Candidate Multilinear Maps from LWE-variants

Andrew H. Russell[*] Nigel Smart[†]

August 10, 2015

Abstract

Multilinear maps are widely used cryptographic primitives in key exchange, functional encryption, obfuscation, and more. However, the only constructions we have are based on non-standard assumptions and several have been successfully attacked in various ways. We propose a new multilinear map from the GSW homomorphic encryption scheme that has its security based on learning with errors-esque (LWE) problems. Our construction exploits certain asymmetries and properties of decrypted elements in GSW. These features allow us to hide the decryption key and publish it as a zero-tester. Though the security of our construction does not fully reduce to standard assumptions, like LWE, we present our result as a step forward in basing multilinear maps on more reasonable assumptions.

1 Introduction

Multilinear maps. Since the use of bilinear pairings to construct identity-based encryption by Boneh and Franklin [BF01] there has been interest in the natural extension to cryptographic maps with multilinearity $\kappa \geq 2$. Boneh and Silverberg [BS03] outlined numerous applications for multilinear maps, but were pessimistic about finding constructions in the field of algebraic geometry. Since then, there has been some progress in the field. Garg, Gentry, and Halevi [GGH13a] first proposed the idea of *graded encoding schemes*, which approximate true multilinear maps, and gave a candidate based on ideal lattices (referred to as the GGH construction). Subsequently, Coron, Lepoint, and Tibouchi gave a similar construction but over the integers (the CLT and CLT15 constructions [CLT13, CLT15]), and Gentry, Gorbunov, and Halevi proposed a construction where the structure of the map is dependent on an underlying acyclic graph [GGH14a].

Multilinear maps have a range of applications, such as an N-way Diffie-Hellman key exchange, broadcast encryption with short keys [BS03] or functional encryption (without obfuscation) [BLR$^+$15]. Arguably the most important and prominent application that motivates the use of multilinear maps is general program obfuscation, originally outlined in [GGH$^+$13b] using the GGH scheme, due to its prolific use as a basis for modern cryptographic protocols.

Cryptanalysis of current constructions. Unfortunately, the security of these graded encoding schemes is not a well-understood problem, and there has yet to be a candidate that is based on more standard assumptions. The original CLT and GGH candidates have been successfully attacked

[*] Pomona College, ahr12012@mymail.pomona.edu. Work performed while at the University of Bristol.

[†] nigel@cs.bris.ac.uk

[CHL$^+$14, CGH$^+$15, HJ15]. The original CLT scheme was broken completely [CHL$^+$14, CGH$^+$15]. The GGH construction has a weak-discrete log attack, and a total break in the particular use-case of key exchange [CGH$^+$15, HJ15]. There have been multiple attempts to fix the CLT scheme and avoid these so-called "zeroizing attacks" [BWZ14, CLT14, CLT15]. However, only the updated CLT construction [CLT15] seems be successful in avoiding these attacks at the cost of some efficiency and increased complexity. The graph-induced construction in [GGH14a] is known to be insecure in some situations (particularly those involving encodings of zero) through the "approximate trapdoor" attacks outlined in the original paper, which makes it harder to reason about security in various applications, as it must be proven that those situations do not arise in each specific use of the map.

Our contributions. We propose a new multilinear map based on the GSW homomorphic encryption scheme [GSW13]. The primary advantage of our scheme is that sensitive parameters—the zero-tester, zero-tested encodings, encodings of 0—are either the products of (unknown) uniform matrices or instances of a LWE-esque problem. In particular, our candidate completely avoids the zeroizing attacks used to attack the CLT and GGH schemes [CHL$^+$14, CGH$^+$15, HJ15]. While a full reduction to standard assumptions does not seem possible at this point, we present our construction both as a step forward in securing multilinear maps under standard assumptions and as an alternative post-zeroizing map to the CLT15 construction.

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 strategy of basing graded encoding schemes on homomorphic encryption schemes as is done in all current constructions; essentially, what security problems arise when zero-testing functionality is added to a HE scheme. An additional issue with current multilinear map constructions is efficiency. None of the current candidates, including this one, have reasonable parameters for use in the real world. Any improvements would have a large impact on the practical viability of flagship applications that rely heavily on multilinear maps like obfuscation.

Additionally, though there have been several results securing some applications in generic models, the rise of zeroizing attacks has questioned the validity of these assumptions. It would be useful to prove certain protocols secure under weaker assumptions, such as those presented here.

Overview of the GSW HE scheme. The encoding of elements and homomorphic operations in our scheme borrow heavily from the GSW homomorphic encryption scheme. We recall the main details of the scheme in broad strokes here. Ciphertexts in GSW look like the following, where $\mu \in \mathbb{Z}_q$ is the plaintext message and A is the public key (a "uniform" matrix) with private key $\vec{s}$ (a vector such that $A \cdot \vec{s} = \vec{e}$, a small vector) hiding the plaintext with some noise R (a binary matrix):

$$C = \mathsf{Flatten}(\mu I_N + \mathsf{BitDecomp}(R \cdot A))$$

We will define the Flatten and BitDecomp functions more precisely later, but for now we simply observe that for any matrix B, $\mathsf{Flatten}(B)$ is always a matrix with 0/1 entries, $\mathsf{Flatten}(B) \cdot \vec{v} = B\vec{v}$, and $\mathsf{BitDecomp}(B) \cdot \vec{v} = B\vec{s}$ where $\vec{v}$ is a particular type of vector that exhibits these properties.

More precisely, we have:

$$\begin{aligned} C \cdot \vec{v} &= \mathsf{Flatten}(\mu I_N + \mathsf{BitDecomp}(R \cdot A)) \cdot \vec{v} \\ &= (\mu I_N + \mathsf{BitDecomp}(R \cdot A)) \cdot \vec{v} \\ &= \mu \vec{v} + R \cdot A \cdot \vec{s} \\ &= \mu \vec{v} + R \cdot \vec{e} \\ &= \mu \vec{v} + error \end{aligned}$$

The plaintext μ can then be recovered because $R \cdot \vec{e}$ is small.

Getting a zero-testing element. The main thing to note about the GSW scheme is that the decryption key $\vec{v}$ exhibits some interesting properties that allow us some asymmetry with respect to ciphertexts, plaintexts, and noise growth. To illustrate, we recall why GSW ciphertexts are multiplicatively homomorphic (with respect to the decryption key):

$$\begin{aligned} C_1 C_2 \cdot \vec{v} &= C_1 \cdot (\mu_2 \vec{v} + \vec{e_2}) \\ &= \mu_2 \mu_1 \vec{v} + \mu_2 \vec{e_1} + C_1 \vec{e_2} \end{aligned}$$

Note how the resulting noise factor is dependent on the plaintext μ_2 and the error in the first ciphertext $\vec{e_1}$. This is a nice optimization that can help keep error terms small if $\mu_2 = 0$; however, we can exploit this property ourselves by creating a ciphertext where the error $\vec{e_1}$ is large (uniformly random). That way, if our plaintext $\mu_2 = 0$, then the result properly "decrypts," whereas if $\mu_2 = 0$ then the result will be garbage and unusable to an adversary, thus giving us a zero-testing element for the scheme. To do this without breaking certain necessary hardness assumptions for multilinear maps we need to accomplish two things: (a) hide the decryption key and (b) require that a user multiply the ciphertext C on the left by our "malformed ciphertext" C' that has "too much noise." We do this by generating a private random matrix T and changing our encodings to be $C = TC_\mu T^{-1}$, where C_μ is the original GSW ciphertext. We can then take our malformed ciphertext to be $C' = \mathsf{BitDecomp}(U) \cdot T^{-1}$, for a uniformly random U, and our hidden decryption key to be $T(\vec{v} + \vec{\epsilon})$ for a small vector $\vec{\epsilon}$. This gives us the desired properties:

$$C' \cdot C \cdot T(\vec{v} + \vec{\epsilon}) = \mu(U\vec{s}) + \vec{e} + C_\mu \vec{\epsilon}$$

Of course, this is simply a starting point; for example, we also introduce the idea of levels through an element z, as in other multilinear schemes, so that we can only zero-test at a specific level-κ.

2 Preliminaries

For a vector $\vec{v} \in \mathbb{Z}^N$ let $\|\vec{v}\|_k$ denote the ℓ_k-norm of the vector for any $k \in \mathbb{N} \cup \{\infty\}$. For any $m \in \mathbb{N}$ we denote the set $\{1, 2, \ldots, m\}$ by $[m]$.

2.1 Bit operations

The GSW paper outlines a few operations on vectors. For some k and q, let $\ell = \lceil \log q \rceil$ and $N = k\ell$, and let $\vec{a}$ be a k-dimensional vector of ring elements in $\mathbb{Z}_q$. Then, the bit-decomposition function $\mathsf{BitDecomp} : \mathbb{Z}_q^k \to \mathbb{Z}_q^N$ is given by

$$\mathsf{BitDecomp}(\vec{a}) = (a_{1,0}, \ldots, a_{1,\ell-1}, \ldots, a_{k,0}, \ldots, a_{k,\ell-1})$$

where $a_{i,j}$ is the j-th bit in the i-th entry of $\vec{a}$. That is, we decompose a vector $\vec{a}$ that has ℓ-bit coefficients into an N-dimensional vector with 0/1 coefficients. $\mathsf{BitDecomp}^{-1} : \mathbb{Z}_q^N \to \mathbb{Z}_q^k$ is the inverse of this operation. Specifically, for a vector $\vec{a} = (a_{1,0}, \ldots, a_{1,\ell-1}, \ldots, a_{k,0}, \ldots, a_{k,\ell-1}) \in \mathbb{Z}_q^N$, we have

$$\mathsf{BitDecomp}^{-1}(\vec{a}) = \left(\sum 2^j a_{1,j}, \ldots, \sum 2^j a_{k,j}\right)$$

Note that we define this function for any vector $\vec{a} \in \mathbb{Z}_q^N$, not just vectors with 0/1 coefficients (which is what $\mathsf{BitDecomp}$ produces). Next, for a vector $\vec{a} \in \mathbb{Z}_q^N$, define $\mathsf{Flatten} : \mathbb{Z}_q^N \to \mathbb{Z}_q^N$ by composing these functions:

$$\mathsf{Flatten}(\vec{a}) = \mathsf{BitDecomp}(\mathsf{BitDecomp}^{-1}(\vec{a}))$$

Finally, define $\mathsf{Powersof2} : \mathbb{Z}_q^k \to \mathbb{Z}_q^N$ by

$$\mathsf{Powersof2}(\vec{a}) = (a_1, 2a_1, \ldots, 2^{\ell-1}a_1, \ldots, a_k, 2a_k, \ldots, 2^{\ell-1}a_k)$$

As in GSW, the important properties of these operations are:

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

We extend these operations to matrices by performing them on each row. Thus, for matrices $A \in \mathbb{Z}_q^{N \times k}$, $C \in \mathbb{Z}_q^{N \times N}$ and a vector $\vec{b} \in \mathbb{Z}_q^k$ 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 Flatten function keeps our encodings small. As in GSW, we set $k = (n+1)$ where n is set to produce secure LWE instances.

2.2 GSW encryption

In this section we define a few procedures that correspond to encryption and decryption in GSW. We set $N = (n+1)\lceil \log q \rceil$ wherever we have parameters n and q.

$(\vec{v}, A) \leftarrow \mathsf{generate}(1^\lambda, n, m, q, \chi)$: For parameters $\lambda, n, m, q \in \mathbb{N}$ and χ, an error distribution on $\mathbb{Z}_q^{m \times 1}$, generate outputs a GSW key pair $(\vec{v}, A) \in \mathbb{Z}_q^N \times \mathbb{Z}_q^{m \times n}$, , by drawing $B \leftarrow \mathbb{Z}_q^{m \times n}$, $\vec{t} \leftarrow \mathbb{Z}_q^{n \times 1}$, and $\vec{e} \leftarrow \chi$. Set $\vec{b} = B\vec{t} + \vec{e}$ and $\vec{s} = (1, -t_1, -t_2, \ldots, -t_n)^T \in \mathbb{Z}_q^{(n+1) \times m}$. Set $\vec{v} = \mathsf{Powersof2}(s) \in \mathbb{Z}_q^N$ and $A = \left(\vec{b} \mid B\right) \in \mathbb{Z}_q^{m \times (n+1)}$. Finally, output $(\vec{v}, A)$. Note that $A\vec{s} = \vec{b} - B\vec{t} = \vec{e}$. We use this fact for decryption.

$C \leftarrow \mathsf{encrypt}(\mu, A, m)$: For a plaintext $\mu \in \mathbb{Z}_q$, a public key $A \in \mathbb{Z}_q^{m \times (n+1)}$ as output by the generate function, and dimension $m \in \mathbb{N}$, encrypt outputs $\mathsf{Flatten}(\mu I_N + \mathsf{BitDecomp}(R \cdot A))$ where $R \leftarrow \{0,1\}^{N \times m}$. When A and m are unambiguous from context we will write $E(\mu) = \mathsf{encrypt}(\mu, A, m)$.

$\mu \leftarrow \mathsf{decrypt}(C, \vec{v})$: Given a ciphertext C of a plaintext μ and a decryption key $\vec{v}$ as output by the $\mathsf{generate}$ function, $\mathsf{decrypt}$ recovers μ from the product $C\vec{v}$.

Note that decryption is possible because $\vec{v} = \mathsf{Powersof2}(\vec{s})$ and $A\vec{s} = \vec{b} - B\vec{t} = \vec{e}$, so by the properties of the bit operations above, $\mathsf{BitDecomp}(R \cdot A)\vec{v} = R\vec{e}$ for any $R \in \mathbb{Z}_q^{N \times m}$. Thus, as we can write $C = \mathsf{Flatten}(\mu I_N + \mathsf{BitDecomp}(R \cdot A))$ for a random binary matrix $R \leftarrow \{0,1\}^{N \times m}$, we have:

$$\begin{aligned} C\vec{v} &= (\mu I_N + \mathsf{BitDecomp}(R \cdot A))\vec{v} \\ &= \mu\vec{v} + R\vec{e} \end{aligned}$$

Thus, as $\vec{e}$ is small and the first ℓ coefficients of $\vec{v}$ are of the form 2^i for $0 \leq i \leq \ell - 1$, we can recover μ.

From [GSW13] we have a couple important homomorphic properties, which we summarize in the following lemma:

Lemma 1 ([GSW13])**.** *Let $(\vec{v}, A) \leftarrow \mathsf{generate}(1^\lambda, n, m, q, \chi)$ for some λ, n, m, q, χ. Then addition and multiplication are homomorphic with respect to* $\mathsf{decrypt}$ *and* $\mathsf{encrypt}$*:*

- $\mathsf{decrypt}(E(\mu_1 + \mu_2), \vec{v}) = \mathsf{decrypt}(E(\mu_1) + E(\mu_2), \vec{v})$
- $\mathsf{decrypt}(E(\mu_1\mu_2), \vec{v}) = \mathsf{decrypt}(E(\mu_1) \cdot E(\mu_2), \vec{v})$

It is obvious that the first property holds; we briefly recall the proof from [GSW13] that shows the second does as well:

$$\begin{aligned} C_1 \cdot C_2\vec{v} &= C_1(\mu_2\vec{v} + \vec{e_2}) \\ &= \mu_2(\mu_1\vec{v} + \vec{e_1}) + C_1\vec{e_2} \\ &= \mu_1\mu_2\vec{v} + \mu_2\vec{e_1} + C_1\vec{e_2} \end{aligned}$$

Clearly, we can also recover $\mu_1\mu_2$ from this as long as μ_2 and the errors $\vec{e_1}, \vec{e_2}$ are all small relative to q because $C_1 \in \{0,1\}^{N \times N}$ is the result of the $\mathsf{Flatten}$ function.

We also have a brief security result from [GSW13, Reg05]:

Lemma 2 ([GSW13, Reg05])**.** *Let params $= (n, q, \chi, m)$ be such that the $LWE_{n,q,\chi}$ assumption holds. Then, for $m = O(n \log q)$, and A, R as generated above, the joint distribution $(A, R \cdot A)$ is computationally indistinguishable from uniform over $\mathbb{Z}_q^{m \times (n+1)} \times \mathbb{Z}_q^{N \times (n+1)}$.*

3 Our construction

We are now ready to outline our map, which is an instantiation of a graded encoding scheme as in prior works [CLT13, GGH13a, GGH14a, CLT15]. Refer to Appendix A for an explicit definition of graded encoding schemes.

In general, a fresh encoding of a plaintext $\mu \in \mathbb{Z}_q$ at level-i will look like the following, where $z \leftarrow \mathbb{Z}_q$ and $T \leftarrow \mathbb{Z}_q^{N \times N}$:

$$C = z^{-i}T \cdot E(\mu) \cdot T^{-1}$$

recalling that $E(\mu) = \mathsf{Flatten}\left(\mu I_N + \mathsf{BitDecomp}(R \cdot A)\right))$ is a GSW encryption of μ. Thus, encodings are simply GSW ciphertexts (which are small, as they are a result of the $\mathsf{Flatten}$ function) pre/post-multiplied by a random matrix T and its inverse and by the parameter z^{-i} to introduce the notion of levels. For an encoding C of a plaintext μ we will denote the corresponding GSW encryption by C_μ. Let $\vec{1}$ denote the row vector $(1, 1, \ldots, 1) \in \mathbb{Z}_q^{1\times N}$.

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. Let n, q, χ be such that the $\mathrm{LWE}_{n,q,\chi}$ assumption holds for a security level of λ. Set $m = O(n \log q)$ and $N = (n+1)\ell$ where $\ell = \lceil \log q \rceil$. Let χ be an error distribution on $\mathbb{Z}_q$. α is the size (in bits) of freshly encoded or sampled plaintext elements. We generate the GSW key pair $(\vec{v}, A) \leftarrow \mathsf{generate}(1^\lambda, n, m, \chi)$. We generate the zero-testing parameter $\boldsymbol{p}_{zt}$ by first drawing uniform matrices $U \leftarrow \mathbb{Z}_q^{N\times(n+1)}$ and $T \leftarrow \mathbb{Z}_q^{N\times N}$ and setting

$$\vec{w} = \vec{1} \cdot \mathsf{BitDecomp}(U) \cdot T^{-1}$$

Then draw a uniform element $z \leftarrow \mathbb{Z}_q$ and a "somewhat small" element $\vec{\epsilon} \leftarrow (\mathcal{D}_{\mathbb{Z},\sigma})^N$ for a parameter σ. We then set

$$\vec{u} = z^\kappa T \cdot (\vec{v} + \vec{\epsilon})$$

We generate the sampling/encoding parameters

$$B_i = T \cdot E(2^i) \cdot T^{-1} \quad \forall i \in \{0, 1, \ldots, \alpha - 1\}$$

We generate a level-1 encoding of 1

$$Y = z^{-1} T \cdot E(1) \cdot T^{-1}$$

For a parameter τ we generate the rerandomization parameters

$$X_i = z^{-1} T \cdot E(0) \cdot T^{-1} \quad \forall i \in [\tau]$$

Note that these are level-1 encodings of 0. We output $\mathsf{pp} = (\kappa, q, N, \alpha, \{B_i\}_{i=0}^{\alpha-1}, Y, \{X_i\}_{i=1}^{\tau})$, and $\boldsymbol{p}_{zt} = (\vec{w}, \vec{u})$.

Zero testing: $\mathsf{isZero}(C, \boldsymbol{p}_{zt}) \overset{?}{=} 0/1$**.** To zero-test an encoding $C = z^{-\kappa} T C_\mu T^{-1}$ of μ at level-κ, output 1 if $|\langle \vec{w}, C\vec{u} \rangle| < q^{3/4}$ and 0 otherwise. To see briefly why this is correct, recall the properties of the bit operations and GSW decryption from Section 2. Then we have the following, where $\vec{e}_C$ is the error present in the GSW encryption C_μ:

$$\begin{aligned}
C\vec{u} &= z^{-\kappa} T C_\mu T^{-1} \cdot \vec{u} \\
&= z^{-\kappa} T C_\mu T^{-1} \cdot z^\kappa T \cdot (\vec{v} + \vec{\epsilon}) \\
&= (z^{-\kappa} z^\kappa) T C_\mu (\vec{v} + \vec{\epsilon}) \\
&= T(C_\mu \vec{v} + C_\mu \vec{\epsilon}) \\
&= T(\mu \vec{v} + \vec{e}_C + C_\mu \vec{\epsilon})
\end{aligned}$$

which gives us

$$\begin{aligned}\langle\vec{w}, C\vec{u}\rangle &= \vec{1}\cdot\mathsf{BitDecomp}(U)\cdot T^{-1}T\cdot(\mu\vec{v}+\vec{e}_C+C_\mu\vec{\epsilon})\\ &= \vec{1}\cdot\mathsf{BitDecomp}(U)\cdot(\mu\vec{v}+\vec{e}_C+C_\mu\vec{\epsilon})\\ &= \vec{1}\cdot(\mu(U\vec{s})+\mathsf{BitDecomp}(U)\cdot(\vec{e}_C+C_\mu\vec{\epsilon}))\\ &= \vec{1}\cdot(\mu(big)+small)\end{aligned}$$

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

Lemma 3. *Let our parameters be set as above. In particular, let $C = z^{-\kappa}TC_\mu T^{-1}$ where C_μ is a GSW encryption of $\mu \in \mathbb{Z}_q$. That is, $C\mu\vec{v} = \mu\vec{v}+\vec{e}_C$ for some vector $\vec{e}_C$. Suppose that $B_C > \|C_\mu\|_\infty$, $B_e > \|\vec{e}_C\|_\infty$, $B_\epsilon > \|\vec{\epsilon}\|_\infty$ and $q > N^4(B_e+B_CB_\epsilon)^{4/3}$. Then $|\langle w, C\vec{u}\rangle| < q^{3/4}$ if and only if $\mu = 0$.*

Addition & subtraction. Addition and subtraction are the corresponding matrix operations for two encodings C_1 and C_2 at the same level-i, . As $C_{\mu_1}+C_{\mu_2}$ is a valid GSW ciphertext of $\mu_1+\mu_2$ by Lemma 1, the correctness with respect to zero-testing for $C_1+C_2 = z^{-i}T(C_{\mu_1}+C_{\mu_2})T^{-1}$ follows from Lemma 3. Subtraction follows similarly.

Multiplication. Multiplication in our scheme is simply matrix multiplication. Suppose that C_1 and C_2 are encodings at level-i and level-j, respectively, with $i+j \leq \kappa$ and corresponding GSW ciphertexts C_{μ_1} and C_{μ_2}. First note that

$$\begin{aligned}C_1\cdot C_2 &= (z^{-i}TC_{\mu_1}T^{-1})\cdot(z^{-j}TC_{\mu_2}T^{-1})\\ &= z^{-i-j}T(C_{\mu_1}\cdot C_{\mu_2})T^{-1}\end{aligned}$$

which is a level-$(i+j)$ encoding of $\mu_1\mu_2$ by Lemma 3, because $C_{\mu_1}\cdot C_{\mu_2}$ correctly decrypts to $\mu_1\mu_2$, by Lemma 1, so long as μ_2 is small—for this reason our message space is restricted to control noise growth, as in CLT and GGH.

Sampling: $C^{(0)} \leftarrow \mathsf{samp}(\mathsf{pp})$. We borrow this idea of public sampling/encoding from [BWZ14]. To sample a level-0 encoding, first draw $\mu \leftarrow [0, 2^\alpha)\cap\mathbb{Z}$. Let $\vec{m} = \mathsf{BitDecomp}(\mu) \in \{0,1\}^\alpha$. Then output $C^{(0)} = \sum_{i=0}^{\alpha-1} m_iB_i$, which we can see is a level-0 encoding of μ because each B_i is a level-0 encoding of 2^i.

Rerandomization: $C' \leftarrow \mathsf{rerand}(\mathsf{pp}, C)$. TODO.

Encoding: $C \leftarrow \mathsf{enc}(\mathsf{pp}, C^{(0)}, i)$. To encode a level-0 encoding $C^{(0)}$ at level-i, first compute $Y' \leftarrow \mathsf{rerand}(\mathsf{pp}, Y)$, which will be a rerandomized level-1 encoding of 1. Then output $C = C^{(0)}\cdot(Y')^i$.

Extraction: $sk \leftarrow \mathsf{ext}(\mathsf{pp}, \boldsymbol{p}_{zt}, C^{(\kappa)})$. As in [GGH13a, CLT15] 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 4. *This is the lemma that states we can extract a random function of λ bits from a level-κ encoding.*

3.1 Asymmetric variants

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. We can also break the multiplicative commutativity of encodings (plaintexts would still commute) and enforce a particular order of multiplication by generating many matrices T_i rather than a single matrix T.

3.2 Setting parameters

TODO.

4 Security

4.1 Graded DDH

In this section we instantiate the Graded Decisional Diffie-Hellman assumption for our scheme, introduced in [GGH13a] as the graded encoding scheme analogue of the multilinear decisional Diffie-Hellman assumption in [BS03]. 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 $(\mathsf{pp}, \boldsymbol{p}_{zt}) \leftarrow \mathsf{setup}(1^\lambda, 1^\kappa)$ where:*

$$\mathcal{D}_0[(\mathsf{pp}, \boldsymbol{p}_{zt})] = \left\{ \left(\{C'_i\}_{i\in[\kappa+1]}, \gamma\right) \;\middle|\; \begin{array}{rl} \forall i \in [\kappa+1], & U_i \leftarrow \mathsf{samp}(\mathsf{pp}), \\ & C_i \leftarrow \mathsf{enc}(\mathsf{pp}, 1, U_i) \\ & C'_i \leftarrow \mathsf{rerand}(\mathsf{pp}, C_i) \\ \multicolumn{2}{l}{\gamma \leftarrow \mathsf{ext}(\mathsf{pp}, \boldsymbol{p}_{zt}, U_{\kappa+1} \cdot \prod_{i\in[\kappa]} C_i)} \end{array} \right\}$$

and

$$\mathcal{D}_1[(\mathsf{pp}, \boldsymbol{p}_{zt})] = \left\{ \left(\{C'_i\}_{i\in[\kappa+1]}, \gamma\right) \;\middle|\; \begin{array}{rl} \forall i \in [\kappa+1], & U_i \leftarrow \mathsf{samp}(\mathsf{pp}), \\ & C_i \leftarrow \mathsf{enc}(\mathsf{pp}, 1, U_i) \\ & C'_i \leftarrow \mathsf{rerand}(\mathsf{pp}, C_i) \\ \multicolumn{2}{l}{\gamma \leftarrow \{0,1\}^\lambda} \end{array} \right\}$$

We conjecture that this problem is hard for our scheme.

4.2 Overview

The security of our scheme is based on the LWE problem in a few ways (though a full reduction does not seem possible at this point). First, GSW encryptions are protected by the fact that LWE pairs are indistinguishable from random; we can view (fresh) GSW encryptions as indistinguishable from random binary matrices due to Lemma 1 in [GSW13]. Of course, this hardness assumption necessarily breaks in our case because we have a zero-testing procedure, but in the case that the plaintext is non-zero we cause a “decryption” error so that the plaintext is a multiple of a secret random matrix and cannot be recovered. Zero-tested encodings of zero can also be viewed as LWE instances where the public key is not published and some values are necessarily small. This feature allows us to avoid the zeroizing attacks from before. Thus, we conjecture that problems such as GDDH, SubM, and DLIN are hard in our scheme.

4.3 Zeroizing attacks, SubM, DLIN

Attacks on previous schemes [CHL$^+$14, CGH$^+$15, HJ15] that make use of encodings of 0 fail against our scheme for a number of reasons. The principal advantage that our scheme has over others is that zero-tested encodings of 0 are not exact multiples of noise elements, and instead result in LWE-esque instances with some additive noise. To illustrate, a zero-tested encoding of 0 in [CLT13] or [GGH13a] is of the form hr where h is a "small-ish" noise element, and r is the noise present in the encoding. In our scheme, however, a zero-tested encoding of 0 is of the form

$$\langle \vec{w}, C\vec{u} \rangle = \vec{1} \cdot \mathsf{BitDecomp}(U) \cdot (\vec{e}_C + C_\mu \vec{\epsilon})$$

where $\vec{e}_C$ is the noise present in the encoding (the r from before). Note that this is essentially a LWE-esque instance $C_\mu \vec{\epsilon} + \vec{e}_C$ (except C_μ is unknown to the adversary, unlike in typical LWE instances) multiplied by a uniform binary matrix with the resulting entries summed by the row vector $\vec{1}$. In particular, the noise in the encoding is $\vec{e}_C$, but we have the additive term $C_\mu \vec{\epsilon}$ which breaks the previous attacks that rely on the exact multiples for things like computing GCDs and determinants.

The most recent CLT construction [CLT15] changes the zero-testing parameter so that zero-tested elements have some additive term ax_0 over the integers, due to the hidden modulus x_0, so that the attack fails in a similar way. For that reason the authors conjecture that the subgroup membership (SubM) and decision linear (DLIN) problems are hard in their new version, as the attacks that break those assumptions in the old CLT and GGH no longer apply. Similarly, we believe those problems to be hard in our map.

4.4 The homomorphism of Flatten

It is helpful to note that the Flatten function does not preserve all algebraic structure with respect to Powersof2 vectors. For example, matrix multiplication does not work in this sense:

$$\begin{aligned} \mathsf{Flatten}(AB) \cdot \mathsf{Powersof2}(\vec{v}) &= AB \cdot \mathsf{Powersof2}(\vec{v}) \\ &\neq \mathsf{Flatten}(A) \cdot \mathsf{Flatten}(B) \cdot \mathsf{Powersof2}(\vec{v}) \\ &= \mathsf{Flatten}(A) \cdot B \cdot \mathsf{Powersof2}(\vec{v}) \end{aligned}$$

Thus we would expect the use of this function to only reduce the number of possible algebraic attacks.

4.5 Attacks on the zero-testing element

We first want to rule out that we cannot get useful information out of the zero-testing elements themselves. Note that our zero-testing elements are the following vectors:

$$\begin{aligned} \vec{u} &= z^\kappa T \cdot (\vec{v} + \vec{\epsilon}) \\ \vec{w} &= \vec{1} \cdot \mathsf{BitDecomp}(U) \cdot T^{-1} \end{aligned}$$

where z^κ, T, and U are uniformly random elements. Multiplying them together we get:

$$z^\kappa \vec{1} \cdot (U\vec{s} + \mathsf{BitDecomp}(U) \cdot \vec{\epsilon})$$

4.6 Getting parameters from zero-tested encodings

Though we have avoided the zeroizing attacks of [CHL$^+$14, CGH$^+$15], we now draw our attention to using the zero-testing element in breaking the scheme. A zero-tested encoding at level-i will look like the following, where $\vec{e}_C$ is the error in the GSW encryption after some number of operations:

$$\langle \vec{w}, C\vec{u} \rangle = z^{\kappa - i} \vec{1} \cdot (\mu(U\vec{s}) + \mathsf{BitDecomp}(U) \cdot (\vec{e}_C + C_\mu \vec{\epsilon}))$$

This is a standard "LWE instance" in the case that $\mu \neq 0$ as U is a uniformly random matrix. However, in the case that $\mu = 0$ and $i = \kappa$ we have:

$$\langle \vec{w}, C\vec{u} \rangle = \vec{1} \cdot \mathsf{BitDecomp}(U) \cdot (\vec{e}_C + C_\mu \vec{\epsilon}) \tag{1}$$

We can view this as an LWE instance where the "public" key C_μ is the product of κ GSW encryptions (binary matrices), the secret is drawn from the discrete Gaussian $\mathcal{D}_{\mathbb{Z}^N, \sigma'}$, and the error $\vec{e}_C$ is drawn from the discrete Gaussian $\mathcal{D}_{\mathbb{Z}^N, \sigma}$. We then take a random subset sum of this LWE vector by pre-multiplying with $\vec{1} \cdot \mathsf{BitDecomp}(U)$. We conjecture that it is infeasible to extract useful parameters from this resulting integer based on the hardness of the related problems.

4.7 RLWE variant

We could consider working over the ring $\mathcal{R}_q = \mathcal{R}/q\mathcal{R}$, where $R = \mathbb{Z}[x]/\langle x^n + 1 \rangle$, instead of $\mathbb{Z}_q$ (and let $N = 2\ell$ instead of $N = (n+1)\ell$). We then could set the zero-testing vector $\vec{u} = T \cdot (h\vec{v} + \vec{\epsilon})$ for some small ring element $h \in \mathcal{R}_q$. This would then make zero-tested encodings of 0 look like:

$$\vec{1} \cdot \mathsf{BitDecomp}(U) \cdot (h\vec{e}_C + C_\mu \vec{\epsilon})$$

This would then make the resulting value a ring element that looks like the RLWE-esque instance $hr + e$.

5 An application: N-way key exchange

We now present the canonical use-case for multilinear maps: N-way one round key exchange, which we outline for our construction 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 random level-0 encoding U_i. Then the player computes $C_i \leftarrow \mathsf{enc}(\mathsf{pp}, 1, U_i)$ and publishes this level-1 encoding.
3. The ith player now computes the level-κ encoding
$$C_i' = U_i \cdot \prod_{j \neq i \in [N]} C_j$$
using the multiplication operator of the scheme.
4. Finally, each player computes $s \leftarrow \mathsf{ext}(\mathsf{pp}, \boldsymbol{p}_{zt}, C_i')$ to extract the shared secret s.

Correctness follows from Lemma 4, 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.

[BLR$^+$15] Dan Boneh and Kevin Lewi and Mariana Raykova and Amit Sahai and Mark Zhandry and Joe Zimmerman. Semantically Secure Order-Revealing Encryption: Multi-Input Functional Encryption Without Obfuscation. EUROCRYPT 2015.

[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.

[Reg05] On lattices, learning with errors, random linear codes, and cryptography. In *STOC*, pages 84-93, 2005.

A Graded Encoding Schemes

Graded encoding schemes are an approximation of multilinear maps. In particular, encodings in graded encoding schemes can be randomized rather than deterministic. We first recall the original definition of multilinear maps:

Definition 1 (Multilinear Maps [BS03])**.** *A map $e : G_1^\kappa \to G_2$ is a κ-*multilinear map *if it satisfies the following properties:*

1. *G_1 and G_2 are groups of the same prime order p;*
2. *If $a_1, \dots, a_\kappa \in \mathbb{Z}$ and $g_1, \dots, g_\kappa \in G_1$ then*

$$e(g_1^{a_1}, \dots, g_\kappa^{a_\kappa}) = e(g_1, \dots, g_\kappa)^{a_1 \cdots a_\kappa};$$

3. *If g is a generator of G_1 then $e(g, \dots, g)$ is a generator of G_2.*

Graded encoding schemes are similar, except they give rise to many maps $G_i \times G_j \to G_{i+j}$ through the multiplication operation, and encodings can be randomized. We begin by recalling the definition of graded encoding systems:

Definition 2 (Graded Encoding Systems [GGH13a])**.** *A κ-*graded encoding system *for a ring R is a system of sets $\mathcal{S} = \{S_v^{(\alpha)} \in \{0,1\}^* \mid v \in \mathbb{N}, \alpha \in R\}$ that satisfies the following properties:*

1. *For every $v \in \mathbb{N}$, the sets $\{S_v^{(\alpha)} \mid \alpha \in R\}$ are disjoint;*
2. *There are binary operations $+$ and $-$ such that for every α_1, α_2, $v \in \mathbb{N}$, and every $u_1 \in S_v^{(\alpha_1)}$ and $u_2 \in S_v^{(\alpha_2)}$ we have*

$$u_1 + u_2 \in S_v^{(\alpha_1+\alpha_2)}$$
$$u_1 - u_2 \in S_v^{(\alpha_1-\alpha_2)}$$

where $\alpha_1 + \alpha_2$ and $\alpha_1 - \alpha_2$ are addition and subtraction in the ring R;

3. *There is an (associative) binary operation $\cdot$ such that for every $\alpha_1, \alpha_2 \in R$ and $v_1, v_2 \in \mathbb{N}$ such that $v_1 + v_2 \leq \kappa$ and every $u_1 \in S_v^{(\alpha_1)}$ and $u_2 \in S_v^{(\alpha_2)}$ we have*

$$u_1 \cdot u_2 \in S_{v_1+v_2}^{(\alpha_1 \cdot \alpha_2)}$$

where $\alpha_1 \cdot \alpha_2$ is multiplication in R.

The related definition of graded encoding schemes specifies the public sampling and rerandomization of elements, which we adapt from [CLT15]:

Definition 3 (Graded Encoding Schemes)**.** *A (symmetric) κ-*graded encoding scheme *for a ring R is a system of sets $\mathcal{S} = \{S_v^{(\alpha)} \in \{0,1\}^* \mid v \in \mathbb{N}, \alpha \in R\}$ consists of the following PPT procedures:*

$\mathsf{setup}(1^\lambda, 1^\kappa)$: *For a security parameter λ and multilinearity level κ, output $(\mathsf{pp}, \boldsymbol{p}_{zt})$ where pp is the description of a graded encoding system and $\boldsymbol{p}_{zt}$ is a zero-test parameter;*

$\mathsf{samp}(\mathsf{pp})$: *Outputs a random level-0 encoding $u \in S_0^{(\alpha)}$ for a random $\alpha \in R$;*

$\mathsf{enc}(\mathsf{pp}, i, u)$: *For a level $i \leq k$ and a level-0 encoding $u \in S_0^{(\alpha)}$, outputs a level-i encoding $u' \in S_i^{(\alpha)}$;*

$\mathsf{rerand}(\mathsf{pp}, i, u)$: *For a level-i encoding $u \in S_i^{(\alpha)}$, outputs another level-i encoding $u' \in S_i^{(\alpha)}$ such that for any two $u_1, u_2 \in S_i^{(\alpha)}$ the output distributions of* $\mathsf{rerand}(\mathsf{pp}, i, u_1)$ *and* $\mathsf{rerand}(\mathsf{pp}, i, u_2)$ *are nearly the same;*

$\mathsf{neg}(\mathsf{pp}, u)$: *For a level-i encoding $u \in S_i^{(\alpha)}$, outputs another level-i encoding $u' \in S_i^{(-\alpha)}$;*

$\mathsf{add}(\mathsf{pp}, u_1, u_2)$: *For two level-i encodings $u_1 \in S_i^{(\alpha_1)}$ and $u_2 \in S_i^{(\alpha_2)}$, outputs another level-i encoding $u' \in S_i^{(\alpha_1+\alpha_2)}$;*

$\mathsf{mult}(\mathsf{pp}, u_1, u_2)$: *For a level-i encoding $u_1 \in S_i^{(\alpha_1)}$ and a level-j encoding$u_2 \in S_j^{(\alpha_2)}$ where $i+j \leq \kappa$, outputs a level-$(i+j)$ encoding $u' \in S_{i+j}^{(\alpha_1 \cdot \alpha_2)}$;*

$\mathsf{isZero}(\mathsf{pp}, u, \boldsymbol{p}_{zt})$: *For a level-κ encoding $u \in S_\kappa^{(\alpha)}$ and zero-testing parameter $\boldsymbol{p}_{zt}$, outputs 1 if $\alpha = 0$ and 0 otherwise with negligible probability;*

$\mathsf{ext}(\mathsf{pp}, \boldsymbol{p}_{zt}, u)$: *For a level-k encoding $u \in S_\kappa^{(\alpha)}$ and zero-testing parameter $\boldsymbol{p}_{zt}$, outputs a λ-bit string s such that:*

1. *For any $\alpha \in R$ and $u_1, u_2 \in S_\kappa^{(\alpha)}$, $\mathsf{ext}(\mathsf{pp}, \boldsymbol{p}_{zt}, u_1) = \mathsf{ext}(\mathsf{pp}, \boldsymbol{p}_{zt}, u_2)$;*
2. *The distribution $\{\mathsf{ext}(\mathsf{pp}, \boldsymbol{p}_{zt}, u) \mid \alpha \leftarrow R, u \leftarrow S_\kappa^{(\alpha)}\}$ is nearly uniform over $\{0,1\}^\lambda$.*

B Proof of Lemma 3

By hypothesis we have $q > N^4(B_e + B_C B_\epsilon)^{4/3}$ giving us

$$q^{3/4} > N^3(B_e + B_C B_\epsilon) \tag{2}$$

Now suppose that $\mu = 0$. Then by (2) we have

$$\begin{aligned}
|\,\langle \vec{w}, C\vec{u}\rangle\,| &= |\vec{1} \cdot \mathsf{BitDecomp}(U) \cdot (\vec{e}_C + C_\mu \vec{\epsilon})| \\
&\leq N^2(B_e + NB_C B_\epsilon) \\
&\leq N^3(B_e + B_C B_\epsilon) \\
&< q^{3/4}
\end{aligned}$$

TODO: show other direction.

C Proof of Lemma 4