Obfuscation from Low Noise Multilinear Maps

Abstract. Multilinear maps enable homomorphic computation on encoded values and a public procedure to check if the computation on the encoded values results in a zero. Encodings in known candidate constructions of multilinear maps have a (growing) noise component, which is crucial for security. For example, noise in GGH13 multilinear maps grows with the number of levels that need to be supported and must remain below the maximal noise supported by the multilinear map for correctness. A smaller maximal noise, which must be supported, is desirable both for reasons of security and efficiency.
In this work, we put forward new candidate constructions of obfuscation for which the maximal supported noise is polynomial (in the security parameter). Our constructions are obtained by instantiating a modification of the Lin's [EUROCRYPT 2016] obfuscation construction with composite order variants of the GGH13 multilinear maps. For these schemes, we show that the maximal supported noise only needs to grow polynomially in the security parameter. We prove the security of these constructions in the weak multilinear map model that captures *all known* vulnerabilities of GGH13 maps. Finally, we investigate the security of the considered composite order variants of GGH13 multilinear maps from a cryptanalytic standpoint.

1 Introduction

Program obfuscation aims to make computer programs "unintelligible" while keeping their functionalities intact. The known obfuscation constructions [GGH+13b, BR14, BGK+14, PST14, AGIS14, Zim15, AB15, GLSW15, BMSZ16, Lin16, BD16, GMM+16] are all based on new candidate constructions [GGH13a, CLT13, CLT15, GGH15] of multilinear maps [BS02], security of which is poorly understood [GGH13a, CHL+15, CGH+15, HJ16, CLLT16, MSZ16].

Briefly, multilinear maps (a.k.a. graded encodings) allow "leveled" homomorphic computations of an a-priori bounded degree (say κ) arbitrary polynomials on "encoded" values. Furthermore, they provide a mechanism to publicly check if the result of a polynomial computation is a zero or not. At a high level, known obfuscation methods map the program to a sequence of encodings. These encodings are such that the output of the program on a specific input is zero if and only if the output of a corresponding input dependent polynomial (of degree κ) on the encoded values yields a zero.

Noise in GGH-based Obfuscations. Encodings in the known candidate multilinear map[1] constructions are generated to have a noise component (referred

[1] Throughout this paper, we use multilinear maps to refer to private encoding multilinear maps. In other words, no public low-level encodings of zero are provided in our constructions.

to as "fresh" encoding/noise[2]) that is necessary for security. Homomorphic computations on these fresh encodings yield encodings with increased noise due to accumulation of the fresh noise (hence called "accumulated" noise). In the candidate construction by Garg, Gentry and Halevi [GGH13a] (a.k.a. GGH), the noise level in the fresh level-1 encodings can be set to be as low as a polynomial in the security parameter, without hurting the security. However, the noise level in the fresh level-i encodings needs to grows exponentially in i.[3] Furthermore, the accumulated noise also grows with the number of homomorphic multiplications. The GGH construction is parameterized by a modulus q that needs to be greater than the maximum supported noise (referred to as "noise bound") of any encoding in the system in order to preserve functionality. Most obfuscation constructions involve homomorphic multiplication of polynomially many "fresh" encodings. Therefore, these constructions need to support an exponentially large noise. The only exception to this is the recent beautiful construction of Lin [Lin16] that only needs a constant number of multiplications on composite-order multilinear maps. However, this construction still needs to give out "fresh" encodings at polynomially high levels. Thus it would still need exponential noise if one was to use a composite order variant of GGH multilinear maps (e.g. the one eluded to in the first EPRINT draft of GGH [GGH12] or the one from [GLW14b, Appendix B.3]). Another alternative is to use a composite order variant of the [CLT13, CLT15] multilinear maps, e.g. the one by Gentry et al. [GLW14b, Appendix B.3 and B.4]. Note that in the CLT based constructions the number of primes needed is always polynomial in the security parameter. This is the case even if the construction itself uses a constant number of slots, as is the case in Lin's scheme. This use of polynomially many primes is essential for security — specifically, in order to avoid lattice attacks. Consequently, the noise growth in this case is also exponential (as elaborated on in [GLW14b, Appendix B.5]).

In the context of GGH multilinear maps, the use of an exponential "noise bound," and hence the modulus q, is not desirable in light of the recent NTRU attacks[4] [ABD16, KF16]. It is desirable to have a much smaller value of q (say $\mathsf{poly}(\lambda)$). Furthermore, having a smaller modulus offers asymptotic efficiency improvements.

Weak Multilinear Map Model for GGH. Typically, candidate obfuscation schemes (including the above constructions) are proven secure in so-called *ideal graded encoding model*, that does not capture all the known vulnerabilities of the GGH multilinear maps [GGH13a, HJ16, MSZ16]. In particular, Miles, Sahai and

[2] By "fresh" encodings we mean that it is generated via the encoding procedure using the secret parameters and *not* produced as a result of homomorphic computations.

[3] The reported noise is for the recommended version of the GGH multilinear maps [GGH12, Section 6.4]. This recommendation was made in [GGH13a] with the goal of avoiding averaging attacks [GS02, NR06, DN12]. Similar, recommendation is made in [ABD16, Section 4.2].

[4] Specifically, the subfield lattice attack is sub-exponential as soon as q is super-polynomial. Furthermore, using attack of [KF16] becomes polynomial for power-of-two cyclotomic fields when $q = 2^{\Omega(\sqrt{n \log \log n})}$.

Zhandry [MSZ16] exploit these vulnerabilities of GGH to show attacks against obfuscation constructions. In light of these attacks, [MSZ16] proposed the *weak multilinear map model* that better captures the known vulnerabilities GGH multilinear maps. Subsequently, Garg et al. [GMM+16] gave an obfuscation scheme provable secure in this model.

In this work, we ask the following question.

Can we construct an obfuscation scheme using low noise multilinear maps and prove its security in the weak multilinear map model?

1.1 Our Result

In this work, we resolve the above question affirmatively providing new candidate indistinguishability obfuscation constructions such that: (i) they requires a modulus q which grows polynomially in the security parameter, and (ii) they are provably secure in the weak multilinear map model.

Our construction is instantiated using composite order GGH multilinear maps[5] that are same as the composite order proposal of [GGH12] except that we use a specific choice of the Lagrange Coefficients used in Chinese Remaindering in our construction.[6] This specific choice of Lagrange Coefficients is done in order to strengthen security — specifically, in order to obtain a proof of security in the weak multilinear map model. We evaluate the security of the GGH composite order multilinear maps (with our choice of Lagrange Coefficients) in light of known attack strategies (see Section 6.3).

Next, in order to enable constructions with low noise, we suggest two ways to modify the GGH sampling procedure [GGH12, Section 6.4] such that: (i) The first more conservative variation is simple and its security can be reduced to the security of the original GGH sampling procedure. (ii) Our second more aggressive variant departs from the GGH sampling procedure more but obtains better efficiency in terms of the dimension of the lattice necessary. From a cryptanalytic standpoint (see Section 6.3), we do not know of any attacks against this more aggressive variation.

[5] In the first draft of [GGH12], authors suggested a composite order variant of multilinear maps. However, in later versions they restricted their claims to the prime order construction. This was in light of the weak-discrete log attacks they found against their construction. However, these attacks worked only when public encodings of zero are provided and rendered assumptions such as subgroup hiding easy. In particular, all known attacks against composite order GGH maps use low level encodings of zero [GGH13a] or some specific high-level encodings of zero [CGH+15]. In light of the Miles et al. attacks [MSZ16] we envision more (potential) attacks but they are all captured by the weak multilinear map model.

[6] We do not provide public encodings of zero in our constructions. Therefore, they are insufficient to instantiate the assumptions made by Gentry et al. [GLW14a, GLSW15].

Additional Contribution. As mentioned earlier, recent work by Garg et al. [GMM+16] provides the first construction of obfuscation in the weak multilinear map model. However, this construction works by converting a circuit into a branching program. Our work also provides a direct construction (obtained by slightly modifying our main construction) for circuits, for which security can be argued in the weak multilinear map model. Previous works [AB15, Zim15] in this direction proved security in only in the ideal graded encoding model.

Independent Work. In a concurrent and independent work, Lin and Vaikuntanathan [LV16] obtain a construction which when instantiated with GGH multilinear maps would yield a construction that supports low noise. However, a bonus of our scheme is that it is proved secure in the weak multilinear map model. Furthermore, the techniques introduced in this work are orthogonal to the work of Lin and Vaikuntanathan [LV16] and are of independent interest.

1.2 Technical Overview

We start from a brief overview of Lin's construction [Lin16].

Overview of Lin's construction: $i\mathcal{O}$ from constant-degree multilinear map. It has two main steps.

- **Step-1: Stronger bootstrapping.** All existing candidates of indistinguishability obfuscation ($i\mathcal{O}$ for short) for all circuits (i.e., P/Poly) rely on "bootstrapping" $i\mathcal{O}$ for weaker class of circuits. Known techniques [GGH+13b, CLTV15] require $i\mathcal{O}$ for NC^1 to start with: the idea is to first construct a scheme only for NC^1 circuits and then use cryptographic primitives (e.g., fully homomorphic encryption) to "bootstrap" this into a construction for P/Poly. In contrast, [Lin16] uses a much stronger bootstrapping technique that only requires $i\mathcal{O}$ (with some necessary efficiency requirements) for *specific* constant-degree circuits (as opposed to *general* NC^1 circuits in the earlier constructions). To realize that, only multilinear map supporting constant number of multiplications suffice. Such specific circuit class is referred to as the "seed class" and denoted by $\mathcal{C}_{\mathsf{seed}}$ in the following.
- **Step-2: Special purpose $i\mathcal{O}$ for $\mathcal{C}_{\mathsf{seed}}$.** In the second step, [Lin16] gives a candidate $i\mathcal{O}$-construction for this seed class. The construction builds on the techniques from [AB15, Zim15] for obfuscating NC^1 circuits directly while ensuring constant-degree computation. Lin then proves that her construction is secure in the ideal graded encoding model.

Our techniques: main steps. To achive our result, we build on the bootstraping result of [Lin16] and focus on building the $i\mathcal{O}$-candidate (Step-2) for $\mathcal{C}_{\mathsf{seed}}$ such that it only requires a polynomial sized modulus and is secure in weak multilinear map model. Our main steps of construction are as follows:

1. **Composite-order GGH multilinear map.** We propose a composite-order extension of GGH multilinear map candidate. Our candidate is the same as

the first proposal of GGH maps (as in the first EPRINT version of [GGH12]) except that we use specific Lagrange Coefficients used in Chinese Remaindering in the construction. This choice allows us to argue security is the weak multilinear map model.

2. **Security in the weak multilinear map model.** We strengthen the security of basic $i\mathcal{O}$ construction of [Lin16] via so-called *self-fortification* technique similar to [GMM+16]. As a result we are able to prove that our construction is secure in the (GGH-based) weak multilinear map model (see Appendix C for details on the model).
3. **GGH with low-noise.** We propose two modification of composite-order GGH multilinear maps such that all "fresh" encodings that need to provided in our construction can be provided with noise of size $\mathsf{poly}(\lambda)$. Moreover, any κ degree computation would result into final encodings with noise of size $O(\mathsf{exp}(\kappa)\mathsf{poly}(\lambda))$. Using the fact that $\mathcal{C}_{\mathsf{seed}}$ has constant degree (i.e., κ is constant), we obtain polynomial sized modulus q. For the first variant, we argue that an attacker against our modified construction can be translated to an attack against the original construction. On the other hand, our second more aggressive variant of the sampling procedure works with better efficiency while still not being vulnerable to known attacks.

Overview of composite-order GGH multilinear map. An instance of the GGH scheme is parameterized by the security parameter λ and the required multilinearity level $\kappa \leq \text{poly}(\lambda)$. Based on these parameters, consider the $2n$th cyclotomic ring $R = \mathbb{Z}[X]/(X^n + 1)$ where n is a power of 2 (n is set large enough to ensure security), and a modulus q that defines $R_q = R/qR$ (with q large enough to support functionality). The secret encoding procedure encodes elements of a quotient ring $R/\mathcal{I}$, where $\mathcal{I}$ is a principal ideal $\mathcal{I} = \langle \boldsymbol{g} \rangle \subset R$, generated by $\boldsymbol{g}$. In the composite order setting, $\boldsymbol{g}$ is equal to a product of several (say t) "short" ring elements $\boldsymbol{g}_1, \boldsymbol{g}_2, \ldots, \boldsymbol{g}_t$. These ring elements are chosen such that the norms $N(\boldsymbol{g}_i) = |R/\langle \boldsymbol{g}_i \rangle|$ are equal to "large" (exponential in λ) primes p_i for each $\boldsymbol{g}_i$. By the Chinese Remainder Theorem (CRT for short) one can observe that the following isomorphism $R/\mathcal{I} \cong R/\mathcal{I}_1 \times \ldots \times R/\mathcal{I}_t$ for ideals $\mathcal{I}_i = \langle \boldsymbol{g}_i \rangle$ holds. Hence each element $\boldsymbol{e} \in R/\mathcal{I}$ has an equivalent CRT representation in $R/\mathcal{I}_1 \times \ldots \times R/\mathcal{I}_t$ that is denoted by $(\boldsymbol{e}[\![1]\!], \ldots, \boldsymbol{e}[\![t]\!])$. Recall that, in this representation it holds that $\boldsymbol{e} \equiv \boldsymbol{e}[\![i]\!] \bmod \mathcal{I}_i$ and $\boldsymbol{e}[\![i]\!]$ is called the value of $\boldsymbol{e}$ in the i-th *slot*; moreover, any arithmetic operation over $R/\mathcal{I}$ can be done "slot-wise." The short generator $\boldsymbol{g}$ (and all $\boldsymbol{g}_i$) is kept secret, and no "good" description of $\mathcal{I}$ (or of $\mathcal{I}_i$) is made public.

Let $\mathbb{U}$ denote the universe such that $\mathbb{U} = [\ell]$.[7] To enforce the restricted multilinear structure (a.k.a. straddling sets) secrets $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_\ell$ are sampled randomly from R_q (and hence, they are "not short"). The sets $\mathbf{v} \subseteq \mathbb{U}$ are called the levels. An encoding of an element $\boldsymbol{a} \in R/\mathcal{I}$ at a level $\mathbf{v}$ is given by $\boldsymbol{e} = [\boldsymbol{c}/\prod_{i \in \mathbf{v}} \boldsymbol{z}_i]_q \in R_q$ where $\boldsymbol{c}$ is a "short element" in $\boldsymbol{a} + \mathcal{I}$ sampled via a specific

[7] In the actual construction the structure of the elements of $\mathbb{U}$ are much involved. But for simplicity here we just assume $\mathbb{U} = [\ell]$ that suffices to convey the main idea.

procedure.[8] The quantity $\|\boldsymbol{c}\|$ is called the noise of the encoding $\boldsymbol{e}$ and is denoted by $\mathsf{noise}(\boldsymbol{e})$. Rigorous calculation from the sampling procedure (c.f. Sec 2) shows that $\mathsf{noise}(\boldsymbol{e}) = O(\mathsf{exp}(t, |\mathbf{v}|))$. Note that in Lin's construction [Lin16] as well as our construction, t will be a constant, but $|\mathbf{v}|$ is not.

The arithmetic computations are restricted by the levels of the encodings: addition is allowed between encodings in the same level whereas multiplication is allowed at levels $\mathbf{v}$ and $\mathbf{v}'$ when $\mathbf{v} \cap \mathbf{v}' = \emptyset$.[9] Furthermore, GGH map provides a public zero-testing mechanism to check if any given encoding at level $\mathbb{U}$ is an encoding of an element that is equal to $0 \bmod \boldsymbol{g}$ (equivalently $0 \bmod \boldsymbol{g}_i$ in the i-th slot for all $i \in [t]$). Notice that since the map allows κ-degree computations, the noise in the top-level encoding resulting after such a computation can be at most $O(\mathsf{exp}(\kappa, t, \ell))$.

Reducing noise in GGH. We first elaborate on the GGH sampling procedure [GGH12, Section 6.4] as follows: To encode at level $\mathbf{v} \subseteq \mathbb{U}$, the encoding procedure samples a ring element from the fractional ideal $\langle \boldsymbol{g}/\boldsymbol{z}_{\mathbf{v}} \rangle$, where $\boldsymbol{z}_{\mathbf{v}} = \prod_{i \in \mathbf{v}} \boldsymbol{z}_i$.[10] Hence, the amount of noise generated by the encoding procedure depends on the size of the generator $\boldsymbol{g}/\boldsymbol{z}_{\mathbf{v}}$, which is in turn dominated by the size of $1/\boldsymbol{z}_{\mathbf{v}}$. Generally, following [GGH13a], one can sample *atoms* $\boldsymbol{z}_i$ such that their inverse $1/\boldsymbol{z}_i$ is short in K, say n^2/q (where K is the quotient field of R). Now, expressing $\boldsymbol{z}_{\mathbf{v}}$ as $\boldsymbol{z}_{\mathbf{v}} = \prod_{i \in \mathbf{v}} \boldsymbol{z}_i$ we obtain $\|1/\boldsymbol{z}_{\mathbf{v}}\| = O(\mathsf{exp}(|\mathbf{v}|)/q^{|\mathbf{v}|})$. We show in Section 2 that the noise of the fresh encoding is dominated by $\|\boldsymbol{z}_{\mathbf{v}}\| \cdot \|1/\boldsymbol{z}_{\mathbf{v}}\|$ which grows exponentially with $|\mathbf{v}|$, i.e., the cardinality of $\mathbf{v}$. As mentioned earlier, in Lin's construction, some elements are encoded at levels $\mathbf{v}$ of cardinality polynomial in λ resulting in fresh encodings of noise $O(\mathsf{exp}(\lambda))$.

To handle the noise in encodings more carefully, we provide two possible techniques specific to our construction. The first more conservative technique is fairly simply and works by choosing the degree n of the ring R sufficiently large (larger than the size of the circuit we obfuscate). With this parameter choice we can guarantee with probability close to 1 that for all levels at which we encode and the zero-testing level that $\|\boldsymbol{z}_{\mathbf{v}}\| \cdot \|1/\boldsymbol{z}_{\mathbf{v}}\| = \mathsf{poly}(n)$. This comes at the expense of making the degree n of the ring R grow with the size of the circuit (which is still polynomial in the security parameter). This change doesn't affect the security of the scheme in any way.

The second more aggressive technique follows a different strategy and avoids the dependence of n on the size of the circuit. We first observe that, in our obfuscation construction many combinations of $\prod_i \boldsymbol{z}_i$ (i.e. many subsets of $[\ell]$) terms actually never arise. We illustrate our main idea with a toy example. Assume that we only need to encode in levels $\hat{\mathbf{v}}_i = [\ell] \setminus \{i\}$ and $\mathbf{v}_i = \{i\}$ for all

[8] We use the notation $[\cdot]_q$ to denote operations in R_q.

[9] Note that in the actual construction we use different restriction for multiplication due to difference in the structure of the straddling levels and the universe.

[10] Notice that $\boldsymbol{g}/\boldsymbol{z}_{\mathbf{v}}$ is in K. We generally use $\boldsymbol{a}/\boldsymbol{b} \in K$ to denote "division" in the quotient field K of R for $\boldsymbol{a}, \boldsymbol{b} \in R$. This is not to be confused with the notation $\boldsymbol{a} \cdot [\boldsymbol{b}^{-1}]_q \in R$ which is a multiplication operation in R where the inverse $[\boldsymbol{b}^{-1}]_q$ is in R_q.

$i \in [\ell-1]$ (and not at the level $\{\ell\}$). Now, if we follow above sampling procedure then clearly we will end up with $\|\boldsymbol{z}_{\hat{\mathbf{v}}_i}\| \cdot \|1/\boldsymbol{z}_{\hat{\mathbf{v}}_i}\| = O(\mathsf{exp}(\ell-1))$. Instead, we follow a different strategy, namely we sample all the $\boldsymbol{z}_i$ for $i \in [\ell-1]$ except the last $\boldsymbol{z}_\ell$ term "as usual", i.e. such that $1/\boldsymbol{z}_i$ is "short" in K. However for the one remaining term (i.e. $\boldsymbol{z}_\ell$) we instead sample another value $\boldsymbol{z}^\star$, such that $1/\boldsymbol{z}^\star$ is "short" in K and then set

$$\boldsymbol{z}_\ell := \left[\frac{\boldsymbol{z}^\star}{(\prod_{i\in[\ell-1]} \boldsymbol{z}_i)}\right]_q.$$

Furthermore, we require that for $i \in [\ell-1]$, $1/[\boldsymbol{z}_i^{-1}]_q$ is also short in K. We can now compute a value $\boldsymbol{z}_{\hat{\mathbf{v}}_i} := \boldsymbol{z}^\star \cdot [\boldsymbol{z}_i^{-1}]_q$.[11] Observe that it holds that $[\boldsymbol{z}_{\hat{\mathbf{v}}_i}]_q = [\prod_{i\in[\ell]\setminus\{i\}} \boldsymbol{z}_i]_q$ as desired. Moreover, $1/\boldsymbol{z}_{\hat{\mathbf{v}}_i}$ is now short in K:

$$\|1/\boldsymbol{z}_{\hat{\mathbf{v}}_i}\| = \|1/(z^\star \cdot [\boldsymbol{z}_i^{-1}]_q)\| \leq \sqrt{n} \cdot \|1/z^\star\| \cdot \|1/[\boldsymbol{z}_i^{-1}]_q\|,$$

which is "short". The cost incurred by this modification is that $1/\boldsymbol{z}_\ell$ may *not* be "short" in K. However, this will not pose a problem as $\boldsymbol{z}_\ell$ is not used to sample encodings anyway (recall that we do *not* require to sample at level $\{\ell\}$). We show that such modification brings down the noise bound of fresh encodings to $O(\mathsf{poly}(\ell)\mathsf{exp}(t))$ and maximum noise bound (in any encoding produced in our construction) to $O(\mathsf{poly}(\ell)\mathsf{exp}(\kappa, t))$. In our scheme, both κ and t will be constants.

Our security model: weak multilinear map model. Typically, obfuscation candidates[12] were proven secure in the so-called *ideal graded encoding* model. In contrast, we prove security of our construction in the *weak multilinear map model* [MSZ16], a model that captures all currently known vulnerabilities of multilinear maps. This model is similar to ideal multilinear map model (a.k.a., the ideal graded encoding model). However, it additionally allows for computation on ring elements resulting from zero-test performed on encodings of 0. The security definition requires that the adversary can not come up with a polynomial which evaluates to 0 over these post-zero ring elements. In the composite order setting we require that the adversary can not come up with a polynomial which evaluates to 0 in *any* of the slots. Unlike the ideal model, this model is *not entirely agnostic* about the underlying multilinear map instantiation. In particular, our weak multilinear map model is based on the composite-order GGH multilinear maps and captures *all all* our attack direction investigated in our cryptanalysis.

Self-fortification from constant-degree multilinear maps. To prove security of our obfuscation candidate in the weak multilinear map model, we make another modification to Lin's obfuscation scheme for $\mathcal{C}_{\mathsf{seed}}$ using the self-fortification technique similar to [GMM+16].

[11] Notice that, the inverse is in R_q but the product is in R

[12] There are some works e.g. [BMSZ16, BD16] that prove security of their constructions in slightly stronger models than the ideal graded encoding model which captures some attacks on multilinear maps.

Recall that multilinear maps allow for testing of zero-encoding at the universe set (a.k.a. the top level). All known attacks against multilinear map candidates exploit the "sensitive information" leaked upon a successful zero-test. To protect against these attacks, the idea of [GMM+16] is to render this "sensitive leakage" useless by "masking" with a PRF output. Similarly, we achieve this by augmenting the given circuit $\mathcal{C}$ with a *parallel* PRF computation, the output of which would be used to mask the leakage from real computation. More care is required so that the PRF computation does *not* affect the actual computation of $\mathcal{C}$ and "comes alive" only after a successful zero-test.

Before we describe our transformation, let us first describe the techniques of obfuscating circuits directly of [AB15, Zim15] and that is also used in [Lin16]. At a high level, consider a universal circuit $\mathcal{U}$ that takes as input the circuit (to obfuscate) $\mathcal{C}$ and the input x to $\mathcal{C}$ and outputs $\mathcal{C}(x)$. The obfuscation consists of a collection of values in $R/\mathcal{I}$ encoded at carefully chosen levels (i.e., straddling sets). Multiple slots are used where w.l.o.g. the first slot is used for actual computation and a bunch of other slots are added with random values. These random values along with the choice of straddling sets ensure that the random values are nullified only with a correct (and consistent) evaluation corresponding to some input x. More precisely, a correct evaluation leads to an encoding of $(\mathcal{U}(\mathcal{C},x) \bmod \boldsymbol{g}_1, 0 \bmod \boldsymbol{g}_2, \ldots, 0 \bmod \boldsymbol{g}_t)$ at the highest level $\mathbb{U}$; zero-testing of which would reveal the output. On the other hand, any incorrect computation would not cancel out all random values, and hence would result in a non-zero value in $\bmod \boldsymbol{g}$ with all but negligible probability.

Our idea is to add an extra slot (say the second slot) for PRF computation such that a correct computation would produce an encoding of $(\mathcal{U}(\mathcal{C},x) \bmod \boldsymbol{g}_1, \boldsymbol{g}_2 \cdot \mathcal{U}(\mathcal{C}^{\mathsf{PRF}_\psi}, x), 0 \bmod \boldsymbol{g}_3, \ldots, 0 \bmod \boldsymbol{g}_{t+1})$[13] at the top level.[14] Notice that due to a $\boldsymbol{g}_2$ multiplier in the second slot, the computation is not affected by the PRF output as the value in the second slot is still $0 \bmod \boldsymbol{g}_2$. Nonetheless, we show that a successful zero-test returns a ring element (say $\boldsymbol{f}$) in $R/\mathcal{I}$ that has a blinding (additive) factor $\boldsymbol{\alpha} \cdot \mathcal{C}^{\mathsf{PRF}_\psi}(x)$ for some $\boldsymbol{\alpha} \in R/\mathcal{I}$. Furthermore, we are able to show that as long as $\boldsymbol{\alpha}$ is invertible in (the composite order quotient ring) $R/\mathcal{I}$ the CRT representation of $\boldsymbol{f}$ given by $(\boldsymbol{f}[\![1]\!], \ldots, \boldsymbol{f}[\![t]\!])$ is "somewhat random" in each slot (formally, $\boldsymbol{f}[\![i]\!]$ has high min-entropy).

Cryptanalysis. In Section 6.2, we discuss our change to composite order generators $\boldsymbol{g}$ from a cryptanalytic perspective. In a nutshell, existing lattice attacks, such as attacks against overstretched NTRU assumptions [ABD16, KF16], do not exploit the specific distribution of instances, but rather geometric properties (i.e. noise terms being short). Thus, our construction resists currently known lattice attacks and there is no reason to believe choosing composite generators $\boldsymbol{g} = \prod_i \boldsymbol{g}_i$ leads to less secure schemes than choosing primes ones. However, we do know that top-level encodings of zero, with correlated randomness, can be

[13] $\mathcal{C}^{\mathsf{PRF}_\psi}$ is a circuit for computing PRF with the key ψ.

[14] In the construction this is implemented by canceling out the PRF value by multiplying with an appropriate encoding that encodes a value which is $0 \bmod \boldsymbol{g}_2$ in the second slot.

dangerous. This is especially the case if they can be used to obtain an element in the ideal $\langle \boldsymbol{g} \rangle$. In the composite order setting, we expect potential attacks if an element in the ideal $\langle \boldsymbol{g}_i \rangle$ for any i can be computed. However, all these potential attacks are captured by the weak multilinear map model that we consider. At a high level, our proof in the weak multilinear map model guarantees that no element in the ideal $\langle \boldsymbol{g}_i \rangle$ for any i can be computed.

In Section 6.3, we discuss reasons for the believed security of our more aggressive variation of the GGH sampling procedure.

1.3 Roadmap

The rest of the paper is organized as follows. In Sec. 2 we provide a composite-order GGH multilinear map candidate. In Sec. 3 we briefly mention Lin's bootstrapping theorem and a few related definitions. Our main $i\mathcal{O}$-construction is provided in Sec. 4. In Sec. 5 we provide our modifications on the composite-order GGH multilinear map to achieve low noise. We conclude the main body of the paper in Sec. 6 with a cryptanalytic discussion of our modifications to the asymmetric GGH multilinear maps. The formal description of weak multilinear map model is provided in Appendix C and the preliminaries can be found in Appendix A and Appendix B.

2 Composite-Order GGH Graded Encodings

In this section we describe a version of the GGH graded encoding scheme [GGH13a] that supports operations over composite-order groups. Composite order instantiations are known over the integers [CLT13, CLT15], but no composite-order instantiations of the GGH graded encoding scheme were explicitly described so far. Below we describe a composite-order instantiation of GGH graded encoding scheme that also has a few extra properties, which allow us to use it to instantiate the *self fortification* paradigm of Garg et al. [GMM+16]. Our new scheme differs from the GGH scheme only with respect to the instance generation and encoding procedures. In a nutshell, the ideal generator $\boldsymbol{g}$ is sampled as a product of pairwise coprime factors $\boldsymbol{g}_i$, each of which has (large) prime norm.

We use the cyclotomic field $K = \mathbb{Q}[X]/(X^n+1)$ and the ring $R = \mathbb{Z}[X]/(X^n+1)$. Somewhat more subtle changes will be necessary in the encoding procedure. Given elements $(\boldsymbol{a}_1, \ldots, \boldsymbol{a}_\ell) \in R/\langle \boldsymbol{g}_1 \rangle \times \cdots \times R/\langle \boldsymbol{g}_\ell \rangle$ for the slots, we will reconstruct an element $\boldsymbol{a} \in R$ using the Chinese Remainder Theorem. The Chinese Remainder Theorem basis has to be chosen carefully such that reconstructed elements $\boldsymbol{a} \in R$ have small size. We will use a Chinese Remainder Theorem basis of the form $(\gamma_i \cdot \prod_{j \neq i} \boldsymbol{g}_j)_i$. One technical requirement for arguing security of our obfuscator in the weak multilinear maps model is that the γ_i are units in $R/\langle \boldsymbol{g} \rangle$. This condition can be met by reconstructing the $\gamma_i \in R$ from $(1, \ldots, (\prod_{j \neq i} \boldsymbol{g}_j)^{-1}, \ldots, 1)$ and *not* reducing the basis elements $\gamma_i \cdot \prod_{j \neq i} \boldsymbol{g}_j$ modulo $\langle \boldsymbol{g} \rangle$.

2.1 Our Scheme

We will now describe our instantiation of composite-order GGH scheme more formally. Let n be a power of 2. Just like the GGH construction, we use the cyclotomic field $K = \mathbb{Q}[X]/(X^n + 1)$ and the rings $R = \mathbb{Z}[X]/(X^n + 1)$ and $R_q = R/qR$. Let $\mathbf{v}_{zt}$ be a $k_1 \times k_2$ matrix of non-negative integers; we call $\mathbf{v}_{zt}$ the (straddling) universe. We refer to $k_1 \times k_2$ matrices[15] $\mathbf{v}$ of non-negative integers as *levels* and define their weight as $|\mathbf{v}| = \sum_{i,j} \mathbf{v}_{ij}$.

Instance generation: $(\mathsf{params}, \mathsf{sparams}, \boldsymbol{p}_{zt}) \leftarrow \mathsf{InstGen}(1^\lambda, 1^\ell, \mathbf{v}_{zt})$**.**

- Choose invertible $\boldsymbol{z}_{ij} \leftarrow R_q^\times$ for $(i,j) \in [k_1] \times [k_2]$ uniformly at random repeatedly until for all i, j, $\|1/\boldsymbol{z}_{ij}\| < n^2/q$ in K (Lemma 8, Appendix B) [16].
- For all $i \in [\ell]$ sample $\boldsymbol{g}_i \leftarrow D_{\mathbb{Z}^n,\sigma}$ with $\sigma = \lambda\sqrt{n}$ repeatedly until the following conditions are met: (i) $\|\boldsymbol{g}_i\| \leq \sigma\sqrt{n}$ and $\boldsymbol{g}_i$ is invertible in R_q, (ii) $\|1/\boldsymbol{g}_i\| \leq n^c$ (in K)[17] for an appropriate constant c, (iii) $N(\boldsymbol{g}_i) \geq 2^{\Omega(n)}$ is a prime and (iv) for all distinct i, j the ideals $\langle \boldsymbol{g}_i \rangle$ and $\langle \boldsymbol{g}_j \rangle$ are co-prime. As argued in GGH such $(\boldsymbol{g}_1, \ldots, \boldsymbol{g}_\ell)$ can be obtained after an expected polynomial number of trials under mild number-theoretic assumptions.

Denote the product $\prod_{i=1}^{\ell} \boldsymbol{g}_i$ by $\boldsymbol{g}$. Define the ideals $\mathcal{I}_i = \langle \boldsymbol{g}_i \rangle$ and $\mathcal{I} = \langle \boldsymbol{g} \rangle$. Note that by the Chinese Remainder Theorem (CRT for short) we have $R/\mathcal{I} \cong R/\mathcal{I}_1 \times \cdots \times R/\mathcal{I}_\ell$ as the ideals $\mathcal{I}_i$ are pairwise coprime. Any element $\boldsymbol{a}$ in the modular ring $R/\mathcal{I}$ can be represented via the CRT isomorphism as a tuple $(\boldsymbol{a}_1, \ldots, \boldsymbol{a}_\ell)$ in $R/\mathcal{I}_1 \times \cdots \times R/\mathcal{I}_\ell$ and vice versa. We will use a particular CRT basis with additional properties. Specifically, let $\gamma_1, \ldots, \gamma_\ell \in R$ be elements such that $\gamma_i \prod_{j \neq i} \boldsymbol{g}_j \equiv 1 \bmod \boldsymbol{g}_i$ and $\gamma_i \equiv 1 \bmod \langle \boldsymbol{g}_j \rangle$ for $j \neq i$. Such $\gamma_i \in R$ can be found by standard Lagrange interpolation. We further assume that the γ_i have been reduced with Babai's roundoff algorithm (c.f. Appendix A.2) with respect to $\mathcal{I} = \langle \boldsymbol{g} \rangle$, i.e. it holds that for all i we have $\|\gamma_i\| \leq \frac{n}{2} \cdot \|\boldsymbol{g}\|$. We will perform CRT reconstruction with respect to the basis $\{\gamma_i \cdot \prod_{j \neq i} \boldsymbol{g}_j\}_{i \in [\ell]}$, i.e. an element $(\boldsymbol{a}_1, \ldots, \boldsymbol{a}_\ell) \in R/\mathcal{I}_1 \times \cdots \times R/\mathcal{I}_\ell$ is embedded into R via

$$\Phi_B(\boldsymbol{a}_1, \ldots, \boldsymbol{a}_\ell) = \sum_i \boldsymbol{a}_i \cdot \gamma_i \cdot \prod_{j \neq i} \boldsymbol{g}_j$$

We assume that each $\boldsymbol{a}_i$ is represented in R and has been reduced with respect to $\mathcal{I}_i = \langle \boldsymbol{g}_i \rangle$ with Babai's roundoff algorithms, i.e. $\|\boldsymbol{a}_i\| \leq \frac{n}{2}\|\boldsymbol{g}_i\|$. The instance generation procedure ensures that $\|\boldsymbol{g}_i\| \leq \lambda \cdot n$, thus we also get that $\|\boldsymbol{g}\| \leq n^{\frac{\ell}{2}} \prod_i \|\boldsymbol{g}_i\| \leq \lambda^\ell n^{\frac{3}{2}\ell}$. Using this, we can bound the size of $\Phi_B(\boldsymbol{a}_1, \ldots, \boldsymbol{a}_n)$ by

$$\|\Phi_B(\boldsymbol{a}_1, \ldots, \boldsymbol{a}_\ell)\| \leq n^{(\ell+1)/2} \sum_i \|\boldsymbol{a}_i\| \cdot \|\gamma_i\| \cdot \prod_{j \neq i} \|\boldsymbol{g}_j\| \leq \frac{\ell}{4} \lambda^{2\ell} n^{O(\ell)} \tag{2.1}$$

[15] Here, we are using matrices to denote levels instead of sets in order to be consistent with our construction later.

[16] This condition is necessary to ensure correctness of the encoding procedure.

[17] This technical condition is needed for the zero-test to work.

Looking ahead, we have this particular choice of the γ_i as we will later need these terms to be invertible in $R/\langle \boldsymbol{g} \rangle$ in one of our security proof. In this context, notice that the ouput of Φ_B is not reduced modulo $\mathcal{I}$, as this would destroy this particular structure of the γ_i.

Next, we sample the zero testing parameter $\boldsymbol{p}_{zt}$. Let $\boldsymbol{z}_{\mathbf{v}_{zt}} \in R$ be computed by $\boldsymbol{z}_{\mathbf{v}_{zt}} = \prod_{i,j} \boldsymbol{z}_{ij}^{\mathbf{v}_{zt}(i,j)}$ as a product in R, i.e. we have that $\|\boldsymbol{z}_{\mathbf{v}_{zt}}\| \leq n^{O(|\mathbf{v}_{zt}|)} \cdot q^{|\mathbf{v}_{zt}|}$ and $\|1/\boldsymbol{z}_{\mathbf{v}_{zt}}\| \leq n^{\frac{5}{2}|\mathbf{v}_{zt}|}/q^{|\mathbf{v}_{zt}|}$. We sample an element $\boldsymbol{h}^\star$ from a discrete gaussian with parameter $\sqrt{q} \cdot \|\boldsymbol{z}_{\mathbf{v}_{zt}}/\boldsymbol{g}\| \leq \sqrt{q} \cdot \sqrt{n} \cdot \|\boldsymbol{z}_{\mathbf{v}_{zt}}\| \cdot \|1/\boldsymbol{g}\|$ over the fractional ideal $\langle \boldsymbol{z}_{\mathbf{v}_{zt}}/\boldsymbol{g} \rangle$. The choice of this gaussian parameter ensures that we can efficiently sample from this distribution via the GPV sampler (Theorem 2). We compute $\boldsymbol{h} = \boldsymbol{h}^\star \cdot \boldsymbol{g}/\boldsymbol{z}_{\mathbf{v}_{zt}} \in K$ and note that $\boldsymbol{h} \in R$. If for any $i \in \{1, \ldots, \ell\}$ it holds that $\boldsymbol{h} \in \langle \boldsymbol{g}_i \rangle$ we reject $\boldsymbol{h}$ and resample it until $\boldsymbol{h} \notin \langle \boldsymbol{g}_i \rangle$ for all i. We then set $\boldsymbol{p}_{zt} = [\boldsymbol{h} \cdot \boldsymbol{z}_{\mathbf{v}_{zt}} \cdot \boldsymbol{g}^{-1}]_q$. Notice that by Lemma 4 the size of $\boldsymbol{h}^\star$ is bounded by $O(\sqrt{q} \cdot n \cdot \|\boldsymbol{z}_{\mathbf{v}_{zt}}\| \cdot \|1/\boldsymbol{g}\|)$. We can therefore bound the size of $\boldsymbol{h}$ in K by

$$\begin{aligned}
\|\boldsymbol{h}\| &\leq n \cdot \|\boldsymbol{h}^\star\| \cdot \|\boldsymbol{g}\| \cdot \|1/\boldsymbol{z}_{\mathbf{v}_{zt}}\| \\
&\leq O(\sqrt{q} \cdot n \cdot \|\boldsymbol{g}\| \cdot \|1/\boldsymbol{g}\| \cdot \|\boldsymbol{z}_{\mathbf{v}_{zt}}\| \cdot \|1/\boldsymbol{z}_{\mathbf{v}_{zt}}\|) \\
&\leq O(\sqrt{q} \cdot n^{O(1)} \cdot \|\boldsymbol{z}_{\mathbf{v}_{zt}}\| \cdot \|1/\boldsymbol{z}_{\mathbf{v}_{zt}}\|),
\end{aligned}$$

i.e. the length of $\boldsymbol{h}$ is dominated by the product $\|\boldsymbol{z}_{\mathbf{v}_{zt}}\| \cdot \|1/\boldsymbol{z}_{\mathbf{v}_{zt}}\|$. For the above choice of $\boldsymbol{z}_{\mathbf{v}_{zt}}$ we get $\|\boldsymbol{z}_{\mathbf{v}_{zt}}\| \cdot \|1/\boldsymbol{z}_{\mathbf{v}_{zt}}\| = n^{O(|\mathbf{v}_{zt}|)}$ and therefore $\|\boldsymbol{h}\| = O(\sqrt{q} \cdot n^{O(|\mathbf{v}_{zt}|)})$, which means that the length of $\boldsymbol{h}$ depends exponentially on $|\mathbf{v}_{zt}|$. The instance-generation procedure outputs the public parameters $\mathsf{params} = (n, q)$, the public zero-test parameters $\boldsymbol{p}_{zt}$ and the secret parameters $\mathsf{sparams} = (\boldsymbol{g}, \{\boldsymbol{z}_{ij}\}, B)$.

Encoding of $(\boldsymbol{a}_1, \ldots, \boldsymbol{a}_\ell)$ at level $\mathbf{v}$: $\boldsymbol{u} \leftarrow \mathsf{enc}(\mathsf{sparams}, \mathbf{v}, (\boldsymbol{a}_1, \ldots, \boldsymbol{a}_\ell))$.

First embed $(\boldsymbol{a}_1, \ldots, \boldsymbol{a}_\ell)$ into R by computing $\boldsymbol{a} = \Phi_B(\boldsymbol{a}_1, \ldots, \boldsymbol{a}_\ell)$. Next, set $\boldsymbol{z}_{\mathbf{v}} = \prod_{i,j} \boldsymbol{z}_{ij}^{\mathbf{v}_{ij}}$ and notice that it holds $\|\boldsymbol{z}_{\mathbf{v}}\| \leq n^{O(|\mathbf{v}|)} \cdot q^{|\mathbf{v}|}$ and $\|1/\boldsymbol{z}_{\mathbf{v}}\| = n^{O(|\mathbf{v}|)}/q^{|\mathbf{v}|}$. Sample an element $\boldsymbol{d}^\star$ from a discrete gaussian with parameter $\lambda \cdot \|\boldsymbol{g}/\boldsymbol{z}_{\mathbf{v}}\| \leq \lambda \cdot \sqrt{n} \cdot \|\boldsymbol{g}\| \cdot \|1/\boldsymbol{z}_{\mathbf{v}}\|$ over the fractional ideal $\langle \boldsymbol{g}/\boldsymbol{z}_{\mathbf{v}} \rangle$ and set $\boldsymbol{d} = \boldsymbol{d}^\star \cdot \boldsymbol{z}_{\mathbf{v}}/\boldsymbol{g} \in R$. The choice of this gaussian parameter ensures that we can efficiently sample from this distribution via the GPV sampler (Theorem 2). Output the encoding $\left[\frac{\boldsymbol{a}+\boldsymbol{d}\cdot\boldsymbol{g}}{\boldsymbol{z}_{\mathbf{v}}}\right]_q \in R_q$.

Notice that the noise level of the encoding is bounded by

$$\|\boldsymbol{a} + \boldsymbol{d} \cdot \boldsymbol{g}\| \leq \|\boldsymbol{a}\| + \sqrt{n} \cdot \|\boldsymbol{d}\| \cdot \|\boldsymbol{g}\|.$$

We can bound $\|\boldsymbol{a}\|$ by $n^{O(\ell)}$ via Equation (2.1). We can bound the size of $\boldsymbol{d}^\star$ by $O(\lambda \cdot n \cdot \|\boldsymbol{g}\| \cdot \|1/\boldsymbol{z}_{\mathbf{v}}\|)$ via Lemma 4 (Appendix A.2) and therefore get a bound on $\|\boldsymbol{d}\|$ by

$$\begin{aligned}
\|\boldsymbol{d}\| &\leq n \cdot \|\boldsymbol{d}^\star\| \cdot \|\boldsymbol{z}_{\mathbf{v}}\| \cdot \|1/\boldsymbol{g}\| \\
&\leq O(n \cdot \lambda \cdot n \cdot \|\boldsymbol{g}\| \cdot \|1/\boldsymbol{z}_{\mathbf{v}}\| \cdot \|\boldsymbol{z}_{\mathbf{v}}\| \cdot \|1/\boldsymbol{g}\|) \\
&\leq O(n^{O(1)} \cdot \|\boldsymbol{z}_{\mathbf{v}}\| \cdot \|1/\boldsymbol{z}_{\mathbf{v}}\|),
\end{aligned}$$

i.e. the size of $\boldsymbol{d}$ is dominated by $\|\boldsymbol{z}_{\mathbf{v}}\| \cdot \|1/\boldsymbol{z}_{\mathbf{v}}\|$. By the choice of $\boldsymbol{z}_{\mathbf{v}}$ we have $\|\boldsymbol{z}_{\mathbf{v}}\| \cdot \|1/\boldsymbol{z}_{\mathbf{v}}\| \leq n^{O(|\mathbf{v}|)}$, which is exponential in $|\mathbf{v}|$. Overall, we get that the noise level is bounded by $\|\boldsymbol{a} + \boldsymbol{d} \cdot \boldsymbol{g}\| \leq O(n^{O(\ell)} + n^{O(|\mathbf{v}|)})$.

Adding and multiplying encodings. It is easy to see that the encoding as above is additively homomorphic over $R/\mathcal{I} \cong R/\mathcal{I}_1 \times \cdots \times R/\mathcal{I}_\ell$ for a bounded number of additions, in the sense that adding encodings at the same level yields an encoding of the sum at the same level $\mathbf{v}$. By the triangle inequality, the size of the numerator of the sum can be bounded by the sum of the sizes of the numerators of the summands. More precisely, let $\boldsymbol{z}_{\mathbf{v}} = \prod_{i,j} \boldsymbol{z}_{ij}^{\mathbf{v}_{ij}}$. It holds that

$$\sum_i \left[\frac{\boldsymbol{a}_i + \boldsymbol{d}_i \boldsymbol{g}}{\boldsymbol{z}_{\mathbf{v}}}\right]_q = \left[\frac{\sum_i \boldsymbol{a}_i + (\sum_i \boldsymbol{d}_i)\boldsymbol{g}}{\boldsymbol{z}_{\mathbf{v}}}\right]_q,$$

and it holds that $\|\sum_i \boldsymbol{a}_i + (\sum_i \boldsymbol{d}_i)\boldsymbol{g}\| \leq \sum_i \|\boldsymbol{a}_i + \boldsymbol{d}_i \boldsymbol{g}\|$.

Moreover, since $\mathcal{I}$ is an ideal in R, multiplying two encodings at levels $\mathbf{v}_1$ and $\mathbf{v}_2$ yields an encoding of the product at level $\mathbf{v}_1 + \mathbf{v}_2$, where the size of the numerator grows as the product of the sizes of the numerators of the multiplicands. Specifically

$$\left[\frac{\boldsymbol{a}_1 + \boldsymbol{d}_1 \boldsymbol{g}}{\boldsymbol{z}_{\mathbf{v}_1}}\right]_q \cdot \left[\frac{\boldsymbol{a}_2 + \boldsymbol{d}_2 \boldsymbol{g}}{\boldsymbol{z}_{\mathbf{v}_2}}\right]_q = \left[\frac{\boldsymbol{a}_1 \cdot \boldsymbol{a}_2 + (\boldsymbol{a}_1 \boldsymbol{d}_2 + \boldsymbol{a}_2 \boldsymbol{d}_1 + \boldsymbol{d}_1 \boldsymbol{d}_2 \boldsymbol{g})\boldsymbol{g}}{\boldsymbol{z}_{\mathbf{v}_1 + \mathbf{v}_2}}\right]_q,$$

and it holds that $\|\boldsymbol{a}_1 \cdot \boldsymbol{a}_2 + (\boldsymbol{a}_1 \boldsymbol{d}_2 + \boldsymbol{a}_2 \boldsymbol{d}_1 + \boldsymbol{d}_1 \boldsymbol{d}_2 \boldsymbol{g})\boldsymbol{g}\| \leq \sqrt{n} \cdot \|\boldsymbol{a}_1 + \boldsymbol{d}_1 \boldsymbol{g}\| \cdot \|\boldsymbol{a}_2 + \boldsymbol{d}_2 \boldsymbol{g}\|$.

Finally, notice that via the Chinese Remainder Theorem additions and multiplications in $R/\mathcal{I}$ correspond to component wise additions and multiplications in the *slots* $R/\mathcal{I}_i$.

Zero testing: $\mathsf{isZero}(\mathsf{params}, \boldsymbol{p}_{zt}, \boldsymbol{u}) \stackrel{?}{=} 0/1$. Recall that we are testing if an encoding $\boldsymbol{u}$ is $0 \pmod{\mathcal{I}}$, which which is exactly the case if $\boldsymbol{u}$ is identically 0 in all slots. To test if a level $\mathbf{v}_{zt}$ encoding $\boldsymbol{u} = [\boldsymbol{c}/\boldsymbol{z}_{\mathbf{v}_{zt}}]_q$ is an encoding of 0 $(\text{mod } \mathcal{I})$, we just multiply it in R_q by $\boldsymbol{p}_{zt}$ and check whether the resulting element $\boldsymbol{w} = [\boldsymbol{p}_{zt} \cdot \boldsymbol{u}]_q$ is short (e.g., shorter than $q^{3/4}$). Namely, we use the test

$$\mathsf{isZero}(\mathsf{params}, \boldsymbol{p}_{zt}, \boldsymbol{u}) \;=\; \begin{cases} 1 \text{ if } \|[\boldsymbol{p}_{zt}\boldsymbol{u}]_q\|_\infty < q^{3/4} \\ 0 \text{ otherwise} \end{cases} \tag{2.2}$$

We will now argue correctness of our zero-testing procedure. This is analogous to the GGH construction but needs a reproof as we are now working with a polynomial size q (GGH used a super-polynomial sized q). Let $\boldsymbol{u} = [\boldsymbol{c}/\boldsymbol{z}_{\mathbf{v}_{zt}}]$ be a correctly computed encoding at level $\mathbf{v}_{zt}$ and assume that q is large enough such that the noise level $\|\boldsymbol{c}\|$ of $\boldsymbol{u}$ is bounded by $q^{1/8}$. First assume that $\boldsymbol{u}$ is an encoding of zero at level $\mathbf{v}_{zt}$. Then it holds that $\boldsymbol{c} = \boldsymbol{r} \cdot \boldsymbol{g}$ for an $\boldsymbol{r} \in R$. We can bound the size of $\boldsymbol{r}$ by

$$\|\boldsymbol{r}\| = \|\boldsymbol{c} \cdot \boldsymbol{g}^{-1}\| \leq \sqrt{n} \cdot \|\boldsymbol{c}\| \cdot \|\boldsymbol{g}^{-1}\| \leq q^{1/8} \cdot n^{O(1)}.$$

Thus it holds that

$$[\boldsymbol{p}_{zt} \cdot \boldsymbol{u}]_q = \left[\frac{\boldsymbol{h} \cdot \boldsymbol{z}_{\mathbf{v}_{zt}}}{\boldsymbol{g}} \cdot \frac{\boldsymbol{r} \cdot \boldsymbol{g}}{\boldsymbol{z}_{\mathbf{v}_{zt}}} \right]_q = [\boldsymbol{h} \cdot \boldsymbol{r}]_q.$$

We can bound the size of $\boldsymbol{h} \cdot \boldsymbol{r}$ by

$$\|\boldsymbol{h} \cdot \boldsymbol{r}\| \leq \sqrt{n} \cdot \|\boldsymbol{h}\| \cdot \|\boldsymbol{r}\| \leq \sqrt{q} \cdot q^{1/8} \cdot n^{O(|\mathbf{v}_{zt}|)} \leq q^{5/8} \cdot n^{O(|\mathbf{v}_{zt}|)}.$$

Thus, if we choose q sufficiently large such that the above is upper bounded by $q^{3/4}$, then encodings of zero will pass the zero test.

Now assume that $\boldsymbol{c}$ is not an encoding of zero, i.e. it holds that $\boldsymbol{c} \notin \langle \boldsymbol{g} \rangle$. The zero test computes a value

$$\boldsymbol{w} = [\boldsymbol{p}_{zt} \cdot \boldsymbol{u}]_q = [\boldsymbol{h} \cdot \boldsymbol{c}/\boldsymbol{g}]_q.$$

Assume that the zero test fails on $\boldsymbol{w}$, i.e. it holds that $\|\boldsymbol{w}\| \leq q^{3/4}$. Then it holds that

$$\|\boldsymbol{w} \cdot \boldsymbol{g}\| \leq \sqrt{n} \cdot \|\boldsymbol{w}\| \cdot \|\boldsymbol{g}\| \leq q^{3/4} n^{O(1)} < q/2$$

and

$$\|\boldsymbol{h} \cdot \boldsymbol{c}\| \leq \sqrt{n} \cdot \|\boldsymbol{h}\| \cdot \|\boldsymbol{c}\| \leq \|\boldsymbol{h}\| \cdot B \cdot n^{O(1)} \leq \sqrt{q} \cdot n^{O(|\mathbf{v}_{zt}|)} \cdot q^{1/8} \leq q^{5/8} n^{O(|\mathbf{v}_{zt}|)} < q/2$$

hold. But this means that $\boldsymbol{w} \cdot \boldsymbol{g} = \boldsymbol{h} \cdot \boldsymbol{c}$ in R, as this equality holds modulo q as both sides are smaller than $q/2$. Since R is a unique factorization domain and none of the irreducible factors of $\boldsymbol{g}$ divides $\boldsymbol{h}$ (by construction of $\boldsymbol{h}$), it must holds that $\boldsymbol{c} \in \langle \boldsymbol{g} \rangle$, which is a contradiction.

Thus, if we choose q sufficiently large depending on the $n^{O(1)}$ and $n^{O(|\mathbf{v}_{zt}|)}$ factors above, we can conclude that the zero test has perfect correctness.

2.2 Discussion on Noise

Notice that the size of the blinding term $\boldsymbol{h}$ in the zero testing parameter $\boldsymbol{p}_{zt}$ and the noise level $\|\boldsymbol{a} + \boldsymbol{d} \cdot \boldsymbol{g}\|$ depend exponentially on the size of the straddling set $\mathbf{v}_{zt}$ for the zero testing parameter and encoding level $\mathbf{v}$ respectively. As discussed in the description of the instance generation and encoding procedures, the critical terms that are responsible for this exponential dependency are the products $\|\boldsymbol{z}_{\mathbf{v}}\| \cdot \|1/\boldsymbol{z}_{\mathbf{v}}\|$ for $\mathbf{v} = \mathbf{v}_{zt}$ when we sample $\boldsymbol{h}$ and levels $\mathbf{v}$ at which we encode. Looking ahead, in Section 5 we will remove this exponential dependency by providing a new sampling procedure for the $\boldsymbol{z}_{ij}$ terms that is custom-made for the straddling sets used in our construction in Section 4. This new sampling procedure will ensure that $\|\boldsymbol{z}_{\mathbf{v}}\| \cdot \|1/\boldsymbol{z}_{\mathbf{v}}\| \leq n^{O(1)}$ for all levels $\mathbf{v}$ at which we encode and $\mathbf{v} = \mathbf{v}_{zt}$. This will ensure that all encodings have polynomial noise level and $\boldsymbol{h}$ has length $\sqrt{q} \cdot n^{O(1)}$. Moreover, the size of the CRT encoded values $\boldsymbol{a} = \Phi_B(\boldsymbol{a}_1, \ldots, \boldsymbol{a}_\ell)$ also depends exponentially on the number of slots ℓ, but this will not pose a problem as our construction in Section 4 uses a constant number of slots.

As in the discussion of the zero test, the scheme is correct if we can guarantee that the noise level (i.e. the size of the numerator) never exceeds (say) $q^{1/8}$. Thus, we will always choose the parameter q at last as a function of all remaining parameters, including the circuit we want to evaluate. This will become important in Section 5, where we can actually choose q to be polynomial in n and the size of a (universal) circuit.

3 Bootstrapping $i\mathcal{O}$ for Special Purpose Circuits

In this section, we state the main results from [Lin16] relevant to our work. The main result of [Lin16] is as follows:

Theorem 1 (Bootstrapping $i\mathcal{O}$ for constant degree circuits, [Lin16], Theorem 5). *Assume sub-exponential hardness of LWE, and the existence of a sub-exponentially secure constant-degree PRG. There exist a family of circuit classes of constant degree, such that, $i\mathcal{O}$ for that family with universal efficiency can be bootstrapped into $i\mathcal{O}$ for P/poly.*

Universal efficiency means the following: $i\mathcal{O}$ for constant degree circuits has universal efficiency if the run-time of the obfuscator is independent of the degree of the computation. More precisely, there is a universal polynomial p such that for every circuit C of degree d, obfuscating C takes time $p(1^\lambda, |C|)$, for a sufficiently large λ.

Moreover, in Lin's $i\mathcal{O}$ construction, it does not suffice that the circuits of seed class of a constant degree. In fact, the degree of multilinearity required from multilinear maps grows with the *type degree* and *input types* of the special circuits used for bootstrapping in the above theorem.

One of the main contributions of [Lin16] is to prove that the seed class of circuits indeed have constant number of input types as well as constant type degree. For the purpose of being self-contained, we define the input types and type degree first.

Definition 1 (Type Function, [Lin16], Definition 18). *Let Σ be any alphabet where every symbol in Σ is represented as a binary string of length $\ell \in \mathbb{N}$. Let $\mathcal{U}(\star, \star)$ be an arithmetic circuit over domain $\Sigma^c \times \{0,1\}^m$ with some $m, c \in \mathbb{N}$. We say that $\mathcal{U}$ has c input types and assign every wire $w \in \mathcal{U}$ with a type $\boldsymbol{t}_w \in \mathbb{N}^{c+1}$ through the following recursively defined function $\boldsymbol{t}_w = \mathsf{type}(\mathcal{U}, w)$.*

- ***Base Case:*** *If w is the i^{th} input wire,*
  - *If $i \in [(k-1)\ell + 1, k\ell]$ for some $k \in [c]$ (meaning that w describes x^k), assign type $\boldsymbol{t}_w = \boldsymbol{1}_k$ (a vector with one at position k and zeros everywhere else).*
  - *If $i \in [c\ell + 1, c\ell + m]$ (meaning that w describes the circuit $\mathcal{C}$), assign type $\boldsymbol{t}_w = \boldsymbol{1}_{c+1}$.*
- ***Recursion:*** *If w is the output wire of gate g with input wires u, v of types $\boldsymbol{t}_u = \mathsf{type}(\mathcal{U}, u)$ and $\boldsymbol{t}_v = \mathsf{type}(\mathcal{U}, v)$ respectively.*

- *If g is an addition/subtraction gate and $\boldsymbol{t}_u = \boldsymbol{t}_v$, then assign type $\boldsymbol{t}_w = \boldsymbol{t}_u$.*
- *Otherwise (i.e., g is a multiplication gate or $\boldsymbol{t}_u \neq \boldsymbol{t}_v$), then assign $\boldsymbol{t}_w = \boldsymbol{t}_u + \boldsymbol{t}_v$.*

Definition 2 (Type Degree). *We define the type degree of the following objects:*

- *The type degree of a wire w of $\mathcal{U}$ is $\mathsf{tdeg}(\mathcal{U}, w) = |\mathsf{type}(\mathcal{U}, w)|_1$.*
- *The type degree of $\mathcal{U}$ is $\mathsf{tdeg}(\mathcal{U}) = \max_{w \in \mathcal{U}}(\mathsf{tdeg}(\mathcal{U}, w))$.*

The fact that the seed class of [Lin16] has constant input types and constant type degree is summarized in the following lemma.

Lemma 1 (The Special-Purpose Circuits Have Constant Type-Degree, [Lin16], Lemma 5). *The class of special purpose circuits $\{\mathcal{P}_\lambda^{T,n}\}$ has universal arithmetic circuits $\{U_\lambda\}$ of constant $c^{T,n}$ input-types, constant type degree $\mathsf{tdeg}^{T,n}$, and size $u(1^\lambda, n, \log T)$, for a universal polynomial u independent of T, n.*

Given the above lemma, [Lin16] gives an $i\mathcal{O}$ construction in ideal graded encoding model, where the oracle has degree $d = O(\mathsf{tdeg} + c)$, i.e. a constant. In our work, we give an $i\mathcal{O}$ construction that improves upon the construction of [Lin16] in two ways. We show that our construction is secure against all known attacks including annihilation attacks [MSZ16] and has only a polynomial noise growth as mentioned in Section 1.

4 Construction of the Obfuscator

In this section, we give our $i\mathcal{O}$ construction for the seed class of circuits from [Lin16] that is secure in our weak multilinear map model. We build on the construction from [Lin16] in composite-order ideal graded encoding model, and use new ideas to achieve security in the weak multilinear map model and constant noise growth.

[Lin16] gives a construction for obfuscation which obfuscates circuits with multi-bit outputs directly. The reason stated in [Lin16] is the following: Direct conversion from multi-bit output circuit C to single-bit output circuit $\bar{C}$ by taking an additional input for index of output wire as $\bar{C}(x, i) = C(x)_i$ might not preserve constant type degree of C (crucial for the construction). This is because the multiplexer circuit that chooses the i^{th} output depending on input i might not have constant type degree. In this work, we observe that obfuscating one-bit output circuits suffices if we give out a different obfuscation per-output bit of the circuit. Let $C_i = C(x)_i$ denote the circuit that that outputs the i^{th} bit of the circuit. We can easily construct C_i by removing some gates of C that do not contribute to i^{th} output wire. This transformation cannot increase the type-degree. Hence, for simplicity, we only focus on obfuscating Lin's seed class of circuits for one bit output.

Construction Overview. Let $\mathcal{C}$ be a circuit with universal arithmetic circuit $\mathcal{U}(x, \mathcal{C})$ that has a single bit output. Recall that $x \in \Sigma^c$ and each input wire takes in a symbol from Σ as input. At a high level, in Lin's [Lin16] construction, for every input wire for possible symbol, encodings are given per symbol bit. Also, encodings are given per description bit of the circuit $\mathcal{C}$. Then given an input x, an evaluator can simply pick the encodings corresponding to $x, \mathcal{C}$, and homomorphically evaluate $\mathcal{U}$ on encodings of x and $\mathcal{C}$ to obtain an encoding of $\mathcal{U}(x, \mathcal{C})$, which can then be zero-tested. This basic idea is not secure and [AB15, Lin16] need a composite ring with many primes to make it secure. The actual computation happens in one of the sub-rings and computation on random elements happen in other sub-rings to protect against the input-mixing attacks as well as low-level zeroes. Moreover, they also need a carefully chosen straddling sets (to encode the elements) to ensure input consistency.

In our case, the goal is to prove security against post-zeroizing computations as well. For this, as already mentioned in the introduction, the main idea is the following: We add one more sub-ring where a PRF is computed.[18] The key idea is that though the PRF is being computed in only one of the sub-rings, after zero-testing it would yield a random ring element in all the sub-rings, in particular, a random element in $R \bmod \mathcal{I}$, where $\mathcal{I} = \langle \boldsymbol{g} \rangle$ (c.f. Section 2 for definitions of R and $\mathcal{I}$). So we start by computing one-bit PRF on input x in one of the sub-rings.

To argue security, intuitively, we would need that the PRF output has sufficient min-entropy. But since PRF has one-bit output similar to $\mathcal{U}$, it does not have enough min-entropy. So the final idea is to compute multiple PRFs in parallel and combine them to get a ring element. In doing this, we need to use an unbounded addition gate and need to take care that it does not blow up the type-degree of the computation. For this, we ensure that all PRF outputs before being added are at the same type-degree or straddling set and also have the same El-Gamal randomness of the encodings. Recall that [AB15, Lin16] use El-Gamal encodings to encode elements and to be able to add two encodings without increasing the type-degree, it is important that they have the same randomness r term.

Finally, the straddling sets are matrices of polynomial size and as already pointed out in Section 2.2 if we pick a $\boldsymbol{z}_{ij}$ corresponding to each entry in the matrix, the noise of encodings would be too high. We explain in Section 5, how we change the GGH instantiation of Section 2 to control the noise growth.

4.1 Setting and Parameters

Consider an arbitrary circuit class $\{\mathcal{C}_\lambda\}$ with universal circuits $\{\mathcal{U}_\lambda\}$. The universal circuit $\mathcal{U} = \mathcal{U}_\lambda$ has the following parameters:

- alphabet Σ with $|\Sigma|$ symbols, each of length ℓ, both $|\Sigma|$ and ℓ being $\mathsf{poly}(\lambda)$,
- domain $\Sigma^c \times \{0,1\}^m$, that is, every circuit $\mathcal{C} \in \mathcal{C}_\lambda$ has input $x = x^1, \cdots, x^c$ where $x^k \in \Sigma$ for every $k \in [c]$ and can be described by an m-bit string,

[18] We note that such a PRF can be computed using constant input types and constant type degree. See more details in Appendix D.2.

– degree of the universal circuit $d = \deg(\mathcal{U})$,
– an output wire o, denote by $\mathbf{t} = \mathsf{type}(\mathcal{U}, o) \in \mathbb{N}^{c+1}$ the type of the output wire (see Definition 1). Note that $\mathbf{t}[k]$ denotes the type degree of x^k in the output wire.

Recall the ring $R = \mathbb{Z}[X]/(X^n + 1)$ defined in the composite-order GGH graded encoding scheme (see Section 2.1). In our construction, we will use PRF circuits with 1 bit output. Our construction uses n independent PRFs, where n is the dimension of R. Let $\mathcal{C}^{\mathsf{PRF}^t} : \Sigma^c \to \{0,1\}$ be a PRF for all $t \in [n]$. As already shown in [Lin16], these circuits also satisfy the constraints for constant input types and constant type degree as the seed-class (c.f. Lemma 14). More precisely, $\mathcal{C}^{\mathsf{PRF}^t}(x)$ is a circuit computing 1 bit for every $t \in [n]$, and each circuit can be described by an m-bit string.

Encoding Levels: We specify the levels used in the $i\mathcal{O}$ construction in Figure 1. All levels are represented as a $(|\Sigma| + 1) \times (c + 2)$ matrix over $\mathbb{N}$.

Notation: In the following construction, we abuse the notations 0/1 to refer to both bits 0/1 and ring elements $\mathbf{0/1}$.

$$\forall k \in [c], s \in \Sigma, \quad \mathbf{v}_s^k = \left[\begin{array}{ccccc|c} & & (k) & & & \\ & 0 \cdots & 0 & \cdots & 0 & 0 \\ & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots \\ (s) & 0 \cdots & 1 & \cdots & 0 & 0 \\ & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots \\ & 0 \cdots & 0 & \cdots & 0 & 0 \\ \hline & 0 \cdots & 0 & \cdots & 0 & 0 \end{array}\right] \qquad \forall k \in [c], s \in \Sigma, \quad \hat{\mathbf{v}}_s^k = \left[\begin{array}{ccccc|c} & & (k) & & & \\ & 0 \cdots & \mathbf{t}[k] & \cdots & 0 & 0 \\ & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots \\ (s) & 0 \cdots & 0 & \cdots & 0 & 0 \\ & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots \\ & 0 \cdots & \mathbf{t}[k] & \cdots & 0 & 0 \\ \hline & 0 \cdots & 1 & \cdots & 0 & 0 \end{array}\right]$$

$$\mathbf{v}^* = \left[\begin{array}{ccc|c} 0 \cdots & 0 & & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 \cdots & 0 & & 0 \\ \hline 0 \cdots & 0 & & 1 \end{array}\right] \quad \mathbf{v}^{c+1} = \left[\begin{array}{ccccc|c} 0 \cdots & 0 & 1 & & & 0 \\ \vdots & \ddots & \vdots & & & \vdots \\ 0 \cdots & 0 & 1 & & & 0 \\ \hline 0 \cdots & 0 & 0 & & & 0 \end{array}\right] \quad \widetilde{\mathbf{v}} = \left[\begin{array}{ccccc|c} 0 \cdots & 0 & 0 & & & 0 \\ \vdots & \ddots & \vdots & & & \vdots \\ 0 \cdots & 0 & 0 & & & 0 \\ \hline 0 \cdots & 0 & 1 & & & 0 \end{array}\right] \quad \bar{\mathbf{v}} = \left[\begin{array}{ccc|c} 0 \cdots & 0 & & 1 \\ \vdots & \ddots & \vdots & \vdots \\ 0 \cdots & 0 & & 1 \\ \hline 0 \cdots & 0 & & 0 \end{array}\right]$$

$$\mathbf{v}_{zt} = \left[\begin{array}{ccccc|c} \mathbf{t}[1] & \cdots & \mathbf{t}[c] & \mathbf{t}[c+1]+1 & & 1 \\ \vdots & \ddots & \vdots & & & \vdots \\ \mathbf{t}[1] & \cdots & \mathbf{t}[c] & \mathbf{t}[c+1]+1 & & 1 \\ \hline 1 & \cdots & 1 & 1 & & D \end{array}\right] \quad \text{where } D = d + c + 2$$

Figure 1. Levels used in the obfuscation.

4.2 Our Obfuscator

Input: Security parameter λ, program description $\mathcal{C} \in \mathcal{C}_\lambda$.
Output: Obfuscated program with the same functionality as $\mathcal{C}$.
Algorithm: Our obfuscator is as follows:

1. Instantiate a $(c+3)$-composite graded encoding scheme $(\mathsf{params}, \mathsf{sparams}, \boldsymbol{p}_{zt}) \leftarrow \mathsf{InstGen}(1^\lambda, 1^{c+3}, \mathbf{v}_{zt})$, and receive a ring $\mathcal{R} \cong \mathcal{R}_1 \times \mathcal{R}_2 \times \cdots \times \mathcal{R}_{c+3}$. Note that $\mathcal{R}_i \cong \mathbb{Z}_{p_i}$ for some prime p_i for all $i \in [c+3]$. Hence, given $\mathsf{sparams}$ it is easy to sample a uniform element in any of the sub-rings.
2. Compute encoding $Z^* = [w^*]_{\mathbf{v}^*}$ for $w^* = (1, 1, 1, \rho_1^*, \cdots, \rho_c^*)$ where $\rho_k^* \xleftarrow{\$} \mathcal{R}_{k+3}$ for $\forall k \in [c]$.
3. **Encode the input symbol.** For $\forall k \in [c]$, encode the k-th input symbol:
   - For every symbol $s \in \Sigma$, sample $r_s^k \xleftarrow{\$} \mathcal{R}$ and compute $R_s^k = \left[r_s^k\right]_{\mathbf{v}_s^k}$.
   - For $\forall j \in [\ell]$, sample $y_j^k \xleftarrow{\$} \mathcal{R}_1$.
   - For every symbol $s \in \Sigma$, and every j-th bit s_j, compute encoding $Z_{s,j}^k = \left[r_s^k \cdot w_{s,j}^k\right]_{\mathbf{v}_s^k + \mathbf{v}^*}$ for $w_{s,j}^k = \left(y_j^k, s_j, s_j, \rho_{s,j,1}^k, \cdots, \rho_{s,j,c}^k\right)$ where $\left(\rho_{s,j,1}^k, \cdots, \rho_{s,j,c}^k\right) \xleftarrow{\$} \mathcal{R}_4 \times \cdots \times \mathcal{R}_{c+3}$.
4. **Encode the circuit and PRFs.** Compute encoding $R^{c+1} = \left[r^{c+1}\right]_{\mathbf{v}^{c+1}}$ where $r^{c+1} \xleftarrow{\$} \mathcal{R}$. For $\forall t \in [n]$, generate the following encodings for program description: We will encode the circuit $\mathcal{C}$ in $\mathcal{R}_2$ and circuit $\mathcal{C}^{\mathsf{PRF}^t}$ in $\mathcal{R}_3$.
   (a) For $\forall j \in [m]$, compute encoding $Z_{t,j}^{c+1} = \left[r^{c+1} \cdot w_{t,j}^{c+1}\right]_{\mathbf{v}^{c+1} + \mathbf{v}^*}$ for $w_{t,j}^{c+1} = \left(y_{t,j}^{c+1}, \mathcal{C}_j, \mathcal{C}_j^{\mathsf{PRF}^t}, \rho_{t,j,1}^{c+1}, \cdots, \rho_{t,j,c}^{c+1}\right)$ where $y_{t,j}^{c+1} \xleftarrow{\$} \mathcal{R}_1$ and $\left(\rho_{t,j,1}^{c+1}, \cdots, \rho_{t,j,c}^{c+1}\right) \xleftarrow{\$} \mathcal{R}_4 \times \cdots \times \mathcal{R}_{c+3}$.
   (b) Compute encoding $Z_{t,m+1}^{c+1} = \left[r^{c+1} \cdot w_{t,m+1}^{c+1}\right]_{\mathbf{v}^{c+1} + \mathbf{v}^*}$ for $w_{t,m+1}^{c+1} = \left(y_{t,m+1}^{c+1}, 1, \mathbf{e}^t, \rho_{t,m+1,1}^{c+1}, \cdots, \rho_{t,m+1,c}^{c+1}\right)$ where $\mathbf{e}^t$ is an element in the ring R of the composite order GGH graded encoding scheme (see Section 2.1),[19] $y_{t,m+1}^{c+1} \xleftarrow{\$} \mathcal{R}_1$ and $\left(\rho_{t,m+1,1}^{c+1}, \cdots, \rho_{t,m+1,c}^{c+1}\right) \xleftarrow{\$} \mathcal{R}_4 \times \cdots \times \mathcal{R}_{c+3}$. During computation, these encodings will be used to combine the n one-bit PRF computations into a ring element.
5. Encode c elements for the purpose of canceling ρ in the last c slots: For $\forall k \in [c]$ sample $\hat{w}^k = \left(\hat{y}^k, \hat{\beta}^k, \hat{\alpha}^k, \hat{\rho}_1^k, \cdots, \hat{\rho}_c^k\right)$ where $\hat{y}^k, \hat{\beta}^k, \hat{\alpha}^k, \hat{\rho}_1^k, \cdots, \rho_c^k$ are all uniformly random except that $\hat{\rho}_k^k = 0$ and generate the following encodings:
   For all $s \in \Sigma$, sample $\hat{r}_s^k \xleftarrow{\$} \mathcal{R}$ and compute encodings $\hat{R}_s^k = \left[\hat{r}_s^k\right]_{\hat{\mathbf{v}}_s^k}$ and $\hat{Z}_s^k = \left[\hat{r}_s^k \cdot \hat{w}^k\right]_{\hat{\mathbf{v}}_s^k + \mathbf{v}^*}$.

[19] The values of $\mathbf{e}^t$ will be specified later in the proof of Theorem 4 (Appendix D.5), which is crucial for proving post zeroizing security, but does not affect the correctness of the obfuscator.

For the following: denote $\hat{y} = \prod_{k=1}^{c} \hat{y}^k, \hat{\beta} = \prod_{k=1}^{c} \hat{\beta}^k, \hat{\alpha} = \prod_{k=1}^{c} \hat{\alpha}^k, \hat{w} = \prod_{k=1}^{c} \hat{w}^k = \left(\hat{y}, \hat{\beta}, \hat{\alpha}, 0, \cdots, 0\right)$.

6. Encode an element to cancel out the PRF computation in the 3^{rd} slot: Compute encodings $\widetilde{R} = [\widetilde{r}]_{\widetilde{\mathbf{v}}}$ and $\widetilde{Z} = [\widetilde{r} \cdot \widetilde{w}]_{\widetilde{\mathbf{v}}+\mathbf{v}^*}$ for $\widetilde{r} \xleftarrow{\$} \mathcal{R}$ and $\widetilde{w} = \left(\widetilde{y}, \widetilde{\beta}, 0, \widetilde{\rho}_1, \cdots, \widetilde{\rho}_c\right)$ where $\widetilde{y}, \widetilde{\beta}, \widetilde{\rho}_1, \cdots, \widetilde{\rho}_c$ are all uniformly random in respective sub-rings.
7. Encode an element for the purpose of authentication of computation: Compute encodings $\bar{R} = [\bar{r}]_{\bar{\mathbf{v}}}$ and $\bar{Z} = [\bar{r} \cdot \bar{w}]_{\bar{\mathbf{v}}+D\mathbf{v}^*}$, where $D = d + c + 2$, for $\bar{r} \xleftarrow{\$} \mathcal{R}$ and $\bar{w} = \hat{w} \cdot \widetilde{w} \cdot (\bar{y}, n, 0, 0, \cdots, 0)$, where $\bar{y} = \sum_{t=1}^{n} \left(\bar{y}_t \cdot y_{t,m+1}^{c+1}\right)$ for $\bar{y}_t = \mathcal{U}\left(\left\{y_j^1\right\}_{j\in[\ell]}, \cdots, \left\{y_j^c\right\}_{j\in[\ell]}, \left\{y_{t,j}^{c+1}\right\}_{j\in[m]}\right)$.

8. **The obfuscation.** The obfuscated program consists of the following:
   - The evaluation parameters $\mathsf{params}, \boldsymbol{p}_{zt}$.
   - The encoding Z^*.
   - For $\forall k \in [c], \forall s \in \Sigma$, the encodings $R_s^k, \hat{R}_s^k, \hat{Z}_s^k$, and for $\forall j \in [\ell]$, $Z_{s,j}^k$.
   - R^{c+1}, and for $\forall t \in [n], \forall j \in [m+1]$, $Z_{t,j}^{c+1}$.
   - The encodings $\widetilde{R}, \widetilde{Z}, \bar{R}, \bar{Z}$.

Efficiency: It is easy to see that the number of encodings in the obfuscated program is bounded by $\mathsf{poly}(1^\lambda, S(\lambda))$, where $S(\lambda)$ is the size of $\mathcal{U}_\lambda$. The size of each encoding and ℓ_1-norm of $\mathbf{v}_{zt}$ are also bounded by $\mathsf{poly}(1^\lambda, S(\lambda))$. It is easy to check that all poly above are fixed universal polynomials. Therefore the size of obfuscation is bounded by $p(1^\lambda, S(\lambda))$ for a universal polynomial, which satisfies the universal efficiency requirement in Section 3.

4.3 Evaluating an Obfuscated Program and Correctness

To evaluate the program on an input $x = x^1, \ldots, x^c \in \Sigma^c$, we will use following encodings:

$$\left\{\left(R_{x^k}^k, Z_{x^k,j}^k\right)\right\}_{k\in[c],j\in[\ell]}, \quad \left\{\left(R^{c+1}, Z_{t,j}^{c+1}\right)\right\}_{t\in[n],j\in[m+1]},$$
$$\left\{\left(\hat{R}_{x^k}^k, \hat{Z}_{x^k}^k\right)\right\}_{k\in[c]}, \quad \left(\widetilde{R}, \widetilde{Z}\right), \left(\bar{R}, \bar{Z}\right), Z^*.$$

We in-line the analysis of *correctness* in the description of the evaluation below.

1. For every $t \in [n]$, do the following:
   (a) Consider the encodings $\left(R_{x^k}^k, Z_{x^k,j}^k\right)$ for $k \in [c], j \in [\ell]$, and $\left(R^{c+1}, Z_{t,j}^{c+1}\right)$ for $j \in [m]$. Apply the circuit $\mathcal{U}$ on these pairs of encodings. More specifically, we recursively associate every wire α in $\mathcal{U}$ with a pair of encodings $\left(R_\alpha = [r_\alpha]_{\mathbf{v}_\alpha}, Z_\alpha = [r_\alpha \cdot w_\alpha]_{\mathbf{v}_\alpha + d_\alpha \mathbf{v}^*}\right)$ in El-Gamal form as follows:

Input: Two pairs of encodings $\left(R_\alpha = [r_\alpha]_{\mathbf{v}_\alpha}, Z_\alpha = [r_\alpha \cdot w_\alpha]_{\mathbf{v}_\alpha + d_\alpha \mathbf{v}^*}\right)$, $\left(R_\beta = [r_\beta]_{\mathbf{v}_\beta}, Z_\beta = [r_\beta \cdot w_\beta]_{\mathbf{v}_\beta + d_\beta \mathbf{v}^*}\right)$, encoding $Z^* = [w^*]_{\mathbf{v}^*}$, and an operator op,

Output: A pair of encodings $\left(R_\sigma = [r_\sigma]_{\mathbf{v}_\sigma}, Z_\sigma = [r_\sigma \cdot w_\sigma]_{\mathbf{v}_\sigma + d_\sigma \mathbf{v}^*}\right)$

Algorithm:

i. Permute the operands to ensure that $\delta = d_\beta - d_\alpha \geq 0$.

ii. Consider the operator op:

- **Multiplication:** If $\mathsf{op} = \times$, then $R_\sigma = R_\alpha \times R_\beta$ and $Z_\sigma = Z_\alpha \times Z_\beta$. ($r_\sigma = r_\alpha \cdot r_\beta$, $\mathbf{v}_\sigma = \mathbf{v}_\alpha + \mathbf{v}_\beta$, and $d_\sigma = d_\alpha + d_\beta$.)
- **Addition/Subtraction:** If $\mathsf{op} = +/-$ and $\mathbf{v}_\alpha \neq \mathbf{v}_\beta$, then $R_\sigma = R_\alpha \times R_\beta$ and $Z_\sigma = Z_\alpha \times R_\beta \times (Z^*)^\delta + / - Z_\beta \times R_\alpha$. ($r_\sigma = r_\alpha \cdot r_\beta$, $\mathbf{v}_\sigma = \mathbf{v}_\alpha + \mathbf{v}_\beta$, and $d_\sigma = d_\beta$.)
- **Constrained Addition/Subtraction:** If $\mathsf{op} = +/-$ and $\mathbf{v}_\alpha = \mathbf{v}_\beta = \mathbf{v}$ (by induction it is guaranteed that $r_\alpha = r_\beta = r$), then $R_\sigma = R_\alpha$ and $Z_\sigma = Z_\alpha \times (Z^*)^\delta + / - Z_\beta$. ($r_\sigma = r$, $\mathbf{v}_\sigma = \mathbf{v}$, and $d_\sigma = d_\beta$.)

Figure 2. Computation over El-Gamal encodings

- **Base Case:** For every $k \in [c]$ and every $j \in [\ell]$, the j^{th} input wire of x^k is associated with pair $\left(R^k_{x^k}, Z^k_{x^k,j}\right)$. For every $j \in [m]$, the j^{th} program bit is associated with $\left(R^{c+1}, Z^{c+1}_{t,j}\right)$.
- **Recursion:** For every gate $g \in \mathcal{U}$ with input wires α, β and output wire σ, apply the computation as described in Figure 2, over the encodings $Z^*, (R_\alpha, Z_\alpha), (R_\beta, Z_\beta)$ and the operator of g.

A pair of encodings for the output wire o is obtained:

$$\left(R_{\mathcal{U}} = [r_{\mathcal{U}}]_{\mathbf{v}_{\mathcal{U}}}, Z_{t,\mathcal{U}} = [r_{\mathcal{U}} \cdot w_{t,\mathcal{U}}]_{\mathbf{v}_{\mathcal{U}} + d\mathbf{v}^*}\right),$$

where (let $\mathbf{1}$ denote an all-one vector, $\mathbf{0}$ an all-zero vector, and let $\mathbf{1}_i$ denote a vector with one at position i and zeros everywhere else)

$$\begin{aligned}
\mathbf{v}_{\mathcal{U}} &= \left[\begin{array}{ccccc|c} \mathbf{t}[1] \cdot \mathbf{1}_{x^1} & \cdots & \mathbf{t}[c] \cdot \mathbf{1}_{x^c} & \mathbf{t}[c+1] \cdot \mathbf{1} & \mathbf{0} \\ 0 & \cdots & 0 & 0 & 0 \end{array}\right], \\
w_{t,\mathcal{U}} &= \left(\mathcal{U}\left(\left\{y^1_j\right\}_{j\in[\ell]}, \cdots, \left\{y^c_j\right\}_{j\in[\ell]}, \left\{y^{c+1}_{t,j}\right\}_{j\in[m]}\right), \mathcal{U}(x, \mathcal{C}), \mathcal{U}\left(x, \mathcal{C}^{\mathsf{PRF}^t}\right),\right. \\
&\quad \left.\star, \cdots, \star\right) \\
&= \left(\bar{y}_t, \mathcal{C}(x), \mathcal{C}^{\mathsf{PRF}^t}(x), \star, \cdots, \star\right).
\end{aligned}$$

In the above, the values denoted by $\star$ do not matter for correctness, and hence are not mentioned explicitly.

(b) Take the product of $(R_{\mathcal{U}}, Z_{t,\mathcal{U}})$ with $\left(R^{c+1}, Z^{c+1}_{t,m+1}\right)$ and obtain a pair of encodings (computation done as in Figure 2):

$$\left(\ddot{R}_{\mathcal{U}} = [\ddot{r}_{\mathcal{U}}]_{\ddot{\mathbf{v}}_{\mathcal{U}}}, \ddot{Z}_{t,\mathcal{U}} = [\ddot{r}_{\mathcal{U}} \cdot \ddot{w}_{t,\mathcal{U}}]_{\ddot{\mathbf{v}}_{\mathcal{U}}+(d+1)\mathbf{v}^*}\right), \text{ where}$$
$$\ddot{\mathbf{v}}_{\mathcal{U}} = \left[\begin{array}{ccccc|c} \mathbf{t}[1]\cdot\mathbf{1}_{x^1} & \cdots & \mathbf{t}[c]\cdot\mathbf{1}_{x^c} & (\mathbf{t}[c+1]+1)\cdot\mathbf{1} & & \mathbf{0} \\ 0 & \cdots & 0 & 0 & & 0 \end{array}\right],$$
$$\ddot{w}_{t,\mathcal{U}} = w_{t,\mathcal{U}} \cdot w^{c+1}_{t,m+1} = \left(\bar{y}_t \cdot y^{c+1}_{t,m+1}, \mathcal{C}(x), \mathcal{C}^{\mathsf{PRF}^t}(x) \cdot \mathbf{e}^t, \star, \cdots, \star\right).$$

Remark 1. Note that our construction ensures that $\left(\ddot{R}_{\mathcal{U}}, \ddot{Z}_{t,\mathcal{U}}\right)$ has the same level and same $\ddot{r}_{\mathcal{U}}$ for every $t \in [n]$. This is crucial to do the next step of addition of n terms using constrained addition. This ensures that the addition does not grow the levels of multilinearity needed.

2. Take the sum of $\left\{\left(\ddot{R}_{\mathcal{U}}, \ddot{Z}_{t,\mathcal{U}}\right)\right\}_{t\in[n]}$ and obtain a pair of encodings:

$$\left(\ddot{R}_{\mathcal{U}} = [\ddot{r}_{\mathcal{U}}]_{\ddot{\mathbf{v}}_{\mathcal{U}}}, \ddot{Z}_{\mathcal{U}} = [\ddot{r}_{\mathcal{U}} \cdot \ddot{w}_{\mathcal{U}}]_{\ddot{\mathbf{v}}_{\mathcal{U}}+(d+1)\mathbf{v}^*}\right), \text{ where}$$
$$\ddot{w}_{\mathcal{U}} = \sum_{t=1}^{n} \ddot{w}_{t,\mathcal{U}} = \left(\bar{y}, n \cdot \mathcal{C}(x), \mathcal{C}^{\mathsf{PRF}}(x), \star, \cdots, \star\right),$$

where $\mathcal{C}^{\mathsf{PRF}}(x) = \sum_{t\in[n]} \mathbf{e}^t \mathcal{C}^{\mathsf{PRF}^t}(x)$.

3. Take the product of $\left(\ddot{R}_{\mathcal{U}}, \ddot{Z}_{\mathcal{U}}\right)$ with the product of $\left\{\left(\hat{R}^k_{x^k}, \hat{Z}^k_{x^k}\right)\right\}_{k\in[c]}$ and obtain a pair:

$$\left(\hat{R}_{\mathcal{U}} = [\hat{r}_{\mathcal{U}}]_{\hat{\mathbf{v}}_{\mathcal{U}}}, \hat{Z}_{\mathcal{U}} = [\hat{r}_{\mathcal{U}} \cdot \hat{w}_{\mathcal{U}}]_{\hat{\mathbf{v}}_{\mathcal{U}}+(d+1+c)\mathbf{v}^*}\right), \text{ where}$$
$$\hat{\mathbf{v}}_{\mathcal{U}} = \left[\begin{array}{ccccc|c} \mathbf{t}[1]\cdot\mathbf{1} & \cdots & \mathbf{t}[c]\cdot\mathbf{1} & (\mathbf{t}[c+1]+1)\cdot\mathbf{1} & & \mathbf{0} \\ 1 & \cdots & 1 & 0 & & 0 \end{array}\right],$$
$$\hat{w}_{\mathcal{U}} = \hat{w} \cdot \ddot{w}_{\mathcal{U}} = \left(\hat{y}\bar{y}, \hat{\beta}n \cdot \mathcal{C}(x), \hat{\alpha} \cdot \mathcal{C}^{\mathsf{PRF}}(x), 0, \cdots, 0\right).$$

4. Take the product of $\left(\hat{R}_{\mathcal{U}}, \hat{Z}_{\mathcal{U}}\right)$ with $\left(\widetilde{R}, \widetilde{Z}\right)$ and obtain a pair:

$$\left(\widetilde{R}_{\mathcal{U}} = [\widetilde{r}_{\mathcal{U}}]_{\widetilde{\mathbf{v}}_{\mathcal{U}}}, \widetilde{Z}_{\mathcal{U}} = [\widetilde{r}_{\mathcal{U}} \cdot \widetilde{w}_{\mathcal{U}}]_{\widetilde{\mathbf{v}}_{\mathcal{U}}+D\mathbf{v}^*}\right), \text{ where}$$
$$\widetilde{\mathbf{v}}_{\mathcal{U}} = \left[\begin{array}{ccccc|c} \mathbf{t}[1]\cdot\mathbf{1} & \cdots & \mathbf{t}[c]\cdot\mathbf{1} & (\mathbf{t}[c+1]+1)\cdot\mathbf{1} & & \mathbf{0} \\ 1 & \cdots & 1 & 1 & & 0 \end{array}\right],$$
$$\widetilde{w}_{\mathcal{U}} = \widetilde{w} \cdot \hat{w}_{\mathcal{U}} = \left(\widetilde{y}\hat{y}\bar{y}, \widetilde{\beta}\hat{\beta}n \cdot \mathcal{C}(x), 0, 0, \cdots, 0\right).$$

5. Subtract the pair $\left(\bar{R}, \bar{Z}\right)$ from $\left(\widetilde{R}_{\mathcal{U}}, \widetilde{Z}_{\mathcal{U}}\right)$ and obtain the pair:

$$\begin{aligned}
&\left(\bar{R}_{\mathcal{U}} = [\bar{r}_{\mathcal{U}}]_{\bar{\mathbf{v}}_{\mathcal{U}}}, \bar{Z}_{\mathcal{U}} = [\bar{r}_{\mathcal{U}} \cdot \bar{w}_{\mathcal{U}}]_{\bar{\mathbf{v}}_{\mathcal{U}} + D\mathbf{v}^*}\right), \text{ where} \\
&\bar{\mathbf{v}}_{\mathcal{U}} = \left[\begin{array}{ccccc|c} \mathbf{t}[1] \cdot \mathbf{1} & \cdots & \mathbf{t}[c] \cdot \mathbf{1} & (\mathbf{t}[c+1]+1) \cdot \mathbf{1} & \mathbf{1} & \\ 1 & \cdots & 1 & 1 & & 0 \end{array}\right], \\
&\bar{w}_{\mathcal{U}} = \left(0, \widetilde{\beta}\hat{\beta}n \cdot (\mathcal{C}(x) - 1), 0, 0, \cdots, 0\right).
\end{aligned}$$

6. Finally, apply zero testing on $\bar{Z}_{\mathcal{U}}$. If $\mathsf{isZero}(\mathsf{params}, \boldsymbol{p}_{zt}, \bar{Z}_{\mathcal{U}}) = 1$ then output 1, otherwise output 0.
   As analyzed above, in an honest evaluation, $\bar{Z}_{\mathcal{U}}$ is an encoding of 0 under $\mathbf{v}_{zt}$ iff $\mathcal{C}(x) = 1$ with high probability over choice of $\widetilde{\beta}, \hat{\beta}$. Hence the correctness of the evaluation procedure follows.

Security. We prove *security* of our obfuscator in the weak multilinear map model in Appendix D. We describe the weak multilinear map model formally in Appendix C. For a detailed explanation of how this model captures all known vulnerabilities of GGH multilinear maps, see [MSZ16].

In Section 5, we describe our modification of GGH instantiation for our obfuscation scheme that achieves the desired noise growth and hence, a $\mathsf{poly}(\lambda)$ modulus q.

5 Modifying GGH to Obtain Polynomial Modulus q

In this section, we will provide two possible modifications to the sampling procedure of GGH scheme described in Section 2. We can then obtain obfuscation with low noise by instantiating our scheme (in Section 4) with either of these modified GGH multilinear maps. The modifications are specific to our obfuscation scheme and may not work in general.

Obtaining polynomial sized modulus from polynomial noise. First, we show that once we have ensured that it holds for all fresh encodings as well as zero-testing parameter that the noise is at most $n^{O(1)}$, we can choose the modulus q as a fixed polynomial depending on all other parameters. Recall that the multiplicative depth of the universal circuit that is used by the obfuscator in Section 4 is a fixed constant and the number of additions is a fixed polynomial. Thus, we can conclude that also the noise levels of encodings of intermediate values is also at most $n^{O(1)}$. Finally, as the size of the term $\boldsymbol{h}$ in the zero-testing parameter $\boldsymbol{p}_{zt} = \boldsymbol{h} \cdot \boldsymbol{z}_{\mathbf{v}_{zt}}/\boldsymbol{g}$ is bounded by $O(\sqrt{q} \cdot n^{O(1)})$, applying the zero-test to top-level encodings of zero yields elements of size at most $\sqrt{q} \cdot n^{O(1)}$. It is therefore sufficient to choose q as a sufficiently large polynomial (depending on all other parameters) to ensure correctness of the zero-test.

Reducing the noise of encodings. Before, we describe the above modifications in detail, we discuss what we want to achieve more formally. Recall that

in sampling procedure described in Section 2, the noise term of the fresh encoding at level $\mathbf{v}$ is $O(n^{O(1)}\|\boldsymbol{z}_{\mathbf{v}}\| \cdot \|1/\boldsymbol{z}_{\mathbf{v}}\|)$. To achieve a construction where multilinear maps only need to support $\mathsf{poly}(\lambda)$ maximal noise, we need to ensure that the noise in all fresh encodings as well zero-testing parameter is at most $\mathsf{poly}(\lambda)$. That is, for a encoding level $\mathbf{v}$, $\|\boldsymbol{z}_{\mathbf{v}}\| \cdot \|1/\boldsymbol{z}_{\mathbf{v}}\| = O(n^{O(1)})$. However, as discussed in Section 2.2, in general the noise growth depends exponentially on $|\mathbf{v}| = \sum_{i,j} \mathbf{v}_{ij}$. While there seems little hope to improve this in the general case, the straddling sets used in the construction in Section 4 are of a very specific form. In particular, there is only a small number of levels at which elements are encoded, c.f. to Figure 1. Both of our approaches to obtain low noise would use this fact crucially.

Next, discuss the two possible modifications separately that ensure that for all levels $\mathbf{v}$ at which we encode and also for $\mathbf{v} = \mathbf{v}_{zt}$ that $\|\boldsymbol{z}_{\mathbf{v}}\| \cdot \|1/\boldsymbol{z}_{\mathbf{v}}\| \leq n^{O(1)}$.

The Conservative Option. Here, we observe that if all components $\boldsymbol{z}_{ij}$ are chosen uniformly at random in $\mathbb{R}_q^\times$, then each $\boldsymbol{z}_{\mathbf{v}} \in \mathbb{R}_q^\times$ from the table in Figure 1 is also distributed uniformly at random. To see this, not that in every level $\mathbf{v}$ in Figure 1, one of the entries $\mathbf{v}_{ij}$ of $\mathbf{v}$ is 1, including the zero-testing level $\mathbf{v}_{zt}$. This means that $\boldsymbol{z}_{\mathbf{v}}$ is of the form $\boldsymbol{z}_{ij} \cdot \boldsymbol{z}^\star$, where $\boldsymbol{z}^\star$ is independent of $\boldsymbol{z}_{ij}$. Therefore $\boldsymbol{z}_{\mathbf{v}}$ is uniform in $R_q^\times$. We can conclude by Lemma 8 that $\|1/\boldsymbol{z}_{\mathbf{v}}\| \leq n^2/q$, except with probability $2/n$. Also, it is easy to see that by uniformity condition $\|\boldsymbol{z}\| \leq \sqrt{n}q$.

Next, we apply a union bound over all the levels at which fresh encodings are generated in obfuscation scheme. The number of levels L at which we need to encode is upper bounded by $O(c \cdot |\Sigma|)$, both c and Σ depend only on the circuit we obfuscate, but not on the degree n of the ring R (or to put it differently, we choose the ring R at last). Now a union bound yields that $\|\boldsymbol{z}_{\mathbf{v}}\| \cdot \|1/\boldsymbol{z}_{\mathbf{v}}\| \leq n^{O(1)}$ holds for all levels $\mathbf{v}$ at which we encode, except with probability $2L/n$. This means that if we guarantee that n is bigger than (say) $4L$, then the above holds with probability at least $1/2$. This probability can in fact be made a constant arbitrarily close to 1. This means we need to reject our (entire) choices of the $\boldsymbol{z}_{ij}$ in expectation 2 times until we found a suitable choice. From a security standpoint, this means that the good choices of the $\boldsymbol{z}_{ij}$ are very dense in the space of all possible choices, meaning that we do not weaken the multilinear maps. An obvious drawback of this option is the rather large choice of n, which depends on the size of the circuit being obfuscated, though it is still just polynomial in the security parameter λ.

The Aggressive option. We will now discuss a more aggressive sampling procedure that avoids the union bound above, thereby avoiding the dependence of n on the number of levels L at which we encode.

In this approach, we partition the levels into *independent* and *dependent* levels. In a nutshell, all levels $\mathbf{v}_s^k$, $\mathbf{v}^\star$ and $\tilde{\mathbf{v}}$ (these are the matrices with only a single 1 component) will be considered independent whereas the levels $\hat{\mathbf{v}}_s^k$, $\mathbf{v}^{c+1}$, $\bar{\mathbf{v}}$ and $\mathbf{v}_{zt}$ will be considered dependent. Sampling the $\boldsymbol{z}_{\mathbf{v}}$ for the independent levels $\mathbf{v}$ is easy, because they only rely $\boldsymbol{z}_{ij}$. Also, notice that for the dependent levels $\mathbf{v}^{c+1}$, $\tilde{\mathbf{v}}$ and $\bar{\mathbf{v}}$ we can sample the $\boldsymbol{z}_{\mathbf{v}^{c+1}}$, $\boldsymbol{z}_{\tilde{\mathbf{v}}}$ and $\boldsymbol{z}_{\bar{\mathbf{v}}}$ directly, since their

components are never used individually. Dealing with the dependent levels $\hat{\mathbf{v}}_s^k$ and $\mathbf{v}_{zt}$ will require more work. First consider the a new dependent level $\hat{\mathbf{v}}^k$ given by

$$\hat{\mathbf{v}}^k = \left[\begin{array}{ccccc|c} & & (k) & & & \\ 0 & \cdots & \mathbf{t}[k] & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & \mathbf{t}[k] & \cdots & 0 & 0 \\ \hline 0 & \cdots & 1 & \cdots & 0 & 0 \end{array}\right]$$

i.e. the k-th column of $\hat{\mathbf{v}}^k$ is $t[k]$ everywhere but in the last component. We can express $\hat{\mathbf{v}}_s^k$ as $\hat{\mathbf{v}}_s^k = \hat{\mathbf{v}}^k - t[k] \cdot \mathbf{v}_s^k$ (again c.f. to Figure 1) and therefore

$$\boldsymbol{z}_{\hat{\mathbf{v}}_s^k} = \boldsymbol{z}_{\hat{\mathbf{v}}^k} \cdot ([\boldsymbol{z}_{\mathbf{v}_s^k}^{-1}]_q)^{t[k]},$$

where we compute the inversion in R_q but the product in R. If we ensure that both $1/\boldsymbol{z}_{\hat{\mathbf{v}}^k}$ and $1/[\boldsymbol{z}_{\mathbf{v}_s^k}^{-1}]_q$ are short in K, say at most n^2/q then we can conclude that $1/\boldsymbol{z}_{\hat{\mathbf{v}}_s^k}$ is also short in K as

$$\|1/\boldsymbol{z}_{\hat{\mathbf{v}}_s^k}\| \leq n^{\frac{t[k]+1}{2}} \cdot \|1/\boldsymbol{z}_{\hat{\mathbf{v}}^k}\| \cdot \|1/[\boldsymbol{z}_{\mathbf{v}_s^k}^{-1}]_q\|^{t[k]} \leq n^{O(1)}/q^{t[k]+1}$$

where we recall that $t[k]$ is a constant. This yields that $\|\boldsymbol{z}_{\hat{\mathbf{v}}_s^k}\| \cdot \|1/\boldsymbol{z}_{\hat{\mathbf{v}}_s^k}\| \leq n^{O(1)}$ as desired.

Finally, notice that we can sample $\boldsymbol{z}_{\hat{\mathbf{v}}^k}$ directly without hurting consistency, as the $\boldsymbol{z}_{s'k}$ term corresponding to the last row of the k-th column is never used individually. In other words, we can first sample $\boldsymbol{z}_{\hat{\mathbf{v}}^k}$ and then set

$$\boldsymbol{z}_{s'k} = \boldsymbol{z}_{\hat{\mathbf{v}}^k} \cdot \left(\prod_{s \in \Sigma} \boldsymbol{z}_{\mathbf{v}_s^k}^{t[k]}\right)^{-1}.$$

Notice that we don't have any guarantee that $1/\boldsymbol{z}_{s'k}$ is short in K, but that does not pose a problem as $\boldsymbol{z}_{s'k}$ is never used individually by the encoding procedure. Finally, notice that we can express $\boldsymbol{z}_{\mathbf{v}_{zt}}$ as

$$\boldsymbol{z}_{\mathbf{v}_{zt}} = \boldsymbol{z}_{\hat{\mathbf{v}}^1} \ldots \boldsymbol{z}_{\hat{\mathbf{v}}^c} \cdot \boldsymbol{z}_{\mathbf{v}^{c+1}}^{t[c+1]+1} \cdot \boldsymbol{z}_{\tilde{\mathbf{v}}} \cdot \boldsymbol{z}_{\bar{\mathbf{v}}} \cdot \boldsymbol{z}_{\mathbf{v}^\star}^D.$$

We can conclude that $\|1/\boldsymbol{z}_{\mathbf{v}_{zt}}\| \leq n^{O(c+t[c+1]+D)}/q^{c+t[c+1]+D+3}$, and therefore $\|\boldsymbol{z}_{\mathbf{v}_{zt}}\| \cdot \|1/\boldsymbol{z}_{\mathbf{v}_{zt}}\| \leq n^{O(1)}$. This is because $c, t[c+1], D$ are all constants.

Thus we modify our instance generation algorithm as follows. Instead of sampling all $\boldsymbol{z}_{ij}$ individually, we sample the following denominators directly, under the constraint that the size of their inverse in K is bounded by n^2/q: $\boldsymbol{z}_{\hat{\mathbf{v}}^k}$ for $k \in [c]$, $\boldsymbol{z}_{\mathbf{v}_s^k}$ for $s \in \Sigma$ and $k \in [c]$, $\boldsymbol{z}_{\mathbf{v}^{c+1}}$, $\boldsymbol{z}_{\tilde{\mathbf{v}}}$, $\boldsymbol{z}_{\bar{\mathbf{v}}}$ and $\boldsymbol{z}_{\mathbf{v}^\star}$. We additionally impose the constraint that $1/[\boldsymbol{z}_{\mathbf{v}_s^k}^{-1}]_q$ is small in K, where $[\boldsymbol{z}_{\mathbf{v}_s^k}^{-1}]_q$ is the inverse of $\boldsymbol{z}_{\mathbf{v}_s^k}$ in R_q. Imposing the two constraints $\|1/\boldsymbol{z}_{\mathbf{v}_s^k}\| \leq n^2/q$ and $\|1/[\boldsymbol{z}_{\mathbf{v}_s^k}^{-1}]_q\| \leq n^2/q$ does not change the rejection probability significantly: If $\boldsymbol{z}$ is uniform in the

unit group $R_q^\times$, then $\boldsymbol{z}^{-1}$ is also uniform in $R_q^\times$. For a uniform $\boldsymbol{z}$ in $\mathbb{R}_q^\times$ it holds that $\|1/\boldsymbol{z}\| \leq n^2/q$, except with probability $2/n$ (Lemma 8). Consequently, by a union bound we have that both $\|1/\boldsymbol{z}\| \leq n^2/q$ and $\|1/[\boldsymbol{z}^{-1}]_q\| \leq n^2/q$, except with probability $4/n$. Concluding, we have ensured that it holds for all levels $\mathbf{v}$ at which we encode and also for $\mathbf{v} = \mathbf{v}_{zt}$ that $\|\boldsymbol{z}_\mathbf{v}\| \cdot \|1/\boldsymbol{z}_\mathbf{v}\| \leq n^{O(1)}$.

6 Discussion of Modifications and Cryptanalytical Perspective

In this section, we discuss our modifications of the GGH multilinear maps from a cryptanalytic standpoint. Specifically, we make the following two changes to the GGH multilinear maps constructions:

1. We use generators $\boldsymbol{g}$ of composite structure rather than a prime. Furthermore, we choose specific Lagrange Coefficients.
2. We make a modification to the sampling procedure of the asymmetric multilinear maps (namely, the sampling of the denominators $\boldsymbol{z}_{ij}$).

6.1 Already Known Attacks on GGH

We start by considering the already known attacks on GGH and how they are relevant to our construction. In short, the basic cryptanalytic survey of [GGH13a] still holds in our setting. Furthermore, our understanding of the attacks on GGH is improved by the zeroizing attacks [GGH13a, CHL+15, CGH+15, HJ16, CLLT16] and the annihilation attacks [MSZ16].

The most potent attacks that GGH found against their construction are the averaging attacks [GS02, NR06, DN12]. To avoid these attacks, they suggested special sampling procedures (inspired by the GPV sampling procedure). We use the same sampling procedures in our construction. Therefore, we expect that our construction will resist averaging attacks.

No obfuscation construction provides any encoding of zero below the highest level. Therefore, zeroizing attacks do not apply to our constructions. Note that all obfuscations constructions in the literature use this guideline.

Furthermore, our construction is hardened against annihilation attacks via self-fortification. Therefore, it also resists the annihilation attacks analogous to the obfuscation construction of Garg et al. [GMM+16].

Two very recent works [ABD16, KF16] showed that GGH multilinear maps can be attacked for rather broad choices of cyclotomic rings if the modulus q is super-polynomially larger than the length of the error term in the encodings. Obfuscation constructions in the literature can be made to resist these attacks by choosing the dimension of the lattice carefully. Our instantiation is resilient against these attacks as we choose our modulus q to be only polynomially larger than the error terms in the encodings. This is an added bonus of our construction.

6.2 Composite-Order GGH Multilinear Maps

Recall that we choose the ideal generator $\boldsymbol{g}$ as a composite (which is a product of several large primes) instead of as a prime element in R to provide several independent *slots* in the plaintext space, which are required for the [AB15] circuit obfuscation technique. Note that in our construction the generator $\boldsymbol{g}$ itself (or, even a small multiple of it) is never exposed, as it immediately compromise security, even in the case of a prime $\boldsymbol{g}$.

Further, choosing $\boldsymbol{g}$ as a composite merely constitutes a change of the distribution from which $\boldsymbol{g}$ is chosen. While [GGH13a] propose to choose $\boldsymbol{g}$ from a discrete gaussian distribution, there is no supporting evidence such as a worst-to-average case reduction that this choice is favorable over other distributions. In fact, there are no known lattice attacks that can distinguish encodings with plaintext space generated by composite generators from encodings with plaintext space generated by prime generator, or attacks that utilize the specific distribution from which $\boldsymbol{g}$ is chosen. Generally, lattice attacks (e.g. [LLL82, SE93, GS02, GNR10]) rely only on geometric properties rather than distributional properties and solve worst case (rather than average case problems).[20]

Analogous to GGH, one new line of attacks that we investigate for our scheme is when the attacker can obtain a rather small elements in $\langle \boldsymbol{g}_i \rangle$ for some i, where $\boldsymbol{g} = \prod_i \boldsymbol{g}_i$. This would break our scheme. However, we do not know any methods for obtaining such small elements in our construction.

Next, we note that our scheme (like all other obfuscation schemes in the literature) does not hide relatively large (specifically, of size $\sqrt{q}$) elements in $\langle \boldsymbol{h} \rangle$. Such elements can be obtained by performing zero-testing operation at top level encodings of zero. However, such large elements in $\langle \boldsymbol{h} \rangle$, or elements of this size in any $\langle \boldsymbol{g}_i \rangle$ are not useful for any of the attacks. In light of the above discussion, the most potent new attacks could arise if an adversary can obtain elements in one of the ideals $\langle \boldsymbol{g}_i \rangle$ depending on the circuit that is obfuscated. Such an attack would be a generalization of the annihilation attacks by Miles et al [MSZ16].

To avoid this line of attacks, we define our weak multilinear map model to be a strengthening of the one considered by Miles et al. [MSZ16]. In our model, we declare an attack successful if the attacker can obtain any (small or large) element in any of the ideals $\langle \boldsymbol{g}_i \rangle$.

Recall that self-fortification in our composite order setting works by computing a PRF in a separate slot. Due to our specific choice of the Lagrange basis $(\gamma_i)_i$ (see Section 2.1), we can show that computing a PRF in a single slot is sufficient to randomize the output of the zero test with *well spread* entropy (see

[20] To the best of our knowledge, the only attacks against lattice based schemes using specific distributions are attacks against signature schemes [NR06, DN12], where the *shape* of distribution of signatures is *learned*. However, in this case the distribution is exposed to the adversary, whereas in our case (as well as in the case of all lattice based encryption schemes) the adversary does not directly obtain samples from the error distribution.

Thm 4). However, the specific choice of Lagrange Coefficients affects only the elements that are encoded and not the randomness chosen to encode them. Based on this argument, we do not expect the choice of the Lagrange Coefficients to affect the security of our construction.

To conclude, from our current understanding, any distributions which is both *short* and has high entropy is a legitimate choice for the distribution of $\boldsymbol{g}$. Furthermore, there is currently no reason to believe that the composite structure provides any handle in either breaking the underlying multilinear maps or distinguishing obfuscated circuits.

6.3 Modified Asymmetric Multilinear Maps

Recall that the safeguard of GGH against averaging attacks requires sampling of noise terms from fractional ideals. Hence, to achieve low noise, it is necessary that for every level $\mathbf{v}$ at which we encode and the zero-testing level that $\|\boldsymbol{z}_{\mathbf{v}}\| \cdot \|1/\boldsymbol{z}_{\mathbf{v}}\| \leq n^{O(1)}$. In the previous section, we provided two techniques for sampling for $\boldsymbol{z}_{ij}$ that ensured this condition.

The conservative option, choosing ring R with sufficiently large degree n, ensures that this condition is met with probability close to 1 if the $\boldsymbol{z}_{ij}$ are chosen uniformly at random. Thus, we do not have to change the sampling procedure of the asymmetric GGH multilinear maps; the condition we need holds with high probability.

The more aggressive option, choosing he degree of the ring R independent of the size of the circuit we obfuscate, achieves better efficiency at the expense of enforcing a correlation between the denominators $\boldsymbol{z}_{ij}$. However, we note that there is no known distinguishing attack that can exploit correlations among the $\boldsymbol{z}_{ij}$'s directly. Finally, the distribution of the $\boldsymbol{z}_{ij}$ only depends on the straddling set, not the circuit and it is the same for two functionally equivalent circuits. The indirect attacks could exploit a variant of the NTRU scheme that uses these correlated $\boldsymbol{z}_{ij}$ values. However, we do not know any attacks in this setting. Note that GGH multilinear maps already use $\boldsymbol{z}_{ij}$ that are correlated (though more weakly than our choice) and no attacks are known to benefit from those correlations either.

References

AB15. Benny Applebaum and Zvika Brakerski. Obfuscating circuits via composite-order graded encoding. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015, Part II*, volume 9015 of *LNCS*, pages 528–556. Springer, Heidelberg, March 2015.

ABD16. Martin R. Albrecht, Shi Bai, and Léo Ducas. A subfield lattice attack on overstretched NTRU assumptions - cryptanalysis of some FHE and graded encoding schemes. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part I*, volume 9814 of *LNCS*, pages 153–178. Springer, Heidelberg, August 2016.

AGHS12. Shweta Agrawal, Craig Gentry, Shai Halevi, and Amit Sahai. Discrete gaussian leftover hash lemma over infinite domains. Cryptology ePrint Archive, Report 2012/714, 2012.

AGIS14. Prabhanjan Vijendra Ananth, Divya Gupta, Yuval Ishai, and Amit Sahai. Optimizing obfuscation: Avoiding Barrington's theorem. In Gail-Joon Ahn, Moti Yung, and Ninghui Li, editors, *ACM CCS 14*, pages 646–658. ACM Press, November 2014.

AR05. Dorit Aharonov and Oded Regev. Lattice problems in np cap conp. *J. ACM*, 52(5):749–765, 2005.

BD16. Zvika Brakerski and Or Dagmi. Shorter circuit obfuscation in challenging security models. *IACR Cryptology ePrint Archive*, 2016:418, 2016.

BF11. Dan Boneh and David Mandell Freeman. Linearly homomorphic signatures over binary fields and new tools for lattice-based signatures. In Dario Catalano, Nelly Fazio, Rosario Gennaro, and Antonio Nicolosi, editors, *PKC 2011*, volume 6571 of *LNCS*, pages 1–16. Springer, Heidelberg, March 2011.

BGK+14. Boaz Barak, Sanjam Garg, Yael Tauman Kalai, Omer Paneth, and Amit Sahai. Protecting obfuscation against algebraic attacks. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 221–238. Springer, Heidelberg, May 2014.

BMSZ16. Saikrishna Badrinarayanan, Eric Miles, Amit Sahai, and Mark Zhandry. Post-zeroizing obfuscation: New mathematical tools, and the case of evasive circuits. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 764–791. Springer, Heidelberg, May 2016.

BR14. Zvika Brakerski and Guy N. Rothblum. Virtual black-box obfuscation for all circuits via generic graded encoding. In Yehuda Lindell, editor, *TCC 2014*, volume 8349 of *LNCS*, pages 1–25. Springer, Heidelberg, February 2014.

BS02. Dan Boneh and Alice Silverberg. Applications of multilinear forms to cryptography. Cryptology ePrint Archive, Report 2002/080, 2002. http://eprint.iacr.org/2002/080.

CGH+15. Jean-Sébastien Coron, Craig Gentry, Shai Halevi, Tancrède 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 Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part I*, volume 9215 of *LNCS*, pages 247–266. Springer, Heidelberg, August 2015.

CHL+15. Jung Hee Cheon, Kyoohyung Han, Changmin Lee, Hansol Ryu, and Damien Stehlé. Cryptanalysis of the multilinear map over the integers. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part I*, volume 9056 of *LNCS*, pages 3–12. Springer, Heidelberg, April 2015.

CLLT16. Jean-Sébastien Coron, Moon Sung Lee, Tancrède Lepoint, and Mehdi Tibouchi. Cryptanalysis of GGH15 multilinear maps. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part II*, volume 9815 of *LNCS*, pages 607–628. Springer, Heidelberg, August 2016.

CLT13. Jean-Sébastien Coron, Tancrède Lepoint, and Mehdi Tibouchi. Practical multilinear maps over the integers. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 476–493. Springer, Heidelberg, August 2013.

CLT15. Jean-Sébastien Coron, Tancrède Lepoint, and Mehdi Tibouchi. New multilinear maps over the integers. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part I*, volume 9215 of *LNCS*, pages 267–286. Springer, Heidelberg, August 2015.

CLTV15. Ran Canetti, Huijia Lin, Stefano Tessaro, and Vinod Vaikuntanathan. Obfuscation of probabilistic circuits and applications. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015, Part II*, volume 9015 of *LNCS*, pages 468–497. Springer, Heidelberg, March 2015.

DN12. Léo Ducas and Phong Q. Nguyen. Learning a zonotope and more: Cryptanalysis of NTRUSign countermeasures. In Xiaoyun Wang and Kazue Sako, editors, *ASIACRYPT 2012*, volume 7658 of *LNCS*, pages 433–450. Springer, Heidelberg, December 2012.

Gar15. Sanjam Garg. *Candidate Multilinear Maps*. Association for Computing Machinery and Morgan & Claypool, New York, NY, USA, 2015.

GGH12. Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. Cryptology ePrint Archive, Report 2012/610, 2012. http://eprint.iacr.org/2012/610.

GGH13a. Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In Thomas Johansson and Phong Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 1–17. Springer, Heidelberg, May 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 *54th FOCS*, pages 40–49. IEEE Computer Society Press, October 2013.

GGH15. Craig Gentry, Sergey Gorbunov, and Shai Halevi. Graph-induced multilinear maps from lattices. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015, Part II*, volume 9015 of *LNCS*, pages 498–527. Springer, Heidelberg, March 2015.

GLSW15. Craig Gentry, Allison Bishop Lewko, Amit Sahai, and Brent Waters. Indistinguishability obfuscation from the multilinear subgroup elimination assumption. In Venkatesan Guruswami, editor, *56th FOCS*, pages 151–170. IEEE Computer Society Press, October 2015.

GLW14a. Craig Gentry, Allison B. Lewko, and Brent Waters. Witness encryption from instance independent assumptions. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 426–443. Springer, Heidelberg, August 2014.

GLW14b. Craig Gentry, Allison Bishop Lewko, and Brent Waters. Witness encryption from instance independent assumptions. Cryptology ePrint Archive, Report 2014/273, 2014. http://eprint.iacr.org/2014/273.

GMM+16. Sanjam Garg, Eric Miles, Pratyay Mukherjee, Amit Sahai, Akshayaram Srinivasan, and Mark Zhandry. Secure obfuscation in a weak multilinear map model. In *TCC 2016-B, Part I*, LNCS. Springer, Heidelberg, November 2016.

GNR10. Nicolas Gama, Phong Q. Nguyen, and Oded Regev. Lattice enumeration using extreme pruning. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 257–278. Springer, Heidelberg, May 2010.

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

GS02. Craig Gentry and Michael Szydlo. Cryptanalysis of the revised NTRU signature scheme. In Lars R. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 299–320. Springer, Heidelberg, April / May 2002.

HJ16. Yupu Hu and Huiwen Jia. Cryptanalysis of GGH map. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part I*, volume 9665 of *LNCS*, pages 537–565. Springer, Heidelberg, May 2016.

KF16. Paul Kirchner and Pierre-Alain Fouque. Comparison between subfield and straightforward attacks on NTRU. *IACR Cryptology ePrint Archive*, 2016:717, 2016.

Lin16. Huijia Lin. Indistinguishability obfuscation from constant-degree graded encoding schemes. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part I*, volume 9665 of *LNCS*, pages 28–57. Springer, Heidelberg, May 2016.

LLL82. A. K. Lenstra, H. W. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, 1982.

LV16. Huijia Lin and Vinod Vaikuntanathan. Indistinguishability obfuscation from ddh-like assumptions on constant-degree graded encodings. Cryptology ePrint Archive, Report 2016/795 (Accepted to FOCS 2016), 2016. http://eprint.iacr.org/2016/795.

MR04. Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on Gaussian measures. In *45th FOCS*, pages 372–381. IEEE Computer Society Press, October 2004.

MSZ16. Eric Miles, Amit Sahai, and Mark Zhandry. Annihilation attacks for multilinear maps: Cryptanalysis of indistinguishability obfuscation over GGH13. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part II*, volume 9815 of *LNCS*, pages 629–658. Springer, Heidelberg, August 2016.

NR06. Phong Q. Nguyen and Oded Regev. Learning a parallelepiped: Cryptanalysis of GGH and NTRU signatures. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 271–288. Springer, Heidelberg, May / June 2006.

PST14. Rafael Pass, Karn Seth, and Sidharth Telang. Indistinguishability obfuscation from semantically-secure multilinear encodings. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 500–517. Springer, Heidelberg, August 2014.

Reg04. Oded Regev. New lattice-based cryptographic constructions. *J. ACM*, 51(6):899–942, 2004.

SE93. C. P. Schnorr and M. Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. In *Math. Programming*, pages 181–191, 1993.

SS11. Damien Stehlé and Ron Steinfeld. Making NTRU as secure as worst-case problems over ideal lattices. In Kenneth G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 27–47. Springer, Heidelberg, May 2011.

Zim15. Joe Zimmerman. How to obfuscate programs directly. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*, pages 439–467. Springer, Heidelberg, April 2015.

A Preliminaries

Notations. The natural security parameter throughout this paper is λ, and all other quantities are implicitly assumed to be functions of λ. We use standard big-O notation to classify the growth of functions, and say that $f(\lambda) = \tilde{O}(g(\lambda))$ if $f(\lambda) = O(g(\lambda) \cdot log^c\ \lambda)$ for some fixed constant c. We let $\mathsf{poly}(\lambda)$ denote an unspecified function $f(\lambda) = O(\lambda^c)$ for some constant c. A *negligible* function, denoted generically by $\mathsf{negl}(\lambda)$, is an $f(\lambda)$ such that $f(\lambda) = o(\lambda^{-c})$ for every fixed constant c. We say that a function is *overwhelming* if it is $1 - \mathsf{negl}(\lambda)$.

The *statistical distance* between two distributions X and Y over a domain D is defined to be $\frac{1}{2}\sum_{d\in D} |\Pr[X = d] - \Pr[Y = d]|$. We say that two ensembles of distributions $\{X_\lambda\}$ and $\{Y_\lambda\}$ are *statistically indistinguishable* if for every λ the statistical distance between X_λ and Y_λ is negligible in λ.

Two ensembles of distributions $\{X_\lambda\}$ and $\{Y_\lambda\}$ are *computationally indistinguishable* if for every probabilistic poly-time non-uniform (in λ) machine $\mathcal{A}$, $|\Pr[\mathcal{A}(1^\lambda, X_\lambda) = 1] - \Pr[\mathcal{A}(1^\lambda, Y_\lambda) = 1]|$ is negligible in λ. The definition is extended to non-uniform families of poly-sized circuits in the standard way.

Lemma 2 (Schwarz-Zippel Lemma). *Let $\mathbb{F}$ be a finite field and let $p \in \mathbb{F}[x_1, \ldots, x_n]$ be a multivariate polynomial of degree at most d. Further let $X_1, \ldots, X_n$ be independently distributed random variables on $\mathbb{F}$ such that $H_\infty(X_i) \geq k$ for all i. Then it holds that*

$$\Pr[p(X_1, \ldots, X_n) = 0] \leq \frac{d}{2^k},$$

where the probability runs over the random choices of $X_1, \ldots, X_n$.

A.1 Lattices

We denote set of complex number by $\mathbb{C}$, real numbers by $\mathbb{R}$, the rationals by $\mathbb{Q}$ and the integers by $\mathbb{Z}$. For a positive integer n, $[n]$ denotes the set $\{1, \ldots, n\}$. By convention, vectors are assumed to be in column form and are written using bold lower-case letters, e.g. $\boldsymbol{x}$. The ith component of $\boldsymbol{x}$ will be denoted by x_i. We will use $\boldsymbol{x}^T$ to denotes the transpose of $\boldsymbol{x}$. For a vector $\boldsymbol{x}$ in $\mathbb{R}^n$ or $\mathbb{C}^n$ and $p \in [1, \infty]$, we define the ℓ_p norm as $\|\boldsymbol{x}\|_p = \left(\sum_{i\in[n]} |x_i|^p\right)^{1/p}$ where $p < \infty$, and $\|\boldsymbol{x}\|_\infty = \max_{i\in[n]} |x_i|$ where $p = \infty$. Whenever p is not specified, $\|\boldsymbol{x}\|$ is assumed to represent the ℓ_2 norm (also referred to as the Euclidean norm).

Matrices are written as bold capital letters, e.g. $\boldsymbol{X}$, and the ith column vector of a matrix $\boldsymbol{X}$ is denoted $\boldsymbol{x}_i$. Finally we will denote the transpose and the inverse (if it exists) of a matrix $\boldsymbol{X}$ with $\boldsymbol{X}^T$ and $\boldsymbol{X}^{-1}$ respectively.

A lattice Λ is an additive discrete sub-group of $\mathbb{R}^n$, i.e., it is a subset $\Lambda \subset \mathbb{R}^n$ satisfying the following properties:

(subgroup) Λ is closed under addition and subtraction,
(discrete) there is a real $\varepsilon > 0$ such that any two distinct lattice points $\boldsymbol{x} \neq \boldsymbol{y} \in \Lambda$ are at distance at least $\|\boldsymbol{x} - \boldsymbol{y}\| \geq \varepsilon$.

Let $\boldsymbol{B} = \{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_k\} \subset \mathbb{R}^n$ consist of k linearly independent vectors in $\mathbb{R}^n$. The lattice generated by the $\boldsymbol{B}$ is the set

$$\mathcal{L}(\boldsymbol{B}) = \{\boldsymbol{B}\boldsymbol{z} = \sum_{i=1}^{k} z_i \boldsymbol{b}_i : \boldsymbol{z} \in \mathbb{Z}^k\},$$

of all the integer linear combinations of the columns of $\boldsymbol{B}$. The matrix $\boldsymbol{B}$ is called a *basis* for the lattice $\mathcal{L}(\boldsymbol{B})$. The integers n and k are called the *dimension* and *rank* of the lattice. If $n = k$ then $\mathcal{L}(\boldsymbol{B})$ is called a *full-rank* lattice. We will only be concerned with full-rank lattices, hence unless otherwise mentioned we will assume that the lattice considered is full-rank.

For lattices $\Lambda' \subseteq \Lambda$, the quotient group Λ/Λ' (also written as $\Lambda \bmod \Lambda'$) is well-defined as the additive group of distinct *cosets* $\boldsymbol{v} + \Lambda'$ for $\boldsymbol{v} \in \Lambda$, with addition of cosets defined in the usual way.

A.2 Gaussians on Lattices

Review of Gaussian measure over lattices presented here follows the development by prior works [Reg04, AR05, MR04, GPV08, AGHS12]. For any real $s > 0$, define the (spherical) *Gaussian function* $\rho_s : \mathbb{R}^n \to (0, 1]$ with[21] parameter s as:

$$\forall \boldsymbol{x} \in \mathbb{R}^n, \rho_s(\boldsymbol{x}) = \mathsf{exp}(-\pi\langle \boldsymbol{x}, \boldsymbol{x}\rangle/s^2) = \mathsf{exp}(-\pi\|\boldsymbol{x}\|^2/s^2).$$

For any real $s > 0$, and n-dimensional lattice Λ, define the (spherical) *discrete Gaussian distribution* over Λ as:

$$\forall \boldsymbol{x} \in \Lambda, D_{\Lambda,s}(\boldsymbol{x}) = \frac{\rho_s(\boldsymbol{x})}{\rho_s(\Lambda)}.$$

Gentry, Peikert and Vaikuntanathan provide an efficient algorithm to sample from a discrete gaussian given a *good* basis.

Theorem 2 ([GPV08] Theorem 3.3). *There exists an efficient algorithm* $\mathsf{SampleD}$*, which given a basis* $\|\boldsymbol{B}\|$ *of an* n*-dimensional lattice* Λ *and a parameter* $s \geq \|\boldsymbol{B}\| \cdot \omega(\sqrt{\log(n)})$ *efficiently samples a distribution within negligible distance of* $D_{\Lambda,s}$

Smoothing Parameter. Micciancio and Regev [MR04] introduced a lattice quantity called the *smoothing parameter*, and related it other lattice parameters.

Definition 3 (Smoothing Parameter, [MR04, Definition 3.1]). *For an* n*-dimensional lattice* Λ*, and positive real* $\varepsilon > 0$*, we define its* smoothing parameter *denoted* $\eta_\varepsilon(\Lambda)$*, to be the smallest* s *such that* $\rho_{1/s}(\Lambda^* \setminus \{\mathbf{0}\}) \leq \varepsilon$*.*

[21] The Gaussian function can be defined more generally as being centered around a specific vector $\boldsymbol{c}$ instead of $\mathbf{0}$ as done here. The simpler definition considered here suffices for our purposes.

Intuitively, for a small enough ε, the number $\eta_\varepsilon(\Lambda)$ is sufficiently larger than a fundamental parallelepiped of Λ so that sampling from the corresponding Gaussian "wipes out the internal structure" of Λ. The following Lemma 3 formally provide this claim. Finally Lemma 4 provides bounds on the length of a vector sampled from a Gaussian.

Lemma 3 ([GPV08, Corollary 2.8]). *Let Λ, Λ' be n-dimensional lattices, with $\Lambda' \subseteq \Lambda$. Then for any $\varepsilon \in (0, \frac{1}{2})$, any $s \geq \eta_\varepsilon(\Lambda')$, the distribution of $(D_{\Lambda,s} \pmod{\Lambda'})$ is within a statistical distance at most 2ε of uniform over $(\Lambda \pmod{\Lambda'})$.*

Lemma 4 ([MR04, Lemma 4.4] and [BF11, Proposition 4.7]). *For any n-dimensional lattice Λ, and $s \geq \eta_\varepsilon(\Lambda)$ for some negligible ε, then for any constant $\delta > 0$ we have*

$$\Pr_{\boldsymbol{x} \leftarrow D_{\Lambda,s}}\left[(1-\delta)s\sqrt{\frac{n}{2\pi}} \leq \|\boldsymbol{x}\| \leq (1+\delta)s\sqrt{\frac{n}{2\pi}}\right] \geq 1 - \mathsf{negl}(n).$$

Invertibility of ring elements. Let R denote the $2n^{th}$ cyclotomic ring and let R_q denote R/qR for a prime q. We note that R_q is also a ring and not all elements in it are invertible. Let $R_q^\times$ denote the set of elements in R_q that are invertible. We next provide a lemma of Stehlé and Steinfeld that points out that a (large enough) random element is R_q is also in $R_q^\times$ with large probability.

Lemma 5 ([SS11, Lemma 4.1]). *Let $n \geq 8$ be a power of 2 such that $X^n + 1$ splits into n linear factors modulo $q \geq 5$. Let $\sigma \geq \sqrt{n \ln(2n(1+1/\delta))/\pi} \cdot q^{1/n}$, for an arbitrary $\delta \in (0, 1/2)$. Then*

$$\Pr_{f \leftarrow D_{Z^n,\sigma}}[f \pmod{q} \notin R_q^\times] \leq n(1/q + 2\delta).$$

We will use the following simple lemma to lower bound the length of the shortest vector in an ideal lattice via its norm.

Lemma 6. *Let $\mathcal{I} \subset \mathbb{R}$ be an ideal lattice. Then it holds that $\lambda_1(\mathcal{I}) \geq \sqrt{n} \cdot N(\mathcal{I})^{1/n}$.*

Babai's Roundoff Algorithm We will need to compute short representatives of residual classes $x \bmod \mathcal{I} \in R/\mathcal{I}$ for ideals $\mathcal{I} = \langle \boldsymbol{g} \rangle$. A simple algorithm for this task is Babai's roundoff algorithm. Given an $x \in R$, we can find a small representative $\hat{x}$ of $x \bmod \mathcal{I}$ by computing

$$\hat{x} = x - \lfloor x \cdot \boldsymbol{g}^{-1} \rceil \cdot \boldsymbol{g},$$

where the $\lfloor \cdot \rceil$ operation round each component to the nearest integer. Clearly, it holds that $\hat{x} \equiv x \bmod \mathcal{I}$ and

$$\begin{aligned}\|\hat{x}\| &= \|x - \lfloor x \cdot \boldsymbol{g}^{-1} \rceil \cdot \boldsymbol{g}\| = \|(x \cdot \boldsymbol{g}^{-1} - \lfloor x \cdot \boldsymbol{g}^{-1} \rceil) \cdot \boldsymbol{g}\| \\ &\leq \sqrt{n} \cdot \|x \cdot \boldsymbol{g}^{-1} - \lfloor x \cdot \boldsymbol{g}^{-1} \rceil\| \cdot \|\boldsymbol{g}\| \leq \frac{n}{2} \cdot \|\boldsymbol{g}\|,\end{aligned}$$

as $x \cdot \boldsymbol{g}^{-1} - \lfloor x \cdot \boldsymbol{g}^{-1} \rceil \in K$ is a field element with coefficients of size at most $1/2$. Therefore, if $\boldsymbol{g}$ is short then so is $\hat{x}$.

B Preliminaries for our modified GGH construction

Most parts of this section are taken verbatim from [Gar15]. We keep this part for completeness.

B.1 Number Fields, Ring of Integers and Ideal Lattices

A *number field* can be defined as field extension $K = \mathbb{Q}(\zeta)$ obtained by adjoining an abstract element ζ to the field of rationals, where ζ satisfies the relation $f(\zeta) = 0$ for some irreducible polynomial $f(X) \in \mathbb{Q}[X]$, which is a monic (a polynomial whose leading coefficient is 1) polynomial without loss of generality. The polynomial $f(X)$ is called the *minimal polynomial* of ζ, and the *degree* n of the number field is the degree of f. Because $f(\zeta) = 0$, the number field K can be seen as an n-dimensional vector space over $\mathbb{Q}$ with basis $\{1, \zeta, \ldots, \zeta^{n-1}\}$. Associating ζ with indeterminate X yields an isomorphism between K and $\mathbb{Q}[X]/f(X)$.

The ring of integers $\mathcal{O}_K$, of a number field K of degree n, is a free $\mathbb{Z}$-module of rank n, i.e., the set of all $\mathbb{Z}$-linear combinations of some *integral basis* $\{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\} \subset \mathcal{O}_K$. Such a set is called an *integral basis*, and it is also a $\mathbb{Q}$-basis for K.

The case of Cyclotomic Number Fields. Let $\zeta_m = e^{2\pi\sqrt{-1}/m} \in \mathbb{C}$ denote a *primitive* m-th root of unity. (Recall that an mth root of unity is said to be a *primitive* root if it is not a kth root for some $0 < k < m$.) The m-th *cyclotomic polynomial*, denote by $\Phi_m(X)$, is defined as the product

$$\Phi_m(X) = \prod_{k \in \mathbb{Z}_m^*} (X - \zeta_m^k).$$

Observe that the values ζ^k run over all the primitive m^{th} roots of unity in $\mathbb{C}$, thus $\Phi_m(X)$ has degree $n = \varphi(m)$, where $\varphi(m)$ denotes the *Euler's totient* or *phi function*. Recall that if m is a positive integer, then $\varphi(m)$ is the number of integers in the set $\{1, 2, \ldots, m\}$ that are relatively prime to m.

The cyclotomic polynomial $\Phi_m(X)$ may be computed by (exactly) dividing $X^n - 1$ by the cyclotomic polynomials of the proper divisors of n previously computed recursively (setting, $\Phi_1(X) = X - 1$) by the same method:

$$\Phi_m(X) = \frac{X^m - 1}{\prod_{\substack{d|m \\ d<m}} \Phi_d(X)}.$$

We will be most interested in the case when $m \geq 2$ is a power of 2 in which case $\Phi_m(X) = X^{m/2} + 1$. The mth *cyclotomic field* $\mathbb{Q}(\zeta_m)$ (with $m > 2$) is obtained by adjoining ζ_m to $\mathbb{Q}$. The ring of integers in $\mathbb{Q}(\zeta_m)$ is $\mathbb{Z}(\zeta_m)$. This ring $\mathbb{Z}(\zeta_m)$ is called the *cyclotomic ring*.

Coefficient Embedding. There is also a *coefficient embedding* $\tau : K \to \mathbb{Q}^n$. As mentioned earlier, since $f(\zeta) = 0$, there is an isomorphism between $\mathbb{Q}[X] \pmod{f(X)}$ and K given by $X \to \zeta$. So, K can be represented as a n-dimensional vector space over $\mathbb{Q}$ using the *power basis* $\{1, \zeta, \ldots, \zeta^{n-1}\}$, and τ maps an element of K to its associated coefficient vector. When identifying an element $a \in K$ as a coefficient vector, i.e., $\tau(a)$ we denote it as a boldface vector $\boldsymbol{a}$. Note that the addition of vectors is done component-wise, while the multiplication is done as polynomials modulo $f(X)$. We define the *coefficient norm* of a as the norm of the vector $\boldsymbol{a}$. Specifically, we define the ℓ_p coefficient norm of a, denoted as $\|a\|_p$ or $\|\boldsymbol{a}\|_p$ as $\left(\sum_{i\in[n]} a_i^p\right)^{\frac{1}{p}}$ for $p < \infty$, and as $\max_{i\in[n]} |a_i|$ for $p = \infty$. (As always we assume the ℓ_2 norm when p is omitted.) We will use the following lemma.

Lemma 7. *Let $K = \mathbb{Q}[X]/(X^n + 1)$, for any positive integer n. $\forall \boldsymbol{a}, \boldsymbol{b} \in K$ and $\boldsymbol{c} = \boldsymbol{a} \cdot \boldsymbol{b}$ we have that*

$$\|\boldsymbol{c}\| \leq \sqrt{n} \cdot \|\boldsymbol{a}\| \cdot \|\boldsymbol{b}\|.$$

Definition 4 (Ideal). *An* (integral) ideal $\mathcal{I} \subseteq \mathcal{O}_K$ *is a nontrivial (i.e., nonempty and nonzero*[22]*) additive subgroup that is closed under multiplication by $\mathcal{O}_K$ – that is, $r \cdot g \in \mathcal{I}$ for any $r \in \mathcal{O}_K$ and $g \in \mathcal{I}$. A* fractional ideal $\mathcal{I} \subset K$ *is a set such that $d \cdot \mathcal{I}$ is an integral ideal for some $d \in \mathcal{O}_K$. The* inverse $\mathcal{I}^{-1}$ *of an ideal $\mathcal{I}$ is the set $\{a \in K : a \cdot \mathcal{I} \subseteq \mathcal{O}_K\}$.*

Definition 5. *An ideal $\mathcal{I}$ is* principal *if $\mathcal{I} = \langle g \rangle$ for $g \in \mathcal{O}_K$ – that is, if one generator suffices.*

B.2 Technical Lemmata

Lemma 8. *Let $\boldsymbol{z} \leftarrow R_q$ be chosen uniformly at random. Then it holds that $\|1/\boldsymbol{z}\| \leq n^2/q$, except with probability at most $\frac{2}{n}$.*

Proof. In order to upper-bound the L_2 norm of the coefficient embedding, we will first upper-bound the L_∞ norm of the canonical embedding.

Let $\sigma : K \to \mathbb{C}^n$ be the canonical embedding of K into $\mathbb{C}^n$. Let $(\boldsymbol{z}_0, \ldots, \boldsymbol{z}_{n-1})$ be the coefficient representation of $\boldsymbol{z} \in R$. Each component $\sigma_j(\boldsymbol{z})$ of $\sigma(\boldsymbol{z})$ is the evaluation of $\boldsymbol{z}$ at an n-th root of unity $\xi_j \in \mathbb{C}$, i.e.

$$\sigma_j(\boldsymbol{z}) = \sum_{i=0}^{n-1} \boldsymbol{z}_i \cdot \xi_j^i.$$

As $\sigma_j : K \to \mathbb{C}$ is a field homomorphism, it holds that $\sigma_j(1/\boldsymbol{z}) = 1/\sigma_j(\boldsymbol{z})$. Thus, it holds that $\|\sigma(1/\boldsymbol{z})\|_\infty = \max_j(|\sigma_j(1/\boldsymbol{z})|) = \max_j(1/|\sigma_j(\boldsymbol{z})|)$. As we show below, in order to establish an upper-bound on $\|1/\boldsymbol{z}\|$ it is sufficient to establish a lower bound on the $|\sigma_j(\boldsymbol{z})|$.

[22] Some texts also define the trivial set $\{0\}$ as an ideal, but in this work it is more convenient to exclude it.

Note that each of $\boldsymbol{z}_0, \boldsymbol{z}_1, \ldots, \boldsymbol{z}_{n-1}$ are independently chosen. So we can fix $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_{n-1}$ to some worst case values and only consider the random choice of $\boldsymbol{z}_0$. It holds that

$$\Pr[|\sigma_j(\boldsymbol{z})| < q/n^2] = \Pr[|\boldsymbol{z}_0 + \sum_{i=1}^{n-1} \boldsymbol{z}_i \cdot \xi_j^i| < q/n^2] \leq \max_{\boldsymbol{z}^\star \in \mathbb{C}} \Pr[|\boldsymbol{z}_0 + \boldsymbol{z}^\star| < q/n^2].$$

However, since $\boldsymbol{z}_0$ is is a uniformly random integer between $-q/2$ and $q/2$ (along the real line), it holds for any choice of $\boldsymbol{z}^\star \in \mathbb{C}$ that

$$\Pr[|\boldsymbol{z}_0 + \boldsymbol{z}^\star| < q/n^2] \leq \frac{2 \cdot q/n^2}{q} = \frac{2}{n^2}.$$

A union bound yields that

$$\Pr[\exists j : |\sigma_j(\boldsymbol{z})| < q/n^2] \leq \sum_{i=0}^{n-1} \Pr[|\sigma_j(\boldsymbol{z})| < q/n^2] \leq \frac{2}{n}.$$

This in turn implies

$$\Pr[\forall j : |\sigma_j(\boldsymbol{z})| \geq q/n^2] \geq 1 - \frac{2}{n}.$$

Note that if for some $\boldsymbol{z}$ we have that $\forall j : |\sigma_j(1/\boldsymbol{z})| \leq n^2/q$, then this implies that $\|\sigma(1/\boldsymbol{z})\|_\infty \leq n^2/q$ as $\|\sigma(\boldsymbol{x})\|_\infty = \max_j |\sigma_j(\boldsymbol{x})|$. For cyclotomic fields of order power-of-two it holds that $\|\boldsymbol{x}\|_2 = \frac{1}{\sqrt{n}}\|\sigma(\boldsymbol{x})\|_2$ by Parseval's identity. Thus it holds that $\|1/\boldsymbol{z}\|_2 \leq \frac{1}{\sqrt{n}}\|\sigma(1/\boldsymbol{z})\|_2 \leq \|\sigma(1/\boldsymbol{z})\|_\infty \leq \frac{n^2}{q}$, which concludes the proof.

We will need a generalization of the Schwarz Zippel Lemma to the composite modular rings used by our graded encoding scheme.

Lemma 9. *Let R be a cyclotomic ring and let $\boldsymbol{g} = \boldsymbol{g}_1 \cdots \boldsymbol{g}_\ell \in R$ be generator of an ideal as sampled by our instance generation (i.e. the $N(\boldsymbol{g}_i)$ are large primes). Let $p \in R[x_1, \ldots, x_m]$ be an m-variate polynomial of degree d on R and let $X_1, \ldots, X_m$ be independently distributed random variables on R such that $H_\infty(X_i \bmod \langle \boldsymbol{g}_j \rangle) \geq k$ for all i and j. Then it holds that*

$$\Pr[p(X_1, \ldots, X_n) \notin (R/\langle \boldsymbol{g} \rangle)^\times] \leq \frac{d\ell}{2^k},$$

where the probability runs over the random choices of $X_1, \ldots, X_n$.

Proof. By Lemma 2 it hold that $\Pr[p(X_1, \ldots, X_n) \equiv 0 \bmod \boldsymbol{g}_j] \leq \frac{d}{2^k}$ for all i, as $R/\langle \boldsymbol{g}_j \rangle$ is a prime field of size $N(\boldsymbol{g}_i)$. A union bound yields

$$\Pr[p(X_1, \ldots, X_n) \notin R^\times] = \Pr[\exists j : p(X_1, \ldots, X_n) \equiv 0 \bmod \boldsymbol{g}_j] \leq \ell \cdot \frac{d}{2^k}$$

Recall that, in our construction, a pseudorandom function is being computed in a specific manner. For our proof, we need the output of this function to be uniform over $\mathsf{mod}\,\mathcal{I}$. We prove that under appropriate choice of parameters this is indeed true.

Lemma 10. *Let $\boldsymbol{g} = \boldsymbol{g}_1 \cdots \boldsymbol{g}_\ell$ be a generator sampled via our instance generation algorithm and let X be a random variable on R such that it holds for each pair $x_1, x_2 \in R$ in the support of X that $\|x_1 - x_2\| \leq \sqrt{n}$. Then it holds for all i that $H_\infty(X \,\mathsf{mod}\, \boldsymbol{g}_i) = H_\infty(X)$.*

Proof. The factors $\boldsymbol{g}_i$ are sampled such that $N(\boldsymbol{g}_i) = p$ for a prime p of size at least $2^{\Omega(n)}$. By Lemma 6 it holds that

$$\lambda_1(\mathcal{I}) \geq \sqrt{n} N(\mathcal{I})^{1/n} = \sqrt{n} p^{1/n} = \sqrt{n} 2^{\Omega(1)} > \sqrt{n}.$$

Let S be the support of X. It holds for all all pairs $x_1, x_2 \in S$ that $\|x_1 - x_2\| \leq \sqrt{n} < \lambda_1(\mathcal{I})$. Thus, it holds that $x_1 - x_2 \notin \mathcal{I}$ and therefore $x_1 \neq x_2 \,\mathsf{mod}\, \mathcal{I}$, i.e. the map $x \mapsto x \,\mathsf{mod}\, \mathcal{I}$ is collision free on S. We conclude that $H_\infty(X \,\mathsf{mod}\, \mathcal{I}) = H_\infty(X)$.

Observe that it holds for each pair of elements x_1, x_2 from the boolean hypercube $\{0,1\}^n \subseteq R$ that $\|x_1 - x_2\| \leq \sqrt{n}$. Thus, any distribution X on $\{0,1\}^n$ fulfills the requirements of Lemma 10 and we can conclude the following.

Corollary 1. *Let $\boldsymbol{g} = \boldsymbol{g}_1 \cdots \boldsymbol{g}_\ell$ be a generator sampled via our instance generation algorithm and let X be any distribution on $\{0,1\}^n \subseteq R$. Then it holds for all i that $H_\infty(X \,\mathsf{mod}\, \boldsymbol{g}_i) = H_\infty(X)$.*

C The Weak Multilinear Map Model

In this section, we will describe the *weak multilinear map model* put forth by Miles, Sahai and Zhandry [MSZ16]. Our model differs slightly from theirs as it is based on composite-order GGH multilinear maps. In this model all parties have access to an oracle $\mathcal{M}$ implementing the graded encoding scheme. Informally, similar to [BGK+14], $\mathcal{M}$ will allow algebraic operations to be performed on encodings through so-called "handles" on the encodings. However, unlike [BGK+14], it will also allow arbitrary polynomial computation on the ring elements produced via "successful zero-tests," through a second type of handles.[23]

Similar to [BGK+14] we start by defining the weak multilinear map system.

Definition 6 (Weak Multilinear Map System). *Let $R = \mathbb{Z}[X]/X^n + 1$ be the $2n$-th cyclotomic ring of integers and $\boldsymbol{g}_1, \boldsymbol{g}_2, \ldots \boldsymbol{g}_t \in R$ be "short" elements in the ring such that $|R/\langle \boldsymbol{g}_i \rangle|$ is a prime of size $\omega(\mathsf{poly}(\lambda))$ for all $i \in [t]$. Denote*

[23] A reader familiar with [MSZ16] can note that this step is analogous to the type-2 query in that model.

the ideal generated by each $\boldsymbol{g}_i$ *by* $\mathcal{I}_i = \langle \boldsymbol{g}_i \rangle$ *and by the product* $\boldsymbol{g} = \prod_{i \in [t]} \boldsymbol{g}_i$ *by* $\mathcal{I} = \langle \boldsymbol{g} \rangle$*. Let* $\boldsymbol{v}_{zt}$ *be the zero testing level. Then an encoding* e *of an element* $\boldsymbol{t} \in R$ *at the level* $\boldsymbol{v}$ *is denoted as* $e = (\!|\boldsymbol{t}|\!)_{\boldsymbol{v}}$*. For any such encoding* $e = (\!|\boldsymbol{t}|\!)_{\boldsymbol{v}}$*, the corresponding ring element* $\boldsymbol{t}$ *is called its representation and the set* $\boldsymbol{v}$ *its level. We define the following operations over the encodings.*

Addition: *Given two encodings* $e_1 = (\!|\boldsymbol{t}_1|\!)_{\boldsymbol{v}_1}$ *and* $e_2 = (\!|\boldsymbol{t}_2|\!)_{\boldsymbol{v}_2}$ *where* $\boldsymbol{v}_1 = \boldsymbol{v}_2$*,* $e_1 + e_2$ *is defined to be the encoding given by* $(\!|\boldsymbol{t}_1 + \boldsymbol{t}_2|\!)_{\boldsymbol{v}_1}$*. Similarly,* $e_1 - e_2$ *is defined to be the encoding given by* $(\!|\boldsymbol{t}_1 - \boldsymbol{t}_2|\!)_{\boldsymbol{v}_1}$*.*

Multiplication: *Given two encodings* $e_1 = (\!|\boldsymbol{t}_1|\!)_{\boldsymbol{v}_1}$ *and* $e_2 = (\!|\boldsymbol{t}_2|\!)_{\boldsymbol{v}_2}$*,* $e_1 \cdot e_2$ *is defined to be the element given by* $(\!|\boldsymbol{t}_1 \cdot \boldsymbol{t}_2|\!)_{\boldsymbol{v}_1 + \boldsymbol{v}_2}$*.*

Ring Multiplication: *Given a ring element* $\boldsymbol{a} \in R$ *and an encoding* $e = (\!|\boldsymbol{t}|\!)_{\boldsymbol{v}}$*, the ring multiplication* $\boldsymbol{a} \cdot e$ *is defined to be the encoding given by* $e' = (\!|\boldsymbol{a} \cdot \boldsymbol{t}|\!)_{\boldsymbol{v}}$*.*[24]

Zero Testing: *For any encoding* $e = (\!|\boldsymbol{t}|\!)_{\boldsymbol{v}_{zt}}$*, it returns 1 if and only if:*

$$\boldsymbol{t} \pmod{\mathcal{I}} = 0$$

We now proceed to describe the weak multilinear map model. Similar to [BGK+14] we consider a stateful oracle $\mathcal{M}$ mapping encodings to "generic" representations called handles. There are two types of handles that $\mathcal{M}$ generates: *encoding* handles that are corresponding to encodings and *ring* handles that are corresponding to the elements in the ring R (obtained after successful zero-tests). The handles are denoted by $\mathsf{H}_{\mathsf{Enc}}(e)$ for an encoding e and $\mathsf{H}_{\mathsf{Rng}}(\boldsymbol{a})$ for any ring element $\boldsymbol{a} \in R$. We do not specify how the handles are generated. However, we require that the value of the handles, $\mathsf{H}_{\mathsf{Enc}}(e), \mathsf{H}_{\mathsf{Rng}}(\boldsymbol{a})$ are independent of the corresponding encoding e and the corresponding ring element $\boldsymbol{a}$ respectively. The oracle maintains two tables L_{enc} and L_{rng} where L_{enc} stores encoding-handle pairs $(e, \mathsf{H}_{\mathsf{Enc}}(e))$ and similarly L_{rng} stores pairs of the form $(a, \mathsf{H}_{\mathsf{Rng}}(a))$ where $\mathsf{H}_{\mathsf{Rng}}(a)$ is a ring handle corresponding to ring element $\boldsymbol{a} \in R$. $\mathcal{M}$ provides the user with the following interfaces.

- **Initialization.** The oracle $\mathcal{M}$ is initialized with the parameters of the weak multilinear map system. Additionally, it is initialized with the encoding-handle table L_{enc} of initial encodings-handles pair and the ring-handle table L_{rng} with $\emptyset$. After $\mathcal{M}$ has been initialized, all subsequent calls to the initialization interfaces fail.
- **Algebraic operations.** Depending on the type of query it executes the following steps.
  - *Both are encoding handles:* Given two encoding handles $\mathsf{H}_{\mathsf{Enc}}(e_1), \mathsf{H}_{\mathsf{Enc}}(e_2)$ and an operation $\circ \in \{+, -, \cdot\}$, $\mathcal{M}$ first locates the relevant encodings $e_1 = (\!|\boldsymbol{t}_1|\!)_{\mathbf{v}_1}$, $e_2 = (\!|\boldsymbol{t}_2|\!)_{\mathbf{v}_2}$ in the handle table L_{enc}. If any of the input handles does not appear in the table L_{enc} (that is, if the handle was not previously generated by $\mathcal{M}$) the call to $\mathcal{M}$ fails. If the expression $e_1 \circ e_2$ is undefined (i.e., $\mathbf{v}_1 \neq \mathbf{v}_2$ for $\circ \in \{+, -\}$ or $\mathbf{v}_1 + \mathbf{v}_2 \not\leq \mathbf{v}_{zt}$ for $\circ = \cdot$)

[24] Note that we abuse the notation "$\cdot$" to denote both ring multiplication and multiplication between encodings.

the call fails. Otherwise, $\mathcal{M}$ generates a new encoding handle $\mathsf{H}_{\mathsf{Enc}}(e')$ for $e' = e_1 \circ e_2$. It appends the pair $(e', \mathsf{H}_{\mathsf{Enc}}(e'))$ into the table L_{enc} and returns $\mathsf{H}_{\mathsf{Enc}}(e')$.

- *An encoding handle and a ring element:* Given a ring element $\boldsymbol{a} \in R$, an encoding handle $\mathsf{H}_{\mathsf{Enc}}(e)$ and a multiplication operation $\cdot$ first it checks if the encoding handle already exists in the corresponding table L_{enc}.[25] If it does not exist then this call fails. Otherwise, it computes the new encoding $e' = \boldsymbol{a} \cdot e$ via ring multiplication and generates the new handle $\mathsf{H}_{\mathsf{Enc}}(e')$. It appends the entry $(e', \mathsf{H}_{\mathsf{Enc}}(e'))$ into the table L_{enc} and outputs $\mathsf{H}_{\mathsf{Enc}}(e')$.

– **Zero testing.** Given a encoding-handle $\mathsf{H}_{\mathsf{Enc}}(e)$ as input, $\mathcal{M}$ first locates the corresponding encoding $e = (\!|\boldsymbol{t}|\!)_{\mathbf{v}}$ in L_{enc}. If it is not found then (that is, if $\mathsf{H}_{\mathsf{Enc}}(e)$ was not previously generated by $\mathcal{M}$) then call to $\mathcal{M}$ fails. Otherwise, it performs zero-test on e. If the zero test fails, then this call fails. If it passes (i.e. returns 1) then recall from Definition 6 that $\boldsymbol{t} = 0 \mod \boldsymbol{g}$ which, in turn implies that $\boldsymbol{t}$ must be of the form $\boldsymbol{t} = \boldsymbol{a}'\boldsymbol{g}$. So it computes the ring element $\boldsymbol{a}' = \boldsymbol{t}/\boldsymbol{g}$, generates the corresponding ring handle $\mathsf{H}_{\mathsf{Rng}}(\boldsymbol{a}')$, appends the pair $(\boldsymbol{a}', \mathsf{H}_{\mathsf{Rng}}(\boldsymbol{a}'))$ into the table L_{rng} and outputs $\mathsf{H}_{\mathsf{Rng}}(\boldsymbol{a}')$.

– **Post-zeroizing computation.** Given a non-zero polynomial p of bounded degree and a sequence of ring handles $\mathsf{H}_{\mathsf{Rng}}(\boldsymbol{a}_1), \cdots, \mathsf{H}_{\mathsf{Rng}}(\boldsymbol{a}_v)$, $\mathcal{M}$ first locates the corresponding elements $\boldsymbol{a}_1, \cdots, \boldsymbol{a}_v$ in the table L_{rng}. If any of them is not found in L_{rng} (that is not generated by the above zero-test query) then call to $\mathcal{M}$ fails. Otherwise, $\mathcal{M}$ evaluates the polynomial $\widehat{\boldsymbol{a}} := p(\boldsymbol{a}_1, \cdots, \boldsymbol{a}_v)$. Then it checks if $\exists\ i \in [t]$, for which $\widehat{\boldsymbol{a}} = 0 \pmod{\mathcal{I}_i}$.[26] If the check fails, it returns 0. Otherwise, it returns 1. Furthermore, in this case $\mathcal{M}$ reveals its entire state including both lists L_{enc} and L_{rng} and the secrets $\boldsymbol{g}_1, \ldots, \boldsymbol{g}_t$.[27]
Note that the construction does not need access to the post-zeroing computation. Only the attacker gets access to these queries.

Remark 2. We note that one natural restriction that is implicitly placed on the attacker is that the attacker is not allowed to use the ring elements stored in the handle-table L_{rng} in multiplying with the encodings itself. This is a reasonable restriction because all ring elements generated after zero-test (the ones with corresponding handles in L_{rng}) are "large" and multiplying it with any encoding makes the numerator in that encoding large enough such that no zero-test can be performed on it.

[25] Note that the only operation we allow is the multiplication. Moreover, for GGH construction (and for its modification that we consider) addition of a ring element to an encoding is not well-defined.

[26] Note that here the model is slightly stronger than the model of [MSZ16] as an exactly equivalent model here would have checked if the value is 0 in each slot, instead of checking at least one slot.

[27] Intuitively, if the adversary is able to query such a polynomial then it wins. Formally, this is captured in the model by making the oracle to output the entire state of the oracle.

C.1 $i\mathcal{O}$ in the Weak Multilinear Map Model

We now define the indistinguishability obfuscation property in an idealized model where all algorithms have access to an oracle $\mathcal{M}$. Later we will prove that our construction achieves this definition in the weak multilinear map model in which $\mathcal{M}$ is an oracle as described above. As mentioned earlier, our construction doesn't need the post-zeroing computation and these queries are meant to provide the attacker with additional power.

Definition 7 ($i\mathcal{O}$ in an $\mathcal{M}$-idealized model [BGK+14]). *For a (possibly randomized) oracle $\mathcal{M}$, and a circuit class $\{\mathcal{C}_\ell\}_{\ell\in\mathbb{N}}$, we say that a uniform PPT oracle machine $\mathcal{O}$ is a Indistinguishability Obfuscator for $\{\mathcal{C}_\ell\}_{\ell\in\mathbb{N}}$ in the $\mathcal{M}$-idealized model, if the following conditions are satisfied:*

- *Functionality: For every $\ell \in \mathbb{N}$, every $C \in \mathcal{C}_\ell$, every input x to C, and for every possible coins for $\mathcal{M}$:*

$$\Pr[(\mathcal{O}^{\mathcal{M}}(C))(x) \neq C(x)] \leq \mathsf{negl}(|C|),$$

  where the probability is over the coins of $\mathcal{O}$.
- *Polynomial Slowdown: there exist a polynomial* poly *such that for every $\ell \in \mathbb{N}$ and every $C \in \mathcal{C}_\ell$, we have that $|\mathcal{O}^{\mathcal{M}}(C)| \leq \mathsf{poly}(|C|)$.*
- *Unbounded Simulation for every PPT adversary $\mathcal{A}$ there exist a possibly unbounded simulator $\mathcal{S}$, and a negligible function μ such that for all PPT distinguishers D, for every $\ell \in \mathbb{N}$ and every $C \in \mathcal{C}_\ell$:*

$$|\Pr[D(\mathcal{A}^{\mathcal{M}}(\mathcal{O}^{\mathcal{M}}(C))) = 1] - \Pr[D(\mathcal{S}^{C}(1^{|C|})) = 1]| \leq \mu(|C|) \ ,$$

  where the probabilities are over the coins of D, $\mathcal{A}$, $\mathcal{S}$, $\mathcal{O}$ and $\mathcal{M}$.

D Security Proof

Before we give a formal proof of security of our construction in the weak multilinear map model, we give some definitions and tools that would be useful in the security proof. These properties are similar to the ones needed in [AB15, Lin16]. Parts of this section have been taken verbatim from [AB15, Lin16].

D.1 Useful Definitions for Security Proof

We use the same distributions on rings as in [AB15, Lin16] and we define it below.

Definition 8. *An ensemble of probability distributions $\{\mathcal{N}_k\}$ is k-admissible if $\mathcal{N}_k$ samples a poly(k)-bit integer with the property that the min-entropy of every prime factor of $\mathcal{N}_k$ is at least $\Omega(k)$. An ensemble of probability distributions over rings $\{\mathcal{R}_k\}$ is k-admissible if $\mathcal{R}_k \cong \mathbb{Z}_N$ and the random variable N is k-admissible.*

It is not hard to see that every small fixed integer x is likely to be co-prime to $y \xleftarrow{\$} \mathcal{N}_k$. Using this, [AB15] proved the following useful lemma.

Lemma 11 ([AB15], Corollary 5.7). *Let $L \in \mathbb{N}$ and let $\mathcal{L} \subseteq \mathbb{Z} \setminus \{0\}$ be a list of L integers such that all $x \in \mathcal{L}$, $|x| \leq 2^{poly(\lambda)}$. Let $\mathcal{R} \cong \mathbb{Z}_N$ be a ring where N is chosen from some $(\log L + \omega(\log \lambda))$-admissible distribution. Then, the probability that there exists $x \in \mathcal{L}$ which is not a unit in $\mathcal{R}$ is $\mathsf{negl}(\lambda)$.*

Level respecting adversaries. Here, we define the level function as well as the level respecting adversaries. At a high level, a level-respecting adversary is an algebraic adversary, that is, one who performs only legal operations.

Definition 9 (Partial order of vectors). *For an integer $\tau \in \mathbb{N}$, we view vectors in $\mathbb{N}^\tau$ as multisets over the universe $[\tau]$. We define a partial ordering on vectors $\mathbb{N}^\tau$ as follows. We say that $\boldsymbol{v} \leq \boldsymbol{w}$ if for all $i \in [\tau]$ it holds that $\boldsymbol{v}[i] \leq \boldsymbol{w}[i]$. If there exists a coordinate i for which the above does not hold, we say that $\boldsymbol{v} \not\leq \boldsymbol{w}$.*

Definition 10 (The level function). *For an arithmetic circuit C and a sequence of vectors $\{\boldsymbol{v}_1, \cdots, \boldsymbol{v}_\ell\}$, we define an assignment of levels to every wire w in C via the following recursive process:*

- *If w is the i^{th} input wire, label it with level $\boldsymbol{v}_i$.*
- *If w is the output wire of a multiplication gate in C with input wires u_1 and u_2 with levels $\boldsymbol{v}_1 \neq \perp$ and $\boldsymbol{v}_2 \neq \perp$ separately, then label it with level $\boldsymbol{v}_1 + \boldsymbol{v}_2$.*
- *If w is the output wire of an addition/subtraction gate in C with input wires u_1 and u_2 with levels $\boldsymbol{v}_1 \neq \perp$ and $\boldsymbol{v}_2 \neq \perp$ separately, then label it with level $\boldsymbol{v}_1$ if $\boldsymbol{v}_1 = \boldsymbol{v}_2$; $\perp$ otherwise.*

Definition 11 (Level-respecting arithmetic circuits). *We say that an arithmetic circuit C is $((\boldsymbol{v}_1, \cdots, \boldsymbol{v}_\ell), \boldsymbol{v}_{zt})$-respecting if the output wire w of C has level $\boldsymbol{v}_w \neq \perp$ such that $\boldsymbol{v}_w \leq \boldsymbol{v}_{zt}$. We simply write $\boldsymbol{v}_{zt}$-respecting when $(\boldsymbol{v}_1, \cdots, \boldsymbol{v}_\ell)$ is clear from context.*

Next we give a bound on the size of the coefficients of a polynomial computed by an arithmetic circuit of bounded size and bounded degree.

Lemma 12. *Let C be a arithmetic circuit of size s and degree d. Then the polynomial P_C has bounded norm $|P_C|_1 \leq 2^{sd}$ (where the norm refers to the ℓ_1 norm of the coefficient vector of P_C).*

Proof. We prove by induction. If the output gate of C is a multiplication gate, then consider the two circuits representing the input wires to this gate. These circuits have size $\leq (s-1)$ and degrees d_1, d_2 such that $d_1 + d_2 \leq d$. By inductive hypothesis $|P_C|_1 \leq 2^{(s-1)d_1} \cdot 2^{(s-1)d_2} \leq 2^{sd}$. If the output gate of C is an addition/subtraction gate, then the input wires have size $s-1$ and degrees at most d, hence $|P_C|_1 \leq 2^{(s-1)d} + 2^{(s-1)d} \leq 2^{sd}$.

Definition 12. *Let $P(X_1, \cdots, X_n)$ be a polynomial. We say that P is X_i-free if all monomials that contain X_i take zero coefficient. We extend this notion to monomials and say that P is $\left(\prod X_i^{d_i}\right)$-free if all monomials that are divisible by $\left(\prod X_i^{d_i}\right)$ take zero coefficient. For a set of monomials $\{M_1, \cdots, M_k\}$ we say that P is $\{M_1, \cdots, M_k\}$-free if it is M_j-free for all $j \in [k]$.*

D.2 Other Useful Tools from [Lin16]

As mentioned in the $i\mathcal{O}$ construction (see Section 4), we would compute the circuit as well a PRF on the same input jointly in order to argue security against post-zeroizing attacks. Hence, we need to argue that we can compute a PRF on polynomial sized domain (same as inputs for seed class circuits) using constant input types and constant type degree. For this, we note that the seed class of circuits in [Lin16] internally compute a puncturable PRF (PPRF) and hence, it proves that given a suitable PRG, the class of PPRF required has constant degree, constant input types and constant type degree. We state the claims from [Lin16] below.

The special purpose circuits require a PPRF function with input domain $\{0, \ldots, T\}$, key domain $\{0,1\}^\lambda$, and range $\{0,1\}^{L(\lambda)}$ for $L(\lambda)$ long enough to supply the random coins for one-bit output functional encryption scheme bFE and randomized encodings RE; hence $L(\lambda) = poly(\lambda, n, \log T)$. The following lemma provides such a PPRF in constant degree.

Lemma 13 ([Lin16], Lemma 4). *Assume the existence of a degree-d PRG with $\lambda^{1+\varepsilon}$-stretch for some constant $d \in \mathbb{N}$ and $\varepsilon > 0$. For every polynomial D and L, there is a degree deg' PPRF scheme with input domain $\{0, \ldots, D(\lambda)\}$, key domain $\{0,1\}^\lambda$, and range $\{0,1\}^{L(\lambda)}$, where $deg' \in \mathbb{N}$ is some constant depending on d, ε, D and L. Furthermore, if the underlying PRG is subexponentially secure, then so is the PPRF.*

Lemma 14 ([Lin16], Claim 4). *If PRG has degree $d(\lambda)$, then all output bits of PPRF in the special purpose circuits have type degree $poly(d(\lambda))$ over same input types as special purpose circuits.*

D.3 Unbounded Simulation Security

To prove security, we need to show that for any PPT adversary $\mathcal{A}$, for any circuit C, there exists an unbounded time simulator $\mathcal{S}$ that simulates the view of the adversary. Since we are in the weak multilinear map model, the obfuscation that is given to $\mathcal{A}$ consists of handles to various encodings depending on the circuit C. Note that the levels $(\mathbf{v}_1, \ldots, \mathbf{v}_\ell)$ at which these encodings are generated are independent of the actual circuit being obfuscated. Hence, since the encodings are just a collection of random handles, $\mathcal{S}$ emulates them by sampling a collection of random handles $\{\mathsf{H}_{\mathsf{Enc}}(e_i)\}$ on its own and records $(\star, \mathbf{v}_i, \mathsf{H}_{\mathsf{Enc}}(e_i))$. It then gives $\{\mathsf{H}_{\mathsf{Enc}}(e_i)\}$ to $\mathcal{A}$.

Now, the simulator needs to simulate the zero-test queries as well as the post-zeroizing computation as part of weak multilinear map model. We describe these below. First note that since we are in the oracle model, it suffices to consider only those polynomials for zero-testing that are level-respecting or algebraic. Before we provide our simulator, we make some structural claims on the polynomials being zero-tested.

Bounding the number of semi-monomials. Fix a circuit $\mathcal{C}$ and polynomial P that is $\mathbf{v}_{zt}$-respecting. We can re-write P as a sum of terms in the form of $M(\boldsymbol{r}) \cdot Q(\boldsymbol{w})$, where M is a monomial and Q is a polynomial. Namely $P = \sum_i M_i(\boldsymbol{r}) \cdot Q_i(\boldsymbol{w})$. Each term in the summation has distinct $M(\boldsymbol{r})$ and is referred to as a "semi-monomial". There are at most $L = 2^{\mathsf{poly}(\lambda)}$ terms in the summation, for the following reason.

Lemma 15. *There are at most $L = 2^{\mathsf{poly}(\lambda)}$ distinct $M(\boldsymbol{r})$ monomials.*

Proof. Since P is $\mathbf{v}_{zt}$-respecting, it is easy to see that the degree of P is bounded by $|\mathbf{v}_{zt}|_1$, and so is the degree of any monomial $M(\boldsymbol{r})$ in P. Therefore, the number of distinct monomials is bounded by $L = |\boldsymbol{r}|^{|\mathbf{v}_{zt}|_1}$ where $|\boldsymbol{r}|$ is the number of r variables. In the $i\mathcal{O}$ construction both $|\boldsymbol{r}|$ and $|\mathbf{v}_{zt}|_1$ are bounded by $\mathsf{poly}(\lambda)$. Therefore $L = 2^{\mathsf{poly}(\lambda)}$.

In our construction $\mathcal{R}$ will be chosen randomly such that $\mathcal{R} \cong \mathbb{Z}_N$ where N is chosen from some $(\log L + \omega(\log \lambda))$-admissible distribution (see Definition 8 for admissible distributions). This setting is chosen so that by Lemmas 11, 12, 15, the coefficient of the monomials are units in $\mathcal{R}$. This would be used in proving successful simulation of zero-test queries.

Structural Analysis on the Polynomials. For each semi-monomial $M(\boldsymbol{r}) \cdot Q(\boldsymbol{w})$ we have the following lemma:

Lemma 16. *There exists a constant a and $\bar{w}$-free polynomial $Q'(\boldsymbol{w})$ such that*

$$Q(\boldsymbol{w}) = a \cdot \bar{w} - Q'(\boldsymbol{w}).$$

Proof. First of all, we note that the structure of our sets prevents $\bar{w}$ from being multiplied by any of the other w variables for the following reason. $\bar{w}$ is encoded at level $\geq D\mathbf{v}^*$, and the other w variables are encoded at level $\geq \mathbf{v}^*$. Any product of $\bar{w}$ and another w variable will be at level $\geq (D+1)\mathbf{v}^*$. Since $(D+1)\mathbf{v}^* \not\leq \mathbf{v}_{zt}$, contradiction follows.

Lemma 17. *For every $k \in [c]$, the polynomial Q (and hence also the polynomial Q' from Lemma 16) is $\left(\hat{w}^k\right)^2$-free.*

Proof. $\hat{w}^k$ is encoded at level $\hat{\mathbf{v}}_s^k + \mathbf{v}^*$ for some $s \in \Sigma$, thus $\left(\hat{w}^k\right)^2$ is encoded at level $\geq \hat{\mathbf{v}}_{s_1}^k + \hat{\mathbf{v}}_{s_2}^k$ for some $s_1, s_2 \in \Sigma$. Since $\hat{\mathbf{v}}_{s_1}^k + \hat{\mathbf{v}}_{s_2}^k \not\leq \mathbf{v}_{zt}$, contradiction follows.

The three main cases. We distinguish between the following three exhaustive cases of semi-monomials.

- **Invalid I:** It holds that $\hat{w} \nmid Q'(\boldsymbol{w})$.
- **Invalid II:** It holds that $\hat{w} \mid Q'(\boldsymbol{w})$, namely (by Lemma 17) there exists $Q''(\boldsymbol{w})$ which is $\{\hat{w}_1, \cdots, \hat{w}_c\}$-free such that $Q'(\boldsymbol{w}) = \hat{w} \cdot Q''(\boldsymbol{w})$. However,

  $$Q''(\boldsymbol{w}) \neq a \cdot \widetilde{w} \cdot \sum_{t=1}^{n} \left(w_{t,m+1}^{c+1} \cdot \mathcal{U}\left(\left\{ w_{x^1,j}^{1} \right\}_{j\in[\ell]}, \cdots, \left\{ w_{x^c,j}^{c} \right\}_{j\in[\ell]}, \left\{ w_{t,j}^{c+1} \right\}_{j\in[m]} \right) \right)$$

  for every possible input $x \in \Sigma^c$.
- **Valid:** There exists $x \in \Sigma^c$ such that

  $$Q(\boldsymbol{w}) = a \cdot \left(\bar{w} - \hat{w} \cdot \widetilde{w} \cdot \sum_{t=1}^{n} \left(w_{t,m+1}^{c+1} \cdot \mathcal{U}\left(\left\{ w_{x^1,j}^{1} \right\}_{j\in[\ell]}, \cdots, \left\{ w_{x^c,j}^{c} \right\}_{j\in[\ell]}, \left\{ w_{t,j}^{c+1} \right\}_{j\in[m]} \right) \right) \right).$$

Our Simulator $\mathcal{S}$ for Zero-Testing and Post-Zeroizing Computation.
Fix a circuit $\mathcal{C}$ and polynomial P which is $\mathbf{v}_{zt}$-respecting,

- **Simulating zero testing:**
  1. **Decomposition:** $\mathcal{S}$ first "decomposes" P as a sum of terms in the form $M(\boldsymbol{r}) \cdot Q(\boldsymbol{w})$, where M is a monomial and Q is a polynomial. There are at most $L = 2^{\mathsf{poly}(\lambda)}$ of them by Lemma 15.
  2. **Zero-testing each monomial:** For each term $M(\boldsymbol{r}) \cdot Q(\boldsymbol{w})$, $\mathcal{S}$ distinguishes between the following cases:
     - In cases **Invalid I** and **Invalid II**, $\mathcal{S}$ determines that Q is non-zero.
     - In case **Valid**, $\mathcal{S}$ queries its oracle $\mathcal{C}$ on input x and obtains y. It determines that Q is zero if and only if $y = 1$.
  3. **Summarizing:** If for every term $M(\boldsymbol{r}) \cdot Q(\boldsymbol{w})$ the output of Q is determined to be zero, then $\mathcal{S}$ outputs 1 (meaning that the output of P is zero) and gives a random handle $\mathsf{H}_{\mathsf{Rng}}(\mathsf{rng})$ to $\mathcal{A}$. Otherwise, $\mathcal{S}$ outputs 0.
- **Simulating post-zeroizing computation:** $\mathcal{S}$ always outputs 0 (meaning that the post-zeroing computation fails).

The simulator will produce a list $\mathcal{L} = 2^{\mathsf{poly}(\lambda)}$ of L integers of absolute value at most $2^{\mathsf{poly}(\lambda)}$. In particular, this list would be a subset of the coefficients of the polynomial P computed by the adversary. Since P is computable by a purely arithmetic circuit of size $\mathsf{poly}(\lambda)$ and degree at most $||\mathbf{v}_{zt}||_1$, the bounds follow from Lemmas 12 and 15. We will show that as long as all the elements of $\mathcal{L}$ are units in $\mathcal{R}$, the simulation is successful. This happens with high probability by Lemma 11.

Remark 3. Note that above we allow for zero-testing at levels lower than $\mathbf{v}_{zt}$ as well and prove what is referred to as the **Strong Algebraic Security** in [Lin16]. In fact, we would prove that any polynomial at a level $\mathbf{v} < \mathbf{v}_{zt}$ is not a zero with high probability over the randomness of encodings. This would be crucial in proving security against post-zerozing computations in our scenario. For security, we want that the adversary cannot come up with any polynomial that results in a zero over encondings $\{\mathsf{H}_{\mathsf{Rng}}(\mathsf{rng}_i)\}$.

D.4 Correctness of Simulating Zero Test

Theorem 3. *The output of $\mathcal{S}$ in the zero test is correct with probability $1 - \mathsf{negl}(\lambda)$.*

Proof. For each term $M(\boldsymbol{r})Q(\boldsymbol{w})$, by lemmas 18, 20, 21 (will be proved in the following), in any of the three cases the emulation of $\mathcal{S}$ is correct except with probability $\frac{\mathsf{negl}(\lambda)}{L}$. There are at most L terms, by union bound the output of $\mathcal{S}$ in the zero test is correct except with probability $\mathsf{negl}(\lambda)$.

Conditioned on this happening, if all of Q evaluates to 0, simulation is correct. In the other case, polynomial P can be seen as a polynomial over variables r's and the coefficient as $Q(\boldsymbol{w})$. Since one of the $Q(\boldsymbol{w})$ evaluates to non-zero, this polynomial is not identically 0. Hence, by Lemma 9 when values r's are randomly chosen then the probability that the P evaluates to 0 is at most $\mathsf{negl}(\lambda)$. By union bound over all the polynomials queried by the adversary, error probability is at most $\mathsf{negl}(\lambda)$.

Next, we prove that the simulation of each of the semi-monomials is correct. In this section, by $a[\![k]\!]$ we denote the component of a in subring $\mathcal{R}_k$ for $k \in [c{+}3]$.

Lemma 18 (Invalid I). *If $\hat{w} \nmid Q'(\boldsymbol{w})$, then $\Pr[Q(\boldsymbol{w}) = 0] = \frac{\mathsf{negl}(\lambda)}{L}$, where the probability is taken over the randomness of $\mathcal{R}$ and $\boldsymbol{w}$ variables.*

Proof. Recall that $\hat{w} = \prod_{k=1}^{c} \hat{w}^k$. Below we prove that if there is a $k \in [c]$ such that $\hat{w}^k \nmid Q'$, then Q outputs zero with small probability.

Recall that $Q(\boldsymbol{w}) = a \cdot \bar{w} - Q'(\boldsymbol{w})$. Consider the evaluation of Q over the $(k+3)^{rd}$ sub-ring, $Q(\boldsymbol{w})[\![k+3]\!]$. Since $\bar{w}[\![k+3]\!] = 0$, it holds that $Q(\boldsymbol{w})[\![k+3]\!] = Q'(\boldsymbol{w})[\![k+3]\!]$. Recall that $Q'(\boldsymbol{w})[\![k+3]\!] = Q'[\![k+3]\!](\boldsymbol{w}[\![k+3]\!])$ (i.e., the evaluation of $Q'[\![k+3]\!]$ over $\boldsymbol{w}[\![k+3]\!]$).

Since $\hat{w}^k \nmid Q'$, there exists a polynomial $Q'_1(\boldsymbol{w})$ and a $\hat{w}^k$-free polynomial $Q'_2(\boldsymbol{w})$ such that $Q'_2(\boldsymbol{w})$ is not identically zero and that $Q'(\boldsymbol{w}) = \hat{w}^k Q'_1(\boldsymbol{w}) + Q'_2(\boldsymbol{w})$. Since $\hat{w}^k[\![k+3]\!] = 0$, it holds that $Q'(\boldsymbol{w})[\![k+3]\!] = Q'_2(\boldsymbol{w})[\![k+3]\!]$. Note that $Q'_2(\boldsymbol{w})$ contains at least one non-zero monomial, with coefficient α. By Lemma 12, α has bounded ℓ_1 norm. Therefore by Lemma 11, with overwhelming probability α is a unit, and thus $\alpha[\![k+3]\!]$ is non-zero. Hence $Q'_2[\![k+3]\!]$ (and also $Q'[\![k+3]\!]$) is not identically zero.

Recall that all w variables, except $\hat{w}^k$, contain random ρ elements (in particular, $\left\{\rho^{k'}_{s,j,k}\right\}_{s\in\Sigma, j\in[\ell], k'\in[c]}, \left\{\rho^{c+1}_{t,j,k}\right\}_{t\in[n], j\in[m+1]}, \left\{\hat{\rho}^{k'}_{k}\right\}_{k'\neq k}, \widetilde{\rho}_k$) in the $(k+3)^{rd}$ slot. By Lemma 9, the probability that $Q'[\![k+3]\!]$ evaluates to zero over randomly chosen ρ variables in the $(k+3)^{rd}$ sub-ring $\mathcal{R}_{k+3}$ is $\frac{\mathsf{negl}(\lambda)}{L}$ (by using the fact that the degree of Q' is polynomial and that $\mathcal{R}$ is $(\log L + \omega(\log\lambda))$-admissible).

Lemma 19. *If $\hat{w} \mid Q'(\boldsymbol{w})$, then there exists an input $x = x^1, \cdots, x^c$ such that Q' is free of variables $\left\{w^k_{s,j}\right\}_{k\in[c], s\neq x^k, j\in[\ell]}$.*

Proof. Assume for the purpose of contradiction that there exists $k \in [c], j_1, j_2 \in [\ell]$ and $s_1, s_2 \in \Sigma$ such that $s_1 \neq s_2$ and that Q' is neither $w^k_{s_1,j_1}$-free nor $w^k_{s_2,j_2}$-free. $w^k_{s_1,j_1}$ and $w^k_{s_2,j_2}$ are encoded at levels $\mathbf{v}^k_{s_1} + \mathbf{v}^*$ and $\mathbf{v}^k_{s_2} + \mathbf{v}^*$ respectively.

Since $\hat{w} \mid Q'(\boldsymbol{w})$ and $\hat{w}^k \mid Q'(\boldsymbol{w})$, there exists $s' \in \Sigma$ such that Q' is at level $\geq \hat{\mathbf{v}}^k_{s'}$. Thus $Q'(\boldsymbol{w})$ is encoded at level $\geq \mathbf{v}^k_{s_1} + \mathbf{v}^k_{s_2} + \hat{\mathbf{v}}^k_{s'}$. Since $\mathbf{v}^k_{s_1} + \mathbf{v}^k_{s_2} + \hat{\mathbf{v}}^k_{s'} \not\leq \mathbf{v}_{zt}$, contradiction follows.

Lemma 20 (Invalid II). *If $\hat{w} \mid Q'(\boldsymbol{w})$, namely there exists $Q''(\boldsymbol{w})$ which is $\{\hat{w}_1, \cdots, \hat{w}_c\}$-free such that $Q'(\boldsymbol{w}) = \hat{w} \cdot Q''(\boldsymbol{w})$. However,*

$$Q''(\boldsymbol{w}) \neq a \cdot \widetilde{w} \cdot \sum_{t=1}^{n} \left(w^{c+1}_{t,m+1} \cdot \mathcal{U}\left(\left\{ w^1_{x^1,j} \right\}_{j\in[\ell]}, \cdots, \left\{ w^c_{x^c,j} \right\}_{j\in[\ell]}, \left\{ w^{c+1}_{t,j} \right\}_{j\in[m]} \right) \right)$$

for every possible input $x \in \Sigma^c$, then $\Pr[Q(\boldsymbol{w}) = 0] = \frac{\mathsf{negl}(\lambda)}{L}$.

Proof. Consider the x from Lemma 19, $Q''(\boldsymbol{w})$ is a polynomial over $\left\{ w^k_{x^k,j} \right\}_{k\in[c],j\in[\ell]}$, $\left\{ w^{c+1}_{t,j} \right\}_{t\in[n],j\in[m+1]}, \widetilde{w}$. Consider $Q''(\boldsymbol{w})[\![1]\!]$ $(= Q''[\![1]\!](\boldsymbol{w}[\![1]\!]))$, since all these w variables contain random y values $(= \left\{ y^k_j \right\}_{k\in[c],j\in[\ell]}, \left\{ y^{c+1}_{t,j} \right\}_{t\in[n],j\in[m+1]}, \widetilde{y})$ in the first slot, we have

$$Q''[\![1]\!]\left(\boldsymbol{w}[\![1]\!]\right) \neq a[\![1]\!] \cdot \widetilde{y} \cdot \sum_{t=1}^{n} \left(y^{c+1}_{t,m+1} \cdot \mathcal{U}\left(\left\{ y^1_j \right\}_{j\in[\ell]}, \cdots, \left\{ y^c_j \right\}_{j\in[\ell]}, \left\{ y^{c+1}_{t,j} \right\}_{j\in[m]} \right) \right).$$

Consider the evaluation of Q in the first sub-ring:

$$Q(\boldsymbol{w})[\![1]\!] = \prod_{k\in[c]} \hat{y}^k \cdot \left(a[\![1]\!] \cdot \widetilde{y} \cdot \sum_{t=1}^{n} \left(y^{c+1}_{t,m+1} \cdot \mathcal{U}\left(\left\{ y^1_j \right\}_{j\in[\ell]}, \cdots, \left\{ y^c_j \right\}_{j\in[\ell]}, \left\{ y^{c+1}_{t,j} \right\}_{j\in[m]} \right) \right) - Q''[\![1]\!]\left(\boldsymbol{w}[\![1]\!]\right) \right)$$

is not identically zero. By Lemma 9, the probability that $Q[\![1]\!]$ evaluates to zero over randomly chosen y variables in the first sub-ring $\mathcal{R}_1$ is $\frac{\mathsf{negl}(\lambda)}{L}$ (by using the fact that the degree of Q is polynomial and that $\mathcal{R}$ is $(\log L + \omega(\log \lambda))$-admissible).

Lemma 21 (Valid). *If there exists $x \in \Sigma^c$ such that*

$$Q(\boldsymbol{w}) = a \cdot \left(\bar{w} - \hat{w} \cdot \widetilde{w} \cdot \sum_{t=1}^{n} \left(w^{c+1}_{t,m+1} \cdot \mathcal{U}\left(\left\{ w^1_{x^1,j} \right\}_{j\in[\ell]}, \cdots, \left\{ w^c_{x^c,j} \right\}_{j\in[\ell]}, \left\{ w^{c+1}_{t,j} \right\}_{j\in[m]} \right) \right) \right),$$

then, if $\mathcal{C}(x) = 1$ then $\Pr[Q = 0] = 1$; if $\mathcal{C}(x) = 0$ then $\Pr[Q = 0] = \frac{\mathsf{negl}(\lambda)}{L}$.

Proof. In this case, $Q'(\boldsymbol{w}) = \hat{w} \cdot Q''(\boldsymbol{w})$. Consider the x from Lemma 19, it holds that

$$Q''(\boldsymbol{w}) = a \cdot \widetilde{w} \cdot \sum_{t=1}^{n} \left(w^{c+1}_{t,m+1} \cdot \mathcal{U}\left(\left\{ w^1_{x^1,j} \right\}_{j\in[\ell]}, \cdots, \left\{ w^c_{x^c,j} \right\}_{j\in[\ell]}, \left\{ w^{c+1}_{t,j} \right\}_{j\in[m]} \right) \right).$$

First notice that a must be non-zero, or else Q is identically zero. Then by Lemma 11, a is a unit in $\mathcal{R}$ except with probability $\frac{\mathsf{negl}(\lambda)}{L}$.

By definition $Q(\boldsymbol{w})$ evaluates to zero on all sub-rings except the second. Therefore it suffices to test whether $Q(\boldsymbol{w})[\![2]\!]$ is zero or not.

$$Q(\boldsymbol{w})[\![2]\!] = a[\![2]\!] \cdot \hat{\beta}\widetilde{\beta} \cdot \left(n - \sum_{t=1}^{n} \mathcal{U}\left(x, \mathcal{C}\right) \right).$$

If $\mathcal{U}\left(x, \mathcal{C}\right) = 1$ (i.e., $\mathcal{C}(x) = 1$), then $Q(\boldsymbol{w})[\![2]\!]$ equals zero with probability 1 and so does $Q(\boldsymbol{w})$. Otherwise, in the case $\mathcal{C}(x) = 0$, $Q(\boldsymbol{w})[\![2]\!]$ is a non-zero polynomial (with a non-zero coefficient $a[\![2]\!]$) over random $\widetilde{\beta}$ and $\left\{\hat{\beta}^k\right\}_{k\in[c]}$. By Lemma 9, $Q[\![2]\!]$ (and hence Q) is non-zero except with probability $\frac{\mathsf{negl}(\lambda)}{L}$ (by using the fact that the degree of Q is polynomial and that $\mathcal{R}$ is $(\log L + \omega(\log \lambda))$-admissible).

D.5 Correctness of Simulating Post-Zeroizing Computation

We first prove the following claim about the encoding that results in a successful zero-test.

Lemma 22. *If $\mathcal{S}$ outputs 1 for the zero test on a polynomial P, then $P = \sum_{i=1}^{d} M_i(\boldsymbol{r}) \cdot Q_i(\boldsymbol{w})$ and with probability $1 - \mathsf{negl}(\lambda)$ it holds that d is polynomial in λ. In fact, $d \leq |\Sigma|^c$.*

Proof. Recall that if $\mathcal{S}$ outputs 1 for the zero test, then with probability $1 - \mathsf{negl}(\lambda)$ every term $M_i(\boldsymbol{r}) \cdot Q_i(\boldsymbol{w})$ is in the valid case. For each $M_i(\boldsymbol{r}) \cdot Q_i(\boldsymbol{w})$ term, by Lemmas 19 and 21 there is a unique $x \in \Sigma^c$ such that

$$Q_i(\boldsymbol{w}) = a \cdot \left(\bar{w} - \hat{w} \cdot \widetilde{w} \cdot \sum_{t=1}^{n} \left(w_{t,m+1}^{c+1} \cdot \mathcal{U}\left(\left\{ w_{x^1,j}^{1} \right\}_{j\in[\ell]}, \cdots, \left\{ w_{x^c,j}^{c} \right\}_{j\in[\ell]}, \left\{ w_{t,j}^{c+1} \right\}_{j\in[m]} \right) \right) \right).$$

and thus

$$M_i(\boldsymbol{r}) = a \cdot \left(\bar{r} - \hat{r} \cdot \widetilde{r} \cdot \sum_{t=1}^{n} \left(r_{t,m+1}^{c+1} \cdot \mathcal{U}\left(\left\{ r_{x^1,j}^{1} \right\}_{j\in[\ell]}, \cdots, \left\{ r_{x^c,j}^{c} \right\}_{j\in[\ell]}, \left\{ r_{t,j}^{c+1} \right\}_{j\in[m]} \right) \right) \right).$$

In other words, every $M_i(\boldsymbol{r})$ is defined by a unique $x \in \Sigma^c$, and every $x \in \Sigma^c$ can define at most one $M_i(\boldsymbol{r})$ term. Since the number of possible inputs is at most $|\Sigma|^c$, the lemma follows.

Theorem 4. *If $\{\boldsymbol{e}^t\}_{t\in[n]}$ in the construction are set as follows:*

$$\boldsymbol{e}^t = X^t \in R,$$

where R is the ring corresponding to the composite-order GGH defined in Section 2 and if zero-test of $[M(\boldsymbol{r}) \cdot Q(\boldsymbol{w})]_{\boldsymbol{v}}$ returns 1 ($\boldsymbol{v}$ is the level of $M(\boldsymbol{r}) \cdot Q(\boldsymbol{w})$), then with overwhelming probability the following statements are true:

– $\boldsymbol{v} = \boldsymbol{v}_{zt}$.

– *There exists* $x \in \Sigma^c$ *such that*

$$Q(\boldsymbol{w}) = a \cdot \left(\bar{w} - \hat{w} \cdot \widetilde{w} \cdot \sum_{t=1}^{n} \left(w_{t,m+1}^{c+1} \cdot \mathcal{U} \left(\left\{ w_{x^1,j}^{1} \right\}_{j\in[\ell]}, \cdots, \left\{ w_{x^c,j}^{c} \right\}_{j\in[\ell]}, \left\{ w_{t,j}^{c+1} \right\}_{j\in[m]} \right) \right) \right).$$

– $\mathcal{C}(x) = 1$.

– *The corresponding encoding* $(\!|\boldsymbol{a}|\!)_{\boldsymbol{v}} := [M(\boldsymbol{r}) \cdot Q(\boldsymbol{w})]_{\boldsymbol{v}}$ *has the property that*

$$\boldsymbol{a} = \left(\alpha \cdot \mathcal{C}^{\mathsf{PRF}}(x) + \boldsymbol{d}_x \right) \cdot \boldsymbol{g} = a' \cdot \boldsymbol{g},$$

where α *is a unit in* $\mathcal{R}$*, and* $\boldsymbol{d}_x \in \mathcal{R}$*. Recall that* $\mathcal{C}^{\mathsf{PRF}}(x) = \sum_{t\in[n]} \boldsymbol{e}^t \mathcal{C}^{\mathsf{PRF}^t}(x) = \sum_{t\in[n]} X^t \mathcal{C}^{\mathsf{PRF}^t}(x)$.

Proof. The first three statements follow from Lemmas 18, 20, 21 and guarantee the evaluation is done correctly. Recall that for $\forall i \in [c+3]$, let $\gamma_i \in \mathcal{R}$ be such that $\gamma_i \cdot \prod_{j\neq i} \boldsymbol{g}_j = 1 (\bmod \mathcal{I}_i)$. These correspond to CRT reconstruction. Since

$$\begin{aligned}
\hat{w}_{\mathcal{U}} =& \hat{w} \cdot \sum_{t=1}^{n} \left(w_{t,m+1}^{c+1} \cdot \mathcal{U} \left(\left\{ w_{x^1,j}^{1} \right\}_{j\in[\ell]}, \cdots, \left\{ w_{x^c,j}^{c} \right\}_{j\in[\ell]}, \left\{ w_{t,j}^{c+1} \right\}_{j\in[m]} \right) \right) \\
=& \left(\hat{y}\bar{y}, \hat{\beta} n, \hat{\alpha} \cdot \mathcal{C}^{\mathsf{PRF}}(x), 0, \cdots, 0 \right) \\
=& \hat{y}\bar{y} \cdot \gamma_1 \cdot \prod_{j\neq 1} \boldsymbol{g}_j + \hat{\beta} n \cdot \gamma_2 \cdot \prod_{j\neq 2} \boldsymbol{g}_j + \hat{\alpha} \cdot \mathcal{C}^{\mathsf{PRF}}(x) \cdot \gamma_3 \cdot \prod_{j\neq 3} \boldsymbol{g}_j, \\
\widetilde{w} =& \left(\widetilde{y}, \widetilde{\beta}, 0, \widetilde{\rho}_1, \cdots, \widetilde{\rho}_c \right) \\
=& \widetilde{y} \cdot \gamma_1 \cdot \prod_{j\neq 1} \boldsymbol{g}_j + \widetilde{\beta} \cdot \gamma_2 \cdot \prod_{j\neq 2} \boldsymbol{g}_j + \sum_{k\in[c]} \left(\widetilde{\rho}_k \cdot \gamma_{k+3} \cdot \prod_{j\neq k+3} \boldsymbol{g}_j \right), \\
\bar{w} =& \left(\widetilde{y}\hat{y}\bar{y}, \widetilde{\beta}\hat{\beta} n, 0, 0, \cdots, 0 \right) \\
=& \widetilde{y}\hat{y}\bar{y} \cdot \gamma_1 \cdot \prod_{j\neq 1} \boldsymbol{g}_j + \widetilde{\beta}\hat{\beta} n \cdot \gamma_2 \cdot \prod_{j\neq 2} \boldsymbol{g}_j,
\end{aligned}$$

we have

$$(\!|\boldsymbol{a}|\!)_{\mathbf{v}} = (\bar{R}, \bar{Z}) - \left(\widetilde{R}_{\mathcal{U}}, \widetilde{Z}_{\mathcal{U}}\right) = \bar{Z} \times \widetilde{R}_{\mathcal{U}} - \widetilde{Z}_{\mathcal{U}} \times \bar{R}, \text{ and}$$

$$\begin{aligned}
\boldsymbol{a} = &\left[\bar{r} \cdot \left(\widetilde{y}\hat{y}\bar{y} \cdot \gamma_1 \cdot \prod_{j\neq 1} \boldsymbol{g}_j + \widetilde{\beta}\hat{\beta}n \cdot \gamma_2 \cdot \prod_{j\neq 2} \boldsymbol{g}_j\right) + \bar{\boldsymbol{d}}\boldsymbol{g}\right] \cdot \left[\hat{r}_{\mathcal{U}} \cdot \widetilde{r} + \widetilde{\boldsymbol{d}}_{R_{\mathcal{U}}}\boldsymbol{g}\right] \\
&- \left[\hat{r}_{\mathcal{U}} \cdot \left(\hat{y}\bar{y} \cdot \gamma_1 \cdot \prod_{j\neq 1} \boldsymbol{g}_j + \hat{\beta}n \cdot \gamma_2 \cdot \prod_{j\neq 2} \boldsymbol{g}_j + \hat{\alpha} \cdot \mathcal{C}^{\mathsf{PRF}}(x) \cdot \gamma_3 \cdot \prod_{j\neq 3} \boldsymbol{g}_j\right) + \hat{\boldsymbol{d}}_{\mathcal{U}}\boldsymbol{g}\right] \\
&\cdot \left[\widetilde{r} \cdot \left(\widetilde{y} \cdot \gamma_1 \cdot \prod_{j\neq 1} \boldsymbol{g}_j + \widetilde{\beta} \cdot \gamma_2 \cdot \prod_{j\neq 2} \boldsymbol{g}_j + \sum_{k\in[c]} \left(\widetilde{\rho}_k \cdot \gamma_{k+3} \cdot \prod_{j\neq k+3} \boldsymbol{g}_j\right)\right) + \widetilde{\boldsymbol{d}}\boldsymbol{g}\right] \cdot \left[\bar{r} + \bar{\boldsymbol{d}}_R\boldsymbol{g}\right] \\
= &\left\{\boldsymbol{d}_x - \hat{r}_{\mathcal{U}}\hat{\alpha}\gamma_3\mathcal{C}^{\mathsf{PRF}}(x)\left[\widetilde{r}\bar{r}\left(\widetilde{y} \cdot \gamma_1 \cdot \prod_{j\neq 1,3} \boldsymbol{g}_j + \widetilde{\beta} \cdot \gamma_2 \cdot \prod_{j\neq 2,3} \boldsymbol{g}_j + \sum_{k\in[c]} \left(\widetilde{\rho}_k \cdot \gamma_{k+3} \cdot \prod_{j\neq 3,k+3} \boldsymbol{g}_j\right)\right)\right.\right. \\
&\left.\left. + \widetilde{\boldsymbol{d}}\bar{r} \prod_{j\neq 3} \boldsymbol{g}_j\right]\right\}\boldsymbol{g},
\end{aligned}$$

where all the $\boldsymbol{d}$ terms come from the encoding procedure, and $\boldsymbol{d}_x$ depends on the input x.

Now we need to prove that the multiplicative term with $\mathcal{C}^{\mathsf{PRF}}(x)$ denoted by α in the theorem statement is a unit in $\mathcal{R}$ with high probability. We will prove this by proving that α is a unit in all sub-rings w.h.p.

Let us consider the first sub-ring $\mathcal{R}_1$. First of all, γ_3 is an inverse in $\mathcal{R}$, and $\hat{r}_{\mathcal{U}}\hat{\alpha}$ is a unit in $\mathcal{R}$ except with negligible probability, and so are $\widetilde{r}\bar{r}$ and $\widetilde{\boldsymbol{d}}\bar{r}$. Then consider the polynomial

$$\widetilde{r}\bar{r}\left(\widetilde{y} \cdot \gamma_1 \cdot \prod_{j\neq 1,3} \boldsymbol{g}_j + \widetilde{\beta} \cdot \gamma_2 \cdot \prod_{j\neq 2,3} \boldsymbol{g}_j + \sum_{k\in[c]} \left(\widetilde{\rho}_k \cdot \gamma_{k+3} \cdot \prod_{j\neq 3,k+3} \boldsymbol{g}_j\right)\right) + \widetilde{\boldsymbol{d}}\bar{r} \prod_{j\neq 3} \boldsymbol{g}_j.$$

In the first sub-ring $\mathcal{R}_1$, it is $\widetilde{r}\bar{r}\widetilde{y}\gamma_1 \cdot \prod_{j\neq 1,3} \boldsymbol{g}_j$, which is a unit except with negligible probability.

A similar argument works for all sub-rings except $\mathcal{R}_3$. Now we will argue that $\alpha[\![3]\!]$ is a unit.

Notice that $\gamma_1 \prod_{j\neq 1,3} \boldsymbol{g}_j$, $\gamma_2 \prod_{j\neq 2,3} \boldsymbol{g}_j$, $\gamma_{k+3} \prod_{j\neq 3,k+3} \boldsymbol{g}_j$, $\prod_{j\neq 3} \boldsymbol{g}_j$ are all units in the third sub-ring, and that $\widetilde{y}, \widetilde{\beta}, \widetilde{\rho}_k, \widetilde{\boldsymbol{d}}, \widetilde{r}, \bar{r}$ are all uniformly random, hence the entire polynomial is also a unit in the third sub-ring except with negligible probability.

This concludes that α is a unit in $\mathcal{R}$ with all but negligible probability.

Theorem 5. *The probability that the adversary succeeds in post-zeroizing queries is* $\mathsf{negl}(\lambda)$.

Proof. Let a'_i be as defined in above theorem for the $M_i(\boldsymbol{r}) \cdot Q_i(\boldsymbol{w})$. Then, if a polynomial P given by the adversary in encodings results in a zero, then the adversary gets a handle to a ring element $\mathsf{rng} = \sum_{i=1}^{d} a'_i$, where d is polynomial in λ by Lemma 22. Now, by the security of the PRF and using the fact that P has a polynomial number of semi-monomials, we can replace the output of each of the bit-PRFs with a uniform bit. Recall that above $a'_i = \alpha_i \cdot \mathcal{C}^{\mathsf{PRF}}(x) + \boldsymbol{d}_{x,i}$, where $\mathcal{C}^{\mathsf{PRF}}(x) = \sum_{t\in[n]} X^t \mathcal{C}^{\mathsf{PRF}^t}(x)$. That is, through a hybrid argument, we can get $a'_i = \alpha_i Y(x) + \boldsymbol{d}_{x,i}$ where $Y(x) = \sum_{t\in[n]} X^t \cdot b_{x,t}$ where $b_{x,t} \xleftarrow{\$} \{0,1\}$. Note that $H_\infty(Y(x)) \geq n$. Hence, by Corollary 1, $H_\infty(Y(x) \bmod \langle \boldsymbol{g}_i \rangle) = H_\infty(Y(x)) = H_\infty(a'_i)$, where $\langle \boldsymbol{g}_i \rangle$ is used to define the ring $\mathcal{R}_i$. In particular, $\mathcal{R}_i = R \bmod \langle \boldsymbol{g}_i \rangle$.

Since α_i is a unit in $\mathcal{R}$ by Theorem 4 with all but negligible probability,

$$H_\infty(\mathsf{rng} \bmod \langle \boldsymbol{g}_i \rangle) = H_\infty((\sum_{i=1}^{d} a'_i) \bmod \langle \boldsymbol{g}_i \rangle) \geq n.$$

Now, given handles to many ring elements $\mathsf{rng}_1, \ldots, \mathsf{rng}_k$ after successful zero-tests, any bounded degree polynomial p provided by the adversary on these ring elements will be non-zero in all sub-rings with overwhelming probability by Lemma 9. Hence, post-zeroizing simulation is correct.