Obfuscating Conjunctions under Entropic Ring LWE[*]

Zvika Brakerski[†] Weizmann

Vinod Vaikuntanathan[‡] MIT

Hoeteck Wee[§] ENS

Daniel Wichs [¶] Northeastern

Abstract

We show how to securely obfuscate *conjunctions*, which are functions $f(x_1, \ldots, x_n) = \bigwedge_{i \in I} y_i$ where $I \subseteq [n]$ and each literal y_i is either just x_i or $\neg x_i$ e.g., $f(x_1, \ldots, x_n) = x_1 \wedge \neg x_3 \wedge \neg x_7 \cdots \wedge x_{n-1}$. Whereas prior work of Brakerski and Rothblum (CRYPTO 2013) showed how to achieve this using a non-standard object called cryptographic multilinear maps, our scheme is based on an "entropic" variant of the Ring Learning with Errors (Ring LWE) assumption. As our core tool, we prove that hardness assumptions on the recent multilinear map construction of Gentry, Gorbunov and Halevi (TCC 2015) can be established based on entropic Ring LWE. We view this as a first step towards proving the security of additional multilinear map based constructions, and in particular program obfuscators, under standard assumptions.

Our scheme satisfies virtual black box (VBB) security, meaning that the obfuscated program reveals nothing more than black-box access to f as an oracle, at least as long as (essentially) the conjunction is chosen from a distribution having sufficient entropy.

Keywords: Obfuscation, Conjunctions, Evasive Functions, Ring Learning with Errors.

[*]This work was done in part while the authors were visiting the Simons Institute for the Theory of Computing, supported by the Simons Foundation and by the DIMACS/Simons Collaboration in Cryptography through NSF grant CNS-1523467.

[†]Email: `zvika.brakerski@weizmann.ac.il`.

[‡]Email: `vinodv@csail.mit.edu`. Research supported in part by DARPA Safeware grant, NSF grants CNS-1350619 and CNS-1414119, Alfred P. Sloan Research Fellowship, Microsoft Faculty Fellowship, the NEC Corporation, and a Steven and Renee Finn Career Development Chair from MIT.

[§]E-mail: `wee@di.ens.fr`. CNRS, INRIA and Columbia University. Research supported in part by ERC Project aSCEND (639554) and NSF Award CNS-1445424.

[¶]Email: `wichs@ccs.neu.edu`. Research supported by NSF grants CNS-1347350, CNS-1314722, CNS-1413964.

1 Introduction

Program Obfuscation [6, 24, 4, 19] is a central cryptographic primitive which enables one to "encrypt" a program in a way that preserves its input-output behavior, yet hides its inner workings. There are several definitions of what it means to "hide the inner workings" of a program, including the virtual black-box definition and the weaker indistinguishability obfuscation definition of Barak et al. [4]. Rather unfortunately, Barak et al. also showed that *general purpose* virtual black-box obfuscation is unachievable. Still, notwithstanding the bleak outlook projected by this result, several positive results for obfuscation have emerged recently.

In particular, several specific and useful classes of functions have been shown to be virtual black-box obfuscatable. This includes constructions of point function obfuscators either in the random oracle model [6, 29] or assuming exponentially strong one-way functions [34], hyperplane obfuscators assuming strong DDH [7], and very recently conjunction obfuscators and average-case evasive function obfuscators under strong assumptions on multilinear maps [5, 3]. Weakening the definition of obfuscation to an indistinguishability-based notion [4, 23] (called IO obfuscation), Garg, Gentry, Halevi, Raykova, Sahai and Waters [19] showed how to IO-obfuscate any family of polynomial-size circuits.

In this work, we address the question of whether VBB obfuscation of advanced functionalities can be based on standard assumptions. Our contribution is the construction of an average-case virtual black-box obfuscator for conjunctions assuming an entropic version of the Ring learning with errors (Ring LWE) assumption. Our construction uses the techniques of [5] and the recent multilinear map candidate of Gentry, Gorbunov and Halevi [20]. Our main contribution is the *first proof technique* for a non-trivial obfuscator under an assumption related to a standard and well-studied problem (namely, a generalization of Ring LWE to entropic secrets).

Conjunctions. We define conjunctions as functions of the form $f(x_1, \ldots, x_n) = \bigwedge_{i \in I} y_i$ with literals y_i being either x_i or $\neg x_i$ and $I \subseteq [n]$. Alternatively we can think of this as checking that the values $x_i : i \in I$ match some fixed pattern while values outside of I can be arbitrary. Perhaps the simplest way to represent conjunctions, which we will use by default in this work, is as a vector $v \in \{0, 1, \star\}^n$ where we define $F_v(x_1, \ldots, x_n) = 1$ iff for all $i \in [n]$ we have $x_i = v_i$ or $v_i = \star$. We refer to $\star$ as a "wildcard".

Conjunction Obfuscation. A conjunction obfuscator takes as input a conjunction function F_v and outputs an obfuscated program Π_v such that $\Pi_v(x) = F_v(x)$ for all x. Our goal is to achieve virtual black box (VBB) security which says that the code of the program Π_v reveals no more information than having black-box access to an oracle for the function F_v. We relax this requirement by considering a *distributional VBB security*, where we only require the above to hold when v is chosen from a distribution that has sufficient entropy, even when conditioned on the wildcard locations $\{i \ : \ v_i = \star\}$.

Our Results and Assumption. We show how to obfuscate conjunctions with distributional VBB security under a variant of the Ring LWE assumption, which we call entropic Ring LWE.

The Ring LWE assumption (for a ring $\mathcal{R}$) says that, when $s \in \mathcal{R}$ is a random secret ring element then $(\mathbf{A}, s\mathbf{A} + \mathbf{e})$ is indistinguishable from uniform, where $\mathbf{A} \in \mathcal{R}^m$ is random and $\mathbf{e} \in \mathcal{R}^m$ is a short Gaussian error. The entropic Ring LWE assumption says that, when $s_1, \ldots, s_n \in \mathcal{R}$ are random (short) *public* ring elements and $x \in \{0, 1\}^n$ is a *secret* bit-vector chosen from a high entropy distribution, then the Ring LWE assumption holds with $s = \prod_i s_i^{x_i}$ as the secret. See Definition 2.7 for a precise statement.

1.1 Our Techniques

Directed Encoding. We rely on (a special case of) the construction of [20] which can be thought of as a variant of a multilinear map, that we call a directed encoding. For public keys $\mathbf{A}, \mathbf{A}' \in \mathcal{R}^m$, we define an encoding of a short ring element $s \in \mathcal{R}$ under $\mathbf{A} \to \mathbf{A}'$ as a short matrix $\mathbf{R} \in \mathcal{R}^{m \times m}$ such that

$$\mathbf{A}\mathbf{R} = s\mathbf{A}' + \mathbf{e}$$

where $\mathbf{e}$ is short Gaussian error. This allows us to multiply encodings $\mathbf{R}^\times = \mathbf{R}_1 \cdot \mathbf{R}_2$ so that, if $\mathbf{R}_1$ is an encoding of s_1 under $\mathbf{A}_0 \to \mathbf{A}_1$ and $\mathbf{R}_2$ is an encoding of s_2 under $\mathbf{A}_1 \to \mathbf{A}_2$ then $\mathbf{R}^\times$ is an encoding of

$s_1 \cdot s_2$ under $\mathbf{A}_0 \to \mathbf{A}_2$.

Furthermore, we can detect if a value $\mathbf{R}$ is an encoding of 0 under $\mathbf{A} \to \mathbf{A}'$ by checking whether $\mathbf{AR}$ is short. This also allows us to check for equality of encoded values.

Conjunction Obfuscator Construction. To obfuscate a conjunction F_v with $v \in \{0, 1, \star\}^\ell$ we do the following:

- Choose random short ring elements $\{s_{i,b}, r_{i,b} : \ i \in [\ell], b \in \{0, 1\}\}$ subject to $s_{i,0} = s_{i,1}$ if $v_i = \star$.
- Create encodings $R_{i,b}$ of $r_{i,b}$ and encodings $S_{i,b}$ of $r_{i,b} \cdot s_{i,b}$ under $\mathbf{A}_{i-1} \to \mathbf{A}_i$.
- Choose random short ring element $r_{\ell+1}$. Create the encodings $R_{\ell+1}$ of $r_{\ell+1}$ and $S_{\ell+1}$ of $r_{\ell+1} \cdot \prod_{i=1}^{\ell} s_{i,v_i}$ where we let $s_{i,\star} = s_{i,0} = s_{i,1}$ when $v_i = \star$. These encodings are under $\mathbf{A}_\ell \to \mathbf{A}_{\ell+1}$.

Set the obfuscated program to be

$$\Pi_v = (\{S_{i,b}, R_{i,b}\}_{i \in [\ell], b \in \{0,1\}}, R_{\ell+1}, S_{\ell+1})$$

To evaluate Π_v on an input $x \in \{0, 1\}^\ell$ we compute

$$S^* = \left(\prod_{i=1}^{\ell} S_{i,x_i}\right) R_{\ell+1} \quad , \quad R^* = \left(\prod_{i=1}^{\ell} R_{i,x_i}\right) S_{\ell+1}$$

If $F_v(x) = 1$ then both S^* and R^* are encodings of the same value $r_{\ell+1} \prod_{i=1}^{\ell} s_{i,v_i}$ under $\mathbf{A}_0 \to \mathbf{A}_{\ell+1}$ and if $F_v(x) = 0$ then S^*, R^* are extremely unlikely to encode the same value. Therefore, we can compute the output of the program by testing for equality of encoded values.

To argue security, we rely on (entropic) Ring LWE to replace the components of the program Π_v by random elements independent of v. As an important step of the proof, we show that the encodings satisfy a decisional Diffie Hellman (DDH) like security property: Given the encodings of ring elements $r_0, r_0 s_0, r_1, r_1 s_1$ under $\mathbf{A} \to \mathbf{A}'$, one cannot distinguish whether $s_0 = s_1$ or whether s_0, s_1 are independent.

The Directed Encoding Abstraction. In the body of the paper, we present the construction through the abstraction of directed encoding schemes (as opposed to directly, starting from Ring LWE, as above). As Halevi recently observed [25], we still lack a commonly accepted syntax for describing the intended functionality of multi-linear maps as well as a succinct description of the underlying hardness assumptions, along with a candidate that realizes the functionality and the hardness assumptions under simple and plausible intractability assumptions. We view our abstraction of directed encodings as an important step in that direction. We adopt the syntax and the candidate for directed encodings from [20]; the novelty of this work lies in (i) putting forth concrete hardness assumptions about directed encodings, (ii) showing that the functionality and these hardness assumptions for directed encodings already suffices for the application to obfuscating conjunctions as in [5], and most importantly, (iii) demonstrating a reduction of these hardness assumptions to standard ring LWE assumptions and a simple and plausible strengthening there-of.

1.2 Discussion

All of the known approaches for obfuscation beyond point functions rely on multi-linear maps. A crucial theoretical limitation of these approaches is that they all rely on non-standard assumptions; we have few candidates for multi-linear maps [18, 12, 20, 13] and the corresponding assumptions are presently poorly understood and not extensively studied in cryptanalysis, and in many cases, broken [8, 11, 28, 9, 32, 10]. Indeed, these latter attacks highlight the importance of obtaining constructions and developing techniques that work under standard cryptographic assumptions, as is the focus of this work.

Our work may be viewed as taking a step towards basing obfuscation on standard assumptions, starting with conjunctions, which is an important special case of evasive functions, namely functions for which it is hard to find an input that evaluates to 0. As articulated by Badrinarayanan et al. [2], we *can* hope to obfuscate the class of evasive functions in a way that survives "all known attacks" on the multilinear maps, where no encodings of 0 can be created by a generic-model adversary. The reason this is meaningful is that all of the recent attacks on candidate multi-linear maps rely crucially on the ability of the adversary to create encodings of 0. We leave as an important open problem to extend our construction to the class of all evasive functions.

On the Entropic Ring LWE Assumption. A few works have studied entropic variants of the LWE assumption. Goldwasser, Kalai, Peikert and Vaikuntanathan [22] showed that, roughly speaking, LWE with n-dimensional secrets over $\mathbb{Z}_q$ drawn from a distribution with min-entropy k is at least as hard as LWE with uniformly random secrets in $O(k/\log q)$ dimensions. However, such a result is not known for Ring LWE to the best of our knowledge. Another source of difficulty is that the min-entropy of our secrets is $o(\log q)$, much too small for the results in [22] to be applicable, even if they do extend to the Ring LWE setting.

We view the question of understanding the entropic ring LWE assumption, both in the [22] range of parameters (namely, $k = \omega(\log q)$) as well as our more aggressive range of parameters (namely, $k = o(\log q)$) to be very interesting questions for future research, with many potential applications.

On Coron's Attack. Coron [10] recently came up with an attack against the multiparty key exchange protocol based on the GGH15 encoding scheme [20]. This attack relies on extraneous properties of the key-agreement protocol of GGH15 (which had no security reduction) and does not seem to contain any new insights that could be leveraged to attack the Ring-LWE assumption or its entropic variant on which our scheme is based. In a bit more detail, Coron's attack relies on the adversary being able to obtain many "encodings" of 0, a "feature" that is inherent to the key exchange protocol, but not present in our scheme. Indeed, we believe the attack only strengthens the premise of our paper, which focuses on provable security under a (almost) standard assumption, namely an entropic variant of Ring-LWE.

Organization of the Paper. We present an abstract framework for the syntax of directed encodings and its underlying assumptions in Section 3, and its instantiation in Section 4. In particular, the hardness assumptions are presented in Sections 3.1 (abstract) and 4.3 (concrete). We present our conjunction obfuscator in Section 5 relying only on our abstract framework.

2 Preliminaries

2.1 Average Min-Entropy

We use information theoretic tools similar to those in [5].

Definition 2.1 (average min-entropy [15])**.** *Let X and Z be (possibly dependent) random variables, the average min entropy of X conditioned on Z is:*

$$\widetilde{\mathbf{H}}_\infty(X|Z) = -\log\left(\mathop{\mathbb{E}}_{z\leftarrow Z}\left[2^{-\mathbf{H}_\infty(X|Z=z)}\right]\right).$$

Lemma 2.1 ([15])**.** *Let X, Y, Z be (possibly dependent) random variables, where the support of Z is of size $\leq 2^\ell$. Then $\widetilde{\mathbf{H}}_\infty(X|Y,Z) \geq \widetilde{\mathbf{H}}_\infty(X|Y) - \ell$.*

2.2 Distributional VBB Obfuscation

The notion of obfuscation discussed in this paper is distributional (or average case) VBB [16, 27, 26, 4], defined as follows.

Definition 2.2 (Distributional VBB)**.** *Consider a circuit family $\mathcal{C} = \{\mathcal{C}_n\}_{n\in\mathbb{N}}$ with input size n and let Obf be a p.p.t. algorithm, which takes as input a circuit $C \in \mathcal{C}$, a security parameter $\lambda \in \mathbb{N}$, and outputs a boolean circuit $\mathsf{Obf}(1^\lambda, C)$ (which is itself not necessarily in $\mathcal{C}$). Let $\mathcal{D}$ be a class of distribution ensembles $D = \{D_\lambda\}_{\lambda\in\mathbb{N}}$ that sample $(C, \mathsf{aux}) \leftarrow D_\lambda$ with $C \in \mathcal{C}$.*

Obf *is an* obfuscator *for the distribution class $\mathcal{D}$ over the circuit family $\mathcal{C}$, if it satisfies the following properties:*

1. Preserving Functionality: *There is some negligible function* $\nu(\lambda) = \text{negl}(\lambda)$ *such that for all circuits* $C \in \mathcal{C}$ *we have* $\Pr[\forall x \in \{0,1\}^n \; : \; C(x) = \mathsf{Obf}(1^\lambda, C)(x)] \geq 1 - \nu(\lambda)$*, where the probability is over the coin tosses of* Obf*.*

2. Polynomial Slowdown: *For every* $\lambda \in \mathbb{N}$ *and* $C \in \mathcal{C}$*, the circuit* $\mathsf{Obf}(1^\lambda, C)$ *is of size at most* $\text{poly}(|C|, \lambda)$*.*

3. *Distributional* Virtual Black-Box: *For every (non-uniform) polynomial size adversary* Adv*, there exists a (non-uniform) polynomial size simulator* Sim*, such that for every* distribution ensemble $D = \{D_\lambda\} \in \mathcal{D}$*, and every (non-uniform) polynomial size predicate* $P \; : \; \mathcal{C} \to \{0,1\}$*:*

$$\left| \Pr_{(C,\mathsf{aux}) \sim D_\lambda, \mathsf{Obf}, \mathsf{Adv}}[\mathsf{Adv}(\mathsf{Obf}(1^\lambda, C), \mathsf{aux}) = P(C)] - \Pr_{(C,\mathsf{aux}) \sim D_\lambda, \mathsf{Sim}}[\mathsf{Sim}^C(1^\lambda, 1^{|C|}, \mathsf{aux}) = P(C)] \right| = \text{negl}(\lambda)$$

2.3 Conjunctions and Conjunction Obfuscators

The class of conjunctions $\mathcal{C}^{\mathsf{conj}} = \{\mathcal{C}_n^{\mathsf{conj}}\}_{n \in \mathbb{N}}$ is defined as $\mathcal{C}_n^{\mathsf{conj}} = \{F_v : \{0,1\}^n \to \{0,1\}\}_{v \in \{0,1,\star\}^n}$ where, for every $v = (v_1, \ldots, v_n) \in \{0,1,\star\}^n$ and input $x = (x_1, \ldots, x_n) \in \{0,1\}^n$

$$F_v(x) = 1 \text{ if and only if for all } i \in [n],\ v_i = \star \text{ or } x_i = v_i$$

As an abuse of notation, we also use F_v denote the canonical circuit representation of the function F_v, from which it is easy to recover the value v.

We can define the set $w = \{i : \; v_i = \star\}$ corresponding to the "wildcard locations". The classes of conjunctions we are able to obfuscate are those where there is sufficient entropy in v, even when w is fully known. The following definition is adapted from [5] to also handle auxiliary input.

Definition 2.3 (Entropy Given Wildcards)**.** *Let* $D = \{D_\lambda\}$ *be a distribution ensemble that samples* $(F_v, \mathsf{aux}) \leftarrow D_\lambda$ *with* $F_v \in \mathcal{C}_{n(\lambda)}^{\mathsf{conj}}$ *for some polynomial* $n(\cdot)$*. We say that* D *has* $\alpha(\lambda)$ entropy given wildcards *if* $\widetilde{\mathbf{H}}_\infty(v|w, \mathsf{aux}) \geq \alpha(\lambda)$ *where* $(F_v, \mathsf{aux}) \leftarrow D_\lambda$ *and* $w = \{i \; : v_i = \star\}$*.*

We let $\mathcal{D}_\alpha$ *denote the class of all efficiently samplable distribution ensembles* $D = \{D_\lambda\}$ *such that* D *has* $\alpha(\lambda)$*-entropy given wildcards.*

Definition 2.4. *We say that* Obf *is a* $\alpha(\lambda)$*-distributional VBB obfuscator for conjunctions if it is a distributional VBB obfuscator for the class* $\mathcal{D}_\alpha$ *consisting of all distribution ensembles* D *such that* D *has* $\alpha(\lambda)$ *entropy given wildcards.*

We mention an alternate definition of obfuscation security that we call $\alpha(\lambda)$-entropic security.

Definition 2.5. *A conjunction obfuscator* Obf *satisfies* $\alpha(\lambda)$*-entropic security if there exists a polynomial-size simulator* Sim *such that for all efficiently samplable distributions* $D \in \mathcal{D}_\alpha$ *that have* $\alpha(\lambda)$*-entropy given wildcards, we have*

$$(\mathsf{Obf}(1^\lambda, F_v), \mathsf{aux}) \overset{c}{\approx} (\mathsf{Sim}(1^\lambda, 1^{|v|}), \mathsf{aux}).$$

where $(F_v, \mathsf{aux}) \leftarrow D_\lambda$*.*

Note that, in the above definition, the simulator does not depend on the distribution D and therefore, we can think of this definition as saying that the obfuscation hides all properties of the distribution. Also note that the simulator here does not get any oracle access to F_v and therefore does not learn anything at all. As such it's clear that such a definition cannot be achieved for general circuits (where oracle access to the circuit can provide some useful information) but it does make sense for evasive functions.

2.3.1 A Lemma on Entropic Security and Distributional VBB

Lemma 2.2. *If a conjunction obfuscator* Obf *satisfies the functionality preserving and polynomial slowdown properties and has* $\alpha(\lambda)$*-entropic security, then it is an* $(\alpha(\lambda) + 1)$*-distributional VBB obfuscator for conjunctions.*

Proof. Let Obf be an obfuscator satisfying $\alpha(\lambda)$-entropic security with simulator Sim'. Let $D = \{D_\lambda\}$ be any efficiently samplable distribution having $\alpha(\lambda)+1$ entropy given wildcards, meaning that for $(F_v, \mathsf{aux}) \leftarrow D_\lambda$ we have $\widetilde{\mathbf{H}}_\infty(v|\mathsf{aux}, w) \geq \alpha(\lambda)$ where $w = \{i \ : v_i = \star\}$. Furthermore, there is some polynomial $n(\lambda)$ such that $(F_v, \mathsf{aux}) \leftarrow D_\lambda$ has $|v| = n(\lambda)$. For polynomial size adversary Adv, define a (non-uniform) polynomial size simulator $\mathsf{Sim}_{\mathsf{Adv}}(1^\lambda, 1^n) = \mathsf{Adv}(\mathsf{Sim}'(1^\lambda, 1^n))$.

Let $P : \mathcal{C}^{\mathsf{conj}} \to \{0,1\}$ be any polynomial-size predicate. Define the efficiently samplable distribution $D' = \{D'_\lambda\}$ that samples $(v, \mathsf{aux}') \leftarrow D'_\lambda$ by choosing $(F_v, \mathsf{aux}) \leftarrow D_\lambda$ and setting $\mathsf{aux}' = (\mathsf{aux}, P(F_v))$. Then, by applying Lemma 2.1,

$$\begin{aligned}\widetilde{\mathbf{H}}_\infty(v|\mathsf{aux}', w) &= \widetilde{\mathbf{H}}_\infty(v|\mathsf{aux}, w, P(F_v)) \\ &\geq \widetilde{\mathbf{H}}_\infty(v|\mathsf{aux}, w) - 1 \geq \alpha(\lambda).\end{aligned}$$

Therefore, by $\alpha(\lambda)$-entropic security, we have

$$(\mathsf{Obf}(1^\lambda, F_v), \mathsf{aux}, P(F_v)) \overset{c}{\approx} (\mathsf{Sim}'(1^\lambda, 1^n), \mathsf{aux}, P(F_v))$$

when $(v, \mathsf{aux}' = (\mathsf{aux}, P(F_v))) \leftarrow D'_\lambda$. In particular, this means that

$$\left| \Pr_{(F_v,\mathsf{aux})\sim D_\lambda}[\mathsf{Adv}(\mathsf{Obf}(1^\lambda, F_v), \mathsf{aux}) = P(F_v)] - \Pr_{(F_v,\mathsf{aux})\sim D_\lambda}[\mathsf{Adv}(\mathsf{Sim}'(1^\lambda, 1^{|v|}), \mathsf{aux}) = P(F_v)] \right| = \operatorname{negl}(\lambda)$$

by recalling that $\mathsf{Sim}_{\mathsf{Adv}}(1^\lambda, 1^n) = \mathsf{Adv}(\mathsf{Sim}'(1^\lambda, 1^n))$ we get the proof of the lemma. □

2.4 Lattice Preliminaries

For a vector $\mathbf{x}$, we let $||\mathbf{x}||$ denote its ℓ_2 norm and $||\mathbf{x}||_\infty$ denote its infinity norm. For a matrix $\mathbf{R} \in \mathbb{Z}^{m \times m}$ we define $||\mathbf{R}|||$ (resp. $||\mathbf{R}||_\infty$) as the ℓ_2 (resp. infinity) length of the longest column of $\mathbf{R}$. Let $D_{\mathbb{Z}^m,\sigma}$ be the truncated discrete Gaussian distribution over $\mathbb{Z}^m$ with parameter σ, that is, we replace the output by $\mathbf{0}$ whenever the $||\cdot||_\infty$ norm exceeds $\sqrt{m}\cdot\sigma$ (this is statistically close to the discrete Gaussian distribution with parameter σ as long as $m = \omega(\log\lambda)$).

Lemma 2.3 (Lattice Trapdoors [1, 21, 31])**.** *There is an efficient randomized algorithm* $\mathsf{TrapSamp}(1^n, 1^m, q)$ *that, given any integers $n \geq 1$, $q \geq 2$, and sufficiently large $m = \Omega(n \log q)$, outputs a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and a trapdoor matrix $\mathbf{T} \in \mathbb{Z}^{m\times m}$ such that the distribution of $\mathbf{A}$ is* $\operatorname{negl}(n)$*-close to uniform.*

Moreover, there is an efficient algorithm $\mathsf{GaussSamp}$ *that with overwhelming probability over all random choices, does the following: For any $\mathbf{u} \in \mathbb{Z}_q^n$, and large enough $s = \Omega(\sqrt{n\log q})$, the randomized procedure* $\mathsf{GaussSamp}(\mathbf{A}, \mathbf{T}, \mathbf{u}, s)$ *outputs a vector $\mathbf{r} \in \mathbb{Z}^m$ with norm $||\mathbf{r}||_\infty \leq ||\mathbf{r}||_2 \leq s\sqrt{n}$ (with probability 1). Furthermore, the following distributions of the tuple $(\mathbf{A}, \mathbf{T}, \mathbf{U}, \mathbf{R})$ are within* $\operatorname{negl}(n)$ *statistical distance of each other for any polynomial $k \in \mathbb{N}$:*

- $(\mathbf{A}, \mathbf{T}) \leftarrow \mathsf{TrapSamp}(1^n, 1^m, q)$*;* $\mathbf{U} \leftarrow \mathbb{Z}_q^{n\times k}$*;* $\mathbf{R} \leftarrow \mathsf{GaussSamp}(\mathbf{A}, \mathbf{T}, \mathbf{U}, s)$*.*
- $(\mathbf{A}, \mathbf{T}) \leftarrow \mathsf{TrapSamp}(1^n, 1^m, q)$*;* $\mathbf{R} \leftarrow (D_{\mathbb{Z}^m,s})^k$*;* $\mathbf{U} := \mathbf{A}\mathbf{R} \pmod q$*.*

This also extends to the ring setting with $\mathbf{A} \in \mathcal{R}_q^m, \mathbf{T} \in \mathcal{R}^{m\times m}, \mathbf{U} \in \mathcal{R}^k, \mathbf{R} \in \mathcal{R}^{m\times k}$.

Lemma 2.4 ("Noise smudging" [14])**.** *Let $\beta > 0$ and $y \in \mathbb{Z}$ be arbitrary. Then, the statistical distance between the distributions $D_{\mathbb{Z},\beta}$ and $D_{\mathbb{Z},\beta+y}$ is at most $|y|/\beta q$.*

The lemma below (restated from [33, Lemma 10]) states that most "small" polynomials are units in the ring $R_q = \mathbb{Z}_q[x]/(x^n+1)$.

Lemma 2.5. *Let $n \geq 8$ be a power of 2 such that x^n+1 splits into n linear factors modulo prime $2^n \geq q \geq 5$. (In particular, setting $q = 1 \pmod{2n}$ satisfies this condition). Let $\sigma = \Omega(\sqrt{n\log q\log n})$. Then,*

$$\Pr[s \leftarrow D_{\mathbb{Z}^n,\sigma} : \ s \notin R_q^\times] = O(n/q)$$

2.5 The Ring Learning with Errors Problem

We start by defining a simple special case of the ring LWE problem [30]. We define the operator $\mathsf{MakePoly}$ such that for all rings $\mathcal{R}$, if $\mathbf{a} \in \mathcal{R}^n$, then $\mathsf{MakePoly}(\mathbf{a}) \in \mathcal{R}[x]$ is the polynomial whose coefficients are the elements of $\mathbf{a}$. If D is a distribution over $\mathcal{R}^n$ then $\mathsf{MakePoly}(D)$ is the respective distribution over $\mathcal{R}[x]$.

Definition 2.6. *Let n be a power of 2, and let $\mathcal{R} = \mathbb{Z}[x]/\langle x^n + 1\rangle$. Let q be such that $q \equiv 1 \pmod{2n}$ and define $\mathcal{R}_q = \mathcal{R}/q\mathcal{R}$. Let $m \in \mathbb{N}$ and let χ be a distribution over the integers. The $\mathsf{PLWE}_{n,m,q,\chi}$ problem is the problem of distinguishing $\{(a_i, a_i \cdot s + e_i \pmod{x^n+1, q})\}_{i\in[m]}$ from $\{(a_i, u_i)\}_{i\in[m]}$, where $s \xleftarrow{\$} \mathsf{MakePoly}(\chi^n)$, $a_i \xleftarrow{\$} \mathsf{MakePoly}(\mathbb{Z}_q^n)$, $e_i \xleftarrow{\$} \mathsf{MakePoly}(\chi^n)$, $u_i \xleftarrow{\$} \mathsf{MakePoly}(\mathbb{Z}_q^n)$.*

The following is an immediate corollary from [17], together with a standard Hermite Normal Form reduction, see e.g. [30].

Corollary 2.6. *Let n, m, q be as in Definition 2.6 above. Then for all $B \geq \sqrt{n}\cdot(nm/\log(nm))^{1/4}\cdot\omega(\sqrt{\log n})$, there exists a B-bounded distribution χ such that $\mathsf{PLWE}_{n,m,q,\chi}$ is at least as hard as quantumly approximating the shortest vector in worst case ideal lattice over $\mathbb{Z}[\zeta_{2n}]$ to within $\widetilde{O}\left(n \cdot (nm/\log(nm))^{1/4} \cdot (q/B)\right)$ factor.*

We also define an entropic version of the problem as follows.

Definition 2.7 (α-entropic PLWE)**.** *Let n, m, q, χ be parameters of λ and $\mathcal{R}_q$ be as in Definition 2.6, and let $D = \{D_\lambda\}$ be an efficiently samplable distribution with $(x, z) \leftarrow D_\lambda$ having $x \in \{0,1\}^\ell$ for some $\ell = \ell(\lambda)$ and $\widetilde{\mathbf{H}}_\infty(x|z) \geq \alpha(\lambda)$. The α-entropic $\mathsf{PLWE}_{n,m,q,\chi}$ problem is to distinguish*

$$\left(\{s_j\}_{j\in[\ell]}, z, \{(a_i, a_i \cdot s + e_i)\}_{i\in[m]}\right) \qquad \textit{from} \qquad \left(\{s_j\}_{j\in[\ell]}, z, \{(a_i, u_i)\}_{i\in[m]}\right) ,$$

where $s_j \xleftarrow{\$} \mathsf{MakePoly}(\chi^n)$, we let $(x, z) \xleftarrow{\$} D$ and set $s = \prod_{j\in[\ell]} s_j^{x_j}$, and as above $a_i \xleftarrow{\$} \mathsf{MakePoly}(\mathbb{Z}_q^n)$, $e_i \xleftarrow{\$} \mathsf{MakePoly}(\chi^n)$, $u_i \xleftarrow{\$} \mathsf{MakePoly}(\mathbb{Z}_q^n)$. All operations are over the ring $\mathcal{R}_q$.

3 Directed Encoding Schemes

Directed encoding schemes are a special case of graph-induced multi-linear maps of Gentry, Gorbunov and Halevi [20], specialized to a line. Let R_M be a ring. In this section, we present an abstract framework for the syntax of directed encodings and its underlying assumptions, and we describe our instantiation in Section 4.

Definition 3.1 (Directed encoding scheme)**.** *A* directed encoding scheme *associated with a message space $\mathcal{M} \subseteq R_M$ is a tuple of p.p.t. algorithms* ($\mathsf{Setup}, \mathsf{Encode}, \mathsf{REncode}, \mathsf{Mult}, \mathsf{EqualTest}$) *which work as follows.*

- $\mathsf{Setup}(1^\lambda, 1^L)$*, on input a security parameter λ and an upper-bound L on the number of levels, generates a public key* PK *and a private encoding key* EK*.*
- $\mathsf{Encode}_{\mathsf{PK}_0\to\mathsf{PK}_1}(\mathsf{EK}_0, s)$*, on input a "source" key-pair* $(\mathsf{PK}_0, \mathsf{EK}_0)$*, a "target" public key* PK_1*, and a message $m \in \mathcal{M}$, outputs an encoding c.*
- $\mathsf{REncode}_{\mathsf{PK}_0\to\mathsf{PK}_1}(1^\lambda)$*, on input a "source" public key* PK_0 *and a "target" public key* PK_1*, outputs an encoding c.*
- $\mathsf{EqualTest}_{\mathsf{PK}_0\to\mathsf{PK}_1}(c_0, c_1)$*, on input two public keys* PK_0 *and* PK_1*, and two encodings c_0, c_1, outputs a bit b (signifying* accept *or* reject*).*
- $\mathsf{Mult}(c_1, c_2)$*, on input two encodings c_1, c_2, outputs an encoding $c_\times$.*

We also extend Mult to handle multiple encodings in the natural way:

$$\mathsf{Mult}(c_1, c_2, c_3, \ldots) = \mathsf{Mult}(c_1, \mathsf{Mult}(c_2, \mathsf{Mult}(c_3, \ldots))).$$

Informally, correctness stipulates that the encodings uniquely determine the underlying message, and that we can multiply up to L encodings.

Figure 1: The experiments $\mathsf{exp}^{\mathsf{gxdh}}$ and $\mathsf{exp}^{\mathsf{rand}}$ in the SXDH security game. See Definition 1.

```
$\underline{\mathsf{exp}^{\mathsf{gxdh}}(1^\lambda, 1^L)\text{:}}$

1: $(\mathsf{PK}_0, \mathsf{EK}_0) \leftarrow \mathsf{Setup}(1^\lambda, 1^L)$;
2: $(\mathsf{PK}_1, \mathsf{EK}_1) \leftarrow \mathsf{Setup}(1^\lambda, 1^L)$;
3: $r_0, r_1, s \leftarrow \mathcal{D}_M$;
4: $c_b \leftarrow \mathsf{Encode}_{\mathsf{PK}_0 \to \mathsf{PK}_1}(\mathsf{EK}_0, r_b)$;
5: $d_b \leftarrow \mathsf{Encode}_{\mathsf{PK}_0 \to \mathsf{PK}_1}(\mathsf{EK}_0, s \cdot r_b)$;
6: Output $(\mathsf{PK}_0, \mathsf{PK}_1, \mathsf{EK}_0, c_0, c_1, d_0, d_1)$.
```

```
$\underline{\mathsf{exp}^{\mathsf{rand}}(1^\lambda, 1^L)\text{:}}$

1: $(\mathsf{PK}_0, \mathsf{EK}_0) \leftarrow \mathsf{Setup}(1^\lambda, 1^L)$;
2: $(\mathsf{PK}_1, \mathsf{EK}_1) \leftarrow \mathsf{Setup}(1^\lambda, 1^L)$;
3:
4: $c_b, d_b \leftarrow \mathsf{REncode}_{\mathsf{PK}_0 \to \mathsf{PK}_1}(1^\lambda)$;
5:
6: Output $(\mathsf{PK}_0, \mathsf{PK}_1, \mathsf{EK}_0, c_0, c_1, d_0, d_1)$.
```

Remark 3.1. *We could extend this by an "addition" operation or adding a zero test capability, but we don't need it here.*

Correctness. We write $\mathcal{M}^i = \{s_1 s_2 \cdots s_i : s_1, s_2, \ldots, s_i \in \mathcal{M}\}$. We require the following correctness properties from the scheme to hold with probability $1 - \text{negl}(\lambda)$ over $(\mathsf{PK}_0, \mathsf{EK}_0), (\mathsf{PK}_1, \mathsf{EK}_1), (\mathsf{PK}_2, \mathsf{EK}_2)$. For each $s \in \mathcal{M}^L$, there exists a family of sets $\mathcal{E}^{(1)}_{\mathsf{PK}_0,\mathsf{PK}_1,s}, \ldots, \mathcal{E}^{(L)}_{\mathsf{PK}_0,\mathsf{PK}_1,s}$ where $\mathcal{E}^{(1)}_{\mathsf{PK}_0,\mathsf{PK}_1,s} \subseteq \cdots \subseteq \mathcal{E}^{(L)}_{\mathsf{PK}_0,\mathsf{PK}_1,s}$ such that:

- For all $s \in \mathcal{M}$, $\mathsf{Encode}_{\mathsf{PK}_0 \to \mathsf{PK}_1}(\mathsf{EK}_0, s) \in \mathcal{E}^{(1)}_{\mathsf{PK}_0,\mathsf{PK}_1,s}$.
- For all $s_1, s_2 \in \mathcal{M}^L$ and all $(c_1, c_2) \in \mathcal{E}^{(L)}_{\mathsf{PK}_0,\mathsf{PK}_1,s_1} \times \mathcal{E}^{(L)}_{\mathsf{PK}_0,\mathsf{PK}_1,s_2}$, we have $\mathsf{EqualTest}_{\mathsf{PK}_0 \to \mathsf{PK}_1}(c_1, c_2) = 1$ iff $s_1 = s_2$.
- For all i, j for which $i + j \leq L$, for all $(s_1, s_2) \in \mathcal{M}^i \times \mathcal{M}^j$ and all $(c_1, c_2) \in \mathcal{E}^{(i)}_{\mathsf{PK}_0,\mathsf{PK}_1,s_1} \times \mathcal{E}^{(j)}_{\mathsf{PK}_1,\mathsf{PK}_2,s_2}$ we have $\mathsf{Mult}(c_1, c_2) \in \mathcal{E}^{(i+j)}_{\mathsf{PK}_0,\mathsf{PK}_2,s_1 s_2}$.

Note that there is no correctness requirement for "malformed encodings" that do not belong to some $\mathcal{E}^{(L)}_{\mathsf{PK}_0,\mathsf{PK}_1,s}$.

3.1 Security Assumptions

For the security properties, we define an efficiently samplable distribution $\mathcal{D}_M$ over $\mathcal{M}$. We require the following security properties from the encoding scheme.

Property 1 (Graded External DH)**.** *For every polynomial $L = L(\lambda)$ the following experiments* $\mathsf{exp}^{\mathsf{gxdh}}(1^\lambda, 1^{L(\lambda)})$ *and* $\mathsf{exp}^{\mathsf{rand}}(1^\lambda, 1^{L(\lambda)})$ *should produce indistinguishable outputs.*

We can also define two special sub-cases of the GXDH assumption which are already useful. We define the *2-element GXDH*, denoted GXDH-2, the same way as above but modify the experiments $\mathsf{exp}^{\mathsf{gxdh}}$, $\mathsf{exp}^{\mathsf{rand}}$ to only output $(\mathsf{PK}_0, \mathsf{PK}_1, \mathsf{EK}_0, c_0, d_0)$. We also define the *1-element GXDH*, denoted GXDH-1, the same way as above but modify $\mathsf{exp}^{\mathsf{gxdh}}$ and $\mathsf{exp}^{\mathsf{rand}}$ to only output $(\mathsf{PK}_0, \mathsf{PK}_1, \mathsf{EK}_0, c_0)$. The following is clear.

Proposition 3.1. *The GXDH security property implies GXDH-2 and GXDH-1.*

Our second assumption is entropic security which is similar in spirit to, but weaker than, the "GCAN" assumption from [5]. In the GCAN assumption on multilinear maps, the adversary was given access to the complete encoding parameters of the scheme (which roughly correspond to the encoding keys EK_0 and EK_1 below). However, in our setting this assumption is seems too strong and quite possibly incorrect. We thus notice that so long as the adversary is unable to obtain the encoding key EK_1 for the "target", we can establish security based on entropic RLWE (see Section 4.3) and this is in fact sufficient for our construction (see Section 5).

Property 2 (α-Entropic Security). *For a parameter $\alpha = \alpha(\lambda)$, we say that the encoding scheme is α-entropic secure if for all polynomial $\ell = \ell(\lambda), L = L(\lambda)$ and all efficiently samplable distributions $D = \{D_\lambda\}_{\lambda \in \mathbb{N}}$ such that $(x, z) \leftarrow D_\lambda$ contains $x \in \{0,1\}^{\ell(\lambda)}$ and $\widetilde{\mathbf{H}}_\infty(x|z) \geq \alpha(\lambda)$ the following two distributions are computationally indistinguishable:*

$$\left(\mathsf{PK}_0, \mathsf{PK}_1, \mathsf{EK}_0, z, s_1, s_2, \ldots, s_\ell, c \leftarrow \mathsf{Encode}_{\mathsf{PK}_0 \to \mathsf{PK}_1}(\mathsf{EK}_0, s_x)\right)$$

and

$$\left(\mathsf{PK}_0, \mathsf{PK}_1, \mathsf{EK}_0, z, s_1, s_2, \ldots, s_\ell, c \leftarrow \mathsf{REncode}_{\mathsf{PK}_0 \to \mathsf{PK}_1}(1^\lambda)\right),$$

where $(\mathsf{PK}_0, \mathsf{EK}_0) \leftarrow \mathsf{Setup}(1^\lambda, 1^L)$*,* $(\mathsf{PK}_1, \mathsf{EK}_1) \leftarrow \mathsf{Setup}(1^\lambda, 1^L)$*,* $s_1, s_2, \ldots, s_\ell \leftarrow \mathcal{D}_M$*,* $(x, z) \leftarrow D_\lambda$ *and* $s_x = \prod_{i=1}^{\ell} s_i^{x_i}$.

4 An Instantiation of a Directed Encoding Scheme

Let $\mathcal{R}$ be a degree-n number ring $\mathbb{Z}[x]/(f(x))$ for some degree-n polynomial $f(x)$. (We will be mostly agnostic of the specifics of what $f(x)$ is, but encourage the reader to think of a cyclotomic polynomial $f(x) = x^n + 1$ where n is a power of two.) Let q be a rational prime, and $\mathcal{R}_q = \mathcal{R}/q\mathcal{R}$ be the quotient ring. Let $\sigma \in \mathbb{R}^+$ be the Gaussian standard deviation parameter.

4.1 The Encoding Scheme

- $\mathsf{Setup}(1^\lambda, 1^L)$ runs $(\mathbf{A}, \mathbf{T}) \leftarrow \mathsf{TrapSamp}(1^\lambda)$ and outputs

  $$(\mathsf{PK}, \mathsf{EK}) = (\mathbf{A}, \mathbf{T}) \in \mathcal{R}_q^m \times \mathcal{R}^{m \times m}.$$

  Set

  $$n = \Theta(L\lambda \log(L\lambda)), \quad m = \Theta(nL \log q),$$
  $$q = (L\lambda)^{\Theta(L)}, \quad \sigma = \Theta(\sqrt{nL \log q})$$

- $\mathcal{M} = \{s \in \mathcal{R} : \|s\|_\infty \leq m\}$.

- $\mathsf{Encode}_{\mathbf{A}_0 \to \mathbf{A}_1}(s; \mathbf{T}_0)$, where $(\mathbf{A}_0, \mathbf{T}_0), (\mathbf{A}_1, \mathbf{T}_1) \leftarrow \mathsf{Setup}(1^\lambda)$ and $s \in \mathcal{R}$, works as follows.

  – Compute $\mathbf{b}_1 = s\mathbf{A}_1 + \mathbf{e}_1 \in \mathcal{R}_q^m$, where $\mathbf{e}_1 \leftarrow D_{\mathcal{R}^m, \sigma}$.

  – Output a matrix $\mathbf{R}_{0 \to 1} \leftarrow \mathsf{GaussSamp}(\mathbf{A}_0, \mathbf{b}_1; \mathbf{T}_0; \sigma)$.

  We note that $\mathbf{R}_{0 \to 1} \in \mathcal{R}^{m \times m}$ and

  $$\mathbf{A}_0 \mathbf{R}_{0 \to 1} = \mathbf{b}_1 = s\mathbf{A}_1 + \mathbf{e}_1 \text{ (over } \mathcal{R}_q)$$

- $\mathsf{REncode}_{\mathbf{A}_0 \to \mathbf{A}_1}(1^\lambda)$ is the public encoding procedure that simply samples a matrix $\mathbf{R}_{0 \to 1} \leftarrow D_{\mathbb{Z}^n, \sigma}^{m \times m}$.

- $\mathsf{Mult}(\mathbf{R}, \mathbf{R}') = \mathbf{R}\mathbf{R}'$, where multiplication is done over $\mathcal{R}_q$.

- $\mathsf{EqualTest}_{\mathbf{A}_0 \to \mathbf{A}_1}(\mathbf{R}_0, \mathbf{R}_1)$ outputs "yes" if

  $$\|\mathbf{A}_0(\mathbf{R}_0 - \mathbf{R}_1)\|_\infty \leq q/8$$

  and "no" otherwise. (Note that this procedure does not depend on $\mathbf{A}_1$ at all.)

As remarked above, we never use addition or extraction, in contrast to the encoding schemes of [18, 12, 13, 20].

4.2 Correctness of the Encoding Scheme

Fix $\mathbf{A}_0, \mathbf{A}_1 \in \mathcal{R}_q^m$ throughout. We define

$$\mathcal{E}^{(i)}_{\mathbf{A}_0,\mathbf{A}_1,s} = \{\mathbf{R} : \|\mathbf{R}\|_\infty \leq 2^{i-1}m^i\}$$

and

$$\|\mathbf{A}_0\mathbf{R} - s\mathbf{A}_1\|_\infty \leq 2^{i-1}m^i\}$$

First, it is easy to check that the output of $\mathsf{Encode}_{\mathbf{A}_0 \to \mathbf{A}_1}(s; \mathbf{T}_0)$ lies in $\mathcal{E}^{(1)}_{\mathbf{A}_0,\mathbf{A}_1,s}$. We then establish correctness in the following lemmas:

Lemma 4.1 (Correctness of $\mathsf{EqualTest}$)**.** *Suppose $q \geq 16Lm^L$. Then, for any $\mathbf{R}_0 \in \mathcal{E}^{(L)}_{\mathbf{A}_0,\mathbf{A}_1,s_0}, \mathbf{R}_1 \in \mathcal{E}^{(L)}_{\mathbf{A}_0,\mathbf{A}_1,s_1}$, we have*

$$\|\mathbf{A}_0(\mathbf{R}_0 - \mathbf{R}_1)\|_\infty \leq q/8 \textit{ iff } s_0 = s_1.$$

Proof. The direction $\Leftarrow$ is straight-forward. Now, suppose $s_0 \neq s_1$. Then, we have

$$\|\mathbf{A}_0(\mathbf{R}_0 - \mathbf{R}_1)\|_\infty \geq \|(s_0 - s_1)\mathbf{A}_1\|_\infty - 2m^{2L} > q/8$$

The last line holds since, with overwhelming probability over the choice of $\mathbf{A}_1$, it holds that for all $s \neq 0$, $\|s\mathbf{A}_1\|_\infty > q/4$. □

Lemma 4.2 (Correctness of Mult)**.** *For all i, j for which $i + j \leq L$, for all $(s_1, s_2) \in \mathcal{M}^i \times \mathcal{M}^j$ and all $(c_1, c_2) \in \mathcal{E}^{(i)}_{\mathsf{PK}_0,\mathsf{PK}_1,s_1} \times \mathcal{E}^{(j)}_{\mathsf{PK}_1,\mathsf{PK}_2,s_2}$ we have $\mathsf{Mult}(c_1, c_2) \in \mathcal{E}^{(i+j)}_{\mathsf{PK}_0,\mathsf{PK}_2,s_1s_2}$*

Proof. First, observe that

$$\|\mathbf{R}_0\mathbf{R}_1\|_\infty \leq \|\mathbf{R}_0\|_\infty \|\mathbf{R}_1\|_\infty \leq 2^{i+j-1}m^{i+j}.$$

Now, write $\mathbf{A}_0\mathbf{R}_0 = s_0\mathbf{A}_1 + \mathbf{e}_0$ and $\mathbf{A}_1\mathbf{R}_1 = s_1\mathbf{A}_1 + \mathbf{e}_1$. Let us unfold the expression $\mathbf{A}_0 \cdot \mathbf{R}_0\mathbf{R}_1$:

$$\begin{aligned}
\mathbf{A}_0 \cdot \mathbf{R}_0\mathbf{R}_1 &= (s_0\mathbf{A}_1 + \mathbf{e}_0)\mathbf{R}_1 && \text{rewriting } \mathbf{A}_0\mathbf{R}_0 \\
&= s_0(s_1\mathbf{A}_2 + \mathbf{e}_1) + \mathbf{e}_0\mathbf{R}_1 && \text{rewriting } \mathbf{A}_1\mathbf{R}_1 \\
&= s_0s_1\mathbf{A}_2 + \mathbf{e}_0\mathbf{R}_1 + s_0\mathbf{e}_1
\end{aligned}$$

Hence,

$$\|\mathbf{A}_0 \cdot \mathbf{R}_0\mathbf{R}_1 - s_0s_1\mathbf{A}_2\|_\infty = \|\mathbf{e}_0\mathbf{R}_1 + s_0\mathbf{e}_1\|_\infty \leq 2^{i+j-1}m^{i+j}$$

where we used the bounds

$$\begin{gathered}
\|\mathbf{e}_0\|_\infty \leq 2^{i-1}m^i, \quad \|\mathbf{R}_1\|_\infty \leq 2^{j-1}m^j, \\
\|s_0\|_\infty \leq m^i, \quad \|\mathbf{e}_1\|_\infty \leq 2^{j-1}m^j
\end{gathered}$$

□

4.3 Security of the Encoding Scheme

We prove the security properties of our encoding scheme (see Section 3.1) under the PLWE and the entropic PLWE assumption (see Section 2.5). These security properties are essentially analogous to the GXDH and GCAN assumptions from the work of Brakerski and Rothblum [5]. However, the key novelty in this work is that we are able to establish these security properties based on the hardness of problems relating to learning with errors over rings [30].

Lemma 4.3 (Graded External Diffie-Hellman)**.**
Let n be a power of 2, and let $\mathcal{R} = \mathbb{Z}[x]/\langle x^n + 1\rangle$. Let $q = 2^{\omega(\log \lambda)}$ be such that $q \equiv 1 \pmod{2n}$ and define $\mathcal{R}_q = \mathcal{R}/q\mathcal{R}$. Let $m \in \mathbb{N}$ and let χ be a distribution over the integers. Then, our encoding scheme satisfies the Graded external Diffie-Hellman (GXDH) property (Property 1) assuming the hardness of the $\mathsf{PLWE}_{n,m,q,\chi}$ problem (Definition 2.6).

Proof. We first show that the distributions

$$\Big(\mathbf{A}_1, r_0\mathbf{A}_1 + \mathbf{e}_0, r_1\mathbf{A}_1 + \mathbf{e}_1, sr_0\mathbf{A}_1 + \mathbf{e}_0', sr_1\mathbf{A}_1 + \mathbf{e}_1'\Big) \tag{1}$$

$$\text{and } \Big(\mathbf{A}_1, \mathbf{U}_0, \mathbf{U}_1, \mathbf{U}_0', \mathbf{U}_1'\Big) \tag{2}$$

are computationally indistinguishable under PLWE, where $(\mathbf{A}_b, \mathbf{T}_b) \leftarrow \mathsf{TrapSamp}(1^\lambda)$, $\mathbf{e}_b, \mathbf{e}_b' \leftarrow D_{\mathcal{R}^m,\sigma}$, $s, r_0, r_1 \leftarrow \mathcal{R}_M$ (the message space), and $\mathbf{U}_b, \mathbf{U}_b' \leftarrow \mathcal{R}^m$ (uniformly chosen from the ring).

We show the indistinguishability of (1) and (2) through a sequence of hybrid distributions.

Hybrid 1 is distribution (1).

Hybrid 2. This is the distribution

$$\Big(\mathbf{A}_1, r_0\mathbf{A}_1 + \mathbf{e}_0, r_1\mathbf{A}_1 + \mathbf{e}_1, s\cdot(r_0\mathbf{A}_1 + \mathbf{e}_0) + \mathbf{e}_0',$$
$$s\cdot(r_1\mathbf{A}_1 + \mathbf{e}_1) + \mathbf{e}_1'\Big)$$

where $\mathbf{e}_1 \leftarrow D_{\mathcal{R}^m,\sigma}$ and $\mathbf{e}_1' \leftarrow D_{\mathcal{R}^m,\sigma'}$.

Hybrids 1 and 2 are statistically indistinguishable assuming that $\sigma' = \omega(\log\lambda)\cdot||s||_\infty\cdot\sigma$ (by Lemma 2.4). Roughly speaking, the only difference between hybrids 1 and 2 is in the noise distribution of the last two PLWE samples, where in the former, the noise is drawn from $D_{\mathcal{R}^m,\sigma'}$, and in the latter, they are computed as $s\mathbf{e}_b + \mathbf{e}_b'$. If the magnitude of the noise terms $\mathbf{e}_b'$ is much larger than the norm of $s\mathbf{e}_b$, by "noise smudging" (Lemma 2.4), these two distributions have statistical distance negl(λ).

Hybrid 3. This is the distribution

$$\Big(\mathbf{A}_1, \mathbf{U}_0, \mathbf{U}_1, s\mathbf{U}_0 + \mathbf{e}_0', s\mathbf{U}_1 + \mathbf{e}_1'\Big)$$

where $\mathbf{e}_1' \leftarrow D_{\mathcal{R}^m,\sigma'}$. Hybrids 2 and 3 are computationally indistinguishable under the $\mathsf{PLWE}_{n,m,q,\sigma}$ assumption.

Hybrid 4 is distribution (2). That is,

$$\Big(\mathbf{A}_1, \mathbf{U}_0, \mathbf{U}_1, \mathbf{U}_0', \mathbf{U}_1'\Big)$$

Hybrids 3 and 4 are computationally indistinguishable under the $\mathsf{PLWE}_{n,m,q,\sigma'}$ assumption.

To finish the proof, note that the indistinguishability of distributions 1 and 2 immediately imply the indistinguishability of the following two distributions

$$\Big(\mathbf{A}_0, \mathbf{T}_0, \mathbf{A}_1, \mathsf{GaussSamp}(\mathbf{A}_0, r_b\mathbf{A}_1 + \mathbf{e}_b;\ \mathbf{T}_0), \mathsf{GaussSamp}(\mathbf{A}_0, sr_b\mathbf{A}_1 + \mathbf{e}_b';\ \mathbf{T}_0)\Big)$$
$$\approx_c \Big(\mathbf{A}_0, \mathbf{T}_0, \mathbf{A}_1, \mathsf{GaussSamp}(\mathbf{A}_0, \mathbf{U}_b;\ \mathbf{T}_0), \mathsf{GaussSamp}(\mathbf{A}_0, \mathbf{U}_b';\ \mathbf{T}_0)\Big)$$
$$\approx_s \Big(\mathbf{A}_0, \mathbf{T}_0, \mathbf{A}_1, \mathbf{R}_b, \mathbf{R}_b'\Big)$$

where $\mathbf{R}_b, \mathbf{R}_b' \leftarrow D_{\mathcal{R}^m,\sigma}$. Since this is exactly the distribution generated by $\mathsf{REncode}$, this establishes GXDH. □

Entropic security of our encoding scheme follows directly from the entropic PLWE assumption. We state the lemma below.

Lemma 4.4 (Entropic Security). *Let n be a power of 2, and let $\mathcal{R} = \mathbb{Z}[x]/\langle x^n + 1\rangle$. Let $q = 2^{\omega(\log\lambda)}$ be such that $q \equiv 1 \pmod{2n}$ and define $\mathcal{R}_q = \mathcal{R}/q\mathcal{R}$. Let $m \in \mathbb{N}$ and let χ be a distribution over the integers. Then, for every α, our encoding scheme satisfies α-entropic security (Property 2) under the α-entropic $\mathsf{PLWE}_{n,m,q,\chi}$ assumption (Definition 2.7).*

5 The Conjunction Obfuscator

Let $(\mathsf{Setup}, \mathsf{Encode}, \mathsf{Mult}, \mathsf{EqualTest})$ be a directed encoding scheme with associated with a message space $\mathcal{M} \subseteq R_M$ for some ring R_M, and a distribution $\mathcal{D}_M$ over $\mathcal{M}$. Given a conjunction $F_v \in \mathcal{C}_\ell^{\mathsf{conj}}$ represented via a vector $v \in \{0,1,\star\}^\ell$, the obfuscator $\Pi_v \leftarrow \mathsf{Obf}(1^\lambda, F_v)$ proceeds as follows.

- Choose $(\mathsf{PK}_i, \mathsf{EK}_i) \leftarrow \mathsf{Setup}(1^\lambda, 1^{\ell+1})$
  for $i \in \{0,\ldots,\ell+1\}$.
- Choose uniformly random $s_{i,b}, r_{i,b} \leftarrow \mathcal{D}_M$ for every $i \in [\ell]$ and $b \in \{0,1\}$ subject to the condition that for every i such that $v_i = \star$, we set $s_{i,0} = s_{i,1}$. For such positions i, we define $s_{i,\star} = s_{i,0} = s_{i,1}$.
- Compute $S_{i,b} = \mathsf{Encode}_{\mathsf{PK}_{i-1}\to\mathsf{PK}_i}(\mathsf{EK}_{i-1}, s_{i,b}r_{i,b})$ and $R_{i,b} = \mathsf{Encode}_{\mathsf{PK}_{i-1}\to\mathsf{PK}_i}(\mathsf{EK}_{i-1}, r_{i,b})$ for every $i \in [\ell]$ and $b \in \{0,1\}$.
- Compute $S_{\ell+1} = \mathsf{Encode}_{\mathsf{PK}_\ell\to\mathsf{PK}_{\ell+1}}(\mathsf{EK}_\ell, r_{\ell+1}\cdot\prod_{i=1}^\ell s_{i,v_i})$ and $R_{\ell+1} = \mathsf{Encode}_{\mathsf{PK}_\ell\to\mathsf{PK}_{\ell+1}}(\mathsf{EK}_\ell, r_{\ell+1})$.

The description[1] of the obfuscated program Π_v consists of

$$\Pi_v = \Big(\{\mathsf{PK}_i\}_{i\in\{0,\ldots,\ell+1\}}, \{S_{i,b}, R_{i,b}\}_{i\in[\ell],b\in\{0,1\}}, S_{\ell+1}, R_{\ell+1}\Big)$$

The obfuscated program Π_v, on input $x \in \{0,1\}^\ell$, proceeds as follows.

- Compute $S^* \leftarrow \mathsf{Mult}(S_{1,x_1}, S_{2,x_2}, \ldots, S_{\ell,x_\ell}, R_{\ell+1})$ and $R^* \leftarrow \mathsf{Mult}(R_{1,x_1}, R_{2,x_2}, \ldots, R_{\ell,x_\ell}, S_{\ell+1})$
- Output whatever $\mathsf{EqualTest}_{\mathsf{PK}_0\to\mathsf{PK}_{\ell+1}}(S^*, R^*)$ outputs.

Theorem 5.1 (Distributional virtual black-box). *Based on the GXDH and the $\alpha(\lambda)$-entropic security properties on the directed encoding scheme, our obfuscator is an $(\alpha(\lambda)+1)$-distributional VBB obfuscator for conjunctions (Definition 2.4).*

We first prove that the obfuscator is functionality preserving. The polynomial slowdown property follows directly by inspection.

Lemma 5.2 (Functionality). *There is a negligible function $\nu(\lambda)$ such that, for every $\ell \in \mathbb{N}$ and every $v \in \{0,1,\star\}^\ell$:*

$$\Pr\left[\forall x \in \{0,1\}^\ell \;:\; \Pi_v(x) = F_v(x)\right] \geq 1 - \nu(\lambda).$$

where the probability is over $\Pi_v \leftarrow \mathsf{Obf}(1^\lambda, F_v)$.

Informally, functionality follows from the fact that with high probability,

$$F_v(x) = 1 \;\Leftrightarrow\; s_{i,v_i} = s_{i,x_i}\ \forall i \in [\ell] \;\Leftrightarrow\; \prod_{i\in[\ell]} s_{i,v_i} = \prod_{i\in[\ell]} s_{i,x_i}$$

[1]It suffices to give out $\mathsf{PK}_0, \mathsf{PK}_{\ell+1}$ instead of $(\mathsf{PK}_0,\ldots,\mathsf{PK}_{\ell+1})$.

Proof. Fix any $\ell \in \mathbb{N}$ and $v \in \{0,1,\star\}^\ell$ and $x \in \{0,1\}^\ell$. Let $\Pi_v \leftarrow \mathsf{Obf}(1^\lambda, F_v)$. During the evaluation of the obfuscated program Π_v on input x the values S^*, R^* that are computed satisfy $S^* \in \mathcal{E}^{(\ell+1)}_{\mathsf{PK}_0,\mathsf{PK}_{\ell+1},s^*}$ and $R^* \in \mathcal{E}^{(\ell+1)}_{\mathsf{PK}_0,\mathsf{PK}_{\ell+1},r^*}$ where

$$s^* = r_{\ell+1} \prod_{i=1}^{\ell} s_{i,x_i} r_{i,x_i} \quad \text{and} \quad r^* = r_{\ell+1} \prod_{i=1}^{\ell} s_{i,v_i} r_{i,x_i}$$

The program outputs 1 iff $\mathsf{EqualTest}_{\mathsf{PK}_0 \to \mathsf{PK}_{\ell+1}}(S^*, R^*) = 1$ which, by the correctness of the encoding scheme, happens iff $s^* = r^*$. This happens with probability 1 if $F_v(x) = 1$ and therefore correctness always holds in this case. On the other hand, if $F_v(x) = 0$ then let j be some index such that $v_j \neq \star$ and $x_j \neq v_j$. We have

$$\begin{aligned} \Pr[F_v(x) \neq \Pi_v(x)] &= \Pr_{\{s_{i,b}, r_{i,b}\}, r_{\ell+1}}[r_{\ell+1} \prod_{i=1}^{\ell} s_{i,x_i} r_{i,x_i} = r_{\ell+1} \prod_{i=1}^{\ell} s_{i,v_i} r_{i,x_i}] \\ &\leq \Pr[\mathsf{NotInv}] + \Pr_{s_{j,x_j}}[s_{j,x_j} = z] \end{aligned}$$

where NotInv is the event that one of $r_{i,b}, s_{i,b}, r_{\ell+1}$ (other than s_{j,x_j}) is non-invertible and $z = \prod_{i \neq j} s_{i,v_i}/s_{i,x_i}$. By lemma 2.5, we can bound the first probability by $\mathrm{poly}(\lambda)/q$ and the second probability is $\leq 2^{-\mathbf{H}_\infty(\mathcal{D}_M)} \leq 1/q$. By our choice of q, this is $\leq 2^{-\ell}\nu(\lambda)$ for some negligible $\nu(\lambda)$. Taking a union bound over all $x \in \{0,1\}^\ell$, we get

$$\Pr[\forall x \in \{0,1\}^\ell \ : \ \Pi_v(x) = F_v(x)] \geq 1 - \nu(\lambda)$$

which proves the lemma. □

Next we prove that the obfuscator is secure. It suffices to prove that the obfuscator satisfies $\alpha(\lambda)$-entropic security (Definition 2.5), as we can then rely on Lemma 2.2 to argue that this implies $(\alpha(\lambda)+1)$-distributional VBB security.

Lemma 5.3 (Security). *Based on the GXDH and $\alpha(\lambda)$-entropic security properties of the directed encoding scheme, our obfuscator satisfies $\alpha(\lambda)$-entropic security (Definition 2.5).*

Proof. The simulator $\mathsf{Sim}(1^\lambda, 1^\ell)$ chooses $(\mathsf{PK}_i, \mathsf{EK}_i) \leftarrow \mathsf{Setup}(1^\lambda, 1^{\ell+1})$ for $i \in \{0, \ldots, \ell+1\}$. It chooses $\widetilde{S}_{i,b} \leftarrow \mathsf{REncode}_{\mathsf{PK}_{i-1} \to \mathsf{PK}_i}()$, $\widetilde{R}_{i,b} \leftarrow \mathsf{REncode}_{\mathsf{PK}_{i-1} \to \mathsf{PK}_i}()$ for $i \in [\ell+1]$. Finally it outputs the simulated program $\widetilde{\Pi} = \left(\{PK_i\}_{i \in \{0,\ldots,\ell+1\}}, \{\widetilde{S}_{i,b}, \widetilde{R}_{i,b}\}_{i \in [\ell], b \in \{0,1\}}, \widetilde{S}_{\ell+1}, \widetilde{R}_{\ell+1}\right)$.

We want to show that, for any efficiently samplable distribution $D = \{D_\lambda\} \in \mathcal{D}_\alpha$ having $\alpha(\lambda)$-entropy given wildcards, the real distribution (Π_v, aux) is indistinguishable from the simulated $(\widetilde{\Pi}, \mathsf{aux})$ where $(F_v, \mathsf{aux}) \leftarrow D_\lambda$, $\Pi_v \leftarrow \mathsf{Obf}(1^\lambda, F_v)$ and $\widetilde{\Pi} \leftarrow \mathsf{Sim}(1^\lambda, 1^\ell)$. We do so via a series of hybrids.

Hybrid 0. This is the real distribution (Π_v, aux) consisting of:

$$\left(\{\mathsf{PK}_i\}_{i \in \{0,\ldots,\ell+1\}}, \{S_{i,b}, R_{i,b}\}_{i \in [\ell], b \in \{0,1\}}, S_{\ell+1}, R_{\ell+1}, \mathsf{aux}\right).$$

Hybrid 1. This is the distribution:

$$\left(\{\mathsf{PK}_i\}_{i \in \{0,\ldots,\ell+1\}}, \{S_{i,b}, R_{i,b}\}_{i \in [\ell], b \in \{0,1\}}, \boxed{\widetilde{S}_{\ell+1}}, R_{\ell+1}, \mathsf{aux}\right).$$

where everything is the same as in Hybrid 0 except that we now choose $\widetilde{S}_{\ell+1} \leftarrow \mathsf{REncode}_{\mathsf{PK}_\ell \to \mathsf{PK}_{\ell+1}}()$.

Lemma 5.4. *Assuming α-entropic security of the encoding scheme, Hybrid 1 is indistinguishable from Hybrid 0.*

Proof. Assume that a PPT adversary Adv can distinguish between Hybrid 0 and 1 with probability $\varepsilon(\lambda)$. By the $\alpha(\lambda)$-entropic security of D we know that $(F_v, \mathsf{aux}) \leftarrow D_\lambda$ satisfies $\widetilde{\mathbf{H}}_\infty(v|\mathsf{aux}, w) \geq \alpha(\lambda)$ where $w = \{i \ : \ v_i = \star\}$. Then we construct an efficiently samplable distribution $D' = \{D'_\lambda\}$ such that $(x, z) \leftarrow D'_\lambda$ satisfies $\widetilde{\mathbf{H}}_\infty(x|z) \geq \alpha(\lambda)$ and a PPT adversary Adv' that breaks entropic security of the encoding with the distribution D'.

Define $(x, z) \leftarrow D'_\lambda$ where $x \in \{0,1\}^{2\ell+1}$ is chosen as follows. First, select $(v, \mathsf{aux}) \leftarrow D_\lambda$ and, for $i \in [\ell]$, set:

- $x_{2i-1} := 1, x_{2i} := 0$ if $v_i = 0$ or $v_i = \star$
- $x_{2i-1} := 0, x_{2i} := 1$ otherwise

Set $x_{2\ell+1} := 1$. The side information is going to be the $z = (\mathsf{aux}, w = \{i \ : \ v_i = \star\})$. Since v can be recovered from x, z we have $\widetilde{\mathbf{H}}_\infty(x|z) \geq \widetilde{\mathbf{H}}_\infty(v|w, \mathsf{aux}) \geq \alpha(\lambda)$.

The adversary Adv' gets $2\ell + 1$ random elements $\widetilde{s}_1, \ldots, \widetilde{s}_{2\ell+1}$, keys $\mathsf{PK}, \mathsf{PK}', \mathsf{EK}$, side information $z = (\mathsf{aux}, w)$ and an encoding c. It defines $r_{\ell+1} = \widetilde{s}_{2\ell+1}$ and:

- $s_{i,0} := \widetilde{s}_{2i-1}$ and $s_{i,1} := \widetilde{s}_{2i}$ for $i \in [\ell] \setminus w$,
- $s_{i,0} := \widetilde{s}_{2i-1}, s_{i,1} := \widetilde{s}_{2i-1}$ for $i \in w$.

Note that this ensures that $r_{\ell+1} \prod_{i\in[\ell]} s_{i,v_i} = \prod_{i\in[2\ell+1]} \widetilde{s}_i^{x_i}$. The adversary Adv' chooses $(\mathsf{PK}_i, \mathsf{EK}_i) \leftarrow \mathsf{Setup}(1^\lambda, 1^{\ell+1})$ for $i \in [\ell - 1]$ and sets $\mathsf{PK}_\ell = \mathsf{PK}$, $\mathsf{EK}_\ell = \mathsf{EK}$ and $\mathsf{PK}_{\ell+1} = \mathsf{PK}'$. It chooses $r_{i,b} \leftarrow \mathcal{D}_M$ $i \in [\ell]$ $b \in \{0,1\}$ at random. It uses the above values to define $S_{i,b}$, $R_{i,b}$, and $R_{\ell+1}$ the same way as the obfuscation scheme. Finally, it sets $S_{\ell+1} := c$. It then runs:

$$\mathsf{Adv}\left(\{\mathsf{PK}_i\}_{i\in\{0,\ldots,\ell+1\}}, \{S_{i,b}, R_{i,b}\}_{i\in[\ell],b\in\{0,1\}}, S_{\ell+1}, R_{\ell+1}, \mathsf{aux}\right)$$

and outputs what it outputs.

It's easy to see that if $c \leftarrow \mathsf{Encode}_{\mathsf{PK}\to\mathsf{PK}'}(\mathsf{EK}, \prod_{i=1}^{2\ell+1} \widetilde{s}_i^{x_i})$ then this matches the distribution in Hybrid 0 while if $c \leftarrow \mathsf{REncode}_{\mathsf{PK}\to\mathsf{PK}'}()$ then this matches the distribution of Hybrid 1. Therefore the advantage of Adv' in the in the entropic security game is the same as that of Adv in distinguishing Hybrids 0 and 1. □

Hybrid 2. This is the distribution

$$\left(\{\mathsf{PK}_i\}_{i\in\{0,\ldots,\ell+1\}}, \{S_{i,b}, R_{i,b}\}_{i\in[\ell],b\in\{0,1\}}, \widetilde{S}_{\ell+1}, \boxed{\widetilde{R}_{\ell+1}}, \mathsf{aux}\right).$$

where everything is the same as in Hybrid 1 except that we now choose $\widetilde{R}_{\ell+1} \leftarrow \mathsf{REncode}_{\mathsf{PK}_\ell\to\mathsf{PK}_{\ell+1}}()$.

Lemma 5.5. *Hybrid 2 is indistinguishable from Hybrid 1 by GXDH-1 security of the encoding scheme.*

Proof. Assume a PPT adversary Adv can distinguish Hybrids 1 and 2 with advantage ε. We construct an adversary Adv' that has advantage ε in the GXDH-1 security game.

The adversary $\mathsf{Adv}'(\mathsf{PK}, \mathsf{PK}', \mathsf{EK}, c)$ needs to distinguish between $c \leftarrow \mathsf{Encode}_{\mathsf{PK}\to\mathsf{PK}'}(\mathsf{EK}, r)$ where $r \leftarrow \mathcal{D}_M$ and $c \leftarrow \mathsf{REncode}_{\mathsf{PK}\to\mathsf{PK}'}()$. It samples the distribution of Hybrid 1 by setting $\mathsf{PK}_\ell := \mathsf{PK}$, $\mathsf{EK}_\ell := \mathsf{EK}$, $\mathsf{PK}_{\ell+1} := \mathsf{PK}'$ and $R_{\ell+1} := c$ and otherwise selects the rest of the components as in Hybrid 1. Finally it runs Adv on the sampled values and outputs what it outputs.

It's easy to see that if $c \leftarrow \mathsf{Encode}_{\mathsf{PK}\to\mathsf{PK}'}(\mathsf{EK}, r)$ where $r \leftarrow \mathcal{D}_M$ then the above matches Hybrid 1, and if $c \leftarrow \mathsf{REncode}_{\mathsf{PK}\to\mathsf{PK}'}()$ then the above matches Hybrid 2. Therefore the advantage of Adv' in the in the GXDH-1 security game is the same as that of Adv in distinguishing Hybrids 1 and 2. □

Hybrid 3.j. For $j \in \{0, \dots, \ell\}$ we define Hybrid 3.j as the distribution

$$\left(\{\mathsf{PK}_i\}_{i \in \{0,\dots,\ell+1\}}, \{S_{i,b}, R_{i,b}\}_{i \in [j], b \in \{0,1\}}, \boxed{\{\widetilde{S}_{i,b}, \widetilde{R}_{i,b}\}_{i \in \{j+1,\dots,\ell\}, b \in \{0,1\}}}, \widetilde{S}_{\ell+1}, \widetilde{R}_{\ell+1} \right)$$

where everything is the same as in Hybrid 2 except that we now choose $\widetilde{R}_{i,b}, \widetilde{S}_{i,b} \leftarrow \mathsf{REncode}_{\mathsf{PK}_{i-1} \to \mathsf{PK}_i}()$ for $i \in \{j+1, \dots, \ell\}, b \in \{0,1\}$.

Note that Hybrid 3.ℓ is the same as Hybrid 2 and Hybrid 3.0 is the same as $(\widetilde{\Pi}, \mathsf{aux})$ where $\widetilde{\Pi}$ is the simulated program . Therefore it suffices to prove the following.

Lemma 5.6. *For all $j \in [\ell]$ Hybrid 3.j is indistinguishable from Hybrid 3.$(j-1)$ by the GXDH security of the encoding scheme.*

Proof. We first define an intermediate distribution Hybrid' 3.j which is the same as Hybrid 3.j except that when $v_j \neq \star$ then the values $\{S_{j,b}, R_{j,b}\}_{b \in \{0,1\}}$ are replaced by random $\{\widetilde{S}_{j,b}, \widetilde{R}_{j,b}\}_{b \in \{0,1\}}$.

We claim that Hybrid 3.j and Hybrid' 3.j are indistinguishable. In particular, if there is a PPT adversary Adv that distinguishes Hybrid 3.j and Hybrid' 3.j with advantage ε then we construct a PPT adversary Adv' with advantage ε in the GXDH problem. The adversary $\mathsf{Adv}'(\mathsf{PK}, \mathsf{PK}', \mathsf{EK}, c_0, c_1, d_0, d_1)$ samples the distribution of Hybrid 3.j except that it sets $\mathsf{PK}_{j-1} := \mathsf{PK}, \mathsf{EK}_{j-1} := \mathsf{EK}, \mathsf{PK}_j := \mathsf{PK}'$ and, when $v_j \neq \star$, it plugs in $R_{j,0} := c_0, R_{j,1} := c_1, S_{j,0} := d_0, S_{j,1} := d_1$. It then runs Adv on the sampled distribution. It is easy to see that when Adv' gets as input a GXDH tuple then the distribution it samples matches Hybrid 3.j and else it matches Hybrid' 3.j which proves the claim.

Next we define and intermediate distribution Hybrid" 3.j which is the same as Hybrid' 3.j except that when $v_j = \star$ then the values $\{S_{j,0}, R_{j,0}\}$ are also replaced by random $\{\widetilde{S}_{j,0}, \widetilde{R}_{j,0}\}$.

We claim that Hybrid' 3.j and Hybrid" 3.j are indistinguishable. In particular, if there is a PPT distinguisher Adv that distinguishes Hybrid' 3.j and Hybrid" 3.j with advantage ε then we construct a PPT distinguisher Adv' with advantage ε in the GXDH-2 problem. The adversary $\mathsf{Adv}'(\mathsf{PK}, \mathsf{PK}', \mathsf{EK}, c, d)$ samples the distribution of Hybrid' 3.j except that it sets $\mathsf{PK}_{j-1} := \mathsf{PK}, \mathsf{EK}_{j-1} := \mathsf{EK}, \mathsf{PK}_j := \mathsf{PK}'$ and, when $v_j = \star$, it plugs in $R_{j,0} := c, S_{j,0} := d$. It then runs Adv on the sampled distribution. It is easy to see that when Adv' gets as input a GXDH tuple then the distribution it samples matches Hybrid' 3.j and else it matches Hybrid" 3.j which proves the claim.

Lastly, we claim that Hybrid" 3.j and Hybrid 3.$(j-1)$ are indistinguishable. The proof of this is identical to that showing the indistinguishability of Hybrid' 3.j and Hybrid" 3.j.

Combining the above, we get the proof of the lemma. ☐

Combining the above we see that Hybrid 0 (obfuscated program) is indistinguishable from Hybrid 3.0 (simulated program) which proves the lemma.

☐

References

[1] Miklós Ajtai. Generating hard instances of the short basis problem. In *ICALP*, pages 1–9, 1999.

[2] Saikrishna Badrinarayanan, Eric Miles, Amit Sahai, and Mark Zhandry. Post-zeroizing obfuscation: The case of evasive circuits. Cryptology ePrint Archive, Report 2015/167, 2015. `http://eprint.iacr.org/`.

[3] Boaz Barak, Nir Bitansky, Ran Canetti, Yael Tauman Kalai, Omer Paneth, and Amit Sahai. Obfuscation for evasive functions. In *TCC*, pages 26–51, 2014.

[4] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. *J. ACM*, 59(2):6, 2012.

[5] Zvika Brakerski and Guy N. Rothblum. Obfuscating conjunctions. In *CRYPTO (2)*, pages 416–434, 2013.

[6] Ran Canetti. Towards realizing random oracles: Hash functions that hide all partial information. In *CRYPTO*, pages 455–469, 1997.

[7] Ran Canetti, Guy N. Rothblum, and Mayank Varia. Obfuscation of hyperplane membership. In *TCC*, pages 72–89, 2010.

[8] Jung Hee Cheon, Kyoohyung Han, Changmin Lee, Hansol Ryu, and Damien Stehlé. Cryptanalysis of the multilinear map over the integers. In *EUROCRYPT*, 2015. Also, Cryptology ePrint Archive, Report 2014/906.

[9] Jung Hee Cheon, Changmin Lee, and Hansol Ryu. Cryptanalysis of the new CLT multilinear maps. Cryptology ePrint Archive, Report 2015/934, 2015. `http://eprint.iacr.org/`.

[10] Jean-Sebastien Coron. Cryptanalysis of GGH15 multilinear maps. Cryptology ePrint Archive, Report 2015/1037, 2015. `http://eprint.iacr.org/`.

[11] 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 *CRYPTO I*, pages 247–266, 2015.

[12] Jean-Sébastien Coron, Tancrède Lepoint, and Mehdi Tibouchi. Practical multilinear maps over the integers. In *CRYPTO (1)*, pages 476–493, 2013.

[13] Jean-Sébastien Coron, Tancrède Lepoint, and Mehdi Tibouchi. New multilinear maps over the integers. In *CRYPTO I*, pages 267–286, 2015.

[14] Yevgeniy Dodis, Shafi Goldwasser, Yael Tauman Kalai, Chris Peikert, and Vinod Vaikuntanathan. Public-key encryption schemes with auxiliary inputs. In *TCC*, pages 361–381, 2010.

[15] Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM J. Comput.*, 38(1):97–139, 2008.

[16] Yevgeniy Dodis and Adam Smith. Correcting errors without leaking partial information. In *STOC*, pages 654–663, 2005.

[17] Léo Ducas and Alain Durmus. Ring-LWE in polynomial rings. In *Public Key Cryptography - PKC 2012*, pages 34–51, 2012.

[18] Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In *EUROCRYPT*, pages 1–17, 2013.

[19] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *FOCS*, pages 40–49, 2013. Also, Cryptology ePrint Archive, Report 2013/451.

[20] Craig Gentry, Sergey Gorbunov, and Shai Halevi. Graph-induced multilinear maps from lattices. In *TCC*, pages 498–527, 2015.

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

[22] Shafi Goldwasser, Yael Tauman Kalai, Chris Peikert, and Vinod Vaikuntanathan. Robustness of the learning with errors assumption. In *Innovations in Computer Science - ICS*, pages 230–240, 2010.

[23] Shafi Goldwasser and Guy N. Rothblum. On best-possible obfuscation. In *TCC*, pages 194–213, 2007.

[24] Satoshi Hada. Zero-knowledge and code obfuscation. In *ASIACRYPT*, pages 443–457, 2000.

[25] Shai Halevi. Graded encoding, variations on a scheme. Cryptology ePrint Archive, Report 2015/866, 2015. `http://eprint.iacr.org/`.

[26] Dennis Hofheinz, John Malone-Lee, and Martijn Stam. Obfuscation for cryptographic purposes. In *TCC*, pages 214–232, 2007.

[27] Susan Hohenberger, Guy N. Rothblum, Abhi Shelat, and Vinod Vaikuntanathan. Securely obfuscating re-encryption. In *TCC*, pages 233–252, 2007.

[28] Yupu Hu and Huiwen Jia. Cryptanalysis of GGH map. Cryptology ePrint Archive, Report 2015/301, 2015. `http://eprint.iacr.org/`.

[29] Ben Lynn, Manoj Prabhakaran, and Amit Sahai. Positive results and techniques for obfuscation. In *EUROCRYPT*, pages 20–39, 2004.

[30] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. *J. ACM*, 60(6):43, 2013.

[31] Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In *EUROCRYPT*, pages 700–718, 2012.

[32] Brice Minaud and Pierre-Alain Fouque. Cryptanalysis of the new multilinear map over the integers. Cryptology ePrint Archive, Report 2015/941, 2015. `http://eprint.iacr.org/`.

[33] Damien Stehlé and Ron Steinfeld. Making NTRU as secure as worst-case problems over ideal lattices. In *EUROCRYPT*, pages 27–47, 2011.

[34] Hoeteck Wee. On obfuscating point functions. In *STOC*, pages 523–532, 2005.