Online-Offline Functional Encryption for Bounded Collusions

Shweta Agrawal* Alon Rosen†

Abstract

We give a new construction of bounded key functional encryption. Our scheme is well suited for optimization in an *online-offline model* that allows for preparation in an offline phase, where a majority of the computation is done before the data becomes available. This is followed by an efficient online phase, which is performed when the data becomes known. Such a model has been considered in the context of Attribute-Based Encryption by Hohenberger and Waters (PKC'14).

The online component of our scheme significantly outperforms the best previously known construction of bounded key functional encryption by Gorbunov, Vaikuntanathan and Wee (CRYPTO'12), and in fact quasi-linearly depends *only on the message size* in contrast to the GVW12 ciphertext, which additionally grows as $O(q^4)$ for q queries. Security of our scheme is based on the Ring LWE assumption, which is comparable to the assumption underlying the GVW scheme and is well-established compared to those underlying known constructions of unbounded key functional encryption (based on multilinear maps and/or obfuscation).

To prove security of our scheme, we introduce a new proof technique, which we call *noisy functional encryption*. Arguing security via this technique requires the encryptor to artificially add noise to the decryption equation, providing an intriguing tradeoff between correctness and security. This technique appears to be quite general and we believe it is likely to have other applications.

*IIT Delhi, India. Email: shweta.a@gmail.com. Work done (in part) while visiting IDC Herzliya, supported by the ERC under the EU's Seventh Framework Programme (FP/2007-2013) ERC Grant Agreement n. 307952. Work done (in part) while visiting the Simons Institute for the Theory of Computing, supported by the Simons Foundation and by the DIMACS/Simons Collaboration in Cryptography through NSF grant CNS-1523467.

†Efi Arazi School of Computer Science, IDC Herzliya, Israel. Email: alon.rosen@idc.ac.il. Supported by ISF grant no. 1255/12, NSF-BSF Cyber Security and Privacy grant no. 2014/632, and by the ERC under the EU's Seventh Framework Programme (FP/2007-2013) ERC Grant Agreement n. 307952. Work done (in part) while visiting the Simons Institute for the Theory of Computing, supported by the Simons Foundation and by the DIMACS/Simons Collaboration in Cryptography through NSF grant CNS-1523467.

1 Introduction

Functional encryption (FE) [SW05, SW] generalizes public key encryption to allow fine grained access control on encrypted data. In functional encryption, a user can be provided with a secret key corresponding to a function g, denoted by SK_g. Given SK_g and ciphertext $\mathsf{CT}_{\mathbf{x}}$ encrypting a message $\mathbf{x}$, the user may run the decryption procedure to learn the value $g(\mathbf{x})$. Security of the system guarantees that nothing beyond $g(\mathbf{x})$ can be learned from $\mathsf{CT}_{\mathbf{x}}$ and SK_g.

Recent years have witnessed significant progress towards constructing functional encryption for advanced functionalities [BF01, Coc01, BW06, BW07, GPV08, CHKP10, ABB10, GPSW06, BSW07, KSW08, LOS+10, AFV11, Wat12, GVW13, GGH+13c, GGH+13b, GVW15]. However, for the most general notion of functional encryption – one that allows the evaluation of arbitrary efficiently-computable functions $g(\mathbf{x})$ of the message $\mathbf{x}$, and is secure against general adversaries, the only known constructions rely on indistinguishability obfuscation [GGH+13b] or alternatively on certain hardness assumptions on multilinear maps [GGHZ14]. Reliance on such primitives is in some sense inherent, as full-fledged functional encryption has been recently shown to imply indistinguishability obfuscation [AJ15, BV15, AJS]. Unfortunately, all known candidate multilinear map constructions [GGH13a, CLT13, GGH15] as well as indistinguishability obfuscation [GGH+13b] have been recently broken [CHL+15, CGH+15, HJ15, CJL, CFL+, MSZ], rendering all constructions of general FE uninstantiable.

Faced with this state of affairs, it is increasingly important to base constructions of functional encryption on well understood hardness assumptions. Additionally, as functional encryption gains in popularity, a significant challenge that lies enroute to making it practical is reducing the costs of encryption and key generation. One potential way around these obstacles is to focus on weaker notions of functional encryption which may be amenable to more efficient realizations as well as based on more well-established hardness assumptions.

1.1 Bounded-Collusion Functional Encryption

Gorbunov, Vaikuntanathan and Wee [GVW12] studied the notion of *bounded collusion functional encryption*. In this notion, the size of the ciphertext may depend polynomially on the number q of function queries. The notion of q bounded FE is very appealing, since in practice it may often be possible to a-priori upper bound the number of function keys that are released. Indeed, for the special case of Identity Based Encryption (IBE), the question of designing schemes with bounded collusions has been considered in a number of works [DKXY02, CHH+07, GLW12].

The [GVW12] construction is ingenious, relying only on public key encryption and leveraging ideas from multiparty computation (we outline their construction in Section A.1). However, its encryption algorithm is inherently unsuited to the online-offline model – to encrypt a message $\mathbf{x}$, the encryptor must secret share it into $N = O(q^4)$ shares, and encrypt each one with a one-query FE scheme. This results in encryption time that degrades as $O(q^4)$ *after* the message becomes available. As both the size of datasets and the number of queries grows, this blowup can be prohibitive, rendering the scheme impractical.

1.2 Online-Offline Cryptography

In recent years, as computation is increasingly moving to small devices such as mobile phones, a trend for *online-offline cryptography* has been mounting. At a high level, this model allows for an expensive preparation or offline phase, where a majority of the computation is done before the data becomes available. This is followed by an efficient online phase, which is performed when the data becomes known. As noted by Hohenberger and Waters [HW14], an illustrative motivating scenario is to consider a mobile device with restrictions on power consumption. In this case, it is desirable to distribute the computation so that offline computations can be performed when the device is plugged into a power source and online computations are performed on demand, without significantly draining the battery.

Examples of online-offline cryptography abound in the literature. The notion of online-offline cryptography was introduced in the context of signatures, by Even, Goldreich and Micali [EGM89] in 1989, and has been studied extensively since then in the context of various cryptographic primitives. Examples include signatures [ST01], zero knowledge [CLP13], delegation of computation [CKV10], attribute based encryption [HW14]. In the context of delegation of computation, a client wishes to outsource computation of some expensive function f to a server. Here, it is standard to require the client to do a significant amount of work (proportional to the size of f) in an offline phase and then do substantially less work in the online phase, hopefully amortizing the initial investment over several runs. In the context of multiparty computation, this question has received particular attention. We refer the reader to [Bea95, IPS08, BDOZ11, DPSZ12, IKM+13, LR14] and references therein.

Another classic primitive studied in this model is garbled circuits [Yao82], or more generally randomized encodings [IK02, AIK06]. As observed by Applebaum et al. [AIK11], in the setting where we desire a weak computational device in a field to perform computation on sensitive data, the only known solution relies on performing an offline phase before the device is sent to the field, and an efficient online phase conducted on field. In the context of functional encryption related primitives, offline-online versions have already been considered for identity based encryption [GMC08, LZ09] and attribute based encryption [HW14].

1.3 Our Results

We give a new construction of bounded key Functional Encryption that is highly suitable for the online-offline model. The online component of our scheme significantly outperforms the best previously known construction of bounded key functional encryption by Gorbunov, Vaikuntanathan and Wee (CRYPTO'12), and in fact quasi-linearly depends *only on the message size* in contrast to the GVW12 ciphertext, which additionally grows as $O(q^4)$ for q queries. Security is based on the Ring LWE assumption, which is comparable to the assumption underlying the GVW scheme (they rely on public key encryption) and is well-established compared to those underlying known constructions of unbounded key functional encryption (based on multilinear maps and/or obfuscation).

Just as in the GVW scheme, the ciphertext size of our scheme scheme grows with the maximal circuit size $|C_g|$ of the function $g(\mathbf{x})$ to be evaluated. However, while for GVW the ciphertext size degrades as $O(q^4|C_g|)$, in our scheme the total ciphertext size is $O(q^2|C_g|)$. Even more importantly, the complexity of the online phase in our scheme is $O(|\mathbf{x}|)$, where $|\mathbf{x}|$ is the size of the plaintext. In other words, online encryption time is *independent* of both $|C_g|$ and q. We note that beyond having

significantly larger ciphertexts than our scheme, the GVW scheme appears to be unamenable for online-offline optimization (see Section A.1).

As a side comment we mention that for the restricted class of FE for bounded degree polynomials, our scheme enjoys a *local updateability* property, namely, if a single element of the plaintext changes, only a constant number of encryptions have to be updated. However, we lose this property after bootstrapping to general circuits, since the randomized encodings we rely on are only output local, and not input local.

Our construction crucially relies on a newly introduced notion of *noisy functional encryption*. We provide a new construction for noisy functional encryption for linear functions, secure against q queries, and use it to obtain a construction of online-offline FE for quadratic forms. The latter is then carefully extended to FE for bounded degree polynomials using an inductive approach. (While such a construction could have been obtained in a straightforward way by linearizing the message and invoking any functional encryption for linear functions, we note that this would violate the online-offline and local updateability properties.)

1.4 Techniques and Ideas

We proceed to describe the main ideas underlying our construction. In what follows, we assume familiarity of the reader with the Ring Learning With Errors (RLWE) assumption (refer Section 2 for a refresher). Our construction builds on the dual Regev public key cryptosystem [GPV08, Reg09]. Recall that the ciphertext for a vector $\mathbf{x}$ in the dual Regev cryptosystem is given by:

$$\mathbf{d} = \mathbf{w} \cdot s + p \cdot \boldsymbol{\eta}$$
$$\mathbf{c} = \mathbf{u} \cdot s + p \cdot \boldsymbol{\mu} + \mathbf{x}$$

Here, $s, \boldsymbol{\eta}, \boldsymbol{\mu}$ play the role of randomness, sampled afresh each time by the encryptor, ($\mathbf{w}$, $\mathbf{u}$) are in the public key and the secret key is a *low norm* matrix $\mathbf{E}$ such that $\mathbf{E}^\intercal \mathbf{w} = \mathbf{u}$.

The starting point of our work is the observation that the Dual Regev ciphertext may be viewed in the following way. The vector $\mathbf{c}$ looks precisely like symmetric key fully homomorphic ciphertexts of the BV FHE cryptosystem [BV11b] with FHE secret s, and the vector $\mathbf{d}$ can be seen as a *randomness carrier* for the dual Regev CT. The role of the randomness carrier, as the name suggests, is to "carry" the randomness s which was used in construction of $\mathbf{c}$, so that given the secret key $\mathbf{E}$, the term $\mathbf{c} - \mathbf{E}^\intercal \mathbf{d}$ results in cancellation of the term $\mathbf{u} \cdot s$.

Thus s plays the role of the the symmetric FHE secret key in $\mathbf{c}$, while also playing the role randomness in the dual Regev encryption system. This dual view of s raises the following question: is it possible to compute on $\mathbf{c}$ treating it as FHE symmetric key CTs, to obtain an FHE CT for $g(\mathbf{x})$ for some g, and then switch back to the view of dual Regev encryption and cancel out the large term using a secret key $\mathbf{E}_g$?

Surprisingly, this idea works in a direct and elegant way for linear functions. It is easy to see that given a linear function vector $\mathbf{g}$, one may take linear combinations $\sum g_i c_i$ to obtain a symmetric key ciphertext

$$c_{\mathbf{g}} = \mathbf{g}^\intercal \mathbf{u} \cdot s + \mathbf{g}^\intercal \boldsymbol{\mu} + \langle \mathbf{g}, \mathbf{x} \rangle$$

Now, the key generator who knows $\mathbf{g}$ and the public key $\mathbf{u}$, may compute $u_{\mathbf{g}} = \mathbf{g}^\intercal \mathbf{u}$, sample a preimage $\mathbf{e}_{\mathbf{g}}$ such that $\mathbf{e}_{\mathbf{g}}^\intercal \mathbf{w} = u_{\mathbf{g}}$ and decrypt the dual Regev ciphertext $(\mathbf{w}, c_{\mathbf{g}})$ in the natural way

to obtain $\langle \mathbf{g}, \mathbf{x} \rangle$. This is the essential idea used in recent constructions of FE for linear functions [ABCP15, ALS].

The key reason this works for linear functions is that the FHE ciphertext $c_{\mathbf{g}}$ is encrypted under the label $u_{\mathbf{g}}$ which is publicly computable. However, extending this idea to beyond linear functions poses at least two problematic issues, as follows.

1. If we multiply the CTs c_i and c_j using BV style fully homomorphic encryption [BV11b, BV11a], the resultant CT is encrypted under a label $u_{\mathbf{g}}$ which is not publicly computable. Indeed, the label $u_{\mathbf{g}}$ depends not only on the function $\mathbf{g}$ and the public key $\mathbf{u}$ but also on the original ciphertext $\mathbf{c}$ which the KeyGen algorithm has no way of knowing.

2. FHE ciphertexts are malleable. In the linear FE scheme described above, even if the attacker malleates the ciphertext $\mathbf{c}$ which encrypts $\mathbf{x}$ to a new CT that encrypts $\mathbf{x} + \mathbf{y}$, this does not help him distinguish the challenge ciphertext in the security game. This is because the key $\mathsf{SK}_{\mathbf{g}}$ will decrypt the above malleated ciphertext to $\mathbf{g}^\intercal(\mathbf{x}_b + \mathbf{y})$ which does not offer any advantage if $\mathbf{g}^\intercal \mathbf{x}_0 = \mathbf{g}^\intercal \mathbf{x}_1$.

   However, this is no longer the case for non-linear functions. For example, if an adversary has a key $\mathbf{e}_{ij}$ to compute the monomial $x_i x_j$ then it holds that $\mathbf{x}_0[i]\mathbf{x}_0[j] = \mathbf{x}_1[i]\mathbf{x}_1[j]$ for an admissible adversary. But now, suppose he could malleate the ciphertext to encrypt $\mathbf{x} + \mathbf{y}$ then it may no longer hold that $(\mathbf{x}_0 + \mathbf{y})[i](\mathbf{x}_0 + \mathbf{y})[j] = (\mathbf{x}_1 + \mathbf{y})[i](\mathbf{x}_1 + \mathbf{y})[j]$.

We address both these issues by providing additional advice – encryptions of $c_i \cdot s$, which enable the keygen to compute $u_{\mathbf{g}}$ that does not depend on the ciphertext, as well as which binds the ciphertext to the secret key [1], ruling out the malleability attack. We now proceed to describe our construction.

Through the remainder of the informal discussion, we will focus on FE to compute monomials. This is without loss of generality, since there already exist constructions for linear FE [ABCP15, ALS] which can be leveraged to support arbitrary quadratic polynomials once we build FE that can handle monomials [2]. Additionally, the ability to handle multiplication is the primary bottleneck in the construction of these systems, analogously to FHE. A trivial solution would be to linearize the polynomail and encrypt each $x_i x_j$ term separately using the Linear FE scheme, but this destroys the online-offline and local updateability properties of the scheme as we shall see below.

Let us consider the ring LWE based symmetric key FHE scheme of [BV11b]. The main observation in [BV11b] was that if:

$$c_i = u_i \cdot s + \mu_i + x_i$$
$$c_j = u_j \cdot s + \mu_j + x_j$$

then the decryption equation can be written as

$$x_i x_j \approx c_i c_j + (u_i u_j)s^2 - (u_j c_i)s - (u_i c_j)s$$

[1] Note that in predicate encryption systems, the need for binding does not arise, since the message is already bound to the lwe secret by construction.

[2] FE for linear functions and FE for monomials must be compatible for this to work, but this will the case.

[BV11b] observed that it is sufficient to add one ciphertext element per level of the circuit to propagate the computation. A bit more formally, the 3 tuple $(c_i c_j,\ u_i c_j + u_j c_i,\ u_i u_j)$ is a legitimate level 2 FHE ciphertext, decryptable by the secret key s.

In the context of FE for monomials, things are significantly more complex, since we must return a key that allows the decryptor to learn $x_i x_j$ and nothing else. However, the above decryption equation holds potential even for this setting. In particular, observe that in the above equation, the parenthesis can be shifted so that:

$$x_i x_j \approx c_i c_j + u_i u_j(s^2) - u_j(c_i s) - u_i(c_j s)$$

Now, if we use the Linear FE scheme to encrypt the terms in parenthesis, then we can have the decryptor recover the term $u_i u_j(s^2) - u_j(c_i s) - u_i(c_j s)$. More formally, let LinFE be an FE scheme that supports computation of linear functions. Constructions for the same were provided recently [ABCP15, ALS]. Now if,

$$\mathsf{CT} = \mathsf{LinFE.Enc}(s^2, c_1 \cdot s, \ldots, c_w \cdot s)$$
$$\mathsf{SK} = \mathsf{LinFE.KeyGen}(u_i u_j, -0-, u_i, -0-, u_j, -0-)$$

then, $\mathsf{LinFE.Dec}(\mathsf{SK}, \mathsf{CT})$ should yield the above term by correctness. And this is sufficient, since the term $c_i c_j$ can compute directly by the decryptor, which enables him to recover $x_i x_j$.

A bit more abstractly, we observe that a quadratic plaintext can be represented a quadratic polynomial which is quadratic in terms that are known publicly, and only *linear* in secret terms (upto multiplication by s). Since the number of secret terms $c_i s$ which must be encrypted, is only linear in w, we appear to avoid the quadratic blowup caused by linearization. Looking ahead, one may hope that by relinearizing the *randomness* of the dual Regev CT, which is the secret key of the BV FHE, one can propagate the computation down the circuit a la BV, and recover the desired value in the end.

This intuition, while appealing, is very misleading. We describe three thorny issues that arise. First off, note that if we permit the decryptor to learn the term $u_i u_j s^2 - u_j c_i s - u_i c_j s$ exactly, then he can recover exact quadratic equations in the secret s, completely breaking the security of the scheme. To handle this, we resort to the standard trick that takes trivial to intractable – adding noise. Thus, instead of using Linear FE, we will use a *noisy linear functional encryption scheme* in the sense of Section 3, which we construct in Section 5. This takes care of the above attack, as we shall see below. However, to handle Q queries, we need Q fresh noise terms so this introduces the dependence of the CT size on Q.

However, the scheme remains unsimulatable. This is because in FE, the adversary also requests function keys, and the simulated ciphertexts and function keys must work correctly together. Typically, the simulator "programs" these to satisfy the requisite dependencies. However, when we attempt this here, we have to contend with w^2 linear equations in w variables, and adding the noise terms above does not help resolve this issue. To handle this, we introduce additional terms in the key so as to create sufficient degrees of freedom. In the simulation however, these terms must depend on the challenge ciphertext, so we must carefully argue indistinguishability. The details of this construction and proof are provided in Section 4.

The third issue is presented by the cross terms $c_i s$ – if encryptions of $c_i s$ are required to compute level 2 CTs, say c_k^2 (where superscript denotes level), then to proceed to level 3, we require encryptions of $c_k^2 s$, and this involves generating an encryption of $c_i c_j s$. We handle this by again

plugging in the efficient quadratic system from above, this time for messages $y_i = c_i$ and $y_j = c_j s$. By carefully generating ciphertexts encrypting cross terms by recursively plugging in our quadratic scheme, we achieve a constant factor blowup per level. Thus, at level d, we can provide advice of size $O(2^d)$ which enables the decryptor to compute CTs that encrypt level d monomials. The CT so far is succinct, and contains encryptions of the terms we want. However, recall that releasing keys to decrypt these CTs as-is is insecure, so we must have the encryptor add encryptions of noise terms, and this is what introduces the dependency of CT size on Q. Details of this construction are described in Section 6.

Now, we come to the key point. In the resulting construction for quadratic forms, the number of ciphertext elements that depend on the message vector $\mathbf{x} \in \mathbb{Z}_{p_0}^w$ is only $O(w)$ the remaining $O(w^2)$ encryptions are of i.i.d noise terms which may be generated offline. This results in an online encryption complexity of $O(w)$ which is *independent of the number of queries requested by the adversary*, in contrast to [GVW12]. Additionally, even the induction and bootstrapping do not violate this property, as discussed in Section 6.

1.5 Organization of the paper

The paper is organized as follows. In Section 2, we describe preliminary definitions and notation used throughout the paper. In Section 3, we define the notion of noisy functional encryption which we rely on crucially for our construction. In Section 4, we describe a construction for FE for quadratic forms for the online-offline model, which relies on a a construction for noisy functional encryption for linear functions, secure against Q queries, provided in Section 5. In Section 6, we extend our quadratic FE to FE for bounded degree polynomials.

2 Preliminaries

In this section, we define the preliminaries we require for our constructions. We begin with defining the notation that we will use throughout the paper.

Notation. We say a function $f(n)$ is *negligible* if it is $O(n^{-c})$ for all $c > 0$, and we use $\mathrm{negl}(n)$ to denote a negligible function of n. We say $f(n)$ is *polynomial* if it is $O(n^c)$ for some $c > 0$, and we use $\mathrm{poly}(n)$ to denote a polynomial function of n. We say an event occurs with *overwhelming probability* if its probability is $1 - \mathrm{negl}(n)$. The function $\lg x$ is the base 2 logarithm of x. The notation $\lfloor x \rceil$ denotes the nearest integer to x, rounding towards 0 for half-integers.

2.1 Functional Encryption

Let $\mathcal{X} = \{\mathcal{X}_\lambda\}_{\lambda \in \mathbb{N}}$ and $\mathcal{Y} = \{\mathcal{Y}_\lambda\}_{\lambda \in \mathbb{N}}$ denote ensembles where each $\mathcal{X}_\lambda$ and $\mathcal{Y}_\lambda$ is a finite set. Let $\mathcal{G} = \{\mathcal{G}_\lambda\}_{\lambda \in \mathbb{N}}$ denote an ensemble where each $\mathcal{G}_\lambda$ is a finite collection of circuits, and each circuit $g \in \mathcal{G}_\lambda$ takes as input a string $x \in \mathcal{X}_\lambda$ and outputs $g(x) \in \mathcal{Y}_\lambda$.

A functional encryption scheme $\mathcal{F}$ for $\mathcal{G}$ consists of four algorithms $\mathcal{F} = (\mathsf{FE.Setup}, \mathsf{FE.Keygen}, \mathsf{FE.Enc}, \mathsf{FE.Dec})$ defined as follows.

- $\mathsf{FE.Setup}(1^\lambda)$ is a p.p.t. algorithm takes as input the unary representation of the security parameter and outputs the master public and secret keys $(\mathsf{MPK}, \mathsf{MSK})$.

- $\mathsf{FE.Keygen}(\mathsf{MSK}, g)$ is a p.p.t. algorithm that takes as input the master secret key MSK and a circuit $g \in \mathcal{G}_\lambda$ and outputs a corresponding secret key SK_g.
- $\mathsf{FE.Enc}(\mathsf{MPK}, x)$ is a p.p.t. algorithm that takes as input the master public key MPK and an input message $x \in \mathcal{X}_\lambda$ and outputs a ciphertext CT.
- $\mathsf{FE.Dec}(\mathsf{SK}_g, \mathsf{CT}_x)$ is a deterministic algorithm that takes as input the secret key SK_g and a ciphertext CT_x and outputs $g(x)$.

In order to capture the notion of *online-offline* processing, we will split the $\mathsf{FE.Enc}(\mathsf{MPK}, x)$ into two procedures:

- $\mathsf{FE.EncOff}(\mathsf{MPK})$ is a p.p.t. algorithm that takes as input the master public key MPK and outputs a "temporary" state st.
- $\mathsf{FE.EncOn}(\mathsf{MPK}, st, x)$ is a p.p.t. algorithm that takes as input the master public key MPK a state st and an input message $x \in \mathcal{X}_\lambda$ and outputs a ciphertext CT.

We will denote by $\mathsf{FE.Enc}(\mathsf{MPK}, x)$ the output of $\mathsf{FE.EncOn}(\mathsf{MPK}, st, x)$ where $st = \mathsf{FE.EncOff}(\mathsf{MPK})$. We will be generally interested in the running time of $\mathsf{FE.EncOn}$.

Definition 2.1 (Correctness)**.** A functional encryption scheme $\mathcal{F}$ is correct if for all $g \in \mathcal{G}_\lambda$ and all $x \in \mathcal{X}_\lambda$,

$$\Pr\left[\begin{array}{l}(\mathsf{MPK}, \mathsf{MSK}) \leftarrow \mathsf{FE.Setup}(1^\lambda);\\ \mathsf{FE.Dec}\Big(\mathsf{FE.Keygen}(\mathsf{MSK}, g), \mathsf{FE.Enc}(\mathsf{MPK}, x)\Big) \neq g(x)\end{array}\right] = \mathrm{negl}(\lambda)$$

where the probability is taken over the coins of $\mathsf{FE.Setup}$, $\mathsf{FE.Keygen}$, and $\mathsf{FE.Enc}$.

2.2 Indistinguishability-based Definition of Security

There has been extensive research on understanding the "right" definition of security for functional encryption. In this paper we will consider the standard (adaptive) indistinguishability based definition.

Definition 2.2. A functional encryption scheme $\mathcal{F}$ for a function family $\mathcal{G}$ is secure in the adaptive indistinguishability game, denoted as IND secure, if for all probabilistic polynomial-time adversaries $\mathcal{A}$, the advantage of $\mathcal{A}$ in the following experiment is negligible in the security parameter λ:

1. Public Key: Challenger returns MPK to the adversary.
2. Pre-Challenge Key Queries: $\mathcal{A}$ may adaptively request keys for any functions $g_1, \ldots, g_{\ell'} \in \mathcal{G}$. In response, $\mathcal{A}$ is given the corresponding keys SK_{g_i}.
3. $\mathcal{A}(1^\lambda, \mathsf{MPK})$ outputs the challenges $(\mathbf{x}_0, \mathbf{x}_1) \in \mathcal{X}$ to the challenger, subject to the restriction that $g_i(\mathbf{x}_0) = g_i(\mathbf{x}_1)$ for all $i \in [\ell']$
4. Challenge CT: $\mathcal{A}$ requests the challenge ciphertext, to which challenger chooses a random bit b, and returns the ciphertext $\mathsf{CT}_{\mathbf{x}_b}$.

5. Key Queries: The adversary may continue to request keys for additional functions, subject to the restriction that $g_i(\mathbf{x}_0) = g_i(\mathbf{x}_1)$ for all $i \in \{\ell' + 1, \ldots, \ell\}$. In response, $\mathcal{A}$ is given the corresponding keys SK_{g_i}.

6. $\mathcal{A}$ outputs a bit b', and succeeds if $b' = b$.

The *advantage* of $\mathcal{A}$ is the absolute value of the difference between its success probability and $1/2$. We note that without loss of generality, in the selective game, the challenge ciphertext can be returned along with the public key.

2.3 Lattice Preliminaries

An *m-dimensional lattice* Λ is a full-rank discrete subgroup of $\mathbb{R}^m$. A *basis* of Λ is a linearly independent set of vectors whose span is Λ.

Gaussian distributions. Let L be a discrete subset of $\mathbb{Z}^n$. For any vector $\mathbf{c} \in \mathbb{R}^n$ and any positive parameter $\sigma \in \mathbb{R}_{>0}$, let $\rho_{\sigma,\mathbf{c}}(\mathbf{x}) := \mathsf{Exp}\left(-\pi\|\mathbf{x} - \mathbf{c}\|^2/\sigma^2\right)$ be the Gaussian function on $\mathbb{R}^n$ with center $\mathbf{c}$ and parameter σ. Let $\rho_{\sigma,\mathbf{c}}(L) := \sum_{\mathbf{x}\in L} \rho_{\sigma,\mathbf{c}}(\mathbf{x})$ be the discrete integral of $\rho_{\sigma,\mathbf{c}}$ over L, and let $\mathcal{D}_{L,\sigma,\mathbf{c}}$ be the discrete Gaussian distribution over L with center $\mathbf{c}$ and parameter σ. Specifically, for all $\mathbf{y} \in L$, we have $\mathcal{D}_{L,\sigma,\mathbf{c}}(\mathbf{y}) = \frac{\rho_{\sigma,\mathbf{c}}(\mathbf{y})}{\rho_{\sigma,\mathbf{c}}(L)}$. For notational convenience, $\rho_{\sigma,\mathbf{0}}$ and $\mathcal{D}_{L,\sigma,\mathbf{0}}$ are abbreviated as ρ_σ and $\mathcal{D}_{L,\sigma}$, respectively.

The following lemma gives a bound on the length of vectors sampled from a discrete Gaussian.

Lemma 2.3 ([MR07, Lemma 4.4])**.** *Let Λ be an n-dimensional lattice, let $\mathbf{T}$ be a basis for Λ, and suppose $\sigma \geq \|\mathbf{T}\|_{\mathsf{GS}} \cdot \omega(\sqrt{\log n})$. Then for any $\mathbf{c} \in \mathbb{R}^n$ we have*

$$\Pr\left[\|\mathbf{x} - \mathbf{c}\| > \sigma\sqrt{n} : \mathbf{x} \xleftarrow{\mathrm{R}} \mathcal{D}_{\Lambda,\sigma,\mathbf{c}}\right] \;\leq\; \mathrm{negl}(n)$$

We will also need the "noise smudging" or "noise drowning lemma" as follows.

Lemma 2.4 (Drowning Lemma)**.** *[GKPV10] Let $n \in \mathbb{N}$. For any real $\sigma = \omega(\sqrt{\log n})$, and any $\mathbf{c} \in \mathbb{Z}^n$,*

$$\mathsf{SD}(\mathcal{D}_{\mathbb{Z}^n,\sigma},\ \mathcal{D}_{\mathbb{Z}^n,\sigma,\mathbf{c}}) \leq \|\mathbf{c}\|/\sigma$$

Our constructions are based on the hardness of the ring LWE problem [LPR10]. Let $R = \mathbb{Z}[x]/(\phi)$ where $\phi = x^n + 1$ and n is a power of 2. Let $R_q \triangleq R/qR$ where q is a large prime satisfying $q = 1 \mod 2n$.

Ring Learning with Errors. The ring learning with errors assumption, denoted by RLWE, [LPR10] is analogous to the standard LWE assumption introduced by Regev [Reg09]. Let χ be a probability distribution on R_q. For $s \in R_q$, let $A_{s,\chi}$ be the probability distribution on $R_q \times R_q$ obtained by choosing an element $a \in R_q$ uniformly at random, choosing $e \leftarrow \chi$ and outputting $(a, a \cdot s + e)$.

Definition 2.5 ($\mathsf{Ring\ Learning\ With\ Errors}$ - $\mathsf{RLWE}_{\phi,q,\chi}$)**.** The decision R-$\mathsf{LWE}_{\phi,q,\chi}$ problem is: for $s \leftarrow R_q$, given a $\mathrm{poly}(n)$ number of samples that are either (all) from $A_{s,\chi}$ or (all) uniformly random in $R_q \times R_q$, output 0 if the former holds and 1 if the latter holds.

Theorem 2.6 ([LPR10])**.** *Let $r \geq \omega(\sqrt{\log n})$ be a real number and let R, q be as above. Then, there is a randomized reduction from $2^{\omega(\log n)} \cdot (q/r)$ approximate* RSVP *to* $\mathsf{RLWE}_{\phi,q,\chi}$ *where χ is the discrete Gaussian distribution with parameter r. The reduction runs in time* $\mathrm{poly}(n, q)$.

3 Noisy Functional Encryption

Intuitively noisy FE is like regular FE, except that the function value is recovered only up to some noise. This property turns out to be surprisingly useful, as we shall see in subsequent sections. We believe this primitive may be of independent interest, for example in differential privacy applications, where it is desirable to not reveal the function value exactly to safeguard privacy. We define the notion formally below.

An (ϵ, δ, Q)-Noisy functional encryption scheme $\mathcal{F}$ for some circuit family $\mathcal{G}$ consists of four algorithms $\mathcal{F} = (\mathsf{FE.Setup}, \mathsf{FE.Keygen}, \mathsf{FE.Enc}, \mathsf{FE.Dec})$ as in the case of regular functional encryption. However, now we require the correctness condition to hold only up to some error parameter $\delta > 0$ and define security assuming that the number of queries asked is less than Q, and the challenge message evaluations on all the keys requested by the adversary differ by at most ϵ. These properties are described formally next.

Definition 3.1 (δ-Correctness)**.** Let $\delta > 0$. A functional encryption scheme $\mathcal{F}$ is δ-correct if for all $g \in \mathcal{G}_\lambda$ and all $\mathbf{x} \in \mathcal{X}_\lambda$,

$$\Pr\left[\begin{array}{l}(\mathsf{MPK}, \mathsf{MSK}) \leftarrow \mathsf{FE.Setup}(1^\lambda); \\ \mathsf{FE.Dec}\Big(\mathsf{FE.Keygen}(\mathsf{MSK}, g), \mathsf{FE.Enc}(\mathsf{MPK}, \mathbf{x})\Big) \notin [g(\mathbf{x}) - \delta, g(\mathbf{x}) + \delta]\end{array}\right] = \mathrm{negl}(\lambda)$$

where the probability is taken over the coins of $\mathsf{FE.Setup}$, $\mathsf{FE.Keygen}$, and $\mathsf{FE.Enc}$.

Security is defined in the standard indistinguishability setting and should require that for challenge messages $\mathbf{x}_0, \mathbf{x}_1 \in \mathcal{X}_\lambda$, if it holds that:

- For every queried key g_i, we have $|g_i(\mathbf{x}_0) - g_i(\mathbf{x}_1)| \leq \epsilon$.
- The total number of queries requested is less than Q, for some polynomial Q. We note that it may be that $Q >> w$.

Then, we require that the adversary cannot distinguish $\mathsf{CT}(\mathbf{x}_0)$ from $\mathsf{CT}(\mathbf{x}_1)$ with non negligible advantage. Formally, we define the $\mathsf{Noisy\text{-}IND}$ game as follows.

Definition 3.2 ((ϵ, Q) $\mathsf{Noisy\text{-}AD\text{-}IND}$ security.)**.** Let $\epsilon, \delta > 0$ such that $\epsilon/\delta < \mathrm{negl}(\lambda)$. A δ-correct functional encryption scheme $\mathcal{F}$ for a function family $\mathcal{G}$ is (ϵ, Q)-secure in the noisy indistinguishability game if for all probabilistic polynomial-time adversaries $\mathcal{A}$, the advantage of $\mathcal{A}$ in the following experiment is negligible in the security parameter λ:

1. **Public Key:** Challenger returns MPK to the adversary.
2. **Pre-Challenge Key Queries:** $\mathcal{A}$ may adaptively request keys for any functions $g_1, \ldots, g_{\ell_1} \in \mathcal{G}$ for $i \in [\ell_1]$ and $\ell_1 \leq Q$. In response, $\mathcal{A}$ is given the corresponding keys $\mathsf{SK}(g_i)$.

3. **Challenge Ciphertext:** $\mathcal{A}(1^\lambda)$ outputs the challenges $(\mathbf{x}_0, \mathbf{x}_1) \in \mathcal{X}$ to the challenger. The challenger checks that $|g_i(\mathbf{x}_0) - g_i(\mathbf{x}_1)| \leq \epsilon$ for all functions g_i requested so far. If not, it outputs $\perp$ and aborts. Otherwise, the challenger chooses a random bit b, and returns the ciphertext $\mathsf{CT}(\mathbf{x}_b)$.

4. **Post-Challenge Key Queries:** The adversary may request $\ell_2 \in [Q - \ell_1]$ keys for functions of its choice, subject to the same restrictions as above. In response, $\mathcal{A}$ is given the corresponding keys.

5. **Guess.** $\mathcal{A}$ outputs a bit b', and succeeds if $b' = b$.

The *advantage* of $\mathcal{A}$ is the absolute value of the difference between its success probability and $1/2$.

4 Online Offline FE for Quadratic forms from Ring LWE

The intuition for the construction is discussed in Section 1. Here we describe our construction formally. Our construction uses as a black box a noisy linear FE scheme, denoted by (ϵ, δ, Q)-NLinFE that computes noisy inner products modulo p_1 and supports Q queries upto correctness fudge factor δ as long as the legitimate decryption values differ upto ϵ. A construction for the same is provided in Section 5.

For the construction below supporting quadratic polynomials, we will require three prime moduli $p_0 < p_1 < p_2$ where p_0 serves as the message space for the final quadratic scheme (think of $p_0 = 2$), p_1 serves as the message space for the NLinFE scheme, and p_2 is the public key and ciphertext space for NLinFE. This scheme can be generalized to bounded degree polynomials, as we shall show in Section 6. Then, for supporting degree d polynomials, we will have a tower of moduli $p_0 < p_1 < \ldots < p_d$.

We will additionally require a "tower of discrete Gaussian distributions" where each successive discrete Gaussian is wide enough to "flood" the previous (see Lemma 2.4). We will have $\mathcal{D}_0 < \mathcal{D}_1 < \mathcal{D}_2$ for degree two polynomials.

$\mathsf{FE.Setup}(1^\lambda, 1^w)$: On input a security parameter λ and a parameter w denoting the length of message vectors, do:

1. Invoke $\mathsf{NLinFE.Setup}(1^\lambda, 1^{w+2})$ to obtain $\mathsf{NLinFE.MPK}$ and $\mathsf{NLinFE.MSK}$.
2. Sample $\mathbf{u} \leftarrow R_{p_1}^w$.
3. For $1 \leq j \leq i \leq w$, compute f_{ij} as follows:
   - Sample $\tilde{\mu}_{ij} \leftarrow \mathcal{D}_1$ for $1 \leq j \leq i \leq w$.
   - Sample $t_i \in R_{p_1}$ for $i \in [0, w]$.
   - Define
   $$f_{ij} = u_i u_j t_0 - u_j t_i - u_i t_j + p_0 \cdot \tilde{\mu}_{ij}$$

   Set $\mathbf{f} = (f_{ij})_{1 \leq j \leq i \leq w} \in R^L$ where $L = |\{i, j : 1 \leq j \leq i \leq w\}|$.
4. Output $\mathsf{MPK} = (\mathsf{NLinFE.MPK}, \mathbf{u})$, $\mathsf{MSK} = (\mathsf{NLinFE.MSK}, \mathbf{f})$.

$\mathsf{FE.Enc}(\mathsf{MPK}, \mathbf{x})$: On input public parameters MPK, and message vector $\mathbf{x} \in R_{p_0}^w$ do:

1. Sample $s_1 \xleftarrow{\mathrm{R}} R_{p_1}$ and $\boldsymbol{\mu} \leftarrow \mathcal{D}_0^w$, and compute "message carrier"
$$\mathbf{c} = \mathbf{u} \cdot s_1 + p_0 \cdot \boldsymbol{\mu} + \mathbf{x} \in R_{p_1}^w.$$
2. Let $\mathbf{b} = \mathsf{NLinFE.Enc}\,(s_1^2, c_1 s_1, \ldots, c_w s_1, 0)$.
3. Output $\mathsf{CT} = (\mathbf{c}, \mathbf{b})$

$\mathsf{FE.KeyGen}(\mathsf{MPK}, \mathsf{MSK}, \mathbf{g})$: On input the public parameters MPK, the master secret key MSK, and a function $\mathbf{g} = \sum_{1 \le j \le i \le w} g_{ij} x_i x_j$, represented as a coefficient vector $(g_{ij}) \in \mathbb{Z}_{p_0}^L$ do:

1. Compute
$$\mathbf{u_g} = \sum_{1 \le j \le i \le w} g_{ij}\,(u_i u_j, 0....0, -u_i, 0...0, -u_j, 0...0, f_{ij}) \in R_{p_1}^{w+2}.$$
2. Compute $\mathsf{SK_g} = \mathsf{NLinFE.KeyGen}(\mathbf{u_g})$ and output it.

$\mathsf{FE.Dec}(\mathsf{MPK}, \mathsf{SK_g}, \mathsf{CT_x})$: On input the public parameters MPK, a secret key $\mathsf{SK_g}$ for polynomial $\sum_{1 \le j \le i \le w} g_{ij} x_i x_j$, and a ciphertext $\mathsf{CT_x} = (\mathbf{c}, \mathbf{b})$, compute
$$\sum_{1 \le j \le i \le w} g_{ij} c_i c_j + \mathsf{NLinFE.Dec}(\mathbf{b}, \mathsf{SK_g}) \bmod p_0$$
and output it.

4.0.1 Correctness

We establish correctness of the above scheme.

Let $1 \le j \le i \le w$. By definition
$$x_i + p_0 \cdot \mu_i = c_i - u_i s_1$$
$$x_j + p_0 \cdot \mu_j = c_j - u_j s_1$$
Letting $\mu_{ij} = x_i \mu_j + x_j \mu_i + \mu_i \mu_j$, we have
$$x_i x_j + p_0 \cdot \mu_{ij} = c_i c_j - c_i u_j s_1 - c_j u_i s_1 + u_i u_j s_1^2 \tag{4.1}$$

We denote the noise added by the scheme NLinFE by $p_0 \cdot \rho_{ij}$. By correctness of the linear scheme NLinFE, we have that
$$\begin{aligned}
\mathsf{NLinFE.Dec}(\mathbf{b}, \mathsf{SK_g}) &= -c_i u_j s_1 - c_j u_i s_1 + u_i u_j s_1^2 + p_0 \cdot \rho_{ij} \\
\text{Hence, } c_i c_j + \mathsf{NLinFE.Dec}(\mathbf{b}, \mathsf{SK_g}) &= c_i c_j - c_i u_j s_1 - c_j u_i s_1 + u_i u_j s_1^2 + p_0 \cdot \rho_{ij} \\
&= x_i x_j + p_0 \cdot \mu_{ij} + p_0 \cdot \rho_{ij} \\
\sum_{1 \le j \le i \le w} g_{ij} \big(c_i c_j + \mathsf{NLinFE.Dec}(\mathbf{b}, \mathsf{SK_g})\big) &= \sum_{1 \le j \le i \le w} g_{ij}\; x_i x_j + p_0 \cdot \Big(\sum_{1 \le j \le i \le w} g_{ij}\; (\mu_{ij} + \rho_{ij}) \Big) \\
&= \sum_{1 \le j \le i \le w} g_{ij}\; x_i x_j \bmod p_0 \quad \text{as desired.}
\end{aligned}$$

4.1 Security

For ease of exposition, we will consider the special case where $\mathbf{x}_0$ and $\mathbf{x}_1$ differ only in a single coordinate, namely the w^{th} one. This restriction can be removed with a careful change of basis operation in $R^w_{p_0}$, and is specified in Appendix G. For the special case in consideration, we note the following properties about admissible key requests and some implications that will be used crucially in the proof.

- $\mathbf{x}_0[i]\mathbf{x}_0[j] = \mathbf{x}_1[i]\mathbf{x}_1[j]$ for all $i, j \neq w$ by assumption and $\mathbf{x}_0[i]\mathbf{x}_0[w] = \mathbf{x}_1[i]\mathbf{x}_1[w]$ iff it holds that $\mathbf{x}_0[i]\mathbf{x}_0[w] = \mathbf{x}_1[i]\mathbf{x}_1[w] = 0$. We will refer to these monomials as admissible monomials. Note that since $\mathbf{x}_0[i] = \mathbf{x}_1[i]$ and $\mathbf{x}_0[w] \neq \mathbf{x}_1[w]$ by assumption, $\mathbf{x}_0[i]\mathbf{x}_0[w] = \mathbf{x}_1[i]\mathbf{x}_1[w] = 0$ can only occur when $\mathbf{x}_0[i] = \mathbf{x}_1[i] = 0$.
- It follows that an admissible quadratic polynomial query may only contain a monomial $x_i x_w$ for some i, if it holds that $\mathbf{x}_0[i]\mathbf{x}_0[w] = \mathbf{x}_1[i]\mathbf{x}_1[w] = 0$.
- Hence it suffices to ensure that the simulated keys for $x_i x_j$, where $i, j \neq w$ decrypt correctly and the simulated key for $x_i x_w$ decrypts correctly as long as $\mathbf{x}_0[i]\mathbf{x}_0[w] = \mathbf{x}_1[i]\mathbf{x}_1[w] = 0$. In particular note that the simulator is not required to provide a correctly working key for $x_i x_w$ when $\mathbf{x}_0[i]\mathbf{x}_0[w] \neq \mathbf{x}_1[i]\mathbf{x}_1[w]$.
- In the proof below, the keys and challenge ciphertext will be programmed so that decryption provides the honest answer for all admissible monomials. The simulator will be designed to construct keys even for non-admissible monomials but these will not decrypt correctly, since the simulator cannot know the value $\mathbf{x}_b[w]$ and hence cannot know the value of $\mathbf{x}_b[i]\mathbf{x}_b[w]$ unless $\mathbf{x}_b[i] = 0$. However, the adversary will never notice this, since he may not request these keys.

Theorem 4.1. *The construction in Section 4 achieves adaptive indistinguishability (*AD-IND*) based security.*

Proof. We will prove the theorem via a sequence of hybrids, where the first hybrid is the real world with challenge $\mathbf{x}_0$ and the last hybrid is the real world with challenge $\mathbf{x}_1$.

The Hybrids. Our Hybrids are described below.

Hybrid 0. This is the real world with message $\mathbf{x}_0$. In hybrid 0, f_{ij} is picked as follows:

1. Sample $t_0, \ldots, t_w \leftarrow R_{p_1}$.
2. Set $f_{ij} = u_i u_j t_0 - u_j t_i - u_i t_j + p_0 \cdot \tilde{\mu}_{ij} \mod p_1$. Here, $\tilde{\mu}_{ij}$ is noise chosen to drown μ_{ij}.

Hybrid 1. In this hybrid, the only thing that is different is that the challenger picks f_{ij} to depend on the challenge ciphertext. Specifically,

1. Sample $t_0, \ldots, t_w \leftarrow R_{p_1}$.
2. Set

$$f_{ij} = \left(x_i x_j - c_i c_j\right) - \left(u_i u_j t_0 - u_j t_i - u_i t_j - p_0 \cdot \tilde{\mu}_{ij}\right) \mod p_1$$

Hybrid 2. In this hybrid, we change the input for $\mathsf{NLinFE.Enc}$ to $(t_0, t_1, \ldots, t_w, 1)$ where t_i are chosen uniformly in R_{p_1} for $i \in \{0, \ldots, w\}$.

Hybrid 3. In this hybrid, we change the message vector in $\mathbf{c}$ to $\mathbf{x}_1$.

Hybrids 4 and 5. In Hybrid 4 we change the input to $\mathsf{NLinFE.Enc}$ to $(s_1^2, c_1 s_1, \ldots, c_w s_1, 0)$ as in Hybrid 1. In Hybrid 5, we change f_{ij} to be chosen independent of the ciphertext as in Hybrid 0. This is the real world with message $\mathbf{x}_1$.

Indistinguishability of Hybrids. Below we establish that consecutive hybrids are indistinguishable.

Claim 4.2. *Hybrid 0 is statistically indistinguishable from Hybrid 1.*

Proof. The only difference between Hybrid 0 and Hybrid 1 is in the way f_{ij} is sampled.
In hybrid 0, we have:

1. Sample $t_0, \ldots, t_w \leftarrow R_{p_1}$.
2. Set $f_{ij} = u_i u_j t_0 - u_j t_i - u_i t_j + p_0 \cdot \tilde{\mu}_{ij}$.

In Hybrid 1, f_{ij} is picked as follows:

1. Sample $t_0, \ldots, t_w \leftarrow R_{p_1}$.
2. Set $f_{ij} = \big((c_i - \hat{c}_i)(c_j - \hat{c}_j) - c_i c_j\big) - \big(u_i u_j t_0 - u_j t_i - u_i t_j + p_0 \cdot \tilde{\mu}_{ij}\big) \mod p_1$.

In Hybrid 1, we have:

$$
\begin{aligned}
f_{ij} &= \big(x_i x_j - c_i c_j\big) - \big(u_i u_j t_0 - u_j t_i - u_i t_j - p_0 \cdot \tilde{\mu}_{ij}\big) \\
&= \big(u_i u_j s_1^2 - u_i c_j s_1 - u_j c_i s_1 - p_0 \cdot \mu_{ij}\big) - \big(u_i u_j t_0 - u_j t_i - u_i t_j - p_0 \cdot \tilde{\mu}_{ij}\big) \qquad \text{by equation 2.2} \\
&= u_i u_j (s_1^2 - t_0) - u_i (c_j s_1 - t_j) - u_j (c_i s_1 - t_i) - p_0 \cdot \mu_{ij} + p_0 \cdot \tilde{\mu}_{ij} \\
&= u_i u_j t'_0 - u_i t'_j - u_j t'_i - p_0 \cdot \mu_{ij} + p_0 \cdot \tilde{\mu}_{ij} \\
&\approx u_i u_j t'_0 - u_i t'_j - u_j t'_i + p_0 \cdot \tilde{\mu}_{ij} \quad \text{as long as } \tilde{\mu}_{ij} \text{ floods } \mu_{ij}.
\end{aligned}
$$

Here, we define $t'_0 = s_1^2 - t_0$ and $t'_i = c_i s_1 - t_i$ for $i \in [w]$. Note that $t'_0, \ldots, t'_w$ are i.i.d due to addition of i.i.d $t_0, \ldots, t_w$. Thus, we get that the distributions of f_{ij} in Hybrid 0 and Hybrid 1 are statistically indistinguishable. Since the only thing that changed between Hybrid 0 and Hybrid 1 is the manner of sampling f_{ij}, we get that Hybrid 0 and Hybrid 1 are statistically indistinguishable. □

Claim 4.3. *Assuming adaptive IND security of the noisy linear FE scheme* NLinFE*, Hybrid 1 and Hybrid 2 are indistinguishable.*

Proof. Given an adversary $\mathcal{B}$ who distinguishes between Hybrid 0 and 1, we construct an adversary $\mathcal{A}$ who breaks the adaptive IND security of the noisy linear FE scheme NLinFE.

$\mathcal{A}$ runs as follows:

1. The challenger of the NLinFE scheme provides $\mathsf{NLinFE.MPK}$ to $\mathcal{A}$. $\mathcal{A}$ picks $\mathbf{u} \in R_{p_1}^w$, and sends $\mathsf{MPK} = (\mathsf{NLinFE.MPK}, \mathbf{u})$ to $\mathcal{B}$.

2. $\mathcal{A}$ picks $s_1 \leftarrow R_{p_1}$, and $\mathbf{t} \leftarrow R_{p_1}^{w+1}$.

3. $\mathcal{A}$ computes
$$c_i = u_i s_1 + p_0 \cdot \mu_i + \mathbf{x}_0[i] \quad \forall i \in [w].$$

4. When $\mathcal{B}$ requests a key for monomial $x_i x_j$, $\mathcal{A}$ does the following:
   - $\mathcal{A}$ computes $\mathbf{t}'$ as:
   $$t'_0 = s_1^2 - t_0, \quad \{t'_i = c_i s_1 - t_i\}_{i \in [w]}$$
   - $\mathcal{A}$ computes
   $$f_{ij} = u_i u_j t'_0 - u_j t'_i - u_i t'_j + p_0 \cdot \tilde{\mu}_{ij}$$
   - $\mathcal{A}$ parses this as a linear key with coefficients
   $$(u_i u_j, 0....0, u_i, 0...0, u_j, 0...0, f_{ij}) \in R_{p_1}^{w+2}.$$
   Here, u_i appears at the j^{th} position, u_j appears at the i^{th} position.
   - $\mathcal{A}$ forwards this key request to the linear FE challenger and returns its response $\mathsf{NLinFE.SK}_{ij}$ to $\mathcal{B}$.

5. $\mathcal{B}$ outputs challenges $\mathbf{x}_0, \mathbf{x}_1 \in R_{p_0}^w$. $\mathcal{A}$ chooses challenge messages $\mathbf{z}_0, \mathbf{z}_1$ as:
$$\begin{aligned} \mathbf{z}_0 &= \left(s_1^2, c_1 s_1, \ldots, c_w s_1,\ 0\right) && \in R_{p_1}^{w+2} \\ \mathbf{z}_1 &= (t_0,\ t_1,\quad \ldots \quad, t_w,\ 1) && \in R_{p_1}^{w+2} \end{aligned}$$
   $\mathcal{A}$ returns $(\mathbf{z}_0, \mathbf{z}_1)$ to the Linear FE challenger.

6. When the linear FE challenger sends challenge $\mathsf{NLinFE.CT}(\mathbf{z}_b)$ to $\mathcal{A}$, it sends $\mathsf{CT} = \left(\mathsf{NLinFE.CT}(\mathbf{z}_b), \mathbf{c}\right)$ to $\mathcal{B}$. Recall that it computed $\mathbf{c}$ itself in Step 3.

7. $\mathcal{B}$ may request more keys which are answered as before.

8. When $\mathcal{B}$ outputs a guess bit b, $\mathcal{A}$ forwards this guess to the linear FE challenger.

Now, we argue that for all the requested keys, the difference between the decryption values on the challenge messages $\mathbf{z}_0$ and $\mathbf{z}_1$ is very small. By setting the parameter ϵ of the (ϵ, δ, Q)-NLinFE scheme to be an upper bound on this difference, we obtain by the security of the noisy linear FE scheme that $\mathsf{NLinFE.CT}(\mathbf{z}_0)$ and $\mathsf{NLinFE.CT}(\mathbf{z}_1)$ are indistinguishable. We proceed to show this formally. We will let $Q = w^2$. Consider a key request for monomial $x_i x_j$.

In Hybrid 1, the function for the monomial $x_i x_j$ evaluated on $\mathbf{z}_0$ yields:

$$u_i u_j s_1^2 - u_i c_j s_1 - u_j c_i s_1$$

In Hybrid 2 the function for the monomial $x_i x_j$ evaluated on $\mathbf{z}_1$ yields:

$$\begin{aligned}
&u_i u_j t_0 - u_i t_j - u_j t_i + f_{ij} \\
&= u_i u_j t_0 - u_i t_j - u_j t_i + \big(u_i u_j (s_1^2 - t_0) - u_j (c_i s_1 - t_i) - u_i (c_i s_1 - t_j) + p_0 \cdot (\tilde{\mu}_{ij} - \mu_{ij})\big) \\
&\approx u_i u_j s_1^2 - u_i c_j s_1 - u_j c_i s_1 + p_0 \cdot \tilde{\mu}_{ij} \quad \text{since } \tilde{\mu}_{ij} \text{ floods } \mu_{ij}.
\end{aligned}$$

Thus, the decryption values in both worlds are approximately the same upto an additive factor of $p_0 \cdot \tilde{\mu}_{ij}$. We set ϵ in the NLinFE scheme to be larger than $p_0 \cdot \tilde{\mu}_{ij}$, we obtain by the guarantee of Noisy Linear FE that Hybrids 1 and 2 are indistinguishable[3]. □

Claim 4.4. *Assume Regev public key encryption is semantically ecure. Then, Hybrid 2 is indistinguishable from Hybrid 3.*

Proof. We recall Regev public key encryption. We set $(\mathsf{PK}, \mathsf{SK}) = (\mathbf{u}, s_1)$ as the public and secret key. The ciphertext for message $\mathbf{x}$ is computed $\mathbf{c} = \mathbf{u} \cdot s_1 + p_0 \cdot \boldsymbol{\mu} + \mathbf{x}$, where $\boldsymbol{\mu}$ is suitably chosen noise.

Given an adversary $\mathcal{B}$ who distinguishes between Hybrid 2 and Hybrid 3, we build an adversary $\mathcal{A}$ who breaks the semantic security of Regev public key encryption. The adversary $\mathcal{A}$ receives $\mathsf{PK} = \mathbf{u}$ upon which, it simulates the view of $\mathcal{B}$ as follows:

- Run $\mathsf{NLinFE.Setup}$ to obtain $\mathsf{NLinFE.MPK}$ and $\mathsf{NLinFE.MSK}$. Return $\mathsf{MPK} = (\mathsf{NLinFE.MPK}, \mathbf{u})$ to $\mathcal{B}$.
- When $\mathcal{B}$ requests a key, construct it honestly as in Hybrid 0.
- When $\mathcal{B}$ outputs challenges $\mathbf{x}_0, \mathbf{x}_1$, $\mathcal{A}$ forwards these to the PKE challenger.
- $\mathcal{A}$ receives $\mathbf{c}$ where $\mathbf{c} = \mathbf{u} \cdot s_1 + p_0 \cdot \boldsymbol{\mu} + \mathbf{x}_b$ for a random bit b.
- $\mathcal{A}$ computes $\mathbf{b} = \mathsf{NLinFE.CT}(t_0, \ldots, t_w, 1)$ and returns $(\mathbf{c}, \mathbf{b})$ to $\mathcal{B}$.
- $\mathcal{B}$ may request more keys which are handled as before. Finally, when $\mathcal{B}$ outputs a guess bit b, $\mathcal{A}$ outputs the same.

Clearly, if $b = 0$, then $\mathcal{B}$ sees the distribution of Hybrid 2, whereas if $b = 1$, it sees the distribution of Hybrid 3. Also, the advantage of the attacker $\mathcal{B}$ in distinguishing Hybrids 2 and 3 translates directly to the advantage of the attacker $\mathcal{A}$ in breaking the semantic security of Regev public key encryption. Hence the claim follows. □

Hybrid 4 is analogous to Hybrid 1 and indistinguishability can be argued along the same lines as that between Hybrids 1 and 2, while Hybrid 5 is analogous to Hybrid 0 but with message $\mathbf{x}_1$. This is the real world with message $\mathbf{x}_1$. Indistinguishability between Hybrids 4 and 5 can be argued along the same lines as that between Hybrids 0 and 1. □

[3]Recall that in the security game of noisy linear FE, the challenge messages must only evaluate to approximately same values for all functions that are queried. The difference between the decryption values is allowed to differ up to ϵ which is a parameter to the scheme.

Discussion. Note that the above construction enjoys a useful online-offline property – only $O(w)$ elements of the CT depend on the message while the remaining $O(w^2)$ elements are only encryptions of noise. Thus, the encryptions of noise may be generated offline, which speeds up online encryption.

It also enjoys an intriguing *locality* property – if 1 element out of w changes, then our solution requires only changing two ciphertext elements.

5 Noisy Functional Encryption for Linear Functions

In this section, we construct a noisy functional encryption scheme for linear functions which supports Q queries for a fixed polynomial Q. Security posits that an adversary cannot distinguish between encryptions of $\mathbf{z}_0$ and $\mathbf{z}_1$ as long as $|\mathbf{g}_i(\mathbf{z}_0) - \mathbf{g}_i(\mathbf{z}_1)| \leq \epsilon$ for every key $\mathbf{g}_i$ requested, as long as the number of requested keys is less than Q. We emphasize that Q can be greater than the dimension of the message/key vectors, namely w – indeed this is the case we will be interested in.

To support $Q > w$ queries so as to achieve the security definition stated above, we artificially add noise to decryption, so that any two messages whose decryption under a key is equal upto ϵ noise, will decrypt to only approximately correct values but will have indistinguishable ciphertexts.

For ease of notation, our description below assumes a stateful keygen. We get rid of this restriction using standard tricks as described in Appendix C.

5.1 Construction.

The algorithms for the (ϵ, δ, Q) noisy linear FE scheme, denoted by NLinFE are defined as follows.

$\mathsf{FE.Setup}(1^\lambda, 1^Q, 1^w, p_1)$: On input a security parameter λ, a parameter w denoting the length of the function and message vectors, a parameter Q denoting the number of queries supported and a modulus p_1 denoting the space of the message and function vectors, set $(\mathsf{MPK}, \mathsf{MSK}) = \mathsf{LinFE.Setup}(1^{w+1+Q})$.

$\mathsf{FE.KeyGen}(\mathsf{MSK}, \mathbf{g}, i)$: On input MSK, a function vector $\mathbf{g} \in \mathbb{Z}_{p_1}^w$ and an index $i \in [Q]$ denoting query number, do:

1. Sample $\gamma_i \leftarrow \mathcal{D}_1$, where $\mathcal{D}_1$ is a discrete Gaussian of width large enough to flood ϵ.
2. Output $\mathsf{SK}_{\mathbf{g}} = \mathsf{LinFE.KeyGen}(\mathbf{g} \| \gamma_i \| \mathbf{e}_i)$ where $\mathbf{e}_i \in \mathbb{Z}^Q$ is the i^{th} unit vector. Note that the LinFE key explicitly contains the vector $(\mathbf{g} \| \gamma_i \| \mathbf{e}_i)$.

$\mathsf{FE.Enc}(\mathsf{MPK}, \mathbf{z})$: On input public key MPK, a message vector $\mathbf{z} \in \mathbb{Z}_{p_1}^w$, do:

1. Sample $\boldsymbol{\delta} \leftarrow \mathcal{D}_2^Q$ and $\mu \leftarrow \mathcal{D}$ subject to the following constraints:
   - $\mathcal{D}_2$ is a discrete Gaussian of width large enough to flood $\mathcal{D}_1$. Thus, it will hold that $\delta_i + \gamma_i \approx \delta_i$, where γ_i is chosen by keygen.
   - $\mathcal{D}$ has width large enough so that μ is distributed indistinguishably from $\mu + 1$.
2. Let $\mathsf{CT}_{\mathbf{z}} = \mathsf{LinFE.Enc}(\mathbf{z} \| \mu \| \boldsymbol{\delta})$

$\mathsf{FE.Dec}(\mathsf{SK}_{\mathbf{g}_i}, \mathsf{CT}_{\mathbf{z}})$: On input a secret key $\mathsf{SK}_{\mathbf{g}_i}$ for function $\mathbf{g}_i$, and a ciphertext $\mathsf{CT}_{\mathbf{z}}$:

1. Compute $\mathsf{LinFE.Dec}(\mathsf{CT}_{\mathbf{z}}, \mathsf{SK}_{\mathbf{g}_i})$ to recover $\mathbf{g}_i^\intercal \mathbf{z} + \mu \cdot \gamma_i + \delta_i$.

Appropximate correctness is evident since we recovered the correct value upto noise $\mu \cdot \gamma_i + \delta_i$.

Note. We remark that the keygen algorithm can be made stateless by using standard tricks, as in [GVW12, Section 6]. We refer the reader to Appendix C for more details.

5.2 Security.

Next, we argue that the above scheme is secure. We have that for every key query $\mathbf{g}_i$, $i \in [Q]$, it holds that $\mathbf{g}_i^\intercal(\mathbf{z}_0 - \mathbf{z}_1) = \epsilon_i$. We argue via a sequence of hybrids.

Hybrid 0. This is the real world with message $\mathbf{z}_0$. The challenge CT encrypts $\mathbf{y}_0 = (\mathbf{z}_0 \| \mu \| \boldsymbol{\delta})$.

Hybrid 1. In this world, we generate the challenge CT for the message $\mathbf{y}_0 = (\mathbf{z}_0 \| \mu \| \hat{\boldsymbol{\delta}} + \boldsymbol{\delta})$ where $\boldsymbol{\delta}$ floods $\hat{\boldsymbol{\delta}}$. For the keys, we set $\boldsymbol{\gamma} = \hat{\boldsymbol{\delta}}$.

Hybrid 2. In this world, the noise in the Q keys changes to $\boldsymbol{\gamma} = \hat{\boldsymbol{\delta}} + \boldsymbol{\epsilon}$.

Hybrid 3. In this world, we change the message in the challenge CT to $\mathbf{y}_1 = (\mathbf{z}_1 \| \mu + 1 \| \boldsymbol{\delta})$.

Hybrid 4. In this world, we rewrite the message in the challenge CT as $\mathbf{y}_1 = (\mathbf{z}_1 \| \mu \| \boldsymbol{\delta})$. This is the real world with message $\mathbf{z}_1$.

Indistinguishability of Hybrids. It is evident that Hybrids 0 and 1 are statistically indistinguishable, since $\boldsymbol{\delta}$ floods $\hat{\boldsymbol{\delta}}$. By the same argument, Hybrid 1 and 2 are statistically indistinguishable since $\hat{\boldsymbol{\delta}}$ floods $\boldsymbol{\epsilon}$ and Hybrid 3 and Hybrid 4 are statistically indistinguishable since μ is distributed indistinguishably from $\mu + 1$. The chief transition that must be argued is that between Hybrids 2 and 3, which we argue now.

Claim 5.1. *If the exact linear scheme is* AD-IND *secure, then Hybrids 2 and 3 are indistinguishable.*

Proof. Given an adversary $\mathcal{A}$ who can distinguish between Hybrids 2 and 3, we will construct an adversary $\mathcal{B}$ who will break the security of the exact linear scheme. $\mathcal{B}$ plays the AD-IND game with the LinFE challenger, denoted by $\mathcal{C}$.

1. The LinFE challenger $\mathcal{C}$ outputs the public key MPK, which $\mathcal{B}$ forwards to $\mathcal{A}$.
2. $\mathcal{A}$ requests a key $\mathbf{g}_i$ for $i \in [\ell_1]$, where $\ell_1 \leq Q$. $\mathcal{B}$ chooses γ_i as in the real world and requests a key for $\hat{\mathbf{g}}_i = (\mathbf{g} \| \gamma_i \| \mathbf{e}_i)$ to the LinFE challenger $\mathcal{C}$. The challenger returns $\mathsf{LinFE.SK}(\hat{\mathbf{g}}_i)$ which $\mathcal{B}$ gives $\mathcal{A}$.
3. $\mathcal{A}$ outputs two challenges $\mathbf{z}_0, \mathbf{z}_1 \in \mathbb{Z}_{p_1}^w$. $\mathcal{B}$ checks that $\mathbf{g}_i^\intercal(\mathbf{z}_0 - \mathbf{z}_1) = \epsilon_i \leq \epsilon$ for all queries $\mathbf{g}_i$ requested so far. If this condition does not hold, output $\perp$ and abort.

4. $\mathcal{B}$ chooses $\hat{\delta}_i = \gamma_i - \epsilon_i$ for $i \in [\ell_1]$. The remaining $\hat{\delta}_{\ell_1+1} \ldots \hat{\delta}_Q$ it chooses as in the real world. Next, it constructs $\hat{\mathbf{y}}_0 = (\mathbf{z}_0 \| \mu \| \hat{\boldsymbol{\delta}})$ and $\hat{\mathbf{y}}_1 = (\mathbf{z}_1 \| \mu + 1 \| \mathbf{0})$ and returns these to the LinFE challenger $\mathcal{C}$ as the challenge messages.

5. $\mathcal{A}$ may request more keys $\mathbf{g}_i$ so that $\mathbf{g}_i^\intercal(\mathbf{z}_0 - \mathbf{z}_1) = \epsilon_i$. $\mathcal{B}$ chooses $\gamma_i = \hat{\delta}_i + \epsilon_i$ and requests a key for $\hat{\mathbf{g}}_i = (\mathbf{g} \| \gamma_i \| \mathbf{e}_i)$ to the LinFE challenger. The challenger returns $\mathsf{LinFE.SK}(\hat{\mathbf{g}}_i)$, which $\mathcal{B}$ gives $\mathcal{A}$.

6. $\mathcal{C}$ outputs the challenge ciphertext $\mathsf{CT}(\hat{\mathbf{y}}_b)$. $\mathcal{B}$ adds noise $\boldsymbol{\delta}$ to the last Q coordinates and returns this to $\mathcal{A}$ as the challenge CT. Thus, $\mathcal{A}$ sees an encryption of either $\mathbf{y}_0 = (\mathbf{z}_0 \| \mu \| \hat{\boldsymbol{\delta}} + \boldsymbol{\delta})$ or $\mathbf{y}_1 = (\mathbf{z}_1 \| \mu + 1 \| \boldsymbol{\delta})$.

7. When $\mathcal{A}$ outputs a guess for bit b, $\mathcal{B}$ outputs the same.

Observe that the query $\hat{\mathbf{g}}_i$ is an admissible query for the LinFE challenger because:

$$\begin{aligned}
\hat{\mathbf{g}}_i^\intercal \hat{\mathbf{y}}_0 &= \mathbf{g}_i^\intercal \mathbf{z}_0 + \mu \cdot \gamma_i + \hat{\delta}_i \\
\hat{\mathbf{g}}_i^\intercal \hat{\mathbf{y}}_1 &= \mathbf{g}_i^\intercal \mathbf{z}_1 + \mu \cdot \gamma_i + \gamma_i \\
&= \mathbf{g}_i^\intercal \mathbf{z}_0 - \epsilon_i + \mu \cdot \gamma_i + \hat{\delta}_i + \epsilon_i \\
&= \mathbf{g}_i^\intercal \mathbf{z}_0 + \mu \cdot \gamma_i + \hat{\delta}_i \\
&= \hat{\mathbf{g}}_i^\intercal \hat{\mathbf{y}}_0
\end{aligned}$$

If the LinFE challenger $\mathcal{C}$ returns an encryption of $\hat{\mathbf{y}}_0$, then $\mathcal{A}$ sees an encryption of $\mathbf{y}_0 = (\mathbf{z}_0 \| \mu \| \hat{\boldsymbol{\delta}} + \boldsymbol{\delta})$, otherwise it sees an encryption of $\mathbf{y}_1 = (\mathbf{z}_1 \| \mu + 1 \| \boldsymbol{\delta})$. In the former case we obtain the distribution of Hybrid 2, in the latter case of Hybrid 3. □

Hence, we argued that for Q (for $Q > w$) queries, our noisy linear FE scheme satisfies (ϵ, Q) $\mathsf{Noisy\text{-}AD\text{-}IND}$ security as per Definiton 3.2.

5.3 Perspective: Random Self Reducibility

Recently, Bitansky and Vaikuntanathan [BV16] showed that indistinguishability obfuscation and functional encryption enjoy "random self reducibility". We will only discuss one aspect of their results here. In the context of functional encryption, one of the things they show, is (informally) that if an FE scheme is secure for inputs that are sampled from some distribution χ, then there is a transformation that converts it to an FE scheme that is secure for arbitrarily chosen inputs. This transformation however assumes that the underlying FE scheme supports the evaluation of PRFs (in particular, the function must be able to evaluate the decryption function of a symmetric key encryption scheme, see Figure 2 in Section 4 of [BV16]).

We remark that one implication of our results is to provide an analogous statement for much weaker classes of functions, namely bounded degree polynomials. To see this in the context of quadratic polynomials, note that the message dependent component of our ciphertext is succinct, namely $O(w)$ for messages of size w, and to support w^2 queries, the encryptor must provide w^2 encryptions of noise terms.

We note that these w^2 noise terms need not be i.i.d for the security argument to go through, they can be correlated. This is because the purpose of the noise terms added by noisy FE are

used to flood the difference terms $\epsilon_1, \ldots, \epsilon_Q$ between hybrids 1 and 2 in Section 4. Since the terms $\epsilon_1, \ldots, \epsilon_Q$ are correlated, it suffices to choose correlated noise terms to flood them as well (as shown in Appendix F).

Thus, if the $Q = w^2$ noise encryptions in our quadratic FE scheme could be replaced with only $O(w)$ noise encryptions $\hat{\delta}_1, \ldots, \hat{\delta}_w$ and the w^2 encryptions of $\hat{\delta}_i \hat{\delta}_j$ could be computed on the fly under a public LWE label, then these terms could now be added as the noise in noisy FE and the remainder of the argument would proceed as before. Combined with the already succinct message dependent ciphertexts, the succinct representation of noise would yield an overall succinct scheme. Thus, using our construction, the problem of succinctly computing quadratic functions on arbitrary messages $\mathbf{x}$ reduces to the problem of succinctly computing quadratic functions on noise terms. We refer the reader to Appendix F for more details.

6 Shallow to Deep FE via Induction

In this section, we will describe how our quadratic FE scheme can be bootstrapped to support circuits of arbitrary polynomial depth, secure against bounded collusions.

This transformation proceeds in two steps as follows.

1. First, we convert our bounded query FE for quadratic forms into a bounded query FE for bounded degree polynomials via induction. This is the most technical part of the overall transformation, since we cannot resort to linearization to preserve the online-offline property of our construction.

2. Next, we use the bootstrapping theorem of [GVW12] which converts a q query FE scheme for bounded degree polynomials into a q query scheme for all polynomial circuits. The bootstrapping relies on the result of Applebaum et al. [AIK06] which states that every polynomial time computable function f can be represented by constant degree to obtain a bounded query FE scheme for all polynomial sized circuits.

We note that in addition to the encryptions of the noise terms which can be generated offline, encryptions of the randomness needed for the randomized encoding may also be generated offline. Thus, even for all polynomial sized circuits, the final bounded collusion FE scheme only requires $O(w)$ online encryptions – there is no dependence on Q for the online encryption stage.

6.1 FE for Bounded Degree Polynomials via Induction

Notation. To begin, we set up some notation that will be useful throughout this section. Let $\mathcal{C}^k$ denote the set of advice provided to compute the ciphertexts of level k and a box, e.g. $\boxed{\mathrm{x}}$ denote the Regev encryption of x under a publicly computable label. Additionally c^k denotes a ciphertext at level k, i.e. an encryption of level k monomial. $\mathsf{N}\,\boxed{\cdot}$ denotes the noise used in generating the encryption $\boxed{\cdot}$ and $\mathsf{L}\,\boxed{\cdot}$ denotes the LWE label for the same.

Intuition. Before we begin, we provide some intuition. Recall that the quadratic scheme described in Section 4 enables us to create an LWE ciphertext, under a publicly computable label, for a quadratic function of the plaintext ($x_i x_j$ for example). Note that the $O(w^2)$ ciphertexts, encrypting

terms $x_i x_j$ for $i \in [w]$, can be computed using only $O(w)$ message dependent ciphertexts and $O(w^2)$ encryptions of noise terms.

The main observation of this section is that there is no need to stop at level 2 – the computation can be propagated down a circuit efficiently so that at every new level, the message dependent terms suffer only a constant factor blowup over those in the previous level. Hence, for any constant depth d, the number of message dependent ciphertext components are $O(2^d \cdot w)$ as against $O(w^d)$), and all the remaining encryptions are of noise.

Induction. The above intuition can be formalized using an inductive argument. The intuition for the induction is not difficult to see even if the details are hairy. In what follows we focus only on the message dependent components of the ciphertext and ignore encryptions of noise, as these need only be added at the last level of the circuit. Recall that the level 1 ciphertexts $c_i = \boxed{x_i}$ for $i \in [w]$ along with advice $\boxed{c_i \cdot s}$ were sufficient to compute the message dependent ciphertext components at level 2. Hence, at any level k, given c^{k-1} and $\boxed{c^{k-1} \cdot s}$, we would be in a position to compute c^k using the quadratic scheme of Section 4. Thus we have that to compute ciphertext at level k, it suffices to provide advice that is linear in the size of level $k-1$ ciphertexts c^{k-1}.

This intuition is complicated by the fact that we are not given c^{k-1} directly, but rather *advice*, a set of ciphertexts $\mathcal{C}^{k-1}$ which enables the decryptor to compute c^{k-1} on the fly. The question then, is whether there exists advice linear in the size of the set $\mathcal{C}^{k-1}$ which suffices to compute the advice $\boxed{c^{k-1} \cdot s}$ required to compute c^k on the fly. We answer this affirmatively below. Throughout this section, we assume circular security of LWE. This is for notational convenience as well as efficiency. This assumption can be removed by choosing new randomness s_i for each level i as in leveled fully homomorphic encryption.

Theorem 6.1. *Define the message set at level k, denoted by $\mathcal{M}^k$ recursively to include:*

1. *[Rule 1.]* $\mathcal{C}^{k-1}$
2. *[Rule 2.]* $\mathcal{M}^{k-1} \cdot s$
3. *[Rule 3.]* $\mathsf{N}\boxed{\mathcal{M}^{k-1}}$
4. *[Rule 4.]* $\mathsf{N}\boxed{\mathcal{M}^{k-1}} \cdot s$

Let $\mathcal{C}^k = \boxed{\mathcal{M}^k}$. Then, given $\bigcup_{i \in [k]} \mathcal{C}^i$, the following can be computed.

1. $\boxed{c^{k-1}}$ *and* $\mathsf{N}\boxed{c^{k-1}}$.
2. $\boxed{c^{k-1} \cdot s}$ *and* $\mathsf{N}\boxed{c^{k-1} \cdot s}$.

Note that $\boxed{c^{k-1} \cdot s}$ and c^{k-1} suffice to compute c^k by Section 4. $\boxed{c^{k-1}}$ will be needed to compute level $k+1$ ciphertexts and the noise terms will ease the analysis. Thus, the advice size at level k is larger than the advice at level $k-1$ by a factor of 4. That is,

$$|\mathcal{C}^k| \leq 4 \cdot |\mathcal{C}^{k-1}|$$

Proof. Proof by induction. Since for bootstrapping it suffices to build (online-offline) FE for degree 3 polynomials, we analyze the first 3 levels below for concreteness. The generalization to arbitrary (bounded) levels is provided in Appendix D.

Computation for first 3 levels. We will show that the above rules suffice to compute ciphertexts of level 3 monomials. In what follows, we will not enlist the messages encrypted at any level exhaustively, but only enlist the subset that is required for a given step.

At level 1, encryptions are provided for:

$$\mathcal{M}^1 \subseteq \{x_i,\ s\ \}, \qquad \text{and } \mathcal{C}^1 \subseteq \Big\{ \underbrace{\boxed{x_i}}_{c_i},\ \boxed{s} \Big\}$$

At level 2, encryptions are provided for:

$$\mathcal{M}^2 \subseteq \{c_i,\ x_i \cdot s,\ s^2,\ \ \mathsf{N}\,\boxed{x_i}\ ,\ \mathsf{N}\,\boxed{s}\ ,\ \mathsf{N}\,\boxed{x_i} \cdot s\ ,\ \ \mathsf{N}\,\boxed{s} \cdot s\ \}$$

We claim that:

Claim 6.2. *$\mathcal{C}^2$ defined as above suffices to compute $\boxed{c_i}$ as well as $\boxed{c_i \cdot s}$.*

Proof. Since $c_i \in \mathcal{C}^1 \subset \mathcal{M}^2$, $\boxed{c_i} \in \mathcal{C}^2$. Now, let us consider $\boxed{c_i \cdot s}$. We have by additive homomorphism that:

$$\begin{aligned} \boxed{c_i \cdot s} &= \boxed{(u_i \cdot s + \mu_i + x_i)\ \cdot s} \\ &= u_i \boxed{s^2} + \boxed{\mu_i \cdot s} + \boxed{x_i \cdot s} \\ &= u_i \boxed{s^2} + \boxed{\mathsf{N}\,\boxed{x_i} \cdot s} + \boxed{x_i \cdot s} \end{aligned}$$

The boxed terms are provided as part of $\mathcal{C}^2$, hence we are done. □

Next, we consider level 3. We claim that:

Claim 6.3. *$\mathcal{C}^3$ suffices to compute $\boxed{c_{ij}^2 \cdot s}$ as well as $\boxed{c_{ij}^2}$.*

Proof. We examine each term in turn below.

Computing $\boxed{c_{ij}^2 \cdot s}$. We have that,

$$\boxed{c_{ij}^2 \cdot s} = \boxed{(c_i c_j) \cdot s} + \boxed{\Big(- u_i \boxed{c_j \cdot s} - u_i \boxed{c_i \cdot s} + u_i u_j \boxed{s^2} \Big) \cdot s}$$

Thus, to compute $\boxed{c_{ij}^2 \cdot s}$, we are required to compute $\boxed{c_i c_j s}$, $\boxed{\boxed{c_i \cdot s} \cdot s}$ and $\boxed{\boxed{s^2} \cdot s}$.

We claim that it is sufficient to provide advice to compute $\boxed{c_i c_j s}$, since this advice subsumes the remaining terms. To see this, note that $c_i c_j s$ can be seen as a quadratic term in c_i and $c_j \cdot s$, and to use the quadratic scheme to compute an encryption of the product, we are required to compute $\boxed{c_i}$, $\boxed{c_j \cdot s}$ as well as the advice $\boxed{\boxed{c_i} \cdot s}$ and $\boxed{\boxed{c_i \cdot s} \cdot s}$. We examine each of the terms below.

1. Consider $\boxed{c_i}$, $\boxed{c_j \cdot s}$: These terms can be computed using the advice $\mathcal{C}^2$ as established by claim 6.2.

2. $\boxed{\boxed{c_i} \cdot s}$: This term can be expanded as $\boxed{\left(\mathsf{L}\boxed{c_i} \cdot s + \mathsf{N}\boxed{c_i} + c_i\right) \cdot s}$ which may be computed by linear homomorphism given advice $\boxed{s^2}$ [4], $\boxed{\mathsf{N}\boxed{c_i} \cdot s}$ and $\boxed{c_i \cdot s}$. By rule 4, $\mathcal{M}^3$ contains $\mathsf{N}\boxed{c_i} \cdot s$ and by we know that $\mathcal{M}^2$, it contains s^2. The encryption of $c_i \cdot s$ can be computed by the induction hypothesis. Thus, the decryptor can compute $\boxed{\boxed{c_i} \cdot s}$.

3. $\boxed{\boxed{c_i \cdot s} \cdot s}$: This term can be expanded as in the previous case to yield the requirement of the following advice terms:

   (a) $\boxed{c_i \cdot s^2}$: This in turn can be computed homomorphically from $\boxed{x_i \cdot s^2}$ and $\boxed{\mathsf{N}\boxed{x_i} \cdot s^2}$. These terms are provided in $\mathcal{C}^3$ by rule 2. In more detail, since $\mathcal{M}^2$ contains $x_i \cdot s$ and $\mathsf{N}\boxed{x_i} \cdot s$, rule 2 implies that $\mathcal{M}^3$ will contain $x_i \cdot s^2$ and $\mathsf{N}\boxed{x_i} \cdot s^2$, which implies that $\mathcal{C}^3$ will contain $\boxed{x_i \cdot s^2}$ and $\boxed{\mathsf{N}\boxed{x_i} \cdot s^2}$ as desired.

   (b) $\boxed{\mathsf{N}\boxed{c_i \cdot s} \cdot s}$: Note that $\mathsf{N}\boxed{c_i \cdot s}$ can be computed given $\mathsf{N}\boxed{s^2}$, $\mathsf{N}\boxed{x_i \cdot s}$ and $\mathsf{N}\boxed{\mu_i \cdot s}$ where the last term can be written as $\mathsf{N}\boxed{\mathsf{N}\boxed{x_i} \cdot s}$. Thus, we need encryptions of:

   $$\mathsf{N}\boxed{s^2} \cdot s, \quad \mathsf{N}\boxed{x_i \cdot s} \cdot s, \quad \mathsf{N}\boxed{\mathsf{N}\boxed{x_i} \cdot s} \cdot s$$

   Note that s^2, $x_i \cdot s$, $\mathsf{N}\boxed{x_i} \cdot s$ are in $\mathcal{M}^2$, hence $\mathcal{C}^2$ contains

   $$\left\{ \boxed{s^2}, \ \boxed{x_i \cdot s}, \ \boxed{\mathsf{N}\boxed{x_i} \cdot s} \right\}$$

   Hence, by rules 3 and 4,

   $$\left\{ \mathsf{N}\boxed{s^2}, \ \mathsf{N}\boxed{x_i \cdot s}, \ \mathsf{N}\boxed{\mathsf{N}\boxed{x_i} \cdot s}, \ \mathsf{N}\boxed{s^2} \cdot s, \ \mathsf{N}\boxed{x_i \cdot s} \cdot s, \ \mathsf{N}\boxed{\mathsf{N}\boxed{x_i} \cdot s} \cdot s \right\} \subseteq \mathcal{M}^3$$

   which implies that $\mathcal{C}^3$ contains the desired encryptions.

Computing $\boxed{c_{ij}^2}$. Recall that $c_{ij}^2 = c_i c_j + \left(- u_i \boxed{c_j \cdot s} - u_i \boxed{c_i \cdot s} + u_i u_j \boxed{s^2} \right)$. Thus, by additive homomorphism, we require:

4. (a) $\boxed{c_i c_j}$: This requires $\boxed{c_i}$ as well as $\boxed{\boxed{c_i} \cdot s}$ which are accounted for in steps 1 and 2 above.

   (b) $\boxed{\boxed{c_j \cdot s}}$: This requires $\boxed{c_j \cdot s}$ and $\boxed{\mathsf{N}\boxed{c_j \cdot s}}$. The former is accounted for by I.H., and for the latter, note that $\mathsf{N}\boxed{c_j \cdot s}$ can be computed using additive homomorphism of terms in $\mathcal{M}^3$ as discussed in Step 3b above. Hence, $\boxed{\mathsf{N}\boxed{c_j \cdot s}} \in \mathcal{C}^3$ as desired.

[4] Note that the label $\mathsf{L}\boxed{c_i}$ is public and may be multiplied by the decryptor to $\boxed{s^2}$

□

The generalization of the proof to arbitrary bounded depth polynomials requires us to establish the induction step. These details are deferred to Appendix D.

□

Thus, we have established Theorem 6.1 for depth 3 circuits. In more detail, we have shown that if the encryptor provides the $4^3 \cdot |\mathbf{x}| = O(w)$ encryptions for the terms specified by the above rules, then the decryptor can compute level 3 ciphertexts c_i^3 on the fly. Recall that c_i^3 encrypts all monomials at level 3, such as $x_1x_2x_3x_4$ under a publicly computable label. Thus, the key generator knows the LWE label under which the function value is encrypted and can provide a key for the same. For concreteness, if we denote this label by L_f, then the decryptor recovers terms:

$$c_f = L_f \cdot s + \mu_f + f(\mathbf{x}) \quad \mathbf{c}_0 = \mathbf{w} \cdot s + \boldsymbol{\mu}_0 \quad \text{for some } \mathbf{w} \text{ in the public key}$$

Now, the key is a short preimage $\mathbf{e}$ such that $\langle \mathbf{w}, \mathbf{e} \rangle = L_f$ and the decryptor can recover $f(\mathbf{x})$.

Description of PolyFE. In more detail, consider a scheme PolyFE, in which the encryptor computes encryptions of all the terms in $\mathcal{M}^1 \cup \mathcal{M}^2 \cup \mathcal{M}^3$. Now, as established by claim 6.3, the decryptor can compute $\boxed{c_{ij}^2 \cdot s}$ by computing an equation QuEq which has a single quadratic term of public elements $\boxed{c_i}$ and $\boxed{c_j \cdot s}$ and remaining linear terms in variables from $\mathcal{M}^1 \cup \mathcal{M}^2 \cup \mathcal{M}^3$. Say that the linear equation LinEq in variables $m_1, \ldots, m_L \in \{\mathcal{M}^1 \cup \mathcal{M}^2 \cup \mathcal{M}^3\}$ has coefficients $a_1, \ldots, a_L$ for $L = |\mathcal{M}^1 \cup \mathcal{M}^2 \cup \mathcal{M}^3|$. The key generator now provides a noisy linear key for the linear function $a_1, \ldots, a_L$ so that the decryptor may compute LinEq on the values provided by the encryptor.

Since $\boxed{c_i}$ and $\boxed{c_j \cdot s}$ are provided as part of $\mathcal{C}^1 \cup \mathcal{C}^2 \cup \mathcal{C}^3$, the decryptor can compute equation QuEq as in the construction of Section 4.

Security can be argued exactly as in Section 4, by observing that even for level 3 (or more generally for level d) the key generator must provide a noisy linear FE key, just as for level 2, albeit for a larger dimensional message/function space. Hence the argument from Section 4 generalizes naturally – we defer the details to the full version.

Splitting Encryption as Online-Offline. As seen above, if the encryptor provides $O(w)$ encryptions for the terms in $\bigcup_{k \in [3]} \mathcal{M}^k$, then the decryptor can compute level 3 ciphertexts c_i^3 on the fly. This is the online part of the encryption algorithm.

As discussed, it is not secure to release keys to decrypt the above ciphertexts as-is. To provide a secure decryption key, the encryptor must additionally encrypt Q' noise terms to support q queries, where $q' = O(q^2)$ suffices. This is the offline part of the encryption algorithm.

Thus, we have constructed an online-offline FE scheme PolyFE for bounded degree polynomials. The security notion we achieve is inherited from the underlying Linear FE scheme, which, by [ALS] is adaptive indistinguishability. For the case of bounded degree polynomials, this is equivalent to non-adaptive simulation security by [O'N10].

7 Putting it together : Online-Offline Bounded FE for all circuits

In this section, we describe how to put together all the pieces from the previous sections to build a bounded collusion FE scheme for all circuits, denoted by BddFE. The underlying scheme PolyFE is assumed to be non-adaptive simulation secure, i.e. $\mathsf{NA\text{-}SIM}$, as discussed above. This yields a final security notion of $\mathsf{NA\text{-}SIM}$, i.e. $(\mathrm{poly}, \mathrm{poly}, 0)$-$\mathsf{SIM}$ which means the adversary may make an unbounded number of pre-challenge queries, ask for an unbounded number of challenge ciphertexts but may not make any post-challenge key queries. While the construction of [GVW12] supports the stronger $\mathsf{AD\text{-}SIM}$ security in addition to $\mathsf{NA\text{-}SIM}$, this notion can only be achieved for a scheme that supports a *single* ciphertext. We consider the model of a single ciphertext and bounded number of keys overly restrictive, and do not attempt to achieve this notion. This is also the approach taken by [GKP+13].

7.1 Bootstrapping Functional Encryption

Gorbunov et al. [GVW12] show that q query FE for NC_1 can be bootstrapped to q query FE for all circuits. At a high level, their approach can be summarized as follows. Let $\mathcal{C}$ be a family of polynomial sized circuits. Let $C \in \mathcal{C}$ and let $\mathbf{x}$ be some input. Let $\tilde{C}(\mathbf{x}, R)$ be a randomized encoding of C that is computable by a constant depth circuit with respect to inputs x and R. Then consider a new family of circuits $\mathcal{G}$ defined by:

$$G_{C,\Delta}(\mathbf{x}, R_1, \ldots, R_S) = \tilde{C}\Big(\mathbf{x}; \bigoplus_{a \in \Delta} R_a\Big)$$

As observed by [GVW12, Section 6], $G_{C,\Delta}(\cdot,\cdot)$ 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 $C(\mathbf{x})$.

As observed by [ALS], q query FE that supports evaluation for constant degree polynomials is a much weaker object than q query FE for all of NC_1. Note that NC_1 is a powerful class that is, in particular, capable of evaluating a pseudorandom function (PRF). Indeed, the suggested approach by [GVW12] for bootstrapping *unbounded* query FE for NC_1 to unbounded query FE for all circuits does require evaluating a PRF. However for the case of bounded queries, we only need FE to support the evaluation of constant degree polynomials. In particular, our construction from Section 6.1 suffices for bootstrapping, to yield q-query online-offline FE for all circuits.

The construction follows the blueprint described in [ALS], but we recap it here for completeness. As in [GVW12, ALS], let (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, the key generator identifies 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 as shown by [GVW12]. Let $L \stackrel{\text{def}}{=} (\ell^3 + S \cdot m)$.

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

$\mathsf{BddFE.KeyGen}(\mathsf{msk}, \mathsf{C})$): Upon input the master secret key and a circuit C, do:

1. Choose 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. Set $\mathsf{BddFE.SK}_C = \{\mathsf{SK}_i = \mathsf{PolyFE.KeyGen}(\mathsf{PolyFE.msk}, \mathsf{P_i})\}_{\mathsf{i}\in[\mathsf{k}]}$ and output it.

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

1. Choose $R_1, \ldots, R_S \leftarrow \{0,1\}^m$, where m is the size of the random input in the randomized encoding.
2. Set $\mathsf{CT}_{\mathbf{x}} = \mathsf{PolyFE.Enc}(\mathsf{PolyFE.mpk}, \mathbf{x}, \mathsf{R_1}, \ldots, \mathsf{R_s})$ and output it.

$\mathsf{BddFE.Dec}(\mathsf{mpk}, \mathsf{CT}_{\mathbf{x}}, \mathsf{SK_C})$: Upon input a ciphertext $\mathsf{CT}_{\mathbf{x}}$ for vector $\mathbf{x}$, and a secret key SK_C for circuit C, do the following:

1. Compute $G_{C,\Delta}(\mathbf{x}, R_1, \ldots, R_S) = \mathsf{PolyFE.Dec}(\mathsf{CT}_{\mathbf{x}}, \mathsf{SK}_C)$.
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 immediately from the correctness of PolyFE and the correctness of randomized encodings. We provide an argument for security and the description of the online-offline phase in Appendix E.

Acknowledgements. We thank Damien Stehlé and Chris Peikert for helpful discussions.

References

[ABB10] Shweta Agrawal, Dan Boneh, and Xavier Boyen. Efficient lattice (H)IBE in the standard model. In *EUROCRYPT*, pages 553–572, 2010.

[ABCP15] Michel Abdalla, Florian Bourse, Angelo De Caro, and David Pointcheval. Simple functional encryption schemes for inner products. Cryptology ePrint Archive, Report 2015/017, 2015. http://eprint.iacr.org/ To appear in PKC'15.

[AFV11] Shweta Agrawal, David Mandell Freeman, and Vinod Vaikuntanathan. Functional encryption for inner product predicates from learning with errors. In *Asiacrypt*, 2011.

[AIK06] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Computationally private randomizing polynomials and their applications. *Computational Complexity*, 15(2):115–162, 2006.

[AIK11] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. How to garble arithmetic circuits. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*, pages 120–129, 2011.

[AJ15] Prabhanjan Ananth and Abhishek Jain. Indistinguishability obfuscation from compact functional encryption. In *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part I*, pages 308–326, 2015.

[AJS] Prabhanjan Ananth, Abhishek Jain, and Amit Sahai. Indistinguishability obfuscation from functional encryption for simple functions. Eprint 2015/730.

[ALS] Shweta Agrawal, Benoit Libert, and Damien Stehle. Fully secure functional encryption for linear functions from standard assumptions, and applications. https://eprint.iacr.org/2015/608.

[BDOZ11] Rikke Bendlin, Ivan Damgård, Claudio Orlandi, and Sarah Zakarias. Semi-homomorphic encryption and multiparty computation. In *EUROCRYPT*, 2011.

[Bea95] Donald Beaver. Precomputing oblivious transfer. In Don Coppersmith, editor, *Advances in Cryptology CRYPT0 95*, volume 963 of *Lecture Notes in Computer Science*, pages 97–109. Springer Berlin Heidelberg, 1995.

[BF01] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. In *CRYPTO*, pages 213–229, 2001.

[BSW07] John Bethencourt, Amit Sahai, and Brent Waters. Ciphertext-policy attribute-based encryption. In *IEEE Symposium on Security and Privacy*, pages 321–334, 2007.

[BV11a] Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*, pages 97–106, 2011.

[BV11b] Zvika Brakerski and Vinod Vaikuntanathan. Fully homomorphic encryption from ring-lwe and security for key dependent messages. In *Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings*, pages 505–524, 2011.

[BV15] Nir Bitansky and Vinod Vaikuntanathan. Indistinguishability obfuscation from functional encryption. *IACR Cryptology ePrint Archive*, 2015:163, 2015.

[BV16] Nir Bitansky and Vinod Vaikuntanathan. Indistinguishability obfuscation: From approximate to exact. In *Theory of Cryptography - 13th International Conference, TCC 2016-A, Tel Aviv, Israel, January 10-13, 2016, Proceedings, Part I*, pages 67–95, 2016.

[BW06] Xavier Boyen and Brent Waters. Anonymous hierarchical identity-based encryption (without random oracles). In *CRYPTO*, pages 290–307, 2006.

[BW07] Dan Boneh and Brent Waters. Conjunctive, subset, and range queries on encrypted data. In *TCC*, pages 535–554, 2007.

[CFL+] Jung Hee Cheon, Pierre-Alain Fouque, Changmin Lee, Brice Minaud, and Hansol Ryu. Cryptanalysis of the new clt multilinear map over the integers. Eprint 2016/135.

[CGH+15] Jean-Sébastien Coron, Craig Gentry, Shai Halevi, Tancrede Lepoint, Hemanta K Maji, Eric Miles, Mariana Raykova, Amit Sahai, and Mehdi Tibouchi. Zeroizing without low-level zeroes: New mmap attacks and their limitations. In *Advances in Cryptology–CRYPTO 2015*, pages 247–266. Springer, 2015.

[CHH+07] Ronald Cramer, Goichiro Hanaoka, Dennis Hofheinz, Hideki Imai, Eike Kiltz, Rafael Pass, Abhi Shelat, and Vinod Vaikuntanathan. Bounded cca2-secure encryption. In *Proceedings of the Advances in Crypotology 13th International Conference on Theory and Application of Cryptology and Information Security*, ASIACRYPT'07, 2007.

[CHKP10] David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. Bonsai trees, or how to delegate a lattice basis. In *EUROCRYPT*, pages 523–552, 2010.

[CHL+15] 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.

[CJL] Jung Hee Cheon, Jinhyuck Jeong, and Changmin Lee. An algorithm for ntru problems and cryptanalysis of the ggh multilinear map without a low level encoding of zero. Eprint 2016/139.

[CKV10] Kai-Min Chung, Yael Tauman Kalai, and Salil P. Vadhan. Improved delegation of computation using fully homomorphic encryption. In *Advances in Cryptology - CRYPTO 2010, 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings*, pages 483–501, 2010.

[CLP13] Ran Canetti, Huijia Lin, and Omer Paneth. Public-coin concurrent zero-knowledge in the global hash model. In *Proceedings of the 10th Theory of Cryptography Conference on Theory of Cryptography*, TCC'13, 2013.

[CLT13] Jean-Sébastien Coron, Tancrède Lepoint, and Mehdi Tibouchi. Practical multilinear maps over the integers. In *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I*, pages 476–493, 2013.

[Coc01] Clifford Cocks. An identity based encryption scheme based on quadratic residues. In *IMA Int. Conf.*, pages 360–363, 2001.

[DKXY02] Yevgeniy Dodis, Jonathan Katz, Shouhuai Xu, and Moti Yung. Key-insulated public key cryptosystems. In *Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques: Advances in Cryptology*, EUROCRYPT '02, 2002.

[DPSZ12] Ivan Damgrd, Valerio Pastro, Nigel Smart, and Sarah Zakarias. Multiparty computation from somewhat homomorphic encryption. In *Advances in Cryptology CRYPTO 2012*, 2012.

[EGM89] Shimon Even, Oded Goldreich, and Silvio Micali. On-line/off-line digital signatures. In *CRYPTO*, 1989.

[GGH13a] Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In *EUROCRYPT*, 2013.

[GGH+13b] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *FOCS*, 2013. http://eprint.iacr.org/.

[GGH+13c] Sanjam Garg, Craig Gentry, Shai Halevi, Amit Sahai, and Brent Waters. Attribute-based encryption for circuits from multilinear maps. In *CRYPTO*, 2013.

[GGH15] Craig Gentry, Sergey Gorbunov, and Shai Halevi. Graph-induced multilinear maps from lattices. In *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part II*, pages 498–527, 2015.

[GGHZ14] Sanjam Garg, Craig Gentry, Shai Halevi, and Mark Zhandry. Fully secure functional encryption without obfuscation. In *IACR Cryptology ePrint Archive*, volume 2014, page 666, 2014.

[GKP+13] Shafi Goldwasser, Yael Tauman Kalai, Raluca A. Popa, Vinod Vaikuntanathan, and Nickolai Zeldovich. Reusable garbled circuits and succinct functional encryption. In *STOC*, pages 555–564, 2013.

[GKPV10] Shafi Goldwasser, Yael Tauman Kalai, Chris Peikert, and Vinod Vaikuntanathan. Robustness of the learning with errors assumption. In *ITCS*, 2010.

[GLW12] Shafi Goldwasser, Allison Lewko, and David Wilson. Bounded-collusion ibe from key homomorphism. In *Theory of Cryptography*, Lecture Notes in Computer Science, 2012.

[GMC08] Fuchun Guo, Yi Mu, and Zhide Chen. Identity-based online/offline encryption. In *Financial Cryptography and Data Security*, 2008.

[GPSW06] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *ACM Conference on Computer and Communications Security*, pages 89–98, 2006.

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*, pages 197–206, 2008.

[GVW12] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Functional encryption with bounded collusions from multiparty computation. In *CRYPTO*, 2012.

[GVW13] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Attribute based encryption for circuits. In *STOC*, 2013.

[GVW15] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Predicate encryption for circuits from lwe. Cryptology ePrint Archive, Report 2015/029, 2015. http://eprint.iacr.org/.

[HJ15] Y. Hu and H. Jia. Cryptanalysis of GGH map. Cryptology ePrint Archive: Report 2015/301, 2015.

[HW14] Susan Hohenberger and Brent Waters. Online/offline attribute-based encryption. In Hugo Krawczyk, editor, *Public-Key Cryptography PKC 2014*, volume 8383 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2014.

[IK02] Yuval Ishai and Eyal Kushilevitz. Perfect constant-round secure computation via perfect randomizing polynomials. In *Automata, Languages and Programming, 29th International Colloquium, ICALP 2002, Malaga, Spain, July 8-13, 2002, Proceedings*, pages 244–256, 2002.

[IKM+13] Yuval Ishai, Eyal Kushilevitz, Sigurd Meldgaard, Claudio Orlandi, and Anat Paskin-Cherniavsky. On the power of correlated randomness in secure computation. In *Theory of Cryptography*, 2013.

[IPS08] Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. Founding cryptography on oblivious transfer efficiently. In *Advances in Cryptology CRYPTO 2008*, pages 572–591, 2008.

[KSW08] Jonathan Katz, Amit Sahai, and Brent Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In *EUROCRYPT*, pages 146–162, 2008.

[LOS+10] Allison B. Lewko, Tatsuaki Okamoto, Amit Sahai, Katsuyuki Takashima, and Brent Waters. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In *EUROCRYPT*, pages 62–91, 2010.

[LPR10] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In *EUROCRYPT*, volume 6110, 2010.

[LR14] Yehuda Lindell and Ben Riva. Cut-and-choose based two-party computation in the online/offline and batch settings. In *Crypto*, 2014.

[LZ09] JosephK. Liu and Jianying Zhou. An efficient identity-based online/offline encryption scheme. In Michel Abdalla, David Pointcheval, Pierre-Alain Fouque, and Damien Vergnaud, editors, *Applied Cryptography and Network Security*, volume 5536 of *Lecture Notes in Computer Science*, pages 156–167. Springer Berlin Heidelberg, 2009.

[MR07] Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on gaussian measures. *SIAM Journal on Computing (SICOMP)*, 37(1):267–302, 2007. extended abstract in FOCS 2004.

[MSZ] Eric Miles, Amit Sahai, and Mark Zhandry. Annihilation attacks for multilinear maps: Cryptanalysis of indistinguishability obfuscation over ggh13. Eprint 2016/147.

[O'N10] Adam O'Neill. Definitional issues in functional encryption. Cryptology ePrint Archive, Report 2010/556, 2010. http://eprint.iacr.org/.

[Reg09] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *J.ACM*, 56(6), 2009. extended abstract in STOC'05.

[SS10] Amit Sahai and Hakan Seyalioglu. Worry-free encryption: Functional encryption with public keys. In *Proceedings of the 17th ACM Conference on Computer and Communications Security*, CCS '10, 2010.

[ST01] Adi Shamir and Yael Tauman. Improved online/offline signature schemes. In Joe Kilian, editor, *Advances in Cryptology CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, 2001.

[SW] Amit Sahai and Brent Waters. Functional encryption:beyond public key cryptography. Power Point Presentation, 2008. http://userweb.cs.utexas.edu/bwaters/presentations/files/functional.ppt.

[SW05] Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In *EUROCRYPT*, pages 457–473, 2005.

[Wat12] Brent Waters. Functional encryption for regular languages. In *Crypto*, 2012.

[Yao82] Andrew Chi-Chih Yao. Protocols for secure computations (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November 1982*, pages 160–164, 1982.

A Supplementary material and Appendices

A.1 The GVW Construction

In this section we sketch the GVW scheme and discuss why it appears unsuitable for Online-Offline optimization. The scheme can be summarized as follows.

- The first ingredient they need is a single key FE scheme for all circuits. A construction for this was provided by Sahai and Seyalioglu in [SS10].
- Next, the single FE scheme is generalized to a q query scheme for NC_1 circuits. This gerenalization is fairly complex, we provide an outline here. At a high level, they run N copies of the single key scheme, where $N = O(q^4)$. The encryptor encrypts the views of the BGW MPC protocol for N parties, computing some functionality related to C. They rely on the fact that BGW is non-interactive when used to compute bounded degree functions. To generate a secret key, KeyGen chooses a random subset of the single query FE keys, where the parameters are set so that the subsets have small pairwise intersections. This subset of keys enables the decryptor to recover sufficiently many shares of $C(x)$ which allows him to recover $C(x)$. [GVW12] argue that an attacker with q keys only learns a share x_i when two subsets of keys intersect, but since the subsets were chosen to have small pairwise intersections, this does not occur often enough to recover enough shares of x. Finally, by the security of secret sharing, x remains hidden.
- As the last step they "bootstrap" the q query FE for NC_1 to q query FE for all circuits using computational randomized encodings [AIK06]. They must additionally use cover free sets to ensure that fresh randomness is used for each randomized encoding.

Thus, to encrypt a message x, the encryptor must secret share it into $N = O(q^4)$ shares, and encrypt each one with the one query FE. Since they use Shamir secret sharing with polynomial of degree t and $t = O(q^2)$, note that at most $O(q^2)$ shares can be generated offline, since $t + 1$ points will determine the polynomial. Hence $O(q^4)$ shares must be generated in the online phase. This results in an online encryption time that degrades as $O(q^4)$.

B Preliminaries

B.1 Simulation Based Definition of Security against Bounded Collusions

In this section, we define simulation based security for bounded collusions, as in [GVW12, Defn 3.1].

Definition B.1 (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:

$\mathsf{Exp}^{\mathsf{real}}_{\mathcal{F},A}(1^\lambda)$**:**	$\mathsf{Exp}^{\mathsf{ideal}}_{\mathcal{F},\mathrm{Sim}}(1^\lambda)$**:**
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 $\mathcal{V} \stackrel{\text{def}}{=} (C_i, C_i(x), \mathsf{SK}_i)_{i\in[q]}$
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:

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

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.

C Making KeyGen stateless for Noisy Linear FE

We note that the keygen algorithm can be made stateless by using standard tricks, as in [GVW12, Section 6]. To see this, let us examine how noise is added in the above system at present. The encryptor provides encryptions of Q noise terms $\boldsymbol{\delta}$ and for $i \in [Q]$, the key generator appends the i^{th} key vector $\mathbf{g}_i$ with a Q sized unit vector $\mathbf{e}_i$. As a result, decryption of the i^{th} key with the ciphertext results in an additional term $\boldsymbol{\delta}^\intercal \mathbf{e}_i = \delta_i$, which gets added to the legitimate decryption value $\mathbf{g}_i^{\mathbf{z}}$. Thus, to ensure that a fresh noise term δ_i is added for each decryption equation, the key generator must keep track of how many keys it has issued in the past (or receive this information as input).

Cover free sets offer a natural tool to deal with this issue, and this technique was used towards a similar end in [GVW12]. The idea is to choose $Q' > Q$ and have the encryptor provide encryptions of Q' noise values in place of Q. The key generator is modified to be stateless and randomly pick a Q' sized binary vector of weight v in place of $\mathbf{e}_i$. Now the decryption computes a random subset sum of Q' noise terms for every key, which for appropriate setting of Q' and v guarantees that each decryption equation has at least one fresh noise term. It is shown in [GVW12] that $Q' = Q^2$ and $v = \lambda$ suffices.

The careful reader might notice that the summands in this case are discrete Gaussians and a random subset of discrete Gaussians does not yield the original distribution. However, fortunately this is not an issue, since the cover free property implies that every decryption equation has at least one freshly sampled discrete Gaussian that is not used in any other equation. This suffices for security.

We also note that though the general construction of linear FE for inner products modulo a prime by [ALS] is stateful, as noted by the authors of [ALS], this is only for the most general adversary and does not apply to our setting. For our setting, the function queries to the underlying Linear FE are linearly independent allowing for a stateless keygen, due to the fact that each query vector contains at least one unique index which is nonzero in that vector alone, and zero for all other query vectors. This is ensured by the use of cover free sets in the construction. We refer to [ALS] for more details, as this issue is discussed at length there.

D Details of Induction

Induction Step. Assume that the claim is true for level $k-1$. Then we establish that it is true for level k. Namely, we show that using the induction hypothesis and the rules, we can compute $\boxed{c_i^{k-1}}$ and $\boxed{c_j^{k-1} \cdot s}$. The latter, along with c^{k-1} (which is provided by the I.H) suffice to compute ciphertexts at level k, namely c^k.

Computing $\boxed{c_{ij}^{k-1} \cdot s}$. We have that

$$\boxed{c_{ij}^{k-1} \cdot s} = \boxed{\left(c_i^{k-2}\, c_j^{k-2} - u_i^{k-2} \boxed{c_j^{k-2} \cdot s} - u_j^{k-2} \boxed{c_j i^{k-2} \cdot s} + u_i^{k-2} u_j^{k-2} \boxed{s^2}\right) \cdot s}$$

Consider $\boxed{c_i^{k-2}\, c_j^{k-2} \cdot s}$. Computing this using the quadratic scheme requires:

1. $\boxed{c_i^{k-2}}$ and $\boxed{c_j^{k-2} \cdot s}$. These can be computed by the induction hypothesis.

2. $\boxed{c_i^{k-2} \cdot s} = \boxed{(\mathsf{L}\boxed{c_i^{k-2}} \cdot s + \mathsf{N}\boxed{c_i^{k-2}} + c_i^{k-2}) \cdot s}$, hence we require $\boxed{c_i^{k-2} \cdot s}$, $\boxed{s^2}$ and $\boxed{\mathsf{N}\boxed{c_i^{k-2}} \cdot s}$. The first two terms are provided by the induction hypothesis.

   For $\boxed{\mathsf{N}\boxed{c_i^{k-2}} \cdot s}$, notice that $\mathcal{C}_i^{k-2} \in \mathcal{M}^{k-1}$ which implies that $\mathsf{N}\boxed{\mathcal{C}_i^{k-2}} \in \mathcal{C}^{k-1}$. Hence by rule 4, we have that $\mathsf{N}\boxed{\mathcal{C}_i^{k-2}} \cdot s \in \mathcal{M}^k$. Hence $\mathcal{C}^k$ contains $\boxed{\mathsf{N}\boxed{\mathcal{C}_i^{k-2}} \cdot s}$. This suffices to compute $\boxed{\mathsf{N}\boxed{c_i^{k-2}} \cdot s}$ by additive linear homomorphism, as desired.

3. $\boxed{\boxed{c_i^{k-2} \cdot s} \cdot s}$: To compute this we need $\boxed{c_i^{k-2} \cdot s^2}$ and $\boxed{\mathsf{N}\boxed{c_i^{k-2} \cdot s} \cdot s}$.

   (a) Consider $\boxed{c_i^{k-2} \cdot s^2}$. We have that $\mathcal{C}^{k-2} \cdot s \in \mathcal{M}^{k-1}$ by rule 4. Hence $\mathcal{C}^{k-2} \cdot s^2 \in \mathcal{M}^k$ by Rule 2, which implies that $\boxed{\mathcal{C}^{k-2} \cdot s^2} \in \mathcal{C}^k$. This suffices to compute $\boxed{c_i^{k-2} \cdot s^2}$

   (b) Consider $\boxed{\mathsf{N}\boxed{c_i^{k-2} \cdot s} \cdot s}$: We have that $\mathcal{C}^{k-2} \cdot s \in \mathcal{M}^{k-1}$ by Rule 4, which implies that $\mathsf{N}\boxed{\mathcal{C}^{k-2} \cdot s} \in \mathcal{C}^{k-1}$. Thus, again by Rule 4 we get that $\mathsf{N}\boxed{\mathcal{C}^{k-2} \cdot s} \cdot s \in \mathcal{M}^k$. This implies that $\boxed{\mathsf{N}\boxed{\mathcal{C}^{k-2} \cdot s} \cdot s} \in \mathcal{C}^k$.

Claim D.1 (Noise Lemma)**.** *we can compute* $\mathsf{N}\boxed{c_i^k}$ *and* $\mathsf{N}\boxed{c_j^k \cdot s}$ *using only linear homomorphism on the terms* $\left\{ \mathsf{N}\boxed{\mathcal{C}^k}, \mathsf{N}\boxed{\mathcal{M}^k}, \mathsf{N}\boxed{\mathcal{C}^k \cdot s}, \mathsf{N}\boxed{\mathcal{M}^k \cdot s} \right\}$.

Proof. We show that $\mathsf{N}\boxed{c_i^k}$ is a linear combination of terms $\mathsf{N}\boxed{\mathcal{M}^k}$. Note that:

$$\begin{aligned} \boxed{c_i^k} &= \boxed{c_j^{k-1}c_\ell^{k-1} - b_{j\ell}^k} \qquad \text{Note, here } b \text{ lives in larger space.} \\ &= \boxed{c_j^{k-1}c_\ell^{k-1} - \left(u_j^{k-1}\boxed{c_\ell^{k-1} \cdot s} + u_\ell^{k-1}\boxed{c_j^{k-1} \cdot s} - u_j^k u_\ell^k \boxed{s^2}\right)} \end{aligned}$$

Hence, $\mathsf{N}\boxed{c_i^k}$ can be computed using linear combinations of:

$$\begin{aligned} \mathsf{N}\boxed{c_i^k} &\in \left\{ \mathsf{N}\boxed{c_j^{k-1}c_\ell^{k-1}}, \mathsf{N}\boxed{\boxed{c_\ell^{k-1} \cdot s}} \right\} \\ \mathsf{N}\boxed{c_j^{k-1}c_\ell^{k-1}} &\in \left\{ \mathsf{N}\boxed{\boxed{c_j^{k-1}} \cdot s} \right\} \\ &\in \left\{ \mathsf{N}\boxed{\mathsf{N}\boxed{c_j^{k-1}} \cdot s}, \mathsf{N}\boxed{c_j^{k-1} \cdot s} \right\} \end{aligned}$$

$\mathsf{N}\boxed{c_i^k \cdot s}$ can be computed using linear combinations of:

$$\mathsf{N}\boxed{c_i^k \cdot s} \in \left\{\mathsf{N}\boxed{c_j^{k-1}c_\ell^{k-1} \cdot s},\ \mathsf{N}\boxed{\boxed{c_\ell^{k-1} \cdot s} \cdot s}\right\}$$

$$\mathsf{N}\boxed{c_j^{k-1}c_\ell^{k-1}} \in \left\{\mathsf{N}\boxed{\boxed{c_j^{k-1}} \cdot s}\right\}$$

$$\in \left\{\mathsf{N}\boxed{\mathsf{N}\boxed{c_j^{k-1}} \cdot s},\ \mathsf{N}\boxed{c_j^{k-1} \cdot s}\right\}$$

Now, by I.H we have that $\mathsf{N}\boxed{c_i^{k-1}}$ is a linear combination of terms $\mathsf{N}\boxed{\mathcal{C}^{k-1}}$ and $\mathsf{N}\boxed{\mathcal{C}^{k-1} \cdot s}$.

Now consider $\mathsf{N}\boxed{\boxed{c_\ell^{k-1} \cdot s}}$. First note that $\boxed{\boxed{c_\ell^{k-1} \cdot s}}$ may be computed using $\mathsf{N}\boxed{\boxed{c_\ell^{k-1} \cdot s}}$ and $\boxed{c_\ell^{k-1} \cdot s}$ by additive homomorphism. Hence, $\mathsf{N}\boxed{\mathcal{M}^k}$ suffices to compute $\mathsf{N}\boxed{c_i^k}$.

□

E Security of the Construction in Section 7

In this section, we argue that the construction described in Section 7 is secure. Let $\mathsf{RE.Sim}$ be the simulator guaranteed by the security of randomized encodings and $\mathsf{PolyFE.Sim}$ be the simulator guaranteed by the security of the PolyFE scheme. Again, the blueprint is the same as in [ALS], only the underlying scheme is different. We proceed to describe the simulator below.

$\mathsf{Bdd.Sim}\big(\{C_i, C_i(\mathbf{x}), \mathsf{SK}_i\}_{i\in[q^*]}\big)$: The simulator $\mathsf{Bdd.Sim}$ 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 $q^* \leq q$, and must simulate the ciphertext $\mathsf{CT}_\mathbf{x}$. It proceeds as follows:

1. Pick $\Delta_1, \ldots, \Delta_q \subseteq S$ randomly, of size v each.
2. For each $i \in [q^*]$, invoke $\mathsf{RE.Sim}(C_i(\mathbf{x}))$ to learn $G_{C,\Delta_i}(\mathbf{x})$.
3. Let $\mathsf{CT}_\mathbf{x} = \mathsf{PolyFE.Sim}\big(\{G_{C_i,\Delta_i}, G_{C_i,\Delta_i}(\mathbf{x}, R_1, \ldots, R_S), \mathsf{SK}_i\}_{i\in[q^*]}\big)$ and output it.

The correctness of $\mathsf{Bdd.Sim}$ follows from the correctness of $\mathsf{RE.Sim}$ and $\mathsf{PolyFE.Sim}$.

Remark. Note that the most general version of the FE scheme for linear functions mod p presented in [ALS] is stateful, but as observed in [ALS], it can be made stateless for many applications, including the one we consider here. We refer the reader to [ALS] for more details.

Splitting Encryption as Online-Offline. If the encryptor provides $O(w)$ encryptions for the terms in $\bigcup_{k\in[3]} \mathcal{M}^k$, then the decryptor can compute level 3 ciphertexts c_i^3 on the fly. Now, to support q queries on circuits $C \in \mathcal{C}$ of depth d using bootstrapping, the encryptor must additionally provide encryptions of $R_1, \ldots, R_S$ as seen above, where $S = O(q^2)$ and each R_i is of size m which is $O(|C|)$. However, note that these encryptions can also be generated offline, since they do not depend on the message.

Thus, the online complexity of the encryption algorithm remains $O(w)$ while the offline complexity is $O(|C| \cdot q^2)$.

F Correlated Noise suffices in Quadratic FE

For concreteness, consider $Q = w^2$ and $\epsilon_k = \mu_i \mu_j$ for $i, j \in [w]$, $k \in [Q]$ and μ_i chosen from a discrete Gaussian distribution. We show that we may sample $(\hat{\delta}_1, \ldots, \hat{\delta}_w)$ so that $\hat{\delta}_i$ drowns μ_i and generate w^2 terms using these. Thus, instead of sampling $(\hat{\delta}_i, \ldots, \hat{\delta}_Q)$ i.i.d to drown $(\epsilon_1, \ldots, \epsilon_Q)$ and setting $\boldsymbol{\gamma} = \hat{\boldsymbol{\delta}}$, we will sample $\hat{\boldsymbol{\delta}} = (\delta_i)_{i \in [w]}$ using only w i.i.d noise terms $\hat{\delta}_i$ and set $\boldsymbol{\gamma} = (\hat{\delta}_1 \hat{\delta}_1, \ldots \hat{\delta}_i \hat{\delta}_j, \ldots, \hat{\delta}_w \hat{\delta}_w)$.

To justify this choice of $\hat{\boldsymbol{\delta}}$, we show that if

$$\begin{aligned} \Delta(\hat{\delta}_i,\ \hat{\delta}_i + \mu_i) \leq \text{negl} \quad \forall i \in [w] \\ \text{Then,}\ \ \Delta(\hat{\delta}_i \hat{\delta}_j,\ \hat{\delta}_i \hat{\delta}_j + \mu_i \mu_j) \leq \text{negl} \quad \forall i, j \in [w] \end{aligned}$$

This follows from a sequence of straightforward applications of two properties of statistical distance, namely:

1. Statistical distance is a metric on distributions.
2. For every randomized function F, $\Delta\big(F(X), F(Y)\big) \leq \Delta(X, Y)$.

Now, we have:

$$\begin{aligned} \Delta(\hat{\delta}_i,\ \hat{\delta}_i + \mu_i) &\leq \text{negl} \\ \text{By Property (2) ,}\ \Delta\big(\hat{\delta}_i \hat{\delta}_j,\ (\hat{\delta}_i + \mu_i)\hat{\delta}_j\big) &\leq \text{negl} \\ \Delta\big((\hat{\delta}_i + \mu_i)\hat{\delta}_j,\ (\hat{\delta}_i + \mu_i)(\hat{\delta}_j + \mu_j)\big) &\leq \text{negl} \\ \text{By Property (1) ,}\ \Delta\big(\hat{\delta}_i \hat{\delta}_j, (\hat{\delta}_i + \mu_i)(\hat{\delta}_j + \mu_j)\big) &\leq \text{negl} \\ \Delta\big(\hat{\delta}_i \hat{\delta}_j - \hat{\delta}_i \mu_j,\ \hat{\delta}_i \hat{\delta}_j + \mu_i \hat{\delta}_j + \mu_i \mu_j\big) &\leq \text{negl} \\ \text{By Property (2) ,}\ \Delta\big(\hat{\delta}_i \hat{\delta}_j,\ \hat{\delta}_i \hat{\delta}_j - \hat{\delta}_i \mu_j\big) &\leq \text{negl} \\ \text{Hence,}\ \Delta\big(\hat{\delta}_i \hat{\delta}_j,\ \hat{\delta}_i \hat{\delta}_j + \mu_i \hat{\delta}_j + \mu_i \mu_j\big) &\leq \text{negl} \\ \text{Similarly, we obtain}\ \Delta\big(\hat{\delta}_i \hat{\delta}_j,\ \hat{\delta}_i \hat{\delta}_j + \mu_i \mu_j\big) &\leq \text{negl} \quad \text{as desired.} \end{aligned}$$

By the same reasoning, we can get that:

$$\Delta\big(\delta_i \delta_j,\ \delta_i \delta_j + \hat{\delta}_i \hat{\delta}_j\big) \leq \text{negl}$$

G Handling arbitrary challenge messages

In this section, we discuss how to generalise the proof in Section 4.1 to handle arbitrary challenge messages. This can be done in a manner analogous to the proof of linear FE, which is explained in [ABCP15, ALS]. At a high level, we design $\mathbf{B}$ to be a low norm basis that contains $\mathbf{x}_0 - \mathbf{x}_1$ as the last basis vector $\mathbf{b}_w$ and the remaining basis vectors orthogonal to it. The challenger constructs the public parameters, and the ciphertext $\mathbf{c}$, $\mathbf{b}$ along this basis. Along this basis, it holds that the adversary only queries for a monomial involving $x_{\mathbf{b}_w}$ iff $\mathbf{x}_0[\mathbf{b}_i]\mathbf{x}_0[\mathbf{b}_w] = \mathbf{x}_1[\mathbf{b}_i]\mathbf{x}_1[\mathbf{b}_w] = 0$. That is, along this basis, the above restriction on the proof holds. To convert this to an arbitrary basis, the challenger applies a change of basis transformation, which, by the construction of [ALS], is low norm.