Multilinear Maps from RLWE-variants

Andrew H. Russell* Nigel Smart†

August 1, 2015

Abstract

We propose a new multilinear map based on a ring variant of the GSW homomorphic encryption scheme. Our construction exploits the properties that ciphertexts are small, and that the decryption key is large and only present as a multiple of the plaintext during the decryption process. These features allow us to hide the decryption key and publish it as a zero-tester. The security of the scheme is based on variants of the RLWE problem.

1 Introduction

Multilinear maps. Since the use of bilinear pairings to construct identity-based encryption by Boneh and Franklin [**?**] there has been interest in the natural extension to cryptographic maps with multilinearity $\kappa \geq 2$. Boneh and Silverberg [**?**] outlined numerous applications for multilinear maps, but were pessimistic about finding constructions in the field of algebraic geometry. Recently, Garg, Gentry, and Halevi [**?**] proposed the idea of *graded encoding schemes*, which approximate the idea of multilinear maps, and gave the first candidate (referred to as the GGH construction). Subsequently, Coron, Lepoint, and Tibouchi proposed a similar construction but over the integers [**?**, **?**], and Gentry, Gorbunov, and Halevi proposed a construction where the structure of the map is dependent on an underlying acyclic graph [**?**].

Multilinear maps have a range of applications, the most straightforward of which is an N-way non-interactive key exchange [**?**]. Arguably the most important and prominent application that motivates the use of multilinear maps is general program obfuscation, originally outlined in [**?**] using the GGH scheme.

Cryptanalysis of current constructions. Unfortunately, the original CLT and GGH constructions have subsequently been attacked successfully [**?**, **?**, **?**], resulting in total breaks or weak-discrete log attacks, despite multiple attempts to fix them [**?**, **?**]. The graph-induced construction in [**?**] is known to be insecure in some situations through the "approximate trapdoor" attacks outlined in the original paper, which makes it harder to reason about security in various applications, as you must prove that those situations do not arise in each specific use of the map. The updated CLT construction [**?**] proposes a new zero-testing parameter that seems to avoid these attacks at the cost of some efficiency.

*Pomona College, ahr12012@mymail.pomona.edu. Work performed while at the University of Bristol.
†nigel@cs.bris.ac.uk

Our contributions. We propose a new multilinear map based on a variant of the GSW homomorphic encryption scheme found in [**?**]. The primary advantage of our scheme is that its security is based on variants of the RLWE problem. While a full reduction does not seem possible at this point, we provide a construction that relies on a security assumption that differs dramatically from prior constructions and completely avoids the previous attacks.

This scheme also provides the additional advantage that arbitrary plaintext elements can be encoded, unlike in other constructions where the plaintext of a level-0 encoding is unknown to the user.

Future work. It remains an open problem to construct multilinear maps based on standard hard problems, or to construct "true" multilinear maps rather than the graded encoding schemes presented here and in prior works. Further cryptanalysis of this new scheme could also bring about either greater confidence in its security or additional insight into the general problem of providing the zero-testing functionality.

Overview of the map. The encoding of elements and homomorphic operations are exactly as in the GSW HE scheme, except we work over a polynomial ring $\mathcal{R}_q = \mathbb{Z}_q[x]/\langle x^n + 1\rangle$ instead of $\mathbb{Z}_q$. That is, ciphertexts look like the following, where μ is the plaintext and pk is the public key hiding the plaintext:

$$C = \mu + pk$$

In GSW the decryption key sk acts as an "approximate inverse" to pk, so that when we multiply C by sk we get the following:

$$\mu \cdot sk + e$$

The plaintext μ can then be recovered because e is a small value. The main thing to note is that the decryption key is "large," so that when we decrypt the result will be large if and only if μ is non-zero. Thus, if we can somehow securely publish the decryption key we can then zero-test encodings, giving us a graded encoding scheme.

A starting point for hiding the key is to note that ciphertexts are themselves small (binary matrices), as they are a result of the $\mathsf{Flatten}$ function, so that we can add a small value ϵ to the decryption key and still get the desired property:

$$\begin{aligned} C \cdot \boldsymbol{p}_{zt} &= C(sk + \epsilon) \\ &= \mu \cdot sk + e + C\epsilon \\ &= \mu \cdot sk + small \end{aligned}$$

2 Preliminaries

First, let $\mathcal{R} = \mathbb{Z}[x]/\langle x^n + 1\rangle$ where n is a power of 2, and let $\mathcal{R}_q = \mathcal{R}/q\mathcal{R} \cong \mathbb{Z}_q[x]/\langle x^n + 1\rangle$ be the ring that contains $\ell = \lceil \log q \rceil$ bit coefficients.

2.1 Bit operations

The GSW paper outlines a few operations on vectors that we extend to the ring setting, as in [**?**]. For some k, let $\ell = \lceil \log q \rceil$ and $N = k\ell$, and let $\vec{a}$ be a k-dimensional vector of ring elements in $\mathcal{R}_q$. Then, $\mathsf{BitDecomp} = (a_{1,0}, \ldots, a_{1,\ell-1}, \ldots, a_{k,0}, \ldots, a_{k,\ell-1}) \in \mathcal{R}_q^N$ where $a_{i,j}$ is the polynomial whose entries are the j-th bit each coefficient in the i-th entry of $\vec{a}$. That is, we decompose a vector $\vec{a}$ that has polynomials with ℓ-bit coefficients into an N-dimensional vector with polynomials of 0/1 coefficients. $\mathsf{BitDecomp}^{-1}$ is the inverse of this operation. Specifically, for a vector $\vec{a} = (a_{1,0}, \ldots, a_{1,\ell-1}, \ldots, a_{k,0}, \ldots, a_{k,\ell-1})$, we define $\mathsf{BitDecomp}^{-1}(\vec{a}) = (\sum 2^j a_{1,j}, \ldots, \sum 2^j a_{k,j})$. Note that we define this function for any vector $\vec{a} \in \mathcal{R}_q^N$, not just vectors with 0/1 polynomials. Next, for a vector $\vec{a} \in \mathcal{R}_q^N$, define $\mathsf{Flatten}(\vec{a}) = \mathsf{BitDecomp}(\mathsf{BitDecomp}^{-1}(\vec{a}))$. Finally, for a vector $\vec{b} \in \mathcal{R}_q^k$ define $\mathsf{Powersof2}(\vec{b}) = (b_1, 2b_1, \ldots, 2^{\ell-1}b_1, \ldots, b_k, 2b_k, \ldots, 2^{\ell-1}b_k)$. As in GSW, the important properties of these operations are:

- $\left\langle \mathsf{BitDecomp}(\vec{a}), \mathsf{Powersof2}(\vec{b}) \right\rangle = \left\langle \vec{a}, \vec{b} \right\rangle$
- $\left\langle \mathsf{Flatten}(\vec{a}), \mathsf{Powersof2}(\vec{b}) \right\rangle = \left\langle \vec{a}, \mathsf{Powersof2}(\vec{b}) \right\rangle$

We extend these operations to matrices by performing them on each row. Thus, for matrices $A \in \mathcal{R}_q^{N \times k}$, $C \in \mathcal{R}_q^{N \times N}$ and a vector $\vec{b} \in \mathcal{R}_q^k$ we have:

- $\mathsf{BitDecomp}(A) \cdot \mathsf{Powersof2}(\vec{b}) = A \cdot \vec{b}$
- $\mathsf{Flatten}(C) \cdot \mathsf{Powersof2}(\vec{b}) = C \cdot \mathsf{Powersof2}(\vec{b})$

The $\mathsf{Flatten}$ function keeps our encodings small while maintaining correctness with the zero-tester, which is a result of the $\mathsf{Powersof2}$ function. In our scheme we set $k = 2$ so that $N = 2\ell$.

3 Our construction

We are now ready to outline our map, which is an instantiation of a graded encoding scheme as in prior works [**?**, **?**, **?**]. The basic operations (setup, encoding, addition, multiplication) of our scheme are essentially identical to those of the GSW scheme.

Setup: $(\mathsf{pp}, \boldsymbol{p}_{zt}) \leftarrow \mathsf{setup}(1^\lambda, 1^\kappa)$**.** Given the security parameter λ and the multilinearity level κ, we generate the public parameters and the zero-testing element. The parameters q and n define the ring $\mathcal{R}_q$. Set $m = O(\log q)$ and $N = 2\ell$ where $\ell = \lceil \log q \rceil$. Let χ be an error distribution on $\mathcal{R}_q$. We generate the **encoding key** A by drawing $\vec{e} \leftarrow \chi^m$, $\vec{a} \leftarrow \mathcal{R}_q^m$, and $t \leftarrow \mathcal{R}_q$. Set $\vec{b} = \vec{a}t + \vec{e}$ and $\vec{s} = (1, -t)^T$ and $A = \left(\vec{b} \,\middle|\, \vec{a}\right)$. We generate a level-1 encoding $X^{(1)}$ by drawing $R \leftarrow \{0,1\}^{N \times m}$ and setting $X^{(1)} = \mathsf{Flatten}(I_N + \mathsf{BitDecomp}(R \cdot A))$. We generate the **zero-testing parameter** $\boldsymbol{p}_{zt}$ by first drawing $z \leftarrow \mathcal{R}_q$, $\alpha \leftarrow \chi$ and $\vec{\epsilon}, \vec{\delta} \leftarrow \chi^N$ and then setting $\boldsymbol{p}_{zt} = \alpha(z^\kappa \vec{v} + \vec{\delta}) + \vec{\epsilon}$ where $\vec{v} = \mathsf{Powersof2}(\vec{s})$. We then output $\mathsf{pp} = (\kappa, q, n, m, N, X^{(1)})$ and $\boldsymbol{p}_{zt}$.

Encoding: $C \leftarrow \mathsf{enc}(\mathsf{pp}, \mu, i)$**.** In general, a fresh encoding of a plaintext $\mu \in \mathcal{R}_q$ at level-i will look like the following, where $R \xleftarrow{\$} \{0,1\}^{N \times m}$:

$$C = \mathsf{Flatten}\left(z^{-i}(\mu I_N + \mathsf{BitDecomp}(R \cdot A))\right)$$

Note that encodings are small as they are the result of the $\mathsf{Flatten}$ function. One advantage of this scheme is it allows for the public encoding of plaintext elements because we can make A public. First, a user will encode their plaintext μ to level-i by computing:

$$C = \mathsf{Flatten}(\mu \cdot I_N) \cdot \Pi_{j=1}^{i} X^{(1)}$$

using the multiplication operator of the scheme. Astute readers will note that this is the same as multiplying a ciphertext by a constant in GSW. This is also equivalent to multiplying a level-0 encoding with no noise (the plaintext) to level-i.

Note that we do not need to rerandomize elements encoded in this way because matrix inversion is not homomorphic with respect to the $\mathsf{Flatten}$ function (i.e. an adversary could not just multiply by $(X^{(1)})^{-1}$ or something similar to go down a level).

Zero testing: $\mathsf{isZero}(C, \boldsymbol{p}_{zt}) \stackrel{?}{=} 0/1$**.** To zero-test a (fresh) encoding C of μ at level-κ, compute $\vec{w} = C \cdot \boldsymbol{p}_{zt}$ and output 1 if $||\vec{w}|| < P$ and 0 otherwise. To see why this is correct, note that we have:

$$A\vec{s} = \vec{b} - \vec{a}t = \vec{e}$$

Thus, $\mathsf{BitDecomp}(A)\vec{v} = \vec{e}$ as $\vec{v} = \mathsf{Powersof2}(\vec{s})$. Additionally, $\mathsf{Flatten}(C) \cdot \mathsf{Powersof2}(\vec{s}) = C \cdot \mathsf{Powersof2}(\vec{s}) = C \cdot \vec{v}$, so we have:

$$\begin{aligned} C \cdot \boldsymbol{p}_{zt} &= \alpha z^{\kappa} C\vec{v} + C(\alpha\vec{\delta} + \vec{\epsilon}) \\ &= \alpha(z^{\kappa} z^{-\kappa})(\mu + \mathsf{BitDecomp}(R \cdot A))\vec{v} + C(\alpha\vec{\delta} + \vec{\epsilon}) \\ &= \alpha(\mu\vec{v} + \mathsf{BitDecomp}(R \cdot A)\vec{v}) + C(\alpha\vec{\delta} + \vec{\epsilon}) \\ &= \alpha(\mu\vec{v} + R\vec{e}) + C(\alpha\vec{\delta} + \vec{\epsilon}) \end{aligned}$$

Intuitively, note that this element is large if and only if $\mu \neq 0$ because all elements are small except for $\vec{v}$. Formally, we prove the following lemma in the appendix.

Lemma 1. *This is the lemma that formally outlines the size difference between a zero and nonzero encoding.*

Extraction: $sk \leftarrow \mathsf{ext}(\mathsf{pp}, \boldsymbol{p}_{zt}, C^{(\kappa)})$**.** As in [**?**, **?**] we apply the zero-tester to a level-κ encoding $C^{(\kappa)}$ and collect the most significant bits to extract a random function of the underlying plaintext μ.

Lemma 2. *This is the lemma that states we can extract a random function of λ bits from a level-κ encoding.*

Addition & subtraction. For two encodings C_1 and C_2 at the same level-i, define the addition and subtraction functions as follows:

$$\mathsf{add}(C_1, C_2) = \mathsf{Flatten}(C_1 + C_2)$$
$$\mathsf{sub}(C_1, C_2) = \mathsf{Flatten}(C_1 - C_2)$$

As in the GSW scheme, it is clear that these are both correct homomorphic operations with respect to the zero-tester.

Multiplication. Suppose that C_1 encodes a plaintext μ_1 at level-i and C_2 encodes a plaintext μ_2 at level-j with $i + j \leq \kappa$. Then we define the multiplication operator as follows:

$$\mathsf{mult}(C_1, C_2) = \mathsf{Flatten}(C_1 \cdot C_2)$$

To see that this is correct suppose that $i + j = \kappa$ (i.e., we can zero-test). Then:

$$\begin{aligned}
\mathsf{mult}(C_1, C_2) \cdot \boldsymbol{p}_{zt} &= \alpha(z^\kappa) C_1 C_2 \vec{v} + \mathsf{Flatten}(C_1 \cdot C_2)(\alpha\vec{\delta} + \vec{\epsilon}) \\
&= \alpha(z^\kappa z^{-i}) C_1(\mu_2 \vec{v} + \vec{e_2}) + \mathsf{Flatten}(C_1 \cdot C_2)(\alpha\vec{\delta} + \vec{\epsilon}) \\
&= \alpha(z^\kappa z^{-i}) C_1(\mu_2 \vec{v} + \vec{e_2}) + \mathsf{Flatten}(C_1 \cdot C_2)(\alpha\vec{\delta} + \vec{\epsilon}) \\
&= \alpha(z^\kappa z^{-i} z^{-j})(\mu_1\mu_2 \vec{v} + \mu_2 \vec{e_1} + C_1 \vec{e_2}) \\
&\quad + \mathsf{Flatten}(C_1 \cdot C_2)(\alpha\vec{\delta} + \vec{\epsilon}) \\
&= \alpha(\mu_1\mu_2 \vec{v} + \mu_2 \vec{e_1} + C_1 \vec{e_2}) + \mathsf{Flatten}(C_1 \cdot C_2)(\alpha\vec{\delta} + \vec{\epsilon})
\end{aligned}$$

Note that this expression will be small if and only if $\mu_1\mu_2 = 0$ (and μ_2 is also small).

3.1 Asymmetric version

As in CLT and GGH we can instead compute z_i for $i \in [\kappa]$ and encode plaintexts to specific index sets $S \subseteq [\kappa]$ rather than levels. This is required for some applications like obfuscation.

3.2 Setting parameters

As in homomorphic encryption schemes and multilinear maps, the size of the noise at level-κ is dependent on the size of the initial error B_e, the multilinearity level κ, and the initial size of plaintexts B_μ. Specifically, we have the noise bounded by:

$$B_\kappa = (nN \cdot B_\mu)^\kappa \cdot B_e \tag{1}$$

Then, when we apply the zero-testing element at level-κ we have:

$$C\boldsymbol{p}_{zt} = \alpha(\mu\vec{v} + \vec{e_\circ} + C\vec{\delta}) + C\vec{\epsilon} \tag{2}$$

where the size of $e_\circ$ is bounded by B_κ, as in (**??**). In the case of $\mu = 0$ we then have the size of (**??**) bounded by $B_\alpha(B_\kappa + nNB_\delta) + nNB_\epsilon$.

4 Security

4.1 Graded DDH

In this section we instantiate the Graded Decisional Diffie-Hellman assumption for our scheme, introduced in [**?**] as an analogue of the multilinear decisional Diffie-Hellman assumption. For a full definition of graded encoding schemes, refer to Appendix A.

Definition 1 (κ-GDDH)**.** *For a security parameter $\lambda \in \mathbb{N}$ the κ-Graded Decisional Diffie-Hellman assumption is to distinguish between the distributions $\mathcal{D}_0[(\mathsf{pp}, \boldsymbol{p}_{zt})]$ and $D_1[(\mathsf{pp}, \boldsymbol{p}_{zt})]$ for $(\mathsf{pp}, \boldsymbol{p}_{zt}) \leftarrow \mathsf{setup}(\lambda, \kappa)$ where:*

$$\mathcal{D}_0[(\mathsf{pp}, \boldsymbol{p}_{zt})] = \left\{ \left(\{C_i\}_{i\in[\kappa+1]}, \gamma \right) \;\middle|\; \begin{array}{ll} \forall i \in [\kappa+1], & U_i \leftarrow \mathsf{samp}(\mathsf{pp}), \\ & C_i \leftarrow \mathsf{enc}(\mathsf{pp}, 1, U_i) \\ \gamma \leftarrow \mathsf{ext}(\mathsf{pp}, \boldsymbol{p}_{zt}, U_{\kappa+1} \cdot \Pi_{i\in[\kappa]} C_i) & \end{array} \right\}$$

and

$$\mathcal{D}_1[(\mathsf{pp}, \boldsymbol{p}_{zt})] = \left\{ \left(\{C_i\}_{i\in[\kappa+1]}, \gamma \right) \;\middle|\; \begin{array}{ll} \forall i \in [\kappa+1], & U_i \leftarrow \mathsf{samp}(\mathsf{pp}), \\ & C_i \leftarrow \mathsf{enc}(\mathsf{pp}, 1, U_i) \\ \gamma \leftarrow \{0,1\}^\lambda & \end{array} \right\}$$

We conjecture that this problem is hard for our scheme for reasoning given below.

4.2 Overview

Let n, q and χ be set such that the $\mathrm{RLWE}_{n,q,\chi}$ assumption holds. Let $\mathcal{D}_\alpha, \mathcal{D}_\delta, \mathcal{D}_\epsilon$ be discrete Gaussian distributions on $\mathbb{Z}^n$ with widths $\sigma_\alpha, \sigma_\delta, \sigma_\epsilon$ respectively, and let $\ell = O(\log q)$.

The security of our scheme relies on hiding t and z. With these parameters, an adversary could decrypt an encoding at any level. We will make a few RLWE-like assumptions based on these parameters. Note that in all of these assumptions, the "public key" α is not given out like in the classical RLWE problem, which may increase security (though is not known to be necessary).

4.3 Previous attacks

It is not hard to see that our construction avoids the zeroizing attacks ([**?**]) on the [**?**] scheme as our encodings and zero-testing elements simply do not support the right operations for those attacks to work (it is helpful to note that the Flatten function only preserves some homomorphism with respect to Powersof2 vectors, which further restricts the sorts of useful operations that can be performed on encodings).

The weak-DL attack on GGH and the variant matrix-GGH ([**?**, **?**]) worked by attacking the private modulus g, which is a small element of the ring R. This fails in our scheme because do not rely on encodings being small representatives of cosets of $\mathcal{I}_g$—we achieve small encodings (and thus bounded noise growth) through the repeated application of the Flatten function, so our "ideal" is generated by a "uniform" element, namely, a RLWE instance. We also have not been able to attack the scheme in the event that A is made public.

Additionally, we may have also reduced the number of potential vulnerabilities by not publishing level-0 encodings which we do not need for sampling unlike in e.g. [**?**].

4.4 Reduction attack on the RLWE "public key"

One way the scheme could be attacked is through the search-decision equivalence for RLWE from [**?**]. That is, if there exists an oracle $\mathcal{O}$ to decide that $(a, b = at + e)$ is drawn from RLWE parameters or from a uniform distribution, then the secret t can be recovered. In our scheme the zero-testing parameter $\boldsymbol{p}_{zt}$ would act as such an oracle: an adversary could encode 0 to level-0 using a key A' drawn from a uniform distribution and similarly using the original key A. Then, bringing both encodings to level-κ using the public parameters, she could then zero test both. With high probability the uniform-keyed encoding will be "large" and the RLWE-keyed encoding will be small when tested, thus providing the adversary the desired oracle.

This attack does not apply to our scheme because the reduction runs in $\mathsf{poly}(q)$ time, but we need $q > 2^\lambda$ for correctness of the zero-testing procedure. Additionally, we do not publish A as part of the public parameters.

4.5 Attacks on the zero-testing element

We now draw our attention to using the zero-testing element in breaking the scheme, as is done in all attacks on previous schemes. Note that our zero-testing element looks like the following:

$$\begin{aligned}\boldsymbol{p}_{zt} &= \alpha(z^\kappa \vec{v} + \vec{\delta}) + \vec{\epsilon} \\ &= \alpha(z^\kappa \mathsf{Powersof2}(\vec{s}) + \vec{\delta}) + \vec{\epsilon}\end{aligned}$$

$$= \begin{pmatrix} \alpha(z^\kappa + \delta_1) + \epsilon_1 \\ \alpha(z^\kappa 2 + \delta_2) + \epsilon_2 \\ \vdots \\ \alpha(z^\kappa 2^{\ell-1} + \delta_\ell) + \epsilon_\ell \\ -\alpha(tz^\kappa + \delta_{\ell+1}) + \epsilon_{\ell+1} \\ \vdots \\ -\alpha(tz^\kappa 2^{\ell-1} + \delta_{2\ell}) + \epsilon_{2\ell} \end{pmatrix}$$

Thus, we need the following assumption to hold so that we protect the parameters z^κ and t. Note that this assumption is very similar to RLWE, except that for each new sample we use a "small" perturbation of the original secret—if δ_i was drawn from a uniform distribution, this would reduce to RLWE directly.

Assumption 1. *Draw $s \leftarrow \mathcal{R}_q$, $\alpha \leftarrow \mathcal{D}_\alpha$, and $(\delta_i, \epsilon_i) \leftarrow \mathcal{D}_\delta \times \mathcal{D}_\epsilon$ for $i \leq \ell$. Then from the vector $\vec{p} = \big(\alpha(s + \delta_i) + \epsilon_i\big)_{i=1}^{\ell} \in \mathcal{R}_q^\ell$, recover s.*

4.6 Getting parameters from zero-tested encodings

A zero-tested encoding at level-i will look like the following, where $\vec{e_\circ}$ is the error after some number of operations:

$$C\boldsymbol{p}_{zt} = \alpha z^{\kappa-i}(\mu\vec{v} + \vec{e_\circ}) + C(\alpha\vec{\delta} + \vec{\epsilon}) \in \mathcal{R}_q^N$$

which we can rewrite as:

$$C\boldsymbol{p}_{zt} = \alpha z^{\kappa-i}(\mu\vec{v} + \vec{e_\circ} + C\vec{\delta}) + C\vec{\epsilon}$$

Thus, we need the following assumption to hold in the case of $i < \kappa$:

Assumption 2. *Draw $s \leftarrow \mathcal{R}_q$, $\alpha \leftarrow \mathcal{D}_\alpha$, and $(\delta_i, \epsilon_i) \leftarrow \mathcal{D}_\delta \times \mathcal{D}_\epsilon$ for $i \leq \ell$. Then from the vector $\vec{p} = \left(\alpha s \delta_i + \epsilon_i\right)_{i=1}^{\ell} \in \mathcal{R}_q^\ell$, recover s.*

In the case of $\mu = 0$ we have:

$$C\boldsymbol{p}_{zt} = \alpha z^{\kappa - i}(\vec{e_\circ} + C\vec{\delta}) + C\vec{\epsilon} \tag{3}$$

So the following assumption needs to hold in the case of $i = \kappa$:

Assumption 3. *Draw $\alpha \leftarrow \mathcal{D}_\alpha$, and $(\delta_i, \epsilon_i) \leftarrow \mathcal{D}_\delta \times \mathcal{D}_\epsilon$ for $i \leq \ell$. Then from the vector $\vec{p} = \left(\alpha \delta_i + \epsilon_i\right)_{i=1}^{\ell} \in \mathcal{R}_q^\ell$, recover either δ_i or ϵ_i.*

We could also let $C = \mathsf{BitDecomp}(A)$, because A is public, giving us:

$$C\boldsymbol{p}_{zt} = \alpha z^{\kappa}(\vec{e} + \mathsf{BitDecomp}(A)\vec{\delta}) + \mathsf{BitDecomp}(A)\vec{\epsilon}$$

Which reduces to Assumption **??**.

4.7 Some additional notes

Though none of our assumptions reduce directly to RLWE and nor does the entire scheme's security reduce to any of our variants, we remain confident in the scheme's security based on the fact that the application of the zero-testing parameter only produces instances of these RLWE variants that we presume to be hard. That is, encodings of 0 at any level do not seem to present a problem as in previous work. It is our hope that any successful attack on the scheme would then procure new insight into the standard RLWE problem.

Additionally, we have not been able to break the scheme in the case that α is public (acting as a RLWE public key).

5 An application: N-way key exchange

We now present the canonical use-case for our scheme: N-way non-interactive key exchange, which we outline in the following steps.

1. Let $\kappa = N - 1$. A trusted third party obtains the public parameters $(\mathsf{pp}, \boldsymbol{p}_{zt}) \leftarrow \mathsf{setup}(1^\lambda, 1^\kappa)$ and subsequently distributes the parameters to each player.
2. Each player i runs the $\mathsf{samp}(\mathsf{pp})$ algorithm to obtain a level-0 encoding c_i of a random plaintext μ_i. Then the player computes $C_i \leftarrow \mathsf{enc}(\mathsf{pp}, 1, c_i)$ and publishes this level-1 encoding.
3. The ith player now computes the level-κ encoding
$$U_i = c_i \cdot \Pi_{j \neq i \in [N]} C_j$$
using the multiplication operator of the scheme.
4. Finally, each player computes $s \leftarrow \mathsf{ext}(\mathsf{pp}, \boldsymbol{p}_{zt}, U_i)$ to extract the shared secret s.

Correctness follows from Lemma **??**, whereas security follows from the hardness of the GDDH assumption for our scheme.

References

[BF01] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. In *CRYPTO*, volume 2139 of *Lecture Notes in Computer Science*, pages 213-229, 2001.

[BS03] Dan Boneh and Alice Silverberg. Applications of multilinear forms to cryptography. *Contemporary Mathematics*, 324:71?90, 2003.

[BWZ14] Dan Boneh, David J. Wu, and Joe Zimmerman. Immunizing multilinear maps against zeroizing attacks. Cryptology ePrint Archive, Report 2014/930, 2014. http://eprint.iacr.org/.

[CGH+15] Jean-Sebastien Coron and Craig Gentry and Shai Halevi and Tancrede Lepoint and Hemanta K. Maji and Eric Miles and Mariana Raykova and Amit Sahai and Mehdi Tibouchi. Zeroizing Without Low-Level Zeroes: New MMAP Attacks and Their Limitations. Cryptology ePrint Archive, Report 2015/596, 2015. http://eprint.iacr.org/. To appear at CRYPTO 2015.

[CHL+14] Jung Hee Cheon, Kyoohyung Han, Changmin Lee, Hansol Ryu, Damien Stehl. Cryptanalysis of the multilinear maps over the integers. Cryptology ePrint Archive, Report 2014/906, 2014. http://eprint.iacr.org/. To appear at EUROCRYPT 2015.

[CLT13] Jean-Sebastien Coron, Tancrede Lepoint, and Mehdi Tibouchi. Practical multilinear maps over the integers. In *CRYPTO*, pages 476-493, 2013.

[CLT14] Jean-Sbastien Coron, Tancrde Lepoint, and Mehdi Tibouchi. Cryptanalysis of two candidate fixes of multilinear maps over the integers. Cryptology ePrint Archive, Report 2014/975, 2014. http://eprint.iacr.org/.

[CLT15] Jean-Sbastien Coron, Tancrde Lepoint, and Mehdi Tibouchi. New multilinear maps over the integers. In R. Gennaro and M. Robshaw, editors, CRYPTO 2015, Lecture Notes in Computer Science. Springer, 2015. To appear.

[GGH13a] Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In *EUROCRYPT 2013*. Springer Berlin Heidelberg, 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. Cryptology ePrint Archive, Report 2013/451, 2013. http://eprint.iacr.org/.

[GGH14a] Craig Gentry, Sergey Gorbunov, and Shai Halevi. Graph-induced multilinear maps from lattices. Cryptology ePrint Archive, 2014:645, 2014. To appear at TCC 2015.

[GGH+14b] Craig Gentry, Shai Halevi, Hemanta K. Maji, and Amit Sahai. Zeroizing without zeroes: Cryptanalyzing multilinear maps without encodings of zero. Cryptology ePrint Archive, Report 2014/929, 2014. http://eprint.iacr.org/.

[GSW13] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic Encryption from Learning with Errors: Conceptually-Simpler, Asymptotically-Faster, Attribute-Based. In Advances in Cryptology—CRYPTO 2013, R. Canetti and J. Garay, Eds., vol. 8042 of Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2013, pp. 75-92.

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

[KGV14] Alhassan Khedr, Glenn Gulak, and Vinod Vaikuntanathan, SHIELD: Scalable Homomorphic Implementation of Encrypted Data-Classifiers, Cryptology ePrint Archive, Report 2014/838, 2014. http://eprint.iacr.org/.

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

A Graded Encoding Schemes

B Proof of Lemma ??

C Proof of Lemma ??