strongest function privacy notion (and the message privacy as well) can be formalized in a simulation model. There are impossi-

[1] The adversary first encrypts x_j, decrypts it by using the decryption key associated with f, and then finally obtains $f(x_j)$.

bility results for such the simulation-based security (against adaptive unbounded queries attack) notion in the standard model, though some functionality can be achieved in ideal models such as random oracles [9] and generic group models [26, 23]. Unlike to simulation-based security, the indistinguishability-based strongest notion for function privacy, called *full function privacy*, is achievable in the standard model. We consider Agrawal et al.'s formalization for full function privacy [26], which is a generalization of predicate-privacy in attribute-based encryption [19]. Let us briefly explain the model of full function privacy. Adversaries are allowed to interact with two *left-or-right* oracles $\mathsf{KG}_b(\mathsf{mk}, \cdot, \cdot)$ and $\mathsf{Enc}_b(\mathsf{mk}, \cdot, \cdot)$ for randomly chosen coin $b \in \{0, 1\}$, where KG_b takes two functions f_0 and f_1 as input and then returns a decryption key $\mathsf{sk}_{f_b} \leftarrow \mathsf{KG}(\mathsf{mk}, f_b)$ and Enc_b takes two messages x_0 and x_1 as input and then outputs an encryption $\mathsf{ct}_{x_b} \leftarrow \mathsf{Enc}(\mathsf{mk}, x_b)$. In particular, adversaries can adaptively interact with oracles for any polynomial a-priori unbounded number of queries. To exclude inherently inevitable attacks, there is a condition in adversarial queries that for all queries (x_0, x_1) and (f_0, f_1), it should satisfy that

$$f_0(x_0) = f_1(x_1). \tag{1}$$

We also consider a slightly weaker notion for function privacy, called *weak function privacy*, which was introduced by Bishop, Jain, and Kowalczyk [21]. Basically, the weak function privacy is equivalent to the full function privacy, except the condition for adversarial queries. The weak function privacy relaxes the condition in Equation (1) to the following condition.

$$f_0(x_0) = f_0(x_1) = f_1(x_1) = f_1(x_0). \tag{2}$$

Inner Product Encryption. In this paper, we mainly focus on FE for inner-product functionality, which simply called Inner Product Encryption (IPE) [27]. In IPE scheme, the domain of plaintexts is an n-dimensional inner product space $\mathcal{M}$ and each decryption key is associated with a function $f(\cdot) := \langle \cdot, \boldsymbol{y} \rangle$ for some vector $\boldsymbol{y} \in \mathcal{M}$. The inner product functionality has numerous applications (See well-written previous works [27, 21, 28, 29, 22, 23].) In particular, function-private IPE scheme could be useful for statistical analysis on encrypted data, where the statistical analysis itself has sensitive information.

To the best of our knowledge, there are two approaches for fully function-private IPE schemes in the private key setting. One of which is to employ Brakerski and Segev's general transformation from (non-function-private) FE schemes for general circuits [20]. Although the transformation itself is quite efficient in the sense that it just combines symmetric key encryption and functional encryption in a natural way, this approach requires heavy crypto tools such as indistinguishability obfuscation for realizing non-function-private FE for general circuits and therefore quite inefficient. The other approach is relatively practical; it directly constructs IPE scheme by using *dual pairing vector spaces (DPVS)* introduced by Okamoto and Takashima [30, 31]; Bishop, Jain, and Kowalczyk [21] proposed the first function-private IPE scheme and they proved the weak privacy

Scheme	Ciphertext	Dec. Key	Master Key	Privacy	Assum.
BJK [21]	$(2n+2)\ell_{\mathbb{G}_1}$	$(2n+2)\ell_{\mathbb{G}_2}$	$(8n^2+8)\ell_{\mathbb{Z}_q}$	Weak	SXDH
DDM [22]	$(4n+8)\ell_{\mathbb{G}_1}$	$(4n+8)\ell_{\mathbb{G}_2}$	$(8n^2+12n+28)\ell_{\mathbb{Z}_q}$	Full	SXDH
TAO [34]	$(2n+5)\ell_{\mathbb{G}_1}$	$(2n+5)\ell_{\mathbb{G}_2}$	$(4n^2+18n+20)\ell_{\mathbb{Z}_q}$	Full	XDLIN
BJK [21]+ LV Trans [35]	$(4n+2)\ell_{\mathbb{G}_1}$	$(4n+2)\ell_{\mathbb{G}_2}$	$(32n^2+8)\ell_{\mathbb{Z}_q}$	Full	SXDH
Ours 1	$(n+4)\ell_{\mathbb{G}_1}$	$(n+4)\ell_{\mathbb{G}_2}$	$(4n+4)\ell_{\mathbb{Z}_q}$	Weak	SXDH
Ours 2	$(2n+8)\ell_{\mathbb{G}_1}$	$(2n+8)\ell_{\mathbb{G}_2}$	$(6n+4)\ell_{\mathbb{Z}_q}$	Full	SXDH

All schemes employ asymmetric bilinear maps over two groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of order q.
n: the dimension of vector, ℓ_G: the bit length to represent an element in group G

Table 1: Performance Comparison of IPE Schemes in the Standard Model

of their scheme under the *symmetric external Diffie-Hellman (SXDH)* assumption. Subsequently, Datta, Dutta, and Mukhopadhyay [22] proposed the first full function-private IPE scheme under the same SXDH assumption. Recently, Tomida, Abe, and Okamoto improved the efficiency of Datta et al. IPE scheme. In contrast with the previous pairing-based function-private IPE schemes, their scheme can be applied to any type of bilinear pairing groups and they also proved the security under a relatively weaker (External) Decisional Linear Assumption. In this paper, we simply call these three IPE schemes in [21], [22], and [34] by BJK-IPE, DDM-IPE, TAO-IPE respectively. Lin and Vaikuntanathan [35] presented a generic way to bootstrap from a weakly function-private IPE scheme to a fully function-private IPE scheme if the input scheme satisfies a mild condition called multi-instance function hiding. They used BJK-IPE scheme with $2n$-length vectors to obtain fully function-private IPE scheme with n-length vectors.

1.1. Our Result

We propose two IPE schemes. The first scheme satisfies the weak function privacy and the second scheme satisfies the full function privacy, both under the SXDH assumption. Each scheme has better performance than prior works, BJK-IPE and DDM-IPE with the corresponding security notion in the standard model. In comparison with TAO-IPE scheme, ours have comparable performance in the size of both ciphertext and decryption key, but better performance in the size of master key. More precisely, we provide a comparison Table 1. This improvement seems more impressive when comparing with very recent IPE of Kim, Lewi, Mandal, Montgomery, Roy and Wu [23] in generic group model [32, 33]. We achieve comparable results with them with respect to the size of ciphertext and decryption key even in the standard model; Kim et al.'s IPE scheme achieves $(n+1)$ group elements in decryption key and ciphertext. Furthermore, our constructions have square-root smaller master key size than those of all previous function-private IPE schemes [21, 22, 23].

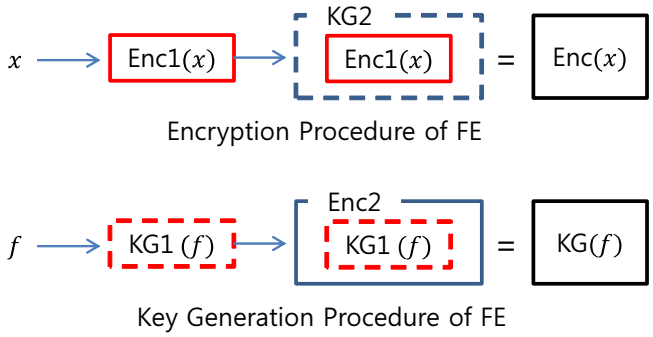

Figure 1: Our first transformation Trans1. The algorithm $(\mathsf{Enc}, \mathsf{KG})$, $(\mathsf{Enc1}, \mathsf{KG1})$ and $(\mathsf{Enc2}, \mathsf{KG2})$ are encryption and key generation of FE, FE1 and FE2, respectively.

As aforementioned, there are only four realized function-private IPE schemes BJK-IPE and DDM-IPE prior to our IPE scheme in the standard model. All these schemes essentially employs the information-theoretic property of the DPVS which is implemented with asymmetric bilinear maps. Contrary to previous schemes, we achieve function-private IPE schemes via a different technique; that is, we do not use DPVS but a generalization of Brakerski and Segev transformation from (non-function-private) FE for general functionality [20]. Our approach consists of two steps. First, we generalize the Brakerski-Segev transformation and then analyze the security of our generalizations in several levels of function privacy. We believe that our transformations are of independent interest since our approach may be applicable to other functionalities. We will consider the detailed advantages of our transformations later. In the second step, we modify an existing non-function-private IPE scheme and also propose a new FE scheme for multi-exponentiation, to be well harmonized with our transformations. Then, by applying our transformation to these FE schemes, we directly obtain new practical function-private IPE schemes. Let us explain overviews of two our steps below.

Step 1 - General Transformation. We first propose two transformations Trans1 and Trans2 from a message-private FE scheme FE1 for functionality $\mathcal{F}$ to a function-private FE for the same functionality. In our transformations, we use another message-private functional encryption FE2 for specific functionality, which is related to decryption circuit of FE1. We give a pictorial description of our first transformation Trans1 in Figure 1. Our second transformation Trans2 is obtained by applying the Naor-Yung double encryption methodology [36] to Trans1.

The proposed transformations Trans1 and Trans2 are generalizations of Brak-

erski and Segev's basic[2] and final transformations, respectively [20].[3] Let us briefly explain how to consider the Brakerski-Segev transformation as an instance of ours. We focus on their basic scheme since both their final scheme and our Trans2 are obtained by applying the Naor-Yung double encryption methodology to their basic scheme and our Trans1, respectively. One difference between ours and their basic scheme is that Brakerski and Segev use a semantically secure symmetric encryption scheme instead of FE1 in Trans1. In fact, a semantically secure symmetric encryption scheme can be utilized as a functional encryption scheme for arbitrary functionality, in particular, secure against *non-key query attack*.[4] More precisely, given a symmetric encryption scheme $(\mathsf{S.Setup}, \mathsf{S.Enc}, \mathsf{S.Dec})$, an FE scheme $(\mathsf{FE.Setup}, \mathsf{FE.Enc}, \mathsf{FE.KG}, \mathsf{FE.Dec})$ for arbitrary functionality $\mathcal{F}$ can be constructed as follows.

$$\begin{array}{ll} \mathsf{FE.Setup} = \mathsf{S.Setup}(1^\lambda) \to \mathsf{mk}, & \mathsf{FE.Enc} = \mathsf{S.Enc}(\mathsf{mk}, \boldsymbol{x}) \to \mathsf{ct}_{\boldsymbol{x}}, \\ \mathsf{FE.KG}(\mathsf{mk}, f \in \mathcal{F}) \to \mathsf{sk}_f = \{\mathsf{mk}, f\}, & \mathsf{FE.Dec}(\mathsf{sk}_f, \mathsf{ct}_{\boldsymbol{x}}) \to f\big(\mathsf{S.Dec}(\mathsf{mk}, \mathsf{ct}_{\boldsymbol{x}})\big). \end{array}$$

The semantic security of the symmetric encryption scheme directly implies the message privacy of the FE scheme against non-key query attack. Therefore, the Brakerski-Segev scheme can be considered as an instantiation of our transformation (of course, only if the security against non-key query attack is enough for FE1).

Our analysis shows that several levels of function privacy can be achieved via our transformations. Informally, we achieve that

1. if FE1 and FE2 satisfy IND-based message privacy (against adaptive query attack), then the result of Trans1 achieves IND-based weak function privacy.[5]

2. if FE1 is SIM-based message-private against non-key query attack and FE2 satisfy SIM-based message privacy, then the result of Trans1 achieves SIM-based full function privacy.[6]

3. if FE1 is IND-based message-private against non-key query attack and FE2

[2] In fact, Brakerski and Segev call this basic scheme a failed attempt. Furthermore, prior to Brakerski and Segev, Goldwasser et al. [37] used this scheme to construct reusable garbled circuits and considered only a relatively weaker security notion; the adversary has to specify a challenge function prior to receiving any encryptions.

[3] In fact, the roles of functions and messages in our transformations are exactly opposite to those in the Brakerski-Segev transformation. We think that this format is more natural in terms of transformation from FE1.

[4] Informally, if the adversary does not query any decryption keys, then we say such the attack non-key query attack. The formal definition is given in Section 2.

[5] If we do not clearly mention the adversarial types such as non-key query attack, then we assume that adversary can adaptively access to both encryption and key generation oracles.

[6] There are several impossibility results [9, 10] on SIM-based FE scheme in the standard model. Note that, however, there are also several evidences on possibility of SIM-based FE scheme in the ideal models such as random oracles and generic groups [9, 23]. One may combine our first transformation Trans1 with SIM-secure FE schemes in the ideal models.

satisfies IND-based message privacy, then the result of Trans2 achieves IND-based full function privacy.

Here, for the second and third results, message-private FE1 against non-key query attack is enough. As for the first result, we require FE1 to satisfy the message privacy against adaptive (key and encryption) query attack.

Our generalizations have several advantages in terms of both security and applicability. Brakerski and Segev showed that their basic scheme is not function-private, and then they applied the Naor-Yung double encryption methodology [36] to the basic scheme for achieving IND-based function privacy. By extending Brakerski and Segev's argument about their basic transformation, we can show that their basic scheme does not achieve even (IND-based) weak privacy. Contrary to their basic transformation, we show that our first transformation Trans1 achieves (IND-based) weak function privacy if FE1 satisfies the message-privacy against adaptive query attack. Furthermore, we prove that Trans1 satisfies SIM-based full function privacy if the underlying FE scheme satisfies SIM-based message privacy. At first glance this result looks counter-intuitive since Trans1 (and Brakerski and Segev's basic transformation) can achieve a stronger security notion (that is, SIM-based full function privacy), but not achieve a weaker security notion (that is, IND-based message privacy). However, this is due to the underlying functional encryptions; the SIM-based security notion implies the IND-based security notion and there exists a class of schemes that satisfy IND-based security but not SIM-based security. Although the transformation works well for SIM-based secure schemes, but it does not work for such the class of (IND-secure but not SIM-secure) schemes. Therefore, there is no contradiction in the above results.

To obtain a function-private FE for functionality $\mathcal{F}$, the Brakerski-Segev transformation requires a message-private FE for more expressive functionality than $\mathcal{F}$. (Roughly speaking, the underlying FE scheme has to support a composite function of $\mathcal{F}$ and decryption algorithm of the underlying symmetric encryption scheme.) In our transformation, we requires a message-private FE scheme FE1 for the same functionality $\mathcal{F}$ and another message-private FE for decryption circuit of the first FE scheme FE2. Therefore, if there exists an efficient realization of FE2 for the decryption circuit of FE1, then we can realize an efficient function-private FE scheme for $\mathcal{F}$. In the next paragraph, we present our idea for a pair of FE schemes satisfying the above requirements, in particular, for inner product functionality.

Step 2 - FE1 and FE2 Constructions for Inner Product. As we mentioned above, in order to apply our first transformation Trans1 for inner product functionality, we need two message-private FE schemes as building blocks; the first scheme FE1 is an IPE whose decryption is relatively simple and the second scheme FE2 is an FE scheme whose functionality supports the decryption of FE1.

To this end, we begin with Agrawal et al.'s group-based IPE (ALS-IPE) scheme [28]. The decryption process of ALS-IPE consists of two steps; (1) a

multi-exponentiation and then (2) a solving a discrete log problem (DLP).[7] To the best of our knowledge, unfortunately, there is no practical FE construction for such operations, except FE schemes for general circuits, which is inefficient as we mentioned before. To resolve this, we make several tricks, essentially for modifying the decryption algorithm so that we can construct another efficient FE scheme for the decryption circuit of the first FE scheme. First, we remove the process of solving a DLP procedure at the moment. (We will recover it after applying transformation.) Second, we add an asymmetric bilinear map operation at the end of decryption process.[8] This process seemingly redundant, but necessary to combine with our second FE scheme.[9] Next, we propose another FE scheme for multi-exponentiation with bilinear map, by again modifying the ALS-IPE scheme.

In order to apply our second transformation Trans2, we need another idea to consider the decryption process with "double encrypted" ciphertexts of the modified ALS-IPE scheme. (Recall that Trans2 applies the Naor-Yung double encryption methodology to Trans1.) A problem of the decryption process with double encrypted ciphertexts is that this process contains a branching statement. (For the detailed functionality, see Figure 5.) In general, we don't know how to represent this process by a simple circuit, as much as we can realize an FE scheme for it. Instead, we show that if the decryption algorithm of FE1 satisfies some mild conditions, which are also satisfied by our modification of the ALS-IPE scheme, then there exists an equivalent decryption process not containing branching statements. The resulting decryption process becomes a multi-exponentiation with bilinear. Therefore, similar to Trans1, we can utilize the proposed FE for multi-exponentiation with bilinear.

1.2. Road Map

In the next section, we provide short preliminaries including definitions of FE scheme, complexity assumptions, weak/full function privacy. In Section 3 and 4, we present two our transformations Trans1 and Trans2, respectively, where Trans2 is obtained by applying Naor-Yung double encryption methodology to Trans1. Furthermore, we also provide analyses for two our transformations in the corresponding sections, respectively. In Section 5, we propose several modifications of an existing IPE scheme to construct non-function-private FE schemes

[7] The final process of solving DLP imposes a restriction on the distribution of message space. However, we note that all previous function-private IPE schemes [21, 22, 23] has the same process of solving DLP in decryption algorithms. Nevertheless, there are still numerous applications under this restriction. See [21, 22, 23].

[8] Indeed, the modified scheme is no longer IPE scheme. It outputs $e(g, \bar{g})^{\langle x,y \rangle}$ instead of $\langle x, y \rangle$, where x and y are vectors in input ciphertext and decryption key and g is a generator of $\mathbb{G}_1$. However, it does not matter to our purpose, function-private IPE. The resulting scheme transformed by Trans1 will also output $e(g, \bar{g})^{\langle x,y \rangle}$. Then, we can add a post process of solving a DLP based on $e(g, \bar{g})$ to the decryption algorithm of the transformed scheme.

[9] Indeed, we could not construct an FE scheme for multi-exponentiation *without* bilinear map. If one can construct an efficient FE scheme for such functionality, then we can remove a bilinear map from the first FE scheme.

that could be appropriately combinable with two our transformations. Finally, the resulting schemes are two practical function-private IPE scheme, which have shortest master secret key in comparison with previous works.

2. Preliminaries

We first define notations used in the paper. For a message $\boldsymbol{x}$ and a function f, $|\boldsymbol{x}|$ and $|f|$ denote the bit-length to represent $\boldsymbol{x}$ and f, respectively, where such the representation is uniquely fixed (by the cryptosystem). For example, let $\boldsymbol{x}$ be a vector in $\mathbb{Z}_q^n$ and f be an inner-product function with a length-n vector $\boldsymbol{y} \in \mathbb{Z}_q^n$, that is, $f(\cdot) := \langle \cdot, \boldsymbol{y} \rangle$. Then, we can specify $\boldsymbol{x}$ by $n\lceil \log p \rceil$ bits and f by $\boldsymbol{y}$, so that $|\boldsymbol{x}| = |f| = |\boldsymbol{y}| = n\lceil \log p \rceil$.

2.1. Private-Key Functional Encryption

We provide the syntax and the security notion for functional encryption schemes, in particular, in the private-key setting. We basically follows definitions of Brakerski and Segev [20] and Bishop, Jain, and Kowalczyk [21].

Definition 1 (Private-Key Functional Encryption)**.** *A private-key functional encryption* FE *over a message space* $\mathcal{M}$ *and a function space* $\mathcal{F}$ *consists of the following four algorithms* $\{\mathsf{Setup}, \mathsf{Enc}, \mathsf{KG}, \mathsf{Dec}\}$*.*

- $\mathsf{Setup}(1^\lambda, (\mathcal{F}, \mathcal{M}))$ *takes as input* $\mathcal{F}$*,* $\mathcal{M}$*, and the unary representation* 1^λ *of the security parameter* $\lambda \in \mathbb{N}$ *and outputs a master secret key* mk *and a system parameter* pp*. (If* $(\mathcal{M}, \mathcal{F})$ *is clear from the context, we omit it from the input of* Setup*.)*
- $\mathsf{Enc}(\mathsf{mk}, \boldsymbol{x})$ *takes as input* mk *and a message* $\boldsymbol{x} \in \mathcal{M}$ *and output a ciphertext* ct*.*
- $\mathsf{KG}(\mathsf{mk}, f)$ *takes as input* mk *and a function* $f \in \mathcal{F}$ *and output a decryption key* sk_f*.*
- $\mathsf{Dec}(\mathsf{pp}, \mathsf{ct}, \mathsf{sk})$ *takes as input* pp*,* ct *and* sk *output a result satisfying the following correctness condition.*

We usually omit pp *from the output of* Setup *algorithm and the input of* Dec *algorithm if it is clear from the context. For* $\mathsf{Setup}(1^\lambda) \to \mathsf{mk}, \forall \boldsymbol{x} \in \mathcal{M}, \forall f \in \mathcal{F}$*, there exists a negligible function* ε *in* λ *such that*

$$\Pr\left[\mathsf{Dec}(\mathsf{Enc}(\mathsf{mk}, \boldsymbol{x}), \mathsf{KG}(\mathsf{mk}, f)) \to f(\boldsymbol{x})\right] \geq 1 - \varepsilon(\lambda),$$

where the probability goes over the all internal randomness used in Setup*,* Enc*,* KG*, and* Dec*.*

Remark 1. *In Definition 1, we define private-key functional encryption over a message space* $\mathcal{M}$ *and a function space* $\mathcal{F}$*. In some applications, we may need to define* $\mathcal{M}$ *and* $\mathcal{F}$ *simultaneously with the other parameters. For example, an*

inner-product between two vectors in $\mathbb{Z}_q^n$ for some n, where q is a prime larger than 2^λ. To cover these applications, we may relax the input $(\mathcal{M}, \mathcal{F})$ of Setup *algorithm as additional information to define $(\mathcal{M}, \mathcal{F})$, instead of the exact form of $\mathcal{M}$ and $\mathcal{F}$. As for the inner-product functionality, the dimension of vectors n can be taken as input of* Setup *and $\mathbb{Z}_q$ is defined as output of* Setup*. Then, $\mathcal{M}$ and $\mathcal{F}$ are defined as $\mathbb{Z}_q^n$ and inner-product over $\mathbb{Z}_q^n$, respectively.*

We define message privacy of FE schemes in the private-key setting, which is the strongest indistinguishability-based definition given in [20].

Definition 2 (Message Privacy)**.** *A private-key functional encryption* FE *over a message space $\mathcal{M}$ and a function space $\mathcal{F}$ is* message private *(denoted by $\mathsf{IND}_{\mathsf{ad}}^{\mathsf{MP}}$) if for any probabilistic polynomial-time algorithm $\mathcal{A}$, which has two oracle accesses $\mathsf{KG}(\mathsf{mk}, f)$ and $\mathsf{Enc}_b(\mathsf{mk}, \boldsymbol{x}_0, \boldsymbol{x}_1) := \mathsf{Enc}(\mathsf{mk}, \boldsymbol{x}_b)$ for her choices $f \in \mathcal{F}$ and $\boldsymbol{x}_0, \boldsymbol{x}_1 \in \mathcal{M}$ with the restriction $f(\boldsymbol{x}_0) = f(\boldsymbol{x}_1)$ and $|\boldsymbol{x}_0| = |\boldsymbol{x}_1|$, there exists a negligible function ε in λ such that*

$$\mathsf{Setup}(1^\lambda) \to \mathsf{mk}$$

$$\textit{and } \left| \Pr\left[\mathcal{A}^{\mathsf{KG}(\mathsf{mk},\cdot),\mathsf{Enc}_0(\mathsf{mk},\cdot,\cdot)}(\lambda) \to 1\right] - \Pr\left[\mathcal{A}^{\mathsf{KG}(\mathsf{mk},\cdot),\mathsf{Enc}_1(\mathsf{mk},\cdot,\cdot)}(\lambda) \to 1\right] \right| \leq \varepsilon,$$

where the probability is taken over all the randomness used in algorithms/oracles of FE *and $\mathcal{A}$.*

We also define a weaker notion of message privacy against *non-key query attack* that is equivalent to the message privacy in Definition 2, except that the adversary is given encryption oracle only. That is, the adversary is not allowed to issue decryption key queries for any functions in the non-key query attack model. For the sake of the simplicity, we use a notation $\mathsf{IND}_{\mathsf{non}}^{\mathsf{MP}}$ to denote the IND-based message privacy against non-key query attack.

Next, we give two definitions for function privacy used for our IPE proposals as well as our generic transformations. For the following full function privacy of private-key FE schemes, we use Agrawal et al.'s definition [26] (and so Brakerski and Segev's definition [20]).

Definition 3 (Full Function Privacy)**.** *A private-key functional encryption* FE *over a message space $\mathcal{M}$ and a function space $\mathcal{F}$ is* fully function private *(denoted by $\mathsf{IND}_{\mathsf{ad}}^{\mathsf{FFP}}$) if for any probabilistic polynomial-time algorithm $\mathcal{A}$, which has two oracle accesses $\mathsf{KG}_b(\mathsf{mk}, f_0, f_1) := \mathsf{KG}(\mathsf{mk}, f_b)$ and $\mathsf{Enc}_b(\mathsf{mk}, \boldsymbol{x}_0, \boldsymbol{x}_1) := \mathsf{Enc}(\mathsf{mk}, \boldsymbol{x}_b)$ for her choices $f_0, f_1 \in \mathcal{F}$ and $\boldsymbol{x}_0, \boldsymbol{x}_1 \in \mathcal{M}$ with the restriction $f_0(\boldsymbol{x}_0) = f_1(\boldsymbol{x}_1)$, $|\boldsymbol{x}_0| = |\boldsymbol{x}_1|$, and $|f_0| = |f_1|$, there exists a negligible function ε in λ such that*

$$\mathsf{Setup}(1^\lambda) \to \mathsf{mk} \textit{ and}$$

$$\left| \Pr\left[\mathcal{A}^{\mathsf{KG}_0(\mathsf{mk},\cdot,\cdot),\mathsf{Enc}_0(\mathsf{mk},\cdot,\cdot)}(\lambda) \to 1\right] - \Pr\left[\mathcal{A}^{\mathsf{KG}_1(\mathsf{mk},\cdot,\cdot),\mathsf{Enc}_1(\mathsf{mk},\cdot,\cdot)}(\lambda) \to 1\right] \right| \leq \varepsilon,$$

where the probability is taken over all the randomness used in algorithms/oracles of FE *and $\mathcal{A}$.*

We also define a weaker notion of function privacy, which is firstly introduced by Bishop, Jain, and Kowalczyk [21]. Bishop et al. used this notion for the security of their functional encryption scheme with inner-product functionality. We extend Bishop et al.'s notion for inner-product functionality to general functionality.

Definition 4 (Weak Function Privacy)**.** *A private-key functional encryption* FE *over a message space* $\mathcal{M}$ *and a function space* $\mathcal{F}$ *is* weakly function private *(denoted by* $\mathsf{IND}_{\mathsf{ad}}^{\mathsf{WFP}}$*) if for any probabilistic polynomial-time algorithm* $\mathcal{A}$*, which has two oracle accesses* $\mathsf{KG}_b(\mathsf{mk}, f_0, f_1) := \mathsf{KG}(\mathsf{mk}, f_b)$ *and* $\mathsf{Enc}_b(\mathsf{mk}, \boldsymbol{x}_0, \boldsymbol{x}_1) := \mathsf{Enc}(\mathsf{mk}, \boldsymbol{x}_b)$ *for her choices* $f_0, f_1 \in \mathcal{F}$ *and* $\boldsymbol{x}_0, \boldsymbol{x}_1 \in \mathcal{M}$ *with the restriction* $f_0(\boldsymbol{x}_0) = f_1(\boldsymbol{x}_1) = f_0(\boldsymbol{x}_1) = f_1(\boldsymbol{x}_0)$*,* $|\boldsymbol{x}_0| = |\boldsymbol{x}_1|$*, and* $|f_0| = |f_1|$*, there exists a negligible function* ε *in* λ *such that*

$$\mathsf{Setup}(1^\lambda) \to \mathsf{mk} \text{ and}$$

$$\left| \Pr\left[\mathcal{A}^{\mathsf{KG}_0(\mathsf{mk},\cdot,\cdot),\mathsf{Enc}_0(\mathsf{mk},\cdot,\cdot)}(\lambda) \to 1\right] - \Pr\left[\mathcal{A}^{\mathsf{KG}_1(\mathsf{mk},\cdot,\cdot),\mathsf{Enc}_1(\mathsf{mk},\cdot,\cdot)}(\lambda) \to 1\right] \right| \leq \varepsilon,$$

where the probability is taken over all the randomness used in algorithms/oracles of FE *and* $\mathcal{A}$*.*

2.2. Background on Mathematical Structure and Assumption

We provide a background on mathematical structure and complexity assumption used in our IPE construction and its security analysis.

Definition 5 (Asymmetric Bilinear Map)**.** *We say an algorithm* $\mathcal{G}_{\mathsf{ABG}}$ *is an* asymmetric bilinear group generator *if it takes a security parameter* λ *as input and outputs* $(q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e)$ *where* $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ *are groups of prime order* q *of* λ*-bit length and a map* $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ *satisfies the following properties:*

1. *(bilinearity)* $\forall\ a, b \in \mathbb{Z}_q$ *and* $g_1 \in \mathbb{G}_1, \bar{g}_2 \in \mathbb{G}_2$*,* $e(g_1^a, \bar{g}_2^b) = e(g_1, \bar{g}_2)^{ab}$
2. *(non-degeneracy)* $\exists\ g_1 \in G_1, \bar{g}_2 \in \mathbb{G}_2$*,* $e(g_1, \bar{g}_2)$ *is a generator of* $\mathbb{G}_T$*.*

We assume that group operations in $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ *and bilinear operation are efficiently computable in polynomial time with respect to* λ*.*

Definition 6 (Symmetric eXternal Diffie-Hellman assumption (SXDH))**.** *Let* $(q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e) \leftarrow \mathcal{G}_{\mathsf{ABG}}(1^\lambda)$ *be an asymmetric bilinear group. The Decisional Diffie-Hellman (DDH) problem in* $\mathbb{G}_1$ *is to distinguish between the two distributions* $\mathcal{D}_0 := \{(g_1, g_1^a, g_1^b, g_1^{ab}, \bar{g}_2) : g_1 \xleftarrow{\$} \mathbb{G}_1, \bar{g}_2 \xleftarrow{\$} \mathbb{G}_2, a, b \xleftarrow{\$} \mathbb{Z}_q\}$ *and* $\mathcal{D}_1 := \{(g_1, g_1^a, g_1^b, g_1^{ab+r}, \bar{g}_2) : g_1 \xleftarrow{\$} \mathbb{G}_1, \bar{g}_2 \xleftarrow{\$} \mathbb{G}_2, a, b, r \xleftarrow{\$} \mathbb{Z}_q\}$*. We can similarly define the DDH problem in* $\mathbb{G}_2$ *by changing roles between* g_1 *and* $\bar{g}_2$*. We say the Symmetric eXternal Diffie-Hellman (SXDH) assumption holds if for any polynomial time adversary* $\mathcal{A}$*, both advantages of* $\mathcal{A}$ *in solving the DDH problem in* $\mathbb{G}_1$ *and the DDH problem in* $\mathbb{G}_2$ *are negligible in* λ*, where the advantage* $\boldsymbol{Adv}_{\mathcal{A},\mathbb{G}_i}^{\mathsf{DDH}}(\lambda)$ *is defined as follows.*

$$\boldsymbol{Adv}_{\mathcal{A},\mathbb{G}_i}^{\mathsf{DDH}}(\lambda) := \left| \Pr[\mathcal{A}(\mathcal{D}_\beta) \to \beta] - \frac{1}{2} \right|,$$

$\mathsf{Setup}(1^\lambda)$: Run $\mathsf{Setup1}(1^\lambda) \to \mathsf{mk1}$ and $\mathsf{Setup2}(1^\lambda) \to \mathsf{mk2}$. Output $\mathsf{mk} = \{\mathsf{mk1}, \mathsf{mk2}\}$.

$\mathsf{Enc}(\mathsf{mk}, \boldsymbol{x})$: Run $\mathsf{Enc1}(\mathsf{mk1}, \boldsymbol{x}) \to \mathsf{ct1}$, and then $\mathsf{KG2}(\mathsf{mk2}, F_{\mathsf{ct1}}) \to \mathsf{sk2}_{F_{\mathsf{ct1}}}$, where the function F_{ct1} is defined in Figure 3. Output $\mathsf{ct} = \mathsf{sk2}_{F_{\mathsf{ct1}}}$.

$\mathsf{KG}(\mathsf{mk}, f)$: Run $\mathsf{KG1}(\mathsf{mk1}, f) \to \mathsf{sk1}_f$, and then $\mathsf{Enc2}(\mathsf{mk2}, \mathsf{sk1}_f) \to \mathsf{ct2}$. Output $\mathsf{sk} = \mathsf{ct2}$.

$\mathsf{Dec}(\mathsf{ct}, \mathsf{sk})$: Run $\mathsf{Dec2}(\mathsf{sk}, \mathsf{ct})$ and output its result.

Figure 2: The 1st Transformation $\mathsf{Trans1}$

$$F_{\mathsf{ct1}}(\cdot) = \mathsf{Dec1}(\mathsf{ct1}, \cdot)$$

Figure 3: The function $F_{\mathsf{ct1}}(\cdot)$

where the probability goes over all the randomness used in $\mathcal{A}$ and the generation $\beta \xleftarrow{\$} \{0,1\}$ and $\mathcal{D}_\beta$.

3. Our Transformation for Weak Function Privacy

In this section, we first present our first transformation $\mathsf{Trans1}$. We then analyze security of the transformed FE scheme; we basically utilize $\mathsf{Trans1}$ for achieving $\mathsf{IND}^{\mathsf{WFP}}_{\mathsf{ad}}$ security. Interestingly, we also show that $\mathsf{Trans1}$ can achieve the simulation-based full function privacy if the underlying FE schemes satisfy the simulation-based message privacy.

3.1. Description of $\mathsf{Trans1}$

Let $\mathsf{FE1}$ and $\mathsf{FE2}$ be private-key FE schemes over message spaces $\mathcal{M}_1$ and $\mathcal{M}_2$, respectively, and function spaces $\mathcal{F}_1$ and $\mathcal{F}_2$, respectively, where each scheme consists of four algorithms $\{\mathsf{Setup}\mathsf{i}, \mathsf{Enc}\mathsf{i}, \mathsf{KG}\mathsf{i}, \mathsf{Dec}\mathsf{i}\}$ for $\mathsf{i} \in \{1,2\}$, respectively. We show that if $\mathcal{F}_2$ contains some specific forms of functions defined in Figure 3, then we can transfer $\mathsf{FE1}$ to a private-key FE scheme $\mathsf{FE} = \{\mathsf{Setup}, \mathsf{Enc}, \mathsf{KG}, \mathsf{Dec}\}$ over a message space $\mathcal{M} = \mathcal{M}_1$ and a function space $\mathcal{F} = \mathcal{F}_1$. We provide the precise description of $\mathsf{Trans1}$ in Figure 2.

Assumption on the output of $\mathsf{KG1}$. We assume that for each $f \in \mathcal{F}_1$, the output $\mathsf{sk1}_f$ of the key generation algorithm $\mathsf{KG1}(\mathsf{mk1}, f)$ has the same bit-length. If FE_1 does not satisfy this assumption, we can easily modify $\mathsf{KG1}$ and $\mathsf{Dec1}$; pad empty symbols to the output of $\mathsf{KG1}$ and remove it at the beginning of $\mathsf{Dec1}$.

Correctness. Let $\mathsf{Enc}(\mathsf{mk}, \boldsymbol{x}) \to \mathsf{ct}$ and $\mathsf{KG}(\mathsf{mk}, f) \to \mathsf{sk}$. Then, $\mathsf{ct} = \mathsf{sk2}_{F_{\mathsf{ct1}}}$ and $\mathsf{sk} = \mathsf{ct2}$, where F_{ct1} is defined as in Figure 3 and $\mathsf{Enc2}(\mathsf{mk2}, \mathsf{sk1}_f) \to \mathsf{ct2}$. If we run $\mathsf{Dec2}(\mathsf{sk}, \mathsf{ct})$, it is equivalent to running $\mathsf{Dec2}(\mathsf{ct2}, \mathsf{sk2}_{F_{\mathsf{ct1}}})$ for an encryption $\mathsf{ct2}$ of $\mathsf{sk1}_f$ and a decryption key $\mathsf{sk2}_{F_{\mathsf{ct1}}}$ of the function F_{ct1}. Therefore, by the correctness of FE2, we obtain $F_{\mathsf{ct1}}(\mathsf{sk1}_f) = \mathsf{Dec1}(\mathsf{ct1}, \mathsf{sk1}_f) \to f(\boldsymbol{x})$.

3.2. IND-based Security of Transformed FE Schemes

The IND-based security of the resulting scheme via Trans1 is given in the following theorem:

Theorem 1. *thm If both* FE1 *and* FE2 *are* $\mathsf{IND}^{\mathsf{MP}}_{\mathsf{ad}}$*-secure, then the resulting scheme* FE *via* Trans1 *is* $\mathsf{IND}^{\mathsf{WFP}}_{\mathsf{ad}}$*-secure.*

Proof. We prove the theorem by the standard hybrid argument. First, we define a series of games as follows:

- $\boldsymbol{Game}_0$: In this game, $\mathcal{A}$ has oracle accesses to $\mathsf{KG}_0(\mathsf{mk}, \cdot, \cdot)$ and $\mathsf{Enc}_0(\mathsf{mk}, \cdot, \cdot)$.
- $\boldsymbol{Game}_1$: This is a game with oracles $\mathsf{KG}_0(\mathsf{mk}, \cdot, \cdot)$ and $\mathsf{Enc}_1(\mathsf{mk}, \cdot, \cdot)$.
- $\boldsymbol{Game}_2$: In this game, the oracles $\mathsf{KG}_1(\mathsf{mk}, \cdot, \cdot)$ and $\mathsf{Enc}_1(\mathsf{mk}, \cdot, \cdot)$ are used for all $\mathcal{A}$'s queries.

We will prove that the difference of $\mathcal{A}$'s output distribution between any consecutive games is negligible. Then, by the triangle inequality, we conclude that $\mathcal{A}$'s advantage is negligible.

Difference between $\boldsymbol{Game}_0$ *and* $\boldsymbol{Game}_1$. We assume that there exists an adversary $\mathcal{A}$ having a non-negligible advantage in distinguishing two game $\boldsymbol{Game}_0$ and $\boldsymbol{Game}_1$. More precisely, for some non-negligible function ε_0,

$$\left| \Pr\left[\mathcal{A}^{\mathsf{KG}_0(\mathsf{mk},\cdot,\cdot),\mathsf{Enc}_0(\mathsf{mk},\cdot,\cdot)}(\lambda) \to 1\right] - \Pr\left[\mathcal{A}^{\mathsf{KG}_0(\mathsf{mk},\cdot,\cdot),\mathsf{Enc}_1(\mathsf{mk},\cdot,\cdot)}(\lambda) \to 1\right] \right| > \varepsilon_0.$$

Then, we show that there exists an adversary $\mathcal{B}$ breaking the message privacy $\mathsf{IND}^{\mathsf{MP}}_{\mathsf{ad}}$ of FE1. We first describe $\mathcal{B}$ and analyze later. $\mathcal{B}$ generates FE2 by itself; that is, it runs $\mathsf{Setup2}(1^\lambda) \to \mathsf{mk2}$ and uses mk2 for Enc2 and KG2 algorithms. Note that $\mathcal{B}$ has oracle accesses $\mathsf{Enc1}_b(\mathsf{mk1}, \cdot, \cdot)$ and $\mathsf{KG1}(\mathsf{mk1}, \cdot)$, where b is randomly chosen by the challenger.

- $\mathsf{KG}_0(\mathsf{mk}, \cdot, \cdot)$: On $\mathcal{A}$'s query (f_0, f_1), $\mathcal{B}$ sends f_0 to its own oracle $\mathsf{KG1}(\mathsf{mk1}, \cdot)$ oracle and receive $\mathsf{sk1}_{f_0}$. Then $\mathcal{B}$ runs $\mathsf{Enc2}(\mathsf{mk2}, \mathsf{sk1}_{f_0}) \to \mathsf{ct2}$ and returns the result $\mathsf{sk} = \mathsf{ct2}$.
- $\mathsf{Enc}_b(\mathsf{mk}, \cdot, \cdot)$: On $\mathcal{A}$'s query $(\boldsymbol{x}_0, \boldsymbol{x}_1)$, $\mathcal{B}$ deliveries it to its own oracle $\mathsf{Enc}_b(\mathsf{mk1}, \cdot, \cdot) \to \mathsf{ct1}^*$. Then, $\mathcal{B}$ runs $\mathsf{KG2}(\mathsf{mk2}, F_{\mathsf{ct1}^*}) \to \mathsf{sk2}_{F_{\mathsf{ct1}^*}}$ and returns the result $\mathsf{ct} = \mathsf{sk2}_{F_{\mathsf{ct1}^*}}$.

If $\mathcal{A}$ outputs her guess b', then $\mathcal{B}$ sends b' to the challenger.

Now, we analyze $\mathcal{B}$'s advantage. It is trivial to check the correctness of oracle answers; KG_0's output is well distributed regardless of b since $\mathsf{mk1}$ and $\mathsf{mk2}$ are normally used by $\mathsf{KG1}$ oracle and $\mathcal{B}$, respectively. Enc_b's output is also well distributed according to the challenger's choice b. Next, we have to show that all $\mathcal{B}$'s queries hold the necessary condition $f(\boldsymbol{x}_0) = f(\boldsymbol{x}_1)$. This directly comes from the condition $f_0(\boldsymbol{x}_0) = f_0(\boldsymbol{x}_1) = f_1(\boldsymbol{x}_0) = f_1(\boldsymbol{x}_1)$ for weak function privacy of FE. Therefore, $\mathcal{B}$'s advantage in distinguishing two oracles $\mathsf{Enc1}_0$ and $\mathsf{Enc1}_1$ is exactly the same as $\mathcal{A}$'s advantage in distinguishing two games $\boldsymbol{Game}_0$ and $\boldsymbol{Game}_1$.

Difference between $\boldsymbol{Game}_1$ *and* $\boldsymbol{Game}_2$. Similar to the above proof, we also assume that there exists an adversary $\mathcal{A}$ having a non-negligible advantage in distinguishing two game $\boldsymbol{Game}_1$ and $\boldsymbol{Game}_2$. More precisely, for some non-negligible function ε_1,

$$\left| \Pr\left[\mathcal{A}^{\mathsf{KG}_0(\mathsf{mk},\cdot,\cdot),\mathsf{Enc}_1(\mathsf{mk},\cdot,\cdot)}(\lambda) \to 1\right] - \Pr\left[\mathcal{A}^{\mathsf{KG}_1(\mathsf{mk},\cdot,\cdot),\mathsf{Enc}_1(\mathsf{mk},\cdot,\cdot)}(\lambda) \to 1\right] \right| > \varepsilon_1.$$

Then, we show that there exists an adversary $\mathcal{B}$ breaking the message privacy $\mathsf{IND}^{\mathsf{MP}}_{\mathsf{ad}}$ of $\mathsf{FE2}$. We first describe $\mathcal{B}$ and analyze later. $\mathcal{B}$ generates $\mathsf{FE1}$ by itself; that is, it runs $\mathsf{Setup1}(1^\lambda) \to \mathsf{mk1}$ and uses $\mathsf{mk1}$ for $\mathsf{Enc1}$ and $\mathsf{KG1}$ algorithms. Note that $\mathcal{B}$ has its own oracle accesses $\mathsf{Enc2}_b(\mathsf{mk2}, \cdot, \cdot)$ and $\mathsf{KG2}(\mathsf{mk2}, \cdot)$, where b is randomly chosen by the challenger.

$\mathsf{KG}_b(\mathsf{mk}, \cdot, \cdot)$: On $\mathcal{A}$'s query (f_0, f_1), $\mathcal{B}$ runs $\mathsf{KG1}(\mathsf{mk1}, f_0) \to \mathsf{sk1}_{f_0}$ and $\mathsf{KG1}(\mathsf{mk1}, f_1) \to \mathsf{sk1}_{f_1}$. Then $\mathcal{B}$ sends them to its own oracle $\mathsf{Enc2}_b(\mathsf{mk2}, \cdot, \cdot)$ and receives $\mathsf{ct2}$. Finally, $\mathcal{B}$ returns $\mathsf{sk} = \mathsf{ct2}$ to $\mathcal{A}$.

$\mathsf{Enc}_1(\mathsf{mk}, \cdot, \cdot)$: On $\mathcal{A}$'s query $(\boldsymbol{x}_0, \boldsymbol{x}_1)$, $\mathcal{B}$ runs $\mathsf{Enc1}(\mathsf{mk1}, \boldsymbol{x}_1) \to \mathsf{ct1}$, and deliveries it to its own oracle $\mathsf{KG2}(\mathsf{mk2}, \mathsf{ct1}) \to \mathsf{sk2}_{F_{\mathsf{ct1}}}$. $\mathcal{B}$ returns the result $\mathsf{ct} = \mathsf{sk2}_{F_{\mathsf{ct1}}}$ to $\mathcal{A}$.

If $\mathcal{A}$ outputs her guess b', then $\mathcal{B}$ sends b' to the challenger.

Now, we analyze $\mathcal{B}$'s advantage. It is trivial to check the correctness of oracle answers; Enc_1's output is well distributed regardless of b since $\mathsf{mk1}$ and $\mathsf{mk2}$ are normally used by $\mathcal{B}$ and $\mathsf{KG2}$ oracle, respectively. KG_b's output is also well distributed according to the challenger's choice b. Next, we should show that all $\mathcal{B}$'s oracle queries satisfy the necessary conditions. $\mathcal{B}$ issues message queries $(\mathsf{sk1}_{f_0}, \mathsf{sk1}_{f_1})$ and function queries F_{ct1}, where $\mathsf{ct1}$ is an encryption of $\boldsymbol{x}_1$, so that we have to check whether $F_{\mathsf{ct1}}(\mathsf{sk1}_{f_0}) = F_{\mathsf{ct1}}(\mathsf{sk1}_{f_0})$ holds for all such queries. By definition of F_{ct1}, $F_{\mathsf{ct1}}(\mathsf{sk1}_{f_0}) = \mathsf{Dec1}(\mathsf{ct1}, \mathsf{sk1}_{f_0}) = f_0(\boldsymbol{x}_1)$ and $F_{\mathsf{ct1}}(\mathsf{sk1}_{f_1}) = \mathsf{Dec1}(\mathsf{ct1}, \mathsf{sk1}_{f_1}) = f_1(\boldsymbol{x}_1)$. By the necessary condition for queries of weak function privacy, we know that $f_0(\boldsymbol{x}_1) = f_1(\boldsymbol{x}_1)$ for all $\mathcal{A}$'s queries (f_0, f_1) and $(\boldsymbol{x}_0, \boldsymbol{x}_1)$. Furthermore, our assumption on the output of $\mathsf{KG1}$, we know that $|\mathsf{sk1}_{f_0}| = |\mathsf{sk1}_{f_1}|$, so that all $\mathcal{B}$'s queries satisfy the necessary conditions for queries of message privacy. Therefore, $\mathcal{B}$'s advantage in distinguishing

two oracles $\mathsf{Enc2}_0$ and $\mathsf{Enc2}_1$ is exactly the same as $\mathcal{A}$'s advantage in distinguishing two games $\boldsymbol{Game}_1$ and $\boldsymbol{Game}_2$. □ □

Remark 2. *We note that the basic version of the Brakerski-Segev transformation, which is what they called 'a failed attempt', does not satisfy the weak function privacy. As explained in the introduction, their basic scheme is an instantiation of our transformation, where* FE1 *is implemented by using a semantically secure symmetric encryption scheme. In (the proof of) Theorem 1,* $\mathsf{IND}^{\mathsf{MP}}_{\mathsf{ad}}$ *security seems necessary for* FE1*. (If not, the simulator cannot correctly simulate* KG *oracle in the proof of the difference between* $\boldsymbol{Game}_0$ *and* $\boldsymbol{Game}_1$*.) However, FE schemes implemented by a symmetric encryption scheme can achieve only* $\mathsf{IND}^{\mathsf{MP}}_{\mathsf{non}}$ *security.*

3.3. Simulation-based Security of Transformed FE Schemes

We consider the SIM-based security of FE via Trans1. Intuitively, the SIM-based security for message privacy of FE captures the ideal security that any information the adversary is able to learn from the decryption keys and ciphertexts can be obtained by a simulator with access only to the decryption key queries and the outputs of the functionality, i.e., $f(\boldsymbol{x})$. For function privacy, SIM-security even requests a simulator to generate the decryption key and ciphertext with access only to the outputs of the functionality.

In analyzing of SIM-based security of our construction, we also separate abilities of adversaries into two cases, as we did for IND-based security notion; adversaries have adaptive access to decryption key generation and encryption oracles or no access at all to decryption key generation oracle. When a functional encryption FE achieves simulation-based message privacy against adaptive query, we say that the scheme satisfies *simulation-based message privacy against adaptive query* ($\mathsf{SIM}^{\mathsf{MP}}_{\mathsf{ad}}$ - secure, for short). In the case with non-key query, we say that the scheme satisfies *simulation-based message privacy against non-key query* ($\mathsf{SIM}^{\mathsf{MP}}_{\mathsf{non}}$ - secure, for short). Note that $\mathsf{SIM}^{\mathsf{MP}}_{\mathsf{non}}$ - security only cares about the semantic security of Enc of FE. When a functional encryption FE furthermore achieves simulation-based function privacy against adaptive query, we say that the scheme satisfies *simulation-based function privacy against adaptive query* ($\mathsf{SIM}^{\mathsf{FFP}}_{\mathsf{ad}}$ - secure, for short). For the formal definitions for the above SIM-security notions, refer to Appendix Appendix A, which we borrow from Kim et al. [23]. We now analyze SIM-security of the transformed schemes via Trans1 in the following theorems. We refer the reader to Appendix Appendix B for the proof.

Theorem 2. *If* FE1 *is* $\mathsf{SIM}^{\mathsf{MP}}_{\mathsf{non}}$*- secure and* FE2 *is* $\mathsf{SIM}^{\mathsf{MP}}_{\mathsf{ad}}$*- secure, then the transformed scheme* FE *via* Trans1 *is* $\mathsf{SIM}^{\mathsf{FFP}}_{\mathsf{ad}}$*- secure.*

Remark 3. *In the above theorem,* $\mathsf{SIM}^{\mathsf{FFP}}_{\mathsf{ad}}$*- security of the resulting scheme only relies on the semantic security of* Enc *in* FE1 *differently from* FE2*. This points out that the basic version of the Brakerski-Segev transformation also generates* $\mathsf{SIM}^{\mathsf{FFP}}_{\mathsf{ad}}$*- secure FE.*

$\mathsf{Setup}(1^\lambda)$: Run $\mathsf{Setup1}(1^\lambda)$ twice and obtain $\mathsf{mk1}$ and $\widehat{\mathsf{mk1}}$. Run $\mathsf{Setup2}(1^\lambda) \to \mathsf{mk2}$. Output $\mathsf{mk} = \{\mathsf{mk1}, \widehat{\mathsf{mk1}}, \mathsf{mk2}\}$.

$\mathsf{Enc}(\mathsf{mk}, \boldsymbol{x})$: Run $\mathsf{Enc1}(\mathsf{mk1}, \boldsymbol{x}) \to \mathsf{ct1}$ and $\mathsf{Enc1}(\widehat{\mathsf{mk1}}, \boldsymbol{x}) \to \widehat{\mathsf{ct1}}$, and then run $\mathsf{KG2}(\mathsf{mk2}, F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}) \to \mathsf{sk2}_{F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}}$, where the function $F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}$ is defined in Figure 5. Output $\mathsf{ct} = \mathsf{sk2}_{F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}}$.

$\mathsf{KG}(\mathsf{mk}, f)$: Run $\mathsf{KG1}(\mathsf{mk1}, f) \to \mathsf{sk1}_f$, and then $\mathsf{Enc2}\big(\mathsf{mk2}, (\mathsf{sk1}_f, \perp)\big) \to \mathsf{ct2}$. Output $\mathsf{sk} = \mathsf{ct2}$.

$\mathsf{Dec}(\mathsf{ct}, \mathsf{sk})$: Run $\mathsf{Dec2}(\mathsf{sk}, \mathsf{ct})$ and output its result.

Figure 4: Fully Function-Private Functional Encryption

The domain of $F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}(\cdot,\cdot)$ is $\big(R \times \{\perp\}\big) \bigcup \big(\{\perp\} \times R\big)$, where R is the range of $\mathsf{KG1}$ and $\perp \in R$.

$$F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}(k, \widehat{k}) \text{ outputs } \begin{cases} \perp & \text{if } (k, \widehat{k}) = (\perp, \perp) \\ \mathsf{Dec1}(\mathsf{ct1}, k) & \text{if } k \neq \perp \\ \mathsf{Dec1}(\widehat{\mathsf{ct1}}, \widehat{k}) & \text{if } \widehat{k} \neq \perp \end{cases}$$

Figure 5: The function $F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}$

4. Our Transformation for Full Function Privacy

In this section, we present our second transformation $\mathsf{Trans2}$ that is obtained by applying Naor-Yung double encryption technique [36] to $\mathsf{Trans1}$. We then analyze security of transformed schemes under IND-based security notion.

4.1. Description of $\mathsf{Trans2}$

Similar to our weak function-private construction, we use two private-key functional encryption schemes $\mathsf{FE1}$ and $\mathsf{FE2}$, where each consists of four algorithms $\{\mathsf{Setupi}, \mathsf{Enci}, \mathsf{KGi}, \mathsf{Deci}\}$ for $\mathsf{i} \in \{1, 2\}$.

Assumption on the output of $\mathsf{KG1}$. Similar to weakly function-private FE construction, we assume that for each $f \in \mathcal{F}_1$, the output of the key generation algorithm $\mathsf{KG1}(\cdot,\cdot)$ has the same bit-length, regardless of the input. (We may employ an appropriate padding process.) In addition, we assume that $\perp$ is a special symbol with the same bit-length representation as other decryption keys $\mathsf{sk1}_f$.

Games	KG Query (f_0, f_1)	Enc Query $(\boldsymbol{x}_0, \boldsymbol{x}_1)$
$\boldsymbol{Game}_0$	$\mathsf{KG1}(\mathsf{mk1}, f_0) \to \mathsf{sk1}_{f_0}$, $\mathsf{Enc2}\big(\mathsf{mk2}, (\mathsf{sk1}_{f_0}, \bot)\big)$	$\mathsf{Enc1}(\mathsf{mk1}, \boldsymbol{x}_0) \to \mathsf{ct1}$, $\mathsf{Enc1}(\widehat{\mathsf{mk1}}, \boldsymbol{x}_0) \to \widehat{\mathsf{ct1}}$, $\mathsf{KG2}(\mathsf{mk2}, F_{\mathsf{ct1},\widehat{\mathsf{ct1}}})$
$\boldsymbol{Game}_1$	$\mathsf{KG1}(\mathsf{mk1}, f_0) \to \mathsf{sk1}_{f_0}$, $\mathsf{Enc2}\big(\mathsf{mk2}, (\mathsf{sk1}_{f_0}, \bot)\big)$	$\mathsf{Enc1}(\mathsf{mk1}, \boldsymbol{x}_0) \to \mathsf{ct1}$, $\mathsf{Enc1}(\widehat{\mathsf{mk1}}, \boxed{\boldsymbol{x}_1}) \to \widehat{\mathsf{ct1}}$, $\mathsf{KG2}(\mathsf{mk2}, F_{\mathsf{ct1},\widehat{\mathsf{ct1}}})$
$\boldsymbol{Game}_2$	$\mathsf{KG1}(\widehat{\mathsf{mk1}}, f_1) \to \widehat{\mathsf{sk1}}_{f_1}$, $\mathsf{Enc2}\big(\mathsf{mk2}, \boxed{(\bot, \widehat{\mathsf{sk1}}_{f_1})}\big)$	$\mathsf{Enc1}(\mathsf{mk1}, \boldsymbol{x}_0) \to \mathsf{ct1}$, $\mathsf{Enc1}(\widehat{\mathsf{mk1}}, \boldsymbol{x}_1) \to \widehat{\mathsf{ct1}}$, $\mathsf{KG2}(\mathsf{mk2}, F_{\mathsf{ct1},\widehat{\mathsf{ct1}}})$
$\boldsymbol{Game}_3$	$\mathsf{KG1}(\widehat{\mathsf{mk1}}, f_1) \to \widehat{\mathsf{sk1}}_{f_1}$, $\mathsf{Enc2}\big(\mathsf{mk2}, (\bot, \widehat{\mathsf{sk1}}_{f_1})\big)$	$\mathsf{Enc1}(\mathsf{mk1}, \boxed{\boldsymbol{x}_1}) \to \mathsf{ct1}$, $\mathsf{Enc1}(\widehat{\mathsf{mk1}}, \boldsymbol{x}_1) \to \widehat{\mathsf{ct1}}$, $\mathsf{KG2}(\mathsf{mk2}, F_{\mathsf{ct1},\widehat{\mathsf{ct1}}})$
$\boldsymbol{Game}_4$	$\mathsf{KG1}(\mathsf{mk1}, f_1), \bot) \to \mathsf{sk1}_{f_1}$, $\mathsf{Enc2}\big(\mathsf{mk2}, \boxed{(\mathsf{sk1}_{f_1}, \bot)}\big)$	$\mathsf{Enc1}(\mathsf{mk1}, \boldsymbol{x}_1) \to \mathsf{ct1}$, $\mathsf{Enc1}(\widehat{\mathsf{mk1}}, \boldsymbol{x}_1) \to \widehat{\mathsf{ct1}}$, $\mathsf{KG2}(\mathsf{mk2}, F_{\mathsf{ct1},\widehat{\mathsf{ct1}}})$

Table 2: The description of oracle outputs in games $\boldsymbol{Game}_0, \ldots, \boldsymbol{Game}_4$. $F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}$ is defined in Figure 5.

Correctness. Let $\mathsf{Enc}(\mathsf{mk}, \boldsymbol{x}) \to \mathsf{ct}$ and $\mathsf{KG}(\mathsf{mk}, f) \to \mathsf{sk}$. Then, $\mathsf{ct} = \mathsf{sk2}_{F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}}$ and $\mathsf{sk} = \mathsf{ct2}$, where $F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}$ is defined as in Figure 3 and $\mathsf{Enc2}\big(\mathsf{mk2}, (\mathsf{sk1}_f, \bot)\big) \to \mathsf{ct2}$. If we run $\mathsf{Dec2}(\mathsf{sk}, \mathsf{ct})$, it is equivalent to running $\mathsf{Dec2}(\mathsf{ct2}, \mathsf{sk2}_{F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}})$ for an encryption $\mathsf{ct2}$ of $(\mathsf{sk1}_f, \bot)$ and a decryption key $\mathsf{sk2}_{F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}}$ of the function $F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}$. Therefore, by the correctness of $\mathsf{FE2}$, we obtain $F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}(\mathsf{sk1}_f, \bot) = \mathsf{Dec1}(\mathsf{ct1}, \mathsf{sk1}_f) \to f(\boldsymbol{x})$.

4.2. IND-base Security of Transformed FE Schemes

The IND-based security of the transformed FE schemes by $\mathsf{Trans2}$ is captured as follows:

Theorem 3. *If* $\mathsf{FE1}$ *is* $\mathsf{IND}^{\mathsf{MP}}_{\mathsf{non}}$*-secure and* $\mathsf{FE2}$ *is* $\mathsf{IND}^{\mathsf{MP}}_{\mathsf{ad}}$*-secure, then the transformed scheme* FE *in Figure 4 is* $\mathsf{IND}^{\mathsf{FFP}}_{\mathsf{ad}}$*-secure.*

Proof. We begin with defining a series of games, in which where $\mathcal{A}$ has different oracle accesses. The i-th game is completely specified by its key generation oracle, denoted by $\mathsf{KG}^{(i)}(\mathsf{mk}, \cdot, \cdot)$ and its encryption oracle, denoted by $\mathsf{Enc}^{(i)}(\mathsf{mk}, \cdot, \cdot)$. In Table 2, we give the description of the oracle outputs in a series of games $\boldsymbol{Game}_0, \ldots, \boldsymbol{Game}_4$.

Difference between $\boldsymbol{Game}_0$ *and* $\boldsymbol{Game}_1$*..* Assuming the existence of an adversary $\mathcal{A}$ having a non-negligible advantage in distinguishing two game $\boldsymbol{Game}_0$ and $\boldsymbol{Game}_1$, we construct a simulator $\mathcal{B}$ breaking the message privacy $\mathsf{IND}^{\mathsf{MP}}_{\mathsf{non}}$ of $\mathsf{FE1}$. More precisely, we assume that there exists $\mathcal{A}$ and a non-negligible function ε_0 in λ such that

$$\Big| \Pr\big[\mathcal{A}^{\mathsf{KG}^{(0)}(\mathsf{mk},\cdot,\cdot),\mathsf{Enc}^{(0)}(\mathsf{mk},\cdot,\cdot)}(\lambda) \to 1\big] - \Pr\big[\mathcal{A}^{\mathsf{KG}^{(1)}(\mathsf{mk},\cdot,\cdot),\mathsf{Enc}^{(1)}(\mathsf{mk},\cdot,\cdot)}(\lambda) \to 1\big] \Big| > \varepsilon_0.$$

Then, we will show that our $\mathcal{B}$ construction also has the same advantage ε_0 against $\mathsf{IND}^{\mathsf{MP}}_{\mathsf{non}}$ security of $\mathsf{FE1}$.

Now, we describe $\mathcal{B}$'s oracle simulations. At the very beginning, $\mathcal{B}$ runs $\mathsf{Setup2}(1^\lambda) \to \mathsf{mk2}$ and $\mathsf{Setup1}(1^\lambda) \to \mathsf{mk1}$, and keep the master secret keys $\mathsf{mk2}$ and $\mathsf{mk1}$ during the simulation.

$\mathsf{KG}(\mathsf{mk}, \cdot, \cdot)$: On input (f_0, f_1), $\mathcal{B}$ returns the output of $\mathsf{Enc2}\big(\mathsf{mk2}, (\mathsf{KG1}(\mathsf{mk1}, f_0), \bot)\big)$. ($\mathcal{B}$ already knows $\mathsf{mk2}$ and $\mathsf{mk1}$.)

$\mathsf{Enc}(\mathsf{mk}, \cdot, \cdot)$: On input $(\boldsymbol{x}_0, \boldsymbol{x}_1)$, $\mathcal{B}$ queries $(\boldsymbol{x}_0, \boldsymbol{x}_1)$ to its own oracle $\mathsf{Enc1}_b(\widehat{\mathsf{mk1}}, \cdot, \cdot) \to \widehat{\mathsf{ct1}}$. $\mathcal{B}$ computes $\mathsf{Enc1}(\mathsf{mk1}, \boldsymbol{x}_0) \to \mathsf{ct1}$ and $\mathsf{KG2}(\mathsf{mk2}, F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}) \to \mathsf{sk2}_{F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}}$, and then deliveries $\mathsf{sk2}_{F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}}$ to $\mathcal{A}$.

Finally, if $\mathcal{A}$ outputs b', then $\mathcal{B}$ sends it to the challenger. It is trivial that the challenge bit b, which is used in the oracle $\mathsf{Enc1}_b(\widehat{\mathsf{mk1}}, \cdot, \cdot)$, exactly determines whether $\mathcal{B}$ perfectly simulates $\boldsymbol{Game}_0$ or $\boldsymbol{Game}_1$. Therefore, we conclude that $\mathcal{B}$'s advantage is exactly equal to ε_0.

Difference between $\boldsymbol{Game}_1$ and $\boldsymbol{Game}_2$.. Assuming the existence of an adversary $\mathcal{A}$ having a non-negligible advantage in distinguishing two game $\boldsymbol{Game}_1$ and $\boldsymbol{Game}_2$, we construct a simulator $\mathcal{B}$ breaking the message privacy $\mathsf{IND}^{\mathsf{MP}}_{\mathsf{ad}}$ of $\mathsf{FE2}$. More precisely, we assume that there exists $\mathcal{A}$ and a non-negligible function ε_1 in λ such that

$$\left| \Pr\left[\mathcal{A}^{\mathsf{KG}^{(1)}(\mathsf{mk},\cdot,\cdot),\mathsf{Enc}^{(1)}(\mathsf{mk},\cdot,\cdot)}(\lambda) \to 1\right] - \Pr\left[\mathcal{A}^{\mathsf{KG}^{(2)}(\mathsf{mk},\cdot,\cdot),\mathsf{Enc}^{(2)}(\mathsf{mk},\cdot,\cdot)}(\lambda) \to 1\right] \right| > \varepsilon_1.$$

Then, we will show that our $\mathcal{B}$ construction also has the same advantage ε_1 against the message privacy $\mathsf{IND}^{\mathsf{MP}}_{\mathsf{ad}}$ of $\mathsf{FE2}$.

Now, we describe $\mathcal{B}$'s oracle simulations. At the very beginning, $\mathcal{B}$ runs $\mathsf{Setup1}(1^\lambda) \to \mathsf{mk1}$ and $\mathsf{Setup1}(1^\lambda) \to \widehat{\mathsf{mk1}}$, and keep the master secret keys $\mathsf{mk1}$ and $\widehat{\mathsf{mk1}}$ during the simulation.

$\mathsf{KG}(\mathsf{mk}, \cdot, \cdot)$: On input (f_0, f_1), $\mathcal{B}$ computes $\mathsf{KG1}(\mathsf{mk1}, f_0) \to \mathsf{sk1}_{f_0}$ and $\mathsf{KG1}(\widehat{\mathsf{mk1}}, f_1) \to \widehat{\mathsf{sk1}}_{f_1}$. $\mathcal{B}$ sends $(\mathsf{sk1}_{f_0}, \bot)$ and $(\bot, \widehat{\mathsf{sk1}}_{f_1})$ to its own oracle $\mathsf{Enc2}_b(\mathsf{mk2}, \cdot, \cdot)$. $\mathcal{B}$ deliveries the query result to $\mathcal{A}$.

$\mathsf{Enc}(\mathsf{mk}, \cdot, \cdot)$: On input $(\boldsymbol{x}_0, \boldsymbol{x}_1)$, $\mathcal{B}$ runs $\mathsf{Enc1}(\mathsf{mk1}, \boldsymbol{x}_0) \to \mathsf{ct1}$ and $\mathsf{Enc1}(\widehat{\mathsf{mk1}}, \boldsymbol{x}_1) \to \widehat{\mathsf{ct1}}$. Next, $\mathcal{B}$ sends $F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}$ to $\mathsf{KG2}(\mathsf{mk2}, \cdot)$ oracle and obtains $\mathsf{sk2}_{F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}}$. Then, $\mathcal{B}$ returns the query result to $\mathcal{A}$.

In $\mathcal{B}$'s simulation for KG oracle, $\mathcal{B}$ used the output of the oracle $\mathsf{Enc2}_b(\mathsf{mk2}, \cdot, \cdot)$. For a while, let us assume that all $\mathcal{B}$'s queries satisfy the necessary conditions for oracle queries, and show it later. If $b = 0$, $\mathcal{B}$'s simulation for KG perfectly becomes $\mathsf{KG}^{(1)}$. Otherwise ($b = 1$), $\mathcal{B}$ simulates perfectly $\mathsf{KG}^{(2)}$. Regardless of b, $\mathcal{B}$ perfectly simulates $\mathsf{Enc}^{(1)}(\mathsf{mk}, \cdot, \cdot) = \mathsf{Enc}^{(2)}(\mathsf{mk}, \cdot, \cdot)$ oracle. Therefore, we conclude that $\mathcal{B}$'s advantage is exactly equal to $\mathcal{A}$'s advantage ε_1.

Now, we check whether all $\mathcal{B}$'s queries satisfy the necessary conditions for oracle queries. First, by the assumption on the output of $\mathsf{KG1}$, we know that $\mathcal{B}$'s two inputs on $\mathsf{Enc2}_b(\mathsf{mk2}, \cdot, \cdot)$ are of the equal size. Second, we show that for all queries (f_0, f_1) and $(\boldsymbol{x}_0, \boldsymbol{x}_1)$ issued by $\mathcal{A}$, $F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}(\mathsf{sk1}_{f_0}, \perp) = F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}(\perp, \widehat{\mathsf{sk1}}_{f_1})$, where $(\mathsf{ct1}, \widehat{\mathsf{ct1}})$ is a pair of encryptions $\big(\mathsf{Enc1}(\mathsf{mk1}, \boldsymbol{x}_0), \mathsf{Enc1}(\widehat{\mathsf{mk1}}, \boldsymbol{x}_1)\big)$. By the definition of the function $F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}$, we have $F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}(\mathsf{sk1}_{f_0}, \perp) = \mathsf{Dec1}(\mathsf{ct1}, \mathsf{sk1}_{f_0}) = f_0(x_0) = f_1(x_1) = \mathsf{Dec1}(\widehat{\mathsf{ct1}}, \widehat{\mathsf{sk1}}_{f_1}) = F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}(\perp, \widehat{\mathsf{sk1}}_{f_1})$. This completes the proof for the difference between $\boldsymbol{Game}_1$ and $\boldsymbol{Game}_2$.

Difference between $\boldsymbol{Game}_2$ *and* $\boldsymbol{Game}_3$*..* The proof for the difference between $\boldsymbol{Game}_2$ and $\boldsymbol{Game}_3$ is exactly the same as that for the difference between $\boldsymbol{Game}_0$ and $\boldsymbol{Game}_1$, except for changing the roles of $\widehat{\mathsf{mk1}}$ and $\mathsf{mk1}$. That is, $\mathcal{B}$ generates $\widehat{\mathsf{mk1}}$ and $\mathsf{mk2}$ by itself, and plays a game for distinguishing a bit b used in the challenge oracle $\mathsf{Enc1}_b(\mathsf{mk1}, \cdot, \cdot)$. Similarly to the proof between $\boldsymbol{Game}_0$ and $\boldsymbol{Game}_1$, from the message privacy $\mathsf{IND}^{\mathsf{MP}}_{\mathsf{non}}$ of $\mathsf{FE1}$, we can prove that the advantage of arbitrary adversary distinguishing two games $\boldsymbol{Game}_2$ and $\boldsymbol{Game}_3$ should be bounded by $\mathcal{B}$'s advantage against breaking the message privacy $\mathsf{IND}^{\mathsf{MP}}_{\mathsf{non}}$ of $\mathsf{FE1}$.

Difference between $\boldsymbol{Game}_3$ *and* $\boldsymbol{Game}_4$*..* Assuming the existence of an adversary $\mathcal{A}$ having a non-negligible advantage in distinguishing two game $\boldsymbol{Game}_3$ and $\boldsymbol{Game}_4$, we construct a simulator $\mathcal{B}$ breaking the message privacy $\mathsf{IND}^{\mathsf{MP}}_{\mathsf{ad}}$ of $\mathsf{FE2}$. More precisely, we assume that there exists $\mathcal{A}$ and a non-negligible function ε_3 in λ such that

$$\Big| \Pr\left[\mathcal{A}^{\mathsf{KG}^{(3)}(\mathsf{mk},\cdot,\cdot),\mathsf{Enc}^{(3)}(\mathsf{mk},\cdot,\cdot)}(\lambda) \to 1\right] - \Pr\left[\mathcal{A}^{\mathsf{KG}^{(4)}(\mathsf{mk},\cdot,\cdot),\mathsf{Enc}^{(4)}(\mathsf{mk},\cdot,\cdot)}(\lambda) \to 1\right] \Big| > \varepsilon_3.$$

Then, we will show that our $\mathcal{B}$ construction also has the same advantage ε_3 against the message privacy $\mathsf{IND}^{\mathsf{MP}}_{\mathsf{ad}}$ of $\mathsf{FE2}$.

Now, we describe $\mathcal{B}$'s oracle simulations. At the very beginning, $\mathcal{B}$ runs $\mathsf{Setup1}(1^\lambda) \to \mathsf{mk1}$ and $\mathsf{Setup1}(1^\lambda) \to \widehat{\mathsf{mk1}}$, and keep the master secret keys $\mathsf{mk1}$ and $\widehat{\mathsf{mk1}}$ during the simulation.

$\mathsf{KG}(\mathsf{mk}, \cdot, \cdot)$: On input (f_0, f_1), $\mathcal{B}$ computes $\mathsf{KG1}(\mathsf{mk1}, f_1) \to \mathsf{sk1}_{f_1}$ and $\mathsf{KG1}(\widehat{\mathsf{mk1}}, f_1) \to \widehat{\mathsf{sk1}}_{f_1}$. $\mathcal{B}$ sends $\big((\mathsf{sk1}_{f_1}, \perp), (\perp, \widehat{\mathsf{sk1}}_{f_1})\big)$ to its own oracle $\mathsf{Enc2}_b(\mathsf{mk2}, \cdot, \cdot)$. $\mathcal{B}$ deliveries the query result to $\mathcal{A}$.

$\mathsf{Enc}(\mathsf{mk}, \cdot, \cdot)$: On input $(\boldsymbol{x}_0, \boldsymbol{x}_1)$, $\mathcal{B}$ runs $\mathsf{Enc1}(\mathsf{mk1}, \boldsymbol{x}_1) \to \mathsf{ct1}$ and $\mathsf{Enc1}(\widehat{\mathsf{mk1}}, \boldsymbol{x}_1) \to \widehat{\mathsf{ct1}}$. Next, $\mathcal{B}$ sends $F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}$ to $\mathsf{KG2}(\mathsf{mk2}, \cdot)$ oracle and obtains $\mathsf{sk2}_{F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}}$. Then, $\mathcal{B}$ returns the query result to $\mathcal{A}$.

In $\mathcal{B}$'s simulation for KG oracle, $\mathcal{B}$ used the output of the oracle $\mathsf{Enc2}_b(\mathsf{mk2}, \cdot, \cdot)$. For a while, let us assume that all $\mathcal{B}$'s queries satisfy the necessary conditions for oracle queries, and show it later. If $b = 0$, $\mathcal{B}$'s simulation for KG perfectly becomes $\mathsf{KG}^{(4)}$. Otherwise $(b = 1)$, $\mathcal{B}$ simulates perfectly $\mathsf{KG}^{(3)}$. Regardless of

b, $\mathcal{B}$ perfectly simulates $\mathsf{Enc}^{(3)}(\mathsf{mk},\cdot,\cdot) = \mathsf{Enc}^{(4)}(\mathsf{mk},\cdot,\cdot)$ oracle. Therefore, we conclude that $\mathcal{B}$'s advantage is exactly equal to $\mathcal{A}$'s advantage ε_3.

Now, we check whether all $\mathcal{B}$'s queries satisfy the necessary conditions for oracle queries. First, by the assumption on the output of $\mathsf{KG1}$, we know that $\mathcal{B}$'s two inputs on $\mathsf{Enc2}_b(\mathsf{mk2},\cdot,\cdot)$ are of the equal size. Second, we show that for all queries (f_0, f_1) and $(\boldsymbol{x}_0, \boldsymbol{x}_1)$ issued by $\mathcal{A}$, $F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}(\mathsf{sk1}_{f_1}, \perp) = F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}(\perp, \widehat{\mathsf{sk1}}_{f_1})$, where $(\mathsf{ct1}, \widehat{\mathsf{ct1}})$ is a pair of encryptions $\big(\mathsf{Enc1}(\mathsf{mk1}, \boldsymbol{x}_1), \mathsf{Enc1}(\widehat{\mathsf{mk1}}, \boldsymbol{x}_1)\big)$. By the definition of the function $F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}$, we have $F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}(\mathsf{sk1}_{f_1}, \perp) = \mathsf{Dec1}(\mathsf{ct1}, \mathsf{sk1}_{f_1}) = f_1(x_1) = \mathsf{Dec1}(\widehat{\mathsf{ct1}}, \widehat{\mathsf{sk1}}_{f_1}) = F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}(\perp, \widehat{\mathsf{sk1}}_{f_1})$. This completes the proof for the difference between $\boldsymbol{Game}_3$ and $\boldsymbol{Game}_4$. □

□

5. Application to Inner-Product Functionality

In this section, we show that our transformations can be applied for inner-product functionality. Finally, we obtain a weakly function-private IPE scheme and a fully function-private IPE scheme, both under the SXDH assumption.

For a multi-exponentiation, we use the following notation: let $\mathbb{G}$ be a group of order q, $\boldsymbol{g} = (g_1, \ldots, g_n) \in \mathbb{G}^n, h \in \mathbb{G}$, and $\boldsymbol{x} = (x_1, \ldots, x_n) \in \mathbb{Z}_q^n$. Then, $\boldsymbol{g}^{\boldsymbol{x}}$ and $h^{\boldsymbol{x}}$ denote $\prod_{i=1}^n g_i^{x_i} \in \mathbb{G}$ and $(h^{x_1}, \ldots, h^{x_n})$, respectively.

5.1. Weakly Function-Private IPE Construction

In this subsection, we propose two modifications of Agrawal, Libert and Stehlé's IPE (ALS-IPE) scheme [28]. Then, our first transformation directly implies weakly function-private IPE scheme from these two FE schemes.

First Modification: FE for Inner Product in the Exponent. The first modification is quite simple; it is a private-key version and the decryption algorithm outputs $e(g,\bar{g})^{\langle \boldsymbol{x},\boldsymbol{y}\rangle}$ instead of $\langle \boldsymbol{x},\boldsymbol{y}\rangle$, where e is an asymmetric bilinear map and g and $\bar{g}$ are generators. We provide our modification of the ALS-IPE scheme in Figure 6.

Indeed, the modified scheme is no longer IPE scheme. It outputs $e(g,\bar{g})^{\langle x,y\rangle}$ instead of $\langle x,y\rangle$, where x and y are vectors in input ciphertext and decryption key and g is a generator of $\mathbb{G}_1$. However, it does not matter to our purpose, function-private IPE. The resulting scheme transformed by $\mathsf{Trans1}$ will also output $e(g,\bar{g})^{\langle x,y\rangle}$. Then, we can add a post process of solving a DLP based on $e(g,\bar{g})$ to the decryption algorithm of the transformed scheme.

Theorem 4. *[28] The scheme given in Figure 6 is* $\mathsf{IND}^{\mathsf{MP}}_{\mathsf{ad}}$*-secure under the DDH assumption in* $\mathbb{G}_1$*.*

Second Modification: FE for Multi-exponentiation with Bilinear Map. Then, we propose another FE for $\mathsf{FE2}$ supporting multi-exponentiation with bilinear map; as described in Figure 6, the domain of $F_{\mathsf{ct1}} = \mathsf{Dec1}(\mathsf{ct1},\cdot)$ is $\mathbb{Z}_q^{n+2}$, and $F_{\mathsf{ct1}}(\mathsf{sk}) = e(\mathsf{ct}^{\mathsf{sk}}, \bar{g})$ is a multi-exponentiation with bilinear map, where $\mathsf{ct1}$ is defined in a function and sk is a message.

$\mathsf{Setup}(1^\lambda, 1^n)$: Choose an asymmetric bilinear group $\mathcal{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, e)$ of prime order $q > 2^\lambda$ with generators $g, h \xleftarrow{\$} \mathbb{G}_1$ and $\bar{g} \xleftarrow{\$} \mathbb{G}_2$. Choose $\boldsymbol{s} = (s_1, \ldots, s_n), \boldsymbol{t} = (t_1, \ldots, t_n) \xleftarrow{\$} \mathbb{Z}_q^n$ and compute $h_i = g^{s_i} h^{t_i}$ for $i = 1, \ldots, n$. Output

$$\mathsf{mk} := \big(\mathcal{G}, g, h, \{h_i, s_i, t_i\}_{i=1}^n\big) \text{ and } \mathsf{pp} := \big(\mathcal{G}, \bar{g}\big).$$

$\mathsf{Enc}(\mathsf{mk}, \boldsymbol{x})$: Given a vector $\boldsymbol{x} = (x_1, \ldots, x_n) \in \mathbb{Z}_q^n$, choose $r \xleftarrow{\$} \mathbb{Z}_q$ and output

$$\mathsf{ct}_{\boldsymbol{x}} := \big(g^r,\ h^r,\ g^{x_1} h_1^r, \ldots, g^{x_n} h_n^r\big) \in \mathbb{G}_1^{n+2}.$$

$\mathsf{KG}(\mathsf{mk}, \boldsymbol{y})$: Given a vector $\boldsymbol{y} = (y_1, \ldots, y_n) \in \mathbb{Z}_q^n$, compute and output

$$\mathsf{sk}_{\boldsymbol{y}} := \big(-\langle \boldsymbol{s}, \boldsymbol{y}\rangle,\ -\langle \boldsymbol{t}, \boldsymbol{y}\rangle,\ y_1, \ldots, y_n\big) \in \mathbb{Z}_q^{n+2}.$$

$\mathsf{Dec}(\mathsf{pp}, \mathsf{ct}, \mathsf{sk})$: Given $\mathsf{ct} \in \mathbb{G}_1^{n+2}$ and $\mathsf{sk} \in \mathbb{Z}_q^{n+2}$, compute and output

$$e(\mathsf{ct}^{\mathsf{sk}}, \bar{g}).$$

Figure 6: A Modified Agrawal-Libert-Stehlé IPE

We again modify the key generation and decryption algorithm of the ALS-IPE scheme to construct an FE scheme for F_{ct1}. In the key generation algorithm, it takes as input $\vec{g} = (g_1, \ldots, g_{n+2}) \in \mathbb{G}_1^{n+2}$ (instead of $\vec{y} = (y_1, \ldots, y_{n+2}) \in \mathbb{Z}_q^{n+2}$) and outputs a decryption key for $\vec{g}$

$$\vec{S} = (\prod_{i=1}^{n+2} g_i^{-u_i}, \prod_{i=1}^{n+2} g_i^{-v_i}, g_1, \ldots, g_{n+2}) \in \mathbb{G}_1^{n+4}, \tag{3}$$

(instead of $(-\sum_{i=1}^{n+2} u_i y_i,\ -\sum_{i=1}^{n+2} v_i y_i, y_1,\ \ldots, y_{n+2}) \in \mathbb{Z}_q^{n+4}$), where $\vec{u} = (u_1, \ldots, u_{n+2})$ and $\vec{v} = (v_1, \ldots, v_{n+2})$ are master secret key. In the decryption procedure, it computes and outputs $\prod_i^{n+4} e(S_i, C_i)$ where $\vec{C} = (C_1, \ldots, C_{n+4})$ is a ciphertext of $\vec{w}$. The full description of functional encryption for a multi-exponentiation is given in Figure 7.

Correctness. We check the correctness of the scheme in Figure 7. An encryption for a vector $\boldsymbol{w}$ is the form $\mathsf{ct}_{\boldsymbol{w}} = \big(\bar{g}^r,\ \bar{h}^r,\ \bar{g}^{w_1} \bar{h}_1^r, \ldots, \bar{g}^{w_{n+2}} \bar{h}_{n+2}^r\big)$ and a decryption key for a vector $\boldsymbol{g}$ is the form $\mathsf{sk}_{\boldsymbol{g}} = \big(\boldsymbol{g}^{-\boldsymbol{u}}, \boldsymbol{g}^{-\boldsymbol{v}},\ g_1, \ldots, g_{n+2}\big)$. Let $g_i = g^{z_i}$ for some $z_i \in \mathbb{Z}_q$; that is, $\boldsymbol{g} = g^{\boldsymbol{z}}$ for $\boldsymbol{z} = (z_1, \ldots, z_{n+2})$. Then, $\mathsf{sk}_{\boldsymbol{g}}$ can be rewritten by $(g^{-\langle \boldsymbol{u}, \boldsymbol{z}\rangle}, g^{-\langle \boldsymbol{v}, \boldsymbol{z}\rangle}, g^{z_1}, \ldots, g^{z_{n+2}}) = g^{(-\langle \boldsymbol{u}, \boldsymbol{z}\rangle, -\langle \boldsymbol{v}, \boldsymbol{z}\rangle, z_1, \ldots, z_{n+2})}$. Then, the output of Dec algorithm is

$$\begin{aligned} \textstyle\prod_{i=1}^{n+4} e(S_i, C_i) = &\ e(g, \mathsf{ct})^{(-\langle \boldsymbol{u}, \boldsymbol{z}\rangle, -\langle \boldsymbol{v}, \boldsymbol{z}\rangle, z_1, \ldots, z_{n+2})} \\ = &\ e(g, \bar{g}^{\langle \boldsymbol{w}, \boldsymbol{z}\rangle}) \\ = &\ \textstyle\prod_{i=1}^{n+2} e(g_i^{w_i}, \bar{g}). \end{aligned}$$

$\mathsf{Setup}\big(1^\lambda, (1^{n+2}, \{\mathcal{G}, \bar{g}\}_\lambda)\big)$: Choose $\bar{h} \xleftarrow{\$} \mathbb{G}_2$, $\boldsymbol{u} = (u_1, \ldots, u_{n+2}), \boldsymbol{v} = (v_1, \ldots, v_{n+2}) \xleftarrow{\$} \mathbb{Z}_q^{n+2}$ and compute $\bar{h}_i = \bar{g}^{u_i}\bar{h}^{v_i}$ for $i = 1, \ldots, n+2$. Output

$$\mathsf{mk} := \big(\mathcal{G}, \bar{g}, \bar{h}, \{\bar{h}_i, u_i, v_i\}_{i=1}^{n+2}\big) \text{ and } \mathsf{pp} := \mathcal{G}.$$

$\mathsf{Enc}(\mathsf{mk}, \boldsymbol{w})$: Given a vector $\boldsymbol{w} = (w_1, \ldots, w_{n+2}) \in \mathbb{Z}_q^{n+2}$, choose $r \xleftarrow{\$} \mathbb{Z}_q$ and output

$$\mathsf{ct}_{\boldsymbol{w}} := \big(\bar{g}^r,\ \bar{h}^r,\ \bar{g}^{w_1}\bar{h}_1^r, \ldots, \bar{g}^{w_{n+2}}\bar{h}_{n+2}^r\big) \in \mathbb{G}_2^{n+4}.$$

$\mathsf{KG}(\mathsf{mk}, \boldsymbol{g})$: Given a vector $\boldsymbol{g} = (g_1, \ldots, g_{n+2}) \in \mathbb{G}_1^{n+2}$, compute and output

$$\mathsf{sk}_{\boldsymbol{g}} := \big(\boldsymbol{g}^{-\boldsymbol{u}}, \boldsymbol{g}^{-\boldsymbol{v}},\ g_1, \ldots, g_{n+2}\big) \in \mathbb{G}_1^{n+4}.$$

$\mathsf{Dec}(\mathsf{pp}, \mathsf{ct}, \mathsf{sk})$: Given $\mathsf{ct} = (C_1, \ldots, C_{n+4}) \in \mathbb{G}_2^{n+4}$ and $\mathsf{sk} = (S_1, \ldots, S_{n+4}) \in \mathbb{G}_1^{n+4}$, compute and output

$$\prod_{i=1}^{n+4} e(S_i, C_i).$$

Figure 7: Functional encryption for a multi-exponentiation with bilinear map

The second equality holds since C_i and $(-\langle \boldsymbol{u}, \boldsymbol{z}\rangle, -\langle \boldsymbol{v}, \boldsymbol{z}\rangle, z_1, \ldots, z_{n+2})$ can be considered as an encryption of $\boldsymbol{w}$ and a decryption key for $\boldsymbol{z}$, respectively, of ALS-IPE over $(n+2)$-length vectors, where the roles of $\mathbb{G}_1$ and $\mathbb{G}_2$ are changed.

Theorem 5. *The scheme given in Figure 7 is* $\mathsf{IND}_{\mathsf{ad}}^{\mathsf{MP}}$*- secure under the DDH assumption in* $\mathbb{G}_2$*.*

The proof is essentially the same as that of ALS-IPE scheme. We give the detailed proof in Appendix Appendix B.

Weakly Function-Private IPE. By the correctness of the transformation in Figure 2, the output of the decryption algorithm of IPE in Figure 6 is equal to the transformed scheme; that is, it outputs $e(g, \bar{g})^{\langle \boldsymbol{x}, \boldsymbol{y}\rangle}$ instead of $\langle \boldsymbol{x}, \boldsymbol{y}\rangle$. Therefore, to obtain an IPE scheme, we need a post process of solving the DLP with the base $e(g, \bar{g})$.[10] The final construction for our weakly function-private IPE is given in Figure 8.

The weak function privacy of the scheme in Figure 8 is directly obtained by Theorem 1, Theorem 4, and Theorem 5. We give the resulting theorem as follows.

[10]To this end, we also require that the range of the inner product functionality should be polynomial in λ. We note that all previous function-private functional encryption schemes for inner-product functionality [21, 22, 23] also have the same process of computing discrete logarithm in their decryption algorithms and so the condition for the range of inner product.

$\mathsf{Setup}(1^\lambda, 1^n)$: Choose an asymmetric bilinear group $\mathcal{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, e)$ of prime order $q > 2^\lambda$ with generators $g, h \xleftarrow{\$} \mathbb{G}_1$ and $\bar{g}, \bar{h} \xleftarrow{\$} \mathbb{G}_2$. Choose $\boldsymbol{s} = (s_1, \ldots, s_n), \boldsymbol{t} = (t_1, \ldots, t_n) \xleftarrow{\$} \mathbb{Z}_q^n$ and $\boldsymbol{u} = (u_1, \ldots, u_{n+2}), \boldsymbol{v} = (v_1, \ldots, v_{n+2}) \xleftarrow{\$} \mathbb{Z}_q^{n+2}$. Compute $h_i = g^{s_i} h^{t_i}$ for $i = 1, \ldots, n$ and $\bar{h}_i = \bar{g}^{u_i} \bar{h}^{v_i}$ for $i = 1, \ldots, n+2$. Output

$$\mathsf{mk} := \left(\{h_i, s_i, t_i\}_{i=1}^n, \{\bar{h}_i, u_i, v_i\}_{i=1}^{n+2}\right) \text{ and } \mathsf{pp} := \left(\mathcal{G}, g, h, \bar{g}, \bar{h}\right).$$

$\mathsf{Enc}(\mathsf{mk}, \boldsymbol{x})$: Given a vector $\boldsymbol{x} = (x_1, \ldots, x_n) \in \mathbb{Z}_q^n$, choose $r \xleftarrow{\$} \mathbb{Z}_q$ and compute

$$\mathsf{ct1}_{\boldsymbol{x}} := \left(g^r, h^r, g^{x_1} h_1^r, \ldots, g^{x_n} h_n^r\right) \in \mathbb{G}_1^{n+2}.$$

Output

$$\mathsf{ct}_{\boldsymbol{x}} := \left(\mathsf{ct1}_{\boldsymbol{x}}^{-\boldsymbol{u}}, \mathsf{ct1}_{\boldsymbol{x}}^{-\boldsymbol{v}}, \mathsf{ct}_{\boldsymbol{x}} 1\right) \in \mathbb{G}_1^{n+4}.$$

$\mathsf{KG}(\mathsf{mk}, \boldsymbol{y})$: Given a vector $\boldsymbol{y} = (y_1, \ldots, y_n) \in \mathbb{Z}_q^n$, choose $\bar{r} \xleftarrow{\$} \mathbb{Z}_q$ and compute and output

$$\mathsf{sk}_{\boldsymbol{y}} := \left(\bar{g}^{\bar{r}}, \bar{h}^{\bar{r}}, \bar{g}^{-\langle \boldsymbol{s}, \boldsymbol{y} \rangle} \bar{h}_1^{\bar{r}}, \bar{g}^{-\langle \boldsymbol{t}, \boldsymbol{y} \rangle} \bar{h}_2^{\bar{r}}, \bar{g}^{y_1} \bar{h}_3^{\bar{r}}, \ldots, g^{y_n} \bar{h}_{n+2}^{\bar{r}}\right) \in \mathbb{G}_2^{n+4}.$$

$\mathsf{Dec}(\mathsf{pp}, \mathsf{ct}, \mathsf{sk})$: Given $\mathsf{ct} = (C_1, \ldots, C_{n+4}) \in \mathbb{G}_1^{n+4}$ and $\mathsf{sk} = (S_1, \ldots, S_{n+4}) \in \mathbb{G}_2^{n+4}$, compute and output the discrete logarithm of $\prod_{i=1}^{n+4} e(C_i, S_i)$ to the base $e(g, \bar{g})$.

Figure 8: Weakly Function-Private IPE

Theorem 6. *If we apply the transformation given in Figure 2 to two functional encryptions in Figure 6 and 7, then the the resulting functional encryption is weakly function-private under the SXDH assumption.*

5.2. Full Function-Private IPE Construction

To apply our second transformation Trans2 for full function-privacy, we again use the scheme in Figure 6. However, contrary to the first transformation Trans1, it is not straightforward to apply Trans2 to this scheme. In contrast to Trans1, Trans2 in Figure 4 requires a functional encryption FE2 for special functionality containing a *branching statement*, as in Figure 5. To the best of our knowledge, there is no efficient construction for message-private FE supporting such the functionality under reasonable assumptions, except for indistinguishability obfuscation or multilinear maps. Therefore, it is not trivial to find an efficient construction for FE2 in Trans2.

Removing Branching Statement. Although it is hard to find an efficient FE for such the functionality containing branching statement in general, we show that if we can assume that the range of Dec1 has a group structure and $\mathsf{Dec1}(\cdot, \perp)$ is equal to the identity of the group, then we can remove such the branching statement.

$\mathsf{Setup}(1^\lambda, 1^n)$: Choose an asymmetric bilinear group $\mathcal{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, e)$ of prime order $q > 2^\lambda$ with generators $g, h \xleftarrow{\$} \mathbb{G}_1$ and $\bar{g}, \bar{h} \xleftarrow{\$} \mathbb{G}_2$. Choose $\boldsymbol{s} = (s_1, \ldots, s_n), \hat{\boldsymbol{s}} = (\hat{s}_1, \ldots, \hat{s}_n), \boldsymbol{t} = (t_1, \ldots, t_n), \hat{\boldsymbol{t}} = (\hat{t}_1, \ldots, \hat{t}_n) \xleftarrow{\$} \mathbb{Z}_q^n$ and $\boldsymbol{u} = (u_1, \ldots, u_{n+2}), \boldsymbol{v} = (v_1, \ldots, v_{n+2}) \xleftarrow{\$} \mathbb{Z}_q^{n+2}$. Compute $h_i = g^{s_i} h^{t_i}, \hat{h}_i = g^{\hat{s}_i} h^{\hat{t}_i}$ for $i = 1, \ldots, n$ and $\bar{h}_i = \bar{g}^{u_i} \bar{h}^{v_i}$ for $i = 1, \ldots, n+2$. Output $\mathsf{mk} := \big(\{h_i, \hat{h}_i, s_i, \hat{s}_i, t_i, \hat{t}_i\}_{i=1}^n, \{\bar{h}_i, u_i, v_i\}_{i=1}^{n+2}\big)$ and $\mathsf{pp} := \big(\mathcal{G}, g, h, \bar{g}, \bar{h}\big)$.

$\mathsf{Enc}(\mathsf{mk}, \boldsymbol{x})$: Given a vector $\boldsymbol{x} = (x_1, \ldots, x_n) \in \mathbb{Z}_q^n$, choose $r, \hat{r} \xleftarrow{\$} \mathbb{Z}_q$ and compute

$$\mathsf{ct1}_{\boldsymbol{x}} := \big(g^r, h^r, g^{x_1} h_1^r, \ldots, g^{x_n} h_n^r\big) \in \mathbb{G}_1^{n+2},$$

$$\widehat{\mathsf{ct1}}_{\boldsymbol{x}} := \big(g^{\hat{r}}, h^{\hat{r}}, g^{x_1} (\hat{h}_1)^{\hat{r}}, \ldots, g^{x_n} (\hat{h}_n)^{\hat{r}}\big) \in \mathbb{G}_1^{n+2}.$$

Compute $\boldsymbol{C}_{\boldsymbol{x}} := \big(\mathsf{ct1}_{\boldsymbol{x}}^{-\boldsymbol{u}}, \mathsf{ct1}_{\boldsymbol{x}}^{-\boldsymbol{v}}, \mathsf{ct1}_{\boldsymbol{x}}\big)$, $\widehat{\boldsymbol{C}}_{\boldsymbol{x}} := \big(\widehat{\mathsf{ct1}}_{\boldsymbol{x}}^{-\boldsymbol{u}}, \widehat{\mathsf{ct1}}_{\boldsymbol{x}}^{-\boldsymbol{v}}, \widehat{\mathsf{ct1}}_{\boldsymbol{x}}\big) \in \mathbb{G}_1^{n+4}$. Output $\mathsf{ct}_{\boldsymbol{x}} := \big(\boldsymbol{C}_{\boldsymbol{x}}, \widehat{\boldsymbol{C}}_{\boldsymbol{x}}\big) \in \mathbb{G}_1^{2(n+4)}$.

$\mathsf{KG}(\mathsf{mk}, \boldsymbol{y})$: Given a vector $\boldsymbol{y} = (y_1, \ldots, y_n) \in \mathbb{Z}_q^n$, choose $\bar{r}, \bar{r}' \xleftarrow{\$} \mathbb{Z}_q$ and compute

$$\boldsymbol{S}_{\boldsymbol{y}} := \big(\bar{g}^{\bar{r}}, \bar{h}^{\bar{r}}, \bar{g}^{-\langle \boldsymbol{s}, \boldsymbol{y} \rangle} \bar{h}_1^{\bar{r}}, \bar{g}^{-\langle \boldsymbol{t}, \boldsymbol{y} \rangle} \bar{h}_2^{\bar{r}}, \bar{g}^{y_1} \bar{h}_3^{\bar{r}}, \ldots, g^{y_n} \bar{h}_{n+2}^{\bar{r}}\big) \in \mathbb{G}_2^{n+4},$$

$$\boldsymbol{S}'_{\boldsymbol{y}} := \big(\bar{g}^{\bar{r}'}, \bar{h}^{\bar{r}'}, \bar{h}_1^{\bar{r}'}, \ldots, \bar{h}_{n+2}^{\bar{r}'})\big) \in \mathbb{G}_2^{n+4},$$

Output $\mathsf{sk}_{\boldsymbol{y}} := \big(\boldsymbol{S}_{\boldsymbol{y}}, \boldsymbol{S}'_{\boldsymbol{y}}\big) \in \mathbb{G}_2^{2(n+4)}$.

$\mathsf{Dec}(\mathsf{pp}, \mathsf{ct}, \mathsf{sk})$: Given $\mathsf{ct} = (C_1, \ldots, C_{n+4}, \widehat{C}_1, \ldots, \widehat{C}_{n+4}) \in \mathbb{G}_1^{2(n+4)}$ and $\mathsf{sk} = (S_1, \ldots, S_{n+4}, S'_1, \ldots, S'_{n+4}) \in \mathbb{G}_2^{2(n+4)}$, compute and output the discrete logarithm of $\prod_{i=1}^{n+4} e(C_i, S_i) \cdot e(\widehat{C}_i, S'_i)$ to the base $e(g, \bar{g})$.

Figure 9: Fully Function-Private Functional Encryption for Inner-Product

We assume that the decryption algorithm $\mathsf{Dec1}$ in Figure 4 satisfies the following conditions

1. Let $\mathbb{G}_t$ be a (multiplicative) group. For any valid ciphertext $\mathsf{ct1}$ and any key k, $\mathsf{Dec1}(\mathsf{ct1}, k) \in \mathbb{G}_t$.
2. $\mathsf{Dec1}(\mathsf{ct1}, \perp) = 1_t$, where 1_t is the identity of $\mathbb{G}_t$.

Then, the above conditions immediately imply that for the function in Figure 5, if $(k, \widehat{k}) \neq (\perp, \perp)$, then

$$F_{\mathsf{ct1}, \widehat{\mathsf{ct1}}}(k, \widehat{k}) = \mathsf{Dec1}(\mathsf{ct1}, k) \cdot \mathsf{Dec1}(\widehat{\mathsf{ct1}}, \widehat{k}), \tag{4}$$

where $\cdot$ is a group operation in $\mathbb{G}_t$. However, we note that although we have defined the case $(k, \widehat{k}) \neq (\perp, \perp)$ for completeness, this case is not used in the transformation nor the proof of Theorem 3. Therefore, Equation (4) is sufficient for the definition of $F_{\mathsf{ct1}, \widehat{\mathsf{ct1}}}$ under the above two conditions.

We note that the above conditions do not affect the proof of Theorem 3 since these conditions are only about the output format of $\mathsf{Dec1}$ algorithm and $\mathsf{Dec1}(\mathsf{ct1}, \perp)$ is not used in both the scheme and the proof at all. Therefore, if there is a message-private FE scheme satisfying the above two conditions and it is used as $\mathsf{FE1}$ in Figure 4, then we can still apply Theorem 3, so that the resulting scheme becomes fully function-private.

Construction for $\mathsf{FE2}$. We can slightly modify ALS-IPE to satisfy the above conditions; redefine Dec algorithm for a special input $\mathsf{sk} = \perp$ to output the identity of $\mathbb{G}_t$. An easy way to implement this modification is to define $\perp$ as a zero vector in $\mathbb{Z}_q^{n+2}$.[11] We now consider the function $F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}(k, \widehat{k})$ when we use ALS-IPE as $\mathsf{FE1}$ in our full function-privacy transformation. We have

$$\begin{aligned} F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}(k, \widehat{k}) &= \mathsf{Dec1}(\mathsf{ct1}, k) \cdot \mathsf{Dec1}(\widehat{\mathsf{ct1}}, \widehat{k}) \\ &= e\big(\mathsf{ct1}^k, \bar{g}\big) \cdot e\big(\widehat{\mathsf{ct1}}^{\widehat{k}}, \bar{g}\big) \\ &= e\big(\mathsf{ct1}^k \cdot \widehat{\mathsf{ct1}}^{\widehat{k}}, \bar{g}\big) \\ &= e\big((\mathsf{ct1}, \widehat{\mathsf{ct1}})^{(k,\widehat{k})}, \bar{g}\big). \end{aligned}$$

Note that either k or $\widehat{k}$ is $\perp$, but we represent $\perp$ by a zero vector in $\mathbb{Z}_q^{n+2}$, so that multi-exponentiation using $\perp$ is well defined. The function $F_{\mathsf{ct1},\widehat{\mathsf{ct1}}}(k, \widehat{k})$ eventually becomes a multi-exponentiation with bilinear map; mapping from $\big((\mathsf{ct1}, \widehat{\mathsf{ct1}}), (k, \widehat{k})\big)$ to $e\big((\mathsf{ct1}, \widehat{\mathsf{ct1}})^{(k,\widehat{k})}, \bar{g}\big)$. Therefore, similar to weak function-privacy transformation, we can use the FE scheme for multi-exponentiation with bilinear map (Figure 7) as $\mathsf{FE2}$ of full function-privacy transformation in Figure 4.

Fully Function-Private IPE. We give the full description of the transformed scheme in Figure 9. The full function privacy of the scheme in Figure 9 is directly obtained by Theorem 3, Theorem 4, and Theorem 5. We give the resulting theorem as follows.

Theorem 7. *If we apply the transformation given in Figure 4 to two functional encryptions in Figure 6 and 7, then the the resulting functional encryption is fully function-private under the SXDH assumption.*

References

[1] A. Shamir, Identity-based cryptosystems and signature schemes, in: CRYPTO 1984, LNCS, Springer, 1984, pp. 47–53.

[11] In fact, a zero vector in $\mathbb{Z}_q^{n+2}$ is reserved for the decryption key of a zero vector $\mathbb{Z}_q^n$. We note that this collision does not make any contradictions in both the scheme and the proof. Indeed, the proof of Theorem 3 does not care the output of $\mathsf{Dec1}(\mathsf{ct1}, \perp)$, in particular, if $\mathsf{Dec1}(\mathsf{ct1}, \perp) = 1_t$.

[2] D. Boneh, M. Franklin, Identity-based encryption from the weil pairing, SIAM J. Comput. 32 (3) (2003) 586–615.

[3] C. Cocks, An identity based encryption scheme based on quadratic residues, in: IMA Int. Conf., 2001, pp. 360–363.

[4] A. Sahai, B. Waters, Fuzzy identity-based encryption, in: EUROCRYPT 2005, Vol. 3494 of LNCS, Springer, 2005, pp. 457–473.

[5] V. Goyal, O. Pandey, A. Sahai, B. Waters, Attribute-based encryption for fine-grained access control of encrypted data, in: ACM CCS 2006, ACM, 2006, pp. 89–98.

[6] A. B. Lewko, T. Okamoto, A. Sahai, K. Takashima, B. Waters, Fully secure functional encryption: Attritbute-based encryption and (hierarchical) inner product encryption, in: EUROCRYPT 2010, Vol. 6110 of LNCS, Springer, 2010, pp. 62–91.

[7] J. Katz, A. Sahai, B. Waters, Predicate encryption supporting disjunctions, polynomial equations, and inner products, Journal of Cryptology 26 (2) (2013) 191–224.

[8] T. Okamoto, K. Takashima, Fully secure functional encryption with general relations from the decisional linear assumption, in: CRYPTO 2010, Vol. 6223 of LNCS, Springer, 2010, pp. 191–208.

[9] D. Boneh, A. Sahai, B. Waters, Functional encryption: Definitions and challenges, in: TCC 2011, Vol. 6597 of LNCS, Springer, 2011, pp. 253–273.

[10] A. O'Neill, Definitional issues in functional encryption, IACR Cryptology ePrint Archive (http://eprint.iacr.org/2010/556) (2010) 556.

[11] S. Gorbunov, V. Vaikuntanathan, H. Wee, Functional encryption with bounded collusions via multi-party computation, in: CRYPTO 2012, Vol. 7417 of LNCS, Springer, 2012, pp. 162–179.

[12] S. Agrawal, S. Gorbunov, V. Vaikuntanathan, H. Wee, Functional encryption: new perspective and lower bounds, in: CRYPTO (2) 2013, Vol. 8043 of LNCS, Springer, 2013, pp. 500–518.

[13] M. Bellare, A. O'Neill, Semantically-secure functional encryption: possibility results, impossibility results and the quest for a general definition, in: CANS 2013, Vol. 8257 of LNCS, Springer, 2013, pp. 218–234.

[14] E. Boyle, K. Chung, R. Pass, On extractability obfuscation, in: TCC 2014, Vol. 8349 of LNCS, Springer, 2014, pp. 52–73.

[15] A. D. Caro, V. Iovino, A. Jain, A. O'Neill, O. Paneth, G. Persiano, On the achievability of simulation-based security for functional encryption, in: CRYPTO (2) 2013, Vol. 8043 of LNCS, Springer, 2013, pp. 519–535.

[16] S. Garg, C. Gentry, M. Raykova, A. Sahai, B. Waters, Candidate indistinguishability obfuscation and functional encryption for all circuits, in: FOCS 2013, IEEE Computer Society, 2013, pp. 4–49.

[17] S. Goldwasser, Y. T. Kalai, V. Vaikuntanathan, N. Zeldovich, Reusable garbled circuits and succinct functional encryption, in: STOC 2013, ACM, 2013, pp. 555–564.

[18] S. Garg, C. Gentry, S. Halevi, M. Zhandry, Fully secure functional encryption without obfuscation, in: TCC 2016, Vol. 9563 of LNCS, Springer, 2016, pp. 480–511.

[19] E. Shen, E. Shi, B. Waters, Predicate privacy in encryption systems, in: TCC 2009, Vol. 5444 of LNCS, Springer, 2009, pp. 457–473.

[20] Z. Brakerski, G. Segev, Function-private functional encryption in the private-key setting, in: TCC(2) 2015, Vol. 9015 of LNCS, Springer, 2015, pp. 306–324.

[21] A. Bishop, A. Jain, L. Kowalczyk, Function-hiding inner-product encryption, in: ASIACRYPT(1) 2015, Vol. 9452 of LNCS, Springer, 2015, pp. 470–491.

[22] P. Datta, R. Dutta, S. Mukhopadhyay, Functional encryption for inner-product with full function privacy, in: PKC(1) 2016, Vol. 9614 of LNCS, Springer, 2016, pp. 164–195.

[23] S. Kim, K. Lewi, A. Mandal, H. W. Montgomery, A. Roy, D. J. Wu, Function-hiding inner product encryption is practical, in: IACR Cryptology ePrint Archive 2016:440, 2016.

[24] D. Boneh, A. Raghunathan, G. Segev, Function-private identity-based encryption: hiding the function in functional encryption, in: CRYPTO (2) 2013, Vol. 8043 of LNCS, Springer, 2013, pp. 461–478.

[25] D. Boneh, A. Raghunathan, G. Segev, Function-private subspace-membership encryption and its applications, in: ASIACRYPT (1) 2013, Vol. 8269 of LNCS, Springer, 2013, pp. 255–275.

[26] S. Agrawal, S. Agrawal, S. Badrinarayanan, A. Kumarasubramanian, M. Prabhakaran, A. Sahai, On the practical security of inner product functional encryption, in: PKC 2015, Vol. 9020 of LNCS, Springer, 2015, pp. 777–798.

[27] M. Abdalla, F. Bourse, A. D. Caro, D. Pointcheval, Simple functional encryption schemes for inner products, in: PKC 2015, Vol. 9020 of LNCS, Springer, 2015, pp. 733–751.

[28] S. Agrawal, B. Libert, D. Stehlé, Fully secure functional encryption for inner products, from standard assumptions, in: CRYPTO 2016, Vol. 9816 of LNCS, Springer, 2016, pp. 333–362.

[29] M. Abdalla, F. Bourse, A. D. Caro, D. Pointcheval, Better security for functional encryption for inner product evaluations, in: IACR Cryptology ePrint Archive 2016:011, 2016.

[30] T. Okamoto, K. Takashima, Homomorphic encryption and signatures from vector decomposition, in: Pairing 2008, Vol. 5209 of LNCS, Springer, 2008, pp. 57–74.

[31] T. Okamoto, K. Takashima, Hierarchical predicate encryption for inner-products, in: ASIACRYPT 2009, Vol. 5912 of LNCS, Springer, 2009, pp. 214–231.

[32] D. Boneh, X. Boyen, E. Goh, Hierarchical identity based encryption with constant size ciphertexts, in: IACR Cryptology ePrint Archive 2005:015, 2005.

[33] V. Shoup, Lower bounds for discrete logarithms and related problems, in: EUROCRYPT 1997, Springer, 1997, pp. 256–266.

[34] J. Tomida, M. Abe, T. Okamoto, Efficient functional encryption for inner-product values with full-hiding security, in: Information Security 2016, Vol. 9866 of LNCS, Springer, 2016, pp. 408–425.

[35] H. Lin, V. Vaikuntanathan, Indistinguishability obfuscation from ddh-like assumptions on constant-degree graded encodings, in: FOCS 2016, IEEE Computer Society, 2016, pp. 11–20.

[36] M. Naor, M. Yung, public-key cryptosystems provably secure against chosen ciphertext attacks, in: STOC 1990, ACM, 1990, pp. 427–437.

[37] S. Goldwasser, Y. Kalai, R. A. Popa, V, N. Zeldovich, Resuable garbled circuits and succinct functional encryption, in: STOC 2013, ACM, 2013, pp. 555–564.

Appendix A. Definitions of Simulation-based Security

In the below definitions of SIM-based security we use the the notation $\mathcal{A}^{B(\mathcal{O}(\cdot;\tau),\sigma)}$. It means that $\mathcal{A}$ can issue a query q to B, which forwards a query q to a stateful oracle $\mathcal{O}$. $\mathcal{O}(q;\tau)$ is then executed and outputs a pair (q',τ'), where q' is sent to B and τ is updated to τ' for the next invocation. $B(q',\sigma)$ is then executed and outputs (r_q,σ'). Finally B sends r_q to $\mathcal{A}$ as the response to the query q, and updates σ to σ' for the next invocation.

Definition 7 ($\mathsf{SIM}^{\mathsf{MP}}_{\mathsf{non}}$- and $\mathsf{SIM}^{\mathsf{MP}}_{\mathsf{ad}}$- Security for Private-Key FE)**.** *Let* FE *be a private-key functional encryption for function space* $\mathcal{F}$ *over a message space* $\mathcal{M}$*. For every probabilistic polynomial-time adversary* $\mathcal{A}$ *and a probabilistic polynomial-time simulator* $\mathcal{S} = (\mathcal{S}.\mathsf{setup}, \mathcal{S}.\mathsf{kg}, \mathcal{S}.\mathsf{enc})$*, consider the following two experiments:*

$\mathsf{Real}_{\mathcal{A}}(1^\lambda)$:

1. $(\mathsf{pp}, \mathsf{mk}) \leftarrow \mathsf{Setup}(1^\lambda)$
2. $\alpha \leftarrow \mathcal{A}^{\mathsf{KG}(\mathsf{mk},\cdot),\mathsf{Enc}(\mathsf{mk},\cdot)}(1^\lambda, \mathsf{pp})$
   Let $\{f_{i'} : i' \in [i]\}$ *be queries to* KG *made by* $\mathcal{A}$
   Let $\{\boldsymbol{x}_{j'} : j' \in [j]\}$ *be queries to* Enc *made by* $\mathcal{A}$
3. *Output* $\{\alpha, f_{i'}, \boldsymbol{x}_{j'} : i' \in [i], j' \in [j]\}$

$\mathsf{Ideal}_{\mathcal{A},\mathcal{S}}(1^\lambda)$:

1. $(\mathsf{pp}, \sigma) \leftarrow \mathcal{S}.\mathsf{setup}(1^\lambda)$
2. $\alpha \leftarrow \mathcal{A}^{\mathcal{S}.\mathsf{kg}(\mathcal{O}^{\mathsf{MP}}(\cdot;\tau);\sigma),\mathcal{S}.\mathsf{enc}(\mathcal{O}^{\mathsf{MP}}(\cdot;\tau);\sigma)}(1^\lambda, \mathsf{pp})$
   Let $\{f_{i'} : i' \in [i]\}$ *be queries to* $\mathcal{S}.\mathsf{kg}$ *made by* $\mathcal{A}$
   Let $\{\boldsymbol{x}_{j'} : j' \in [j]\}$ *be queries to* $\mathcal{S}.\mathsf{enc}$ *made by* $\mathcal{A}$
3. *Output* $\{\alpha, f_{i'}, \boldsymbol{x}_{j'} : i' \in [i], j' \in [j]\}$

1. *In the adaptive query case, the stateful algorithms* $\mathcal{S}.\mathsf{kg}$ *and* $\mathcal{S}.\mathsf{enc}$ *simulate the ideal key generation and encryption, respectively. They share the state* σ*, which is initialized by* $\mathcal{S}_1$ *at the beginning of the game. They work as follows:*

   - *Given the decryption key query of* $f_{i+1} \in \mathcal{F}$ *made by* $\mathcal{A}$ *(in adaptive manner),* $\mathcal{S}.\mathsf{kg}$ *forwards it to* $\mathcal{O}^{\mathsf{MP}}$ *to obtain its inputs* $|f_{i+1}|$ *and* $\{(f_{i+1}, j', f_{i+1}(\boldsymbol{x}_{j'})) : j' \in [j]\}$*,* $\boldsymbol{x}_{j'}$*'s are queries of messages until then. It outputs* $(\mathsf{sk}_{f_{i+1}}, \sigma')$*. It updated* σ *by adding a pair* $(f_{i+1}, \mathsf{sk}_{f_{i+1}})$ *into* σ*.*
   - *Given the ciphertext query of* $\boldsymbol{x}_{j+1} \in \mathcal{M}$ *made by* $\mathcal{A}$ *(in adaptive manner),* $\mathcal{S}.\mathsf{enc}$ *forwards it to* $\mathcal{O}^{\mathsf{MP}}$ *to obtain its inputs* $|\boldsymbol{x}_{j+1}|$ *and* $\{(f_{i'}, j+1, f_{i'}(\boldsymbol{x}_{j+1})) : i' \in [i]\}$*, where* $f_{i'}$*'s are queries of functions until then. It outputs* $(\mathsf{ct}_{\boldsymbol{x}_{j+1}}, \sigma')$*.*

   The private-key functional encryption FE *is then said to be satisfied with the simulation-based message privacy against adaptive query attack (*$\mathsf{SIM}^{\mathsf{MP}}_{\mathsf{ad}}$ *- secure, for short) if two distributions above are computationally indistinguishable.*

2. *In the non-key query case, the oracle* KG *in the real game and* $\mathcal{S}.\mathsf{kg}$ *in the ideal game are both empty oracles. Thus two games above simply mean SIM-security of the encryption algorithm of* FE*. That is, given the ciphertext query of* $\boldsymbol{x}_{j+1} \in \mathcal{M}$ *made by* $\mathcal{A}$*,* $\mathcal{S}.\mathsf{enc}$ *obtains its input* $|\boldsymbol{x}_{j+1}|$ *from* $\mathcal{O}^{\mathsf{MP}}$ *and outputs* $(\mathsf{ct}_{\boldsymbol{x}_{j+1}}, \sigma')$*. The private-key functional encryption* FE *is then said to be satisfied with the simulation-based message privacy against non-key query attack (*$\mathsf{SIM}^{\mathsf{MP}}_{\mathsf{non}}$ *- secure, for short) if two distributions above are computationally indistinguishable.*

Definition 8 ($\mathsf{SIM}^{\mathsf{FFP}}_{\mathsf{ad}}$- Security for Private-Key FE)**.** *Let* FE *be a private-key functional encryption for function space* $\mathcal{F}$ *over a message space* $\mathcal{M}$*. For every probabilistic polynomial-time adversary* $\mathcal{A}$ *and a probabilistic polynomial-time simulator* $\mathcal{S} = (\mathcal{S}.\mathsf{setup}, \mathcal{S}.\mathsf{kg}, \mathcal{S}.\mathsf{enc})$*, consider the following two experiments:*

$\mathsf{Real}_{\mathcal{A}}(1^\lambda)$:

1. $(\mathsf{pp}, \mathsf{mk}) \leftarrow \mathsf{Setup}(1^\lambda)$
2. $\alpha \leftarrow \mathcal{A}^{\mathsf{KG}(\mathsf{mk},\cdot),\mathsf{Enc}(\mathsf{mk},\cdot)}(1^\lambda, \mathsf{pp})$
   Let $\{f_{i'} : i' \in [i]\}$ *be queries to* KG *made by* $\mathcal{A}$
   Let $\{\boldsymbol{x}_{j'} : j' \in [j]\}$ *be queries to* Enc *made by* $\mathcal{A}$
3. *Output* $\{\alpha, f_{i'}, \boldsymbol{x}_{j'} : i' \in [i], j' \in [j]\}$

$\mathsf{Ideal}_{\mathcal{A},\mathcal{S}}(1^\lambda)$:

1. $(\mathsf{pp}, \sigma) \leftarrow \mathcal{S}.\mathsf{setup}(1^\lambda)$
2. $\alpha \leftarrow \mathcal{A}^{\mathcal{S}.\mathsf{kg}(\mathcal{O}^{\mathsf{FFP}}(\cdot;\tau);\sigma),\mathcal{S}.\mathsf{enc}(\mathcal{O}^{\mathsf{FFP}}(\cdot;\tau);\sigma)}(1^\lambda, \mathsf{pp})$
   Let $\{f_{i'} : i' \in [i]\}$ *be queries to* $\mathcal{S}.\mathsf{kg}$ *made by* $\mathcal{A}$
   Let $\{\boldsymbol{x}_{j'} : j' \in [j]\}$ *be queries to* $\mathcal{S}.\mathsf{enc}$ *made by* $\mathcal{A}$
3. *Output* $\{\alpha, f_{i'}, \boldsymbol{x}_{j'} : i' \in [i], j' \in [j]\}$

The stateful algorithms $\mathcal{S}.\mathsf{kg}$ *and* $\mathcal{S}.\mathsf{enc}$ *share the state* σ*, which is initialized by* $\mathcal{S}_1$ *at the beginning of the game, and simulate the ideal key generation and encryption, respectively. They work as follows:*

- *Given the decryption key query of* $f_{i+1} \in \mathcal{F}$ *made by* $\mathcal{A}$ *(in adaptive manner),* $\mathcal{S}.\mathsf{kg}$ *forwards it to* $\mathcal{O}^{\mathsf{FFP}}$ *to obtain its inputs* $|f_{i+1}|$ *and* $\{(i+1, j', f_{i+1}(\boldsymbol{x}_{j'})) : j' \in [j]\}$*,* $\boldsymbol{x}_{j'}$*'s are queries of messages until then. It outputs* $(\mathsf{sk}_{f_{i+1}}, \sigma')$*.*
- *Given the ciphertext query of* $\boldsymbol{x}_{j+1} \in \mathcal{M}$ *made by* $\mathcal{A}$ *(in adaptive manner),* $\mathcal{S}.\mathsf{enc}$ *forwards it to* $\mathcal{O}^{\mathsf{FFP}}$ *to obtain its inputs* $|\boldsymbol{x}_{i+1}|$ *and* $\{(i', j+1, f_{i'}(\boldsymbol{x}_{j+1})) : i' \in [i]\}$*, where* $f_{i'}$*'s are queries of functions until then. It outputs* $(\mathsf{ct}_{\boldsymbol{x}_{j+1}}, \sigma')$*.*

The private-key functional encryption FE *is then said to be satisfied with the simulation-based full function privacy against adaptive query attack (*$\mathsf{SIM}^{\mathsf{FFP}}_{\mathsf{ad}}$*-secure, for short) if two distributions above are computationally indistinguishable.*

Appendix B. Missing Proof

Appendix B.1. Proof for Theorem 2

Let $\mathcal{S}_1 = (\mathcal{S}_1.\mathsf{setup}, \mathcal{S}_1.\mathsf{kg}, \mathcal{S}_1.\mathsf{enc})$ be the $\mathsf{SIM}^{\mathsf{MP}}_{\mathsf{non}}$- simulator for FE1 which accesses to the oracle $\mathcal{O}_{\mathsf{MP}_1}$, and $\mathcal{S}_2 = (\mathcal{S}_2.\mathsf{setup}, \mathcal{S}_2.\mathsf{kg}, \mathcal{S}_2.\mathsf{enc})$ be the $\mathsf{SIM}^{\mathsf{MP}}_{\mathsf{ad}}$-simulator for FE2 which accesses to the oracle $\mathcal{O}_{\mathsf{MP}_2}$. Given $\mathcal{S}_1$ and $\mathcal{S}_2$, we construct the $\mathsf{SIM}^{\mathsf{FFP}}_{\mathsf{ad}}$- simulator $\mathcal{S} = (\mathcal{S}.\mathsf{setup}, \mathcal{S}.\mathsf{kg}, \mathcal{S}.\mathsf{enc})$ for FE under the transformation in Figure 2. In the below we denote a decryption key for $f_{i'}$ and a ciphertext for $\boldsymbol{x}_{j'}$ of FE1 by $\mathsf{sk1}_{i'}$ and $\mathsf{ct1}_{j'}$, respectively. $\mathcal{S}$ works as follows:

- $\mathcal{S}.\mathsf{setup}(1^\lambda)$ simply executes $(\mathsf{pp1}, \sigma 1) \leftarrow \mathcal{S}_1.\mathsf{setup}(1^\lambda)$ and $(\mathsf{pp2}, \sigma 2) \leftarrow \mathcal{S}_2.\mathsf{setup}(1^\lambda)$. It outputs (pp, σ), where $\mathsf{pp} := (\mathsf{pp1}, \mathsf{pp2})$ and $\sigma := (\sigma 1, \sigma 2)$.

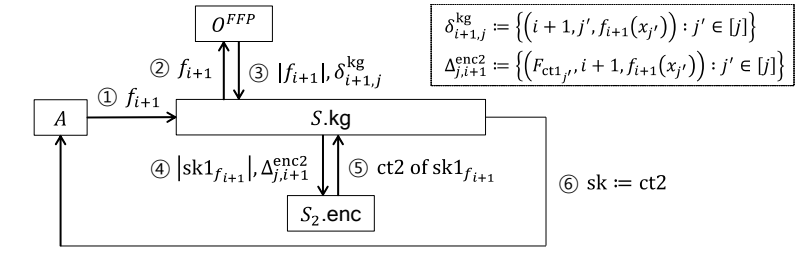

Figure B.10: The description of $\mathcal{S}.\mathsf{kg}$

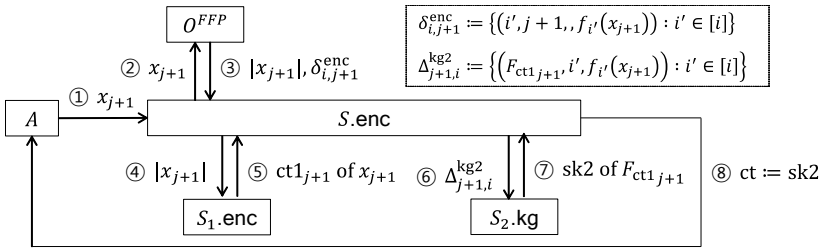

Figure B.11: The description of $\mathcal{S}.\mathsf{enc}$

- The description of $\mathcal{S}.\mathsf{kg}(\mathcal{O}^{\mathsf{FFP}}(\cdot;\tau);\sigma)$ is depicted in Figure B.10. Given the decryption key query of $f_{i+1} \in \mathcal{F}$ made by $\mathcal{A}$, $\mathcal{S}.\mathsf{kg}$ works as follows:

  1. $\mathcal{S}.\mathsf{kg}$ forwards it to $\mathcal{O}^{\mathsf{FFP}}$ to get $|f_{i+1}|$ and $\{(i+1, j', f_{i+1}(\boldsymbol{x}_{j'})) : j' \in [j]\}$.
  2. It invokes $\mathcal{S}_2.\mathsf{enc}$ on the behalf of $\mathcal{O}^{\mathsf{MP}_2}$ in order to obtain a ciphertext $\mathsf{ct2}$ of $\mathsf{FE2}$ on the message $\mathsf{sk1}_{f_{i+1}}$. It generates inputs $|\mathsf{sk1}_{f_{i+1}}| = |f_{i+1}|$ and $\{(F_{\mathsf{ct1}_{j'}}, i+1, f_{i+1}(\boldsymbol{x}_{j'})) : j' \in [j]\}$ for $\mathcal{S}_2.\mathsf{enc}$. $\mathcal{S}_2.\mathsf{enc}$ then outputs $(\mathsf{ct2}, \sigma 2')$. Note that since $\boldsymbol{x}'_j$ has been queried before, its $\mathsf{FE1}$ ciphertext $\mathsf{ct1}_{j'}$ is already known to $\mathcal{S}.\mathsf{kg}$, which has been obtained by $\mathcal{S}.\mathsf{enc}$ in the below.
  3. It provides $\mathsf{sk}_{f_{i+1}} := \mathsf{ct2}$ to $\mathcal{A}$ as the response of the query.

- The description of $\mathcal{S}.\mathsf{enc}(\mathcal{O}^{\mathsf{FFP}}(\cdot;\tau);\sigma)$ is depicted in Figure B.11. Given the encryption query of $\boldsymbol{x}_{j+1} \in \mathcal{M}$ made by $\mathcal{A}$, $\mathcal{S}.\mathsf{enc}$ works as follows:

  1. $\mathcal{S}.\mathsf{enc}$ forwards it to $\mathcal{O}^{\mathsf{FFP}}$ to get $|\boldsymbol{x}_{j+1}|$ and $\{(i', j+1, f_{i+1}(\boldsymbol{x}_{j'})) : i' \in [i]\}$.
  2. It invokes $\mathcal{S}_1.\mathsf{enc}$ on the behalf of $\mathcal{O}^{\mathsf{MP}_1}$ in order to obtain a ciphertext $\mathsf{ct1}_{j+1}$ of $\mathsf{FE1}$ on the message $\boldsymbol{x}_{j+1}$. It feeds an input $|\boldsymbol{x}_{j+1}|$ to $\mathcal{S}_1.\mathsf{enc}$, which outputs $(\mathsf{ct1}_{j+1}, \sigma 1')$.
  3. It then invokes $\mathcal{S}_2.\mathsf{kg}$ on the behalf of $\mathcal{O}^{\mathsf{MP}_2}$ in order to obtain a decryption key $\mathsf{sk2}$ of $\mathsf{FE2}$ on the function $F_{\mathsf{ct1}_{j+1}}$. It generates inputs

$|F_{\mathsf{ct1}}|$ and $\{(F_{\mathsf{ct1}_{j+1}}, i', f_{i'}(\boldsymbol{x}_{j+1})) : i' \in [i]\}$ for $\mathcal{S}_2.\mathsf{kg}$ by combining the given inputs of the previous steps. $\mathcal{S}_2.\mathsf{kg}$ then outputs $(\mathsf{sk2}, \sigma 2')$.

4. It provides $\mathsf{ct}_{\boldsymbol{x}_{j+1}} := \mathsf{sk2}$ to $\mathcal{A}$ as the response of the query.

We now claim that $\mathcal{S}.\mathsf{kg}$ and $\mathcal{S}.\mathsf{enc}$ properly simulate ideal decryption key generation and encryption, respectively. For $\mathcal{A}$'s key generation query f_{i+1}, the real FE key generation produces $\mathsf{sk}_{f_{i+1}} := \mathsf{ct2}_{\mathsf{sk1}_{f_{i+1}}} \leftarrow \mathsf{Enc2}(\mathsf{mk2}, \mathsf{sk1}_{f_{i+1}})$, where $\mathsf{sk1}_{f_{i+1}} := \mathsf{KG1}(\mathsf{mk1}, f_{i+1})$. $\mathcal{S}.\mathsf{kg}$ invokes $\mathcal{S}_2.\mathsf{enc}$ to obtain the simulated FE2 ciphertext $\mathsf{sk}'_{f_{i+1}} := \mathsf{ct2}'_{\mathsf{sk1}_{f_{i+1}}}$ for $\mathsf{sk1}_{f_{i+1}}$, which is computationally indistinguishable from $\mathsf{sk}_{f_{i+1}}$. By the correctness of $\mathcal{S}_2.\mathsf{enc}$ and $\mathsf{ct}_{\boldsymbol{x}_{j'}} = \mathsf{sk2}_{F_{\boldsymbol{x}_{j'}}}$, $\mathsf{Dec}(\mathsf{ct}_{\boldsymbol{x}_{j'}}, \mathsf{sk}_{f_{i+1}}) = \mathsf{Dec2}(\mathsf{ct2}_{\mathsf{sk1}_{f_{i+1}}}, \mathsf{sk2}_{F_{\boldsymbol{x}_{j'}}}) = f_{i+1}(\boldsymbol{x}_{j'}) = \mathsf{Dec2}(\mathsf{ct2}'_{\mathsf{sk1}_{f_{i+1}}}, \mathsf{sk2}_{F_{\boldsymbol{x}_{j'}}})$ $= \mathsf{Dec}(\mathsf{ct}_{\boldsymbol{x}_{j'}}, \mathsf{sk}'_{f_{i+1}})$, for $j' \in [j]$.

For $\mathcal{A}$'s encryption query $\boldsymbol{x}_{j+1}$, the real FE encryption produces $\mathsf{ct}_{\boldsymbol{x}_{j+1}} := \mathsf{sk2}_{F_{\mathsf{ct1}}} \leftarrow \mathsf{KG2}(\mathsf{mk2}, F_{\mathsf{ct1}})$, where F_{ct1} is obtained from $\mathsf{ct1} \leftarrow \mathsf{Enc1}(\mathsf{mk1}, \boldsymbol{x}_{j+1})$. $\mathcal{S}_2$ exploits $\mathcal{S}_1.\mathsf{enc}$ to obtain the simulated FE1 ciphertext $\mathsf{ct1}'$ for $\boldsymbol{x}_{j+1}$, which is computationally indistinguishable from $\mathsf{ct1}$. $\mathcal{S}.\mathsf{kg}$ then uses $\mathcal{S}_2.\mathsf{enc}$ to obtain $\mathsf{ct}'_{\boldsymbol{x}_{j+1}}$ from $F_{\mathsf{ct1}'}$, which is computationally indistinguishable from $\mathsf{ct}_{\boldsymbol{x}_{j+1}}$. By the correctness of $\mathcal{S}_1.\mathsf{enc}$ and $\mathcal{S}_2.\mathsf{kg}$, $\mathsf{Dec}(\mathsf{ct}_{\boldsymbol{x}_{j+1}}, \mathsf{sk}_{f_{i'}}) = \mathsf{Dec1}(\mathsf{ct1}, \mathsf{sk1}_{f_{i'}}) = f_{i'}(\boldsymbol{x}_{j+1}) = \mathsf{Dec1}(\mathsf{ct1}', \mathsf{sk1}_{f_{i'}}) = \mathsf{Dec}(\mathsf{ct}'_{\boldsymbol{x}_{j+1}}, \mathsf{sk}_{f_{i'}})$ for $i' \in [i]$. □

Appendix B.2. Proof Sketch for Theorem 5

The proof is essentially the same as that of ALS-FE scheme [28]. We use sequential games that begins with real game and ends with the game where the adversary has no advantage in distinguishing challenge ciphertext. We denote the event that the adversary wins in $\boldsymbol{Game}_i$ by S_i.

$\boldsymbol{Game}_0$. This is the real game. In the challenger phase, the adversary $\mathcal{A}$ chooses and sends challenge messages $\vec{w}_0, \vec{w}_1 \in \mathbb{Z}_q^n$ and obtains encryption of $\vec{w}_b = (w_{b,1}, \ldots, w_{b,n+2})$ from challenger $\mathcal{B}$. We observe that for $\vec{g} = (g_1, \ldots, g_{n+2}) \in \mathbb{G}_2^{n+2}$ that submitted to key generation oracle, the equation $\vec{e}(\bar{g}^{\vec{w}_0}, \vec{g}) = e(\bar{g}^{\vec{w}_1}, \vec{g})$ must hold in $\mathbb{G}_T$.

$\boldsymbol{Game}_1$ This game is the same with $\boldsymbol{Game}_0$ except that the challenger ciphertext $\mathsf{ct}_{\vec{w}_b} = (C, D, E_1, \ldots, E_n)$ is computed as follows:

$$C = \bar{g}^r \ , \ D = \bar{h}^r \text{ and } E_i = \bar{g}^{w_{b,i}} \cdot C^{u_i} \cdot D^{v_i}, \tag{B.1}$$

where $r \leftarrow \mathbb{Z}_q$. Then the challenge ciphertext $\mathsf{ct}_{\vec{w}_b}$ has the same distribution with $\boldsymbol{Game}_0$. Therefore, we have $\Pr[S_0] = \Pr[S_1]$.

$\boldsymbol{Game}_2$. This game is the same with $\boldsymbol{Game}_1$ except that the challenger ciphertext $\mathsf{ct}_{\vec{w}_b} = (C, D, E_1, \ldots, E_n)$ is computed as follows:

$$C = \bar{g}^r \ , \ D = \bar{h}^{r+r'} \text{ and } E_i = \bar{g}^{w_{b,i}} \cdot C^{u_i} \cdot D^{v_i}, \tag{B.2}$$

where $r, r' \leftarrow \mathbb{Z}_q$. We can easily verify that the distribution of $(C, D, E_1, \ldots, E_n)$ in equations (B.1) and (B.2) are computationally indistinguishable under DDH

assumption.

As in [28], we can show that the challenger ciphertext $\mathsf{ct}_{\vec{w}}$ in $\boldsymbol{Game}_2$ is perfectly hide the information on $b \in \{0,1\}$. Therefore, we have $|\Pr[S_0] - 1/2| = |\Pr[S_0] - \Pr[S_2]| = |\Pr[S_1] - \Pr[S_2]| \leq \mathbf{Adv}^{\mathsf{DDH}}_{\mathcal{B},\mathbb{G}_2}(\lambda)$. □