Fully Secure Functional Encryption for Inner Products, from Standard Assumptions

Shweta Agrawal[1], Benoît Libert[2], and Damien Stehlé[2]

[1] I.I.T. Delhi, India
[2] ENS de Lyon, Laboratoire LIP (U. Lyon, CNRS, ENSL, INRIA, UCBL), France.
benoit.libert@ens-lyon.fr, damien.stehle@ens-lyon.fr –

Abstract. Functional encryption is a modern public-key paradigm where a master secret key can be used to derive sub-keys SK_F associated with certain functions F in such a way that the decryption operation reveals $F(M)$, if M is the encrypted message, and nothing else. Recently, Abdalla *et al.* gave simple and efficient realizations of the primitive for the computation of linear functions on encrypted data: given an encryption of a vector $\boldsymbol{y}$ over some specified base ring, a secret key $SK_{\boldsymbol{x}}$ for the vector $\boldsymbol{x}$ allows computing $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$. Their technique surprisingly allows for instantiations under standard assumptions, like the hardness of the Decision Diffie-Hellman (DDH) and Learning-with-Errors (LWE) problems. Their constructions, however, are only proved secure against *selective* adversaries, which have to declare the challenge messages M_0 and M_1 at the outset of the game.

In this paper, we provide constructions that provably achieve security against more realistic *adaptive* attacks (where the messages M_0 and M_1 may be chosen in the challenge phase, based on the previously collected information) for the same inner product functionality. Our constructions are obtained from hash proof systems endowed with homomorphic properties over the key space. They are (almost) as efficient as those of Abdalla *et al.* and rely on the same hardness assumptions.

In addition, we obtain a solution based on Paillier's composite residuosity assumption, which was an open problem even in the case of selective adversaries. We also propose LWE-based schemes that allow evaluation of inner products modulo a prime p, as opposed to the schemes of Abdalla *et al.* that are restricted to evaluations of integer inner products of short integer vectors. We finally propose a solution based on Paillier's composite residuosity assumption that enables evaluation of inner products modulo an RSA integer $N = p \cdot q$.

We demonstrate that the functionality of inner products over a prime field is very powerful and can be used to construct bounded collusion FE for all circuits.

Keywords. Functional encryption, adaptive security, standard assumptions, DDH, LWE, extended LWE, composite residuosity.

1 Introduction

Functional encryption (FE) [53, 16] is an ambitious generalization of public-key encryption, which overcomes the all-or-nothing, user-based access to data that is inherent to public key encryption and enables fine grained, role-based access that makes it very desirable for modern applications. A bit more formally, given an encryption $\mathsf{enc}(X)$ and a key corresponding to a function F, the key holder only learns $F(X)$ and nothing else. Apart from its theoretical appeal, the concept of FE also finds numerous applications. In cloud computing platforms, users can store encrypted data on a remote server and subsequently provide the server with a key SK_F which allows it to compute the function F of the underlying data without learning anything else.

In some cases, the message $X = (\mathsf{IND}, M)$ consists of an index IND (which can be thought of as a set of descriptive attributes) and a message M, which is sometimes called "payload". One distinguishes FE systems with public index, where IND is publicly revealed by the ciphertext but M is hidden, from those with private index, where IND and M are both hidden. Public index FE is popularly referred to as attribute based encryption.

A Brief History of FE. The birth of Functional Encryption can be traced back to Identity Based Encryption [55, 14] which can be seen as the first nontrivial generalization of Public Key Encryption. However, it was the work of Sahai and Waters [53] that coined the term Attribute Based Encryption, and the subsequent, natural unification of all these primitives under the umbrella of Functional Encryption took place only relatively recently [16, 46]. Constructions of public index FE have matured from specialized – equality testing [14, 10, 33], keyword search [13, 1, 41], boolean formulae [39], inner product predicates [41], regular languages [56] – to general polynomial-size circuits [32, 37, 15] and even Turing machines [34]. The journey of private index FE has been significantly more difficult, with inner product predicate constructions [41, 3] being the state of the art for a long time until the recent elegant generalization to polynomial-size circuits [38].

However, although private index FE comes closer than ever before to the goal of general FE, it falls frustratingly short. This is because all known constructions of private index FE only achieve *weak attribute hiding*, which severely restricts the function keys that the adversary can request in the security game – the adversary may request keys for functions f_i that do not decrypt the challenge ciphertext (IND^*, M^*), i.e., $f_i(\mathsf{IND}^*) \neq 0$ holds for all i. The most general notion of FE – private index, strongly attribute hiding – has been built for the restricted case of bounded collusions [36, 35] or using the brilliant, but ill-understood[3] machinery of multi-linear maps [31] and indistinguishability obfuscation [31]. These constructions provide FE for general polynomial-size circuits and Turing

[3] Indeed, the two candidate multi-linear maps [30, 21] put forth in 2013 were recently found to be insecure [20, 40].

machines [34], but, perhaps surprisingly, there has been little effort to build the general notion of FE ground-up, starting from smaller functionalities.

This appears as a gaping hole that begs to be filled. Often, from the practical standpoint, efficient constructions for a smaller range of functionalities, such as linear functions or polynomials, are extremely relevant, and such an endeavour will also help us understand the fundamental barriers that thwart our attempts for general FE. This motivates the question:

Can we build FE for restricted classes of functions, satisfying standard security definitions, under well-understood assumptions?

Recently, Abdalla, Bourse, De Caro and Pointcheval [2] considered the question of building FE for linear functions. Here, a ciphertext C encrypts a vector $\boldsymbol{y} \in \mathcal{D}^\ell$ over some ring $\mathcal{D}$, a secret key for the vector $\boldsymbol{x} \in \mathcal{D}^\ell$ allows computing $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ and nothing else about $\boldsymbol{y}$. Note that this is quite different from the inner product predicate functionality of [41, 3]: the former computes the actual value of the inner product while the latter tests whether the inner product is zero or not, and reveals a hidden bit M if so. Abdalla *et al.* [2] showed, surprisingly, that this functionality allows for very simple and efficient realizations under standard assumptions like the Decision Diffie-Hellman (DDH) and Learning-with-Errors (LWE) assumptions [50]. The instantiation from DDH was especially unexpected since DDH is not known to easily lend itself to the design of such primitives.[4] What enables this surprising result is that the functionality itself is rather limited – note that with ℓ queries, the adversary can reconstruct the entire message vector. Due to this, the scheme need not provide *collusion resistance*, which posits that no collection of secret keys for functions $F_1, \ldots, F_q$ should make it possible to decrypt a ciphertext that no individual such key can decrypt. Collusion resistance is usually the chief obstacle in proving security of FE schemes. On the contrary, for linear FE constructions, if two adversaries combine their keys, they do get a valid new key, but this key gives them a plaintext which could anyway be computed by their individual plaintexts. Hence, collusion is permitted by the functionality itself, and constructions can be much simpler. As we shall see below, linear FE is already very useful and yields many interesting applications. See Appendix B for some standard applications.

On the downside, Abdalla *et al.* [2] only proved their schemes to be secure against *selective* adversaries, that have to declare the challenge messages M_0, M_1 of the semantic security game upfront, before seeing the master public key mpk. Selective security is usually too weak a notion for practical applications and is often seen as a stepping stone to proving full adaptive security. Historically, most flavors of functional encryption have been first realized for selective adversaries [10, 53, 39, 41, 31] before being upgraded to attain full security. Boneh and Boyen [11] observed that a standard complexity leveraging argument can be used to

[4] And indeed, this unsuitability partially manifests itself in the limitation of message/function space of the aforementioned construction: message/function vectors must be short integer vectors, and the inner product is evaluated over the integers.

argue that a selectively-secure system is also adaptively secure. However, this argument is not satisfactory in general as the reduction incurs an exponential security loss in the message length. Quite recently, Ananth, Brakerski, Segev and Vaikuntanathan [6] described an elegant generic method of building adaptively secure functional encryption systems from selectively secure ones. However their transformation is based on the existence of a *sufficiently expressive* selectively secure FE scheme, where sufficiently secure roughly means capable of evaluating a weak PRF. Since no such scheme from standard assumptions is known, their transformation does not apply to our case, and in any case would significantly increase the complexity of the construction, even if it did.

Our Results. In this paper, we describe fully secure functional encryption systems for the evaluation of inner products on encrypted data. We propose schemes that evaluate inner products of integer vectors, based on DDH, LWE and the Composite Residuosity hardness assumptions. Our DDH-based and LWE-based constructions for integer inner products are of efficiency comparable to those of Abdalla *et al.* [2] and rely on the same standard assumptions. Note that a system based on Paillier's composite residuosity assumption was an open problem even for the case of selective adversaries, which we resolve in this work.

Additionally, we propose schemes that evaluate inner products modulo a prime p or a composite $N = pq$, based on the LWE and Composite Residuosity hardness assumptions. In contrast, the constructions of [2] must restrict the ring $\mathcal{D}$ to the ring of integers, which is a significant drawback. Indeed, although their DDH-based realization allows evaluating $\langle \boldsymbol{x}, \boldsymbol{y} \rangle \mod p$ when the latter value is sufficiently small, their security proof restricts the functionality to the computation of $\langle \boldsymbol{x}, \boldsymbol{y} \rangle \in \mathbb{Z}$.

The functionality of inner products over a prime field is very powerful: we show that it can be bootstrapped all the way to yield a conceptually simple construction for bounded collusion FE for all circuits! The only known construction for general FE handling bounded collusions is by Gorbunov, Vaikuntanathan and Wee [36]. Our construction is conceptually much simpler, albeit a bit more inefficient.

1.1 Overview of techniques

We briefly summarize our techniques below.

Fully secure linear FE: hash proof systems. Our DDH-based construction and its security proof implicitly build on hash proof systems [23]. It involves public parameters comprised of group elements $\big(g, h, \{h_i = g^{s_i} \cdot h^{t_i}\}_{i=1}^{\ell}\big)$, where g, h generate a cyclic group $\mathbb{G}$ of prime order q, and the master secret key is $\mathsf{msk} = (\boldsymbol{s}, \boldsymbol{t}) \in \mathbb{Z}_q^\ell \times \mathbb{Z}_q^\ell$. On input of a vector $\boldsymbol{y} = (y_1, \ldots, y_\ell) \in \mathbb{Z}_q^\ell$, the encryption algorithm computes $(g^r, h^r, \{g^{y_i} \cdot h_i^r\}_{i=1}^{\ell})$ in such a way that a secret key of the form $SK_{\boldsymbol{x}} = (\langle \boldsymbol{s}, \boldsymbol{x} \rangle, \langle \boldsymbol{t}, \boldsymbol{x} \rangle)$ allows computing $g^{\langle \boldsymbol{y}, \boldsymbol{x} \rangle}$ in the same way as in [2]. Despite its simplicity and its efficiency (only one more group element than in [2] is needed in the ciphertext), we show that the above system can be proved fully secure using arguments – akin to those of Cramer and Shoup [22] – which consider what the adversary knows about the master secret key $(\boldsymbol{s}, \boldsymbol{t}) \in \mathbb{Z}_q^\ell \times \mathbb{Z}_q^\ell$ in the information theoretic sense. The security proof is arguably simpler than its counterpart in the selective case [2]. As in all security proofs based on hash proof systems, it uses the fact that the secret key is known to the reduction at any time, which makes it simpler to handle secret key queries without knowing the adversary's target messages $\boldsymbol{y}_0, \boldsymbol{y}_1 \in \mathbb{Z}_q^\ell$ in advance.

While our DDH-based realization only enables efficient decryption when the inner product $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ is contained in a sufficiently small interval, we show how to eliminate this restriction using Paillier's cryptosystem in the same way as in [19, 18]. We thus obtain the first solution based on the Composite Residuosity assumption, which was previously an open problem (even in the case of selective security).

LWE-based fully secure linear FE. Our LWE-based construction builds on the dual Regev encryption scheme from Gentry, Peikert and Vaikuntanathan [33]. Its security analysis requires more work. The master public key contains a random matrix $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$. For simplicity, we restrict ourselves to plaintext vectors and secret key vectors with binary coordinates. Each vector coordinate $i \in \{1, \ldots, \ell\}$ requires a master public key component $\boldsymbol{u}_i^T = \boldsymbol{z}_i^T \cdot \mathbf{A} \in \mathbb{Z}_q^n$, for a small norm vector $\boldsymbol{z}_i \in \mathbb{Z}^m$ made of Gaussian entries which will be part of the master secret key $\mathsf{msk} = \{\boldsymbol{z}_i\}_{i=1}^\ell$. Each $\{\mathbf{u}_i\}_{i=1}^\ell$ can be seen as a syndrome in the GPV trapdoor function for which vector $\mathbf{z}_i$ is a pre-image. Our security analysis will rely on the fact that each GPV syndrome has a large number of pre-images and, conditionally on $\mathbf{u}_i \in \mathbb{Z}_q^n$, each $\mathbf{z}_i$ retains a large amount of entropy. In the security proof, this will allow us to apply arguments similar to those of hash proof systems [23] when we will generate the challenge ciphertext using $\{\mathbf{z}_i\}_{i=1}^\ell$. More precisely, when the first part $\mathbf{c}_0 \in \mathbb{Z}_q^m$ of the ciphertext is a random vector instead of an actual LWE sample $\mathbf{c}_0 = \mathbf{A} \cdot \mathbf{s} + \mathbf{e}_0$, the action of $\{\mathbf{z}_i\}_{i=1}^\ell$ on $\mathbf{c}_0 \in \mathbb{Z}_q^m$ produces vectors that appear statistically uniform to any legitimate adversary. In order to properly simulate the challenge ciphertext using the master secret key $\{\mathbf{z}_i\}_{i=1}^\ell$, we use a variant of the extended LWE assumption [47] (eLWE) so as to have the (hint) values $\{\langle \mathbf{z}_i, \mathbf{e}_0 \rangle\}_{i=1}^\ell$ at disposal. One difficulty is that the reductions from LWE to eLWE proved in [5] and [17] handle a single hint vector $\boldsymbol{z}$. Fortunately, we extend the techniques of Brakerski *et al.* [17] using the gadget matrix from [42] to obtain a reduction from LWE to the multi-hint variant of eLWE that we use in the security proof. More specifically, we prove that the multi-hint variant of eLWE remains at least as hard as LWE when the adversary obtains as many as $n/2$ hints, where n is the dimension of the LWE secret.

Evaluation inner products modulo p. Our construction from the DDH assumption natively supports the computation of inner products modulo a prime p as long as the remainder $\langle \boldsymbol{x}, \boldsymbol{y} \rangle \bmod p$ falls in a polynomial-size interval. Under the Paillier and LWE assumptions, we first show how to compute integer inner products $\langle \boldsymbol{x}, \boldsymbol{y} \rangle \in \mathbb{Z}$. In a second step, we upgrade our Paillier and LWE-based systems so as to compute inner products modulo a composite $N = pq$ and a prime p, respectively, without leaking the actual value $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ over $\mathbb{Z}$.

Hiding anything but the remainder modulo N or p requires additional techniques. One of them, which we use in the context of Paillier-based FE, is to partially blind $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ using noise terms consisting of random multiples of the modulus. In the context of LWE-based FE, this is achieved by using an LWE modulus of the form $q = p \cdot p'$ and multiplying plaintexts by p', so that an inner product modulo q over the ciphertext space natively translates into an inner product modulo p for the underlying plaintexts.

The latter plaintext/ciphertext manipulations do not solve another difficulty which arises from the discrepancy between the base rings of the master key and the secret key vectors: indeed, the master key consists of integer vectors, whereas the secret keys are defined modulo an integer. When the adversary queries a secret key vector $\boldsymbol{x} \in \mathbb{Z}_p^\ell$ (or $\mathbb{Z}_N^\ell$), it gets the corresponding combination modulo p of the master key components. By making appropriate vector queries that are linearly dependent modulo p (and hence valid), an attacker could learn a combination of the master key components which is singular modulo p but invertible over the field of rational numbers: it would then obtain the whole master key! However, note that as long as the adversary only queries secret keys for $\ell - 1$ independent vectors over $\mathbb{Z}_p^\ell$ (or $\mathbb{Z}_N^\ell$), there is no reason not to reveal more than $\ell - 1$ secret keys overall. In order to make sure that the adversary only obtains redundant information by making more than $\ell - 1$ queries, we assume that a trusted authority keeps track of all vectors $\boldsymbol{x}$ for which secret keys were previously given out (more formally, the key generation algorithm is stateful).

Compiling Linear FE to Bounded Collusion General FE. We provide a conceptually simpler way to build q-query Functional Encryption for all circuits. The only known construction for this functionality was suggested by Gorbunov *et al.* in [36]. At a high level, the q-query construction by Gorbunov *et al.* is built in several layers, as follows:

1. They start with a single key FE scheme for all circuits, which was provided by [52].
2. The single FE scheme is compiled into a q-query scheme for NC_1 circuits. This is the most non-trivial part of the construction. They run N copies of the single key scheme, where $N = O(q^4)$. To encrypt, they encrypt the views of some N-party MPC protocol computing some functionality related to C, *à la* "MPC in the head". For the MPC protocol, they use the BGW [8] semi-honest MPC protocol without degree reduction and exploit the fact that this protocol is completely non-interactive when used to compute bounded degree functions. The key generator provides the decryptor with a subset of the single query FE keys, where the subsets are guaranteed to have small pairwise intersections. This subset of keys enables the decryptor to recover sufficiently many shares of $C(x)$ which enables her to compute $C(x)$ via polynomial interpolation. However, an attacker with q keys only learns a share x_i in the clear if two subsets of keys intersect, and due to small pairwise

intersections, this does not occur often enough for him learn sufficiently many shares of x, hence, by the guarantees of secret sharing, input x remains hidden.

3. Finally, they bootstrap the q-query FE for NC_1 to a q-query FE for all circuits using computational randomized encodings [7]. They must additionally use cover-free sets to ensure that fresh randomness is used for each randomized encoding.

Our starting point is the bootstrapping theorem of [36], which states that an FE for NC_1 that handles q queries can be bootstrapped to yield an FE for all polynomial-size circuits that is also secure against q queries. The bootstrapping relies on the beautiful result of Applebaum *et al.* [7, Theorem 4.11] which states that every polynomial time computable function f admits a perfectly correct computational randomized encoding of degree 3. In more details, let $\mathcal{C}$ be a family of polynomial-size circuits. Let $C \in \mathcal{C}$ and let x be some input. Let $\widetilde{C}(x, R)$ be a randomized encoding of C that is computable by a constant depth circuit with respect to inputs x and R. Then consider a new family of circuits $\mathcal{G}$ defined by:

$$G_{C,\Delta}(x, R_1, \ldots, R_S) = \left\{ \widetilde{C}\Big(x; \underset{a \in \Delta}{\oplus} R_a\Big) : \ C \in \mathcal{C},\ \Delta \subseteq [S] \right\},$$

for some sufficiently large S (quadratic in the number of queries q). As observed in [36], circuit $G_{C,\Delta}(\cdot, \cdot)$ is computable by a constant degree polynomial (one for each output bit). Given an FE scheme for $\mathcal{G}$, one may construct a scheme for $\mathcal{C}$ by having the decryptor first recover the output of $G_{C,\Delta}(x, R_1, \ldots, R_S)$ and then applying the decoder for the randomized encoding to recover $C(x)$.

However, to support q queries the decryptor must compute q randomized encodings, each of which needs fresh randomness. This is handled by hardcoding S random elements in the ciphertext and using random subsets $\Delta \subseteq [S]$ (which are cover-free with overwhelming probability) to compute fresh randomness $\underset{a \in \Delta}{\oplus} R_a$ for every query. The authors then conclude that bounded query FE for NC_1 suffices to construct a bounded query FE scheme for all circuits.

We observe that the ingredient required to bootstrap is *not FE for the entire circuit class* NC_1 but rather only the particular circuit class $\mathcal{G}$ as described above. This circuit class, being computable by degree 3 polynomials, may be supported by a linear FE scheme, via linearization of the degree 3 polynomials! To illustrate, let us consider FE secure only for a single key. Then, the functionality that the initial FE must support is exactly the randomized encoding of [7], which, indeed, is in NC_0. Now, to support q queries, we must ensure that each key uses a fresh piece of randomness, and this is provided using a cover-free set family S as in [36] – the key generator picks a random subset $\Delta \subseteq [S]$ and sums up its elements to obtain i.i.d. randomness for the key being requested. To obtain a random element in this manner, addition over the integers does not suffice, we must take addition modulo p. Here, our inner product modulo p construction comes to our rescue!

Putting it together, the encryptor encrypts all degree 3 monomials in the inputs $R_1, \ldots, R_S$ and $x_1, \ldots, x_\ell$. Note that this ciphertext is polynomial in size. Now, for a given circuit C, the keygen algorithm samples some $\Delta \subseteq [S]$ and computes the symbolic degree 3 polynomials which must be released to the decryptor. It then provides the linear FE keys to compute the same. By correctness and security of Linear FE as well as the randomizing polynomial construction, the decryptor learns $C(x)$ and nothing else. The final notion of security that we obtain is non-adaptive simulation based security NA-SIM [46, 36]. For more details, we refer the reader to Section 6.

2 Background

In this section, we recall the hardness assumptions underlying the security of the schemes we will describe. The functionality and security definitions of functional and non-interactive controlled functional encryption schemes are given in Appendix A.

Our first scheme relies on the standard DDH assumption in ordinary (i.e., non-pairing-friendly) cyclic groups.

Definition 1. *In a cyclic group $\mathbb{G}$ of primer order q, the* **Decision Diffie-Hellman** *(*DDH*) problem is to distinguish the distributions $D_0 = \{(g, g^a, g^b, g^{ab}) \mid g \hookleftarrow \mathbb{G},\ a, b \hookleftarrow \mathbb{Z}_q\}$, $D_1 = \{(g, g^a, g^b, g^c) \mid g \hookleftarrow \mathbb{G},\ a, b, c \hookleftarrow \mathbb{Z}_q\}$.*

A variant of our first scheme relies on Paillier's composite residuosity assumption.

Definition 2 ([48]). *Given an RSA modulus $N = pq$ and an integer $s > 0$, for prime numbers p, q, the* s**-Decision Composite Residuosity** *(s-DCR) problem is to distinguish the distributions $D_0 := \{z = z_0^{N^s} \bmod N^{s+1} \mid z_0 \hookleftarrow \mathbb{Z}_N^*\}$ and $D_1 := \{z \hookleftarrow \mathbb{Z}_{N^{s+1}}^*\}$.*

For each $s > 0$, the s-DCR assumption was shown [25] equivalent to the original 1-DCR assumption [48].

Our third construction builds on the Learning-With-Errors (LWE) problem, which is known to be at least as hard as certain standard lattice problems in the worst case [51, 17].

Definition 3. *Let q, α, m be functions of a parameter n. For a secret $\mathbf{s} \in \mathbb{Z}_q^n$, the distribution $A_{q,\alpha,\mathbf{s}}$ over $\mathbb{Z}_q^n \times \mathbb{Z}_q$ is obtained by sampling $\mathbf{a} \hookleftarrow \mathbb{Z}_q^n$ and an $e \hookleftarrow D_{\mathbb{Z},\alpha q}$, and returning $(\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle + e) \in \mathbb{Z}_q^{n+1}$. The Learning With Errors problem $\mathsf{LWE}_{q,\alpha,m}$ is as follows: For $\mathbf{s} \hookleftarrow \mathbb{Z}_q^n$, the goal is to distinguish between the distributions:*

$$D_0(\mathbf{s}) := U(\mathbb{Z}_q^{m \times (n+1)}) \quad \text{and} \quad D_1(\mathbf{s}) := (A_{q,\alpha,\mathbf{s}})^m.$$

We say that a PPT algorithm $\mathcal{A}$ solves $\mathsf{LWE}_{q,\alpha}$ if it distinguishes $D_0(\mathbf{s})$ and $D_1(\mathbf{s})$ with non-negligible advantage (over the random coins of $\mathcal{A}$ and the randomness of the samples), with non-negligible probability over the randomness of $\mathbf{s}$.

3 Fully secure functional encryption for inner products from DDH

In this section, we show that an adaptation of the DDH-based construction of Abdalla *et al.* [2] provides full security under the standard DDH assumption. Like [2], the scheme computes inner products over $\mathbb{Z}$ as long as they land in a sufficiently small interval.

In comparison with the solution of Abdalla *et al.*, we only introduce one more group element in the ciphertext and all operations are just as efficient as in [2]. Our scheme is obtained by modifying [2] in the same way as Damgård's encryption scheme [24] was obtained from the Elgamal cryptosystem. The original DDH-based system of [2] encrypts a vector $\boldsymbol{y} = (y_1, \ldots, y_\ell) \in \mathbb{Z}_q^\ell$ by computing $(g^r, \{g^{y_i} \cdot h_i^r\}_{i=1}^\ell)$, where $\{h_i = g^{s_i}\}_{i=1}^\ell$ are part of the master public key and $\mathsf{sk}_{\boldsymbol{x}} = \sum_{i=1}^\ell s_i \cdot x_i \bmod q$ is the secret key associated with the vector $\boldsymbol{x} = (x_1, \ldots, x_\ell) \in \mathbb{Z}_q^\ell$. Here, we encrypt $\boldsymbol{y}$ in the fashion of Damgård's Elgamal, by computing $(g^r, h^r, \{g^{y_i} \cdot h_i^r\}_{i=1}^\ell)$. The decryption algorithm uses secret keys of the form $\mathsf{sk}_{\boldsymbol{x}} = (\sum_{i=1}^\ell s_i \cdot x_i, \sum_{i=1}^\ell t_i \cdot x_i)$, where $h_i = g^{s_i} \cdot h^{t_i}$ for each i and $\boldsymbol{s} = (s_1, \ldots, s_\ell) \in \mathbb{Z}_q^\ell$ and $\boldsymbol{t} = (t_1, \ldots, t_\ell) \in \mathbb{Z}_q^\ell$ are part of the master key msk.

The scheme and its security proof also build on ideas from the Cramer-Shoup cryptosystem [22, 23] in that the construction can also be seen as an applying a hash proof system [23] with homomorphic properties over the key space.

$\mathsf{Setup}(1^\lambda, 1^\ell)$: Choose a cyclic group $\mathbb{G}$ of prime order $q > 2^\lambda$ with generators $g, h \hookleftarrow \mathbb{G}$. Then, for each $i \in \{1, \ldots, \ell\}$, sample $s_i, t_i \hookleftarrow \mathbb{Z}_q$ and compute $h_i = g^{s_i} \cdot h^{t_i}$. Define $\mathsf{msk} := \{(s_i, t_i)\}_{i=1}^\ell$ and

$$\mathsf{mpk} := \Big(\mathbb{G}, g, h, \{h_i\}_{i=1}^\ell\Big).$$

$\mathsf{Keygen}(\mathsf{msk}, \boldsymbol{x})$: To generate a key for the vector $\boldsymbol{x} = (x_1, \ldots, x_\ell) \in \mathbb{Z}_q^\ell$, compute $\mathsf{sk}_{\boldsymbol{x}} = (s_{\boldsymbol{x}}, t_{\boldsymbol{x}}) = (\sum_{i=1}^\ell s_i \cdot x_i, \sum_{i=1}^\ell t_i \cdot x_i) = (\langle \boldsymbol{s}, \boldsymbol{x}\rangle, \langle \boldsymbol{t}, \boldsymbol{x}\rangle)$.

$\mathsf{Encrypt}(\mathsf{mpk}, \boldsymbol{y})$: To encrypt a vector $\boldsymbol{y} = (y_1, \ldots, y_\ell) \in \mathbb{Z}_q^\ell$, sample $r \hookleftarrow \mathbb{Z}_q$ and compute

$$C = g^r, \qquad D = h^r, \qquad \{E_i = g^{y_i} \cdot h_i^r\}_{i=1}^\ell.$$

Return $C_{\boldsymbol{y}} = (C, D, E_1, \ldots, E_\ell)$.

$\mathsf{Decrypt}(\mathsf{mpk}, \mathsf{sk}_{\boldsymbol{x}}, C_{\boldsymbol{y}})$: Given $\mathsf{sk}_{\boldsymbol{x}} = (s_{\boldsymbol{x}}, t_{\boldsymbol{x}})$, compute

$$E_{\boldsymbol{x}} = (\prod_{i=1}^\ell E_i^{x_i})/(C^{s_{\boldsymbol{x}}} \cdot D^{t_{\boldsymbol{x}}}).$$

Then, compute and output $\log_g(E_{\boldsymbol{x}})$.

The decryption algorithm requires to compute a discrete logarithm. This is in general too expensive. Like in [2], this can be circumvented by imposing that the computed inner product lies in an interval $\{0, \ldots, L\}$, for some polynomially bounded integer L. Then, computing the required discrete logarithm may be performed in time $\widetilde{O}(L^{1/2})$ using Pollard's kangaroo method [49]. Galbraith and Ruprai [29] gave an improved algorithm with complexity $\widetilde{O}(L^{1/2})$. As reported in [9], this can be reduced to $\widetilde{O}(L^{1/3})$ operations by precomputing a table of size $\widetilde{O}(L^{1/3})$. Note that even though the functionality is limited (decryption may not be performed efficiently for all key vectors and for all message vectors), while proving security we will let the adversary query any key vector in $\mathbb{Z}_q^\ell$.

Before proceeding with the security proof, we would like to clarify that, although the scheme of [2] only decrypts values in a polynomial-size space, the usual complexity leveraging argument does not prove it fully secure via a polynomial reduction. Indeed, when ℓ is polynomial in λ, having the inner product $\langle \boldsymbol{y}, \boldsymbol{x} \rangle$ in a small interval does not mean that original vector $\boldsymbol{y} \in \mathbb{Z}_q^\ell$ lives in a polynomial-size universe. In Section 5, we show how to eliminate the small-interval restriction using Paillier's cryptosystem [48].

The security analysis uses similar arguments to those of Cramer and Shoup [22, 23] in that it exploits the fact that mpk does not reveal too much information about the master secret key. At some step, the challenge ciphertext is generated using msk instead of the public key and, as long as msk retains a sufficient amount of entropy from the adversary's view, it will perfectly hide which vector among $\boldsymbol{y}_0, \boldsymbol{y}_1$ is actually encrypted. The reason why we can prove adaptive security is the fact that, as usual in security proofs relying on hash proof systems [22, 23], the reduction knows the master secret key at any time. It can thus correctly answer all secret key queries without knowing the challenge messages $\boldsymbol{y}_0, \boldsymbol{y}_1$ beforehand.

The DDH-based scheme can easily be generalized so as to rely on weaker variants of DDH, like the Decision Linear assumption [12], the k-linear assumption [54] for $k > 2$ or the Matrix DDH assumption [28]. It can also be generalized to rely on other key homomorphic hash proof systems, like those of [23].

Theorem 1. *The scheme provides full security under the* DDH *assumption.*

Proof. The proof uses a sequence of games that begins with the real game and ends with a game where the adversary's advantage is zero. For each i, we denote by S_i the event that the adversary wins in Game i.

Game 0: This is the real game. In this game, the adversary $\mathcal{A}$ is given mpk. In the challenge phase, $\mathcal{A}$ chooses two distinct vectors $\boldsymbol{y}_0, \boldsymbol{y}_1 \in \mathbb{Z}_q^\ell$ and obtains an encryption of $\boldsymbol{y}_\beta = (y_{\beta,1}, \ldots, y_{\beta,\ell})$ for a random bit $\beta \hookleftarrow \{0, 1\}$ sampled by the challenger $\mathcal{B}$. At the end of the game, $\mathcal{A}$ outputs $\beta' \in \{0, 1\}$ and we denote by S_0 the event that $\beta' = \beta$. For any vector $\boldsymbol{x} \in \mathbb{Z}_q^\ell$ submitted to the secret key extraction oracle, it must be the case that $\langle \boldsymbol{x}, \boldsymbol{y}_0 \rangle = \langle \boldsymbol{x}, \boldsymbol{y}_1 \rangle \bmod q$.

Game 1: We modify the generation of the challenge $C_{\boldsymbol{y}_\beta} = (C, D, E_1, \ldots, E_\ell)$. Namely, the challenger $\mathcal{B}$ first computes

$$C = g^r \quad \text{and} \quad D = h^r, \tag{3.1}$$

for a randomly sampled $r \hookleftarrow \mathbb{Z}_q$. Then, it uses $\mathsf{msk} := \{(s_i, t_i)\}_{i=1}^{\ell}$ to compute

$$E_i = g^{y_{\beta,i}} \cdot C^{s_i} \cdot D^{t_i}. \tag{3.2}$$

It can be observed that $C_{\boldsymbol{y}_\beta} = (C, D, E_1, \ldots, E_\ell)$ has the same distribution as in Game 0. We hence have $\Pr[S_1] = \Pr[S_0]$.

Game 2: In this game, we modify again the generation of $C_{\boldsymbol{y}_\beta} = (C, D, E_1, \ldots, E_\ell)$ in the challenge phase. Namely, instead of computing the pair (C, D) as in (3.1), the challenger $\mathcal{B}$ samples $r, \tilde{r} \hookleftarrow \mathbb{Z}_q$ and sets

$$C = g^r \quad \text{and} \quad D = h^{\tilde{r}}.$$

The ciphertext components $(E_1, \ldots, E_\ell)$ are still computed as per (3.2). Under the DDH assumption, this modification should not significantly affect $\mathcal{A}$'s view and we have $|\Pr[S_2] - \Pr[S_1]| \leq \mathbf{Adv}_{\mathcal{B}}^{\mathsf{DDH}}(\lambda)$.

In Game 2, we claim that $\Pr[S_2] = 1/2$, so that $\mathcal{A}$ has no advantage at all. To see this, we first remark that the pair (C, D) can be written $(C, D) = (g^r, h^{r+r'})$ for some uniformly random $r' \hookleftarrow \mathbb{Z}_q$. So, for each $i \in \{1, \ldots, \ell\}$, we also have

$$E_i = g^{y_{\beta,i}} \cdot C^{s_i} \cdot D^{t_i} = g^{y_{\beta,i} + \omega \cdot r' \cdot t_i} \cdot h_i^r,$$

where $\omega = \log_g(h)$. If we define $t_i' = t_i + (\omega \cdot r')^{-1} \cdot (y_{\beta,i} - y_{1-\beta,i})$ for each $i \in \{1, \ldots, \ell\}$, we also have $E_i = g^{y_{1-\beta,i} + \omega \cdot r' \cdot t_i'} \cdot h_i^r$. In other words, the vector

$$(E_1, \ldots, E_\ell) = g^{\boldsymbol{y}_\beta + \omega \cdot r' \cdot \boldsymbol{t}} \cdot (h_1^r, \ldots, h_\ell^r) \tag{3.3}$$

can also be written

$$(E_1, \ldots, E_\ell) = g^{\boldsymbol{y}_{1-\beta} + \omega \cdot r' \cdot \boldsymbol{t}'} \cdot (h_1^r, \ldots, h_\ell^r) \tag{3.4}$$

if we define $\boldsymbol{t}' = \boldsymbol{t} + (\omega \cdot r')^{-1} \cdot (\boldsymbol{y}_\beta - \boldsymbol{y}_{1-\beta}) \bmod q$. Note that, in all secret keys $\mathsf{sk}_{\boldsymbol{x}}$ involving vectors $\boldsymbol{x}$ such that $\langle \boldsymbol{x}, \boldsymbol{y}_\beta \rangle = \langle \boldsymbol{x}, \boldsymbol{y}_{1-\beta} \rangle$, we have $\langle \boldsymbol{t}', \boldsymbol{x} \rangle = \langle \boldsymbol{t}, \boldsymbol{x} \rangle$. Moreover, in each secret key $\mathsf{sk}_{\boldsymbol{x}} = (\langle \boldsymbol{s}, \boldsymbol{x} \rangle, \langle \boldsymbol{t}, \boldsymbol{x} \rangle)$, the information $\langle \boldsymbol{s}, \boldsymbol{x} \rangle$ is redundant with $\langle \boldsymbol{t}, \boldsymbol{x} \rangle$ since it is uniquely determined by $\langle \boldsymbol{t}, \boldsymbol{x} \rangle$ and $\prod_{i=1}^{\ell} h_i^{x_i}$. In fact, together with $\{h_i\}_{i=1}^{\ell}$, $\boldsymbol{t}'$ determines the vector $\boldsymbol{s}' = \boldsymbol{s} + \frac{1}{r'}(\boldsymbol{y}_{1-\beta} - \boldsymbol{y}_\beta) \bmod q$, which satisfies $h_i = g^{s_i'} \cdot h^{t_i'}$ for each $i \in \{1, \ldots, \ell\}$ and $\langle \boldsymbol{s}', \boldsymbol{x} \rangle = \langle \boldsymbol{s}, \boldsymbol{x} \rangle$ for any vector $\boldsymbol{x}$ that can be submitted to the key extraction oracle. It comes that situations (3.3) and (3.4) are equally likely in $\mathcal{A}$'s view as long as all revealed secret keys $\mathsf{sk}_{\boldsymbol{x}}$ involve vectors $\boldsymbol{x}$ such that $\langle \boldsymbol{x}, \boldsymbol{y}_0 \rangle = \langle \boldsymbol{x}, \boldsymbol{y}_1 \rangle$. We conclude that $\Pr[S_2] = 1/2$. In turn, this implies $\Pr[S_0] \leq \mathbf{Adv}_{\mathcal{B}}^{\mathrm{DDH}}(\lambda) + 1/2$, as claimed. □

4 Full security under the LWE assumption

We describe two LWE-based schemes: the first one for integer inner products of short integer vectors, the second one for inner products over a prime field $\mathbb{Z}_p$.

In both cases, the security relies on the hardness of a variant of the extended-LWE problem. The extended-LWE problem introduced by O'Neill, Peikert and Waters [47] and further investigated in [5, 17]. At a high level, the extended-LWE problem can be seen as $\mathsf{LWE}_{\alpha,q}$ with a fixed number m of samples, for which some extra information on the LWE noises is provided: the adversary is provided a given linear combination of the noise terms. More concretely, the problem is to distinguish between the distributions

$$\big(\mathbf{A},\ \mathbf{A}\cdot\boldsymbol{s}+\boldsymbol{e},\ \boldsymbol{z},\ \langle\boldsymbol{e},\boldsymbol{z}\rangle\big)\ \text{ and }\ \big(\mathbf{A},\ \boldsymbol{u},\ \boldsymbol{z},\ \langle\boldsymbol{e},\boldsymbol{z}\rangle\big),$$

where $\mathbf{A} \hookleftarrow \mathbb{Z}_q^{m\times n}, \boldsymbol{s} \hookleftarrow \mathbb{Z}_q^n, \boldsymbol{b} \hookleftarrow \mathbb{Z}_q^m, \boldsymbol{e} \hookleftarrow D_{\mathbb{Z},\alpha q}^m$, and $\boldsymbol{z}$ is sampled from a specified distribution. Note that in [47], a noise was added to the term $\langle\boldsymbol{e},\boldsymbol{z}\rangle$. The LWE to extended-LWE reductions from [5, 17] do not require such an extra noise term.

We will use a variant of extended-LWE for which multiple hints $(\boldsymbol{z}_i, \langle\boldsymbol{e},\boldsymbol{z}_i\rangle)$ are given, for the same noise vector $\boldsymbol{e}$.

Definition 4 (Multi-hint extended-LWE). *Let q, m, t be integers, α be a real and τ be a distribution over $\mathbb{Z}^{t\times m}$, all of them functions of a parameter n. The multi-hint extended-LWE problem $\mathsf{mheLWE}_{q,\alpha,m,t,\tau}$ is to distinguish between the distributions of the tuples*

$$\big(\mathbf{A},\ \mathbf{A}\cdot\boldsymbol{s}+\boldsymbol{e},\ \mathbf{Z},\ \mathbf{Z}\cdot\boldsymbol{e}\big)\ \textit{ and }\ \big(\mathbf{A},\ \boldsymbol{u},\ \mathbf{Z},\ \mathbf{Z}\cdot\boldsymbol{e}\big),$$

where $\mathbf{A} \hookleftarrow \mathbb{Z}_q^{m\times n}, \boldsymbol{s} \hookleftarrow \mathbb{Z}_q^n, \boldsymbol{u} \hookleftarrow \mathbb{Z}_q^m, \boldsymbol{e} \hookleftarrow D_{\mathbb{Z},\alpha q}^m$, and $\mathbf{Z} \hookleftarrow \tau$.

A reduction from LWE to mheLWE is presented in Subsection 4.3.

4.1 Integer inner products of short integer vectors

In the description hereunder, we consider the message space $\mathcal{P} = \{0,\ldots,P-1\}^\ell$, for some integer P and where $\ell \in \mathsf{poly}(n)$ denotes the dimension of vectors to encrypt. Secret keys are associated with vectors in $\mathcal{V} = \{0,\ldots,V-1\}^\ell$ for some integer V. As in the DDH case, inner products are evaluated over $\mathbb{Z}$. However, unlike our DDH-based construction, we can efficiently decrypt without confining inner product values within a small interval: here the inner product between the plaintext and key vectors belongs to $\{0,\ldots,K-1\}$ with $K = \ell PV$, and it is possible to set parameters so that the scheme is secure under standard hardness assumptions while K is more than polynomial in the security parameter. We compute ciphertexts using a prime modulus q, with q significantly larger than K.

$\mathsf{Setup}(1^n, 1^\ell, P, V)$: Set integers $m, q \geq 2$, real $\alpha \in (0,1)$ and distribution τ over $\mathbb{Z}^{\ell\times m}$ as explained below. Set $K = \ell PV$. Sample $\mathbf{A} \hookleftarrow \mathbb{Z}_q^{m\times n}$ and $\mathbf{Z} \hookleftarrow \tau$. Compute $\mathbf{U} = \mathbf{Z}\cdot\mathbf{A} \in \mathbb{Z}_q^{\ell\times n}$. Define $\mathsf{mpk} := (\mathbf{A},\mathbf{U},K,P,V)$ and $\mathsf{msk} := \mathbf{Z}$.

$\mathsf{Keygen}(\mathsf{msk}, \boldsymbol{x})$: Given a vector $\boldsymbol{x} \in \mathcal{V}$, compute and return the secret key $\boldsymbol{z}_{\boldsymbol{x}} := \boldsymbol{x}^T\cdot\mathbf{Z} \in \mathbb{Z}^m$.

$\mathsf{Encrypt}(\mathsf{mpk}, \boldsymbol{y})$**:** To encrypt a vector $\boldsymbol{y} \in \mathcal{P}$, sample $\boldsymbol{s} \hookleftarrow \mathbb{Z}_q^n$, $\boldsymbol{e}_0 \hookleftarrow D_{\mathbb{Z},\alpha q}^m$ and $\boldsymbol{e}_1 \hookleftarrow D_{\mathbb{Z},\alpha q}^\ell$ and compute

$$\boldsymbol{c}_0 = \mathbf{A} \cdot \boldsymbol{s} + \boldsymbol{e}_0 \ \in \mathbb{Z}_q^m,$$
$$\boldsymbol{c}_1 = \mathbf{U} \cdot \boldsymbol{s} + \boldsymbol{e}_1 + \left\lfloor \frac{q}{K} \right\rfloor \cdot \boldsymbol{y} \ \in \mathbb{Z}_q^\ell.$$

Then, return $C := (\boldsymbol{c}_0, \boldsymbol{c}_1)$.

$\mathsf{Decrypt}(\mathsf{mpk}, \boldsymbol{x}, \boldsymbol{z}_{\boldsymbol{x}}, C)$**:** Given $C := (\boldsymbol{c}_0, \boldsymbol{c}_1)$ and a secret key $\boldsymbol{z}_{\boldsymbol{x}}$ for $\boldsymbol{x} \in \mathcal{V}$, compute $\mu' = \langle \boldsymbol{x}, \boldsymbol{c}_1 \rangle - \langle \boldsymbol{z}_{\boldsymbol{x}}, \boldsymbol{c}_0 \rangle \bmod q$ and output the value $\mu \in \{-K + 1, \ldots, K-1\}$ that minimizes $|\lfloor \frac{q}{K} \rfloor \cdot \mu - \mu'|$.

Setting the parameters. Let B_τ be such that with probability $\geq 1 - n^{-\omega(1)}$, each row of sample from τ has norm $\leq B_\tau$. As explained just below, correctness may be ensured by setting

$$\alpha^{-1} \geq K^2 B_\tau \omega(\sqrt{\log n}) \quad \text{and} \quad q \geq \alpha^{-1}\omega(\sqrt{\log n}).$$

The choice of τ is driven by the reduction from LWE to mheLWE (as summarized in Theorem 4), and more precisely from Lemma 4 (another constraint arises from the use of Lemma 9 at the end of the security proof). We may choose $\tau = D_{\mathbb{Z},\sigma_1}^{\ell \times m/2} \times (D_{\mathbb{Z}^{m/2},\sigma_2,\boldsymbol{\delta}_1} \times \ldots \times D_{\mathbb{Z}^{m/2},\sigma_2,\boldsymbol{\delta}_\ell})$, where $\boldsymbol{\delta}_i \in \mathbb{Z}^\ell$ denotes the ith canonical vector, and the standard deviation parameters satisfy $\sigma_1 = \Theta(\sqrt{n \log m} \max(\sqrt{m}, K))$ and $\sigma_2 = \Theta(n^{7/2} m^{1/2} \max(m, K^2) \log^{5/2} m)$.

To ensure security based on $\mathsf{LWE}_{q,\alpha',m}$ in dimension $\geq c \cdot n$ for some $c \in (0,1)$ via Theorems 2 and 4 below, one may further impose that $\ell \leq (1-c) \cdot n$ and $m = \Theta(n \log q)$, to obtain $\alpha' = \Omega(\alpha / (n^6 K \log^2 q \log^{5/2} n))$. Note that $\mathsf{LWE}_{q,\alpha',m}$ enjoys reductions from lattice problems when $q \geq \Omega(\sqrt{n}/\alpha')$.

Combining the security and correctness requirements, we may choose $\alpha' = 1/((n \log q)^{O(1)} \cdot K^2)$ and $q = \Omega(\sqrt{n}/\alpha')$, resulting in LWE parameters that make LWE resist all known attacks running in time 2^λ, as long as $n \geq \widetilde{\Omega}(\lambda \log K)$.

Decryption correctness. To show the correctness of the scheme, we first observe that, modulo q:

$$\begin{aligned}\mu' &= \langle \boldsymbol{x}, \boldsymbol{c}_1 \rangle - \langle \boldsymbol{z}_{\boldsymbol{x}}, \boldsymbol{c}_0 \rangle \\ &= \lfloor q/K \rfloor \cdot \langle \boldsymbol{x}, \boldsymbol{y} \rangle + \langle \boldsymbol{x}, \boldsymbol{e}_1 \rangle - \langle \boldsymbol{z}_{\boldsymbol{x}}, \boldsymbol{e}_0 \rangle.\end{aligned}$$

Below, we show that the magnitude of the term $\langle \boldsymbol{x}, \boldsymbol{e}_1 \rangle - \langle \boldsymbol{z}_{\boldsymbol{x}}, \boldsymbol{e}_0 \rangle$ is $\leq \ell V B_\tau \alpha q \omega(\sqrt{\log n})$ with probability $\geq 1 - n^{-\omega(1)}$. Thanks to the choices of α and q, the latter upper bound is $\leq \lfloor q/K \rfloor / 4$, which suffices to guarantee decryption correctness.

Note that $\boldsymbol{e}_1$ is an integer Gaussian vector of dimension ℓ and standard deviation $\alpha q \geq \omega(\sqrt{\log n})$, and that $\|\boldsymbol{x}\| \leq \sqrt{\ell} V$. As a result, we have that $|\langle \boldsymbol{x}, \boldsymbol{e}_1 \rangle| \leq \sqrt{\ell} V \alpha q \omega(\sqrt{\log n})$ holds with probability $1 - n^{-\omega(1)}$. Similarly, as $\|\boldsymbol{z}_{\boldsymbol{x}}\| \leq \ell V B_\tau$, we obtain that $|\langle \boldsymbol{z}_{\boldsymbol{x}}, \boldsymbol{e}_0 \rangle| \leq \ell V B_\tau \alpha q \omega(\sqrt{\log n})$ holds with probability $1 - n^{-\omega(1)}$.

Full security. In order to prove the security of the scheme, we use the multi-hint extended-LWE from Definition 4.

Theorem 2. *Assume that $\ell \leq n^{O(1)}$, $m \geq 4n \log_2 q$, $q > \ell K^2$ and τ is as described above. Then the functional encryption scheme above is fully secure, under the $\mathsf{mheLWE}_{q,\alpha,m,\ell,\tau}$ hardness assumption.*

Proof. The proof proceeds with a sequence of games that starts with the real game and ends with a game in which the adversary's advantage is negligible. For each i, we call S_i the event that the adversary wins in Game i.

Game 0: This is the genuine full security game. Namely: the adversary $\mathcal{A}$ is given the master public key mpk; in the challenge phase, adversary $\mathcal{A}$ comes up with two distinct vectors $\boldsymbol{y}_0, \boldsymbol{y}_1 \in \mathcal{P}$ and receives an encryption C of $\boldsymbol{y}_\beta$ for $\beta \hookleftarrow \{0,1\}$ sampled by the challenger; when $\mathcal{A}$ halts, it outputs $\beta' \in \{0,1\}$ and S_0 is the event that $\beta' = \beta$. Note that any vector $\boldsymbol{x} \in \mathcal{V}$ queried by $\mathcal{A}$ to the secret key extraction oracle must satisfy $\langle \boldsymbol{x}, \boldsymbol{y}_0 \rangle = \langle \boldsymbol{x}, \boldsymbol{y}_1 \rangle$ over $\mathbb{Z}$ if $\mathcal{A}$ is a legitimate adversary.

Game 1: We modify the generation of $C = (\boldsymbol{c}_0, \boldsymbol{c}_1)$ in the challenge phase. Namely, at the outset of the game, the challenger picks $\boldsymbol{s} \hookleftarrow \mathbb{Z}_q^n$, $\boldsymbol{e}_0 \hookleftarrow D_{\mathbb{Z},\alpha q}^m$ (which may be chosen ahead of time) as well as $\mathbf{Z} \hookleftarrow \tau$. The master public key mpk is computed by setting $\mathbf{U} = \mathbf{Z} \cdot \mathbf{A} \bmod q$. In the challenge phase, the challenger picks a random bit $\beta \hookleftarrow \{0,1\}$ and encrypts $\boldsymbol{y}_\beta$ by computing (modulo q)

$$
\begin{aligned}
\boldsymbol{c}_0 &= \mathbf{A} \cdot \boldsymbol{s} + \boldsymbol{e}_0, \\
\boldsymbol{c}_1 &= \mathbf{Z} \cdot \boldsymbol{c}_0 - \mathbf{Z} \cdot \boldsymbol{e}_0 + \boldsymbol{e}_1 + \lfloor q/K \rfloor \cdot \boldsymbol{y}_\beta,
\end{aligned}
$$

with $\boldsymbol{e}_1 \hookleftarrow D_{\mathbb{Z},\alpha q}^\ell$. As the distribution of C is the same as in Game 0, we have $\Pr[S_1] = \Pr[S_0]$.

Game 2: We modify again the generation of $C = (\boldsymbol{c}_0, \boldsymbol{c}_1)$ in the challenge phase. Namely, the challenger picks $\boldsymbol{u} \hookleftarrow \mathbb{Z}_q^m$, sets $\boldsymbol{c}_0 = \boldsymbol{u}$ and computes $\boldsymbol{c}_1$ using $\boldsymbol{c}_0, \mathbf{Z}$ and $\boldsymbol{e}_0$ as in Game 1.

Under the mheLWE hardness assumption with $t = \ell$, this modification has no noticeable effect on the behavior of $\mathcal{A}$. Below, we prove that $\Pr[S_2] \approx 1/2$, which completes the proof of the theorem.

Let $\boldsymbol{x}^i \in \mathcal{V}$ be the vectors corresponding to the secret key queries made by $\mathcal{A}$. As $\mathcal{A}$ is a legitimate adversary, we have $\langle \boldsymbol{x}^i, \boldsymbol{y}_0 \rangle = \langle \boldsymbol{x}^i, \boldsymbol{y}_1 \rangle$ over $\mathbb{Z}$ for each secret key query $\boldsymbol{x}^i$. Let $g \neq 0$ be the gcd of the coefficients of $\boldsymbol{y}_1 - \boldsymbol{y}_0$ and define $\boldsymbol{y} = (y_1, \ldots, y_\ell) = \frac{1}{g}(\boldsymbol{y}_1 - \boldsymbol{y}_0)$. We have that $\langle \boldsymbol{x}^i, \boldsymbol{y} \rangle = 0$ (over $\mathbb{Z}$) for all i. Consider the lattice $\{\boldsymbol{x} \in \mathbb{Z}^\ell : \langle \boldsymbol{x}, \boldsymbol{y} \rangle = 0\}$: all the queries $\boldsymbol{x}^i$ must belong to that lattice. Without loss of generality, we assume the n_0 first entries of $\boldsymbol{y}$ are zero (for some n_0), and all remaining entries are non-zero. Further, the rows of the following matrix form a basis of a full-dimensional sublattice:

$$\mathbf{X}_{top} = \left(\begin{array}{c|ccccc} \mathbf{I}_{n_0} & & & & & \\ \hline & -y_{n_0+2} & y_{n_0+1} & & & \\ & & -y_{n_0+3} & y_{n_0+2} & & \\ & & & \ddots & \ddots & \\ & & & & y_\ell & y_{\ell-1} \end{array}\right) \in \mathbb{Z}^{(\ell-1)\times\ell}.$$

We may assume that through the secret key queries, the adversary learns exactly $\mathbf{X}_{top}\mathbf{Z}$, as all the queried vectors $\boldsymbol{x}^i$ can be obtained as rational combinations of the rows of $\mathbf{X}_{top}$.

Let $\mathbf{X}_{bot} = \boldsymbol{y}^T \in \mathbb{Z}^{1\times\ell}$. Consider the matrix $\mathbf{X} \in \mathbb{Z}_q^{\ell\times\ell}$ obtained by putting $\mathbf{X}_{top}$ on top of $\mathbf{X}_{bot}$. We claim that $\mathbf{X}$ is invertible modulo q. To see this, observe that

$$\mathbf{X}\mathbf{X}^T = \left(\begin{array}{c|ccccc} & & & & & y_1 \\ I & & & & & \vdots \\ & & & & & y_{n_0} \\ \hline & y_{n_0+1}^2 + y_{n_0+2}^2 & -y_{n_0+1}y_{n_0+3} & & & 0 \\ & -y_{n_0+1}y_{n_0+3} & y_{n_0+2}^2 + y_{n_0+3}^2 & \ddots & & 0 \\ & & \ddots & \ddots & \ddots & \vdots \\ & & & -y_{\ell-2}y_\ell & y_{\ell-1}^2 + y_\ell^2 & 0 \\ y_1 \dots y_{n_0} & 0 & 0 & \dots & 0 & \|\mathbf{y}\|^2 \end{array}\right)$$

It can be proved by induction that its determinant is

$$\det(\mathbf{X}\mathbf{X}^T) = (\prod_{k=n_0+2}^{\ell-1} y_k^2\,) \cdot (\sum_{k=n_0+1}^{\ell} y_k^2)^2.$$

As each of these y_k's is small and non-zero, they are all non-zero modulo q. Similarly, the integer $(\sum_{k=n_0+1}^{\ell} y_k^2)$ is non-zero and $< \ell P^2 < q$. This shows that $(\det \mathbf{X})^2 \neq 0 \bmod q$, which implies that $\mathbf{X}$ is invertible modulo q (here we used the fact that q is prime).

In Game 2, we have $\boldsymbol{c}_1 = \mathbf{Z}\boldsymbol{u} - \boldsymbol{f} + \lfloor q/K \rfloor \cdot \boldsymbol{y}_\beta$, with $\boldsymbol{f} := -\mathbf{Z}\boldsymbol{e}_0 + \boldsymbol{e}_1$. We write:

$$\boldsymbol{c}_1 = \mathbf{X}^{-1} \cdot \mathbf{X} \cdot (\mathbf{Z}\boldsymbol{u} - \boldsymbol{f} + \lfloor q/K \rfloor \cdot \boldsymbol{y}_\beta) \mod q.$$

We will show that the distribution of $\mathbf{X} \cdot \boldsymbol{c}_1 \bmod q$ is (almost) independent of β. As $\mathbf{X}$ is independent of β and invertible over $\mathbb{Z}_q$, this implies that the distribution of $\boldsymbol{c}_1$ is (almost) independent of β and $\Pr[S_2] \approx 1/2$.

The first $\ell - 1$ entries of $\mathbf{X} \cdot \boldsymbol{c}_1$ do not depend on β because $\mathbf{X}_{top} \cdot \boldsymbol{y}_0 = \mathbf{X}_{top} \cdot \boldsymbol{y}_1 \bmod q$.

It remains to prove that the last entry of $\mathbf{X} \cdot \boldsymbol{c}_1 \bmod q$ is (almost) independent of β. For this, we show that the residual distribution of $\mathbf{X}_{bot}\mathbf{Z}$ given the tuple $(\mathbf{A}, \mathbf{ZA}, \mathbf{X}_{top}\mathbf{Z})$ has high entropy. Using (a variant of) the leftover hash lemma with randomness $\mathbf{X}_{bot}\mathbf{Z}$ and seed $\boldsymbol{u}$, we will then conclude that given $(\mathbf{A}, \mathbf{ZA}, \mathbf{X}_{top}\mathbf{Z})$, the pair $(\boldsymbol{u}, \mathbf{X}_{bot}\mathbf{Z}\boldsymbol{u})$ is close to uniform and hence statistically hides $\lfloor q/K \rfloor \cdot \boldsymbol{y}_\beta$ in $\boldsymbol{c}_1$.

Write $\mathbf{A} = (\mathbf{A}_1^T | \mathbf{A}_2^T)^T$ with $\mathbf{A}_1, \mathbf{A}_2 \in \mathbb{Z}_q^{(m/2)\times n}$. Similarly, write $\mathbf{Z} = (\mathbf{Z}_1 | \mathbf{Z}_2)$ with $\mathbf{Z}_1, \mathbf{Z}_2 \in \mathbb{Z}_q^{\ell \times (m/2)}$. Recall that by construction, every entry of $\mathbf{Z}_1$ is independently sampled from a zero-centered integer Gaussian of standard deviation parameter $\sigma_1 = \Theta(\sqrt{n \log m} \max(\sqrt{m}, K))$. Further, every entry of $\mathbf{Z}_2$ is independently sampled from a (not zero-centered) integer Gaussian of standard deviation parameter σ_2 that is larger than σ_1.

Lemma 1. *Conditioned on $(\mathbf{A}, \mathbf{ZA}, \mathbf{X}_{top}\mathbf{Z}_1)$, the row vector $\mathbf{X}_{bot}\mathbf{Z}_1$ is distributed as $\boldsymbol{c} + D_{\|\boldsymbol{y}\|^2\mathbb{Z}^{m/2}, \|\boldsymbol{y}\|\sigma_1, -\boldsymbol{c}}$ for some vector $\boldsymbol{c}$ that depends only on $\mathbf{X}_{top}\mathbf{Z}_1$.*

Proof. Thanks to Lemma 9 in Appendix C, we have that $\mathbf{Z}_2\mathbf{A}_2$ is within $2^{-\Omega(n)}$ statistical distance to uniform. It hence statistically hides the term $\mathbf{Z}_1\mathbf{A}_1$ in $\mathbf{ZA} = \mathbf{Z}_1\mathbf{A}_1 + \mathbf{Z}_2\mathbf{A}_2$, and we obtain that given $(\mathbf{A}, \mathbf{ZA})$, the distribution of each entry of $\mathbf{Z}_1$ is still $D_{\mathbb{Z},\sigma_1}$.

Note that in $\mathbf{X}_{top}\mathbf{Z}_1$ and $\mathbf{X}_{bot}\mathbf{Z}_1$, matrices $\mathbf{X}_{top}$ and $\mathbf{X}_{bot}$ act in parallel on the columns of $\mathbf{Z}_1$. To prove the claim, it suffices to consider the distribution of $\mathbf{X}_{bot}\boldsymbol{z}$ conditioned on $\mathbf{X}_{top}\boldsymbol{z}$, with $\boldsymbol{z}$ sampled from $D_{\mathbb{Z}^\ell,\sigma_1}$. Let $\boldsymbol{b} = \mathbf{X}_{top}\boldsymbol{z} \in \mathbb{Z}^{\ell-1}$ and fix $\boldsymbol{z}_0 \in \mathbb{Z}^\ell$ arbitrary such that $\boldsymbol{b} = \mathbf{X}_{top}\boldsymbol{z}_0$. The distribution of $\boldsymbol{z}$ given that $\mathbf{X}_{top}\boldsymbol{z} = \boldsymbol{b}$ is $\boldsymbol{z}_0 + D_{\Lambda,\sigma_1,-\boldsymbol{z}_0}$, with $\Lambda = \{\boldsymbol{x} \in \mathbb{Z}^\ell : \mathbf{X}_{top}\boldsymbol{x} = \mathbf{0}\}$. By construction of $\mathbf{X}$, we have that $\Lambda = \mathbb{Z}\boldsymbol{y}$. As a result, the conditional distribution of $\mathbf{X}_{bot}\boldsymbol{z}$ is $c + D_{\|\boldsymbol{y}\|^2\mathbb{Z}, \|\boldsymbol{y}\|\sigma_1, -c}$ with $c = \langle \boldsymbol{y}, \boldsymbol{z}_0 \rangle \in \mathbb{Z}$. □

Now, write $\boldsymbol{u} = (\boldsymbol{u}_1^T | \boldsymbol{u}_2^T)^T$ with $\boldsymbol{u}_1, \boldsymbol{u}_2 \in \mathbb{Z}_q^{m/2}$. We have $\boldsymbol{X}_{bot}\mathbf{Z}\boldsymbol{u} = \boldsymbol{X}_{bot}\mathbf{Z}_1\boldsymbol{u}_1 + \boldsymbol{X}_{bot}\mathbf{Z}_2\boldsymbol{u}_2$. Thanks to the claim above and to Lemma 9, we obtain that the distribution of $(\boldsymbol{u}_1, \langle D_{\|\boldsymbol{y}\|^2\mathbb{Z}^{m/2}, \|\boldsymbol{y}\|\sigma_1, -\boldsymbol{c}}, \boldsymbol{u}_1 \rangle)$ is within $2^{-\Omega(n)}$ statistical distance to uniform (note that $D_{\|\boldsymbol{y}\|^2\mathbb{Z}^{m/2}, \|\boldsymbol{y}\|\sigma_1, -\boldsymbol{c}} = \|\boldsymbol{y}\|^2 \cdot D_{\mathbb{Z}^{m/2}, \sigma_1/\|\boldsymbol{y}\|, -\boldsymbol{c}/\|\boldsymbol{y}\|^2}$, that $\|\boldsymbol{y}\|^2$ is invertible modulo q and that $\sigma_1/\|\boldsymbol{y}\|$ satisfies the assumption of Lemma 9). This implies that given $(\mathbf{A}, \mathbf{ZA}, \mathbf{X}_{top}\mathbf{Z})$, the pair $(\boldsymbol{u}, \mathbf{X}_{bot}\mathbf{Z}\boldsymbol{u})$ is close to uniform, which completes the security proof. □

4.2 Inner products modulo a prime p

We now modify the LWE-based scheme above so that it enables secure functional encryption for inner products modulo prime p. The plaintext and key vectors now belong to $\mathbb{Z}_p^\ell$.

Note that the prior scheme evaluates inner products over the integers and is insecure if ported as is to the modulo p setting. To see this, consider the following simple attack in which the adversary requests a single key $\mathbf{x}$ so that inner product with the challenge messages $\mathbf{y}_0$ and $\mathbf{y}_1$ are different by a large multiple of p. Since the functionality posits that the inner product evaluations only agree modulo p, this is an admissible query. However, since decryption is performed over $\mathbb{Z}_q$

with q much larger than p, the adversary can easily distinguish. To prevent this attack, we scale the encrypted message by a factor of q/p (instead of $\lfloor q/K \rfloor$ as in the previous scheme): decryption modulo q forces arithmetic modulo p on the underlying plaintext.

A related difficulty in adapting the previous LWE-based scheme to modular inner products is the distribution of the noise component after inner product evaluation. Ciphertexts are manipulated modulo q, which internally manipulates plaintexts modulo p. If implemented naively, the carries of the plaintext computations may spill outside of the plaintext slots and bias the noise components of the ciphertexts. This may result in distinguishing attacks. To handle this, we take q a multiple of p. This adds some technical complications, as $\mathbb{Z}_q$ is hence not a field anymore.

A different attack is that the adversary may request keys for vectors that are linearly dependent modulo p but linearly independent modulo q. Note that with ℓ such queries, the attacker can recover the master secret key. To prevent this attack, we modify the scheme in that the authority is now stateful and keeps a record of all key queries made so far, so that it can make sure that key queries that are linearly dependent modulo p remain so modulo q. We also take q a power of p to simplify the implementation of this idea.

We note that for our application to bounded query FE for all circuits, all queries will be linearly independent modulo p, hence we will not require a stateful keygen. For details, see Section 6.

We now describe our functional encryption scheme for inner products modulo p.

Setup$(1^n, 1^\ell, p)$**:** Set integers $m, q = p^k$ for some integer k, real $\alpha \in (0,1)$ and distribution τ over $\mathbb{Z}^{\ell \times m}$ as explained below. Sample $\mathbf{A} \hookleftarrow \mathbb{Z}_q^{m \times n}$ and $\mathbf{Z} \hookleftarrow \tau$. Compute $\mathbf{U} = \mathbf{Z} \cdot \mathbf{A} \in \mathbb{Z}_q^{\ell \times n}$. Define $\mathsf{mpk} := (\mathbf{A}, \mathbf{U})$ and $\mathsf{msk} := \mathbf{Z}$.

Keygen$(\mathsf{msk}, \boldsymbol{x}, \mathsf{st})$**:** Given a vector $\boldsymbol{x} \in \mathbb{Z}_p^\ell$, and an internal state st, compute the secret key $\boldsymbol{z}_{\boldsymbol{x}}$ as follows. Recall that Keygen is a stateful algorithm with empty initial State st. At any point in the scheme execution, State st contains at most ℓ tuples $(\boldsymbol{x}_i, \overline{\boldsymbol{x}}_i, \boldsymbol{z}_i)$ where the $\boldsymbol{x}_i$'s are (a subset of the) key queries that have been made so far, and the $(\overline{\boldsymbol{x}}_i, \boldsymbol{z}_i)$'s are the corresponding secret keys. If $\boldsymbol{x}$ is linearly independent from the $\boldsymbol{x}_i$'s modulo p, set $\overline{\boldsymbol{x}} = \boldsymbol{x} \in \mathbb{Z}^\ell$ (with coefficients in $[0,p)$), $\boldsymbol{z}_{\boldsymbol{x}} = \overline{\boldsymbol{x}}^T \cdot \mathbf{Z} \in \mathbb{Z}^m$ and add $(\boldsymbol{x}, \overline{\boldsymbol{x}}, \boldsymbol{z}_{\boldsymbol{x}})$ to st. If $\boldsymbol{x} = \sum_i k_i \boldsymbol{x}_i \bmod p$ for some k_i's in $[0,p)$, then set $\overline{\boldsymbol{x}} = \sum_i k_i \overline{\boldsymbol{x}}_i \in \mathbb{Z}^\ell$ and $\boldsymbol{z}_{\boldsymbol{x}} = \sum_i k_i \boldsymbol{z}_i \in \mathbb{Z}^m$. In both cases, return $(\overline{\boldsymbol{x}}, \boldsymbol{z}_{\boldsymbol{x}})$.

Encrypt$(\mathsf{mpk}, \boldsymbol{y})$**:** To encrypt a vector $\boldsymbol{y} \in \mathbb{Z}_p^\ell$, sample $\boldsymbol{s} \hookleftarrow \mathbb{Z}_q^n$, $\boldsymbol{e}_0 \hookleftarrow D_{\mathbb{Z},\alpha q}^m$ and $\boldsymbol{e}_1 \hookleftarrow D_{\mathbb{Z},\alpha q}^\ell$ and compute

$$\boldsymbol{c}_0 = \mathbf{A} \cdot \boldsymbol{s} + \boldsymbol{e}_0 \in \mathbb{Z}_q^m,$$
$$\boldsymbol{c}_1 = \mathbf{U} \cdot \boldsymbol{s} + \boldsymbol{e}_1 + p^{k-1} \cdot \boldsymbol{y} \in \mathbb{Z}_q^\ell.$$

Then, return $C := (\boldsymbol{c}_0, \boldsymbol{c}_1)$.

Decrypt(mpk, $(\overline{\boldsymbol{x}}, \boldsymbol{z}_{\boldsymbol{x}}), C$)**:** Given $C := (\boldsymbol{c}_0, \boldsymbol{c}_1)$ and a secret key $(\overline{\boldsymbol{x}}, \boldsymbol{z}_{\boldsymbol{x}})$ for $\boldsymbol{x} \in \mathbb{Z}_p^\ell$, compute $\mu' = \langle \overline{\boldsymbol{x}}, \boldsymbol{c}_1 \rangle - \langle \boldsymbol{z}_{\boldsymbol{x}}, \boldsymbol{c}_0 \rangle \bmod q$ and output the value $\mu \in \mathbb{Z}_p$ that minimizes $|p^{k-1} \cdot \mu - \mu'|$.

Decryption correctness. Correctness derives from the following observation:

$$\begin{aligned}\mu' &= \langle \overline{\boldsymbol{x}}, \boldsymbol{c}_1 \rangle - \langle \boldsymbol{z}_{\boldsymbol{x}}, \boldsymbol{c}_0 \rangle \\ &= p^{k-1} \cdot (\langle \boldsymbol{x}, \boldsymbol{y} \rangle \bmod p) + \langle \overline{\boldsymbol{x}}, \boldsymbol{e}_1 \rangle - \langle \boldsymbol{z}_{\boldsymbol{x}}, \boldsymbol{e}_0 \rangle \bmod q.\end{aligned}$$

By adapting the proof of the first LWE-based scheme, we can show that the magnitude of the term $\langle \overline{\boldsymbol{x}}, \boldsymbol{e}_1 \rangle - \langle \boldsymbol{z}_{\boldsymbol{x}}, \boldsymbol{e}_0 \rangle$ is $\leq \ell^2 p^2 B_\tau \alpha q \omega(\sqrt{\log n})$ with probability $\geq 1 - n^{-\omega(1)}$. This follows from the bound $\|\boldsymbol{z}_{\boldsymbol{x}}\| \leq \ell \|\overline{\boldsymbol{x}}\| \leq \ell^2 p^2 B_\tau$.

Setting the parameters. The main difference with the previous LWE-based scheme with respect to parameter conditions is the choice of q of the form $q = p^k$ instead of q prime. As explained just above, correctness may be ensured by setting

$$\alpha^{-1} \geq \ell^2 p^3 B_\tau \omega(\sqrt{\log n}) \quad \text{and} \quad q \geq \alpha^{-1} \omega(\sqrt{\log n}).$$

The choice of τ is driven by Lemma 2 below (the proof requires that σ_1 is large) and the reduction from LWE to mheLWE (as summarized in Theorem 4), and more precisely from Lemma 4. We may choose $\tau = D_{\mathbb{Z},\sigma_1}^{\ell \times m/2} \times (D_{\mathbb{Z}^{m/2},\sigma_2,\boldsymbol{\delta}_1} \times \ldots \times D_{\mathbb{Z}^{m/2},\sigma_2,\boldsymbol{\delta}_\ell})$, where $\boldsymbol{\delta}_i \in \mathbb{Z}^\ell$ denotes the ith canonical vector, and the standard deviation parameters satisfy $\sigma_1 = \Theta(\sqrt{n \log m} \max(\sqrt{m}, K'))$ and $\sigma_2 = \Theta(n^{7/2} m^{1/2} \max(m, K'^2) \log^{5/2} m)$, with $K' = (\sqrt{\ell} p)^{\ell-1}$.

To ensure security based on $\mathsf{LWE}_{q,\alpha',m}$ in dimension $\geq c \cdot n$ for some $c \in (0,1)$ via Theorems 2 and 4 below, one may further impose that $\ell \leq (1-c) \cdot n$ and $m = \Theta(n \log q)$, to obtain $\alpha' = \Omega(\alpha / (n^6 K' \log^2 q \log^{5/2} n))$. Remember that $\mathsf{LWE}_{q,\alpha',m}$ enjoys reductions from lattice problems when $q \geq \Omega(\sqrt{n}/\alpha')$.

Note that the parameter conditions make the scheme efficiency degrade quickly when ℓ increases, as K' is exponential in ℓ. Assume that $p \leq n^{O(1)}$ and $\ell = \Omega(\log n)$. Then σ_1, σ_2, $1/\alpha$, $1/\alpha'$ and q can all be set as $2^{\widetilde{O}(\ell)}$. To maintain security against all $2^{o(\lambda)}$ attacks, one may set $n = \widetilde{\Theta}(\ell \lambda)$.

Theorem 3. *Assume that $\ell \leq n^{O(1)}$, $m \geq 4n \log_2 q$ and τ is as described above. Then the sateful functional encryption scheme above is fully secure, under the $\mathsf{mheLWE}_{q,\alpha,m,\ell,\tau}$ hardness assumption.*

Proof. The sequence of games in the proof of Theorem 2 can be adapted to the modified scheme. The main difficulty is to show that in the adapted version of the last game, the winning probability is close to 1/2. Let us recall that game in detail.

Game $2'$: At the outset of the game, the challenger picks $\boldsymbol{s} \hookleftarrow \mathbb{Z}_q^n$, $\boldsymbol{e}_0 \hookleftarrow D_{\mathbb{Z},\alpha q}^m$ as well as $\mathbf{Z} \hookleftarrow \tau$. The master public key mpk is computed by setting $\mathbf{U} = \mathbf{Z} \cdot \mathbf{A} \bmod q$ and is provided to the adversary. In the challenge phase, adversary $\mathcal{A}$ comes up with two distinct vectors $\boldsymbol{y}_0, \boldsymbol{y}_1 \in \mathbb{Z}_p^\ell$. The challenger picks a random bit $\beta \hookleftarrow \{0,1\}$, $\boldsymbol{u} \hookleftarrow \mathbb{Z}_q^m$ and encrypts $\boldsymbol{y}_\beta$ by computing (modulo q)

$$\begin{aligned}\boldsymbol{c}_0 &= \boldsymbol{u}, \\ \boldsymbol{c}_1 &= \mathbf{Z} \cdot \boldsymbol{c}_0 - \mathbf{Z} \cdot \boldsymbol{e}_0 + \boldsymbol{e}_1 + p^{k-1} \cdot \boldsymbol{y}_\beta,\end{aligned}$$

with $\boldsymbol{e}_1 \hookleftarrow D^\ell_{\mathbb{Z},\alpha q}$. Note that any vector $\boldsymbol{x} \in \mathbb{Z}_p^\ell$ queried by $\mathcal{A}$ to the secret key extraction oracle must satisfy $\langle \boldsymbol{x}, \boldsymbol{y}_0 \rangle = \langle \boldsymbol{x}, \boldsymbol{y}_1 \rangle \bmod p$ if $\mathcal{A}$ is a legitimate adversary. Adversary $\mathcal{A}$ is then given a secret key $(\overline{\boldsymbol{x}}, \boldsymbol{x}_{\boldsymbol{z}})$ as in the real scheme. When $\mathcal{A}$ halts, it outputs $\beta' \in \{0,1\}$ and wins in the event that $\beta' = \beta$.

Define $\boldsymbol{y} = \boldsymbol{y}_1 - \boldsymbol{y}_0 \in \mathbb{Z}_p^\ell$. Let $\boldsymbol{x}^i \in \mathbb{Z}_p^\ell$ be the vectors corresponding to the secret key queries made by $\mathcal{A}$. As $\mathcal{A}$ is a legitimate adversary, we have $\langle \boldsymbol{x}^i, \boldsymbol{y} \rangle = 0 \bmod p$ for each secret key query $\boldsymbol{x}^i$.

We consider the view of the adversary after it has made exactly j key queries that are linearly independent modulo p, for each j from 0 up to $\ell - 1$. In fact, counter j may stop increasing before reaching $\ell - 1$, but without loss of generality, we may assume that it eventually reaches $\ell - 1$. We are to show by induction that for any j, the view of the adversary is almost independent of β. In particular, for all $j < \ell - 1$, this implies that the $(j+1)$th linearly independent key query is almost (statistically) independent of β. It also implies, for $j = \ell - 1$, that the adversary's view through Game $2'$ is almost independent of β, which is exactly what we are aiming for. In what follows, we take $j \in \{0, \ldots, \ell - 1\}$, and assume that state st is independent from β.

At this stage, state st contains j tuples $(\boldsymbol{x}_i, \overline{\boldsymbol{x}}_i, \boldsymbol{z}_i)$. The $\boldsymbol{x}_i$'s form a basis of a vector subspace of the $(\ell - 1)$-dimensional mod-p vector space $\boldsymbol{y}^\perp := \{\boldsymbol{x} \in \mathbb{Z}_p^\ell : \langle \boldsymbol{x}, \boldsymbol{y} \rangle = 0 \bmod p\}$. We extend the $\boldsymbol{x}_i$'s into a basis of $\boldsymbol{y}^\perp$ that is statisticaly independent from β. A way to interpret this is to imagine that the challenger samples itself dummy key queries $(\boldsymbol{x}_i)_{i \in \{j+1,\ldots,\ell-1\}}$ to get a full basis, and creates the corresponding $\overline{\boldsymbol{x}}_i$'s in $\mathbb{Z}^\ell$ (note that the basis of $\boldsymbol{y}^\perp$ only needs to exist, so that the reduction does not have to know it at the beginning of the game). We define $\mathbf{X}_{top} \in \mathbb{Z}^{(\ell-1)\times\ell}$ as the matrix whose ith row is $\overline{\boldsymbol{x}}_i$, for all i (including the genuine and dummy keys). Through the secret key queries, the adversary learns at most $\mathbf{X}_{top}\mathbf{Z} \in \mathbb{Z}^{(\ell-1)\times m}$.

Let $\boldsymbol{x}' \in \mathbb{Z}_p^\ell$ be a vector that does not belong to $\boldsymbol{y}^\perp$, and $\mathbf{X}_{bot} \in \mathbb{Z}^{1\times\ell}$ be the canonical lift of $(\boldsymbol{x}')^T$ over the integers. Consider the matrix $\mathbf{X} \in \mathbb{Z}^{\ell\times\ell}$ obtained by putting $\mathbf{X}_{top}$ on top of $\mathbf{X}_{bot}$. By construction, matrix $\mathbf{X}$ is invertible modulo p, and hence modulo $q = p^k$. Also, by induction and construction, matrix $\mathbf{X}$ is statistically independent from β.

In Game $2'$, we have $\boldsymbol{c}_1 = \mathbf{Z}\boldsymbol{u} - \boldsymbol{f} + p^{k-1} \cdot \boldsymbol{y}_\beta$, with $\boldsymbol{f} := -\mathbf{Z}\boldsymbol{e}_0 + \boldsymbol{e}_1$. We write:

$$\boldsymbol{c}_1 = \mathbf{X}^{-1} \cdot \mathbf{X} \cdot \left(\mathbf{Z}\boldsymbol{u} - \boldsymbol{f} + p^{k-1} \cdot \boldsymbol{y}_\beta\right) \bmod q.$$

We will show that the distribution of $\mathbf{X} \cdot \boldsymbol{c}_1 \bmod q$ is (almost) independent of β. As $\mathbf{X}$ is independent of β and invertible over $\mathbb{Z}_q$, this implies that the distribution of $\boldsymbol{c}_1$ is (almost) independent of β and the winning probability in Game $2'$ is negligibly close to $1/2$.

The first $\ell - 1$ entries of $\mathbf{X} \cdot \boldsymbol{c}_1$ do not depend on β because $p^{k-1} \cdot \mathbf{X}_{top} \cdot \boldsymbol{y}_0 = p^{k-1} \cdot \mathbf{X}_{top} \cdot \boldsymbol{y}_1 \bmod q$.

It remains to prove that the last entry of $\mathbf{X} \cdot \boldsymbol{c}_1 \bmod q$ is (almost) independent of β. Write $\mathbf{A} = (\mathbf{A}_1^T | \mathbf{A}_2^T)^T$ with $\mathbf{A}_1, \mathbf{A}_2 \in \mathbb{Z}_q^{(m/2) \times n}$. Similarly, write $\mathbf{Z} = (\mathbf{Z}_1 | \mathbf{Z}_2)$ with $\mathbf{Z}_1, \mathbf{Z}_2 \in \mathbb{Z}^{\ell \times (m/2)}$. Recall that by construction, every entry of $\mathbf{Z}_1$ is independently sampled from a zero-centered integer Gaussian of standard deviation parameter $\sigma_1 = \Theta(\sqrt{n \log m} \max(\sqrt{m}, K'))$ with $K' = (\sqrt{\ell} p)^{\ell - 1}$. Further, every entry of $\mathbf{Z}_2$ is independently sampled from a (not zero-centered) integer Gaussian of standard deviation parameter σ_2 that is larger than σ_1.

Lemma 2. *Conditioned on* $(\mathbf{A}, \mathbf{ZA}, \mathbf{X}_{top}\mathbf{Z}_1)$*, the row vector* $\mathbf{X}_{bot}\mathbf{Z}_1 \bmod p$ *is within negligible statistical distance from the uniform distribution over* $\mathbb{Z}_p^{m/2}$*.*

Proof. Thanks to Lemma 9, we have that $\mathbf{Z}_2\mathbf{A}_2$ is within $2^{-\Omega(n)}$ statistical distance to uniform over $\mathbb{Z}_q^{(\ell-1) \times m}$. It hence statistically hides the term $\mathbf{Z}_1\mathbf{A}_1$ in $\mathbf{ZA} = \mathbf{Z}_1\mathbf{A}_1 + \mathbf{Z}_2\mathbf{A}_2 \bmod q$, and we obtain that given $(\mathbf{A}, \mathbf{ZA})$, the distribution of each entry of $\mathbf{Z}_1$ is still $D_{\mathbb{Z}, \sigma_1}$.

Note that in $\mathbf{X}_{top}\mathbf{Z}_1$ and $\mathbf{X}_{bot}\mathbf{Z}_1$, matrices $\mathbf{X}_{top}$ and $\mathbf{X}_{bot}$ act in parallel on the columns of $\mathbf{Z}_1$. To prove the claim, it suffices to consider the distribution of $\mathbf{X}_{bot}\boldsymbol{z}$ conditioned on $\mathbf{X}_{top}\boldsymbol{z}$, with $\boldsymbol{z}$ sampled from $D_{\mathbb{Z}^\ell, \sigma_1}$. Let $\boldsymbol{b} = \mathbf{X}_{top}\boldsymbol{z} \in \mathbb{Z}^{\ell - 1}$ and fix $\boldsymbol{z}_0 \in \mathbb{Z}^\ell$ arbitrary such that $\boldsymbol{b} = \mathbf{X}_{top}\boldsymbol{z}_0$. The distribution of $\boldsymbol{z}$ given that $\mathbf{X}_{top}\boldsymbol{z} = \boldsymbol{b}$ is $\boldsymbol{z}_0 + D_{\Lambda, \sigma_1, -\boldsymbol{z}_0}$, with $\Lambda = \{\boldsymbol{x} \in \mathbb{Z}^\ell : \mathbf{X}_{top}\boldsymbol{x} = \mathbf{0}\}$ (where the equality holds over the integers). Note that Λ is a 1-dimensional lattice in $\mathbb{Z}^\ell$.

We can write $\Lambda = \boldsymbol{y}' \cdot \mathbb{Z}$, for some $\boldsymbol{y}' \in \mathbb{Z}^\ell$. Note that there exists $\alpha \in \mathbb{Z}_p \setminus \{0\}$ such that $\boldsymbol{y}' = \alpha \cdot \boldsymbol{y} \bmod p$: otherwise, the vector $\boldsymbol{y}'/p$ would belong to $\Lambda \setminus \boldsymbol{y}' \cdot \mathbb{Z}$, contradicting the definition of $\boldsymbol{y}'$. Further, we have $\|\boldsymbol{y}'\| = \det \Lambda \leq \det \Lambda'$, where Λ' is the lattice spanned by the rows of $\mathbf{X}^{top}$ (see, e.g., [45], for properties on orthogonal lattices). Hadamard's bound implies that $\|\overline{\boldsymbol{y}}\| \leq (\sqrt{\ell} p)^{\ell - 1}$.

By Lemma 7, the fact that $\sigma_1 \geq \sqrt{n}(\sqrt{\ell} p)^{\ell - 1}$ implies that the distribution $(D_{\Lambda, \sigma_1, -\boldsymbol{z}_0} \bmod p\Lambda)$ is within $2^{-\Omega(n)}$ statistical distance from the uniform distribution over $\Lambda / p\Lambda \simeq \boldsymbol{y}\mathbb{Z}_p$. We conclude that the conditional distribution of $(\mathbf{X}_{bot}\boldsymbol{z} \bmod p)$ is within exponentially small statistical distance from the uniform distribution over $\mathbb{Z}_p$ (here we use the facts that p is prime and that $\mathbf{X}_{bot}\boldsymbol{y} \neq 0 \bmod p$, by construction of $\mathbf{X}_{bot}$). ☐

Now, write $\boldsymbol{u} = (\boldsymbol{u}_1^T | \boldsymbol{u}_2^T)^T$ with $\boldsymbol{u}_1, \boldsymbol{u}_2 \in \mathbb{Z}_q^{m/2}$. We have $\boldsymbol{X}_{bot}\mathbf{Z}\boldsymbol{u} = \boldsymbol{X}_{bot}\mathbf{Z}_1\boldsymbol{u}_1 + \boldsymbol{X}_{bot}\mathbf{Z}_2\boldsymbol{u}_2$. Thanks to Lemmas 2 and 10 (a variant of the leftover hash lemma modulo $q = p^k$), we obtain that conditioned on $(\mathbf{A}, \mathbf{ZA}, \mathbf{X}_{top}\mathbf{Z})$, the distribution of $(\boldsymbol{u}_1, \mathbf{X}_{bot}\mathbf{Z}_1\boldsymbol{u}_1)$ is within $2^{-\Omega(n)}$ statistical distance to uniform modulo q (here we used the assumption that $m \geq k + n/(\log p)$). This implies that given $(\mathbf{A}, \mathbf{ZA}, \mathbf{X}_{top}\mathbf{Z})$, the pair $(\boldsymbol{u}, \mathbf{X}_{bot}\mathbf{Z}\boldsymbol{u})$ is close to uniform, which completes the security proof. ☐

4.3 Hardness of multi-hint extended-LWE

In this section, we prove the following theorem, which shows that for some parameters, the mheLWE problem is no easier than the LWE problem.

Theorem 4. *Let $n \geq 100$, $q \geq 2$, $t < n$ and m with $m = \Omega(n \log n)$ and $m \leq n^{O(1)}$. There exists $\xi \leq O(n^4 m^2 \log^{5/2} n)$ and a distribution τ over $\mathbb{Z}^{t \times m}$ such that the following statements hold:*

- *There is a reduction from $\mathsf{LWE}_{q,\alpha,m}$ in dimension $n-t$ to $\mathsf{mheLWE}_{q,\alpha\xi,m,t,\tau}$ that reduces the advantage by at most $2^{\Omega(t-n)}$,*
- *It is possible to sample from τ in time polynomial in n,*
- *Each entry of matrix τ is an independent discrete Gaussian $\tau_{i,j} = D_{\mathbb{Z},\sigma_{i,j},\mathbf{c}_{i,j}}$ for some $\mathbf{c}_{i,j}$ and $\sigma_{i,j} \geq \Omega(\sqrt{mn \log m})$,*
- *With probability $\geq 1 - n^{-\omega(1)}$, all rows from a sample from τ have norms $\leq \xi$.*

Our reduction from LWE to mheLWE proceeds as the reduction from LWE to extended-LWE from [17], using the matrix gadget from [42] to handle the multiple hints. We first reduce LWE to the following variant of LWE in which the first samples are noise-free. This problem generalizes the first-is-errorless LWE problem from [17].

Definition 5 (First-are-errorless LWE). *Let q, α, m, t be functions of a parameter n. The first-are-errorless LWE problem $\mathsf{faeLWE}_{q,\alpha,m,t}$ is defined as follows: For $\mathbf{s} \hookleftarrow \mathbb{Z}_q^n$, the goal is to distinguish between the following two scenarios. In the first, all m samples are uniform over $\mathbb{Z}_q^n \times \mathbb{Z}_q$. In the second, the first t samples are from $A_{q,\{0\},\boldsymbol{s}}$ (where $\{0\}$ denotes the distribution that is deterministically zero) and the rest are from $A_{q,\alpha,\boldsymbol{s}}$.*

Lemma 3. *For any $n > t$, $m, q \geq 2$, and $\alpha \in (0,1)$, there is an efficient reduction from $\mathsf{LWE}_{q,\alpha,m}$ in dimension $n-t$ to $\mathsf{faeLWE}_{q,\alpha,m,t}$ in dimension n that reduces the advantage by at most 2^{-n+t+1}.*

The proof, postponed to the appendices, is a direct adaptation of the one of [17, Le. 4.3].

In our reduction from faeLWE to mheLWE, we use the following gadget matrix from [42, Cor. 10]. It generalizes the matrix construction from [17, Claim 4.6].

Lemma 4. *Let n, m_1, m_2 with $100 \leq n \leq m_1 \leq m_2 \leq n^{O(1)}$. Let $\sigma_1, \sigma_2 > 0$ be standard deviation parameters such that $\sigma_1 \geq \Omega(\sqrt{m_1 n \log m_1})$, $m_1 \geq \Omega(n \log(\sigma_1 n))$ and $\sigma_2 \geq \Omega(n^{5/2}\sqrt{m_1}\sigma_1^2 \log^{3/2}(m_1\sigma_1))$. Let $m = m_1 + m_2$. There exists a probabilistic polynomial time algorithm that given n, m_1, m_2 (in unary) and σ_1, σ_2 as inputs, outputs $\mathbf{G} \in \mathbb{Z}^{m \times m}$ such that:*

- *The top $n \times m$ submatrix of $\mathbf{G}$ is within statistical distance $2^{-\Omega(n)}$ of $\tau = D_{\mathbb{Z},\sigma_1}^{n \times m_1} \times (D_{\mathbb{Z}^{m_2},\sigma_2,\boldsymbol{\delta}_1} \times \ldots \times D_{\mathbb{Z}^{m_2},\sigma_2,\boldsymbol{\delta}_n})^T$ with $\boldsymbol{\delta}_i$ denoting the ith canonical unit vector,*
- *We have $|\det(\mathbf{G})| = 1$ and $\|\mathbf{G}^{-1}\| \leq O(\sqrt{nm_2}\sigma_2)$, with probability $\geq 1 - 2^{-\Omega(n)}$.*

Lemma 5. *Let $n, m_1, m_2, m, \sigma_1, \sigma_2, \tau$ be as in Lemma 4, and $\xi \geq \Omega(\sqrt{nm_2}\sigma_2)$. Let $q \geq 2$, $t \leq n$, $\alpha \geq \Omega(\sqrt{n}/q)$. Let τ_t be the distribution obtained by keeping only the first t rows from a sample from τ. There is a (dimension-preserving) reduction from $\mathsf{faeLWE}_{q,\alpha,m,t}$ to $\mathsf{mheLWE}_{q,2\alpha\xi,m,t,\tau_t}$ that reduces the advantage by at most $2^{-\Omega(n)}$.*

Proof. Let us first describe the reduction. Let $(\mathbf{A}, \boldsymbol{b}) \in \mathbb{Z}_q^m \times \mathbb{Z}_q$ be the input, which is either sampled from the uniform distribution, or from distribution $A^t_{q,\{0\},\boldsymbol{s}} \times A^{m-t}_{q,\alpha,\boldsymbol{s}}$ for some fixed $\boldsymbol{s} \hookleftarrow \mathbb{Z}_q^n$. Our objective is to distinguish between the two scenarios, using an mheLWE oracle. We compute $\mathbf{G}$ as in Lemma 4 and let $\mathbf{U} = \mathbf{G}^{-1}$. We let $\mathbf{Z} \in \mathbb{Z}^{t\times m}$ denote the matrix formed by the top t rows of $\mathbf{G}$, and let $\mathbf{U}' \in \mathbb{Z}^{m\times(m-t)}$ denote the matrix formed by the right $m-t$ columns of $\mathbf{U}$. By construction, we have $\mathbf{ZU}' = \mathbf{0}$. We define $\mathbf{A}' = \mathbf{U}\cdot\mathbf{A} \bmod q$. We sample $\boldsymbol{f} \hookleftarrow D_{\alpha q(\xi^2\mathbf{I}-\mathbf{U}'\mathbf{U}'^T)^{1/2}}$ (thanks to Lemma 4 and the choice of ξ, the matrix $\xi^2\mathbf{I} - \mathbf{U}'\mathbf{U}'^T$ is positive definite). We sample $\boldsymbol{e}'$ from $\{0\}^t \times D^{m-t}_{\alpha q}$ and define $\boldsymbol{b}' = \mathbf{U}\cdot(\boldsymbol{b}+\boldsymbol{e}') + \boldsymbol{f}$. We then sample $\boldsymbol{c} \hookleftarrow D_{\mathbb{Z}^m-\boldsymbol{b}',\sqrt{2}\alpha\xi q}$, and define $\boldsymbol{h} = \mathbf{Z}(\boldsymbol{f}+\boldsymbol{c})$.

Finally, the reduction calls the mheLWE oracle on input $(\mathbf{A}', \boldsymbol{b}'+\boldsymbol{c}, \mathbf{Z}, \boldsymbol{h})$, and outputs the reply.

Correctness is obtained by showing that distribution $A^t_{q,\{0\},\boldsymbol{s}} \times A^{m-t}_{q,\alpha,\boldsymbol{s}}$ is mapped to the mheLWE "LWE" distribution and that the uniform distribution is mapped to the mheLWE "uniform" distribution, up to $2^{-\Omega(n)}$ statistical distances (we do not discuss these tiny statistical discrepancies below). The proof is identical to the reduction analysis in the proof of [17, Le. 4.7]. □

Theorem 4 is obtained by combining Lemmas 3, 4 and 5.

5 Constructions Based on Paillier

In this section, we show how to remove the main limitation of our DDH-based system which is its somewhat expensive decryption algorithm. To this end, we use Paillier's cryptosystem [48] and the property that, for an RSA modulus $N = pq$, the multiplicative group $\mathbb{Z}^*_{N^2}$ contains a subgroup of order N (generated by $(N+1)$) in which the discrete logarithm problem is easy. We also rely on the observation [19, 18] that combining the Paillier and Elgamal encryption schemes makes it possible to decrypt without knowing the factorization of $N = pq$.

5.1 Computing Inner Products over $\mathbb{Z}$

In the following scheme, key vectors $\boldsymbol{x}$ and message vectors $\boldsymbol{y}$ are assumed to be of bounded norm $\|\boldsymbol{x}\| \leq X$ and $\|\boldsymbol{y}\| \leq Y$, respectively. The bounds X and Y are chosen so that $X \cdot Y < N$, where N is the composite modulus of Paillier's cryptosystem. Decryption allows to recover $\langle \boldsymbol{x}, \boldsymbol{y}\rangle \bmod N$, which is exactly $\langle \boldsymbol{x}, \boldsymbol{y}\rangle$ over the integers, thanks to the norm bounds.

$\mathsf{Setup}(1^\lambda, 1^\ell, X, Y)$: Choose safe prime numbers $p = 2p'+1$, $q = 2q'+1$ with sufficiently large primes $p', q' > 2^{l(\lambda)}$, for some polynomial l, and compute $N = pq > XY$. Then, sample $g' \hookleftarrow \mathbb{Z}^*_{N^2}$ and compute $g = g'^{2N} \bmod N^2$, which generates the subgroup of $(2N)$th residues in $\mathbb{Z}^*_{N^2}$ with overwhelming probability. Then, for each $i \in \{1, \ldots, \ell\}$, sample

$s_i \hookleftarrow \{-2^{\lambda+1}N^4, \ldots, 2^{\lambda+1}N^4\}$ and compute $h_i = g^{s_i} \bmod N^2$. Define

$$\mathsf{mpk} := \Big(N, g, \{h_i\}_{i=1}^{\ell}, Y\Big)$$

and $\mathsf{msk} := (\{s_i\}_{i=1}^{\ell}, X)$. The prime numbers p, p', q, q' are no longer needed.

Keygen($\mathsf{msk}, \boldsymbol{x}$)**:** To generate a key for the vector $\boldsymbol{x} = (x_1, \ldots, x_\ell) \in \mathbb{Z}^\ell$ with $\|\boldsymbol{x}\| \le X$, compute $\mathsf{sk}_{\boldsymbol{x}} = \sum_{i=1}^{\ell} s_i \cdot x_i$ over $\mathbb{Z}$.

Encrypt($\mathsf{mpk}, \boldsymbol{y}$)**:** To encrypt a vector $\boldsymbol{y} = (y_1, \ldots, y_\ell) \in \mathbb{Z}^\ell$ with $\|\boldsymbol{y}\| \le Y$, sample $r \hookleftarrow \{0, \ldots, N/4\}$ and compute

$$\begin{aligned} C_0 &= g^r \bmod N^2, \\ C_i &= (1 + y_i N) \cdot h_i^r \bmod N^2, \qquad\qquad \forall i \in \{1, \ldots, \ell\}. \end{aligned}$$

Return $C_{\boldsymbol{y}} = (C_0, C_1, \ldots, C_\ell) \in \mathbb{Z}_{N^2}^{\ell+1}$.

Decrypt($\mathsf{mpk}, \mathsf{sk}_{\boldsymbol{x}}, C_{\boldsymbol{y}}$)**:** Given $\mathsf{sk}_{\boldsymbol{x}} \in \mathbb{Z}$, compute

$$C_{\boldsymbol{x}} = \prod_{i=1}^{\ell} C_i^{x_i} \cdot C_0^{-\mathsf{sk}_{\boldsymbol{x}}} \bmod N^2.$$

Then, compute and output $\log_{(1+N)}(C_{\boldsymbol{x}}) = \frac{C_{\boldsymbol{x}} - 1 \bmod N^2}{N}$.

We note that it is possible to adapt the scheme using the technique of Damgård and Jurik [25] so as to further enlarge the message space (and thus the admissible range of computable inner products). For any $s > 1$, ciphertexts can consist of $C_0 = g^r \bmod N^{s+1}$ and $C_i = (1 + y_i N) \cdot h_i^r \bmod N^{s+1}$ for $i \in \{1, \ldots, \ell\}$, where $(y_1, \ldots, y_\ell) \in \mathbb{Z}$ satisfies $X \cdot Y < N^s$.

As in previous constructions (including those of [2]), our security proof requires inner products to be evaluated over $\mathbb{Z}$, although the decryptor technically computes $\langle \boldsymbol{x}, \boldsymbol{y} \rangle \bmod N$. The reason is that, since secret keys are computed over the integers, our security proof only goes through if the adversary is restricted to only obtain secret keys for vectors $\boldsymbol{x}$ such that $\langle \boldsymbol{x}, \boldsymbol{y}_0 \rangle = \langle \boldsymbol{x}, \boldsymbol{y}_1 \rangle$ over $\mathbb{Z}$.

Theorem 5. *The scheme provides full security under the* 1-DCR *assumption.* (The proof is available in Appendix E).

We remark that, if we restrict each coordinate of $\boldsymbol{x}$ and $\boldsymbol{y}$ to be smaller than the bound $K = \sqrt{N/\ell}$ (note that ℓ is much smaller than N since ℓ is polynomial), we have $\langle \boldsymbol{x}, \boldsymbol{y} \rangle < N$, as desired. In this case, the left hand side of (E.4) is at most $L = 2 \cdot N^{7/2}/\ell^{1/2}$, which allows choosing each s_i in the slightly smaller interval $\{-2^\lambda \cdot L, \ldots, 2^\lambda \cdot L\}$. In turn, this yields secret keys $\mathsf{sk}_{\boldsymbol{x}}$ smaller than $|\sum_{i=1}^{\ell} s_i \cdot x_i| < 2^{\lambda+1} \cdot N^4$, which fits within $\lambda + 4 \log N + O(\ell)$ bits.

5.2 A Construction for Inner Products over $\mathbb{Z}_N$

Here, we show that our first scheme can be adapted in order to compute the inner product $\langle \boldsymbol{y}, \boldsymbol{x} \rangle \bmod N$ instead of computing it over $\mathbb{Z}$. To do this, a first difficulty is that, as in our LWE-based system, private keys are computed over the integers and the adversary may query private keys for vectors that are linearly dependent over $\mathbb{Z}_N^\ell$ but independent over $\mathbb{Z}^\ell$. This problem is addressed as previously, by having the authority keep track of all previously revealed private keys.

In order to prevent ciphertexts from revealing more than $\langle \boldsymbol{y}, \boldsymbol{x} \rangle \bmod N$, we scramble the integer inner product $\langle \boldsymbol{y}, \boldsymbol{x} \rangle$ using a noise term consisting of a random multiple of N. To this end, we use the larger message space of the Damgård-Jurik cryptosystem. By computing ciphertexts modulo N^4, the message space becomes $\mathbb{Z}_{N^3}$. A first idea is to encrypt a vector of $\mathbb{Z}_N^\ell$ by adding a random multiple of N to each coordinate. In the security proof, this will help us drown the integer q in the expression $\langle \boldsymbol{y}, \boldsymbol{x} \rangle = (\langle \boldsymbol{y}, \boldsymbol{x} \rangle \bmod N) + q \cdot N$. The decryption operation first computes an inner product over $\mathbb{Z}_{N^3}$ before reducing the result modulo N. In fact, instead of having the sender blind each coordinate of $\boldsymbol{y}$ using a random multiple of N, it is sufficient to force the decryptor to add a random multiple of N to the integer inner product $\langle \boldsymbol{y}, \boldsymbol{x} \rangle$ before reducing it modulo N.

The security proof relies again on the existence of two equally likely master secret keys that are both compatible with the adversary's view and explain the challenge ciphertext as an encryption of $\boldsymbol{y}_0$ and $\boldsymbol{y}_1$, respectively. In order for the argument to work here, we make sure that a system of equations $\langle \boldsymbol{\Delta t}, \boldsymbol{x}_j \rangle = \langle \boldsymbol{y}_\beta - \boldsymbol{y}_{1-\beta}, \boldsymbol{x}_j \rangle / N$ $(1 \leq j \leq \ell - 1)$ has a solution $\boldsymbol{\Delta t} \in \{-2\ell N, \ldots, 2\ell N\}^\ell$ over $\mathbb{Z}$, where $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{\ell-1} \in \mathbb{Z}_N^\ell$ are the secret key queries made by the adversary. Our solution to this problem is to encode each secret key query $\boldsymbol{x}_j \in \mathbb{Z}_N^\ell$ as a (2ℓ)-dimensional vector $\boldsymbol{X}_j$ of which the second half is kept out of the adversary's control. The scheme thus statefully introduces ℓ "dummy" coordinates in the vectors $\boldsymbol{x} \in \mathbb{Z}_N^\ell$ before generating a secret key for them.

Setup$(1^\lambda, 1^\ell)$**:** Choose safe prime numbers $p = 2p' + 1$, $q = 2q' + 1$ with sufficiently large primes $p', q' > 2^{l(\lambda)}$, for some polynomial l, and compute $N = pq$. Then, sample $g' \hookleftarrow \mathbb{Z}_{N^4}^*$ and compute $g = {g'}^{2N^3} \bmod N^4$, which generates the subgroup of $(2N^3)$th residues in $\mathbb{Z}_{N^4}^*$ with overwhelming probability. For each $i \in \{1, \ldots, 2\ell\}$, sample $s_i \hookleftarrow \{-2^{\lambda+2} N^{10}, \ldots, 2^{\lambda+2} N^{10}\}$ and compute $h_i = g^{s_i} \bmod N^4$. Define

$$\mathsf{mpk} := \Big(N, g, \{h_i\}_{i=1}^{2\ell} \Big)$$

and $\mathsf{msk} := \{s_i\}_{i=1}^{2\ell}$. The prime numbers p, p', q, q' are no longer needed.

Keygen$(\mathsf{msk}, \mathsf{st}, \boldsymbol{x})$**:** To generate the jth secret key for $\boldsymbol{x} = (x_1, \ldots, x_\ell) \in \mathbb{Z}_N^\ell$, given the master secret key msk and the internal state st (which is empty if $j = 1$), the authority runs the following algorithm. At the jth secret key generation, state st contains at most $j - 1$ tuples $\{(i, \boldsymbol{x}_i, \boldsymbol{z}_i)\}_{i \leq j-1}$ where each pair $(\boldsymbol{x}_i, \boldsymbol{z}_i)$ consists of a vector $\boldsymbol{x}_i \in \mathbb{Z}_N^\ell$ for which a secret key was previously generated and $\boldsymbol{z}_i \in \mathbb{Z}$ is the generated secret key. If $\boldsymbol{x}$ is linearly independent of $\{\boldsymbol{x}_i\}_{i \leq j-1}$ over $\mathbb{Z}_N$, define the integer vector

$$\begin{aligned}\boldsymbol{X} &= (X_1, \ldots, X_{2\ell}) \\ &= (x_1, \ldots, x_\ell, 0, \ldots, 1, \ldots, 0) = (\boldsymbol{x} \mid \boldsymbol{e}_j) \in \mathbb{Z}^{2\ell},\end{aligned} \tag{5.1}$$

where $\boldsymbol{e}_j = (0, \ldots, 1, \ldots, 0) \in \mathbb{Z}^\ell$ stands for the jth unit vector of dimension ℓ. Then, compute the secret key $\mathsf{sk}_{\boldsymbol{x}} = (\boldsymbol{X}, \langle \boldsymbol{s}, \boldsymbol{X} \rangle)$, where the inner product $\boldsymbol{z}_{\boldsymbol{x}} = \langle \boldsymbol{s}, \boldsymbol{X} \rangle$ is computed over $\mathbb{Z}$, and add $(\boldsymbol{x}, \boldsymbol{z}_{\boldsymbol{x}})$ to st. If there exists coefficients $\{k_i\}_{i \leq j-1}$ in $\mathbb{Z}_N$ such that $\boldsymbol{x} = \sum_i k_i \boldsymbol{x}_i \bmod N$, then set $\boldsymbol{X} = \sum_i k_i \boldsymbol{X}_i \in \mathbb{Z}^{2\ell}$ and $\boldsymbol{z}_{\boldsymbol{x}} = \sum_i k_i \boldsymbol{z}_i \in \mathbb{Z}$. In both cases, return $(\boldsymbol{X}, \boldsymbol{z}_{\boldsymbol{x}})$.

Encrypt$(\mathsf{mpk}, \boldsymbol{y})$**:** To encrypt $\boldsymbol{y} = (y_1, \ldots, y_\ell) \in \mathbb{Z}_N^\ell$, sample $r \hookleftarrow \{0, \ldots, N/4\}$ as well as $t_{\ell+1}, \ldots, t_{2\ell} \hookleftarrow \mathbb{Z}_{N^2}$ and compute

$$\begin{aligned}&C_0 = g^r \bmod N^4, \\ &C_i = (1+N)^{y_i} \cdot h_i^r \bmod N^4, && \forall i \in \{1, \ldots, \ell\}, \\ &C_i = (1+N)^{t_i N} \cdot h_i^r \bmod N^4, && \forall i \in \{\ell+1, \ldots, 2\ell\}.\end{aligned}$$

Return $C_{\boldsymbol{y}} = (C_0, C_1, \ldots, C_{2\ell}) \in \mathbb{Z}_{N^4}^{2\ell+1}$.

Decrypt$(\mathsf{mpk}, \mathsf{sk}_{\boldsymbol{x}}, C_{\boldsymbol{y}})$**:** Given $\mathsf{sk}_{\boldsymbol{x}} = (\boldsymbol{X}, \boldsymbol{z}_{\boldsymbol{x}})$, where $\boldsymbol{z}_{\boldsymbol{x}} \in \mathbb{Z}$, compute

$$C_{\boldsymbol{x}} = \prod_{i=1}^{2\ell} C_i^{X_i} \cdot C_0^{-\boldsymbol{z}_{\boldsymbol{x}}} \bmod N^4$$

where $\boldsymbol{X} = (X_1, \ldots, X_{2\ell}) \in \mathbb{Z}^{2\ell}$ is as defined in Equation (5.1). Then, compute

$$\mu = \log_{(1+N)}(C_{\boldsymbol{x}}) = \frac{C_{\boldsymbol{x}} - 1 \bmod N^4}{N}$$

over $\mathbb{Z}$ and return $\mu \bmod N$.

We note that the decryption algorithm is correct since

$$C_{\boldsymbol{x}} = (1+N)^{\langle \boldsymbol{y}, \boldsymbol{x} \rangle + N \cdot \sum_{i=\ell+1}^{2\ell} s_i \cdot t_i \bmod N^3} \cdot g^{\sum_{i=1}^{2\ell} s_i \cdot X_i} \bmod N^4,$$

so that reducing μ modulo N yields $\langle \boldsymbol{y}, \boldsymbol{x} \rangle \bmod N$ as desired.

From a security standpoint, the following result is proved in Appendix F.

Theorem 6. *The scheme provides full security under the* 3-DCR *assumption.*

6 Bootstrapping Linear FE to Efficient Bounded FE for all circuits

In this section, we describe how to compile our Linear FE scheme, denoted by LinFE which computes linear functions modulo p (for us $p = 2$), into a bounded collusion FE scheme for all circuits, denoted by BddFE. The underlying scheme LinFE is assumed to be AD-IND secure, which, by [46], is equivalent to non-adaptive simulation secure NA-SIM, since linear functions are "preimage sampleable". We refer the reader to [46] for more details.

Let $\mathcal{C}$ be a family of polynomial-size circuits. Let $C \in \mathcal{C}$ and let $\mathbf{x}$ be some input. Let $\widetilde{C}(\mathbf{x}, R)$ be a randomized encoding of C that is computable by a constant depth circuit with respect to inputs x and R (see [7]). Then consider a new family of circuits $\mathcal{G}$ defined by:

$$G_{C,\Delta}(x, R_1, \ldots, R_S) = \left\{ \widetilde{C}\Big(x; \bigoplus_{a \in \Delta} R_a\Big) : \ C \in \mathcal{C}, \ \Delta \subseteq [S] \right\},$$

for some S to be chosen below. As observed in [36, Section 6], circuit $G_{C,\Delta}(\cdot, \cdot)$ is computable by a constant degree polynomial (one for each output bit). Given an FE scheme for $\mathcal{G}$, one may construct a scheme for $\mathcal{C}$ by having the decryptor first recover the output of $G_{C,\Delta}(\mathbf{x}, R_1, \ldots, R_S)$ and then applying the decoder for the randomized encoding to recover $C(\mathbf{x})$.

Note that to support q queries the decryptor must compute q randomized encodings, each of which needs fresh randomness. As shown above, this is handled by hardcoding sufficiently many random elements in the ciphertext and taking a random subset sum of these to generate fresh random bits for each query. As in [36], the parameters are chosen so that the subsets form a cover-free system, so that every random subset yields fresh randomness (with overwhelming probability).

In more details, we let the set S, v, m be parameters to the construction. Let Δ_i for $i \in [q]$ be a uniformly random subset of S of size v. To support q queries, we identify the set $\Delta_i \subseteq S$ with query i. If $v = O(\lambda)$ and $S = O(\lambda \cdot q^2)$ then the sets Δ_i are cover-free with high probability. For details, we refer the reader to [36, Section 5]. We now proceed to describe our construction. Let $L \triangleq (\ell + S \cdot m)^3$, where $m \in \mathsf{poly}(\lambda)$ is the size of the random input in the randomized encoding and ℓ is the length of the messages to be encrypted.

BddFE.Setup$(1^\lambda, 1^\ell)$**:** Upon input the security parameter λ and the message space $\mathcal{M} = \{0,1\}^\ell$, invoke $(\mathsf{mpk}, \mathsf{msk}) = \mathsf{LinFE.Setup}(1^\lambda, 1^L)$ and output it.

BddFE.KeyGen(msk, C)**:** Upon input the master secret key and a circuit C, do:

1. Sample a uniformly random subset $\Delta \subseteq S$ of size v.
2. Express $C(\mathbf{x})$ by $G_{C,\Delta}(\mathbf{x}, R_1, \ldots, R_S)$, which in turn can be expressed as a sequence of degree 3 polynomials $P_1, \ldots, P_k$, where $k \in \mathsf{poly}(\lambda)$.
3. Linearize each polynomial P_i and let P'_i be its vector of coefficients. Note that the ordering of the coefficients can be aribitrary but should be public.
4. Output $\mathsf{BddFE.SK}_C = \{\mathsf{SK}_i = \mathsf{LinFE.KeyGen}(\mathsf{LinFE.msk}, P'_i)\}_{i \in [k]}$.

BddFE.Enc$(\mathbf{x}, \mathsf{mpk})$**:** Upon input the public key and the plaintext $\mathbf{x}$, do:

1. Sample $R_1, \ldots, R_S \leftarrow \{0,1\}^m$.
2. Compute all symbolic monomials of degree 3 in the variables $x_1, \ldots, x_\ell$ and $R_{i,j}$ for $i \in [S]$, $j \in [m]$. The number of such monomials is $L = (\ell + S \cdot m)^3$. Arrange them according to the public ordering and denote the resulting vector by $\boldsymbol{y}$.

3. Output $\mathsf{CT}_{\mathbf{x}} = \mathsf{LinFE.Enc}(\mathsf{LinFE.mpk}, \boldsymbol{y})$.

BddFE.Dec$(\mathsf{mpk}, \mathsf{CT}_{\mathbf{x}}, \mathsf{SK}_C)$**:** Upon input a ciphertext $\mathsf{CT}_{\mathbf{x}}$ for vector $\mathbf{x}$, and a secret key $\mathsf{SK}_C = \{\mathsf{SK}_i\}_{i\in[k]}$ for circuit C, do the following:
1. Compute $G_{C,\Delta}(\mathbf{x}, R_1, \ldots, R_S) = \{P_i(\mathbf{Y})\}_{i\in[k]} = \{\mathsf{LinFE.Dec}(\mathsf{CT}_{\mathbf{x}}, \mathsf{SK}_i)\}_{i\in[k]}$.
2. Run the decoder for the randomized encoding to recover $C(\mathbf{x})$ from $G_{C,\Delta}(\mathbf{x}, R_1, \ldots, R_S)$.

Correctness follows from the correctness of LinFE and the correctness of randomized encodings.

Security. The definition for q-NA-SIM security is provided in Appendix A. We proceed to describe our simulator Bdd.Sim. Let RE.Sim be the simulator guaranteed by the security of randomized encodings and LinFE.Sim be the simulator guaranteed by the security of the LinFE scheme.

Simulator Bdd.Sim$\big(\{C_i, C_i(\mathbf{x}), \mathsf{SK}_i\}_{i\in[q^*]}\big)$**:** The simulator Bdd.Sim receives the secret key queries C_i, the corresponding (honestly generated) secret keys SK_i and the values $C_i(\mathbf{x})$ for $i \in [q^*]$ where $q^* \le q$, and must simulate the ciphertext $\mathsf{CT}_{\mathbf{x}}$. It proceeds as follows:

1. Sample $\Delta_1, \ldots, \Delta_q \subseteq S$, of size v each.
2. For each $i \in [q^*]$, invoke $\mathsf{RE.Sim}(C_i(x))$ to learn $G_{C_i}(\mathbf{x}, \hat{R}_i)$ for some $\hat{R}_i$ chosen by the simulator. Interpret
$$\hat{R}_i = \bigoplus_{a\in\Delta_i} R_a \quad \text{and } G_{C_i,\Delta_i}(\mathbf{x}, R_1, \ldots, R_S) = G_{C_i}(\mathbf{x}, \hat{R}_i) = \big(P_1(\mathbf{Y}), \ldots, P_k(\mathbf{Y})\big).$$
3. Let $\mathsf{CT}_{\mathbf{x}} = \mathsf{LinFE.Sim}\big(\{G_{C_i,\Delta_i}, G_{C_i,\Delta_i}(\mathbf{x}, R_1, \ldots, R_S), \mathsf{SK}_i\}_{i\in[q^*]}\big)$ and output it.

The correctness of Bdd.Sim follows from the correctness of RE.Sim and LinFE.Sim.

A last remaining technicality is that the most general version of our construction for FE for inner product modulo p is stateful. This is because a general adversary against LinFE may request keys that are linearly dependent modulo p but linearly independent over the integers, thus learning new linear relations in the master secret. This forces the simulator (and hence the key generator) to maintain a state.

However, in our application, we can make do with a stateless variant, since all the queries will be linearly independent over $\mathbb{Z}_2$. To see this, note that in the above application of LinFE, each query is randomized by a unique random set Δ_i. Recall that by cover-freeness, the element $\bigoplus_{a\in\Delta_i} R_a$ must contain at least one fresh random element, say R^*, which is not contained by $\bigcup_{j\neq i} \Delta_j$. Stated a bit differently, if we consider the query vectors of size L, then cover-freeness implies that no query vector lies within the linear span of the remaining queries made by the adversary. For any query Q, there is at least one position $j \in [L]$ so that this position is nonzero in the L vector representing Q but zero for all other vectors. Hence the query vectors are linearly independent over $\mathbb{Z}_2$, for which case, our construction of Section 4.2 is stateless.

Acknowledgements.

This work has been supported in part by ERC Starting Grant ERC-2013-StG-335086-LATTAC. Part of this work was also funded by the "Programme Avenir Lyon Saint-Etienne de l'Université de Lyon" in the framework of the programme "Inverstissements d'Avenir" (ANR-11-IDEX-0007).

References

1. M. Abdalla, M. Bellare, D. Catalano, E. Kiltz, T. Kohno, T. Lange, J. Malone-Lee, G. Neven, P. Paillier, and H. Shi. Searchable encryption revisited: Consistency properties, relation to anonymous ibe, and extensions. In *Proc. of Crypto*, volume 3621 of *LNCS*, pages 205–222. Springer, 2005.
2. M. Abdalla, F. Bourse, A. De Caro, and D. Pointcheval. Simple functional encryption schemes for inner products. In *Proceedings of PKC*, volume 9020 of *LNCS*, pages 733–751. Springer, 2015.
3. S. Agrawal, D. M. Freeman, and V. Vaikuntanathan. Functional encryption for inner product predicates from learning with errors. In *Proc. of Asiacrypt*, volume 7073 of *LNCS*, pages 21–40. Springer, 2011.
4. S. Agrawal, S. Gorbunov, V. Vaikuntanathan, and H. Wee. Functional encryption: New perspectives and lower bounds. In *Crypto*, 2013.
5. J. Alperin-Sheriff and C. Peikert. Circular and kdm security for identity-based encryption. In *Proceedings of PKC*, volume 7293 of *LNCS*, pages 334–352. Springer, 2012.
6. P. Ananth, Z. Brakerski, G. Segev, and V. Vaikuntanathan. The trojan method in functional encryption: From selective to adaptive security, generically. In *Proc. of Crypto*, LNCS. Springer, 2015.
7. B. Applebaum, Y. Ishai, and E. Kushilevitz. Computationally private randomizing polynomials and their applications. *Computational Complexity*, 15(2):115–162, 2006.
8. M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proc. of STOC*, pages 1–10. ACM, 1988.
9. D. Bernstein and T. Lange. Computing small discrete logarithms faster. In *Proc. of Indocrypt'12*, volume 7668 of *LNCS*, pages 317–338. Springer, 2012.
10. D. Boneh and X. Boyen. Efficient selective-id secure identity-based encryption without random oracles. In *Proc. of Eurocrypt*, volume 3027 of *LNCS*, pages 223–238. Springer, 2004.
11. D. Boneh and X. Boyen. Efficient selective identity-based encryption without random oracles. *Journal of Cryptology*, 24(4):659–693, 2011.
12. D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In *Proc. of Crypto*, volume 3152 of *LNCS*, pages 41–55. Springer, 2004.
13. D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano. Public key encryption with keyword search. In *Proc. of Eurocrypt*, volume 3027 of *LNCS*, pages 506–522. Springer, 2004.
14. D. Boneh and M. Franklin. Identity-based encryption from the Weil pairing. *SIAM J. Comput*, 32(3):586–615 (electronic), 2003.

15. D. Boneh, C. Gentry, S. Gorbunov, S. Halevi, V. Nikolaenko, G. Segev, V. Vaikuntanathan, and D. Vinayagamurthy. Fully key-homomorphic encryption, arithmetic abe and compact garbled circuits. In *Proc. of Eurocrypt*, volume 8441 of *LNCS*, pages 533–556. Springer, 2014.
16. D. Boneh, A. Sahai, and B. Waters. Functional encryption: Definitions and challenges. In *Proc. of TCC*, volume 6597 of *LNCS*, pages 253–273. Springer, 2011.
17. Z. Brakerski, A. Langlois, C. Peikert, Regev. O., and D. Stehlé. On the classical hardness of learning with errors. In *Proc. of STOC*, pages 575–584. ACM, 2013.
18. E. Bresson, D. Catalano, and D. Pointcheval. A simple public-key cryptosystem with a double trapdoor decryption mechanism and its applications. In *Proc. of Asiacrypt 2003*, volume 2894 of *LNCS*, pages 37–54. Springer, 2003.
19. J. Camenisch and V. Shoup. Practical verifiable encryption and decryption of discrete logarithms. In *CRYPTO*, pages 126–144, 2003.
20. J.-H. Cheon, K. Han, C. Lee, H. Ryu, and D. Stehlé. Cryptanalysis of the multilinear map over the integers. In *Proc. of EUROCRYPT*, volume 9056 of *LNCS*, pages 3–12. Springer, 2015.
21. J.-S. Coron, T. Lepoint, and M. Tibouchi. Practical multilinear maps over the integers. In *Proc. of CRYPTO*, volume 8042 of *LNCS*, pages 476–493, 2013.
22. R. Cramer and V. Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In *Proc. of CRYPTO*, volume 1462 of *LNCS*, pages 13–25. Springer, 1998.
23. R. Cramer and V. Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In *Proc. of Eurocrypt*, volume 2332 of *LNCS*, pages 45–64. Springer, 2002.
24. I. Damgård. Towards practical public key systems secure against chosen ciphertext attacks. In *Proc. of CRYPTO*, volume 576 of *LNCS*, pages 445–456. Springer, 1991.
25. I. Damgård and M. Jurik. A generalisation, a simplification and some applications of Paillier's probabilistic public-key system. In *Public Key Cryptography*, volume 1992 of *LNCS*, pages 119–136. Springer, 2001.
26. A. De Caro, V. Iovino, A. Jain, A. O'Neill, O. Paneth, and G. Persiano. On the achievability of simulation-based security for functional encryption. In *Crypto'13*, volume 8043 of *LNCS*, pages 519–535. Springer, 2013.
27. L. Ducas, A. Durmus, T. Lepoint, and V. Lyubashevsky. Lattice signatures and bimodal gaussians. In *Proc. of CRYPTO*, volume 8042 of *LNCS*, pages 40–56. Springer, 2013.
28. A. Escala, G. Herold, E. Kiltz, C. Ràfols, and G. Villard. An algebraic framework for Diffie-Hellman assumptions. In *Proc. of Crypto*, volume 8043 of *LNCS*, pages 129–147. Springer, 2013.
29. S. Galbraith and R. Ruprai. Using equivalence classes to accelerate solving the discrete logarithm problem in a short interval. In *Proc. of PKC*, volume 6056 of *LNCS*, pages 368–383. Springer, 2010.
30. S. Garg, C. Gentry, and S. Halevi. Candidate multilinear maps from ideal lattices and applications. In *Proc. of Eurocrypt*, LNCS. Springer, 2013.
31. S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *Proc. of FOCS*, pages 40–49, 2013.
32. S. Garg, C. Gentry, S. Halevi, A. Sahai, and B. Waters. Attribute-based encryption for circuits from multilinear maps. In *Proc. of Crypto*, volume 8043 of *LNCS*, pages 479–499. Springer, 2013.

33. C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *Proc. of STOC*, pages 197–206. ACM, 2008.
34. S. Goldwasser, Y. Tauman Kalai, R. Popa, V. Vaikuntanathan, and N. Zeldovich. How to run turing machines on encrypted data. In *CRYPTO (2)*, pages 536–553, 2013.
35. S. Goldwasser, Y. Tauman Kalai, R. Popa, V. Vaikuntanathan, and N. Zeldovich. Reusable garbled circuits and succinct functional encryption. In *Proc. of STOC*, pages 555–564. ACM Press, 2013.
36. S. Gorbunov, V. Vaikuntanathan, and H. Wee. Functional encryption with bounded collusions via multi-party computation. In *Proc. of Crypto*, volume 7417 of *LNCS*, pages 162–179. Springer, 2012.
37. S. Gorbunov, V. Vaikuntanathan, and H. Wee. Attribute-based encryption for circuits. In *Proc. of STOC*, pages 545–554. ACM Press, 2013.
38. S. Gorbunov, V. Vaikuntanathan, and H. Wee. Predicate encryption for circuits from LWE. In *Proc. of CRYPTO*, LNCS. Springer, 2015.
39. V. Goyal, O. Pandey, A. Sahai, and B. Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *Proc. of ACM-CCS'06*, pages 89–98. ACM Press, 2006.
40. Y. Hu and H. Jia. Cryptanalysis of GGH map. Cryptology ePrint Archive: Report 2015/301, 2015.
41. J. Katz, A. Sahai, and B. Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In *Proc. of EUROCRYPT*, volume 4965 of *LNCS*, pages 146–162. Springer, 2008.
42. S. Ling, D. H. Phan, D. Stehlé, and R. Steinfeld. Hardness of k-LWE and Applications in Traitor Tracing. In *Proc. of CRYPTO*, volume 8616 of *LNCS*, pages 315–334. Springer, 2014.
43. D. Micciancio and P. Mol. Pseudorandom knapsacks and the sample complexity of LWE search-to-decision reductions. In *Proc. of CRYPTO*, volume 6841 of *LNCS*, pages 465–484. Springer, 2011.
44. D. Micciancio and C. Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In *Proc. of EUROCRYPT*, volume 7237 of *LNCS*, pages 700–718. Springer, 2012.
45. P. Q. Nguyen and J. Stern. Merkle-Hellman revisited: A cryptanalysis of the Qu-Vanstone cryptosystem based on group factorizations. In *Proc. of CRYPTO*, volume 1294 of *LNCS*, pages 198–212. Springer, 1997.
46. A. O'Neill. Definitional issues in functional encryption. Cryptology ePrint Archive, Report 2010/556, 2010. `http://eprint.iacr.org/`.
47. A. O'Neill, C. Peikert, and B. Waters. Bi-deniable public-key encryption. In *Proc. of CRYPTO*, volume 6841 of *LNCS*, pages 525–542. Springer, 2011.
48. P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Proc. of EUROCRYPT*, volume 1592 of *LNCS*, pages 223–238. Springer, 1999.
49. J. Pollard. Kangaroos, monopoly and discrete logarithms. *Journal of Cryptology*, 13:433–447, 2000.
50. O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proc. of STOC*, pages 84–93. ACM, 2005.
51. O. Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6), 2009.
52. A. Sahai and H. Seyalioglu. Worry-free encryption: Functional encryption with public keys. In *Proceedings of the 17th ACM Conference on Computer and Communications Security*, CCS '10, 2010.
53. A. Sahai and B. Waters. Fuzzy identity-based encryption. In *Proc. of EUROCRYPT*, volume 3494 of *LNCS*, pages 457–473. Springer, 2005.

54. H. Shacham. A Cramer-Shoup encryption scheme from the linear assumption and from progressively weaker linear variants. Cryptology ePrint Archive: Report 2007/074, 2007.
55. A. Shamir. Identity-based cryptosystems and signature schemes. In *Proc. of CRYPTO*, volume 196 of *LNCS*, pages 47–53. Springer, 1984.
56. B. Waters. Functional encryption for regular languages. In *Crypto*, 2012.

A Definitions for functional encryption

We now recall the syntax of Functional Encryption, as defined by Boneh, Sahai and Waters [16], and their indistinguishability-based security definition.

Definition 6 ([16]). *A functionality F defined over $(\mathcal{K}, \mathcal{Y})$ is a function $F : \mathcal{K} \times \mathcal{Y} \to \Sigma \cup \{\perp\}$, where $\mathcal{K}$ is a key space, $\mathcal{Y}$ is a message space and Σ is an output space, which does not contain the special symbol $\perp$.*

Definition 7. *A functional encryption (FE) scheme for a functionality F is a tuple $\mathcal{FE} = (\mathsf{Setup}, \mathsf{Keygen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ of algorithms with the following specifications:*

$\mathsf{Setup}(1^\lambda)$: *Takes as input a security parameter 1^λ and outputs a master key pair $(\mathsf{mpk}, \mathsf{msk})$.*
$\mathsf{Keygen}(\mathsf{msk}, K)$: *Given the master secret key msk and a key (i.e., a function) $K \in \mathcal{K}$, this algorithm outputs a key sk_K.*
$\mathsf{Encrypt}(\mathsf{mpk}, Y)$: *On input of a message $Y \in \mathcal{Y}$ and the master public key mpk, this randomized algorithm outputs a ciphertext C.*
$\mathsf{Decrypt}(\mathsf{mpk}, \mathsf{sk}_K, C)$: *Given the master public key mpk, a ciphertext C and a key sk_K, this algorithm outputs $v \in \Sigma \cup \{\perp\}$.*

We require that, for all $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda)$, all keys $K \in \mathcal{K}$ and all messages $Y \in \mathcal{Y}$, if $\mathsf{sk}_K \leftarrow \mathsf{Keygen}(\mathsf{msk}, K)$ and $C \leftarrow \mathsf{Encrypt}(\mathsf{mpk}, Y)$, with overwhelming probability, we have $\mathsf{Decrypt}(\mathsf{mpk}, \mathsf{sk}_K, C) = F(K, Y)$ whenever $F(K, Y) \neq \perp$.

In some cases, we will also give a state st as input to algorithm Keygen, so that a stateful authority may reply to key queries in a way that depends on the queries that have been made so far. In that situation, algorithm Keygen may additionally update state st.

Indistinguishability-based security. From a security standpoint, what we expect from a FE scheme is that, given $C \leftarrow \mathsf{Encrypt}(\mathsf{mpk}, Y)$, the only thing revealed by a secret key sk_K about the underlying Y is the function evaluation $F(K, Y)$. In the natural definition of indistinguishability-based security (see, e.g., [16]), one asks that no efficient adversary be able to differentiate encryptions of Y_0 and Y_1 without obtaining secret keys sk_K such that $F(K, Y_0) \neq F(K, Y_1)$.

Definition 8 (Indistinguishability-based security). *A functional encryption scheme $\mathcal{FE} = (\mathsf{Setup}, \mathsf{Keygen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ provides semantic security under chosen-plaintext attacks (or IND-CPA security) if no PPT adversary has non-negligible advantage in the following game, where $q_1 \leq q \in \mathsf{poly}(\lambda)$:*

1. The challenger runs $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda)$ *and the master public key* mpk *is given to the adversary* $\mathcal{A}$.
2. The adversary adaptively makes secret key queries to the challenger. At each query, adversary $\mathcal{A}$ *chooses a key* $K \in \mathcal{K}$ *and obtains* $\mathsf{sk}_K \leftarrow \mathsf{Keygen}(\mathsf{msk}, K)$.
3. Adversary $\mathcal{A}$ *chooses distinct messages* Y_0, Y_1 *subject to the restriction that, if* $\{K_i\}_{i=1}^{q_1}$ *denotes the set of secret key queries made by* $\mathcal{A}$ *at Stage* 2*, it holds that* $F(K_i, Y_0) = F(K_i, Y_1)$ *for each* $i \in \{1, \ldots, q_1\}$*. Then, the challenger flips a fair coin* $\beta \hookleftarrow \{0, 1\}$ *and computes* $C^\star \leftarrow \mathsf{Encrypt}(\mathsf{mpk}, Y_\beta)$ *which is sent as a challenge to* $\mathcal{A}$.
4. Adversary $\mathcal{A}$ *makes further secret key queries for arbitrary keys* $K \in \mathcal{K}$*. However, it is required that* $F(K, Y_0) = F(K, Y_1)$ *at each query* $K \in \{K_{q_1+1}, \ldots, K_q\}$.
5. Adversary $\mathcal{A}$ *eventually outputs a bit* $\beta' \hookleftarrow \{0, 1\}$ *and wins if* $\beta' = \beta$.

The adversary's advantage is defined to be $\mathbf{Adv}_{\mathcal{A}}(\lambda) := |\Pr[\beta' = \beta] - 1/2|$*, where the probability is taken over all coin tosses.*

Definition 8 captures *adaptive* security in that the adversary is allowed to choose the messages Y_0, Y_1 at Stage 3. In [2], Abdalla *et al.* considered a weaker security notion, called *selective* security, where the adversary has to declare the messages Y_0, Y_1 at the very beginning of the game, before even seeing mpk (note that, in this scenario, the adversary can receive the challenge ciphertext at the same time as the public key). In this work, our goal will be to meet the strictly stronger requirements of adaptive security.

Boneh, Sahai and Waters [16] pinpointed shortcomings of indistinguishability-based definitions in the case of general functionalities, where they may fail to rule out intuitively insecure systems. Boneh *et al.* [16] proposed strong simulation-based definitions, but these have been shown to be impossible to realize in the standard model [16, 4].

On the positive side, O'Neill [46] showed that indistinguishability-based security is equivalent to non-adaptive simulation based security (defined below) for a class of functions called preimage sampleable functions, which includes inner products. De Caro *et al.* [26] gave a general method of constructing FE schemes that achieve a meaningful definition of simulation-based security from systems that are only proved secure in the sense of indistinguishability-based definitions. Also, note that the impossibility of achieving adaptive simulation-based security for IBE, exhibited by [16] can be easily adapted to show that adaptive simulation-based security (AD-SIM) is also impossible to achieve for the inner product functionality. Thus, adaptive indistinguishability appears to be the strongest adaptive notion of security that may still be achievable, and for a wide range of practically interesting specific functionalities this notion is believed to suffice. In the following, we will aim at *full* security in the sense of Definition 8.

SIMULATION-BASED SECURITY FOR BOUNDED COLLUSIONS In this section, we define simulation based security for bounded collusions, as in [36, Defn 3.1].

Definition 9 **(q-NA-SIM- and q-AD-SIM- Security).**

Let $\mathcal{F}$ be a functional encryption scheme for a circuit family $\mathcal{C}$. For every p.p.t. adversary $A = (A_1, A_2)$ and a p.p.t. simulator Sim*, consider the following two experiments:*

$\underline{\mathsf{Exp}^{\mathsf{real}}_{\mathcal{F},A}(1^\lambda)}$**:**	$\underline{\mathsf{Exp}^{\mathsf{ideal}}_{\mathcal{F},\mathrm{Sim}}(1^\lambda)}$**:**
1: $(\mathsf{MPK}, \mathsf{MSK}) \leftarrow \mathsf{FE.Setup}(1^\lambda)$	*1:* $\mathsf{MPK} \leftarrow \mathsf{FE.Setup}(1^\lambda)$
2: $(x, st) \leftarrow A_1^{\mathsf{FE.Keygen}(\mathsf{MSK},\cdot)}(\mathsf{MPK})$	*2:* $(x, st) \leftarrow A_1^{\mathsf{FE.Keygen}(\mathsf{MSK},\cdot)}(\mathsf{MPK})$ *Let* $\mathcal{V} \triangleq (C_i, C_i(x), \mathsf{SK}_i)_{i\in[q]}$
3: $\mathsf{CT} \leftarrow \mathsf{FE.Enc}(\mathsf{MPK}, x)$	*3:* $\mathsf{CT}, st' \leftarrow \mathrm{Sim}(\mathsf{MPK}, \mathcal{V}, 1^{\lvert x\rvert})$
4: $\alpha \leftarrow A_2^{\mathcal{O}(\mathsf{MSK},\cdot)}(\mathsf{MPK}, \mathsf{CT}, st)$	*4:* $\alpha \leftarrow A_2^{\mathcal{O}'(\mathsf{MSK},st',\cdot)}(\mathsf{MPK}, \mathsf{CT}, st)$
5: Output (x, α)	*5: Output* (x, α)

Above, C_i denote the queries made by the adversary. We distinguish between two cases of the above experiment:

1. The adaptive experiment*, where:*
   - *the oracle* $\mathcal{O}(\mathsf{MSK}, \cdot) = \mathsf{FE.Keygen}(\mathsf{MSK}, \cdot)$ *and*
   - *the oracle* $\mathcal{O}'(\mathsf{MSK}, st', \cdot)$ *is the simulator, namely* $\mathrm{Sim}^{U_x(\mathsf{MSK},st',\cdot)}(\cdot)$ *and* $U_x(C) = C(x)$ *for any* $C \in \mathcal{C}$*.*

   The simulator algorithm is stateful in that after each invocation, it updates the state st' which is carried over to its next invocation. We call a stateful simulator algorithm Sim admissible *if, on each input C,* Sim *makes just a single query to its oracle $U_x(\cdot)$ on C itself.*
   The functional encryption scheme $\mathcal{F}$ is then said to be q query simulation-secure for one message against adaptive adversaries (q-AD-SIM-secure, for short) *if there is an* admissible *stateful p.p.t. simulator* Sim *such that for every p.p.t. adversary $A = (A_1, A_2)$ that makes at most q queries, the following two distributions are computationally indistinguishable:*

$$\left\{\mathsf{Exp}^{\mathsf{real}}_{\mathcal{F},A}(1^\lambda)\right\}_{\lambda\in\mathbb{N}} \overset{c}{\approx} \left\{\mathsf{Exp}^{\mathsf{ideal}}_{\mathcal{F},\mathrm{Sim}}(1^\lambda)\right\}_{\lambda\in\mathbb{N}}$$

2. The non-adaptive experiment*, where the oracles* $\mathcal{O}(\mathsf{MSK}, \cdot)$ *and* $\mathcal{O}'(\mathsf{MSK}, st, \cdot)$ *are both the "empty oracles" that return nothing.*
   The functional encryption scheme $\mathcal{F}$ is then said to be q query simulation-secure for one message against non-adaptive adversaries (q-NA-SIM-secure, for short) *if there is an* admissible *stateful p.p.t. simulator* Sim *such that for every p.p.t. adversary $A = (A_1, A_2)$ that makes at most q queries, the two distributions above are computationally indistinguishable.*

B Practical applications of Linear FE

On the practical front, Linear FE is already quite useful even when used directly. As pointed out by Abdalla *et al.* [2], the inner product functionality suffices for the computation of linear functions (e.g., sums or averages) over encrypted data. As mentioned in the earlier work of Katz, Sahai and Waters [41], inner products also enable the evaluation of polynomials over encrypted data. To do this, we can simply encode a message M as a vector $\boldsymbol{y} = (1, M, \ldots, M^d) \in \mathcal{D}^{d+1}$ and a degree-d polynomial $P[X] = \sum_{i=0}^{d} p_i X^i$ is encoded as a vector $\boldsymbol{x} = (p_0, p_1, \ldots, p_d) \in \mathcal{D}^{d+1}$ for which the key $SK_{\boldsymbol{x}}$ is generated. Using a similar encoding, we can also evaluate multivariate polynomials of the form $P[X_1, \ldots, X_d] = \prod_{i=1}^{d}(X_i - I_i)$ of small degree $d = O(\log \ell)$. By encoding ℓ-bit messages $M = M[1] \ldots M[\ell]$ as vectors $\boldsymbol{y} = (M[1], \ldots, M[\ell])$, the inner product functionality also allows for the computation of Hamming weights using secret keys $\mathsf{sk}_{\boldsymbol{x}}$ for the all-one vector $\boldsymbol{x} = (1, \ldots, 1)$. More generally, inner products make it possible to compute the Hamming distance between an encrypted n-bit vector $\boldsymbol{y} = (y_1, \ldots, y_n) \in \{0,1\}^n$ and another binary vector $\boldsymbol{x} = (x_1, \ldots, x_n) \in \{0,1\}^n$ embedded in the key, which can be useful in biometric applications. To this end, we can simply encode $\boldsymbol{y}$ and $\boldsymbol{x}$ as vectors $\boldsymbol{Y} = (Y_1, \ldots, Y_{2n}) \in \{-1,1\}^{2n}$ and $\boldsymbol{X} = (X_1, \ldots, X_{2n}) \in \{-1,1\}^{2n}$ such that $X_{2i} = Y_{2i} = 1$ and $Y_{2i+1} = (-1)^{y_i}$, $X_{2i+1} = -(-1)^{x_i}$ for each $i \in \{1, \ldots, n\}$. By doing so, the integer $\langle \boldsymbol{X}, \boldsymbol{Y} \rangle = \sum_{i=1}^{n}(1 - (-1)^{x_i + y_i})$ is exactly twice the Hamming distance between $\boldsymbol{x}$ and $\boldsymbol{y}$.

C Background on lattices

Let Λ be a non-zero lattice. We recall that the *smoothing parameter* of Λ is defined as $\eta_\varepsilon(\Lambda) = \min(s > 0 : \sum_{\hat{\boldsymbol{b}} \in \hat{\Lambda}} \mathsf{Exp}(-\pi \|\hat{\boldsymbol{b}}\|^2/s^2) \leq 1 + \varepsilon)$, where $\hat{\Lambda} = \{\hat{\boldsymbol{b}} \in \mathrm{span}_{\mathbb{R}}(\Lambda) : \hat{\boldsymbol{b}}^T \cdot \Lambda \subseteq \mathbb{Z}\}$ refers to the dual of Λ.

For a matrix $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ for some integers m, n, q, we define the lattice $\Lambda^\perp(\mathbf{A}) = \{\boldsymbol{x} \in \mathbb{Z}^m : \boldsymbol{x}^T \cdot \mathbf{A} = \mathbf{0} \bmod q\}$.

Lemma 6 (Adapted from [44, Le. 2.3 & 2.4]). *Let n, m, q be positive integers, and $\varepsilon, \delta > 0$. Assume that $q = p^k$ for p prime and $k \geq 1$. Assume further that*

$$m \geq \max\left(n + \frac{\log(3 + 4/(\delta\varepsilon))}{\log p}, \frac{n \log q + \log(2 + 2/(\delta\varepsilon))}{\log 2}\right).$$

Then $\eta_\varepsilon(\Lambda^\perp(\mathbf{A})) \leq 2\sqrt{\ln(2m(1 + 2/(\delta\varepsilon)))/\pi}$, except with probability $\leq \delta$ over the uniform choice of $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$.

Let $\Lambda \subseteq \mathbb{R}^n$ be a lattice, $\sigma > 0$ and $\boldsymbol{c} \in \mathbb{R}^n$. We define the *lattice Gaussian distribution* of support Λ, standard deviation parameter σ and center $\boldsymbol{c}$ as:

$$\forall \boldsymbol{b} \in \Lambda : \ D_{\Lambda,\sigma,\boldsymbol{c}}(\boldsymbol{b}) = \frac{\mathsf{Exp}(-\pi \|\boldsymbol{b} - \boldsymbol{c}\|^2/\sigma^2)}{\sum_{\boldsymbol{x} \in \Lambda} \mathsf{Exp}(-\pi \|\boldsymbol{x} - \boldsymbol{c}\|^2/\sigma^2)}.$$

We omit the subscript $\boldsymbol{c}$ when $\boldsymbol{c} = \mathbf{0}$. To implement the primitives described in this work, we only need to be able sample from 1-dimensional lattice Gaussians. Such an efficient sampler is described in [27].

We make use of the following lemmas.

Lemma 7 (Adapted from [33, Cor. 2.8]). *Let $\Lambda' \subseteq \Lambda \subseteq \mathbb{R}^n$ be two lattices wuth the same dimension. Let $\varepsilon \in (0, 1/2)$. Then for any $\boldsymbol{c} \in \mathbb{R}^n$ and any $\sigma \geq \eta_\varepsilon(\Lambda')$, the distribution $D_{\Lambda,\sigma,\boldsymbol{c}} \bmod \Lambda'$ is within statistical distance 2ε from the uniform distribution over Λ/Λ'.*

Lemma 8 (Adapted from [33, Le. 5.2]). *Assume the rows of $\mathbf{A} \in \mathbb{Z}_q^{m\times n}$ generate $\mathbb{Z}_q^n$ and let $\varepsilon \in (0, 1/2)$, $\boldsymbol{c} \in \mathbb{Z}^m$ and $\sigma \geq \eta_\varepsilon(\Lambda^\perp(\mathbf{A}))$. Then for $\boldsymbol{e}$ sampled from $D_{\mathbb{Z}^m,\sigma,\boldsymbol{c}}$, the distribution of the syndrome $\boldsymbol{e}^T \cdot \mathbf{A} \bmod q$ is within statistical distance 2ε of uniform over $\mathbb{Z}_q^n$.*

Note that if $q = p^k$ with p prime, then the rows of $\mathbf{A} \in \mathbb{Z}_q^{m\times n}$ generate $\mathbb{Z}_q^n$ if and only if they generate $\mathbb{Z}_p^n$ (once reduced modulo p). If $\mathbf{A}$ is sampled uniformly, this occurs with probability $\geq 1 - p^{-n}$ when $m \geq 2n \log_2 p$.

Using the Lemmas 6 and 8, we obtain the following result, that we use in the proof of Theorem 2.

Lemma 9. *Let $n, m, q \geq 2$ be positive integers. Assume that $q = p^k$ for p prime and $k \geq 1$. Assume further that $m \geq 2n \log_2 q$. Let $\sigma \geq \Omega(\sqrt{n + \log m})$ and $\boldsymbol{c} \in \mathbb{Z}^m$. Then for $\mathbf{A} \in \mathbb{Z}_q^{m\times n}$ sampled uniformly and $\boldsymbol{e} \in \mathbb{Z}^m$ sampled from $D_{\mathbb{Z}^m,\sigma,\boldsymbol{c}}$, the distribution of the pair $(\mathbf{A}, \boldsymbol{e}^T \cdot \mathbf{A})$ is within statistical distance $2^{-\Omega(n)}$ of uniform over $\mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^n$.*

D Missing material from Section 4

Proof of Lemma 3. The reduction from LWE to faeLWE starts by sampling $\mathbf{A}' \hookleftarrow \mathbb{Z}_q^{t\times n}$. It aborts if it is not full-rank (modulo q): this happens with probability

$$\leq \prod_{p \text{ prime}, p|q} \left(1 - \prod_{0\leq i<t} (1 - p^{-n+i})\right) \leq \prod_{p \text{ prime}, p|q} (4p^{-n+t-1}) \ \leq \ 2^{-n+t+1}.$$

Else, the reduction computes $\mathbf{R} \in \mathbb{Z}_q^{n\times n}$ which is invertible and whose top $t \times n$ submatrix is $\mathbf{A}'$. The reduction also samples $\boldsymbol{s}' \hookleftarrow \mathbb{Z}_q^t$. The first t output samples are $(\boldsymbol{a}'_i, s'_i)$ (for $i \leq t$), where $\boldsymbol{a}'_i$ denote the ith row of $\mathbf{A}'$. The remaining samples are produced by taking a sample $(\boldsymbol{a}, b) \in \mathbb{Z}_q^{n-t} \times \mathbb{Z}_q$ from the given oracle, picking a fresh uniformly random $\boldsymbol{d} \in \mathbb{Z}_q^t$, and returning $(\mathbf{R}^T \cdot (\boldsymbol{d}|\boldsymbol{a}), b + \langle \boldsymbol{s}', \boldsymbol{d}\rangle)$.

Given uniform samples, the reduction outputs uniform samples up to statistical distance 2^{-n+t+1}. Given samples from $A_{q,\alpha,\boldsymbol{s}}$, the reduction outputs t samples from $A_{q,\{0\},\boldsymbol{s}''}$ and the remaining samples from $A_{q,\alpha,\boldsymbol{s}''}$ up to statistical distance 2^{-n+t+1}, with $\boldsymbol{s}'' = \mathbf{R}^{-1} \cdot (\boldsymbol{s}'|\boldsymbol{s})^T \bmod q$. This proves correctness since $\mathbf{R}$ induces a bijection on $\mathbb{Z}_q^n$. $\square$

Leftover hash lemma. We will use the following variant of the leftover hash lemma.

Lemma 10 (Particular case of [43, Le. 2.3]). *Let $q = p^k$ for p prime and $k \geq 1$. Let $m \geq n \geq 1$. Take $\mathcal{X}$ a distribution over $\mathbb{Z}^m$. Let D_0 be the uniform distribution over $\mathbb{Z}_q^{n\times m} \times \mathbb{Z}_q^n$ and D_1 be the distribution of $(\mathbf{A}, \mathbf{A}\cdot \boldsymbol{x}) \in \mathbb{Z}_q^{n\times m} \times \mathbb{Z}_q^n$, where by sampling $\mathbf{A} \hookleftarrow \mathbb{Z}_q^{n\times m}$ and $\boldsymbol{x} \hookleftarrow \mathcal{X}$. Then*

$$\Delta(D_0, D_1) \leq \frac{1}{2}\sqrt{\sum_{i=1}^{k} p^{i\cdot n} \cdot \mathrm{Pr}_i},$$

where Pr_i is the collision probability of two independent samples from $\mathcal{X} \bmod p^i$.

In particular, if the distribution ($\mathcal{X} \bmod p$) is within statistical distance ε from the uniform distribution over $\mathbb{Z}_p^m$, then

$$\Delta(D_0, D_1) \leq \sqrt{(\varepsilon + p^{-m})q^n}.$$

E Proof of Theorem 5

Proof. The proof uses a sequence of games that begins with the real game and ends with a game where the adversary's advantage is exponentially small. For each i, we denote by S_i the event that the adversary wins in Game i.

Game 0: This is the actual security game. Concretely, the adversary $\mathcal{A}$ is given the master public key $\mathsf{mpk} = \big(N, g, \{h_i\}_{i=1}^{\ell}, Y\big)$, where $h_i = g^{s_i} \bmod N^2$ for all $i \in \{1, \ldots, \ell\}$, each s_i being sampled uniformly in $\{-2^{\lambda+1}N^4, \ldots, 2^{\lambda+1}N^4\}$. In the challenge phase, adversary $\mathcal{A}$ chooses two distinct vectors $\boldsymbol{y}_0, \boldsymbol{y}_1 \in \mathbb{Z}^\ell$ of norms $\leq Y$ and obtains an encryption of $\boldsymbol{y}_\beta = (y_{\beta,1}, \ldots, y_{\beta,\ell})$ for a random bit $\beta \hookleftarrow \{0,1\}$ sampled by the challenger $\mathcal{B}$. At the end of the game, $\mathcal{A}$ outputs $\beta' \in \{0,1\}$ and we denote by S_0 the event that $\beta' = \beta$. For any vector $\boldsymbol{x} \in \mathbb{Z}^\ell$ submitted to the secret key extraction oracle, it must be the case that $\langle \boldsymbol{x}, \boldsymbol{y}_0\rangle = \langle \boldsymbol{x}, \boldsymbol{y}_1\rangle$ over $\mathbb{Z}$.

Game 1: This game is like Game 0, except that the challenger aborts the experiment in the event that one of the master secret key components $\{s_i\}_{i=1}^{\ell}$ is too close to the extremities of the interval $\{-2^{\lambda+1}N^4, \ldots, 2^{\lambda+1}N^4\}$. Specifically, we define F_1 to be the event that one of the integers s_i lands in

$$\{-2^{\lambda+1}N^4, \ldots, -2^{\lambda+1}N^4 + 2N^4\} \cup \{2^{\lambda+1}N^4 - 2N^4, \ldots, 2^{\lambda+1}N^4\}.$$

This event only happens with negligible probability $\Pr[F_1] \leq \ell/2^\lambda$, so that $|\Pr[S_1] - \Pr[S_0]| \leq \Pr[F_1] \leq \ell/2^\lambda$.

Game 2: We modify the generation of the challenge $C_{\boldsymbol{y}_\beta} = (C_0, C_1, \ldots, C_\ell)$. Namely, the challenger $\mathcal{B}$ first chooses $z = z_0^N \bmod N^2$, for a randomly drawn $z_0 \hookleftarrow \mathbb{Z}_N^*$ and computes

$$C_0 = z^2 \bmod N^2. \tag{E.1}$$

Then, it uses $\mathsf{msk} := (\{s_i\}_{i=1}^{\ell}, X)$ to compute

$$C_i = (1+N)^{y_{\beta,i}} \cdot C_0^{s_i} \bmod N^2, \qquad \forall i \in \{1, \ldots, \ell\}.$$

The ciphertext $C_{\boldsymbol{y}_\beta}$ has almost the same distribution as in Game 1 as C_0 is now perfectly (instead of statistically) uniform in the subgroup of $(2N)$th residues. We have $|\Pr[S_2] - \Pr[S_1]| \leq 2^{-\lambda}$.

Game 3: We modify again the generation of $C_{\boldsymbol{y}_\beta} = (C_0, C_1, \ldots, C_\ell)$ in the challenge phase. Namely, instead of computing C_0 by first choosing a random Nth residue z in $\mathbb{Z}_{N^2}^*$, the challenger rather samples $z \hookleftarrow \mathbb{Z}_{N^2}^*$ at random, computes C_0 as in as in (E.1) (so that C_0 is a square in $\mathbb{Z}_{N^2}^*$ but not a Nth residue, except with negligible probablity) and sets

$$C_i = (1+N)^{y_{\beta,i}} \cdot C_0^{s_i} \bmod N^2, \qquad \forall i \in \{1, \ldots, \ell\}.$$

Under the 1-DCR assumption, this modification is not noticeable to $\mathcal{A}$, which implies that $|\Pr[S_3] - \Pr[S_2]| \leq \mathbf{Adv}_{\mathcal{B}}^{\text{1-}\mathsf{DCR}}(\lambda)$.

We argue that $|\Pr[S_3]-1/2| < 2^{-l(\lambda)+1}$. To see this, we first remark that, since g generates the subgroup of $(2N)$th residues in $\mathbb{Z}_{N^2}^*$, the ciphertext component C_0 can be written $C_0 = (1+N)^{a_z} \cdot g^{r_z} \bmod N^2$ for some uniformly random $a_z \in \mathbb{Z}_N$ and $r_z \in \mathbb{Z}_{p'q'}$. Note that, with all but negligible probability $2/\min(p,q) < 2^{-l(\lambda)+1}$, integer a_z is invertible modulo N. For each $i \in \{1, \ldots, \ell\}$, we thus have

$$C_i = (1+N)^{y_{\beta,i} + a_z \cdot (s_i \bmod N)} \cdot g^{r_z \cdot s_i} \bmod N^2. \tag{E.2}$$

Since $\gcd(N, p'q') = 1$, there exist $u, v \in \mathbb{Z}$ with $|u| < p'q' < N$ and $|v| < N$ such that $u \cdot N + v \cdot (p'q') = 1$. For each $i \in \{1, \ldots, \ell\}$, let us define the integer

$$s_i' = s_i + (a_z^{-1} \bmod N) \cdot (y_{\beta,i} - y_{1-\beta,i}) \cdot (v \cdot p'q') \in \mathbb{Z}$$

which satisfies the equalities $s_i' = s_i \bmod p'q'$ (and thus $g^{s_i} = g^{s_i'} \bmod N^2$),

$$s_i' = s_i + (a_z^{-1} \bmod N) \cdot (y_{\beta,i} - y_{1-\beta,i}) \bmod N$$

as well as

$$\begin{aligned} C_i &= (1+N)^{y_{1-\beta,i}} \cdot C_0^{s_i'} \bmod N^2 \\ &= (1+N)^{y_{1-\beta,i} + a_z \cdot (s_i' \bmod N)} \cdot g^{r_z \cdot s_i'} \bmod N^2, \qquad \forall i \in \{1, \ldots, \ell\}. \end{aligned} \tag{E.3}$$

Since $|v| < N$, we also have

$$|(a_z^{-1} \bmod N)(y_{\beta,i} - y_{1-\beta,i}) \cdot (v \cdot p'q')| < 2N^4, \tag{E.4}$$

so that $s_i' \in \{-2^{\lambda+1}N^4, \ldots, 2^{\lambda+1}N^4\}$ unless the failure event F_1 of Game 1 occurs.

Observe that, for any vector $\boldsymbol{x}$ such that $\langle \boldsymbol{x}, \boldsymbol{y}_0 - \boldsymbol{y}_1 \rangle = 0$ over $\mathbb{Z}$, we have $\sum_{i=1}^{\ell} s_i \cdot x_i = \sum_{i=1}^{\ell} s_i' \cdot x_i$. It comes that $\mathcal{A}$'s view is compatible with both possible master secret keys $\{s_i\}_{i=1}^{\ell}$ and $\{s'_i\}_{i=1}^{\ell}$ as long as $(a_z^{-1} \bmod N)$ exists and $\mathcal{A}$ only obtains secret keys $\mathsf{sk}_{\boldsymbol{x}}$ for vectors $\boldsymbol{x}$ such that $\langle \boldsymbol{x}, \boldsymbol{y}_0 \rangle = \langle \boldsymbol{x}, \boldsymbol{y}_1 \rangle$ over $\mathbb{Z}$. Moreover, we claim that $\{s'_i\}_{i=1}^{\ell}$ is as plausible as $\{s_i\}_{i=1}^{\ell}$ as a master secret key. Indeed, $\{s_i\}_{i=1}^{\ell}$ are chosen uniformly in an interval which is the concatenation of 2^λ sub-intervals of width $2N^4$ each. So, unless F_1 occurs, each s'_i is as likely as s_i which is either in the same sub-interval or an adjacent one.

Unless event F_1 occurs, Equations (E.2) and (E.3) show that $(C_0, C_1, \ldots, C_\ell)$ can be interpreted as an encryption of $\boldsymbol{y}_\beta$ with the same likelihood for each $\beta \in \{0,1\}$.

When counting probabilities, we find that $\mathcal{A}$'s advantage can be bounded as

$$|\Pr[S_0] - 1/2]| \leq \mathbf{Adv}_{\mathcal{B}}^{\text{1-DCR}}(\lambda) + (\ell+1)2^{-\lambda} + 2^{-l(\lambda)+1},$$

which is thus negligible if the 1-DCR assumption holds. □

F Proof of Theorem 6

Proof. The proof proceeds similarly to the proof of Theorem 5. For each i, we denote by S_i the event that the adversary wins in Game i.

Game 0: This is the actual security game. Concretely, the adversary $\mathcal{A}$ is given the master public key $\mathsf{mpk} = \big(N, g, \{h_i\}_{i=1}^{2\ell}\big)$, where $h = g^{s_i} \bmod N^4$ for all $i \in \{1, \ldots, 2\ell\}$, each s_i being sampled uniformly in $\{-2^{\lambda+1}N^{10}, \ldots, 2^{\lambda+1}N^{10}\}$. In the challenge phase, adversary $\mathcal{A}$ chooses two distinct vectors $\boldsymbol{y}_0, \boldsymbol{y}_1 \in \mathbb{Z}_N^\ell$ and obtains an encryption of $\boldsymbol{y}_\beta = (y_{\beta,1}, \ldots, y_{\beta,\ell})$ for a random bit $\beta \hookleftarrow \{0,1\}$ sampled by the challenger $\mathcal{B}$. When $\mathcal{A}$ halts, it outputs $\beta' \in \{0,1\}$ and we call S_0 the event that $\beta' = \beta$. For any vector $\boldsymbol{x} \in \mathbb{Z}_N^\ell$ submitted to the secret key extraction oracle, it must be the case that $\langle \boldsymbol{x}, \boldsymbol{y}_0 \rangle = \langle \boldsymbol{x}, \boldsymbol{y}_1 \rangle \bmod N$.

Game 1: This game is like Game 0, except that the challenger aborts the game in the event that one of the master secret key components $\{s_i\}_{i=1}^{2\ell}$ or one of the random encryption coins $\{t_{\ell+i}\}_{i=1}^{\ell}$ is too close to the extremities of its interval. Namely, we define F_1 as the event that one of the integers s_i lands in

$$\{-2^{\lambda+2}N^{10}, \ldots, -2^{\lambda+2}N^{10} + 2N^{10}\} \cup \{2^{\lambda+2}N^{10} - 2N^{10}, \ldots, 2^{\lambda+2} \cdot N^{10}\}$$

or one of the $\{t_{\ell+i}\}_{i=1}^{\ell}$ (interpreted as an element of $\{0, \ldots, N^2-1\}$) falls in

$$\{0, \ldots, 2\ell N\} \cup \{N^2 - 2\ell N - 1, \ldots, N^2 - 1\}.$$

(Note that $2\ell N$ is much smaller than N^2 since ℓ is polynomial.) The probability of F_1 is at most

$$\Pr[F_1] \leq \frac{2\ell}{2^{\lambda+1}} + \ell \cdot \frac{4\ell}{N} < \frac{\ell}{2^\lambda} + \frac{\ell^2}{2^{l(\lambda)-1}},$$

which is negligible. We have $|\Pr[S_1] - \Pr[S_0]| < \frac{\ell}{2^\lambda} + \frac{\ell^2}{2^{l(\lambda)-1}}$.

Game 2: We modify the generation of the challenge $C_{\boldsymbol{y}_\beta} = (C_0, C_1, \ldots, C_{2\ell})$. Namely, the challenger $\mathcal{B}$ first chooses $z = z_0^{N^3} \bmod N^4$, for a random $z_0 \hookleftarrow \mathbb{Z}_N^*$ and computes

$$C_0 = z^2 \bmod N^4. \tag{F.1}$$

Then, it samples $t_{\ell+1}, \ldots, t_{2\ell} \hookleftarrow \mathbb{Z}_{N^2}$ and uses $\mathsf{msk} := \{s_i\}_{i=1}^{2\ell}$ to compute

$$\begin{aligned} C_i &= (1+N)^{y_{\beta,i}} \cdot C_0^{s_i} \bmod N^4, & \forall i \in \{1, \ldots, \ell\}, \\ C_i &= (1+N)^{t_i N} \cdot C_0^{s_i} \bmod N^4, & \forall i \in \{\ell+1, \ldots, 2\ell\}. \end{aligned}$$

The ciphertext $C_{\boldsymbol{y}_\beta}$ has nearly the same distribution as in Game 1 and we have $|\Pr[S_2] - \Pr[S_1]| \le 2^{-\lambda}$.

Game 3: We change again the generation of $C_{\boldsymbol{y}_\beta} = (C_0, C_1, \ldots, C_{2\ell})$ in the challenge phase. Namely, instead of computing C_0 by first choosing a random (N^3)th residue z in $\mathbb{Z}_{N^4}^*$, the challenger rather picks $z \hookleftarrow \mathbb{Z}_{N^4}^*$ at random, computes C_0 as in as in (F.1) and sets

$$\begin{aligned} C_i &= (1+N)^{y_{\beta,i}} \cdot C_0^{s_i} \bmod N^4, & \forall i \in \{1, \ldots, \ell\}, \\ C_i &= (1+N)^{t_i N} \cdot C_0^{s_i} \bmod N^4, & \forall i \in \{\ell+1, \ldots, 2\ell\}. \end{aligned}$$

Under the 3-DCR assumption, this modification has no noticeable effect on $\mathcal{A}$, so that $|\Pr[S_3] - \Pr[S_2]| \le \mathbf{Adv}_{\mathcal{B}}^{3\text{-}\mathsf{DCR}}(\lambda)$.

We argue that $|\Pr[S_3] - 1/2| < 2^{-3l(\lambda)+1}$. To see this, we first remark that, since g generates the subgroup of $(2N^3)$th residues, ciphertext component C_0 can be written $C_0 = (1+N)^{a_z} \cdot g^{r_z} \bmod N^4$ for some uniformly random $a_z \in \mathbb{Z}_{N^3}$ and $r_z \in \mathbb{Z}_{p'q'}$. With all but negligible probability $2/\min(p^3, q^3) < 2^{-3l(\lambda)+1}$, integer a_z is invertible modulo N^3. We have

$$\begin{aligned} C_i &= (1+N)^{y_{\beta,i} + a_z \cdot (s_i \bmod N^3)} \cdot g^{r_z \cdot s_i} \bmod N^4, & \forall i \in \{1, \ldots, \ell\}, \\ C_i &= (1+N)^{t_i N + a_z \cdot (s_i \bmod N^3)} \cdot g^{r_z \cdot s_i} \bmod N^4, & \forall i \in \{\ell+1, \ldots, 2\ell\}. \end{aligned}$$

We know that the adversary $\mathcal{A}$ can only query secret keys for vectors $\boldsymbol{x} \in \mathbb{Z}_N^\ell$ such that $\langle \boldsymbol{x}, \boldsymbol{y}_\beta - \boldsymbol{y}_{1-\beta} \rangle = 0 \bmod N$. For each secret key query $\boldsymbol{x}_j$ ($j \in \{1, \ldots, \ell-1\}$) appearing in st by the end of the game, there exists integers $\omega_j \in \mathbb{Z}$ such that $\langle \boldsymbol{x}_j, \boldsymbol{y}_\beta - \boldsymbol{y}_{1-\beta} \rangle = \omega_j \cdot N$ over $\mathbb{Z}$, and $|\omega_j| \le 2\ell N$. We define

$$\begin{aligned} t_i' &= t_i + \omega_{i-\ell}, & \forall i \in \{\ell+1, \ldots, 2\ell-1\}, \\ t_\ell' &= t_\ell, \end{aligned}$$

which are in $\{0, \ldots, N^2-1\}$ unless the failure event F_1 of Game 1 occurs.

Since $\gcd(N^3, p'q') = 1$, there exist $u, v \in \mathbb{Z}$ with $|u| < p'q' < N$ and $|v| < N^3$ such that $u \cdot N^3 + v \cdot (p'q') = 1$. For each $i \in \{1, \ldots, 2\ell\}$, let us define

$$\begin{aligned} s_i' &= s_i + (a_z^{-1} \bmod N^3) \cdot (y_{\beta,i} - y_{1-\beta,i}) \cdot (v \cdot p'q'), & \forall i \in \{1, \ldots, \ell\}, \\ s_i' &= s_i - (a_z^{-1} \bmod N^3) \cdot \omega_{i-\ell} \cdot N \cdot (v \cdot p'q'), & \forall i \in \{\ell+1, \ldots, 2\ell-1\}, \\ s_{2\ell}' &= s_{2\ell}, \end{aligned}$$

which satisfies the equalities $s_i' = s_i \bmod p'q'$ (and thus $g^{s_i} = g^{s_i'} \bmod N^4$) and

$$\begin{aligned}
C_i &= (1+N)^{y_{1-\beta,i}} \cdot C_0^{s_i'} \bmod N^4 \\
&= (1+N)^{y_{1-\beta,i}+a_z\cdot(s_i' \bmod N^3)} \cdot g^{r_z\cdot s_i'} \bmod N^4, && \forall i \in \{1,\ldots,\ell\}, \\
C_i &= (1+N)^{t_i'\cdot N+a_z\cdot(s_i' \bmod N^3)} \cdot g^{r_z\cdot s_i'} \bmod N^4, && \forall i \in \{\ell+1,\ldots,2\ell\}.
\end{aligned}$$

Moreover, since $|v| < N^3$, we have

$$|(a_z^{-1} \bmod N^3)(y_{\beta,i} - y_{1-\beta,i}) \cdot (v \cdot p'q')| < 2N^8,$$

and

$$|(a_z^{-1} \bmod N^3) \cdot \omega_{i-\ell} \cdot N \cdot (v \cdot p'q')| < 2N^{10},$$

so that $s_i' \in \{-2^{\lambda+2}N^{10},\ldots,2^{\lambda+2}N^{10}\}$ for each $i \in \{1,\ldots,2\ell\}$ unless the event F_1 occurs.

Finally, recall that the secret key extraction oracle encodes any vector $\boldsymbol{x}_j \in \mathbb{Z}_N^\ell$ appearing in st as an integer vector of the form $\boldsymbol{X}_j = (X_{j,1},\ldots,X_{j,2\ell}) = (\boldsymbol{x}_j \mid \boldsymbol{e}_j) \in \mathbb{Z}^{2\ell}$ before returning its inner product with the $\{s_i\}_{i=1}^{2\ell}$. Since any of these queried vector $\boldsymbol{x}_j$ satisfies $\langle \boldsymbol{x}_j, \boldsymbol{y}_\beta - \boldsymbol{y}_{1-\beta}\rangle = \omega_j \cdot N$, we have $\sum_{i=1}^{2\ell} s_i \cdot X_{j,i} = \sum_{i=1}^{2\ell} s_i' \cdot X_{j,i}$ over $\mathbb{Z}$, so that $\mathcal{A}$'s view is compatible with both master secret keys $\{s_i\}_{i=1}^{2\ell}$ and $\{s_i'\}_{i=1}^{2\ell}$. By construction, the same holds for the queried vectors not appearing in st. It comes that $\{s_i'\}_{i=1}^{2\ell}$ and $\{t_{\ell+i}'\}_{i=1}^{\ell}$ explain $(C_0,\ldots,C_{2\ell})$ as an encryption of $\boldsymbol{y}_{1-\beta}$ just as well as $\{s_i\}_{i=1}^{2\ell}$ and $\{t_{\ell+i}\}_{i=1}^{\ell}$ explain it as an encryption of $\boldsymbol{y}_\beta$. We thus have $|\Pr[S_3] - 1/2| < 2^{-3l(\lambda)+1}$, as claimed.

The adversary's advantage can thus be bounded as

$$|\Pr[S_0] - 1/2]| \leq \mathbf{Adv}_{\mathcal{B}}^{\text{3-DCR}}(\lambda) + \frac{\ell+1}{2^\lambda} + \frac{\ell^2}{2^{l(\lambda)-1}} + \frac{1}{2^{3l(\lambda)}}.$$

$\square$