CCA-Secure Inner-Product Functional Encryption from Projective Hash Functions

Fabrice Benhamouda[1], Florian Bourse[2], and Helger Lipmaa[3]

[1] IBM Research, Yorktown Heights, NY, USA
[2] ENS, CNRS, INRIA, PSL Research University, Paris, France
[3] Institute of Computer Science, University of Tartu, Estonia

Abstract. In an inner-product functional encryption scheme, the plaintexts are vectors and the owner of the secret key can delegate the ability to compute weighted sums of the coefficients of the plaintext of any ciphertext. Recently, many inner-product functional encryption schemes were proposed. However, none of the known schemes are secure against chosen ciphertext attacks (IND-FE-CCA).
We present a generic construction of IND-FE-CCA inner-product functional encryption from projective hash functions with homomorphic properties. We show concrete instantiations based on the DCR assumption, the DDH assumption, and more generally, any Matrix DDH assumption.

Keywords: DCR, DDH, inner-product functional encryption, projective hash functions, CCA-security

1 Introduction

Traditionally, encryption has been an all-or-nothing affair: either a recipient owns the secret key (and thus can decrypt) or she does not. Functional encryption [32,21,28,10] enables a much more fine-grained handling of encrypted data. Here, the owner of the master key can delegate partial secret keys to various recipients. In a functional encryption scheme for functionality $\mathcal{F}$, the knowledge of a secret key corresponding to some y enables one to decrypt an encryption of z to $\mathcal{F}(y, z)$. As such, functional encryption has many potential applications, and has spurred a long line of research.

A functional encryption scheme can be required to satisfy several different security requirements [28,10]. In the case of the *adaptive* IND-FE-CPA security [28,10], it must be difficult for an adversary to distinguish functional ciphertexts of any two plaintexts z_0 and z_1. This must hold even if the adversary is given an oracle access to the partial secret key generator, where the secret key queries must satisfy the condition that $\mathcal{F}(y, z_0) = \mathcal{F}(y, z_1)$ for each queried y. In the weaker *selective security* model, the adversary is required to choose z_0 and z_1 before seeing the public key and answers to any of the secret key queries. See [28,10] for discussion.

Constructing adaptively IND-FE-CPA secure functional encryption for arbitrary functionalities has been an elusive goal, achieved only recently under strong assumptions like the existence of indistinguishability obfuscation or multilinear maps [19,11,33,20]. However, achieving functional encryption for restricted classes of functionalities is often easier. One of the simplest type of functional encryption schemes is inner-product functional encryption (IPFE).

Inner-Product Functional Encryption. In an inner-product functional encryption scheme, one encrypts a possibly long vector $\vec{z}$, and a recipient who has a partial secret key $k_{\vec{y}}$ can obtain the inner product $\langle \vec{y}, \vec{z} \rangle$ of $\vec{y}$ and $\vec{z}$. Recently, Abdalla, Bourse, De Caro, and Pointcheval [2] proposed the first IPFE schemes based on some of the most standard (and yet useful) cryptographic assumptions like the DDH and the LWE [31] assumptions. Unfortunately, their IPFE schemes are only selectively IND-FE-CPA secure. Subsequent work has reached better security notions while still relying on standard assumptions. In the secret key setting for example, function privacy has been achieved using bilinear maps [7,17], as well as a multi-input variant [4]. Adaptively IND-FE-CPA secure versions of the IPFE schemes of [2] were recently proposed by Agrawal, Libert, and Stehlé [5], together with a new scheme based on the DCR [29].

CCA Security. IND-CPA is a property every public-key encryption (PKE) scheme should have. It ensures that the plaintext is protected from any eavesdropping. However, it does not guarantee any security against active adversaries. The go-to security notion in this case is IND-CCA.[4] Informally, it states that a decryption oracle cannot help the adversary break the semantic security of the scheme, and it has been studied for years in the setting of PKE [30,12]. It has also been examined in the context of identity-based encryption [9,23] and attribute-based encryption [34], which are particular cases of functional encryption. It is thus natural to analyze it for inner-product functional encryption. In our setting of inner-product functional encryption, the decryption queries are as follows: the adversary chooses a ciphertext c and a vector $\vec{y}$ and gets back the decryption of c with $\mathrm{msk}_{\vec{y}}$, a freshly generated secret key for $\vec{y}$. Note that in this case, the decryption oracle is stronger than the partial key generation oracle because it doesn't have any requirement over its input $\vec{y}$, but on the other hand, the adversary doesn't get $\mathrm{msk}_{\vec{y}}$.

To the best of our knowledge, the only paper considering IND-FE-CCA security is [26]. In this paper, Nandi and Pandit construct IND-FE-CCA secure schemes from IND-FE-CPA secure ones with some properties that are verified by a lot of functional encryption schemes: key-policy or ciphertext-policy attribute-based encryption, and functional encryption for regular languages for example. However, this does not apply for inner-product functional encryption, so another technique is required.

In [27], Naor and Yung proposed a generic way of transforming an IND-CPA encryption scheme into an IND-CCA encryption scheme. While this transform

[4] In the current paper, CCA stands for CCA2.

could be adapted to functional encryption, it uses non-interactive zero-knowledge proofs, the constructions of which have strong requirements, such as bilinear groups or the random oracle model.

Our Contributions. In this paper, we propose a generic construction of IND-FE-CCA IPFE. This generic construction yields the first IND-FE-CCA IPFE schemes based on the DDH assumption, the DCR assumption, and any of the MDDH assumptions [18]. MDDH assumptions generalize the DDH assumption and might hold in settings where the DDH assumption cannot hold, as in symmetric bilinear groups.

Our generic construction is based on projective hash functions with homomorphic properties. Projective hash functions (PHFs) were introduced by Cramer and Shoup in [14], as a way to explain their efficient IND-CCA encryption scheme [12] and to extend it to other assumptions. Similarly to the generic IND-CCA encryption in [14], our IND-FE-CCA IPFE uses two PHFs and the second PHF enables to reject ciphertexts which are not well-formed.

If the second PHF is not used in the scheme, we get a generic IND-FE-CPA IPFE. We actually start by describing this generic IND-FE-CPA IPFE as a warm-up for our main contribution, a generic IND-FE-CCA IPFE.

Interestingly, when instantiated using the DDH assumption, this IND-FE-CPA scheme coincides exactly with the DDH-based IPFE of Agrawal et al. [5]. When instantiated using the DCR assumption, it corresponds to a variant of the DCR-based IPFE over $\mathbb{Z}$ of Agrawal et al. that has slightly worse parameters but avoids the use of discrete Gaussian distributions.

As a side contribution, we introduce a tag-based variant of functional encryption, where tags are associated to ciphertexts, together with a slightly weaker IND-TBFE-CCA (i.e., tag-based) security notion, in which the adversary is not allowed to query the decryption oracle with the tag of the challenge ciphertext. To simplify the description of our IND-FE-CCA IPFE scheme, we actually first construct an IND-TBE-CCA IPFE scheme. We then use an adapted version of the generic transformation from tag-based PKE to CCA secure PKE in [22]: the tag is the hash of a fresh verification key for a one-time signature scheme, used to sign the ciphertext. This one-time signature prevents malleability of the ciphertext.

Overview of our Constructions. Our constructions are inspired from the Cramer-Shoup encryption scheme [14]. A Cramer-Shoup ciphertext consists of three parts: a random word $\mathfrak{b}$ in some NP language (e.g., $\mathfrak{b}$ is a DDH tuple), the message masked by a hash of $\mathfrak{b}$ for a (smooth) PHF, and another hash of $\mathfrak{b}$ for a (2-universal) PHF. The hash value of any PHF can be computed both by someone knowing a witness for $\mathfrak{b}$ together with the public key (called projection key), and by someone knowing the secret key (called hashing key). The second hash value is used to reject ill-formed ciphertexts. Without it, the scheme is IND-CPA.

To build an IND-FE-CPA IPFE for vectors of dimension ℓ, we mask each coordinate of the message with a different hash value of the same word $\mathfrak{b}$. If the PHF is homomorphic, a linear combination of the corresponding hashing keys will allow for the decryption of the same linear combination of the coordinates, which is the inner product of the message and the coefficients of the linear combination. In order to reach IND-FE-CCA security and reject ill-formed ciphertexts, we add ℓ independent hash values of $\mathfrak{b}$ for ℓ independent 2-universal homomorphic PHF. We could not naively use only one such hash, because then anyone knowing the unique hashing key would be able to fake the last part of the ciphertext.

Road map. We first provide some general preliminaries and recall definitions related to PHFs and functional encryption in Sect. 2. In this section, we also define the concrete assumptions we are using: DDH, DCR, and MDDH. In Sect. 3, we formally define the properties of the PHF used in our generic IND-FE-CPA IPFE scheme, which is described in Sect. 4. We then move to the CCA setting. In Sect. 5, we define the properties of the second PHF used in our generic IND-FE-CCA IPFE scheme, which is described in Sect. 6.

2 Preliminaries

Let $\mathbb{Z}$ be the set of integers. If n is a positive integer, $\mathrm{spf}(n)$ is its smallest prime factor. If $S \subset \mathbb{Z}$ and $t \in \mathbb{Z}$, then let $S + t = \{s + t : s \in S\}$. If S is a finite set, then $|S|$ is its cardinal.

Let $\mathcal{R}$ be a commutative ring. We denote the set of d-dimensional column vectors over $\mathcal{R}$ by $\mathcal{R}^d$, the set of d-dimensional row vectors by $\mathcal{R}^{1\times d}$, and the set of $\ell \times d$ matrices by $\mathcal{R}^{\ell\times d}$. Unless explicitly said otherwise, each vector is a column vector. We denote vectors by using either boldface lower-case letters or lower-case letters with an arrow over it as in $\boldsymbol{b}$ and $\vec{b}$. We denote matrices by using boldface upper-case letters like in $\boldsymbol{A}$. We have two possible notations for vectors, as we sometimes need to consider vectors of vectors ($\vec{\boldsymbol{b}}$) and vectors of matrices ($\vec{\boldsymbol{A}}$). The ith coefficient of a vector $\boldsymbol{b}$ or $\vec{b}$ is denoted by b_i, while the ith coefficient of a vector of vectors $\vec{\boldsymbol{b}}$ is a vector and is denoted by $\boldsymbol{b}_i$. The jth coefficient of this latter vector is $b_{i,j}$. The same convention is used with coefficients of matrices and coefficients of vectors of matrices.

Within this paper, κ is the security parameter. A function $f(\kappa)$ is *negligible*, if for any polynomial p, $f(\kappa) = O(1/p(\kappa))$.

If $\mathcal{A}$ is a randomized algorithm, then we denote by $\mathcal{A}(x)$ the output distribution of $\mathcal{A}$ on input x. If S is a finite set, we denote by $U(S)$ the uniform distribution. If D is a distribution, we denote by $x \leftarrow_r D$ the assignment of a fresh sample from D to the variable x. If D is a distribution over some set S and if D is clear from context, $x \leftarrow_r D$ is also denoted by $x \leftarrow_r S$. If S is a finite set on which we did not explicitly defined any distribution, $x \leftarrow_r S$ stands for $x \leftarrow_r U(S)$.

Statistical and computational indistinguishability. Let $(A_\kappa)_\kappa$ and $(B_\kappa)_\kappa$ be two ensembles of distributions over some set Ω and indexed by the security parameter κ. In the sequel the security parameter is often omitted for the sake of simplicity. Let $\mathcal{A}$ be an algorithm, called an adversary. The advantage of $\mathcal{A}$ in distinguishing $(A_\kappa)_\kappa$ and $(B_\kappa)_\kappa$ is defined by $\mathsf{Adv}_{\mathcal{A}}(\kappa) = |\Pr_{x \leftarrow_r A_\kappa}[\mathcal{A}(x) = 1] - \Pr_{x \leftarrow_r B_\kappa}[\mathcal{A}(x) = 1]|$.

The distributions A and B are *computationally indistinguishable* if for any (probabilistic) polynomial time $\mathcal{A}$, its advantage $\mathsf{Adv}_{\mathcal{A}}(\kappa)$ is negligible. They are *statistically indistinguishable* if this is true for any (not necessarily polynomial-time) $\mathcal{A}$. The statistical distance $\mathrm{SD}(A, B)$ of distributions A and B is the supremum of the advantage of all adversaries in distinguishing them. Equivalently, if A and B are defined over a finite or countable set Ω,

$$\mathrm{SD}(A, B) = \frac{1}{2} \sum_{y \in \Omega} | \Pr_{x \leftarrow_r A}[x = y] - \Pr_{x \leftarrow_r B}[x = y]| \ . \tag{1}$$

We will often implicitly use the following lemmas.

Lemma 1. *Let S_1 and S_2 be two finite sets. If $S_1 \subseteq S_2$, we have $\mathrm{SD}(U(S_1), U(S_2)) = 1 - |S_1|/|S_2|$. In particular, if $|S_2| = (1 + 1/t) \cdot |S_1|$ for some positive integer t, then $\mathrm{SD}(U(S_1), U(S_2)) = 1/(t + 1)$.*

Proof. $\mathrm{SD}(U(S_1), U(S_2)) = \frac{1}{2}(|S_2 \setminus S_1|/|S_2| + |S_1| \cdot (1/|S_1| - 1/|S_2|)) = 1 - |S_1|/|S_2|$. □

Lemma 2. *Let $S \subseteq \mathbb{Z}$ be an interval and t be an integer. Then $\mathrm{SD}(U(S), U(S + t)) = |t|/|S|$.*

Proof. In the sum in Eq. (1), exactly $2|t|$ terms are non-zero: the ones corresponding to y in $(S \setminus (S + t)) \cup ((S + t) \setminus S)$. And these terms are equal to $1/|S|$. □

Abelian Groups. We extensively use Abelian groups. In particular, in our concrete instantiations, we use prime-order cyclic groups over an elliptic curve or subgroups of the (multiplicative) group $\mathbb{Z}_N^*$, for some positive integer N. We denote the elements of such groups by using the Fraktur script like in $\mathfrak{g}$ or $\mathfrak{b}$. By extension, even in our generic constructions and definitions, we also use this font to indicate values which, in our concrete instantiations, are group elements in such group $\mathbb{G}$ or vectors of such elements. However, we are also considering other Abelian groups (e.g., the group $\mathcal{K}$ of hashing keys of a key-homomorphic PHF in Def. 6) that are not related to cryptographic assumptions and for which group elements are not denoted using the Fraktur script.

Except if explicitly stated otherwise, we use *additive notation* for all our Abelian groups, even when this is not usual (as in the case of subgroups of $\mathbb{Z}_N^*$).

Let $\mathbb{G}$ be an Abelian group. We recall that if $\mathfrak{g}$ is a group element of order M, then we have a canonical monomorphism $w \in \mathbb{Z}_M \mapsto w \cdot \mathfrak{g} \in \mathbb{G}$. If $\mathbb{G}$ is a multiplicative group, this monomorphism corresponds to exponentiation. Hence, we denote the inverse of this monomorphism by $\log_{\mathfrak{g}}$. That is, if $\mathfrak{b} = w \cdot \mathfrak{g}$, then $\log_{\mathfrak{g}} \mathfrak{b} = w$.

Furthermore, let $\mathcal{R}$ be $\mathcal{R} = \mathbb{Z}$ or $\mathcal{R} = \mathbb{Z}_M$ with M being such that the order of any group element in $\mathbb{G}$ divides M. Then $\mathbb{G}$ can be seen as a $\mathcal{R}$-module. This means that for any $w \in \mathcal{R}$ and $\mathfrak{g} \in \mathbb{G}$, $w \cdot \mathfrak{g}$ is well defined. Importantly, by using additive notation, we can use the standard "matrix-vector" notation without prior explanation.

Basic Number Theory. Let N be a positive integer. Let $\varphi(N)$ be the Euler totient function. For any integer a and an odd prime q, the Legendre symbol $\left(\frac{a}{q}\right)$ is defined as $\left(\frac{a}{q}\right) := 0$, if $a \equiv 0 \pmod{q}$, $\left(\frac{a}{q}\right) := +1$, if $a \not\equiv 0 \pmod{q}$ and for some integer y, $a \equiv y^2 \pmod{q}$, and $\left(\frac{a}{q}\right) := -1$, if $a \not\equiv 0 \pmod{q}$ and there is no such y. For any integer a and any positive odd integer N, the Jacobi symbol is defined as the product of the Legendre symbols corresponding to the prime factors of N, $\left(\frac{a}{N}\right) := \prod_{i=1}^{t} \left(\frac{a}{p_i}\right)^{\alpha_i}$, where $N = \prod_{i=1}^{t} p_i^{\alpha_i}$ for distinct primes p_i. Let $J_N = \{a \in \mathbb{Z}_N : \left(\frac{a}{N}\right) = 1\}$; clearly J_N is a subgroup of $\mathbb{Z}_N^*$. The Jacobi symbol can be computed in polynomial time, given only a and N [25, Alg. 2.149].

2.1 Subset Membership Problems and Concrete Assumptions

Our framework uses *subset membership problems*, which were originally defined in [14]. Basically, a subset membership problem defines an NP language $\mathcal{L} \subset \mathcal{X}$, in which a random word in $\mathcal{L}$ is hard to distinguish from a random word in $\mathcal{X} \setminus \mathcal{L}$. In this paper, we consider a slight extension, where we instead require a random word in $\mathcal{L}$ to be hard to distinguish from a random word in a given set $\bar{\mathcal{L}} \subseteq \mathcal{X} \setminus \mathcal{L}$.

More formally, a subset membership problem $\mathbf{P}$ specifies an ensemble $(I_\kappa)_{\kappa \geq 0}$ of distributions. For every value of a security parameter $\kappa \geq 0$, I_κ is a probability distribution of instance descriptions. An instance description $\Lambda = \Lambda[\mathcal{X}, \mathcal{L}, \mathcal{W}, \varrho, \bar{\mathcal{L}}]$ specifies the following: (a) finite, non-empty sets $\mathcal{X}$, $\mathcal{L}$, $\mathcal{W}$, and $\bar{\mathcal{L}}$, such that $\mathcal{L}$ is a proper subset of $\mathcal{X}$ and $\bar{\mathcal{L}}$ is a non-empty subset of $\mathcal{X} \setminus \mathcal{L}$, (b) a binary relation $\varrho \subset \mathcal{X} \times \mathcal{W}$. For $\mathfrak{b} \in \mathcal{X}$ and $w \in \mathcal{W}$, we say that w is a witness for $\mathfrak{b}$ if $(\mathfrak{b}, w) \in \varrho$. We require that instance descriptions and elements of $\mathcal{X}$ and $\mathcal{W}$ can be uniquely encoded as bitstrings of length $\mathrm{poly}(\kappa)$.

A subset membership problem satisfies the following properties: (i) I_κ is efficiently samplable, which means that there exists a probabilistic polynomial time instance sampling algorithm that on input 1^κ samples an instance Λ according to the distribution I_κ; (ii) ϱ is efficiently samplable, which means that there exists a probabilistic polynomial time subset sampling algorithm that on input Λ outputs a random $\mathfrak{b} \in \mathcal{L}$ together with a witness $w \in \mathcal{W}$ for $\mathfrak{b}$; the distribution over ϱ implicitly defines a distribution over $\mathcal{L}$; (iii) $\bar{\mathcal{L}}$ is efficiently samplable; (iv) $\mathcal{X}$ is efficiently recognizable, which means that there exists a deterministic polynomial algorithm that on input (Λ, ζ) checks whether ζ is a valid binary encoding of an element of $\mathcal{X}$; (v) ϱ is efficiently recognizable; (vi) $(\mathcal{L}, \bar{\mathcal{L}})$*-indistinguishability:* a sample from $\mathcal{L}$ is computationally indistinguishable from a sample from $\bar{\mathcal{L}}$.

We do not require the distributions over ϱ, $\mathcal{L}$, and $\bar{\mathcal{L}}$ to be uniform. However, when we do not specify these distributions, we implicitly use the uniform distributions.

Let us now introduce the subset membership problems we use in our concrete instantiations. We name them according to the assumption under which we prove their $(\mathcal{L}, \bar{\mathcal{L}})$-indistinguishability property, namely DDH, MDDH, and DCR.

DDH-Based Subset Membership Problem. Let $\mathbb{G}$ be an additive cyclic group of prime order q, let $\mathcal{X} = \mathbb{G}^2$, let $\mathcal{L}$ be the subgroup of $\mathcal{X}$ generated by $\mathfrak{g} = (\mathfrak{g}_1, \mathfrak{g}_2)^\intercal \in \mathbb{G}^2$, where $\mathfrak{g}_i$ are random generators of $\mathbb{G}$, and let $\bar{\mathcal{L}} = \mathcal{X} \setminus \mathcal{L}$. A witness $w \in \mathcal{W} = \mathbb{Z}_q$ for $\mathfrak{b} \in \mathcal{L}$ is such that $\mathfrak{b} = w\mathfrak{g}$. In other words, we have $\mathcal{W} = \mathbb{Z}_q$ and $\varrho = \{(w \cdot \mathfrak{g}, w) : w \in \mathbb{Z}_q\}$. We set $\Lambda = (\mathbb{G}, \mathfrak{g})$.

This defines a subset membership problem, whose $(\mathcal{L}, \bar{\mathcal{L}})$-indistinguishability property is equivalent to the DDH assumption.

MDDH-Based Subset Membership Problem. For some interesting cryptographic cyclic groups, such as groups with a symmetric pairing, the DDH assumption does not hold. That is why weaker assumptions, such as the decisional linear assumption (DLIN, [8]), have been considered. More recently, Escala et al. introduced the *Matrix Diffie-Hellman* (*MDDH*) assumption family [18] that generalizes DDH and its weaker variants like DLIN. Let us recall the MDDH assumption families in the context of subset membership problems.

Let $\mathbb{G}$ be a cyclic group of prime order q. Let $\mathcal{D}$ be a distribution of matrices in $\mathbb{G}^{t \times d}$ with $d < t$ being two positive integers. Let $\mathfrak{g} \leftarrow_r \mathcal{D}$. Let $\mathcal{X} = \mathbb{G}^t$. Let $\mathcal{L}$ be the subgroup of $\mathcal{X}$ generated by the columns of $\mathfrak{g}$ and let $\bar{\mathcal{L}} = \mathcal{X} \setminus \mathcal{L}$. A witness $\boldsymbol{w} \in \mathcal{W} = \mathbb{Z}_q^d$ for $\mathfrak{b} \in \mathcal{L}$ is such that $\mathfrak{b} = \mathfrak{g} \cdot \boldsymbol{w}$. In other words, we have $\mathcal{W} = \mathbb{Z}_q^d$ and $\varrho = \{(\mathfrak{g} \cdot \boldsymbol{w}, \boldsymbol{w}) : \boldsymbol{w} \in \mathbb{Z}_q^d\}$. We set $\Lambda = (\mathbb{G}, \mathfrak{g})$.

This defines a subset membership problem, whose $(\mathcal{L}, \bar{\mathcal{L}})$-indistinguishability property corresponds to the $\mathcal{D}$-MDDH assumption.

When $d = 1$, $t = 2$, and $\mathcal{D}$ is the uniform distribution over vectors of two generators of $\mathbb{G}$, then we get back the DDH-based subset membership problem.

DCR-Based Subset Membership Problem. Let $N = pq$ be a product of two λ-bit random safe primes $p = 2p' + 1$ and $q = 2q' + 1$, where p' and q' are also primes and where λ is a function of the security parameter κ. Let $N' = p'q'$. Let $s \geq 1$. Write $\mathbb{Z}_{N^{s+1}}^* \cong G_{N^s} \oplus G_{N'} \oplus G_2 \oplus T$, where $\cong$ denotes group isomorphism, $\oplus$ is the direct sum or Cartesian product, G_i are cyclic groups of order i, and T is the order-2 cyclic group generated by $-1 \mod N^{s+1}$. Let $\mathbb{G} = \mathcal{X} = J_{N^{s+1}} \cong G_{N^s} \oplus G_{N'} \oplus T$. We recall that we use additive notation for $\mathbb{G}$. Let $\mathfrak{g}$ be a random generator of $\mathcal{L} \cong G_{N'}$, that is a subgroup of $\mathcal{X}$; $\mathfrak{g}$ can be thought of as a random $2N^s$-th residue. A witness $w \in \mathcal{W} = \mathbb{Z}$ for $\mathfrak{b} \in \mathcal{L}$ is such that $\mathfrak{b} = w \cdot \mathfrak{g}$. Finally, let $\mathfrak{g}_\perp$ be an arbitrary generator of the cyclic group G_{N^s}

(for example $\mathfrak{g}_\perp = 1 + N \in \mathbb{Z}_{N^{s+1}}$, where $+$ here is the additive law of $\mathbb{Z}_{N^{s+1}}$) and let $\bar{\mathcal{L}} = \mathcal{L} + \mathfrak{g}_\perp$. We set $\Lambda = (N, s, \mathfrak{g}, \mathfrak{g}_\perp)$.

One cannot sample uniform witnesses as $\mathcal{W} = \mathbb{Z}$ is infinite. We cannot set $\mathcal{W} = \mathbb{Z}_{N'}$, as computing N' from $\Lambda = (N, s, \mathfrak{g})$ requires to factor N. Instead, we sample witnesses uniformly from $S_N := \{0, \dots, \lfloor N/4 \rfloor - 1\}$. Clearly, $\mathrm{SD}(U(\mathbb{Z}_{N'}), U(S_N)) = 1 - p'q'/(pq/4) = (2p' + 2q' + 1)/(pq) < 2(p+q)/(pq) < 4/\operatorname{spf}(N)$. From this distribution over $\mathcal{W}$, we can derive distributions over ϱ, $\mathcal{L}$, and $\bar{\mathcal{L}} = \mathcal{L} + \mathfrak{g}_\perp$. The two latter distributions are statistically close to uniform.

This setting defines a subset membership problem, whose $(\mathcal{L}, \bar{\mathcal{L}})$-indistinguishability property can be proven under the *Decisional Composite Residuosity* (DCR, [29]) assumption. More precisely, we consider the DCR assumption for moduli that are product of safe primes; the DCR assumption then basically states that in the case $s = 1$, no probabilistic polynomial time adversary can distinguish between uniform elements of $\mathcal{L}$ and $\mathcal{X}$.[5] This is a classical variant of DCR, which is equivalent to the original DCR assumption [29], assuming that safe primes are sufficiently dense (see, e.g., [14]). We prove the following lemma in the full version following [15]:

Lemma 3. *Assuming the DCR assumption, the above subset membership problems is $(\mathcal{L}, \bar{\mathcal{L}})$-indistinguishable. More precisely, if there exists an adversary $\mathcal{A}$ that has advantage $\varepsilon_{\mathcal{A}}$ in breaking $(\mathcal{L}, \bar{\mathcal{L}})$-indistinguishability, then there exists an attacker $\mathcal{B}$ that runs in approximately the same time and that has advantage $\varepsilon_{\mathcal{B}}$ in breaking DCR, such that $\varepsilon_{\mathcal{A}} \leq 2s \cdot \varepsilon_{\mathcal{B}} + 8/\operatorname{spf}(N)$.*

2.2 Projective Hash Functions

In [14], Cramer and Shoup defined the influential notion of *projective hash functions* (PHFs) to construct IND-CPA and even IND-CCA secure public-key encryption schemes. In this section, we recall the definition of a PHF using the notation of [3].

Let $\mathbf{P}$ be a subset membership problem, specifying an ensemble $(I_\kappa)_\kappa$ of instance distributions. A *projective hash function* for $\mathbf{P}$ is a tuple $\mathsf{PHF} = (\mathsf{hashkg}, \mathsf{projkg}, \mathsf{hash}, \mathsf{projhash})$ of four probabilistic polynomial time algorithms:

– $\mathsf{hashkg}(\Lambda)$ generates a hashing key hk in some set $\mathcal{K}$ for the instance $\Lambda = \Lambda[\mathcal{X}, \mathcal{L}, \mathcal{W}, \varrho]$,
– $\mathsf{projkg}(\mathsf{hk})$ (deterministically) derives from the hashing key hk a projection key $\mathfrak{hp}$,
– $\mathsf{hash}(\mathsf{hk}, \mathfrak{b})$ (deterministically) computes the hash value $\mathfrak{H}$ (in some efficiently recognizable set Π) of $\mathfrak{b} \in \mathcal{X}$ under $\mathsf{hk} \in \mathcal{K}$,
– $\mathsf{projhash}(\mathfrak{hp}, \mathfrak{b}, w)$ (deterministically) computes the projected hash value $\mathfrak{pH}$ of $\mathfrak{b} \in \mathcal{L}$ using a witness $w \in \mathcal{W}$.

[5] The original assumption actually does not restrict the the elements to be of Jacobi symbol 1, but doing this restriction yields an equivalent assumption, since we can multiply element of Jacobi symbol -1 by an arbitrary N^s-residue of Jacobi symbol -1.

A PHF must be *complete*, in the following sense:

– For any instance Λ, for any $\mathfrak{b} \in \mathcal{X}$ and $w \in \mathcal{W}$, such that $(\mathfrak{b}, w) \in \varrho$, for any hashing key $\mathsf{hk} \in \mathcal{K}$, if $\mathfrak{hp} \leftarrow \mathsf{projkg}(\mathsf{hk})$, then

$$\mathsf{hash}(\mathsf{hk}, \mathfrak{b}) = \mathsf{projhash}(\mathfrak{hp}, \mathfrak{b}, w) \ .$$

The instance Λ is implicitly included in the hashing key hk and the projection key $\mathfrak{hp}$.

2.3 Functional Encryption

A *functionality* $\mathcal{F}$ defined over $(\mathcal{Y}, \mathcal{Z})$ is a function $\mathcal{Y} \times \mathcal{Z} \to \Sigma \cup \{\perp\}$, where $\mathcal{Y}$ is a key space, $\mathcal{Z}$ is a message space, and Σ is an output space that does not contain the special symbol $\perp$.

A *functional encryption scheme for functionality* $\mathcal{F}$ [28,10] is a tuple $\mathsf{FE} = (\mathsf{setup}, \mathsf{keygen}, \mathsf{enc}, \mathsf{dec})$ of four probabilistic polynomial time algorithms:

$\mathsf{setup}(1^\kappa, \ell)$: generates system parameters pp, and then returns a master secret and public key pair (msk, mpk), where both msk and mpk also contain pp,
$\mathsf{keygen}_{\mathrm{msk}}(y \in \mathcal{Y})$: given a master secret key msk and a key (or a function) y, returns a partial secret key $\mathrm{msk}_y = (\mathrm{pp}, k_y, y)$,
$\mathsf{enc}_{\mathrm{mpk}}(z \in \mathcal{Z})$: given a master public key mpk and a plaintext z, returns a ciphertext c,
$\mathsf{dec}_{\mathrm{msk}_y}(c)$: returns $S \in \Sigma \cup \{\perp\}$.

Note that according to this definition, pp and y are always a part of msk_y, and thus k_y is basically “the rest of” msk_y. The public value ℓ contains some information about y and z that can be made public (e.g., their lengths).

FE must be *complete*, in the sense that if (y, z) is in the domain of $\mathcal{F}$, then for all $(\mathrm{msk}, \mathrm{mpk}) \leftarrow_r \mathsf{setup}(1^\kappa)$, for all $\mathrm{msk}_y \leftarrow_r \mathsf{keygen}_{\mathrm{msk}}(y)$, and for all $c \leftarrow_r \mathsf{enc}_{\mathrm{mpk}}(z)$, it holds that $\mathsf{dec}_{\mathrm{msk}_y}(c) = \mathcal{F}(y, z)$.

Definition 4 (IND-FE-CCA Security). *A functional encryption scheme* $\mathsf{FE} = (\mathsf{setup}, \mathsf{keygen}, \mathsf{enc}, \mathsf{dec})$ *is* IND-FE-CCA secure *(or,* secure against chosen ciphertext attacks*) [26], if no probabilistic polynomial time adversary $\mathcal{A}$ has a non-negligible advantage in the following game:*
1. *The challenger sets* $(\mathrm{msk}, \mathrm{mpk}) \leftarrow_r \mathsf{setup}(1^\kappa, 1^\ell)$ *and sends* mpk *to* $\mathcal{A}$.
2. *$\mathcal{A}$ makes adaptive secret key and decryption queries to the challenger. At each secret key query, $\mathcal{A}$ chooses $y \in \mathcal{Y}$ and obtains* $\mathrm{msk}_y = (\mathrm{pp}, k_y, y) \leftarrow_r \mathsf{keygen}_{\mathrm{msk}}(y)$. *Let y_i be the ith queried secret key.*
At each decryption query, $\mathcal{A}$ chooses a ciphertext c' and $y \in \mathcal{Y}$, then the challenger computes $\mathrm{msk}_y = (\mathrm{pp}, k_y, y) \leftarrow_r \mathsf{keygen}_{\mathrm{msk}}(y)$ *and sends back* $\mathsf{dec}_{\mathrm{msk}_y}(c')$ *to* $\mathcal{A}$.
3. *$\mathcal{A}$ chooses $z_0 \neq z_1$ such that $\mathcal{F}(y_i, z_0) = \mathcal{F}(y_i, z_1)$ for for all queried y_i. $\mathcal{A}$ sends z_0 and z_1 to the challenger. The challenger chooses $\beta \leftarrow_r \{0, 1\}$, and sends $c \leftarrow_r \mathsf{enc}_{\mathrm{mpk}}(z_\beta)$ to $\mathcal{A}$.*

4. *$\mathcal{A}$ makes more secret key queries for keys $y_i \in \mathcal{Y}$, with the condition that $\mathcal{F}(y_i, z_0) = \mathcal{F}(y_i, z_1)$, and possibly some more decryption queries (c', y), with the condition that $c' \neq c$.*
   Let q_{dec} be the number of decryption queries made during the whole game, and let (c'_j, y_j) be the jth decryption query.
5. *$\mathcal{A}$ outputs a bit $\beta_A \in \{0, 1\}$ and wins if $\beta_A = \beta$.*

More precisely, the advantage of $\mathcal{A}$ is defined as

$$\mathsf{Adv}^{\mathsf{ind-fe-cca}}_{\mathsf{FE},\mathcal{A}}(\kappa) := 2 \cdot |\Pr[\beta_A = \beta] - 1/2| \enspace .$$

FE *is* IND-FE-CCA secure*, if* $\mathsf{Adv}^{\mathsf{ind-fe-cca}}_{\mathsf{FE},\mathcal{A}}$ *is negligible for all probabilistic polynomial time adversaries $\mathcal{A}$.*

FE is *IND-FE-CPA secure* (or, *adaptively secure against chosen plaintexts attacks*, [28,10]), if $\mathsf{Adv}^{\mathsf{ind-fe-cca}}_{\mathsf{FE},\mathcal{A}}$ is negligible for all probabilistic polynomial time adversaries $\mathcal{A}$ that make no decryption queries.

The selective IND-FE-CPA security satisfied by [2] has the further requirement that the challenge messages $\vec{z}_0$ and $\vec{z}_1$ have to be chosen before the adversary sees the public key mpk.

Definition 5 (Inner-Product Functional Encryption). *In the* inner-product functional encryption *[2],* $\mathsf{setup}(1^\kappa, \ell)$ *in particular chooses a ring $\mathcal{R}$ and two efficiently recognizable subsets $\mathcal{Y}$ and $\mathcal{Z}$ of $\mathcal{R}^\ell$, each y (resp., z) corresponds to some vector $\vec{y} \in \mathcal{Y} \subseteq \mathcal{R}^\ell$ (resp., $\vec{z} \in \mathcal{Z} \subseteq \mathcal{R}^\ell$), and $\mathcal{F}(\vec{y}, \vec{z}) := \langle \vec{y}, \vec{z} \rangle \in \mathcal{R}$.*

We insist on the fact that $\langle \vec{y}, \vec{z} \rangle$ is computed in $\mathcal{R}$.

3 FE-CPA-Friendly Projective Hash Function

In this section, we first present the properties we need on PHFs in order to build an IND-FE-CPA secure IPFE. Then we show some examples of standard PHFs satisfying them.

3.1 Key Homomorphism and Projection Key Homomorphism

For correctness of the IPFE we will need the following property.

Definition 6 (Key Homomorphism [6]). *A projective hash function* $\mathsf{PHF} = (\mathsf{hashkg}, \mathsf{projkg}, \mathsf{hash}, \mathsf{projhash})$ *for a subset membership problem* $\mathbf{P}$ *is* key-homomorphic*, if it satisfies the following additional properties:*

1. *the set $\mathcal{K}$ of hashing keys and the set Π of hash values are additive Abelian groups, with polynomial time group operations;*
2. *for any instance Λ, and any word $\mathfrak{b} \in \mathcal{X}$, the function* $\mathsf{hk} \in \mathcal{K} \mapsto \mathsf{hash}(\mathsf{hk}, \mathfrak{b}) \in \Pi$ *is a group homomorphism, that is,* $\mathsf{hash}(\mathsf{hk}, \mathfrak{b}) + \mathsf{hash}(\mathsf{hk}', \mathfrak{b}) = \mathsf{hash}(\mathsf{hk} + \mathsf{hk}', \mathfrak{b})$*, for any* $\mathsf{hk}, \mathsf{hk}' \in \mathcal{K}$.

We do not require $\mathcal{K}$ to be finite. In the DCR construction, $\mathcal{K} = \mathbb{Z}$. However, we require that each group element of $\mathcal{K}$ and Π has a unique representation as a bit-string.

The next property, *projection key homomorphism*, is only required in Sect. 5.3 (for the CCA security). We will introduce it already here, since all our concrete examples from Sect. 3.5 coincidentally satisfy this property.

Definition 7 (Projection Key Homomorphism). *A projective hash function* $\mathsf{PHF} = (\mathsf{hashkg}, \mathsf{projkg}, \mathsf{hash}, \mathsf{projhash})$ *for a subset membership problem* $\mathbf{P}$ *is* projection-key-homomorphic *if it satisfies the following additional properties:*

1. *the set $\mathcal{K}$ of hashing keys and the set $\mathcal{K}_{\mathsf{hp}}$ of projection keys are additive Abelian groups, with polynomial time group operations;*
2. *for any instance Λ, the function $\mathsf{hk} \in \mathcal{K} \mapsto \mathsf{projkg}(\mathsf{hk}) \in \mathcal{K}_{\mathsf{hp}}$ is a group homomorphism, that is, $\mathsf{projkg}(\mathsf{hk} + \mathsf{hk}') = \mathsf{projkg}(\mathsf{hk}) + \mathsf{projkg}(\mathsf{hk}')$, for any $\mathsf{hk}, \mathsf{hk}' \in \mathcal{K}$.*

3.2 Strong Diversity

The second property we need for our PHFs is strong diversity. More precisely, we require that for each $\mathfrak{b}$ there exists a (not necessarily efficiently computable) hashing key $\mathsf{hk}_\perp(\mathfrak{b})$, such that hk and $\mathsf{hk} + \mathsf{hk}_\perp(\mathfrak{b})$ result in the same projection key, while the hash value of $\mathfrak{b}$ under the key $\mathsf{hk}_\perp(\mathfrak{b})$ is equal to $\mathfrak{g}_\perp$, where $\mathfrak{g}_\perp$ is a fixed efficiently computable group element.

Definition 8 (Strong diversity). *A key-homomorphic projective hash function* $\mathsf{PHF} = (\mathsf{hashkg}, \mathsf{projkg}, \mathsf{hash}, \mathsf{projhash})$ *for a subset membership problem* $\mathbf{P}$ *is* $(\mathsf{hk}_\perp, \mathfrak{g}_\perp, M_\perp)$*-strongly diverse for a function* $\mathsf{hk}_\perp : \bar{\mathcal{L}} \to \Pi$*, an element* $\mathfrak{g}_\perp$ *of* Π*, and a positive integer* $M_\perp$*, if the following properties are satisfied:*

1. *$\mathfrak{g}_\perp$ and $M_\perp$ can be efficiently computed from Λ;*
2. *the group element $\mathfrak{g}_\perp$ has order $M_\perp$,*
3. *for any hashing key $\mathsf{hk} \in \mathcal{K}$ and any word $\mathfrak{b} \in \bar{\mathcal{L}}$:*

$$\mathsf{projkg}(\mathsf{hk} + \mathsf{hk}_\perp(\mathfrak{b})) = \mathsf{projkg}(\mathsf{hk})\,, \tag{2}$$

$$\mathsf{hash}(\mathsf{hk}_\perp(\mathfrak{b}),\, \mathfrak{b}) = \mathfrak{g}_\perp \quad . \tag{3}$$

We do not require $\mathsf{hk}_\perp$ to be efficiently computable, as we are only using it to bound statistical distance.

In what follows, we will use the following straightforward lemma.

Lemma 9. *If a key-homomorphic PHF is also projection-key homomorphic, then Eq. (2) is true iff* $\mathsf{projkg}(\mathsf{hk}_\perp(\mathfrak{b})) = 0$.

Relation with Diverse Groups. Diverse groups were introduced in [14] as a way to construct PHFs. They can be seen as key-homomorphic projection-key-homomorphic strongly diverse PHFs with the two following differences: $\bar{\mathcal{L}} = \mathcal{X} \setminus \mathcal{L}$ (instead of $\bar{\mathcal{L}} \subseteq \mathcal{X} \setminus \mathcal{L}$), and for any $\mathsf{hk} \in \mathcal{K}$ and any $\mathfrak{b} \in \bar{\mathcal{L}}$, it is only required that $\mathsf{hash}(\mathsf{hk} + \mathsf{hk}_\perp(\mathfrak{b}), \mathfrak{b}) \neq 0$ instead of $\mathsf{hash}(\mathsf{hk} + \mathsf{hk}_\perp(\mathfrak{b}), \mathfrak{b}) = \mathfrak{g}_\perp$. Nevertheless, all the diverse groups we currently know of are also strongly diverse for $\bar{\mathcal{L}} = \mathcal{X} \setminus \mathcal{L}$.

3.3 Translation Indistinguishability

We also require one last statistical property, translation indistinguishability. Informally it says that translating the hashing key of the PHF by a small multiple of $\mathsf{hk}_\perp(\mathfrak{b})$ cannot be detected with non-negligible probability. In the proof, we use this as a statistical argument to conclude after using the computational assumption.

Definition 10 (Translation indistinguishability). *A key-homomorphic projective hash function* $\mathsf{PHF} = (\mathsf{hashkg}, \mathsf{projkg}, \mathsf{hash}, \mathsf{projhash})$ *is* $(\mathsf{hk}_\perp, M_z, \varepsilon_{\mathrm{ti}})$-translation-indistinguishable *for a function* $\mathsf{hk}_\perp : \bar{\mathcal{L}} \to \Pi$*, a positive integer* M_z*, and* $\varepsilon_{\mathrm{ti}} \in [0,1]$*, if for any integer* $z \in \{-M_z, \dots, M_z\}$ *and for any* $\mathfrak{b} \in \bar{\mathcal{L}}$*,*

$$\mathrm{SD}(\mathsf{hashkg}(\Lambda), \mathsf{hashkg}(\Lambda) + z \cdot \mathsf{hk}_\perp(\mathfrak{b})) \le \varepsilon_{\mathrm{ti}} \enspace .$$

Important Particular Case: Key Uniformity. For many key-homomorphic PHFs, like the above described ones based on DDH and MDDH, the output of hashkg is actually uniform over the group $\mathcal{K}$. In this case, the PHF is automatically $(\cdot,\cdot,0)$-translation-indistinguishable. More formally, we have the following lemma.

Lemma 11. *Let* $\mathsf{PHF} = (\mathsf{hashkg}, \mathsf{projkg}, \mathsf{hash}, \mathsf{projhash})$ *be a key-homomorphic PHF such that the distribution of* $\mathsf{hashkg}(\Lambda)$ *is uniform over* $\mathcal{K}$*. Let* $\bar{\mathcal{L}}$ *be a non-empty subset of* $\mathcal{X}$*,* $\mathsf{hk}_\perp$ *be a function from* $\bar{\mathcal{L}}$ *to* Π *and* M_z *be a positive integer. Then* PHF *is* $(\bar{\mathcal{L}}, \mathsf{hk}_\perp, M_z, 0)$*-translation-indistinguishable.*

Proof. Both $\mathsf{hashkg}(\Lambda)$ and $\mathsf{hashkg}(\Lambda) + z \cdot \mathsf{hk}_\perp(\mathfrak{b})$ are uniform group elements in $\mathcal{K}$. □

3.4 FE-CPA Friendliness

In the following, we regroup all 3 properties we have defined under the FE-CPA friendliness property.

Definition 12 (FE-CPA Friendliness). *A projective hash function* $\mathsf{PHF} = (\mathsf{hashkg}, \mathsf{projkg}, \mathsf{hash}, \mathsf{projhash})$ *is* $(\mathsf{hk}_\perp, \mathfrak{g}_\perp, M_\perp, M_z, \varepsilon_{\mathrm{ti}})$-FE-CPA-friendly *for a function* $\mathsf{hk}_\perp$ *from* $\bar{\mathcal{L}}$ *to* Π*, an element* $\mathfrak{g}_\perp$ *of* Π*, and two positive integers* $M_\perp$ *and* M_z*, if it is key-homomorphic,* $(\mathsf{hk}_\perp, \mathfrak{g}_\perp, M_\perp)$*-strongly diverse, and* $(\mathsf{hk}_\perp, M_z, \varepsilon_{\mathrm{ti}})$*-translation-indistinguishable.*

3.5 Examples

In this section, we describe FE-CPA-friendly PHFs for the subset membership problems described in Sect. 2.1.

DDH. Let $\mathbb{G}$ be an additive cyclic group of prime order q, let $\mathcal{X} = \mathbb{G}^2$, let $\mathcal{L}$ be the subgroup of $\mathcal{X}$ generated by $\boldsymbol{\mathfrak{g}} = (\mathfrak{g}_1, \mathfrak{g}_2)^\intercal \in \mathbb{G}^2$, where $\mathfrak{g}_i$ are random generators of $\mathbb{G}$. A witness $w \in \mathcal{W} = \mathbb{Z}_q$ for $\boldsymbol{\mathfrak{b}} \in \mathcal{L}$ is such that $\boldsymbol{\mathfrak{b}} = w \cdot \boldsymbol{\mathfrak{g}}$. We set $\Lambda = (\mathbb{G}, \boldsymbol{\mathfrak{g}})$.

We recall the PHF of Cramer and Shoup [13, Section 8.1.1] defined as follows:

$\mathsf{hashkg}(\Lambda)$**:** output $\mathbf{hk} \leftarrow_r \mathbb{Z}_q^2 = \mathcal{K}$,
$\mathsf{projkg}(\mathbf{hk})$**:** output $\mathfrak{hp} \leftarrow \mathbf{hk}^\intercal \cdot \boldsymbol{\mathfrak{g}} \in \mathbb{G}$,
$\mathsf{hash}(\mathbf{hk}, \boldsymbol{\mathfrak{b}})$**:** output $\mathfrak{H} \leftarrow \mathbf{hk}^\intercal \cdot \boldsymbol{\mathfrak{b}} \in \mathbb{G} = \Pi$,
$\mathsf{projhash}(\mathfrak{hp}, \boldsymbol{\mathfrak{b}}, w)$**:** output $\mathfrak{pH} \leftarrow \mathfrak{hp} \cdot w \in \mathbb{G} = \Pi$.

Lemma 13. *Using above notation, let $\mathfrak{g}_\perp$ an arbitrary generator of $\mathbb{G}$, $M_\perp = q$, M_z be a positive integer, and $\varepsilon_{\mathrm{ti}} = 0$. For any $\mathfrak{b} \in \mathcal{X} \setminus \mathcal{L}$, let $\mathsf{hk}_\perp(\boldsymbol{\mathfrak{b}})$ be defined as follows:*

$$\mathbf{hk}_\perp(\boldsymbol{\mathfrak{b}}) = \frac{\log_{\mathfrak{g}_1} \mathfrak{g}_\perp}{\log_{\mathfrak{g}_1} \mathfrak{b}_1 \cdot \log_{\mathfrak{g}_1} \mathfrak{g}_2 - \log_{\mathfrak{g}_1} \mathfrak{b}_2} \cdot \begin{pmatrix} \log_{\mathfrak{g}_1} \mathfrak{g}_2 \\ -1 \end{pmatrix} \quad \textit{with } \boldsymbol{\mathfrak{b}} = \begin{pmatrix} \mathfrak{b}_1 \\ \mathfrak{b}_2 \end{pmatrix} \in \mathbb{G}^2 \ .$$

Then, the PHF described above is $(\mathsf{hk}_\perp, \mathfrak{g}_\perp, M_\perp, M_z, \varepsilon_{\mathrm{ti}})$-FE-CPA-friendly.

Proof. We first remark that $\mathbf{hk}_\perp(\boldsymbol{\mathfrak{b}})$ is well defined, as $\log_{\mathfrak{g}_1} \mathfrak{b}_1 \cdot \log_{\mathfrak{g}_1} \mathfrak{g}_2 \neq \log_{\mathfrak{g}_1} \mathfrak{b}_2$ since $\boldsymbol{\mathfrak{b}} \notin \mathcal{L}$.

Key Homomorphism is straightforward.

Strong Diversity. Since the space of projection keys is also a group and projkg is a group homomorphism, we can use Lem. 9. Hence, we just need to prove that $\mathsf{projkg}(\mathbf{hk}_\perp(\boldsymbol{\mathfrak{b}})) = 0$ and $\mathsf{hash}(\mathbf{hk}_\perp(\boldsymbol{\mathfrak{b}}), \boldsymbol{\mathfrak{b}}) = \mathfrak{g}_\perp$. This follows from the following two facts:

$$\begin{aligned} \mathsf{projkg}(\mathbf{hk}_\perp(\boldsymbol{\mathfrak{b}})) &= \frac{\log_{\mathfrak{g}_1} \mathfrak{g}_\perp}{\log_{\mathfrak{g}_1} \mathfrak{b}_1 \cdot \log_{\mathfrak{g}_1} \mathfrak{g}_2 - \log_{\mathfrak{g}_1} \mathfrak{b}_2} \cdot \begin{pmatrix} \log_{\mathfrak{g}_1} \mathfrak{g}_2 & -1 \end{pmatrix} \cdot \begin{pmatrix} \mathfrak{g}_1 \\ \mathfrak{g}_2 \end{pmatrix} \ , \\ \mathsf{hash}(\mathbf{hk}_\perp(\boldsymbol{\mathfrak{b}}), \boldsymbol{\mathfrak{b}}) &= \frac{\log_{\mathfrak{g}_1} \mathfrak{g}_\perp}{\log_{\mathfrak{g}_1} \mathfrak{b}_1 \cdot \log_{\mathfrak{g}_1} \mathfrak{g}_2 - \log_{\mathfrak{g}_1} \mathfrak{b}_2} \cdot \begin{pmatrix} \log_{\mathfrak{g}_1} \mathfrak{g}_2 & -1 \end{pmatrix} \cdot \begin{pmatrix} \mathfrak{b}_1 \\ \mathfrak{b}_2 \end{pmatrix} \ . \end{aligned}$$

Translation Indistinguishability follows from Lem. 11. □

MDDH. Let $\Lambda = (\mathbb{G}, \boldsymbol{\mathfrak{g}})$ be defined as in the MDDH subsubsection of Sect. 2.1 on page 7. We recall that $\boldsymbol{\mathfrak{g}} \in \mathbb{G}^{t \times d}$, $\mathcal{X} = \mathbb{G}^t$, $\mathcal{L}$ is the subgroup generated by the columns of $\boldsymbol{\mathfrak{g}}$, and $\bar{\mathcal{L}} = \mathcal{X} \setminus \mathcal{L}$. A witness $\boldsymbol{w} \in \mathcal{W} = \mathbb{Z}_q^d$ for $\boldsymbol{\mathfrak{b}} \in \mathcal{L}$ is such that $\boldsymbol{\mathfrak{b}} = \boldsymbol{\mathfrak{g}} \cdot \boldsymbol{w}$.

We recall the PHF defined by Escala et al. in [18]:

$\mathsf{hashkg}(\Lambda)$**:** output $\mathbf{hk} \leftarrow_r \mathbb{Z}_q^t = \mathcal{K}$,
$\mathsf{projkg}(\mathbf{hk})$**:** output $\boldsymbol{\mathfrak{hp}} \leftarrow \boldsymbol{\mathfrak{g}}^\intercal \cdot \mathbf{hk} \in \mathbb{G}^d$,
$\mathsf{hash}(\mathbf{hk}, \boldsymbol{\mathfrak{b}})$**:** output $\mathfrak{H} \leftarrow \mathbf{hk}^\intercal \cdot \boldsymbol{\mathfrak{b}} \in \mathbb{G} = \Pi$,
$\mathsf{projhash}(\boldsymbol{\mathfrak{hp}}, \boldsymbol{\mathfrak{b}}, w)$**:** output $\mathfrak{pH} \leftarrow \boldsymbol{\mathfrak{hp}}^\intercal \cdot \boldsymbol{w} \in \mathbb{G} = \Pi$.

We can prove the following lemma similarly to Lem. 13:

Lemma 14. *Using above notation, let $\mathfrak{g}_\perp$ an arbitrary generator of $\mathbb{G}$, $M_\perp = q$, M_z be a positive integer, and $\varepsilon_{\mathrm{ti}} = 0$. Let $\mathsf{hk}_\perp(\mathfrak{b})$ be an arbitrary vector satisfying $\mathsf{hk}_\perp(\mathfrak{b})^\intercal \cdot \mathfrak{g} = 0$ and $\mathsf{hk}_\perp(\mathfrak{b})^\intercal \cdot \vec{\mathfrak{b}} = \mathfrak{g}_\perp$, which exists as $\vec{\mathfrak{b}}$ is not in the span of the columns of $\mathfrak{g}$. Then, the PHF described above is $(\mathsf{hk}_\perp, \mathfrak{g}_\perp, M_\perp, M_z, \varepsilon_{\mathrm{ti}})$-FE-CPA-friendly.*

DCR. Let $\Lambda = (N, s, \mathfrak{g}, \mathfrak{g}_\perp)$ be defined as in the DCR subsubsection of Sect. 2.1 on page 7. We have: $\mathbb{G} = \mathcal{X} = J_{N^{s+1}} \cong G_{N^s} \oplus G_{N'} \oplus T$, $\mathcal{L} = G_{N'}$, and $\bar{\mathcal{L}} = \mathcal{L} + \mathfrak{g}_\perp$. The element $\mathfrak{g}$ is a generator of $\mathcal{L}$, while $\mathfrak{g}_\perp$ is a generator of G_{N^s}. We recall that we use additive notation for the group $\mathbb{G}$.

We define the DCR-based PHF as follows:

$\mathsf{hashkg}(\Lambda)$**:** output $\mathsf{hk} \leftarrow_r \{0, \dots, \lfloor M N^{s+1}/4 \rfloor\} =: \mathcal{K}^* \subseteq \mathbb{Z} =: \mathcal{K}$, where M is a positive integer and is a parameter of the scheme,
$\mathsf{projkg}(\mathsf{hk})$**:** output $\mathfrak{hp} \leftarrow \mathsf{hk} \cdot \mathfrak{g} \in \mathbb{G}$,
$\mathsf{hash}(\mathsf{hk}, \mathfrak{b})$**:** output $\mathfrak{H} \leftarrow \mathsf{hk} \cdot \mathfrak{b} \in \mathbb{G} =: \Pi$,
$\mathsf{projhash}(\mathfrak{hp}, \mathfrak{b}, w)$**:** output $\mathfrak{pH} \leftarrow \mathfrak{hp} \cdot w \in \mathbb{G} = \Pi$.

When $M = 2$, this PHF corresponds to the one of Cramer and Shoup in [14].

We insist on the fact that the set of hashing keys is $\mathcal{K} = \mathbb{Z}$ so that it is a group. However, hashkg only samples a hashing key from a finite subset $\mathcal{K}^*$ of $\mathcal{K}$.

Lemma 15. *Using above notation, let $M_\perp = N^s$, M_z be a positive integer, and $\varepsilon_{\mathrm{ti}} = M_z/M$. Let $\mathsf{hk}_\perp$ be defined as follows:*

$$\mathsf{hk}_\perp(\mathfrak{b}) = N' \cdot (N'^{-1} \bmod N^s) \quad (< N' N^s < N^{s+1}/4) \enspace .$$

Then, the PHF described above is $(\mathsf{hk}_\perp, \mathfrak{g}_\perp, M_\perp, M_z, \varepsilon_{\mathrm{ti}})$-FE-CPA-friendly.

Key homomorphism and strong diversity are proven similarly as in the DDH case, while translation indistinguishability follows from Lem. 2. The complete proof is given in the full version.

Interestingly, because of our choice of $\bar{\mathcal{L}}$, $\mathsf{hk}_\perp(\mathfrak{b})$ does not depend on $\mathfrak{b}$. Note also that for $M < M_z/\varepsilon_{\mathrm{ti}}$, this PHF is still key-homomorphic and strongly diverse, but might lack the translation indistinguishability property that is necessary for our application.

4 IND-FE-CPA Inner-Product Functional Encryption

In this section, we first show a generic construction of an IND-FE-CPA secure inner-product functional encryption scheme from a FE-CPA-friendly projective hash function. Then, we show two concrete instantiations, based on the DDH and on the DCR assumptions.

4.1 Generic Construction

We now define our generic construction for IND-FE-CPA secure IPFEs. Intuitively, we use ℓ PHFs in parallel, that are combined during decryption in order to only reveal a linear combination of the hashes, which implies that it only reveals this same linear combination of the messages. This restriction is enforced by the key generation algorithm, which only outputs linear combinations of the hashing keys.

Construction. We suppose that we have a $(\mathsf{hk}_\perp, \mathfrak{g}_\perp, M_\perp, z, \varepsilon_{\mathrm{ti}})$-FE-CPA-friendly projective hash function $\mathsf{PHF} = (\mathsf{hashkg}, \mathsf{projkg}, \mathsf{hash}, \mathsf{projhash})$ for a subset membership problem $\mathbf{P}$. Let $\mathcal{R}$ be the ring $\mathbb{Z}$ or $\mathbb{Z}_{M_\perp}$, let ℓ be a positive integer parameter corresponding to the length of the message and key vectors, and let $\mathcal{Y}$ and $\mathcal{Z}$ two subsets of $\mathcal{R}^\ell$.[6] We always suppose ℓ to be polynomial in the security parameter κ.

We suppose that the following condition is satisfied.

Condition 1. *Using the above notation:*
1. *if $\mathcal{R} = \mathbb{Z}_{M_\perp}$, the order of any hashing key $\mathsf{hk} \in \mathcal{K}$ divides $M_\perp$;*
2. *$\mathcal{Y}$ and $\mathcal{Z}$ are efficiently recognizable subsets of $\mathcal{R}^\ell$;*
3. *for any $\vec{z} \in \mathcal{Z}$ and any i, $z_i \in \{-M_z, \dots, M_z\}$;*
4. *there exists a polynomial time algorithm (in the security parameter κ) that given as input $\mathfrak{c}_{\vec{y}} = \langle \vec{y}, \vec{z} \rangle \cdot \mathfrak{g}_\perp$ for $\vec{y} \in \mathcal{Y}$ and $\vec{z} \in \mathcal{Z}$, can compute $\log_{\mathfrak{g}_\perp} \mathfrak{c}_{\vec{y}} = \langle \vec{y}, \vec{z} \rangle$;*
5. *for any $\vec{y} \in \mathcal{Y}$ and $\vec{z} \in \mathcal{Z}$, $\langle \vec{y}, \vec{z} \rangle$ is the same over $\mathcal{R}$ and over $\mathbb{Z}_{M_\perp}$ (this condition is trivial when $\mathcal{R} = \mathbb{Z}_{M_\perp}$).*

The first subcondition implies that $\mathcal{K}$ is a $\mathcal{R}$-module, which implies that, for any $t \in \mathcal{R}$, $t \cdot \mathsf{hk}$ is well defined. The second subcondition enables keygen and enc to check in polynomial-time the validity of their arguments y and z respectively. The third subcondition is used in the proof to apply the $(\mathsf{hk}_\perp, M_z, \varepsilon_{\mathrm{ti}})$-translation indistinguishability property. The fourth subcondition ensures that decryption can be performed in polynomial time. The last subcondition is similar as the condition in the "over $\mathbb{Z}$ constructions in [5]. If $\mathcal{R} = \mathbb{Z}_{M_\perp}$, then—as in [5]—a simple way to guarantee that subconditions 3 and 5 hold is to assume that $|y_i|, |z_i| < (M_\perp/\ell)^{1/2}$ for each $\vec{y} \in \mathcal{Y}$, $\vec{z} \in \mathcal{Z}$, and $i \leq \ell$. The fourth subcondition can potential restrict the values $|y_i|$ and $|z_i|$ even more.

Our generic IND-FE-CPA IPFE scheme $\mathsf{FE}_{\mathrm{phf}}$ is depicted in Fig. 1.

Security. We define the following set:

$$\Delta\mathcal{Z} := \{\vec{z}_1 - \vec{z}_0 \ : \ \vec{z}_0, \vec{z}_1 \in \mathcal{Z}\} \ .$$

Its cardinality $|\Delta\mathcal{Z}|$ is at most $(4M_z - 1)^\ell$, as the cardinality of $\mathcal{Z}$ is at most $2M_z$.

We have the following security theorem.

[6] Formally, $\mathcal{Y}$ and $\mathcal{Z}$ are collections of subsets indexed by ℓ and Λ.

1. Let $\mathbf{P}$ be a subset membership problem. Let $\mathsf{PHF} = (\mathsf{hashkg}, \mathsf{projkg}, \mathsf{hash}, \mathsf{projhash})$ be a $(\mathsf{hk}_\perp, \mathfrak{g}_\perp, M_\perp, M_z, \varepsilon_{\text{ti}})$-FE-CPA-friendly PHF. We assume that Cond. 1 is satisfied.
2. $\mathsf{setup}(1^\kappa, \ell)$: Sample $\Lambda \leftarrow_r I_\kappa$, and set $\text{pp} \leftarrow (\kappa, \ell, \Lambda)$. Define $\vec{\mathsf{hk}} \in \mathcal{K}^\ell$ and $\vec{\mathfrak{hp}} \in \mathcal{K}_{\mathsf{hp}}^\ell$ by setting
$$\mathsf{hk}_i \leftarrow_r \mathsf{hashkg}(\Lambda)\,, \qquad \mathfrak{hp}_i \leftarrow \mathsf{projkg}(\mathsf{hk}_i)\,.$$
Set $\text{msk} \leftarrow (\text{pp}, \vec{\mathsf{hk}})$ and $\text{mpk} \leftarrow (\text{pp}, \vec{\mathfrak{hp}})$, and return (msk, mpk).
3. $\mathsf{keygen}_{\text{msk}}(\vec{y} \in \mathcal{Y})$: Set $\mathsf{hk}_{\vec{y}} \leftarrow \langle \vec{y}, \vec{\mathsf{hk}} \rangle \in \mathcal{K}$, and return $\text{msk}_{\vec{y}} \leftarrow (\text{pp}, \mathsf{hk}_{\vec{y}}, \vec{y})$.
4. $\mathsf{enc}_{\text{mpk}}(\vec{z} \in \mathcal{Z})$: Sample a random pair $(\mathfrak{b}, w) \in \varrho$. Define $\vec{\mathfrak{c}} \in \Pi^\ell$ by setting
$$\mathfrak{c}_i \leftarrow \mathsf{projhash}(\mathfrak{hp}_i, \mathfrak{b}, w) + z_i \cdot \mathfrak{g}_\perp\,.$$
Return $(\mathfrak{b}, \vec{\mathfrak{c}})$.
5. $\mathsf{dec}_{\text{msk}_{\vec{y}}}(\mathfrak{b}, \vec{\mathfrak{c}})$: Check that $\mathfrak{b} \in \mathcal{X}$ and $\vec{\mathfrak{c}} \in \Pi^\ell$; return $\perp$ if any check fails. Set
$$\mathfrak{c}_{\vec{z}} \leftarrow \langle \vec{y}, \vec{\mathfrak{c}} \rangle - \mathsf{hash}(\mathsf{hk}_{\vec{y}}, \mathfrak{b})\ \ .$$
Return $\log_{\mathfrak{g}_\perp} \mathfrak{c}_{\vec{z}}$.

Fig. 1. Generic inner-product functional encryption FE_{phf} scheme

Theorem 16. *Let* $\mathbf{P}$ *be a subset membership problem. Let* $\mathsf{PHF} = (\mathsf{hashkg}, \mathsf{projkg}, \mathsf{hash}, \mathsf{projhash})$ *be a* $(\mathsf{hk}_\perp, \mathfrak{g}_\perp, M_\perp, M_z, \varepsilon_{\text{ti}})$*-FE-CPA-friendly projective hash function. We assume that Cond. 1 is satisfied. Then the scheme* FE_{phf} *depicted in Fig. 1 is complete and adaptively IND-FE-CPA secure.*

More precisely, if there exists an attacker $\mathcal{A} = \mathcal{A}_{\mathsf{FE}}$ *that has advantage* $\varepsilon_{\mathcal{A}}$ *in breaking the IND-FE-CPA security of* FE_{phf}*, then there exists an attacker* $\mathcal{B}$ *that runs in approximately the same time and that has advantage* $\varepsilon_{\mathcal{B}}$ *in breaking the* $(\mathcal{L}, \bar{\mathcal{L}})$*-indistinguishability, such that*
$$\varepsilon_{\mathcal{A}} \leq 2 \cdot \varepsilon_{\mathcal{B}} + \ell \cdot |\Delta\mathcal{Z}| \cdot \varepsilon_{\text{ti}}\ \ .$$

The proof is provided in App. 16. As a quick overview, the proof is structured in two parts: first we use a computational assumption to show that sampling a word outside of the language for the challenge ciphertext is indistinguishable to the adversary. One this is done, the second part is a statistical argument claiming that the view of the adversary is then almost independent of the chosen bit β.

Remark 17. When $\varepsilon_{\text{ti}} \neq 0$, there is an exponential loss in the security proof in the term $\ell|\Delta\mathcal{Z}|\varepsilon_{\text{ti}}$. This term comes from the fact that at one point we guess the value of $\vec{z}_1 - \vec{z}_0$. This is not complexity leveraging, as the reduction loss is with regards to a statistical property. In particular, we do not need to rely on subexponential computational assumptions. Concretely, in our instantiations with DCR, we just need to take this security loss into account in the parameter

M defining the bound on the size of the hashing key (see Sect. 3.5 and 4.3). This approximately multiplies by $\log |\Delta\mathcal{Z}|$ the size of the secret keys which would be obtained if this security loss was not taken into account.

We also remark that if we used a selective security notion, where the adversary announces $\vec{z}_0$ and $\vec{z}_1$ before obtaining the public key, we would not lose the factor $|\Delta\mathcal{Z}|$. We could then use classical complexity leveraging to go from this selective notion to the adaptive one we are considering. But then, we would need to use sub-exponential $(\mathcal{L}, \bar{\mathcal{L}})$-indistinguishability (if ℓ is polynomial in the security parameter), and the size of the ciphertexts, of the secret and public keys, and of the public parameters (and not just of the secret keys) would be multiplied by $|\Delta\mathcal{Z}|$.

4.2 DDH-Based Instantiation

Let us instantiate the framework with the DDH-based PHF defined in Sect. 3.5 on page 13. We set $\mathcal{R} = \mathbb{Z}_q$ and $M_z = q$ (or any large enough integer). To satisfy Cond. 1, we need to choose the efficiently recognizable subsets $\mathcal{Y}$ and $\mathcal{Z}$ of $\mathcal{R}^\ell$ so that the discrete logarithm of $\langle \vec{y}, \vec{z} \rangle \cdot \mathfrak{g}_\perp \in \mathbb{G}$ is efficient to compute, for any $\vec{y} \in \mathcal{Y}$ and $\vec{z} \in \mathcal{Z}$. We recall that there exist generic algorithms to compute the discrete logarithm of an element $t \cdot \mathfrak{g}_\perp$ in $O(\sqrt{|T|})$ group operations, when t is in an interval T; and in $O(T)$ group operations, when t is in an arbitrary subset of $T \subseteq \mathbb{Z}_q$.

The resulting construction FE_{ddh} coincides with the DDH-based scheme in [5]. An explicit description of FE_{ddh} is provided in the full version. It can be easily extended to use any MDDH-based PHF defined in Sect. 3.5.

Applying Thm. 16, we immediately get the following security theorem.

Theorem 18. *Under the DDH assumption in $\mathbb{G}$, the scheme FE_{ddh} is complete and IND-FE-CPA.*

More precisely, if there exists an attacker $\mathcal{A} = \mathcal{A}_{\mathsf{FE}}$ that has advantage $\varepsilon_\mathcal{A}$ in breaking the IND-FE-CPA security of FE_{ddh}, then there exists an attacker $\mathcal{B}$ that runs in approximately the same time and that has advantage $\varepsilon_\mathcal{B}$ in breaking the DDH assumption, such that $\varepsilon_\mathcal{A} \leq 2 \cdot \varepsilon_\mathcal{B}$.

It is worth noting that the term $\ell \cdot |\Delta\mathcal{Z}| \cdot \varepsilon_{\text{ti}}$ has disappeared because of the key-uniformity.

4.3 DCR-based Instantiation

Let us instantiate the framework with the DCR-based PHF defined in Sect. 3.5 on page 14. We set $\mathcal{R} = \mathbb{Z}$. Contrary the DDH-based instantiation, the discrete logarithm problem in the subgroup generated by $\mathfrak{g}_\perp$ is easy: given $t \cdot \mathfrak{g}_\perp$, we can always efficiently recover t. However, to satisfy Cond. 1, we need to choose $\mathcal{Y}$ and $\mathcal{Z}$ so that for any $\vec{y} \in \mathcal{Y}$ and $\vec{z} \in \mathcal{Z}$, $\langle \vec{y}, \vec{z} \rangle$ is the same modulo $M_\perp = N^s$ and over the integers.

There are many ways to choose the parameters to satisfy this condition. We propose one possible way here.

Example 19 (Example of parameters for our DCR-based instantiation). Let M_y and M_z be positive integers such that $2M_yM_z + 1 \leq M_\perp = N^s$. We set:

$$\mathcal{Y} := \{\vec{y} \in \mathbb{Z}^\ell \, : \, \|\vec{y}\| \leq M_y\}\,, \qquad \mathcal{Z} := \{\vec{z} \in \mathbb{Z}^\ell \, : \, \|\vec{z}\| \leq M_z\}\,,$$
$$M := \ell \cdot 2^\kappa \cdot M_z \cdot |\Delta\mathcal{Z}| \leq \ell \cdot 2^\kappa \cdot M_z \cdot (4 \cdot M_z)^\ell\,,$$

where $\|.\|$ denotes the Euclidean norm, so that $|\langle \vec{y}, \vec{z} \rangle| \leq M_yM_z$ (when the inner-product is over the integers). For the last inequality, we use the rough inequality $|\Delta\mathcal{Z}| \leq (4 \cdot M_z)^\ell$. □

Then, we fix M_y and M_z so that $2M_yM_z + 1 \leq M_\perp$. And we choose M so that M_z/M is negligible.

The concrete DCR-based IPFE scheme $\mathsf{FE}_{\mathrm{dcr}}$ is fully described in the full version. $\mathsf{FE}_{\mathrm{dcr}}$ is length-flexible in the same sense as the cryptosystems of [15,16]. Namely, by fixing the parameter $s \in \mathbb{Z}^+$, one can obtain bigger or smaller sets M_z and M_y. Larger s however makes the scheme less efficient. Note that the sizes of our secret keys is slightly larger than those of [5], due to our security reduction; but we do not need to sample discrete Gaussian, as all the distributions we are using are uniform.

Applying Thm. 16 and Lem. 3, we immediately get the following security theorem.

Theorem 20. *Under the DCR assumption, the scheme* $\mathsf{FE}_{\mathrm{dcr}}$ *is complete and IND-FE-CPA.*

More precisely, if there exists an attacker $\mathcal{A} = \mathcal{A}_{\mathsf{FE}}$ *that has advantage* $\varepsilon_\mathcal{A}$ *in breaking the IND-FE-CPA security of* $\mathsf{FE}_{\mathrm{dcr}}$*, then there exists an attacker* $\mathcal{B}$ *that runs in approximately the same time and that has advantage* $\varepsilon_\mathcal{B}$ *in breaking the DCR assumption, such that* $\varepsilon_\mathcal{A} \leq 4s \cdot \varepsilon_\mathcal{B} + 16/\operatorname{spf}(N) + \ell \cdot |\Delta\mathcal{Z}| \cdot M_z/M$.

Using parameters from Ex. 19, we have the following security bound: $\varepsilon_\mathcal{A} \leq 4s \cdot \varepsilon_\mathcal{B} + 16/\operatorname{spf}(N) + 2^{-\kappa}$. Although there is an exponential loss in the security reduction of Thm. 16, we emphasize that there is no exponential loss using these parameters: the security loss is compensated by these well-chosen parameters. Most importantly, all the algorithms of the resulting scheme run in polynomial time (in the security parameter κ)[7] and the reduction to DCR is polynomial time. There is *no* complexity leveraging and we *do not* require subexponential assumption *nor* exponential-size keys or ciphertexts.

5 FE-CCA-Friendly Projective Hash Functions

In order to achieve IND-FE-CCA security, we will require another kind of PHFs: *tag-based projective hash functions* [1]. In this section, we first define this new tool, as well as the properties we need for our construction. Then we show tag-based PHFs satisfying these properties based on the same 3 examples as previously: DDH, MDDH and DCR.

[7] We recall that the length ℓ of the vectors is assumed to be polynomial in κ.

As both a FE-CPA-friendly PHF and a FE-CCA-friendly PHF are used in our constructions of IND-FE-CCA inner-product functional encryption scheme in Sect. 6, we distinguish the two PHFs by adding a dagger to all the symbols defining the latter PHF. Both PHFs will be used on the same subset membership problem **P**.

5.1 Tag-Based Projective Hash Function

A tag-based projective hash function [1] is defined as a PHF, except that $\mathsf{hash}^\dagger$ and $\mathsf{projhash}^\dagger$ take an additional input (in some efficiently recognizable set $\mathcal{T}$) called a tag τ. We suppose that we can efficiently uniquely encode any 2κ-bit string as a tag τ, as a tag is usually the output of a collision-resistant hash-function. In our constructions, $\mathcal{T}$ is $\mathbb{Z}_M$ for some large integer M.

Definition 21 (Tag-based Projective Hash Function [1]). *Let* **P** *be a subset membership problem, specifying an ensemble* $(I_\ell)_{\ell \geq 0}$ *of instance distributions. A tag-based projective hash function for* **P** *is a tuple* $\mathsf{PHF}^\dagger = (\mathsf{hashkg}^\dagger, \mathsf{projkg}^\dagger, \mathsf{hash}^\dagger, \mathsf{projhash}^\dagger)$ *of four probabilistic polynomial time algorithms:*

– $\mathsf{hashkg}^\dagger(\Lambda)$ *generates a hashing key* $\mathsf{hk}^\dagger$ *in some set* $\mathcal{K}^\dagger$ *for the instance* $\Lambda = \Lambda[\mathcal{X}, \mathcal{L}, \mathcal{W}, \varrho]$,
– $\mathsf{projkg}^\dagger(\mathsf{hk}^\dagger)$ *(deterministically) derives from the hashing key* $\mathsf{hk}^\dagger$ *a projection key* $\mathfrak{hp}^\dagger$ *from the set* $\mathcal{K}_{\mathsf{hp}}$ *of possible projection keys,*
– $\mathsf{hash}^\dagger(\mathsf{hk}^\dagger, \mathfrak{b}, \tau)$ *(deterministically) computes the hash value* $\mathfrak{H}^\dagger$ *(in some efficiently recognizable set* Π*), of* $\mathfrak{b} \in \mathcal{X}$ *under* $\mathsf{hk}^\dagger \in \mathcal{K}^\dagger$*, for the tag* $\tau \in \mathcal{T}$*,*
– $\mathsf{projhash}^\dagger(\mathfrak{hp}^\dagger, \mathfrak{b}, w, \tau)$ *(deterministically) computes the projected hash value* $\mathfrak{pH}^\dagger$ *of* $\mathfrak{b} \in \mathcal{L}$ *using a witness* $w \in \mathcal{W}$*, for the tag* $\tau \in \mathcal{T}$*.*

It has to satisfy the following correctness property:

– *For any instance* Λ*, for any* $\mathfrak{b} \in \mathcal{X}$ *and* $w \in \mathcal{W}$*, s.t.* $(\mathfrak{b}, w) \in \varrho$*, for any hashing key* $\mathsf{hk}^\dagger \in \mathcal{K}^\dagger$*, for any tag* $\tau \in \mathcal{T}$*, if* $\mathfrak{hp}^\dagger \leftarrow \mathsf{projkg}^\dagger(\mathsf{hk}^\dagger)$*, then:*

$$\mathsf{hash}^\dagger(\mathsf{hk}^\dagger, \mathfrak{b}, \tau) = \mathsf{projhash}^\dagger(\mathfrak{hp}^\dagger, \mathfrak{b}, w, \tau) \ .$$

The notions of key homomorphism and projection key homomorphism can be adapted to tag-based PHFs in a straightforward way (key homomorphism has to hold for any tag $\tau \in \mathcal{T}$).

In the sequel, we sometimes omit the term "tag-based" when it is clear from context.

5.2 2-Universality

We now recall the notion of *2-universality*, first introduced by Cramer and Shoup in [14], in order to ensure non-malleability. This will not be directly required by the tag-based PHF we use in the construction, but by a slight modification on it that will be used during the proof. It will ensure that decryption queries made by the adversary do not leak too much information.

Definition 22 (2-universality). *A key-homomorphic tag-based projective hash function* $\mathsf{PHF}^\dagger = (\mathsf{hashkg}^\dagger, \mathsf{projkg}^\dagger, \mathsf{hash}^\dagger, \mathsf{projhash}^\dagger)$ *for a subset membership problem* $\mathbf{P}$ *is* $\varepsilon_{2\mathrm{u}}^\dagger$-2-universal *if for any instance* Λ*, for any* $\mathfrak{b} \in \mathcal{X}$ *and* $\mathfrak{b}' \in \mathcal{X} \setminus \mathcal{L}$*, for any distinct tags* $\tau, \tau' \in \mathcal{T}$*, for any* $\mathfrak{hp}^\dagger \in \mathcal{K}_{\mathsf{hp}}$*, and for any* $\mathfrak{H}^\dagger \in \Pi$*,* $\tilde{\mathfrak{H}}^\dagger \in \Pi$*:*

$$\Pr_{\mathsf{hk}^\dagger}\left[\mathfrak{H}^\dagger = \mathsf{hash}^\dagger(\mathsf{hk}^\dagger, \mathfrak{b}, \tau) \ \wedge\ \mathfrak{H}'^\dagger = \mathsf{hash}^\dagger(\mathsf{hk}^\dagger, \mathfrak{b}', \tau') \ \wedge\ \mathfrak{hp}^\dagger = \mathsf{projkg}^\dagger(\mathsf{hk}^\dagger)\right]$$
$$\leq \varepsilon_{2\mathrm{u}}^\dagger \cdot \Pr_{\mathsf{hk}^\dagger}\left[\mathfrak{H}^\dagger = \mathsf{hash}^\dagger(\mathsf{hk}^\dagger, \mathfrak{b}, \tau) \ \wedge\ \mathfrak{hp}^\dagger = \mathsf{projkg}^\dagger(\mathsf{hk}^\dagger)\right] \ ,$$

where probabilities are taken over $\mathsf{hk}^\dagger \leftarrow_r \mathsf{hashkg}^\dagger(\Lambda)$*. The PHF is* 2-universal *if it is* $\varepsilon_{2\mathrm{u}}^\dagger(\kappa)$*-2-universal for some negligible function* $\varepsilon_{2\mathrm{u}}^\dagger(\kappa)$*.*

In our generic construction, we will not require the PHF used in the construction to be 2-universal, but a variant of it where $\mathsf{hashkg}^\dagger$ is replaced by some other (not necessarily polynomial time) algorithm.

5.3 Universal Translation Indistinguishability

We also need one last statistical property to conclude the proof, as in the IND-FE-CPA case: *universal translation indistinguishability*. It is a strengthening of the previous translation indistinguishability in the sense that the algorithm defining the translation has to be the same for all words.

Definition 23 (Universal translation indistinguishability). *A key-homomorphic tag-based projective hash function* $\mathsf{PHF}^\dagger = (\mathsf{hashkg}^\dagger, \mathsf{projkg}^\dagger, \mathsf{hash}^\dagger, \mathsf{projhash}^\dagger)$ *is* $(\mathsf{hashkg}'^\dagger, M_z, \varepsilon_{\mathrm{uti}}^\dagger)$-universally-translation-indistinguishable *for a (not necessarily polynomial time) algorithm* $\mathsf{hashkg}'^\dagger$ *taking as input* Λ *and outputting a hashing key* $\mathsf{hk}^\dagger$ *in some set* $\mathcal{K}'^{*\dagger} \subseteq \mathcal{K}$*, and for a positive integer* M_z*, if for any integer* z *such that* $|z| \leq M_z$*,*

$$\mathrm{SD}(\mathsf{hashkg}^\dagger(\Lambda), \mathsf{hashkg}^\dagger(\Lambda) + z \cdot \mathsf{hashkg}'^\dagger(\Lambda)) \leq \varepsilon_{\mathrm{uti}}^\dagger \ .$$

Important Particular Case: Key Uniformity. For many key-homomorphic tag-based PHFs, the output of $\mathsf{hashkg}^\dagger$ is actually uniform over the group $\mathcal{K}^\dagger$. In this case, as for translation indistinguishability (Lem. 11), the PHF is automatically $(\mathsf{hashkg}'^\dagger, \cdot, 0)$-universally-translation-indistinguishable, for $\mathsf{hashkg}'^\dagger = \mathsf{hashkg}^\dagger$. More formally, we have the following lemma.

Lemma 24. *Let* $\mathsf{PHF}^\dagger = (\mathsf{hashkg}^\dagger, \mathsf{projkg}^\dagger, \mathsf{hash}^\dagger, \mathsf{projhash}^\dagger)$ *be a key-homomorphic tag-based PHF such that the distribution of* $\mathsf{hashkg}^\dagger(\Lambda)$ *is uniform over* $\mathcal{K}^\dagger$*. Let* M_z *be a positive integer. Then* PHF *is* $(\mathsf{hashkg}^\dagger, M_z, 0)$*-universally-translation-indistinguishable.*

Proof. Both $\mathsf{hashkg}^\dagger(\Lambda)$ and $\mathsf{hashkg}^\dagger(\Lambda) + z \cdot \mathsf{hashkg}^\dagger(\Lambda)$ are uniform group elements in $\mathcal{K}^\dagger$. □

5.4 FE-CCA Friendliness

In the following, we regroup the properties we need under the *FE-CCA friendliness* property. It is used as a shorthand for the sake of readability and regroups projection key homomorphism, universal translation indistinguishability, and 2-universality on a slight modification of the PHF.

Definition 25 (FE-CCA Friendliness). *A tag-based projective hash function* $\mathsf{PHF}^\dagger = (\mathsf{hashkg}^\dagger, \mathsf{projkg}^\dagger, \mathsf{hash}^\dagger, \mathsf{projhash}^\dagger)$ *is* $(\mathsf{hashkg}'^\dagger, \varSigma^\dagger, \varepsilon^\dagger_{\text{2u}}, M_z, \varepsilon^\dagger_{\text{uti}})$-FE-CCA-friendly *for a (not necessarily polynomial time) algorithm* $\mathsf{hashkg}'^\dagger$ *taking as input* $\varLambda$ *and outputting a hashing key* $\mathsf{hk}^\dagger$ *in some set* $\mathcal{K}'^{*\dagger} \subseteq \mathcal{K}$*, and for a positive integer* M_z*, for a subset* $\varSigma^\dagger$ *of* $\mathbb{Z}$*, and for a positive integer* M_z*, if* $\mathsf{PHF}^\dagger$ *is key-homomorphic, projection-key-homomorphic,* $(\mathsf{hashkg}'^\dagger, M_z, \varepsilon^\dagger_{\text{uti}})$*-universally-translation-indistinguishable and if for any* $t \in \varSigma^\dagger$*, the PHF* $(t \cdot \mathsf{hashkg}'^\dagger, \mathsf{projkg}^\dagger, \mathsf{hash}^\dagger, \mathsf{projhash}^\dagger)$ *is* $\varepsilon^\dagger_{\text{2u}}$*-2-universal, where the algorithm* $t \cdot \mathsf{hashkg}'^\dagger$ *runs* $\mathsf{hashkg}'^\dagger$ *and multiplies the output by* t*.*

Important Particular Case: Key Uniformity. For many key-homomorphic PHFs, the output of $\mathsf{hashkg}^\dagger$ is actually uniform over the group $\mathcal{K}^\dagger$. In this case, we have the following lemma which proves FE-CCA friendliness from 2-universality.

Lemma 26. *Let* $\mathsf{PHF}^\dagger = (\mathsf{hashkg}^\dagger, \mathsf{projkg}^\dagger, \mathsf{hash}^\dagger, \mathsf{projhash}^\dagger)$ *be a* $\varepsilon^\dagger_{\text{2u}}$*-2-universal tag-based PHF such that the distribution of* $\mathsf{hashkg}^\dagger(\varLambda)$ *is uniform over* $\mathcal{K}^\dagger$*. Then for any* $t \in \mathbb{Z}$*,* $(t \cdot \mathsf{hashkg}^\dagger, \mathsf{projkg}^\dagger, \mathsf{hash}^\dagger, \mathsf{projhash}^\dagger)$ *is* $\varepsilon^\dagger_{\text{2u}}$*-2-universal.*

Proof. Since $\mathsf{hashkg}^\dagger(\varLambda)$ is uniformly distributed, $t \cdot \mathsf{hashkg}^\dagger(\varLambda)$ is as well, so both schemes are equal. □

5.5 Examples

2-universal tag-based PHFs can be constructed from diverse groups, as in [14]. All the constructions in [14] are key-homomorphic and projection-key-homomorphic. And for well-chosen parameters, they actually are FE-CCA-friendly. Let us now describe these FE-CCA-friendly constructions for our three usual example subset membership problems: DDH, MDDH, and DCRA.

DDH. Let $\mathbb{G}$ be a cyclic group of prime order q, let $\mathcal{X} = \mathbb{G}^2$, let $\mathcal{L}$ be the subgroup of $\mathcal{X}$ generated by $\mathfrak{g} = (\mathfrak{g}_1, \mathfrak{g}_2)^\intercal \in \mathbb{G}^2$, where $\mathfrak{g}_i$ are random generators of $\mathbb{G}^*$. A witness $w \in \mathcal{W} = \mathbb{Z}_q$ for $\mathfrak{b} \in \mathcal{L}$ is such that $\mathfrak{b} = w \cdot \mathfrak{g}$. We set $\varLambda = (\mathbb{G}, \mathfrak{g})$.

We first recall the following 2-universal hash from [1]:

Tag set: $\mathcal{T} = \mathbb{Z}_q$,
$\mathsf{hashkg}^\dagger(\varLambda)$**:** output $\mathbf{hk}^\dagger \leftarrow_r \mathbb{Z}_q^4 =: \mathcal{K}$,
$\mathsf{projkg}^\dagger(\mathbf{hk}^\dagger)$**:** output $\mathfrak{hp}^\dagger \leftarrow \left(\begin{smallmatrix} \mathfrak{g} & \mathbf{0} \\ \mathbf{0} & \mathfrak{g} \end{smallmatrix}\right)^\intercal \cdot \mathbf{hk}^\dagger \in \mathbb{G}^2 =: \mathcal{K}_{\mathsf{hp}}$,
$\mathsf{hash}^\dagger(\mathbf{hk}^\dagger, \mathfrak{b}, \tau)$**:** output $\mathfrak{H}^\dagger \leftarrow \mathbf{hk}^{\dagger\intercal} \cdot \left(\begin{smallmatrix} \mathfrak{b} \\ \tau \cdot \mathfrak{b} \end{smallmatrix}\right) \in \mathbb{G} =: \Pi$;
$\mathsf{projhash}^\dagger(\mathfrak{hp}^\dagger, \mathfrak{b}, w, \tau)$**:** output $\mathfrak{pH}^\dagger \leftarrow \mathfrak{hp}^{\dagger\intercal} \cdot \left(\begin{smallmatrix} w \\ \tau \cdot w \end{smallmatrix}\right) \in \mathbb{G} = \Pi$.

We prove the following lemma in the full version.

Lemma 27. *Using above notation, let* $\mathsf{hashkg}'^{\dagger} = \mathsf{hashkg}^{\dagger}$*,* $\Sigma^{\dagger} = \mathbb{Z}_q$*,* $\varepsilon_{\mathrm{2u}}^{\dagger} = 1/q$*,* M_z *be a positive integer, and* $\varepsilon_{\mathrm{uti}}^{\dagger} = 0$*. Then, the PHF described above is a* $(\mathsf{hashkg}'^{\dagger}, \Sigma^{\dagger}, \varepsilon_{\mathrm{2u}}^{\dagger}, M_z, \varepsilon_{\mathrm{uti}}^{\dagger})$*-FE-CCA-friendly.*

We use a slight extension of this PHF because we need an exponentially small security parameter $\varepsilon_{\mathrm{2u}}^{\dagger}$, due our security reduction. The following PHF can be seen as repeating ν times the PHF of Lem. 27:

Tag set: $\mathcal{T} = \mathbb{Z}_q$,
$\mathsf{hashkg}^{\dagger}(\Lambda)$**:** output $\mathbf{hk}^{\dagger} \leftarrow_r \mathbb{Z}_q^{4\times\nu} =: \mathcal{K}$;
$\mathsf{projkg}^{\dagger}(\mathbf{hk}^{\dagger})$**:** output $\mathbf{\mathfrak{hp}}^{\dagger} \leftarrow \left(\begin{smallmatrix} \mathbf{\mathfrak{g}} & \mathbf{0} \\ \mathbf{0} & \mathbf{\mathfrak{g}} \end{smallmatrix}\right) \cdot \mathbf{hk}^{\dagger} \in \mathbb{G}^{2\times\nu} =: \mathcal{K}_{\mathsf{hp}}$;
$\mathsf{hash}^{\dagger}(\mathbf{hk}^{\dagger}, \mathbf{\mathfrak{b}}, \tau)$**:** output $\mathbf{\mathfrak{H}}^{\dagger} \leftarrow \mathbf{hk}^{\dagger\intercal} \cdot \left(\begin{smallmatrix} \mathbf{\mathfrak{b}} \\ \tau\cdot\mathbf{\mathfrak{b}} \end{smallmatrix}\right) \in \mathbb{G}^{\nu} =: \Pi$;
$\mathsf{projhash}^{\dagger}(\mathbf{\mathfrak{hp}}^{\dagger}, \mathbf{\mathfrak{b}}, w, \tau)$**:** output $\mathbf{\mathfrak{pH}}^{\dagger} \leftarrow \left(\begin{smallmatrix} w \\ \tau\cdot w \end{smallmatrix}\right)^{\intercal} \cdot \mathbf{\mathfrak{hp}}^{\dagger} \in \mathbb{G}^{\nu} = \Pi$.

We prove the following lemma in the full version.

Lemma 28. *Using above notation, let* $\mathsf{hashkg}'^{\dagger} = \mathsf{hashkg}^{\dagger}$*,* $\Sigma^{\dagger} = \mathbb{Z}_q$*,* $\varepsilon_{\mathrm{2u}}^{\dagger} = 1/q^{\nu}$*,* M_z *be a positive integer, and* $\varepsilon_{\mathrm{uti}}^{\dagger} = 0$*. Then, the PHF described above is a* $(\mathsf{hashkg}'^{\dagger}, \Sigma^{\dagger}, \varepsilon_{\mathrm{2u}}^{\dagger}, M_z, \varepsilon_{\mathrm{ti}})$*-FE-CCA-friendly.*

MDDH. The previous construction can be extended in a straightforward way to any MDDH-based subset membership problem in a straightforward way, similar to what is done for our FE-CPA-friendly construction in Sect. 3.5 in page 3.5.

DCR. Let $\Lambda = (N, s, \mathfrak{g}, \mathfrak{g}_{\perp})$ be defined as in the DCR subsubsection of Sect. 2.1 on page 7. We have: $\mathbb{G} = \mathcal{X} = J_{N^{s+1}} \cong G_{N^s} \oplus G_{N'} \oplus T$, $\mathcal{L} = G_{N'}$, and $\bar{\mathcal{L}} = \mathcal{L} + \mathfrak{g}_{\perp}$. The element $\mathfrak{g}$ is a generator of $\mathcal{L}$, while $\mathfrak{g}_{\perp}$ is a generator of G_{N^s}. We recall that we use additive notation for the group $\mathbb{G}$.

We define a PHF as follows:

Tag set: $\mathcal{T} = \{0, \dots, \lfloor N/2 \rfloor\} \subseteq \mathbb{Z}_{N'}$
$\mathsf{hashkg}^{\dagger}(\Lambda)$**:** output $\mathbf{hk}^{\dagger} \leftarrow_r \{0, \dots, \lfloor \nu M^{\dagger} N^{s+1}/2 \rfloor\}^{2\times\nu} =: \mathcal{K}^{*} \subseteq \mathbb{Z}^{2\times\nu} =: \mathcal{K}$,
where $M^{\dagger}$ is a positive integer and is a parameter of the scheme,
$\mathsf{projkg}^{\dagger}(\mathsf{hk}^{\dagger})$**:** output $\mathbf{\mathfrak{hp}}^{\dagger} \leftarrow \left(\begin{smallmatrix} \mathfrak{g} & 0 \\ 0 & \mathfrak{g} \end{smallmatrix}\right)^{\intercal} \cdot \mathbf{hk}^{\dagger} \in \mathbb{G}^{2\times\nu} =: \mathcal{K}_{\mathsf{hp}}$;
$\mathsf{hash}^{\dagger}(\mathbf{hk}^{\dagger}, \mathfrak{b}, \tau)$**:** output $\mathfrak{H}^{\dagger} \leftarrow \mathbf{hk}^{\dagger\intercal} \cdot \left(\begin{smallmatrix} \mathfrak{b} \\ \tau\cdot\mathfrak{b} \end{smallmatrix}\right) \in \mathbb{G}^{\nu} =: \Pi$;
$\mathsf{projhash}^{\dagger}(\mathbf{\mathfrak{hp}}^{\dagger}, \mathfrak{b}, w, \tau)$**:** output $\mathfrak{pH}^{\dagger} \leftarrow \mathbf{\mathfrak{hp}}^{\dagger\intercal} \cdot \left(\begin{smallmatrix} w \\ \tau\cdot w \end{smallmatrix}\right) \in \mathbb{G}^{\nu} = \Pi$.

We prove the following lemma in the full version.

Lemma 29. *Using above notation,* $\Sigma^{\dagger} = \{-N^s + 1, \dots, N^s - 1\} \setminus \{0\}$*,* $\varepsilon_{\mathrm{2u}}^{\dagger} = 1/2^{\nu}$*,* M_z *be a positive integer, and* $\varepsilon_{\mathrm{uti}}^{\dagger} = M_z/M^{\dagger}$*. Define in addition the following algorithm:*

$\mathsf{hashkg}'^{\dagger}(\Lambda)$**:** *output* $\mathbf{hk}^{\dagger} \leftarrow_r \mathbb{Z}_{N'N^s}^{2\times\nu} = \mathcal{K}^{*\dagger}$.

Then, the PHF described above is a $(\mathsf{hashkg}'^{\dagger}, \Sigma^{\dagger}, \varepsilon_{\mathrm{2u}}^{\dagger}, M_z, \varepsilon_{\mathrm{uti}}^{\dagger})$*-FE-CCA-friendly.*

6 IND-FE-CCA Inner-Product Functional Encryption

In this section, we construct IND-FE-CCA inner-product functional encryption from FE-CPA-friendly PHFs and FE-CCA-friendly PHFs. For the sake of readability, we split our construction into two parts: we first show how to construct a CCA secure tag-based variant of inner-product functional encryption from PHFs with the right properties. Then we show how to construct a non tag-based functional encryption that reaches CCA security from the tag-based variant.

6.1 Tag-Based Functional Encryption

We now define tag-based functional encryption. It is an adaptation from the concept of tag-based encryption [24] to the context of functional encryption.

Definition 30. *A* tag-based functional encryption scheme for functionality $\mathcal{F}$ *is a tuple* $\mathsf{TBFE} = (\mathsf{setup}, \mathsf{keygen}, \mathsf{enc}, \mathsf{dec})$ *of four probabilistic polynomial time algorithms:*

$\mathsf{setup}(1^\kappa, \ell)$**:** *first generates system parameters* pp*, and then returns a master secret and public key pair* (msk, mpk)*, where both* msk *and* mpk *also contain* pp*,*

$\mathsf{keygen}_{\mathrm{msk}}(y \in \mathcal{Y})$**:** *given a master secret key* msk *and* y*, returns a partial secret key* $\mathrm{msk}_y = (\mathrm{pp}, k_y, y)$*,*

$\mathsf{enc}_{\mathrm{mpk},\tau}(z \in \mathcal{Z})$**:** *given a master public key* mpk*, a tag* τ*, and a plaintext* z*, returns a ciphertext* c*,*

$\mathsf{dec}_{\mathrm{msk}_y,\tau}(c)$**:** *given a partial secret key* msk_y*, a tag* τ*, and a ciphertext* c*, returns* $S \in \Sigma \cup \{\perp\}$*.*

TBFE must be *complete*, in the sense that if (y, z) is in the domain of $\mathcal{F}$, and τ is a tag, then for all $(\mathrm{msk}, \mathrm{mpk}) \leftarrow \mathsf{setup}(1^\kappa)$, $\mathrm{msk}_y \leftarrow \mathsf{keygen}_{\mathrm{msk}}(y)$, and $c \leftarrow_r \mathsf{enc}_{\mathrm{mpk},\tau}(z; r)$, it holds that $\mathsf{dec}_{\mathrm{msk}_y,\tau}(c) = \mathcal{F}(y, z)$.

In the following definition, we have highlighted differences with the IND-FE-CCA definition, Def. 4.

Definition 31 (IND-TBFE-CCA Security). *A tag-based functional encryption scheme* $\mathsf{TBFE} = (\mathsf{setup}, \mathsf{keygen}, \mathsf{enc}, \mathsf{dec})$ *is* IND-TBFE-CCA secure *(or,* secure against chosen ciphertext attacks*), if no probabilistic polynomial time adversary* $\mathcal{A}$ *has a non-negligible advantage in the following game:*

1. *The challenger sets* $(\mathrm{msk}, \mathrm{mpk}) \leftarrow \mathsf{setup}(1^\kappa, \ell)$ *and sends* mpk *to* $\mathcal{A}$*.*
2. $\mathcal{A}$ *makes adaptive secret key and decryption queries to the challenger. At each secret key query,* $\mathcal{A}$ *chooses* $y \in \mathcal{Y}$ *and obtains* $\mathrm{msk}_y = (\mathrm{pp}, k_y, y) \leftarrow \mathsf{keygen}_{\mathrm{msk}}(y)$*. At each decryption query,* $\mathcal{A}$ *chooses a ciphertext* c'*, a tag* τ'*, and* $y \in \mathcal{Y}$*, then the challenger computes* $\mathrm{msk}_y = (\mathrm{pp}, k_y, y) \leftarrow \mathsf{keygen}_{\mathrm{msk}}(y)$ *and sends back* $\mathsf{dec}_{\mathrm{msk}_y,\tau'}(c')$ *to* $\mathcal{A}$*. Let* y_i *be the ith queried secret key.*
3. $\mathcal{A}$ *chooses a tag* τ*, and* $z_0 \neq z_1$ *such that* $\mathcal{F}(y_i, z_0) = \mathcal{F}(y_i, z_1)$ *for all queried* y_i*. She sends* τ*,* z_0*, and* z_1 *to the challenger. The challenger chooses* $\beta \leftarrow_r \{0, 1\}$*, and sends* $c \leftarrow_r \mathsf{enc}_{\mathrm{mpk},\tau}(z_\beta)$ *to* $\mathcal{A}$*.*

4. *$\mathcal{A}$ makes more secret key queries for keys $y_i \in \mathcal{Y}$, with the condition that $\mathcal{F}(y_i, z_0) = \mathcal{F}(y_i, z_1)$, and decryption queries, with the condition that $\tau' \neq \tau$. Let q_{dec} be the number of decryption queries made during the whole game, and let (y_j, τ'_j, c'_j) be the jth decryption query.*
5. *$\mathcal{A}$ outputs a bit $\beta_A \in \{0,1\}$ and wins if $\beta_A = \beta$.*

More precisely, the advantage of $\mathcal{A}$ is defined as

$$\mathsf{Adv}^{\mathsf{ind-tbfe-cca}}_{\mathsf{TBFE},\mathcal{A}}(\kappa) := 2 \cdot |\Pr[\beta_A = \beta] - 1/2| \ .$$

TBFE *is* secure against chosen ciphertext attacks (or, IND-TBFE-CCA secure)*, if* $\mathsf{Adv}^{\mathsf{ind-tbfe-cca}}_{\mathsf{TBFE},\mathcal{A}}$ *is negligible for all probabilistic polynomial time adversaries* Adv*.*

6.2 Generic Construction

Intuition. The core idea of our construction is similar to the one used in the Cramer-Shoup encryption scheme [12,14]: adding a hash value (from a 2-universal PHF) to ensure that the word $\mathfrak{b}$ is in the language $\mathcal{L}$, to our generic IND-FE-CPA construction in Sect. 4.1. Then, at least information-theoretically, the values $\mathsf{hash}(\mathsf{hk}_i, \mathfrak{b})$ used to decrypt a ciphertext $(\mathfrak{b}, \vec{\mathfrak{c}})$ could be computed using only $\mathfrak{hp}_i$ and do not leak any information from hk_i. We can then conclude using the same ideas as in the IND-FE-CPA security proof of our generic construction.

However, this does not work directly, as checking a 2-universal hash value require to know the corresponding hashing key $\mathsf{hk}^\dagger$, and knowing this hashing key enables to fake these hash values. In other words, with the naive scheme described previously, an attacker knowing a secret key for any $\vec{y}$ could then generate a ciphertext with $\mathfrak{b} \notin \mathcal{L}$, but a valid 2-universal hash values. This completely removes the usefulness of the 2-universal hash value.

Our new idea is the following: instead of using only one hash value, we use ℓ such values. The secret key $\mathrm{msk}_{\vec{y}}$ only enables to check that a linear combination (with coefficient $\vec{y}$) of these hash values is valid. This uses the key homomorphism property. Knowing $\mathrm{msk}_{\vec{y}}$ enables to generate hash values that would be accepted by the decryption oracle with $\vec{y}$, and knowing $\mathrm{msk}_{\vec{y}}$ for multiple vectors $\vec{y}$ enables to generate hash values for any vector in the span of these $\vec{y}$. But intuitively, this is not really an issue, as if the attacker already knows $\mathrm{msk}_{\vec{y}}$, calling the decryption oracle for $\vec{y}$ is of no use to him, as he could decrypt the given ciphertext himself. The proof however is more subtle and requires a careful design of hybrid games to deal with adaptivity and the fact that we are working over a ring and not a field. In particular, we cannot directly rely on the notion of span of vectors. Details can be found in the proof.

Construction. We suppose that we have a $(\mathsf{hk}_\perp, \mathfrak{g}_\perp, M_\perp, z, \varepsilon_{\mathrm{ti}})$-FE-CPA-friendly projective hash function $\mathsf{PHF} = (\mathsf{hashkg}, \mathsf{projkg}, \mathsf{hash}, \mathsf{projhash})$ and a $(\mathsf{hashkg'}^\dagger, \Sigma^\dagger, \varepsilon^\dagger_{2\mathrm{u}}, M_z, \varepsilon^\dagger_{\mathrm{uti}})$-FE-CCA-friendly projective hash function $\mathsf{PHF}^\dagger = (\mathsf{hashkg}^\dagger, \mathsf{projkg}^\dagger, \mathsf{hash}^\dagger, \mathsf{projhash}^\dagger)$ for the subset membership problem $\mathbf{P}$. Let $\mathcal{R}$

1. Let $\mathbf{P}$ be a subset membership problem. Let $\mathsf{PHF} = (\mathsf{hashkg}, \mathsf{projkg}, \mathsf{hash}, \mathsf{projhash})$ be a $(\mathsf{hk}_\perp, \mathfrak{g}_\perp, M_\perp, M_z, \varepsilon_{\mathrm{ti}})$-FE-CPA-friendly PHF, and $\mathsf{PHF}^\dagger = (\mathsf{hashkg}^\dagger, \mathsf{projkg}^\dagger, \mathsf{hash}^\dagger, \mathsf{projhash}^\dagger)$ be a $(\mathsf{hashkg}'^\dagger, \Sigma^\dagger, \varepsilon^\dagger_{2\mathrm{u}}, M_z, \varepsilon^\dagger_{\mathrm{uti}})$-FE-CCA-friendly tag-based PHF. We assume that Cond. 1 is satisfied.
2. $\mathsf{setup}(1^\kappa, \ell)$: Sample $\Lambda \leftarrow_r I_\kappa$, and set pp $\leftarrow (\kappa, \ell, \Lambda)$. For $i = 1, \ldots, \ell$, set
$$\mathsf{hk}_i \leftarrow_r \mathsf{hashkg}(\Lambda)\,, \qquad \mathfrak{hp}_i \leftarrow \mathsf{projkg}(\mathsf{hk}_i)\,,$$
$$\mathsf{hk}^\dagger_i \leftarrow_r \mathsf{hashkg}^\dagger(\Lambda)\,, \qquad \mathfrak{hp}^\dagger_i \leftarrow \mathsf{projkg}^\dagger(\mathsf{hk}^\dagger_i)\,,$$
Set msk $\leftarrow (\mathrm{pp}, \vec{\mathsf{hk}}, \vec{\mathsf{hk}}^\dagger)$ and mpk $\leftarrow (\mathrm{pp}, \vec{\mathfrak{hp}}, \vec{\mathfrak{hp}}^\dagger)$, and return (msk, mpk).
3. $\mathsf{keygen}_{\mathrm{msk}}(\vec{y} \in \mathcal{Y})$: Set $\mathsf{hk}_{\vec{y}} \leftarrow \langle \vec{y}, \vec{\mathsf{hk}} \rangle \in \mathcal{K}$ and $\mathsf{hk}^\dagger_{\vec{y}} \leftarrow_r \langle \vec{y}, \vec{\mathsf{hk}}^\dagger \rangle \in \mathcal{K}^\dagger$, and return $\mathrm{msk}_{\vec{y}} \leftarrow (\mathrm{pp}, \mathsf{hk}_{\vec{y}}, \mathsf{hk}^\dagger_{\vec{y}}, \vec{y})$.
4. $\mathsf{enc}_{\mathrm{mpk},\tau}(\vec{z} \in \mathcal{Z})$: Sample a random pair $(\mathfrak{b}, w) \in \varrho$. For $i = 1, \ldots, \ell$, set
$$\mathfrak{c}_i \leftarrow \mathsf{projhash}(\mathfrak{hp}_i, \mathfrak{b}, w) + z_i \cdot \mathfrak{g}_\perp\,, \qquad \mathfrak{c}^\dagger_i \leftarrow \mathsf{projhash}^\dagger(\mathfrak{hp}^\dagger_i, \mathfrak{b}, w, \tau)\,.$$
Return $(\mathfrak{b}, \vec{\mathfrak{c}}, \vec{\mathfrak{c}}^\dagger)$.
5. $\mathsf{dec}_{\mathrm{msk}_{\vec{y}},\tau}(\mathfrak{b}, \vec{\mathfrak{c}}, \vec{\mathfrak{c}}^\dagger)$: Check that $\mathfrak{b} \in \mathcal{X}$, and $\mathfrak{c}_i \in \Pi$ and $\langle \vec{y}, \vec{\mathfrak{c}}^\dagger \rangle = \mathsf{hash}^\dagger(\mathsf{hk}^\dagger_{\vec{y}}, \mathfrak{b}, \tau)$ for $i = 1, \ldots, \ell$; return $\perp$ if any check fails. Set
$$\mathfrak{c}_{\vec{z}} \leftarrow \langle \vec{y}, \vec{\mathfrak{c}} \rangle - \mathsf{hash}(\mathsf{hk}_{\vec{y}}, \mathfrak{b})\ \ .$$
Return $\log_{\mathfrak{g}_\perp} \mathfrak{c}_{\vec{z}}$.

Fig. 2. Generic inner-product tag-based functional encryption $\mathsf{TBFE}_{\mathrm{phf}}$ from a FE-CPA-friendly PHF and a FE-CCA-friendly tag-based PHF

be the ring $\mathbb{Z}$ or $\mathbb{Z}_{M_\perp}$, let ℓ be a positive integer parameter corresponding to the length of the message and key vectors, and let $\mathcal{Y}$ and $\mathcal{Z}$ be two subsets of $\mathcal{R}^\ell$. We always suppose ℓ to be polynomial in the security parameter κ.

We suppose that Cond. 1 is satisfied, in addition to the following new condition.

Condition 2. *Using the above notation:*
1. *if $\mathcal{R} = \mathbb{Z}_{M_\perp}$, the order of any hashing key $\mathsf{hk} \in \mathcal{K}^\dagger$ divides $M_\perp$; and*
2. *for any $\vec{y} \in \mathcal{Y}$ and $\vec{z} \in \mathcal{Z}$, $\langle \vec{y}, \vec{z} \rangle \in \Sigma^\dagger \cup \{0\} \subseteq \mathcal{R}$.*

Security. We have the following security theorem.

Theorem 32. *Let $\mathbf{P}$ be a subset membership problem. Let $\mathsf{PHF} = (\mathsf{hashkg}, \mathsf{projkg}, \mathsf{hash}, \mathsf{projhash})$ be a $(\mathsf{hk}_\perp, \mathfrak{g}_\perp, M_\perp, M_z, \varepsilon_{\mathrm{ti}})$-FE-CPA-friendly PHF. $(\mathsf{hashkg}'^\dagger, \Sigma^\dagger, \varepsilon^\dagger_{2\mathrm{u}}, M_z, \varepsilon^\dagger_{\mathrm{uti}})$-FE-CCA-friendly projective hash function. Then the scheme $\mathsf{TBFE}_{\mathrm{phf}}$ is complete and IND-TBFE-CCA.*

More precisely, if there exists an adversary $\mathcal{A} = \mathcal{A}_{\mathsf{FE}}$ that has advantage $\varepsilon_{\mathcal{A}}$ in breaking the IND-TBFE-CCA security of $\mathsf{TBFE}_{\mathrm{phf}}$, then there exists an attacker $\mathcal{B}$ that runs in approximately the same time and that has advantage $\varepsilon_{\mathcal{B}}$ in breaking the $(\mathcal{L}, \bar{\mathcal{L}})$-indistinguishability, such that

$$\varepsilon_{\mathcal{A}} \leq 2 \cdot \varepsilon_{\mathcal{B}} + \ell \cdot |\varDelta\mathcal{Z}| \cdot (\varepsilon_{\mathrm{ti}} + 2 \cdot \varepsilon^{\dagger}_{\mathrm{uti}}) + 2 \cdot q_{dec} \cdot |\varDelta\mathcal{Z}| \cdot \varepsilon^{\dagger}_{2\mathrm{u}} \,,$$

where q_{dec} is the number of queries to the decryption oracle.

The proof is in the full version.

Remark 33. In addition to the exponential loss $\ell \cdot |\varDelta\mathcal{Z}| \cdot (\varepsilon_{\mathrm{ti}} + 2 \cdot \varepsilon^{\dagger}_{\mathrm{uti}})$ similar to the one for the generic IND-FE-CPA construction (Thm. 16), there is an addition exponential loss in the security proof in the term $2q_{\mathrm{dec}}|\varDelta\mathcal{Z}|\varepsilon^{\dagger}_{2\mathrm{u}}$. We point out however that the resulting requirement that $|\varDelta\mathcal{Z}|\varepsilon^{\dagger}_{2\mathrm{u}}$ is negligible in the security parameter can easily to achieve: given a $\varepsilon^{\dagger}_{2\mathrm{u}}$-2-universal PHF, we can get a $(\varepsilon^{\dagger}_{2\mathrm{u}})^{\nu}$-2-universal PHF, by repeating it ν-times in parallel. This transformation preserves FE-CCA friendliness. Our examples in Sect. 5.5 actually already uses this trick. We emphasize that the resulting key and ciphertext sizes remain polynomial in the security parameter κ, and that we do not rely on complexity leveraging nor subexponential assumptions (see Rk. 17 on page 16).

Furthermore, as for the IND-FE-CPA construction from translation-indistinguishable key-homomorphic PHF in Sect. 4.1, if we only consider a selective version of IND-TBFE-CCA where the adversary announces $\vec{z}_0$ and $\vec{z}_1$ before receiving the public key, then we would not have this factor $|\varDelta\mathcal{Z}|$.

6.3 DDH-Based Instantiation

Let us instantiate the framework with the DDH-based FE-CPA-friendly PHF defined in Sect. 3.5 on page 13, and the DDH-based FE-CCA-friendly tag-based PHF defined in Sect. 5.5 on page 21. We set $\mathcal{R} = \mathbb{Z}_q$ and $M_z = q$ (or any large enough integer). As for the IND-FE-CPA scheme in Sect. 4.2, we need to choose the efficiently recognizable subsets $\mathcal{Y}$ and $\mathcal{Z}$ of $\mathcal{R}^{\ell}$ so that the discrete logarithm of $\langle \vec{y}, \vec{z} \rangle \cdot \mathfrak{g}_{\perp} \in \mathbb{G}$ is efficient to compute, for any $\vec{y} \in \mathcal{Y}$ and $\vec{z} \in \mathcal{Z}$ in order to satisfy Cond. 2. The resulting construction $\mathsf{TBFE}_{\mathrm{ddh}}$ is depicted in Fig. 3 and can be easily extended to use any MDDH-based PHF defined in Sect. 5.5.

Applying Thm. 32, we immediately get the following security theorem.

Theorem 34. *Under the DDH assumption in $\mathbb{G}$, the scheme $\mathsf{TBFE}_{\mathrm{ddh}}$ depicted in Fig. 3 is complete and IND-TBFE-CCA.*

More precisely, if there exists an attacker $\mathcal{A} = \mathcal{A}_{\mathsf{TBFE}}$ that has advantage $\varepsilon_{\mathcal{A}}$ in breaking the IND-TBFE-CCA security of $\mathsf{TBFE}_{\mathrm{ddh}}$, then there exists an attacker $\mathcal{B}$ that runs in approximately the same time and that has advantage $\varepsilon_{\mathcal{B}}$ in breaking the DDH assumption, such that $\varepsilon_{\mathcal{A}} \leq 2 \cdot \varepsilon_{\mathcal{B}} + 2 \cdot q_{dec} \cdot q^{\ell-\nu}$.

In particular, setting $\nu = \ell + 1$, we have the following bound: $\varepsilon_{\mathcal{A}} \leq 2 \cdot \varepsilon_{\mathcal{B}} + 2 \cdot \frac{q_{\mathrm{dec}}}{q}$.

1. Let $\mathbb{G}$ be a cyclic group of prime order q, $\mathfrak{g}_\perp$ a generator of $\mathbb{G}$.
2. $\mathsf{setup}(1^\kappa, \ell)$: Choose $\mathbf{\mathfrak{g}} \leftarrow_r \mathbb{G}^2$. Set $\mathrm{pp} = (\kappa, \ell, \mathbf{\mathfrak{g}})$. For $i = 1, \ldots, \ell$, set
$$\begin{aligned} \mathbf{hk}_i &\leftarrow_r \mathbb{Z}_q^2 \,, & \mathfrak{hp}_i &\leftarrow \mathbf{hk}_i^{\mathsf{T}} \cdot \mathbf{\mathfrak{g}} \in \mathbb{G}\,, \\ \mathbf{hk}_i^\dagger &\leftarrow_r \mathbb{Z}_q^{4\times\nu}\,, & \mathbf{\mathfrak{hp}}_i^\dagger &\leftarrow \left(\begin{smallmatrix} \mathbf{\mathfrak{g}} & \mathbf{0} \\ \mathbf{0} & \mathbf{\mathfrak{g}} \end{smallmatrix}\right)^{\mathsf{T}} \cdot \mathbf{hk}_i^\dagger \in \mathbb{G}^{2\times nu}\,. \end{aligned}$$
Set $\mathrm{msk} \leftarrow (\mathrm{pp}, \vec{\mathbf{hk}} \in \mathbb{Z}_q^\ell, \vec{\mathbf{hk}}^\dagger \in (\mathbb{Z}_q^{4\times\nu})^\ell)$ and $\mathrm{mpk} \leftarrow (\mathrm{pp}, \vec{\mathfrak{hp}} \in \mathbb{G}, \vec{\mathbf{\mathfrak{hp}}}^\dagger \in (\mathbb{G}^{2\times\nu})^\ell)$. Return $(\mathrm{msk}, \mathrm{mpk})$.
3. $\mathsf{keygen}_{\mathrm{msk}}(\vec{y} \in \mathbb{Z}_q^\ell)$: Set $\mathbf{hk}_{\vec{y}} \leftarrow \langle \vec{y}, \vec{\mathbf{hk}} \rangle \in \mathbb{Z}_q^2$ and $\mathbf{hk}_{\vec{y}}^\dagger \leftarrow \langle \vec{y}, \vec{\mathbf{hk}}^\dagger \rangle \in \mathbb{Z}_q^{4\times\nu}$. Return $\mathrm{msk}_{\vec{y}} \leftarrow (\mathrm{pp}, \mathbf{hk}_{\vec{y}}, \mathbf{hk}_{\vec{y}}^\dagger, \vec{y})$.
4. $\mathsf{enc}_{\mathrm{mpk},\tau}(\vec{z} \in \mathbb{Z}_q^\ell)$: Pick $r \leftarrow_r \mathbb{Z}_q$ and set $\mathbf{\mathfrak{b}} \leftarrow r \cdot \mathbf{\mathfrak{g}} \in \mathbb{G}^2$.
For $i = 1, \ldots, \ell$, set
$$\mathfrak{c}_i \leftarrow z_i \cdot \mathfrak{g}_\perp + r \cdot \mathfrak{hp}_i \in \mathbb{G}\,, \qquad \mathbf{\mathfrak{c}}_i^\dagger \leftarrow \mathbf{\mathfrak{hp}}_i^\dagger \cdot \left(\begin{smallmatrix} r \\ \tau \cdot r \end{smallmatrix}\right) \in \mathbb{G}^\nu \;\;.$$
Return $(\mathbf{\mathfrak{b}}, \vec{\mathfrak{c}} \in \mathbb{G}^\ell, \vec{\mathbf{\mathfrak{c}}}^\dagger \in (\mathbb{G}^\nu)^\ell)$.
5. $\mathsf{dec}_{\mathrm{msk}_{\vec{y}},\tau}(\mathbf{\mathfrak{b}}, \vec{\mathfrak{c}}, \vec{\mathbf{\mathfrak{c}}}^\dagger)$: Check that $\langle \vec{y}, \vec{\mathbf{\mathfrak{c}}}^\dagger \rangle = \mathbf{hk}_{\vec{y}}^{\dagger\mathsf{T}} \cdot \left(\begin{smallmatrix} \mathbf{\mathfrak{b}} \\ \tau\cdot\mathbf{\mathfrak{b}} \end{smallmatrix}\right)$; return $\perp$ if it fails. Set
$$\mathfrak{c}_{\vec{z}} \leftarrow \langle \vec{y}, \vec{\mathfrak{c}} \rangle - \mathbf{hk}_{\vec{y}}^{\mathsf{T}} \cdot \mathbf{\mathfrak{b}} \in \mathbb{G}.$$
Return $\log_{\mathfrak{g}_\perp} \mathfrak{c}_{\vec{z}}$.

Fig. 3. DDH-based inner-product tag-based functional encryption $\mathsf{TBFE}_{\mathrm{ddh}}$

6.4 DCR-Based Instantiations

Let us now instantiate the framework with the DCR-based FE-CPA-friendly PHF defined in Sect. 3.5 on page 14, and the DDH-based FE-CCA-friendly tag-based PHF defined in Sect. 5.5 on page 22. We use the same parameters as for the IND-FE-CPA scheme in Sect. 4.3. The resulting construction $\mathsf{TBFE}_{\mathrm{dcr}}$ is depicted in Fig. 4. We switch back to the multiplicative notation so that the scheme looks more familiar.

Applying Thm. 32 and Lem. 3, we immediately get the following security theorem.

Theorem 35. *Under the DCR assumption, the scheme* $\mathsf{TBFE}_{\mathrm{dcr}}$ *depicted in Fig. 4 is complete and IND-TBFE-CCA.*

More precisely, if there exists an attacker $\mathcal{A} = \mathcal{A}_{\mathsf{TBFE}}$ *that has advantage* $\varepsilon_{\mathcal{A}}$ *in breaking the IND-TBFE-CCA security of* $\mathsf{TBFE}_{\mathrm{ddh}}$*, then there exists an attacker* $\mathcal{B}$ *that runs in approximately the same time and that has advantage* $\varepsilon_{\mathcal{B}}$ *in breaking the DCR assumption, such that* $\varepsilon_{\mathcal{A}} \leq 4s \cdot \varepsilon_{\mathcal{B}} + 16/\operatorname{spf}(N) + \ell \cdot |\Delta\mathcal{Z}| \cdot M_z \cdot (1/M + 2/M^\dagger) + 2 \cdot q_{dec} \cdot |\Delta\mathcal{Z}|/2^\nu$ *.*

Using parameters from Ex. 19 and setting $M^\dagger = M$ and $\nu \geq \kappa + \log_2(2 \cdot q_{\mathrm{dec}} \cdot |\Delta\mathcal{Z}|) = O(\mathrm{poly}(\kappa))$, we have the following security bound: $\varepsilon_{\mathcal{A}} \leq 4s \cdot \varepsilon_{\mathcal{B}} +$

1. Let $N = pq$ be a product of two λ-bit random safe primes. We suppose that Cond. 2 is satisfied. Let $u \leftarrow_r \mathbb{Z}^*_{N^{s+1}}$, $\mathfrak{g} \leftarrow u^{2N^s}$, and $\mathfrak{g}_\perp \leftarrow 1 + N$.
2. $\mathsf{setup}(1^\kappa, \ell)$: Set pp $\leftarrow (\kappa, \ell, N, \mathfrak{g})$. For $i = 1, \dots, \ell$, $j = 1, 2$, and $k = 1, \dots, \nu$, set
$$\mathsf{hk}_i \leftarrow_r \{0, \dots, \lfloor MN^{s+1}/4 \rfloor\}, \qquad \mathfrak{hp}_i \leftarrow \mathfrak{g}^{\mathsf{hk}_i} \in \mathbb{G},$$
$$\mathsf{hk}^\dagger_{k,i,j} \leftarrow_r \{0, \dots, \lfloor \nu M^\dagger N^{s+1}/2 \rfloor\}, \qquad \mathfrak{hp}^\dagger_{k,i,j} \leftarrow \mathfrak{g}^{\mathsf{hk}^\dagger_{k,i,j}} \in \mathbb{G}\ .$$
Set msk $\leftarrow$ (pp, $\vec{\mathsf{hk}} \in \mathbb{Z}^\ell$, $\vec{\mathbf{hk}}^\dagger \in (\mathbb{Z}^{2\times\nu})^\ell$) and mpk $\leftarrow$ (pp, $\vec{\mathfrak{hp}} \in \mathbb{G}^\ell$, $\vec{\mathfrak{hp}}^\dagger \in (\mathbb{G}^{2\times\nu})^\ell$). Return (msk, mpk).
3. $\mathsf{keygen}_{\mathrm{msk}}(\vec{y} \in \mathcal{Y})$: Set $\mathsf{hk}_{\vec{y}} \leftarrow \langle \vec{y}, \vec{\mathsf{hk}} \rangle$ over $\mathbb{Z}$ and $\mathbf{hk}^\dagger_{\vec{y}} \leftarrow \langle \vec{y}, \vec{\mathbf{hk}}^\dagger \rangle$ over $\mathbb{Z}^{2\times\nu}$. Return $\mathrm{msk}_{\vec{y}} \leftarrow (\mathrm{pp}, \mathsf{hk}_{\vec{y}}, \mathbf{hk}^\dagger_{\vec{y}}, \vec{y})$.
4. $\mathsf{enc}_{\mathrm{mpk},\tau}(\vec{z} \in \mathcal{Z})$: Sample $r \leftarrow_r \{0, \dots, \lfloor N/4 \rfloor\}$. Set $\mathfrak{b} \leftarrow \mathfrak{g}^r$. For $i = 1, \dots, \ell$ and $k = 1, \dots, \nu$, set
$$\mathfrak{c}_i \leftarrow (1+N)^{z_i} \cdot \mathfrak{hp}_i^r \in \mathbb{Z}^*_{N^{s+1}}, \qquad \mathfrak{c}^\dagger_{k,i} = \mathfrak{g}^{r\cdot\mathfrak{hp}^\dagger_{i,k,1} + r\cdot\tau\cdot\mathfrak{hp}^\dagger_{i,k,2}} \in \mathbb{G}\ .$$
Return ($\mathfrak{b}$, $\vec{\mathfrak{c}} \in \mathbb{G}^\ell$, $\vec{\mathfrak{c}}^\dagger \in (\mathbb{G}^\nu)^\ell$).
5. $\mathsf{dec}_{\mathrm{msk}_{\vec{y}},\tau}(\mathfrak{b}, \vec{\mathfrak{c}}, \vec{\mathfrak{c}}^\dagger)$: Check that $\sum_{i=1}^\ell y_i \cdot \mathfrak{c}^\dagger_{i,k} = \mathfrak{b}^{\mathsf{hk}^\dagger_{\vec{y},1,k} + \tau\cdot\mathsf{hk}^\dagger_{\vec{y},2,k}}$ for $k = 1, \dots, \nu$; return $\perp$ if any check fails. Set $\mathfrak{c}_{\vec{z}} \leftarrow \prod_{i=1}^\ell \mathfrak{c}_i^{y_i} / \mathfrak{b}^{\mathsf{hk}_{\vec{y}}}$, and return $(\mathfrak{c}_{\vec{z}} - 1)/N \bmod N^s$.

Fig. 4. DCR-based inner-product tag-based functional encryption $\mathsf{TBFE}_{\mathrm{dcr}}$ over the integers (using multiplicative notation for elements of $\mathbb{G} = J^*_{N^{s+1}}$)

$16/\operatorname{spf}(N) + 4 \cdot 2^{-\kappa}$. Similarly to what happens in our DCR-based IND-FE-CPA instantiation in Sect. 4.3, although there is an exponential loss in the security reduction of Thm. 32, we emphasize that there is no exponential loss using these parameters: the security loss is compensated by these well-chosen parameters.

6.5 From Tag-Based Inner-Product Functional Encryption to CCA Security

In the full version, we show how to construct a CCA-secure inner-product functional encryption from the tag-based variant, a one-time signature, and a collision resistant hash function. The transformation is a straightforward application of the generic transformation that has been applied to PKE in [22]: the tag is the hash of a fresh verification key for the one-time signature scheme, used to sign the ciphertext. This prevents malleability.

Acknowledgments. We would like to thank David Pointcheval for useful discussions. This work was partially done while the first author was student at ENS, CNRS, INRIA, and PSL Research University, Paris, France. The first author was supported in part by the CFM Foundation and by the Defense Advanced Research Projects Agency (DARPA) and Army Research Office (ARO) under Contract No. W911NF-15-C-0236. The second author was supported by the European Research Council under the European Community's Seventh Framework Programme (FP7/2007-2013 Grant Agreement no. 339563 – CryptoCloud). This third author was supported by the European Union's Horizon 2020 research and innovation programme under grant agreement No 653497 (project PANORAMIX), and by institutional research funding IUT2-1 of the Estonian Ministry of Education and Research.

References

1. Abdalla, M., Benhamouda, F., Pointcheval, D.: Disjunctions for Hash Proof Systems: New Constructions and Applications. In: EUROCRYPT 2015. LNCS, vol. 9057, pp. 69–100
2. Abdalla, M., Bourse, F., De Caro, A., Pointcheval, D.: Simple Functional Encryption Schemes for Inner Products. In: PKC 2015. LNCS, vol. 9020, pp. 733–751
3. Abdalla, M., Chevalier, C., Pointcheval, D.: Smooth Projective Hashing for Conditionally Extractable Commitments. In: CRYPTO 2009. LNCS, vol. 5677, pp. 671–689
4. Abdalla, M., Raykova, M., Wee, H.: Multi-input inner-product functional encryption from pairings. Cryptology ePrint Archive, Report 2016/425 (2016) http://eprint.iacr.org/.
5. Agrawal, S., Libert, B., Stehlé, D.: Fully Secure Functional Encryption for Linear Functions from Standard Assumptions. In: CRYPTO 2016. LNCS, vol. 9816, pp. 333–362
6. Benhamouda, F., Joye, M., Libert, B.: A new framework for privacy-preserving aggregation of time-series data. ACM Trans. Inf. Syst. Secur. **18**(3) (2016)
7. Bishop, A., Jain, A., Kowalczyk, L.: Function-hiding inner product encryption. In: ASIACRYPT 2015. LNCS, vol. 9452, pp. 470–491
8. Boneh, D., Boyen, X., Shacham, H.: Short Group Signatures. In: CRYPTO 2004. LNCS, vol. 3152, pp. 41–55
9. Boneh, D., Franklin, M.K.: Identity-based encryption from the weil pairing. In: CRYPTO 2001. LNCS, vol. 2139, pp. 213–229
10. Boneh, D., Sahai, A., Waters, B.: Functional Encryption: Definitions and Challenges. In: TCC 2011. LNCS, vol. 6597, pp. 253–273
11. Boyle, E., Chung, K., Pass, R.: On Extractability Obfuscation. In: TCC 2014. LNCS, vol. 8349, pp. 52–73
12. Cramer, R., Shoup, V.: A Practical Public Key Cryptosystem Provably Secure against Adaptive Chosen Ciphertext Attack. In: CRYPTO 1998. LNCS, vol. 1462, pp. 13–25
13. Cramer, R., Shoup, V.: Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. Cryptology ePrint Archive, Report 2001/085 (2001) Full version of [14]. http://eprint.iacr.org/2001/085.
14. Cramer, R., Shoup, V.: Universal Hash Proofs and a Paradigm for Adaptive Chosen Ciphertext Secure Public-Key Encryption. In: EUROCRYPT 2002. LNCS, vol. 2332, pp. 45–64

15. Damgård, I., Jurik, M.: A Generalisation, a Simplification and Some Applications of Paillier's Probabilistic Public-Key System. In: PKC 2001. LNCS, vol. 1992, pp. 119–136
16. Damgård, I., Jurik, M.: A Length-Flexible Threshold Cryptosystem with Applications. In: ACISP 2003. LNCS, vol. 2727, pp. 350–364
17. Datta, P., Dutta, R., Mukhopadhyay, S.: Functional encryption for inner product with full function privacy. In: PKC 2016. LNCS, vol. 9614, pp. 164–195
18. Escala, A., Herold, G., Kiltz, E., Ràfols, C., Villar, J.L.: An Algebraic Framework for Diffie-Hellman Assumptions. In: CRYPTO (2) 2013. LNCS, vol. 8043, pp. 129–147
19. Garg, S., Gentry, C., Halevi, S., Raykova, M., Sahai, A., Waters, B.: Candidate Indistinguishability Obfuscation and Functional Encryption for all Circuits. In: FOCS 2013, pp. 40–49
20. Garg, S., Gentry, C., Halevi, S., Zhandry, M.: Functional Encryption Without Obfuscation. In: TCC 2016-A (2). LNCS, vol. 9563, pp. 480–511
21. Katz, J., Sahai, A., Waters, B.: Predicate Encryption Supporting Disjunctions, Polynomial Equations, and Inner Products. In: EUROCRYPT 2008. LNCS, vol. 4965, pp. 146–162
22. Kiltz, E.: Chosen-Ciphertext Security from Tag-Based Encryption. In: TCC 2006. LNCS, vol. 3876, pp. 581–600
23. Kiltz, E., Vahlis, Y.: CCA2 secure IBE: standard model efficiency through authenticated symmetric encryption. In: CT-RSA 2008. Lecture Notes in Computer Science, vol. 4964, pp. 221–238
24. MacKenzie, P., Reiter, M.K., Yang, K. In: Alternatives to Non-malleability: Definitions, Constructions, and Applications. Springer Berlin Heidelberg, Berlin, Heidelberg (2004) pp. 171–190
25. Menezes, A.J., Oorschot, P.C.V., Vanstone, S.A.: Handbook of Applied Cryptography. CRC Press (1996)
26. Nandi, M., Pandit, T.: Generic conversions from cpa to cca secure functional encryption. Cryptology ePrint Archive, Report 2015/457 (2015) http://eprint.iacr.org/2015/457.
27. Naor, M., Yung, M.: Public-key cryptosystems provably secure against chosen ciphertext attacks. In: STOC 1990, pp. 427–437
28. O'Neill, A.: Definitional Issues in Functional Encryption. Technical Report 2010/556, IACR (2010) Available at http://eprint.iacr.org/2010/556, last retrieved version from 18 Mar 2011.
29. Paillier, P.: Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. In: EUROCRYPT 1999. LNCS, vol. 1592, pp. 223–238
30. Rackoff, C., Simon, D.R.: Non-Interactive Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attack. In: CRYPTO 1991. LNCS, vol. 576, pp. 433–444
31. Regev, O.: On Lattices, Learning with Errors, Random Linear Codes, and Cryptography. In: STOC 2005, pp. 84–93
32. Sahai, A., Waters, B.: Fuzzy Identity-Based Encryption. In: EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473
33. Waters, B.: A Punctured Programming Approach to Adaptively Secure Functional Encryption. In: CRYPTO (2) 2015. LNCS, vol. 9216, pp. 678–697
34. Yamada, S., Attrapadung, N., Hanaoka, G., Kunihiro, N.: Generic constructions for chosen-ciphertext secure attribute based encryption. In: PKC 2011. LNCS, vol. 6571, pp. 71–89