Indistinguishability Obfuscation from Circular Security

Romain Gay[*]
IBM Zurich
romain.rgay@gmail.com

Rafael Pass[†]
Cornell Tech
rafael@cs.cornell.edu

September 5, 2020

Abstract

We show the existence of indistinguishability obfuscators ($i\mathcal{O}$) for general circuits assuming subexponential security of:

(a) the Learning with Error (LWE) assumption (with subexponential modulus-to-noise ratio);

(b) the Decisional Composite Residuosity (DCR) assumption; and,

(c) a *circular security conjecture* regarding the Gentry-Sahai-Water's (GSW) and the Damgård-Jurik (DJ) encryption schemes.

More precisely, the circular security conjecture states that a notion of leakage-resilient security satisfied by GSW (assuming LWE) is retained in the presence of a key-cycle w.r.t. GSW and DJ. Alternatively, we can remove assumption (b) and replace assumption (c) with the circular security conjecture regarding GSW and a "packed" variant of Regev's encryption scheme.

Our work thus places $i\mathcal{O}$ on qualitatively similar assumptions as (unlevelled) FHE, for which known constructions also rely on a circular security conjecture.

[*]Work done in part while at Cornell Tech

[†]Supported in part by NSF Award SATC-1704788, NSF Award RI-1703846, AFOSR Award FA9550-18-1-0267, DARPA SIEVE award HR00110C0086, and a JP Morgan Faculty Award. This research is based upon work supported in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via 2019-19-020700006. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

1 Introduction

The goal of *program obfuscation* is to "scramble" a computer program, hiding its implementation details (making it hard to "reverse-engineer"), while preserving its functionality (i.e, its input/output behavior). In recent years, the notion of *indistinguishability obfuscation ($i\mathcal{O}$)* [BGI+01, GGH+13b] has emerged as the central notion of obfuscation in the cryptographic literature: roughly speaking, this notion requires that obfuscations $i\mathcal{O}(\Pi_1)$, $i\mathcal{O}(\Pi_2)$ of any two *functionally equivalent* circuits Π_1 and Π_2 (i.e., whose outputs agree on all inputs) from some class $\mathcal{C}$ (of circuits of some bounded size) are computationally indistinguishable.

On the one hand, this notion of obfuscation is strong enough for a plethora of amazing applications (see e.g., [SW14, BCP14, BZ14, GGHR14, KNY14, KMN+14, BGL+15, CHJV14, KLW15, CLP15, BPR15, BPW16, BP15].) On the other hand, it may also plausibly exist, whereas stronger notion of obfuscations have run into strong impossibility results, even in idealized models (see e.g., [BGI+01, GK05, CKP15, Ps16, MMN15, LPST16]) Since the original breakthrough candidate constuction of an $i\mathcal{O}$ due to Garg, Gentry, Halevi, Raykova, Sahai, and Waters [GGH+13b], there has been an intensive effort toward obtaining a construction of $i\mathcal{O}$ based on some form of well-studied/nice assumptions. The original work [GGH+13b] provided a *candidate* construction based on high-degree multilinear maps (MLMs) [GGH13a, CLT13, GGH15, CLT15]; there was no proof of security based on an intractability assumption. [PST14] provided the first construction with a reduction-based proof of security, based on a strong notion of security for MLMs, similar to a sort of "Uber assumption". [GLSW14] provided a construction based on a more concrete assumption relying on composite-order MLMs. Unfortunately, both assumptions have been broken for specific candidate constructions of MLMs [CHL+15, MF15].

iO from FE or XiO. Subsequently, several works have been constructing $i\mathcal{O}$ from seemingly weaker primitives, such as Functional Encryption (FE) [AJ15, BV15] or $Xi\mathcal{O}$ [LPST16], while only using standard assumptions, such as Learning with Error (LWE). For both constructions, we actually need to rely on *subexponentially-secure* constructions of either FE or $Xi\mathcal{O}$, as well as subexponential security of LWE. Let us recall the notion of $Xi\mathcal{O}$ as it will be useful to us: roughly speaking, an $Xi\mathcal{O}$ is an $i\mathcal{O}$ with a very weak "exponential" efficiency requirement: the obfuscator is allowed to run in polynomial time in the size of the truth table of the function to be obfuscated, and it is only required that its outputs a program that "slightly" compresses the truth table (technically, it is sublinear in its size).

A breakthrough result by Lin [Lin16] showed how to obtain $i\mathcal{O}$ from just constant-degree MLMs (plus standard assumptions), overcoming the black-box barriers in [Ps16, MMN15]. Her construction relies on the connection between FE and $i\mathcal{O}$. Following this result, a sequence of works (see e.g., [LV16, Lin17, LT17, Agr19, AJKS18, JS18, JLMS19, AJL+19, AP20]) reduced the assumptions and the degree of the MLM, relying on certain types of low-degree pseudorandom generators (PRGs) to instantiate either FE or $Xi\mathcal{O}$. This culminated in the recent work of [GJLS20], which (other than standard assumptions) relies an LWE assumption with *binary noise* in the presence of leakage of the LWE noise: the leakage is a *constant-degree* PRG applied to the LWE noise. While this most recent construction overcomes known attacks, many of the earlier low-degree PRG assumptions used to obtain $i\mathcal{O}$ have been broken using semi-definite programming/sum-of-square algorithms [BBKK18, BHJ+18], and the construction in [GJLS20] requires carefully setting the parameters so as to avoid these attacks.

A very recent work beautiful work by Brakerski et al [BDGM20a] presents a new type of candidate construction of $Xi\mathcal{O}$ by combining a fully-homomorphic encryption (FHE) and a linear-hommorphic encryption (LHE) with certain nice properties (which can be instantiated by the Damgård-Jurik

(DJ) [DJ01] encryption scheme whose security can be based on the Decisional Composite Residuosity (DCR) assumption), and relying on a random oracle. More precisely, they define a new primitive called "split-FHE" and provide a candidate construction of it based on the above primitives and a random oracle, and next show how split-FHE implies $Xi\mathcal{O}$ (which by earlier work implies $i\mathcal{O}$ under standard assumptions). We highlight that [BDGM20a] does not provide any proof of security of the split-FHE construction (even in the random oracle model), but rather informally argue some intuitions, which include a) *circular security* (more on this below) of the FHE and the LHE, and b) a *"correlations conjecture"* that the FHE randomness (after FHE evaluations) does not correlate "too much" with the messages being encrypted. The correlation conjecture is not formalized, as the FHE randomness in known construction actually *does* depend on the message, so the authors simply conjecture that this correlation cannot be exploited by an attacker to break security of the $i\mathcal{O}$ (they also provide heuristic methods to weaken the correlations); as such they only get a heuristic construction.

Summarizing the above, while there have been enormous progress on realizing $i\mathcal{O}$, known constructions are either based on assumptions on primitives that are not well understood (high-degree MLMs, or various low-degree PRGs assumptions), or the construction candidates simply do not have proofs of security.

1.1 Our Results

In this work, we provide a new $i\mathcal{O}$ construction assuming subexponential security of (a) the LWE assumption (with subexponential modulus-to-noise ratio), (b) the DCR assumption, and (c) a natural *circular security conjecture* w.r.t the Gentry-Sahai-Water's (GSW) [GSW13] FHE scheme, and the DJ [DJ01] LHE scheme. Alternatively, we can remove assumption (b) and replace assumption (c) with a circular security conjecture regarding the GSW encryption scheme and a "packed" variant of Regev's encryption scheme [Reg05, PVW08].

On a high-level, our approach follows that in [BDGM20a], but we show how to remove the heuristic arguments while instead relying on concrete circular security assumptions. We believe this constitutes strong evidence for the existence of $i\mathcal{O}$, and places $i\mathcal{O}$ on a similar footing as *unlevelled* FHE (i.e., an FHE that support an a-priori unbounded polyonomial number of operations), for which known constructions also rely on a circular security conjecture [Gen09].

Circular security. Circular security of encryption schemes [CL01, BRS02] considers a scenario where the attacker gets to see not only encryptions of messages, but also *encrypted key cycles*. A particularly simple type of circular security, referred to as *2-circular security* considers an encrypted key cycle of size 2—the attacker gets to see public keys $\mathsf{pk}_1, \mathsf{pk}_2$, a length 2 encrypted secret key cycle, $\mathsf{Enc}^1_{\mathsf{pk}_1}(\mathsf{sk}_2), \mathsf{Enc}^2_{\mathsf{pk}_2}(\mathsf{sk}_1)$, and we require that security of $\mathsf{Enc}^1_{\mathsf{pk}_1}$ still holds (i.e., for any m_0, m_1, $\mathsf{Enc}^1_{\mathsf{pk}_1}(m_0)$ is indistinguishable from $\mathsf{Enc}^1_{\mathsf{pk}_1}(m_0)$. Such encrypted key cycles commonly arise in applications of encryption scheme such as storage systems such as Bitlocker, anonymous credentials [CL01] and most recently to construct (unlevelled) FHE [Gen09]. We refer to the assumption that:

If Enc^1 *and* Enc^2 *are semantically secure, then 2-circular security holds w.r.t.* $\mathsf{Enc}^1, \mathsf{Enc}^2$

as the *2-circular security conjecture (*2CIRC*) w.r.t* $\mathsf{Enc}^1, \mathsf{Enc}^2$. For our purposes, we will allow the key generation procedure of Enc^1 to get the public-key pk_2 of Enc^2 as an input—for instance, this will allow Enc^1 and Enc^2 to operate over the same field.

At first sight, one may be tempted to hope that the 2CIRC holds w.r.t. *any* two encryption schemes $\mathsf{Enc}^1, \mathsf{Enc}^2$—after all, the attacker never actually gets to see the secret key, but rather encryptions of it, which intuitively should hide it by semantical security of the encryption schemes.

Yet, in recent years, counter examples to 2-circular security for public-key encryption schemes have been found [ABBC10, GH10, CGH12, MO14, KRW15, BHW15, KW16, GKW17]. However, all the counter examples are highly artificial, and require carefully embedding some trapdoor mechanism in the encryption scheme that enable decrypting the cirphertext once you see an encryption of the secret key. As far as we are aware, no "natural" counterexamples to $\mathsf{2CIRC}$ are known. Indeed, a common heuristic consists of simply assuming that $\mathsf{2CIRC}$ holds for all "natural" encryption schemes—we refer to this as the $\mathsf{2CIRC}$ *heuristics*. We note that the $\mathsf{2CIRC}$ heuristic is very similar to the Random Oracle Heuristic [BR93]—while "contrived" counterexamples are known (see e.g., [CGH98, MRH04]), it is still a commonly used heuristic for the design on practical protocols.

In this work, we will consider a natural strengthening of the $\mathsf{2CIRC}$ heuristic: We consider whether stronger forms of security of Enc^1 are preserved in the presence of a key cycle. More precisely, we will consider a notion of $\mathcal{O}$-leakage resilient security for Enc^1 where $\mathcal{O}$ is some particular *randomness leakage oracle*; this notion enhances the standard semantic security notion by providing the attacker with some leakage $\mathcal{O}(\mathbf{m}, \mathbf{r})$ on the randomness $\mathbf{r}$ used to encrypt the message $\mathbf{m}$. We say that the $\mathsf{2CIRC}^{\mathcal{O}}$ *conjecture holds w.r.t* $\mathsf{Enc}^1, \mathsf{Enc}^2$ if the following holds:

> If Enc^1 is $\mathcal{O}$-leakage resilient secure and Enc^2 is secure, then $\mathcal{O}$-leakage resilient security of Enc^1 is preserved in the presence of a length 2 key-cycle w.r.t. Enc^1 and Enc^2.

We will also consider a *subexponential* $\mathsf{2CIRC}^{\mathcal{O}}$ conjecture, which is identically defined except it considers subexponential (as opposed to polynomial) security of the encryption schemes.

Our main theorem shows that for a natural notion of randomness leakage $\mathcal{O}_{\mathsf{SRL}}$—which will be referred to as "shielded randomness leakage (SRL)"—$\mathsf{2CIRC}^{\mathcal{O}_{\mathsf{SRL}}}$ w.r.t. two standard encryption schemes (GSW and DJ) together with standard assumptions implies the existence of $i\mathcal{O}$.

Theorem 1.1 (Informally stated)**.** *Assume subexponential security of the LWE (with subexponential modulus-to-noise ratio) and the DCR assumptions, and assume that the subexponential* $\mathsf{2CIRC}^{\mathcal{O}_{\mathsf{SRL}}}$ *conjecture holds w.r.t. GSW and DJ. Then,* $i\mathcal{O}$ *exists for the class of polynomial-size circuits.*

Or, informally, assuming the "subexponential $\mathsf{2CIRC}^{\mathcal{O}_{\mathsf{SRL}}}$ heuristic", subexponential security of LWE and DCR implies the existence of $i\mathcal{O}$.

Alternatively, we can replace the DJ encryption scheme with a "packed" variant of Regev's encryption scheme [Reg05], which we refer to as Packed Regev. (We note that our Packed Regev is very similar to, but actually different from, the Packed Regev in [PVW08].)

Theorem 1.2 (Informally stated)**.** *Assume subexponential security of the LWE (with subexponential modulus-to-noise ratio) assumption, and assume that the subexponential* $\mathsf{2CIRC}^{\mathcal{O}_{\mathsf{SRL}}}$ *conjecture holds w.r.t. GSW and Packed Regev. Then,* $i\mathcal{O}$ *exists for the class of polynomial-size circuits.*

1.2 Shielded Randomness Leakage (SRL) Security

As mentionned above, we consider a notion of *shielded randomness leakage (SRL)* security for FHE. Roughly speaking, the attacker gets to see an FHE encryption $\mathbf{c} = \mathsf{FHE}(\mathbf{m}; \mathbf{r})$, and next gets access to a "leakage oracle" $\mathcal{O}_{\mathsf{SRL}}(\mathbf{m}, \mathbf{r})$ which upon every invocation sends the attacker an "extra noisy" encryption $\mathbf{c}^\star = \mathsf{FHE}(0; \mathbf{r}^\star)$ of 0—we will refer to the random string $\mathbf{r}^\star$ as the "shield". Next, the attacker can select some functions f and value α such that $f(m) = \alpha$ (i.e., we restrict the attacker to picking functions for which it knows the output when applying the function to the message $\mathbf{m}$; this restriction will soon become clear). Finally, the oracle homomorphically evaluates f on the ciphertext $\mathbf{c}$, letting $\mathbf{c}_f = \mathsf{FHE}(f(\mathbf{m}); \mathbf{r}_f)$ denote the evaluated ciphertext, and returns $\mathbf{r}^\star - \mathbf{r}_f$. That is, the attacker gets back the randomness $\mathbf{r}_f$ of the evaluated ciphertext masked by the "shield" $\mathbf{r}^\star$. The reason why the attacker is restricted to picking functions f for which it knows the output α is that for the FHE we consider, given $\mathbf{c}^\star$ and $\mathbf{c}_f$, the attacker can compute $\mathbf{c}^\star - \mathbf{c}_f = \mathsf{FHE}(0 - f(\mathbf{m}); \mathbf{r}^\star - \mathbf{r}_f)$ and thus knowing $\mathbf{r}^\star - \mathbf{r}_f$ reveals $f(\mathbf{m})$. So, by restricting to attackers that already know $\alpha = f(\mathbf{m})$, intuitively, $\mathbf{r}^\star - \mathbf{r}_f$ does not reveal anything else. Indeed, we formally prove that under the LWE assumption, the GSW encryption scheme is *SRL-secure*—that is, $\mathcal{O}_{\mathsf{SRL}}$-leakage resilient secure.

Theorem 1.3 (Informally stated)**.** *Assume the LWE (with subexponential modulus-to-noise ratio) assumption holds. Then, the GSW scheme is SRL-secure.*

On a very high-level, the idea behind the proof is that the encryption $c^\star$ is a projection, $h_{\mathbf{A}}(\mathbf{r}^\star) = \mathbf{A}\mathbf{r}^\star \in \mathbb{Z}_N$, where the randomness $\mathbf{r}^\star$ used to produce $\mathbf{c}^\star$ is a vector in $\mathbb{Z}_N^m$ and $\mathbf{A}$ is a matrix in $\mathbb{Z}_N^{n \times m}$ where $m \gg n$, that is, the map $h_{\mathbf{A}}$ that describes the encryption is compressing. Therefore, some "components" of the "shield" $\mathbf{r}^\star$ remain information-theoretically hidden. And this enables hiding the same component of $\mathbf{r}_f$; furthemore, the component that is not hidden by $\mathbf{r}^\star$ is actually already revealed by $f(m)$, which the attacker knows (as we require it to output $\alpha = f(m)$). The formal proof of this proceeds by considering a (simplified) variant of the Micciancio-Peikert lattice trapdoor method [MP12] for generating the matrix $\mathbf{A}$ (which is part of the public key for GSW) together with a trapdoor that enables sampling short preimages to $h_{\mathbf{A}}$ (i.e. solving the ISIS problem). Whereas traditional trapdoor preimage sampling methods require the preimage to be sampled according to some specific distribution (typically discrete Gaussian) over preimages, we will consider a somewhat different notion: we require that given a target vector $\mathbf{t}$, the distribution of randomly sampled preimages of $\mathbf{t}$ is statistically close to the distribution obtained by starting with any "short" preimage $\mathbf{w}$ of $\mathbf{t}$ and next adding a randomly sampled preimage of 0. Our proof relies on the fact that randomly sampled preimages can be sufficiently longer than $\mathbf{w}$ to ensure that they "smudge" $\mathbf{w}$—we here rely on the fact that modulus-to-noise ratio is subexponential (to enable the smudging).[1]

1.3 Overview of the $Xi\mathcal{O}$ Construction

We proceed to explain our construction of an $Xi\mathcal{O}$. This construction will only rely on *polynomial* security of LWE, an LHE with certain nice properties, $\mathsf{2CIRC}^{\mathcal{O}_{\mathsf{SRL}}}$ w.r.t. GSW and the LHE; to obtain a *subexponentially-secure* $Xi\mathcal{O}$ (which is required to obtain $i\mathcal{O}$ by [LPST16]), we need to strengthen these assumptions to also require subexponential security. We finally note that the DJ LHE satisfies the desired "nice" properties, and additionally show how a packed version of Regev's encryption scheme does so as well.

As mentionned above, on a high-level, our construction follows similar intuitions as the BDGM construction. We combine an FHE (in our case the GSW FHE) with a (special-purpose) LHE to implement an $Xi\mathcal{O}$. In fact, in our approach, we do not directly construct an $Xi\mathcal{O}$, but rather construct an $Xi\mathcal{O}$ with *preprocessing*—this notion, which relaxes $Xi\mathcal{O}$ by allowing the obfuscator to have access to some *long* public parameter pp, was actually already considered in [LPST16] and it was noted there that subexponentially-secure $Xi\mathcal{O}$ with preprocessing also suffices to get $i\mathcal{O}$.

Towards explaining our approach, let us first recall the approach of BDGM using a somewhat different langauge that will be useful for us. As in BDGM, we focus on an instantiation of the LHE using the DJ encryption scheme.

The BDGM construction. The high-level idea is quite simple and very elegant. Recall that an $Xi\mathcal{O}$ is only required to work for programs Π with polynomially many inputs $n = \mathsf{poly}(\lambda)$ where λ is the security parameter, and the obfuscators running time is allowed to be polynomial in n; the only restriction is that the obfuscated code should be sublinear in n—we require a "slight" compression of

[1] As we can use smudging, our lattice trapdoor sampling construction and proof also becomes easier than the one one [MP12].

the truth table. More precisely, the obfuscator is allowed to run in time $\mathsf{poly}(n, \lambda)$ (i.e., polynomial time in the size of the truth table), but must output a circuit of size $\mathsf{poly}(\lambda)n^{1-\varepsilon}$ where $\varepsilon > 0$. Assume that we have access to a special "batched" FHE which enables encrypting (and computing on) long messages of length, say m using a *short randomness* of length $\mathsf{poly}(\lambda) \log m$; and furthermore that 1) given the secret key, and a ciphertext c, we can efficiently recover the ciphertext randomness, and 2) given a ciphertext c and its randomness—which will also be referred to as a "hint"—one can efficiently decrypt. Given such a special FHE, it is easy to construct an $Xi\mathcal{O}$: simply cut the truth table into "chunks" of length n^{ε}, FHE encrypt the program Π, then, homomorphically evalute circuits C_i for indices $i \in [n^{1-\varepsilon}]$ such that given the program Π as input, C_i outputs the i'th "chunk" of the truth table, which we denote by Π_i; finally, release the randomness r_i (i.e., the "hint") of the evaluated ciphertexts. These hints enable compressing n^{ε} bits into $\mathsf{poly}(\lambda) \log(n^{\varepsilon})$ bits and thus the $Xi\mathcal{O}$ is compressing.[2]

Unfortunately, none of the known FHE constructions have short randomness. BDGM, however, observes that there are *linear* homomorphic encryptions schemes (LHE), notably the DJ encryption scheme, that satisfy the above requirements. Now, note that many FHEs are batcheable (with "long" randomness), and have "essentially" linear decryption: decryption is a linear operation on the ciphertext and secret key, and next rounding. So, if we start off with such an FHE, and additionally release an LHE encryption of the FHE secret key, we can get an FHE with the desired "batcheable with short randomness" requirement: we first (linear) homomorphically evaluate the decryption of the FHE ciphertext, and simply release the randomness for the evaluated LHE ciphertext (which now is short).

But there are problems with this approach: (1) since FHE decryption requires performing both a linear operation and *rounding*, we are leaking not only Π_i but also the FHE noises of the evaluated ciphertexts, and (2) the LHE randomness may actually leak more than just the decrypted message (i.e., something about how the ciphertext was obtained). As BDGM shows, both of these problems can be easily overcome if we have access to many fresh LHE encryptions of some "smudging" noise (which is large enough to smudge the FHE noise).[3] So the only remaining problem is to generate these LHE encryptions of smudging noises. It is here that the construction in BDGM becomes heuristic: (1) they propose to use a random oracle to generate a long sequence of randomness; (2) this sequence of randomness can be interpreted as a sequence of LHE encryptions of uniformly random strings u_i for $i = 1, \dots, n^{1-\varepsilon}$, since the DJ LHE has dense ciphertext; (3) next, if we additionally provide an FHE encryption of the LHE secret key $\overline{\mathsf{sk}}$ (note that we now have a circular security issue), we can FHE-homomorphically evaluate a function f_i that decrypts the i'th LHE ciphertext produced by the random oracle, and computes the most significant bits of u_i; and, (4) we finally LHE-evaluate the (partial) decryption of the evaluated FHE ciphertexts (of the most significant bits of u_i); the obtained LHE ciphertexts can now be subtracted to the LHE ciphertexts from the random oracle to get an LHE encryption of noise $u_i - \mathsf{MSB}(u_i)$ of the appropriate size, i.e. smudging but not uniform.

One problem with this approach, however, is that while we do obtain an LHE encryption of appropriate smudging noise, it is not not actually a fresh ciphertext (with fresh randomness). The issue is that the randomness r_{f_i} of the evaluated ciphertext of the most significant bits may (and actually will) depend on the randomness of the original LHE ciphertext obtained by the RO. Another problem is that LHE can only compute the first step of an FHE decryption (namely, the linear operations), the LHE encryption obtained actually encrypts a message of the form: $u_i - \mathsf{MSB}(u_i) +$

[2]The reason we need to cut the truth table into chunks and don't just directly compute the whole output is that the FHE may have a public key that depends polynomially on the length of the messages to be encrypted, so the final obfuscation is only compressing when we have a large number of chunks.

[3]They formally prove the security of their scheme in an idealized model where we have access to an oracle that generates fresh LHE encryptions of smudging noise.

noise_i. As we know, revealing the extra noise is detrimental for security (this is why we are generating LHE encryptions of smudging noises in the first place). Unfortunately, the extra noise that results from partially decrypting the FHE ciphertext depends on u_i, so the lower-order bits of the latter cannot smudge the former. BDGM here simply assumes that the attacker cannot exploit these correlations, and thus only obtain a heuristic construction.

We shall now see how to obtain the appropriate LHE encryption of smudging noises in a provably secure way, relying on *circular SRL-security* of GSW and DJ—that is, $\mathcal{O}_{\mathsf{SRL}}$-leakage resilient circular security holds w.r.t. GSW and DJ.

Removing the RO. Our first task will be to remove the use of the RO. That will actually be very easy: as we have already observed, it suffices to get an $Xi\mathcal{O}$ with preprocessing to obtain $i\mathcal{O}$, so instead of using a random oracle, we will simply use a long random string as a public parameter, and interpret it as LHE encryptions of random strings.

Re-encrypting the FHE. The trickier problem will be to deal with the issue of correlations. We will here rely on the fact that we are considering a particular instantiation of the FHE: namely, using (a batched version of) the GSW encryption scheme. On a high-level, the idea for breaking the correlation is to "refresh" or re-encrypt the evaluated FHE ciphertext (of the most significant bits of the random LHE plaintext) to ensure that the randomness is fresh and independent of the evaluations. That also gives a decryption noise than itself is also independent of the evaluated circuit. If we had access to a fresh extra noisy FHE encryption of 0, then it would be easy to re-randomize a GSW ciphertext: simply add the encryption of 0. So, how do we get an encryption of 0? GSW ciphertexts are not dense, so we cannot put them in the public parameters, and even if they were, we still wouldn't be able to get an encryption of 0 (we would have an encryption of a uniformly random plaintext). The public key of the GSW encryption scheme actually contains a bunch of encryptions of 0, but fewer than the amount we need (or else we wouldn't get a compressing $Xi\mathcal{O}$). Instead, we use the public key of the GSW encryption to generate extra noisy encryptions of 0, and we include the (many) random coins $(r_i^\star)_{i\in[n^{1-\varepsilon}]}$ used to generate these ciphertexts as part of the public parameters of the $Xi\mathcal{O}$ (recall that the public parameters can be as long as we want). This method does indeed enable us to get a fresh FHE encryption of the most significant bits, and thus the correlation has be broken and intuitively, we should be able to get a provably secure construction. But two obstacles remain: (1) we are revealing the randomness used to re-randomize the ciphertexts, and this could hurt security, or render the re-randomization useless and (2) we still have a circular security issue (as we FHE-encrypt the LHE secret key, and LHE-encrypt the FHE secret key). Roughly speaking, the first issue will be solved by relying on SRL-security of GSW, and the second issue will be solved by our circular security conjecture.

In more detail, we note that the re-randomized evaluated FHE ciphertext of $\mathsf{MSB}(u_i)$ and the public parameters $r_i^\star$ are statistically close to freshly generated extra noisy FHE encryption of $\mathsf{MSB}(u_i)$ using randomness $r_i^\star$, and setting the public parameter to $r_i^\star - r_{f_i}$, where r_{f_i} is the randomness of the evaluated ciphertext, before re-randomization. In other words, the re-randomization achieves a notion which we refer to as "weak circuit privacy", where the re-randomized ciphertext is independent of the evaluated function f_i. Furthermore, noisy GSW encryptions of $\mathsf{MSB}(u_i)$ essentially have the form of a noisy GSW encryption of 0, to which $\mathsf{MSB}(u_i)$ is added. So, other than $\mathsf{MSB}(u_i)$, which is truly random, $r_i^\star - r_{f_i}$ is simply an SRL leakage on a GSW encryption of the LHE secret key $\overline{\mathsf{sk}}$! Thus, intuitively, security should now follow from circular SRL security of GSW and DJ.

The final construction. We summarize the final construction of an XiO with preprocessing. The public parameter pp is a long *random* string that consists of two parts:

- The first part $\mathsf{FHE.PubCoin}$ will interpreted as a sequence of rerandomization vectors $\mathbf{r}^\star$;
- The second part $\mathsf{LHE.PubCoin}$ will be interpreted as sequence of LHE encryptions;

The obfuscator, given a security parameter λ and a circuit $\Pi : \{0,1\}^{\log n} \to \{0,1\}$, where $n = \mathsf{poly}(\lambda)$ proceeds as follows:

- **Output the public keys of the FHE and LHE:** The obfuscator generates a fresh key-pair $(\overline{\mathsf{pk}}, \overline{\mathsf{sk}})$ for the DJ LHE, and next generate a key-pair $(\mathsf{pk}, \mathsf{sk})$ for the GSW FHE. (To make it easier for the reader to remember which key refers to which encryption scheme, we place a *line* over all keys, ciphertext and algorithms, that correspond to the *linear* homomorphic encryption.) The modulus N of the GSW encryption is set to be the same that the modulus that defines the message space $\mathbb{Z}_N$ of the LHE scheme. Additionally, it chooses N large enough to enable encrypting messages of size n^ε. Finally, it outputs the public keys $(\mathsf{pk}, \overline{\mathsf{pk}})$.
- **Output an FHE encryption of the circuit:** It outputs an FHE encryption (w.r.t. pk) of the program Π, which we denote by ct_1.
- **Output encrypted key cycle:** It computes ct_2, an FHE encryption of $\overline{\mathsf{sk}}$, and $\overline{\mathsf{ct}}$, an LHE encrytion of sk. It outputs the key cycle $\mathsf{ct}_2, \overline{\mathsf{ct}}$.
- **Output hints:** For every $i \in [n^{1-\varepsilon}]$, it outputs a short "hint" r_i computed as follows:
  - **Evaluate the circuit:** Homomorphically evaluate the circuit C_i (recall that C_i takes a input a program Π and outputs the output of Π on i'th chunk of its truth table, denoted by Π) on ct_1 (recall that ct_1 encrypts Π), and let ct_i denote the resulting evaluated FHE ciphertext.
  - **Compute an FHE encryption $\mathsf{ct}_{\mathsf{MSB},i}$ of $\mathsf{MSB}(u_i)$:** Consider the function $f_i(\Pi, \overline{\mathsf{sk}})$ that ignores the input Π but uses the input $\overline{\mathsf{sk}}$ to decrypt the i'th LHE ciphertext from $\mathsf{LHE.PubCoin}$ into a plaintext u_i and outputs $\mathsf{MSB}(u_i)$. The obfuscator homomorphically evaluates f_i on the ciphertexts $\mathsf{ct}_1, \mathsf{ct}_2$ (where, recall, ct_2 is an encryption of $\overline{\mathsf{sk}}$). Let $\mathsf{ct}_{\mathsf{MSB},i} = \mathsf{FHE}(\mathsf{MSB}(u_i); \mathbf{r}_{f_i})$ denote the resulting evaluated FHE ciphertext.
  - **Rerandomize $\mathsf{ct}_{\mathsf{MSB},i}$ into $\mathsf{ct}'_{\mathsf{MSB},i}$:** It uses the i'th chunk of $\mathsf{FHE.PubCoin}$ to get the randomness $\mathbf{r}^\star_i$; generates an extra noisy FHE encryption of 0 using $\mathbf{r}^\star_i$ and homomorphically adds it to $\mathsf{ct}_{\mathsf{MSB},i}$. Let $\mathsf{ct}'_{\mathsf{MSB},i} = \mathsf{FHE}(\mathsf{MSB}(u_i); \mathbf{r}^\star_i + \mathbf{r}_{f_i})$ denote the new (re-randomized) ciphertext.
  - **Proxy re-encrypt ct_i as an LHE ciphertext $\overline{\mathsf{ct}}_i$:** It uses $\overline{\mathsf{ct}}$ (which, recall, is an LHE encryption of sk) to homomorphically computing the *linear* part of the FHE decryption of ct_i, which yields an LHE encryption of the value $\omega \cdot \Pi_i + \mathsf{noise}_i$ where noise_i is an partial FHE decryption noise, and ω is taken large enough so that the plaintext Π_i can be recovered by rounding.

    Similarly, it homomorphically computes the partial FHE decryption of $\mathsf{ct}'_{\mathsf{MSB},i}$, which yields an LHE encryption of the value $\omega' \cdot \mathsf{MSB}(u_i) + \mathsf{noise}_{\mathsf{MSB},i}$, where once again $\mathsf{noise}_{\mathsf{MSB},i}$ denotes a partial FHE decryption noise, and $\omega' = 1$ for reasons that will become clear later. We rely on the fact that GSW FHE (and many others) admits a flexible "scaled" evaluation algorithm, that can choose which integer ω to use when performing the homomorphic evaluation (this was used also in prior works, including [BDGM20a]). The resulting LHE ciphertext is subtracted to $\mathsf{LHE}(\omega \cdot \Pi_i + \mathsf{noise}_i)$, and therefore yields $\mathsf{LHE}(\omega \cdot \Pi_i + \mathsf{noise}_i - \mathsf{MSB}(u_i) - \mathsf{noise}_{\mathsf{MSB},i})$.

Finally, it homomorphically adds the LHE encryption of u_i that is part of the LHE public coins, to obtain $\overline{\mathsf{ct}}_i = \mathsf{LHE}(m_i)$, where $m_i = \omega \cdot \Pi_i + \mathsf{noise}_i - \mathsf{MSB}(u_i) - \mathsf{noise}_{\mathsf{MSB},i} + u_i = \omega \cdot \Pi_i + \mathsf{noise}_i + \mathsf{noise}_{\mathsf{MSB},i} + \mathsf{LSB}(u_i)$.

The integer ω' is chosen to be equal to 1 so that the smudging noise $\mathsf{LSB}(u_i)$ is directly added to the FHE noises $\mathsf{noise}_i - \mathsf{noise}_{\mathsf{MSB},i}$. As opposed to the value Π_i that we place in the higher-order bits of the plaintext, we need the smudging noise to be at the same level that the FHE noises, so they "blend" together.

– **Release hint r_i for LHE ciphertext $\overline{\mathsf{ct}}_i$:** It uses $\overline{\mathsf{sk}}$ to recover the randomness r_i of $\overline{\mathsf{ct}}_i$ (recall that the LHE we use has a randomness recoverability property), and outputs r_i.

To evaluate the obfuscated program on an input $\mathbf{x} \in \{0,1\}^n$, that pertains to the i'th chunk of the truth table of Π for some $i \in [n^{1-\varepsilon}]$, we compute $\overline{\mathsf{ct}}_i$ just like the obfuscator did (note that this does not require knowing the secret key, but only information contained in the obfuscated code). Finally, decrypt $\overline{\mathsf{ct}}_i$ using the hint r_i to recover the message m_i described above (recall that the LHE we use has the property that ciphertexts can be decrypted if you know the randomness). Finally, perform the rounding step of FHE decryption on m_i to obtain Π_i, which contains $\Pi(\mathbf{x})$.

Outline of the security proof. We provide a very brief outline of the security proof. We will rely on the fact that LHE cirphertexts (of random messages) are dense (in the set of bit strings), and additionally on the fact that both the LHE and the FHE we rely on (i.e., DJ and GSW) satisfy what we refer to as a *weak circuit privacy* notion. This notion, roughly speaking, says that *any* encryption of a message x can be rerandomized into fresh (perhaps extra noisy) encryption of $x + y$, by adding a fresh (perhaps extra noisy) encryption of y.

As usual, the proof proceeds via a hybrid argument. We start from an $Xi\mathcal{O}$ obfuscation of a program Π_0 and transition until we get an $Xi\mathcal{O}$ obfuscation of Π_1, where Π_0 and Π_1 are two functionally equivalent circuits of the same size.

- **Hybrid 0: Honest $Xi\mathcal{O}(\Pi_0)$:** The first hybrid is just the honest obfuscation of the circuit Π_0.

- **Hybrid 1: Switch to freshly encrypted $\mathsf{ct}'_{\mathsf{MSB},i}$:** Hybrid 1 proceeds exactly as Hybrid 0 up until the point that the ciphertexts $\mathsf{ct}_{\mathsf{MSB},i}$ get re-encrypted into $\mathsf{ct}'_{\mathsf{MSB},i}$, with the exception that $\mathsf{FHE.PubCoin}$ are not sampled yet. Next, instead of performing the re-encryption, we sample $\mathsf{ct}'_{\mathsf{MSB},i}$ as a *fresh* extra noisy encryption of $\mathsf{MSB}(u_i)$ using randomness $\mathbf{r}_i^\star$, and setting $\mathsf{FHE.PubCoin}$ to be $\mathbf{r}_i^\star - \mathbf{r}_{f_i}$). We finally continue the experiment in exactly the same way as in Hybrid 0.

  It follows from the "weak circuit privacy" property of the FHE that Hybrid 0 and Hybrid 1 are statistically close. (Note that the advantage of Hybrid 1 is that for each $i \in [n^{1-\varepsilon}]$, the i'th chunk of $\mathsf{FHE.PubCoin}$ can be thought of as SRL leakage on the fresh encryption $\mathsf{ct}'_{\mathsf{MSB},i}$ computed w.r.t. function f_i.

- **Hybrid 2: Switch $\mathsf{LHE.PubCoin}$ to encryptions of random strings:** Hybrid 2 proceeds exactly as Hybrid 1 except that instead of sampling $\mathsf{LHE.PubCoin}$ as a random string, we sample it as fresh LHE encryptions of random strings u_i, for $i = 1, \ldots, n^{1-\varepsilon}$. It follows by the density property of the LHE that Hybrid 2 is statistically close to Hybrid 1.

- **Hybrid 3: Generate $\overline{\mathsf{ct}}_i$ as a fresh encryption:** Hybrid 3 proceeds exactly as Hybrid 2 except that $\overline{\mathsf{ct}}_{\mathbf{y}}$ is generated as a fresh encryption of m_i using fresh randomness r_i, and the i'th chunk of $\mathsf{LHE.PubCoin}$ is instead computed homomorphically by subtracting the LHE encryption of $\mathsf{sk}^\top(\mathsf{ct}_i - \mathsf{ct}_{\mathsf{MSB},i})$ (obtained after homomorphically decrypting ct_i and $\mathsf{ct}'_{\mathsf{MSB},i}$ using $\overline{\mathsf{ct}}$) from the LHE ciphertext $\overline{\mathsf{ct}}_i$. (Recall that $m_i = \mathsf{sk}^\top(\mathsf{ct}_i - \mathsf{ct}_{\mathsf{MSB},i}) + u_i$ so the above way of computing the i'th chunk of $\mathsf{LHE.PubCoin}$ ensures that it is valid encryption of u_i as in Hybrid 2, but this time with non-fresh, homomorphically evaluated randomness).

  It follows from the weak circuit privacy property of the LHE that Hybrid 3 and 2 are statistically close.

  Note that it was possible to define this hybrid since $\mathsf{ct}'_{\mathsf{MSB},i}$ remains exactly the same no matter what the $\mathsf{LHE.PubCoin}$ are. This was not true in Hybrid 0, and we introduced Hybrid 1 to break this dependency.

  Note further that in Hybrid 3, we no longer use $\overline{\mathsf{sk}}$ (i.e., the secret key for LHE); previously it was used to recover r_i.

- **Hybrid 4: Generate $\overline{\mathsf{ct}}_i$ without FHE noises:** Hybrid 4 proceeds exactly as Hybrid 3 except that $\overline{\mathsf{ct}}_i$ is generated as a fresh encryption of $m_i = \omega \cdot \Pi_i^0 + \mathsf{LSB}(u_i)$, whereas in Hybrid 3, it was generated as fresh encryption of $m_i = \omega \cdot \Pi_i^0 + \mathsf{LSB}(u_i) + \mathsf{noise}_i - \mathsf{noise}_{\mathsf{MSB},i}$. That is, we use $\mathsf{LSB}(u_i)$ as a smudging noise to hide the extra noise $\mathsf{noise}_i - \mathsf{noise}_{\mathsf{MSB},i}$. We can do so since (1) the extra FHE noise is small and independent of $\mathsf{LSB}(u_i)$ (2) the rest of the obfuscated code can be generated from the value $\mathsf{LSB}(u_i) + \mathsf{noise}_i - \mathsf{noise}_{\mathsf{MSB},i}$ only (in particular it does not require to know $\mathsf{LSB}(u_i)$ itself). It follows that Hybrid 4 is statistically close to Hybrid 3.

- **Hybrid 5: Switch to encryption of Π^1:** Hybrid 5 proceeds exactly as Hybrid 4 except that we let ct_1 be an encryption of Π^1 (instead of Π^0 in prior hybrids).

  Note that other than the encrypted key cycle, we never use the FHE secret key, and due to Hybrid 3, we no longer use the LHE secret key. So, at first sight, Hybrid 5 ought to be indistinguishable from Hybrid 4 by circular security of the FHE and the LHE. Recall that $\mathsf{FHE.PubCoin}$ leaks something about the randomness used by the FHE encryption $\mathsf{ct}'_{\mathsf{MSB},i}$, but the leakage is exactly an SRL leakage (and note that in the experiment we do know the output α_i of the function f_i that is applied to the plaintexts encrypted in $\mathsf{ct}_1, \mathsf{ct}_2$—namely, it is $\mathsf{MSB}(u_i)$ where u_i is a random string selected in the experiment, see Hybrid 2). Thus, indistinguishability of Hybrid 5 and Hybrid 4 follows from circular SRL-security of the FHE and the LHE.

- **Hybrids 6-10:** For $i \in [5]$, Hybrid $5 + i$ is defined exactly as $5 - i$, except that ct_1 be an encryption of Π^1. Statistical closeness of intermediary hybrids follows just as before.

The above sequence of hybrid allows us to conclude the following theorem.

Theorem 1.4 (Informally stated). *Assume circular SRL-security of the GSW and DJ encryption schemes holds. Then, there exists an XiO for polynomial-size circuits taking inputs of length* $\log(\lambda)$ *where λ is the security parameter.*

An alternative LHE based on Packed Regev. We finally remark that we can obtain an alternative construction of an LHE with the desired properties by considering an *packed* version of the Regev encryption scheme. Our construction is slightly different, but similar in spirit, to the Packed Regev from [PVW08]). Recall that a (plain) Regev public key consist of a pair $\mathbf{A}, \mathbf{s}^\top\mathbf{A} + \mathbf{e}^\top$, where $\mathbf{A} \leftarrow_{\mathsf{R}} \mathbb{Z}_q^{m\times n}$ with $m \geq n\log(q)$, the vector $\mathbf{s} \leftarrow_{\mathsf{R}} \mathbb{Z}_q^n$ is the secret key, and $\mathbf{e} \in \mathbb{Z}_q^m$ is some small "noise" vector. An encryption of a message μ has the form $\mathbf{Ar}, (\mathbf{s}^\top\mathbf{A} + \mathbf{e}^\top)\mathbf{r} + B \cdot \mu$ where $\mathbf{r} \leftarrow_{\mathsf{R}} \{0,1\}^m$ is the encryption randomness and B is a bound on the size of noise (so as to enable decryption). This scheme is linearly homomorphic, but for security, the size of the randomness $|\mathbf{r}|$ needs to be greater than $|n \log(q)|$, which is more than that size of the message: the randomness is too long for our purposes.

To get succinct decryption hints, we simply reuse the same randomness $\mathbf{r}$ for many encryptions using different secret keys $\mathbf{s}_1, \mathbf{s}_2, \ldots \mathbf{s}_\ell$ and different noises $\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_\ell$. The secret key is now a matrix $\mathbf{S} \in \mathbb{Z}_q^{\ell \times n}$, and the public key becomes $(\mathbf{A}, \mathbf{SA} + \mathbf{E})$ where $\mathbf{E} \in \mathbb{Z}_q \ell \times m$ is a noise matrix. The encryption of a vector of messages $\mu = (\mu_1, \ldots, \mu_\ell)$ is then $(\mathbf{Ar}, (\mathbf{SA} + \mathbf{E})\mathbf{r} + B\mu)$. This is the scheme from [PVW08]. Despite the fact that this encryption is still linearly homomorphic, and has the advantage of having rate-1 ciphertext size, its randomness is not short: to carry on the proof of security, we need to rely on the fact that $\mathbf{r}$ contains enough bits of entropy even when the information $\mathbf{Ar}$ (which is short) and $\mathbf{Er}$ (that is long) is leaked. The can only be true with the dimension of $\mathbf{r}$, m, grows with the number of bits that are batched, ℓ.

Thus, we depart from the scheme in [PVW08] by adding a smudging noise[4] in the ciphertext, to hide the information $\mathbf{Er}$. The ciphertext is of the form: $(\mathbf{Ar}, (\mathbf{SA} + \mathbf{E})\mathbf{r} + \mathbf{e}' + B \cdot \mu)$, where $\mathbf{e}'$ is the extra smudging noise that hides the error term $\mathbf{Er}$, ensuring that we only have the short Ar leakage and the usual proof can again be applied.

This scheme is still linearly homomorphic, but the encryption randomness is still large, as even though we reuse $\mathbf{r}$, the added noise terms $\mathbf{e}'$ are large. However, we rely on the fact that knowing $\mathbf{e}'$ is not needed for decrypting. Indeed, to decrypt, we just need to know a small vector $\widetilde{\mathbf{r}} \in \mathbb{Z}_q^m$ such that $\mathbf{A}\widetilde{\mathbf{r}} = \mathbf{Ar}$. That can be used to remove the term $\mathbf{SAr}$ from the ciphertext, and recover $B \cdot \nu$ plus some small noise. To sample such vector, we use standard trapdoor sampling mechanism as in prior works [Ajt96, GPV08, AP09, MP12]. This makes the scheme hintable with succinct hints.

We still have two (minor) obstacles, though. This scheme (as well as Regev's original scheme or the scheme from [PVW08]) does not satisfy two of the other properties needed for our $Xi\mathcal{O}$ construction: (1) density, and (2) weak circuit privacy. But it almost does. *Extra noisy* ciphertexts, where the noise reaches the bound B are actually dense, and for us extra noisy ciphertext, weak circuit privacy also holds (just as it did for GSW). So we can directly instantiate the LHE in our $Xi\mathcal{O}$ construction with this Packed Regev construction, as long as we slightly relax the notion of an LHE to just require density when considering extra noisy ciphertexts.

Thus we can conclude:

Theorem 1.5 (Informally stated). *Assume circular SRL-security of the GSW and Packed Regev encryption schemes holds. Then, there exists an $Xi\mathcal{O}$ for polynomial-size circuits taking inputs of length* $\log(\lambda)$ *where* λ *is the security parameter.*

Concluding the proofs of the main theorems. The proof of Theorems 1.1, 1.2 is finally concluded by upgrading Theorems 1.3, 1.4 and 1.5 to apply also in the subexponential regime, relying on the subexponential $\mathsf{2CIRC}^{\mathcal{O}_{\mathsf{SRL}}}$ conjecture, and finally relying on the transformation from subexponentially-secure $Xi\mathcal{O}$ with pre-processing (and subexponential LWE) to $i\mathcal{O}$ [LPST16].

1.4 Concurrent and Subsequent Work

A concurrent and independent breakthrough result by Jain, Lin and Sahai [JLS20] presents a construction of $i\mathcal{O}$ based on subexponential security of well-founded assumptions: (1) the SXDH assumption on asymmetric bilinear groups, (2) the LWE assumption with subexponential modulus-to-noise ration, (3) a Boolean PRG in NC^0, and (4) an LPN assumption over a *large field* and with a small

[4]Note that using a carefully crafted noise that needs not be of smudging size, as done in [MP12], we can "unskew" the noise $\mathbf{Er}$ and hide the information of $\mathbf{r}$. We favor clarify of the exposition over efficiency and resort to using smudging noises.

error rate $\frac{1}{\ell^\delta}$ where $\delta > 0$ and ℓ is the dimension of the LPN secret. Assumptions (1)-(3) are all pretty well understood; whereas (4), despite being very elegant, revisits a well-known assumption (LPN) over a relatively unexplored range of parameters — in fact, to the best of our knowledge, the only prior work that used LPN for large field and sparse $\frac{1}{\ell^\delta}$ rate is the recent work of [BCGI18]; all other prior works rely on less sparse error rate (typically a constant), and/or use the field $\mathbb{F}_2$.

A concurrent and independent work by Wee and Wichs [WW20] presents a different heuristic instantiation of the BDGM paradigm based only on lattice-based primitives. Similarly to us, their construction proceeds by implementing $Xi\mathcal{O}$ with pre-processing. They also state a new security assumption (involving a PRF and LWE samples) on which security of their construction can be based.

The initial version of our paper did not contain the LWE-based instantiation of the LHE using Packed Regev (we just had the DJ based instantiation). Following up on the initial posting of our paper, but concurrently and independently from the current version which includes our LWE-based LHE, a preprint by Brakerski et al [BDGM20b] also provides an LWE-based way to instantiate the LHE within our framework. Differently from our construction, however, they rely on a variant of the "Dual Regev" encryption scheme, whereas we rely on regular Regev (and they only provide a very rough proof sketch). We plan to provide a more thorough comparison with [BDGM20b, WW20] once we have gone over their constructions in more detail.

2 Preliminaries and Definitions

In this section, we recall some standard definitions and results. Additionaly, we include a formalization of the circular security assumption that we consider.

Attackers, negligible functions, and subexponential security. Below, for simplicity of exposition, we provide definitions for *polynomial security* of all the primitives we consider. As usual, we model attackers as *non-uniform probabilistic polynomial-time algorithms*, denoted *nuPPT*. We say that a function $\mu(\cdot)$ is *negligible* if for every polynomial $p(\cdot)$, there exists some λ_0 such that $\mu(\lambda) \leq \frac{1}{p(\lambda)}$ for all $\lambda > \lambda_0$. The security definitions we consider will require that for every nuPPT $\mathcal{A}$, there exists some negligible function μ such that for all λ, $\mathcal{A}$ succeeds in "breaking security" w.r.t. the security parameter λ with probability at most $\mu(\lambda)$.

All the definitions that we consider can be extend to consider *subexponential security*; this is done by requiring the existence of some constant ϵ such that for all non-uniform probabilistic algorithms $\mathcal{A}$ with running time $\mathsf{poly}(\lambda) \times 2^{\lambda^\epsilon}$ (as opposed to all nuPPT), there exists some negligible function μ such that for all λ, $\mathcal{A}$ succeeds in "breaking security" w.r.t. the security parameter λ with probability at most $\mu(\lambda) \times 2^{\lambda^\epsilon}$ (as opposed to it being just $\mu(\lambda)$).

2.1 Some Standard Lemmas

We recall some standard definitions and lemmas.

Definition 2.1 (bounded ensemble)**.** *Let $f(\cdot)$ be a function that outputs in $\mathbb{N}$. An ensemble of distributions that outputs in $\mathbb{Z}$, denoted by $\{\mathcal{D}_\lambda\}_{\lambda\in\mathbb{N}}$, is said to be f-bounded if there exists a negligible function $\mu(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $\Pr[|x| > f(\lambda), x \leftarrow \mathcal{D}_\lambda] < \mu(\lambda)$.*

We will make use of a special case of the left over hash lemma from [ILL89].

Lemma 2.1 (Left Over Hash lemma)**.** *For all $\lambda, q, d, m \in \mathbb{N}$ such that $m \geq d\lceil\log(q)\rceil + 2\lambda$, the statistical distance between the following distributions is upper bounded by $2^{-\lambda}$:*

$$\left\{\mathbf{A} \leftarrow_{\mathsf{R}} \mathbb{Z}_q^{d\times m}, \mathbf{r} \leftarrow_{\mathsf{R}} [-1,1]^m : (\mathbf{A}, \mathbf{Ar})\right\}$$
$$\left\{\mathbf{A} \leftarrow_{\mathsf{R}} \mathbb{Z}_q^{d\times m}, \mathbf{u} \leftarrow_{\mathsf{R}} \mathbb{Z}_q^{d} : (\mathbf{A}, \mathbf{u})\right\}.$$

We will also make use of the following standard "smudging" lemma.

Lemma 2.2 (Smudging)**.** *For all $B, B' \in \mathbb{N}$ such that $B < B'$, all $x \in [-B, B]$, the statistical distance between the following distributions is upper bounded by $\frac{B}{B'}$:*

$$\left\{u \leftarrow_{\mathsf{R}} [-B', B'] : u\right\}$$
$$\left\{u \leftarrow_{\mathsf{R}} [-B', B'] : u + x\right\}.$$

2.2 Indistinguishability

We start by recalling the standard definition of computational indistinguishability [GM84].

Definition 2.2 (Computational indistinguishability)**.** *Two ensembles $\{\mathcal{D}_\lambda^0\}_{\lambda\in\mathbb{N}}$ and $\{\mathcal{D}_\lambda^1\}_{\lambda\in\mathbb{N}}$ are said to be* computationally indistinguishable w.r.t. a class $\mathcal{C}$ of attacker *if for every algorithm $A \in \mathcal{C}$, there exists a negligible function μ such that for every $\lambda \in \mathbb{N}$,*

$$\left|\Pr[A(1^\lambda, \mathcal{D}_\lambda^0) = 1] - \Pr[A(1^\lambda, \mathcal{D}_\lambda^1) = 1]\right| \leq \mu(\lambda)$$

We simply say that $\{\mathcal{D}_\lambda^0\}_{\lambda\in\mathbb{N}}$ and $\{\mathcal{D}_\lambda^1\}_{\lambda\in\mathbb{N}}$ are computationally indistinguishable *if they are computationally indistinguishable w.r.t. the class of non-uniform Probabilistic Polynomial Time (nuPPT) attackers.*

2.3 Definition of iO

We recall the definition of $i\mathcal{O}$ [BGI+01, GGH+13b]. Given polynomials $n(\cdot), s(\cdot), d(\cdot)$, let $\mathcal{C}_{n,s,d} = \{\mathcal{C}_\lambda\}_{\lambda\in\mathbb{N}}$ denote the class of circuits such that for all $\lambda \in \mathbb{N}$, $\mathcal{C}_\lambda$ is the set of circuits with input size $n(\lambda)$, size at most $s(\lambda)$ and depth at most $d(\lambda)$. We say that a sequence of circuits $\{\Pi_\lambda\}_{\lambda\in\mathbb{N}}$ is contained in $\{\mathcal{C}_\lambda\}_{\lambda\in\mathbb{N}}$ (denoted by $\{\Pi_\lambda\}_{\lambda\in\mathbb{N}} \in \{\mathcal{C}_\lambda\}_{\lambda\in\mathbb{N}}$) if for all $\lambda \in \mathbb{N}$, $\Pi_\lambda \in \mathcal{C}_\lambda$.

Definition 2.3 ($i\mathcal{O}$ for $\mathsf{P/poly}$)**.** *We say that $i\mathcal{O}$ exists for* $\mathsf{P/poly}$ *if for all polynomials $n(\cdot), s(\cdot), d(\cdot)$, there exists a tuple of PPT algorithms* $(\mathsf{Obf}, \mathsf{Eval})$ *such that the following holds:*

- **Correctness:** *For all $\{\Pi_\lambda\}_{\lambda\in\mathbb{N}} \in \mathcal{C}_{n,s,d}$, there exists a negligible function μ such that for all $\lambda \in \mathbb{N}$, all $\mathbf{x} \in \{0,1\}^{n(\lambda)}$,*

$$\Pr[\widetilde{\Pi} \leftarrow \mathsf{Obf}(1^\lambda, \Pi_\lambda) : \mathsf{Eval}(1^\lambda, \widetilde{\Pi}, \mathbf{x}) = \Pi(\mathbf{x})] \geq 1 - \mu(n)$$

- **IND-security:** *For all sequences $\{\Pi_0^\lambda\}_{\lambda\in\mathbb{N}}, \{\Pi_1^\lambda\}_{\lambda\in\mathbb{N}} \in \mathcal{C}_{n,s,d}$ such that for all $\lambda \in \mathbb{N}$, Π_0^λ and Π_1^λ are functionally equivalent circuits, the following ensembles are computationally indistinguishable:*

$$\left\{\widetilde{\Pi} \leftarrow \mathsf{Obf}(1^\lambda, \Pi_\lambda^0) : \widetilde{\Pi})\right\}_{\lambda\in\mathbb{N}}$$
$$\left\{\widetilde{\Pi} \leftarrow \mathsf{Obf}(1^\lambda, \Pi_\lambda^1) : \widetilde{\Pi})\right\}_{\lambda\in\mathbb{N}}$$

2.4 Definition of XiO

We recall the definition of $Xi\mathcal{O}$ with pre-processing [LPST16]. We restrict our attention to circuits with input length $O(\log \lambda)$: Given polynomials $n(\cdot), s(\cdot), d(\cdot)$, let $\mathcal{C}_{\log(n),s,d} = \{\mathcal{C}_\lambda\}_{\lambda \in \mathbb{N}}$ denote the class of circuits such that for all $\lambda \in \mathbb{N}$, $\mathcal{C}_\lambda$ is the set of circuits with input size $\log(n(\lambda))$, size at most $s(\lambda)$ and depth at most $d(\lambda)$.

Definition 2.4 ($Xi\mathcal{O}$ for $\mathsf{P}^{\log}/\mathsf{poly}$)**.** *We say $Xi\mathcal{O}$ exists for $\mathsf{P}^{\log}/\mathsf{poly}$ if there exists a polynomial $p(\cdot)$ and a constant ε, such that for all polynomials $n(\cdot), s(\cdot), d(\cdot)$, there exists a tuple of PPT algorithms* $(\mathsf{Gen}_{\mathsf{Obf}}, \mathsf{Obf}, \mathsf{Eval})$ *such that the following holds:*

- **Correctness:** *For all $\{\Pi_\lambda\}_{\lambda \in \mathbb{N}} \in \mathcal{C}_{\log(n),s,d}$, there exists a negligible function μ such that for all $\lambda \in \mathbb{N}$, all $\mathbf{x} \in \{0,1\}^{n(\lambda)}$,*

$$\Pr[\mathsf{pp} \leftarrow \mathsf{Gen}_{\mathsf{Obf}}(1^\lambda), \widetilde{\Pi} \leftarrow \mathsf{Obf}(\mathsf{pp}, \Pi_\lambda) : \mathsf{Eval}(\mathsf{pp}, \widetilde{\Pi}, \mathbf{x}) = \Pi(\mathbf{x})] \geq 1 - \mu(n)$$

- **Succinctness:** *For all $\{\Pi_\lambda\}_{\lambda \in \mathbb{N}} \in \mathcal{C}_{\log(n),s,d}$, all $\lambda \in \mathbb{N}$, all pp in the support of $\mathsf{Gen}_{\mathsf{Obf}}(1^\lambda)$, all $\widetilde{\Pi}$ in the support of $\mathsf{Obf}(\mathsf{pp}, \Pi_\lambda)$, we have that $\left|\widetilde{\Pi}\right| \leq n(\lambda)^{1-\varepsilon} \cdot p(\lambda, s(\lambda), d(\lambda))$*

- **IND-security:** *For all sequences $\{\Pi_0^\lambda\}_{\lambda \in \mathbb{N}}, \{\Pi_1^\lambda\}_{\lambda \in \mathbb{N}} \in \mathcal{C}_{\log(n),s,d}$ such that for all $\lambda \in \mathbb{N}$, Π_0^λ and Π_1^λ are functionally equivalent circuit, the following ensembles are computationally indistinguishable:*

$$\left\{\mathsf{pp} \leftarrow \mathsf{Gen}_{\mathsf{Obf}}(1^\lambda), \widetilde{\Pi} \leftarrow \mathsf{Obf}(\mathsf{pp}, \Pi^0_\lambda) : (\mathsf{pp}, \widetilde{\Pi})\right\}_{\lambda \in \mathbb{N}}$$
$$\left\{\mathsf{pp} \leftarrow \mathsf{Gen}_{\mathsf{Obf}}(1^\lambda), \widetilde{\Pi} \leftarrow \mathsf{Obf}(\mathsf{pp}, \Pi^1_\lambda) : (\mathsf{pp}, \widetilde{\Pi})\right\}_{\lambda \in \mathbb{N}}$$

The following theorem from [LPST16] connects $Xi\mathcal{O}$ (with pre-processing) with $i\mathcal{O}$ assuming the LWE assumption (we formally define the LWE assumption in Definition 3.4).

Theorem 2.5. *Assume the existence of a subexponentially secure $Xi\mathcal{O}$ for $\mathsf{P}^{\log}/\mathsf{poly}$, and assume subexponential security of the LWE assumption. Then there exists an $i\mathcal{O}$ for P/poly.*

2.5 Definition of Public-Key Encryption

We start by recalling the definition of public key encryption (PKE). For our purposes, we will consider PKE in a Common Reference String (CRS) model, where we first generate a CRS, and next, the key generation algorithm will take the CRS as input. This added generality will be useful to capture scenarios where multiple encryption schemes will be operating over the same field $\mathbb{Z}_N^*$—this field can be specified in the CRS.

Definition 2.6 (Public-Key Encryption)**.** *A Public-Key Encryption (PKE) scheme is a tuple of PPT algorithms* $(\mathsf{CRSgen}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *where:*

- $\mathsf{CRSgen}(1^\lambda)$*: given as input the security parameter $\lambda \in \mathbb{N}$, it outputs a common reference string* crs.

- $\mathsf{Gen}(\mathsf{crs})$*: given as input* crs*, it outputs the pair* $(\mathsf{pk}, \mathsf{sk})$.

- $\mathsf{Enc}_{\mathsf{pk}}(m; r)$*: given as input the public key* pk*, a message $m \in \{0,1\}^*$ and some randomness $r \leftarrow_{\mathsf{R}} \{0,1\}^{\infty}$*[5]*, it outputs a ciphertext* ct.

[5] As usual, since all algorithms are PPT we really only need to consider a finite prefix of $\{0,1\}^\infty$ to define the uniform distribution.

- $\mathsf{Dec}_{\mathsf{sk}}(\mathsf{ct})$*: given as input the secret key* sk *and a ciphertext* ct*, it deterministically outputs a plaintext.*

We furthermore require these algorithms to satisfy the following correctness *condition: for all* $\lambda \in \mathbb{N}$*, all* crs *in the support of* $\mathsf{CRSgen}(1^\lambda)$*, all pairs* $(\mathsf{pk}, \mathsf{sk})$ *in the support* $\mathsf{Gen}(\mathsf{crs})$*, all messages* $m \in \{0,1\}^*$*, all ciphertexts* ct *in the support of* $\mathsf{Enc}_{\mathsf{pk}}(m)$*, we have:*

$$\mathsf{Dec}_{\mathsf{sk}}(\mathsf{ct}) = m.$$

2.6 Definition of Linearly-Homomorphic Encryption

Definition 2.7 (Linearly-Homomorphic Encryption)**.** *For any polynomial* $\ell(\cdot)$*, a PKE scheme* $(\mathsf{CRSgen}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *is said to be a Linearly-Homomorphic Encryption (LHE) with plaintext size* $\ell(\cdot)$*, if there exists a PPT algorithm* Add *such that the following holds:*

- *For all* $\lambda \in \mathbb{N}$*, all* crs *in the support of* $\mathsf{CRSgen}(1^\lambda)$*, all* $(\mathsf{pk}, \mathsf{sk})$ *in the support of* $\mathsf{Gen}(\mathsf{crs})$*, the public key* pk *contains a message space* $(\mathbb{A}_{\mathsf{pk}}, +)$*, which is an Abelian group of size* $|\mathbb{A}| > 2^{\ell(\lambda)}$*.*

- *For all* $\lambda \in \mathbb{N}$*, all* crs *in the support of* $\mathsf{CRSgen}(1^\lambda)$*, all* $(\mathsf{pk}, \mathsf{sk})$ *in the support of* $\mathsf{Gen}(\mathsf{crs})$*, all messages* $m_1, m_2 \in \mathbb{A}_{\mathsf{pk}}$*, all ciphertexts* $\mathsf{ct}_1, \mathsf{ct}_2$ *in the support of* $\mathsf{Enc}_{\mathsf{pk}}(m_1), \mathsf{Enc}_{\mathsf{pk}}(m_2)$ *respectively, the algorithm* $\mathsf{Add}(\mathsf{pk}, \mathsf{ct}_1, \mathsf{ct}_2)$ *determinsitically outputs a ciphertext in the support of* $\mathsf{Enc}_{\mathsf{pk}}(m_1 + m_2)$*, where the addition is performed in* $\mathbb{A}_{\mathsf{pk}}$*.*

2.7 Definition of Fully Homomorphic Encryption

Definition 2.8 (Fully-Homomorphic Encryption)**.** *A PKE scheme* $(\mathsf{CRSgen}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *is said to be a Fully-Homomorphic Encryption (FHE) scheme if there exists a PPT algorithm* Eval *such that for all* $\lambda \in \mathbb{N}$*, all* crs *in the support of* $\mathsf{CRSgen}(1^\lambda)$*, all pairs* $(\mathsf{pk}, \mathsf{sk})$ *in the support of* Gen*, all* $n \in \mathbb{N}$*, all messages* $m_1, \ldots, m_n \in \{0,1\}$*, all ciphertexts* $\mathsf{ct}_1, \ldots, \mathsf{ct}_n$ *in the support of* $\mathsf{Enc}_{\mathsf{pk}}(m_1), \ldots, \mathsf{Enc}_{\mathsf{pk}}(m_n)$ *respectively, all circuits* $f : \{0,1\}^n \to \{0,1\}$*,* $\mathsf{Eval}(\mathsf{pk}, f, \mathsf{ct}_1, \ldots, \mathsf{ct}_n)$ *deterministically outputs an evaluated ciphertext* ct_f *such that* $\mathsf{Dec}_{\mathsf{sk}}(\mathsf{ct}_f) = f(m_1, \ldots, m_n)$*.*

Note that neither the arity nor the depth of the circuit that is homomorphically evaluated is a priori bounded (that is, we are considering unlevelled FHE). The FHE we will be using — namely, from [GSW13] — natively supports arithmetic circuits (with addition and multiplication gates), which capture Boolean circuits.

2.8 Leakage-resilient and Circular Security

We recall the standard notion of (indistinguishability-based) security for encryption schemes; we also consider a stronger form of $\mathcal{O}$-leakage resilient security, where the attacker also gets access to a leakage oracle $\mathcal{O}$ that has access to the message $m^\star$ being encrypted, and the randomness r under which it is encrypted.

Definition 2.9 (Multi-message security)**.** *We say that a public-key encryption scheme* $\mathcal{PKE} = (\mathsf{CRSgen}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *is* secure *if for all polynomials* $s(\cdot)$*, all sequences of pairs of messages* $\{m_\lambda^0, m_\lambda^1\}_{\lambda \in \mathbb{N}}$ *such that for all* $\lambda \in \mathbb{N}$*:* $|m_\lambda^0| = |m_\lambda^1| = s(\lambda)$*, the ensembles* $\{\mathcal{D}_\lambda^0\}_{\lambda \in \mathbb{N}}$ *and* $\{\mathcal{D}_\lambda^1\}_{\lambda \in \mathbb{N}}$ *are computationally indistinguishable, where* $\mathcal{D}_\lambda^b$ *is defined as follows:*

$$\left\{ \ \mathsf{crs} \leftarrow \mathsf{CRSgen}(1^\lambda), (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(\mathsf{crs}), m^\star = m_\lambda^b, r \leftarrow_{\mathsf{R}} \{0,1\}^\infty, \mathsf{ct} = \mathsf{Enc}_{\mathsf{pk}}(m^\star; r) : (\mathsf{crs}, \mathsf{pk}, \mathsf{ct}) \ \right\}$$

We additionally say that $\mathcal{PKE}$ *is* $\mathcal{O}$-leakage resilient secure *if indistinguishability holds w.r.t. all nuPPT distinguishers that get unbounded oracle access to* $\mathcal{O}(m^\star, r)$*, and never make the oracle output* $\perp$*.*

We proceed to defining a notion of 2-circular security w.r.t two encryption schemes $\mathcal{PKE}, \overline{\mathcal{PKE}}$. For our purposes, the key generation algorithm of $\mathcal{PKE}$ will be allowed to depend on the public key of $\overline{\mathcal{PKE}}$ (so they can operate over the same field). To enables this, we make use of the CRS: the CRS of $\mathcal{PKE}$ will be set to the public key for $\overline{\mathcal{PKE}}$. 2-circular security will next require indistinguishability of encryptions using $\mathcal{PKE}$ of any message m^0, m^1 in the presence of encrypted key cycle w.r.t. $\mathcal{PKE}$ and $\overline{\mathcal{PKE}}$. We furthermore generalize this definition to consider leakage-resilient security of $\mathcal{PKE}$.

Definition 2.10 (Circular security). *We say that* 2-circular security *holds w.r.t.* $\mathcal{PKE}, \overline{\mathcal{PKE}}$ *where* $\mathcal{PKE} = (\mathsf{CRSgen}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *and* $\overline{\mathcal{PKE}} = (\overline{\mathsf{CRSgen}}, \overline{\mathsf{Gen}}, \overline{\mathsf{Enc}}, \overline{\mathsf{Dec}})$ *if for all polynomials* $s(\cdot)$, *all sequences of pairs of messages* $\{m^0_\lambda, m^1_\lambda\}_{\lambda\in\mathbb{N}}$ *such that for all* $\lambda \in \mathbb{N}$: $|m^0_\lambda| = |m^1_\lambda| = s(\lambda)$, *the ensembles* $\{\mathcal{D}^0_\lambda\}_{\lambda\in\mathbb{N}}$ *and* $\{\mathcal{D}^1_\lambda\}_{\lambda\in\mathbb{N}}$ *are computationally indistinguishable, where* $\mathcal{D}^b_\lambda$ *is defined as follows:*

$$\left\{ \begin{array}{ll} \overline{\mathsf{crs}} \leftarrow \overline{\mathsf{CRSgen}}(1^\lambda), (\overline{\mathsf{pk}}, \overline{\mathsf{sk}}) \leftarrow \overline{\mathsf{Gen}}(\overline{\mathsf{crs}}), \mathsf{crs} = \overline{\mathsf{pk}}, (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(\mathsf{crs}) & \\ \overline{\mathsf{ct}} \leftarrow \overline{\mathsf{Enc}}_{\overline{\mathsf{pk}}}(\mathsf{sk}), m^\star = (\overline{\mathsf{sk}} \| m^b_\lambda), r \leftarrow_{\mathsf{R}} \{0,1\}^\infty, \mathsf{ct} = \mathsf{Enc}_{\mathsf{pk}}(m^\star; r) \quad : & (\mathsf{crs}, \overline{\mathsf{crs}}, \mathsf{pk}, \overline{\mathsf{pk}}, \mathsf{ct}, \overline{\mathsf{ct}}) \end{array} \right\}.$$

We additionally say that $\mathcal{O}$-leakage resilient 2-circular security holds w.r.t. $\mathcal{PKE}$ and $\overline{\mathcal{PKE}}$ *if indistinguishability holds w.r.t. all nuPPT distinguishers that get unbounded oracle access to* $\mathcal{O}(m^\star, r)$, *and never make the oracle output* $\perp$.

We are finally ready to state the $\mathsf{2CIRC}$ assumption that we will rely on.

Definition 2.11 (2CIRC assumption). *We say that the* $\mathsf{2CIRC}^{\mathcal{O}}$ assumption holds w.r.t. $\mathcal{PKE}$ and $\overline{\mathcal{PKE}}$ *if the following holds: if* $\mathcal{PKE}$ *is* $\mathcal{O}$*-leakage resilient secure, and* $\overline{\mathcal{PKE}}$ *is secure, then* $\mathcal{O}$*-leakage resilient 2-circular security holds w.r.t.* $\mathcal{PKE}$ *and* $\overline{\mathcal{PKE}}$.

We will also consider a *subexponential* $\mathsf{2CIRC}^{\mathcal{O}}$ assumption which is identically defined except it considers subexponential (as opposed to polynomial) security of the encryption schemes.

Definition 2.12 (Subexponential 2CIRC assumption). *We say that the* subexponential $\mathsf{2CIRC}^{\mathcal{O}}$ assumption holds w.r.t. $\mathcal{PKE}$ and $\overline{\mathcal{PKE}}$ *if the following holds: if* $\mathcal{PKE}$ *is subexponentially* $\mathcal{O}$*-leakage resilient secure, and* $\overline{\mathcal{PKE}}$ *is subexponentially secure, then subexponential* $\mathcal{O}$*-leakage resilient 2-circular security holds w.r.t.* $\mathcal{PKE}$ *and* $\overline{\mathcal{PKE}}$.

3 Shielded Randomness Leakage Security of GSW

In this section, we define our notion of Shielded Randomness Leakage (SRL) security, which corresponds to $\mathcal{O}$-leakage resilience security for a particular leakage oracle $\mathcal{O}$. Then, we prove the GSW FHE is SRL secure under the LWE assumption.

3.1 Definition of Shielded Randomness Leakage Security

To define our notion of SRL security, we focus on FHE schemes that satisfy the following properties.

3.1.1 Batch correctness

This property states that decryption of evaluated ciphertexts solely consists of computing the inner product of the evaluated ciphertext with the secret key (both of which are vectors), then rounding. Also, a single scalar obtained by decryption can encode many output bits of the evaluated function. That is, we consider FHE scheme where the crs contains a modulus N_{crs} such that decryption of an evaluated ciphertext yields a scalar in $\mathbb{Z}_N$. Our definition of FHE is flexible with respect to the choice of the modulus N, which we can afford since the LWE assumption holds for essentially any (large enough) modulus. As observed in [Mic19, BDGM19, BDGM20a], most existing FHE schemes can fit this framework.

Definition 3.1 (Batch correctness)**.** *For all poynomials $d(\cdot)$, an FHE scheme* $(\mathsf{CRSgen}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ *for depth-$d(\cdot)$ circuits satisfies batch correctness if there exist a PPT* Eval' *and a polynomial $\sigma(\cdot)$ such that following holds:*

- *For all $\lambda \in \mathbb{N}$, all* crs *in the support of* $\mathsf{CRSgen}(1^\lambda)$*, all* $(\mathsf{pk}, \mathsf{sk})$ *in the support of* $\mathsf{Gen}(\mathsf{crs})$*, we have:* pk *contains $B_{\mathsf{pk}} \in \mathbb{N}$ such that $N_{\mathsf{crs}} \geq 2^\lambda B_{\mathsf{pk}}$; the secret key is of the form:* $\mathsf{sk} \in \mathbb{Z}^{\sigma(\lambda)}$*.*

- *For all $\lambda \in \mathbb{N}$, all* crs *in the support of* $\mathsf{CRSgen}(1^\lambda)$*, all* $(\mathsf{pk}, \mathsf{sk})$ *in the support of* $\mathsf{Gen}(\mathsf{crs})$*, all arities $\nu \in \mathbb{N}$, all messages $m_1, \dots, m_\nu \in \{0,1\}$, all depth-$d(\lambda)$ circuits f of arity ν, all ciphertexts* ct_i *in the support of* $\mathsf{Enc}_{\mathsf{pk}}(m_i)$ *for all $i \in [\nu]$, all scaling factors $\omega < \log(N_{\mathsf{crs}})$, the algorithm* $\mathsf{Eval}'(\mathsf{pk}, f, \omega, \mathsf{ct}_1, \dots, \mathsf{ct}_\nu)$ *deterministically outputs an evaluated ciphertext* $\mathsf{ct}_f \in \mathbb{Z}_{N_{\mathsf{crs}}}^{\sigma(\lambda)}$ *such that:*
$$\mathsf{sk}^\top \mathsf{ct}_f = 2^\omega f(\mathbf{m}) + \mathsf{noise}_f \in \mathbb{Z}_{N_{\mathsf{crs}}},$$
with $|\mathsf{noise}_f| < B_{\mathsf{pk}}$.

Note that one can recover the value $f(\mathbf{m}) \in \mathbb{Z}_{N_{\mathsf{crs}}}$ when using the scaling factor $\omega = 2^\lambda$. That is, we can define $\mathsf{Eval}(\mathsf{pk}, f, \mathsf{ct}_1, \dots, \mathsf{ct}_\nu) = \mathsf{Eval}'(\mathsf{pk}, f, 2^\lambda, \mathsf{ct}_1, \dots, \mathsf{ct}_\nu)$.

3.1.2 Randomness homomorphism

This property states that it is possible to homomorphically evaluates a circuit f not only on the ciphertexts, but also the randomness used by the ciphertexts. The resulting evaluated randomness $\mathbf{r}_f$ belongs to a noisy randomness space $\mathcal{R}^\star$ — typically the fresh randomness comprises noises, and the evaluated randomness consists of larger-magnitude noises. The encryption algorithm $\mathsf{Enc}^\star$ is essentially the same as Enc except it operates on the evaluated (noisier) randomness. The ciphertext obtained by first evaluating the randomness, then using the noisy encryption algorithm $\mathsf{Enc}^\star$ is the same as obtained by directly evaluating the original ciphertexts.

Definition 3.2 (Randomness homomorphism)**.** *An FHE scheme* $\mathcal{FHE} = (\mathsf{CRSgen}, \mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ *for depth-$d(\cdot)$ circuits that satisfies batch correctness (defined above) also satisfies randomness homomorphism if there exists a sequence of noisy randomness spaces $\{\mathcal{R}^\star_\lambda\}_{\lambda \in \mathbb{N}}$, and the following additional PPT algorithms:*

- $\mathsf{Eval}_{\mathsf{rand}}(\mathsf{pk}, f, \mathbf{r}, \mathbf{m})$*: given as input the public key* pk*, a depth-$d(\lambda)$ circuit f of arity ν, random coins $\mathbf{r} = (r_1, \dots, r_\nu)$ where for all $i \in [\nu]$, $r_i \in \{0,1\}^\infty$, and messages $\mathbf{m} \in \{0,1\}^\nu$, it deterministically outputs an evaluated randomness $r_f \in \mathcal{R}^\star$.*

- $\mathsf{Enc}^\star_{\mathsf{pk}}(m; r^\star)$*: given as input the public key* pk*, a message $m \in \mathbb{Z}_{N_{\mathsf{crs}}}$ and the randomness $r^\star \in \mathcal{R}^\star$, it outputs a noisy ciphertext* $\mathsf{ct}^\star$*.*

We furthermore require these algorithms to satisfy the following condition: for every $\lambda \in \mathbb{N}$, all crs *in the support of* $\mathsf{CRSgen}(1^\lambda)$*, all pairs* $(\mathsf{pk}, \mathsf{sk})$ *in the support of* $\mathsf{Gen}(\mathsf{crs})$*, all $\nu \in \mathbb{N}$, all depth-$d(\lambda)$ circuits f of arity ν, all messages $m_i \in \{0,1\}$, all randomness $r_i \in \{0,1\}^\infty$ for $i \in [\nu]$, denoting* $\mathsf{ct}_i = \mathsf{Enc}_{\mathsf{pk}}(m_i; r_i)$ *and* $r_f = \mathsf{Eval}_{\mathsf{rand}}(\mathsf{pk}, f, \mathbf{r}, \mathbf{m})$*, we have:*
$$\mathsf{Eval}'(\mathsf{pk}, f, 0, \mathsf{ct}_1, \dots, \mathsf{ct}_\nu) = \mathsf{Enc}^\star_{\mathsf{pk}}(f(\mathbf{m}); r_f).$$

3.1.3 Shielded Randomness-Leakage security

To define the following Shielded Randomness-Leakage oracle oracle, we restrict ourselves to FHE where the noisy randomness consists of integer vectors. That is, there exists a polynomial $t(\cdot)$ such that the sequence $\{\mathcal{R}^\star_\lambda\}_{\lambda\in\mathbb{N}}$ is such that for all $\lambda \in \mathbb{N}$, $\mathcal{R}^\star_\lambda \subseteq \mathbb{Z}^{t(\lambda)}$. Henceforth, we denote by $\mathbf{r}_1 + \mathbf{r}_2 \in \mathcal{R}^\star_\lambda$ and $\mathbf{r}_1 - \mathbf{r}_2 \in \mathcal{R}^\star_\lambda$ the addition and subtraction in $\mathbb{Z}^{t(\lambda)}$. We denote $\mathcal{R}^\star_\lambda$ by $\mathcal{R}^\star$ for simplicity.

Definition 3.3 (SRL security)**.** *An FHE scheme $\mathcal{FHE}$ for depth $d(\cdot)$ circuits satisfying randomness homomorphism is said to be SRL-secure if it is $\mathcal{O}^{\mathcal{FHE}}_{\mathsf{SRL}}$-leakage resilient secure for the following oracle $\mathcal{O}^{\mathcal{FHE}}_{\mathsf{SRL}}$, where $\mathsf{Eval}_{\mathsf{rand}}$ and $\mathsf{Enc}^\star$ are the algorithms guaranteed to exist by the definition of randomness homomorphism. Similarly, for any PKE scheme $\mathcal{PKE}$, we say 2-circular SRL security holds with respect to $\mathcal{FHE}$ and $\mathcal{PKE}$ if the $\mathcal{O}^{\mathcal{FHE}}_{\mathsf{SRL}}$-leakage resilient 2-circular security holds with respect to $\mathcal{FHE}$ and $\mathcal{PKE}$.*

```
$\underline{\mathcal{O}^{\mathcal{FHE}}_{\mathsf{SRL}}(\mathbf{m}^\star, \mathbf{r})}$:
$\mathbf{r}^\star \leftarrow_{\mathsf{R}} \mathcal{R}^\star$, $\mathsf{ct}^\star = \mathsf{Enc}^\star_{\mathsf{pk}}(0; \mathbf{r}^\star)$
$(f, \alpha) \leftarrow \mathcal{A}(\mathsf{ct}^\star)$
$\mathbf{r}_f = \mathsf{Eval}_{\mathsf{rand}}(\mathsf{pk}, f, \mathbf{r}, \mathbf{m}^\star)$.
If $f(\mathbf{m}^\star) = \alpha$ and $f$ is of depth at most $d$, then $\mathsf{leak} = \mathbf{r}^\star - \mathbf{r}_f \in \mathcal{R}^\star$.
Otherwise, $\mathsf{leak} = \perp$. Return $\mathsf{leak}$.
```

Roughly speaking, given a message $\mathbf{m}^\star$ and randomness $\mathbf{r}$, the oracle $\mathcal{O}^{\mathcal{FHE}}_{\mathsf{SRL}}$ samples fresh random coins $\mathbf{r}^\star$ from which it generates a noisy encryption of zero, that is sent to the adversary. The adcersary next chooses a circuit f and a value $\alpha \in \mathbb{Z}$. The oracle then checks that $f(\mathbf{m}^\star) = \alpha$, upon which it returns the evaluated randomness "shielded" with the randomness $\mathbf{r}^\star$.

In the concrete FHE we consider from [GSW13], the randomness leakage corresponds to the randomness obtained from homomorphically subtracting the encryption $\mathsf{Enc}^\star_{\mathsf{pk}}(0; \mathbf{r}^\star)$ by the evaluated challenge ciphertext. Revealing such leakage allows the adversary to decrypt and recover the value $0 - f(\mathbf{m}^\star)$. By enforcing $f(\mathbf{m}^\star) = \alpha$, we make sure the adversary does not learn anything more than what she already knew (recalling that the notion of $\mathcal{O}$-leakage resilient security only requires indistinguishability w.r.t. adversarys that do not make the oracle output $\perp$).

Whenever the scheme $\mathcal{FHE}$ is clear from context, we simply write $\mathcal{O}_{\mathsf{SRL}}$ to denote $\mathcal{O}^{\mathcal{FHE}}_{\mathsf{SRL}}$.

3.2 SRL Security of the GSW FHE from LWE

We now recall the FHE scheme from [GSW13], whose security relies on the LWE assumption. The variant we present uses a large modulus to permit batching many output bits in a single scalar. We prove the GSW scheme is SRL-secure (as per Definition 3.3) under the LWE assumption.

3.2.1 Learning With Error Assumption

We recall the Learning with Error (LWE) assumption [Reg05] with subexponential modulus-to-noise ratio.

Definition 3.4 (Learning With Error Assumption)**.** *For all polynomials $n(\cdot), B(\cdot)$, all sequences $q = \{q_\lambda\}_{\lambda\in\mathbb{N}}$ such that for every $\lambda \in \mathbb{N}$ the bit size $|q_\lambda|$ is polynomially bounded in λ, all B-bounded ensemble of efficiently sampleable distributions $\chi = \{\chi_\lambda\}_{\lambda\in\mathbb{N}}$, we say LWE holds with respect to q, n, χ*

if for all polynomials $m(\cdot)$, the following ensembles are computationally indistinguishable:

$$\left\{\mathbf{A} \leftarrow_{\mathsf{R}} \mathbb{Z}_{q_\lambda}^{m(\lambda)\times\kappa(\lambda)}, \mathbf{s} \leftarrow \chi_\lambda^{\kappa(\lambda)}, \mathbf{e} \leftarrow \chi_\lambda^{m(\lambda)}, \mathbf{z} = \mathbf{A}\mathbf{s} + \mathbf{e} \in \mathbb{Z}_{q_\lambda}^{m(\lambda)} : (\mathbf{A}, \mathbf{z})\right\}_{\lambda\in\mathbb{N}}.$$
$$\left\{\mathbf{A} \leftarrow_{\mathsf{R}} \mathbb{Z}_{q_\lambda}^{m(\lambda)\times\kappa(\lambda)}, \mathbf{z} \leftarrow_{\mathsf{R}} \mathbb{Z}_{q_\lambda}^{m(\lambda)} : (\mathbf{A}, \mathbf{z})\right\}_{\lambda\in\mathbb{N}}.$$

In [Reg05], Regev showed that solving the LWE problem with modulus q, dimension κ, arbitrary number of samples m, and discrete Gaussian distribution χ of standard deviation $\sigma = \alpha q \geq 2\sqrt{\kappa}$ (this is the distribution over $\mathbb{Z}$ that follows the normal distribution of standard deviation σ, which is $\sigma \cdot \omega(\sqrt{\log(\lambda)})$-bounded) is at least as hard as quantumly approximating the shortest independent vector problem (SIVP) to within an approximation factor $\gamma = \widetilde{\mathcal{O}}(\kappa/\alpha)$ in the *worst case* κ-dimensional lattices. His result only applied to every modulus q that is a prime power, or a product of small (poly-size) distinct primes. Later, in [PRS17], the result was generalized to any modulus q.

As typical, we choose a noise-to-modulus ratio $\alpha = 2^{-\kappa^c}$ for a constant $c \in (0, 1)$, which corresponds to the SIVP problem with an approximation factor $\gamma = \widetilde{\mathcal{O}}(\kappa \cdot 2^{\kappa^c})$, which is believed to be intractable for c small enough. That is, we rely on the following theorem.

Theorem 3.5 ([Reg05, PRS17])**.** *There exists a constant $c \in (0, 1)$ such that for all polynomials $n(\cdot), B(\cdot)$ and sequences $q = \{q_\lambda\}_{\lambda\in\mathbb{N}}$ where for all $\lambda \in \mathbb{N}$, $|q_\lambda|$ is polynomially bounded in λ, such that for all $\lambda \in \mathbb{N}$, the following holds:*

- $B(\lambda) \geq 2\sqrt{\kappa(\lambda)\log(\lambda)}$
- $B(\lambda) \geq q_\lambda 2^{-\kappa(\lambda)^c}$

there exists a B-bounded ensemble χ of efficiently sampleable distributions over $\mathbb{Z}$, such that LWE holds with respect to q, n, χ.

3.2.2 The GSW scheme

We present the FHE from [GSW13]. It is parameterized by a polynomial d that bounds the depth of the circuits that can be homomorphically evaluated. We denote the scheme by GSW_d.

The modulus N used by the scheme is described in the crs. Apart from being sufficiently large to ensure correctness (that is, larger than the noise magnitude obtained when evaluating circuits of depth at most d), no property is required from the modulus.

The reason we take the modulus N from an independently generated crs, instead of having the key generation algorithm generates the modulus itself, is to ensure compatibility of the GSW FHE with another Linearly Homomorphic Encryption scheme, that performs linearly operation over $\mathbb{Z}_N$. This way, both schemes can operate on the same ring $\mathbb{Z}_N$.

• $\underline{\mathsf{Gen}(\mathsf{crs})}$:
Given as input a crs that contains a modulus $N \in \mathbb{N}$ of θ bits for a sufficiently large θ, it chooses $\kappa, B' \in \mathsf{poly}(\lambda)$ and an efficiently sampleable B'-bounded distribution χ over $\mathbb{Z}$ such that LWE holds with respect to N, κ, χ. It sets $w = (\kappa+1)\lceil\log(N)\rceil$, $m = 2(\kappa+1)\lceil\log(N)\rceil+2\lambda$, $\widetilde{B} = (w+1)^d\lceil\log(N)\rceil$ and $B = \widetilde{B}B'm$. We also require that the modulus N is such that $N \geq 2^{2\lambda}B$.

It samples $\mathbf{A} \leftarrow_{\mathsf{R}} \mathbb{Z}_N^{\kappa\times m}$, $\mathbf{s} \leftarrow \chi^\kappa$, $\mathbf{e} \leftarrow \chi^m$, $\mathbf{g} = (1, 2, \ldots, 2^{\lceil\log(N)\rceil-1}) \in \mathbb{Z}_N^{\lceil\log(N)\rceil}$, $\mathbf{G} = \mathrm{Id} \otimes \mathbf{g}^\top \in \mathbb{Z}_N^{(\kappa+1)\times w}$ where $\mathrm{Id} \in \mathbb{Z}_N^{(\kappa+1)\times(\kappa+1)}$ denotes the identity matrix, $\mathbf{U} = \begin{pmatrix}\mathbf{A} \\ \mathbf{s}^\top\mathbf{A} + \mathbf{e}^\top\end{pmatrix} \in \mathbb{Z}_N^{(\kappa+1)\times m}$. It sets $\mathsf{pk} = (B, \mathbf{U}, \mathbf{G})$, and $\mathsf{sk} = (-\mathbf{s}, 1) \otimes \mathbf{g} \in \mathbb{Z}_N^w$. The parameters define the noisy randomness space

$\mathcal{R}^\star = [-2^\lambda \widetilde{B}, 2^\lambda \widetilde{B}]^m$. It outputs $(\mathsf{pk}, \mathsf{sk})$.

• $\underline{\mathsf{Enc}(\mathsf{pk}, m)}$:
Given the public pk, a message $m \in \{0,1\}$, it samples the randomness $\mathbf{R} \leftarrow_{\mathsf{R}} [-1,1]^{m \times w}$ and outputs the ciphertext $\mathsf{ct} = \mathbf{U}\mathbf{R} + m\mathbf{G} \in \mathbb{Z}_N^{(\kappa+1)\times w}$. For any $\mathbf{m} \in \{0,1\}^n$, we denote by $\mathsf{Enc}_{\mathsf{pk}}(\mathbf{m}; \mathbf{r})$ the concatenation of the encryptions $\mathsf{Enc}_{\mathsf{pk}}(m_1; \mathbf{R}_1), \dots, \mathsf{Enc}_{\mathsf{pk}}(m_n; \mathbf{R}_n)$.

• $\underline{\mathsf{Eval}(\mathsf{pk}, f, \mathsf{ct}_1, \dots, \mathsf{ct}_\nu)}$:
Given the public key pk, a depth-$d(\lambda)$ arithmetic $f : \{0,1\}^\nu \to \{0,1\}$, ciphertexts $\mathsf{ct}_1, \dots, \mathsf{ct}_\nu$, it runs $\mathsf{ct}_f \leftarrow \mathsf{Eval}'(\mathsf{pk}, f, \omega, \mathsf{ct}_1, \dots, \mathsf{ct}_\nu)$ with scaling factor $\omega = 2B$, where the algorithm Eval' is described below, for the batch correctness property.

• $\underline{\mathsf{Dec}_{\mathsf{sk}}(\mathsf{ct})}$: The decryption computes the inner product of $\mathsf{ct} \in \{0,1\}^w$ and $\mathsf{sk} \in \mathbb{Z}_N^w$ in $\mathbb{Z}_N$, and rounds the result.

We demonstrate that the GSW FHE satisfies the batch correctness property.

Proposition 1 (Batch correctness)**.** *The GSW FHE described above for depth-$d(\cdot)$ circuits satisfies batch correctness, as per Definition 3.1.*

Proof: We present the following PPT algorithm:

• $\underline{\mathsf{Eval}'(\mathsf{pk}, f, \omega, \mathsf{ct}_1, \dots, \mathsf{ct}_\nu)}$:
Given the public pk, a depth-$d(\lambda)$ arithmetic circuit $f : \{0,1\}^\nu \to \mathbb{Z}_N$, a scaling factor $\omega < \log(N)$, ciphertexts $\mathsf{ct}_1, \dots, \mathsf{ct}_\nu$, it evaluates the circuit gate by gate as follows.

- Addition gate between ct_i and ct_j: return $\mathsf{ct}_i + \mathsf{ct}_j \in \mathbb{Z}_N^{(\kappa+1)\times w}$.
- Multiplication gate between ct_i and ct_j: return $\mathsf{ct}_i \cdot \mathsf{BD}(\mathsf{ct}_j) \in \mathbb{Z}_N^{(\kappa+1)\times w}$, where $\mathsf{BD}(\mathsf{ct}_j) \in \{0,1\}^{w \times w}$ denotes the binary decomposition of $\mathsf{ct}_j \in \mathbb{Z}_N^{(\kappa+1)\times w}$.

By recursively applying the above operations, one can turn the ciphertexts $\mathsf{ct}_1, \dots, \mathsf{ct}_\nu$ into $\mathbf{C}_f^i = \mathbf{U}\mathbf{R}_f^i + f_i(\mathbf{m})\mathbf{G} \in \mathbb{Z}_N^{(\kappa+1)\times w}$, where $f_i(\mathbf{m}) \in \{0,1\}$ denotes the i'th bit of the binary decomposition of $f(\mathbf{m}) \in \mathbb{Z}_N$, that is, $f(\mathbf{m}) = \sum_{i=0}^{\lceil \log(N) \rceil - 1} 2^i f_i(\mathbf{m})$. For all $i \in [0, \lceil \log(N) \rceil - 1]$, we have $\|\mathbf{R}_f^i\|_\infty \leq (w+1)^d$. By definition of the matrix $\mathbf{G}$, choosing the $\kappa \cdot \lceil \log(N) \rceil + i + \omega + 1$'th column of $\mathbf{C}_f^i$ yields:

$$\mathbf{c}_f^i = \Big(\mathbf{A}\mathbf{r}_f^i, (\mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top)\mathbf{r}_f^i + 2^{\omega+i} f_i(\mathbf{m}) \Big) \in \mathbb{Z}_N^{\kappa+1}.$$

Summing up for all $i \in [0, \lceil \log(N) \rceil - 1]$, we get: $\mathsf{ct}'_f = \big(\mathbf{A}\mathbf{r}_f, (\mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top)\mathbf{r}_f + 2^\omega f(\mathbf{m}) \big) \in \mathbb{Z}_N^{\kappa+1}$, where $\mathbf{r}_f = \sum_{i=0}^{\lceil \log(N) \rceil - 1} \mathbf{r}_f^i$ of norm $\|\mathbf{r}_f\|_\infty \leq (w+1)^d \lceil \log(N) \rceil = \widetilde{B}$. It outputs the evaluated ciphertext $\mathsf{ct}_f = \mathsf{BD}\big(\mathsf{ct}'_f\big) \in \{0,1\}^w$.

The evaluated ciphertext $\mathsf{ct}_f \in \{0,1\}^w$ is such that:

$$\mathsf{sk}^\top \mathsf{ct}_f = -\mathbf{s}^\top \mathbf{A}\mathbf{r}_f + (\mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top)\mathbf{r}_f + 2^\omega f(\mathbf{m}) = 2^\omega f(\mathbf{m}) + \mathsf{noise}_f \in \mathbb{Z}_N,$$

where $\mathsf{noise}_f = \mathbf{e}^\top \mathbf{r}_f$. We have $|\mathsf{noise}_f| < \widetilde{B}B'm = B$. $\square$

We turn to proving that it also satisfies the randomness homomorphism property.

Proposition 2 (Randomness homomorphism)**.** *The GSW FHE for depth-$d(\cdot)$ circuits, satisfying bath correctness presented above satisfies the randomness homomorphism property as per Definition 3.2.*

Proof: We present the following PPT algorithms:

- $\underline{\mathsf{Enc}^\star(\mathsf{pk}, m; \mathbf{r}^\star)}$:

Given the public pk, a message $m \in \mathbb{Z}$, the randomness $\mathbf{r}^\star \in [-2^{\lambda/2}\widetilde{B}, 2^{\lambda/2}\widetilde{B}]^m$, it computes $\mathsf{ct}' = \big(\mathbf{A}\mathbf{r}^\star, (\mathbf{s}^\top\mathbf{A} + \mathbf{e}^\top)\mathbf{r}^\star + m\big) \in \mathbb{Z}_N^{\kappa+1}$, and outputs $\mathsf{ct} = \mathsf{BD}(\mathsf{ct}') \in \{0,1\}^w$.

- $\underline{\mathsf{Eval}_{\mathsf{rand}}(\mathsf{pk}, f, (\mathbf{R}_i)_{i\in[\nu]}, (m_i)_{i\in[\nu]})}$:

This algorithm is similar to the ciphertext evaluation algorithm. Namely, given the public pk, a depth-$d(\lambda)$ arithmetic circuit $f : \{0,1\}^\nu \to \mathbb{Z}_N$, randomness $\mathbf{R}_1, \ldots, \mathbf{R}_\nu \in [-1,1]^{m\times w}$, it evaluates the circuit gate by gate as follows.

- Addition gate between $\mathbf{R}_i$ and $\mathbf{R}_j$: return $\mathbf{R}_i + \mathbf{R}_j \in \mathbb{Z}_N^{m\times w}$.
- Multiplication gate between $\mathbf{R}_i$ and $\mathbf{R}_j$: compute $\mathsf{ct}_j = \mathsf{Enc}_{\mathsf{pk}}(m_j; \mathbf{R}_j)$, return $\mathbf{R}_i\mathsf{BD}(\mathsf{ct}_j) + m_i\mathbf{R}_j \in \mathbb{Z}_N^{m\times w}$, where $\mathsf{BD}(\mathsf{ct}_j) \in \{0,1\}^{w\times w}$ denotes the binary decomposition of $\mathsf{ct}_j \in \mathbb{Z}_N^{(\kappa+1)\times w}$.

By recursively applying the above operations, one can turn the randomness $\mathbf{R}_1, \ldots, \mathbf{R}_n$ into $\mathbf{R}_f^i \in \mathbb{Z}_N^{m\times w}$ such that: $\mathbf{C}_f^i = \mathbf{U}\mathbf{R}_f^i + f_i(\mathbf{m})\mathbf{G} \in \mathbb{Z}_N^{(\kappa+1)\times w}$, for all $i \in [0, \lceil\log(N)\rceil - 1]$; and $\|\mathbf{R}_f^i\|_\infty \leq (w+1)^d$. By definition of the matrix $\mathbf{G}$, choosing the $\kappa\lceil\log(N)\rceil+i+1$'th column of $\mathbf{R}_f^i$ yields $\mathbf{r}_f^i \in \mathbb{Z}_N^m$ such that: $\mathbf{c}_f^i = \Big(\mathbf{A}\mathbf{r}_f^i, (\mathbf{s}^\top\mathbf{A} + \mathbf{e}^\top)\mathbf{r}_f^i + 2^i f_i(\mathbf{m})\Big) \in \mathbb{Z}_N^{\kappa+1}$, and $\|\mathbf{r}_f^i\|_\infty \leq (w+1)^d$. Summing up for all $i \in [0, \lceil\log(N)\rceil - 1]$, we get: $\mathbf{r}_f = \sum_{i=0}^{\lceil\log(N)\rceil-1} \mathbf{r}_f^i \in \mathcal{R}^\star$ such that $\mathsf{ct}'_f = \big(\mathbf{A}\mathbf{r}_f, (\mathbf{s}^\top\mathbf{A} + \mathbf{e}^\top)\mathbf{r}_f + f(\mathbf{m})\big) \in \mathbb{Z}_N^{\kappa+1}$. It outputs the evaluated randomness $\mathbf{r}_f$. $\square$

3.2.3 SRL Security

Before proving the SRL security of GSW under the LWE assumption, we describe new trapdoor generation and pre-image sampling algorithms that are inspired by those from [MP12]. As in prior works, the trapdoor generation algorithm generates a matrix $\mathbf{U} \in \mathbb{Z}_N^{d\times m}$ that is statistically close to uniformly random over $\mathbb{Z}_N^{d\times m}$, together with an associated trapdoor $T_{\mathbf{U}}$. The pre-image sampling algorithm, given a target vector $\mathbf{t} \in \mathbb{Z}_N^d$, produces a short pre-image, that is, a short vector $\mathbf{r} \in \mathbb{Z}_N^m$ such that $\mathbf{U}\mathbf{r} = \mathbf{t}$. In these works, the distribution of these short pre-images is independent of the trapdoor — typically they follow a discrete (spherical) Gaussian distribution. Our requirements are slightly different: a pre-image produced by our sampling algorithm when given as input a target vector $\mathbf{t} \in \mathbb{Z}_N^d$ should be statistically close to a pre-image produced by our sampling algorithm when given as input the vector $\mathbf{0} \in \mathbb{Z}_N^d$, shifted by a much smaller pre-image of $\mathbf{t}$. That is, if a very short pre-image is given, adding a somewhat short pre-image of $\mathbf{0}$ (produced by the sampling algorithm) to it will produce a pre-image that looks like a fresh output of the sampling algorithm on input $\mathbf{t}$. This inherently requires smudging size noises, which implies the use of an exponential-size modulus q. In fact this property is not known to hold for existing trapdoor generation and pre-image sampling algorithms using polynomial-size modulus.

We prove this property for the concrete algorithms provided in [MP12], which we simplify since we can afford to use smudging-size noises. We provide a self-contained description of the scheme and its proofs here.

Lattice trapdoors.

- $\underline{\mathsf{TrapGen}(1^\lambda, N, d)}$:

Given as input the security parameter $\lambda \in \mathbb{N}$, a modulus $q \in \mathbb{N}$, a dimension $d \in \mathbb{N}$, it sets $\widetilde{m} = d\lceil\log(N)\rceil + 2\lambda$, $w = d\lceil\log(N)\rceil$, $m = \widetilde{m} + w$, computes the gadget matrix $\mathbf{G} = \mathrm{Id} \otimes \mathbf{g}^\top$ where $\mathrm{Id} \in \mathbb{Z}_N^{d\times d}$ denotes the identity matrix and $\mathbf{g} = (1, 2, \ldots, 2^{\lceil\log(N)\rceil-1}) \in \mathbb{Z}_N^{\lceil\log(N)\rceil}$, $\widetilde{\mathbf{U}} \leftarrow_{\mathsf{R}} \mathbb{Z}_N^{d\times\widetilde{m}}$, $\mathbf{R} \leftarrow_{\mathsf{R}} [-1,1]^{\widetilde{m}\times w}$, $\mathbf{U} = (\widetilde{\mathbf{U}} \| -\widetilde{\mathbf{U}}\mathbf{R} + \mathbf{G}) \in \mathbb{Z}_N^{d\times m}$, $T_{\mathbf{U}} = \mathbf{R}$. It outputs $(\mathbf{U}, T_{\mathbf{U}})$.

- $\underline{\mathsf{PreImSamp}(\mathbf{U}, T_{\mathbf{U}}, \mathbf{t}, B)}$:

Given as input the matrix $\mathbf{U}$, the trapdoor $T_{\mathbf{U}}$, a target vector $\mathbf{t} \in \mathbb{Z}_N^d$ and a bound $B \in \mathbb{N}$, it samples $\mathbf{v} \leftarrow_{\mathsf{R}} [-B2^{\lambda/2}, B2^{\lambda/2}]^m$, sets $\mathbf{b} = \mathsf{BD}(\mathbf{U}\mathbf{v} + \mathbf{t}) \in \{0,1\}^{w\times w}$ which denotes the binary decomposition of $\mathbf{U}\mathbf{v} + \mathbf{t} \in \mathbb{Z}_N^d$. It outputs $\begin{pmatrix}\mathbf{R}\mathbf{b}\\ \mathbf{b}\end{pmatrix} - \mathbf{v} \in \mathbb{Z}_N^m$.

We show the following properties hold.

Proposition 3 (Correctness of $\mathsf{TrapGen}$)**.** *For all λ, N, d, writing $m = 2d\lceil\log(N)\rceil + 2\lambda$, the following distributions have statistical distance at most $2^{-\lambda}$:*

$$\left\{\mathbf{U} \leftarrow_{\mathsf{R}} \mathbb{Z}_N^{d\times m} : \mathbf{U}\right\}$$
$$\left\{(\mathbf{U}, T_{\mathbf{U}}) \leftarrow \mathsf{TrapGen}(1^\lambda, N, d) : \mathbf{U}\right\}.$$

Proof: The proposition follows readily from Lemma 2.1 (left over hash lemma). □

Proposition 4 (Correctness of $\mathsf{PreImSamp}$)**.** *For all $\lambda, q, d, B \in \mathbb{N}$, all $(\mathbf{U}, T_{\mathbf{U}})$ in the support of $\mathsf{TrapGen}(1^\lambda, q, d)$, all $\mathbf{t} \in \mathbb{Z}_N^d$, all $\mathbf{r} \in \mathbb{Z}_N^m$ in the support of $\mathsf{PreImSamp}(\mathbf{U}, T_{\mathbf{U}}, \mathbf{t}, B)$ are such $\mathbf{U}\mathbf{r} = \mathbf{t}$ and $\|\mathbf{r}\|_\infty < B2^{\lambda/2} + w$.*

Proof: Straightforward. □

Proposition 5 (Security)**.** *For all $\lambda, q, d, B \in \mathbb{N}$, writing $m = 2d\lceil\log(N)\rceil + 2\lambda$, for all $\mathbf{w} \in \mathbb{Z}_N^m$ such that $\|\mathbf{w}\|_\infty < B$, the the statistical distance of the two following distributions is upper-bounded by $2^{-\lambda/2}$:*

$$\{(\mathbf{U}, T_{\mathbf{U}}) \leftarrow \mathsf{TrapGen}(1^\lambda, q, d), \widetilde{\mathbf{r}_0} \leftarrow_{\mathsf{R}} \mathsf{PreImSamp}(\mathbf{U}, T_{\mathbf{U}}, \mathbf{0}, B) : \widetilde{\mathbf{r}_0} + \mathbf{w} \in \mathbb{Z}_N^m\}$$
$$\{(\mathbf{U}, T_{\mathbf{U}}) \leftarrow \mathsf{TrapGen}(1^\lambda, q, d), \widetilde{\mathbf{r}} \leftarrow \mathsf{PreImSamp}(\mathbf{U}, T_{\mathbf{U}}, \mathbf{U}\mathbf{w}, B) : \widetilde{\mathbf{r}}\}$$

Proof: By definition of $\mathsf{PreImSamp}$ we have: $\widetilde{\mathbf{r}_0} = \begin{pmatrix}\mathbf{R}\mathbf{b}\\ \mathbf{b}\end{pmatrix} - \mathbf{v}$ where $\mathbf{v} \leftarrow_{\mathsf{R}} [-B2^{\lambda/2}, B2^{\lambda/2}]^m$ and $\mathbf{b} = \mathsf{BD}(\mathbf{U}\mathbf{v}) \in \{0,1\}^w$. Since $\|\mathbf{w}\|_\infty < B$, by Lemma 2.2 (smudging), we have: $\mathbf{v} \approx_s \mathbf{v} + \mathbf{w}$ with statistical distance $2^{-\lambda/2}$. This implies that $\widetilde{\mathbf{r}_0} + \mathbf{w} \approx_s \begin{pmatrix}\mathbf{R}\mathbf{b}'\\ \mathbf{b}'\end{pmatrix} - \mathbf{v}$, where $\mathbf{b}' = \mathsf{BD}(\mathbf{U}\mathbf{v} + \mathbf{U}\mathbf{w})$. The latter is identically distributed to $\mathsf{PreImSamp}(\mathbf{U}, T_{\mathbf{U}}, \mathbf{U}\mathbf{w}, B)$. □

We also prove the following lemma, that will prove useful later.

Lemma 3.1. *For all $\lambda, q, d \in \mathbb{N}$, the following distributions have statistical distance upper-bounded by $2^{-\lambda}$:*

$$\left\{(\mathbf{U}, T_{\mathbf{U}}) \leftarrow_{\mathsf{R}} \mathsf{TrapGen}(1^\lambda, q, d), \mathbf{r} \leftarrow_{\mathsf{R}} [-1,1]^m : (\mathbf{U}, T_{\mathbf{U}}, \mathbf{U}\mathbf{r})\right\}.$$
$$\left\{(\mathbf{U}, T_{\mathbf{U}}) \leftarrow_{\mathsf{R}} \mathsf{TrapGen}(1^\lambda, q, d), \mathbf{v} \leftarrow_{\mathsf{R}} \mathbb{Z}_N^m : (\mathbf{U}, T_{\mathbf{U}}, \mathbf{v})\right\}.$$

Proof: For all $\lambda, q, d \in \mathbb{N}$, there exists an inefficient probabilistic algorithm $\mathcal{A}$ such that the following distributions are identical: $\{(\mathbf{U}, T_{\mathbf{U}}) \leftarrow \mathsf{TrapGen}(1^\lambda, q, d) : (\mathbf{U}, T_{\mathbf{U}})\}$ and $\{(\mathbf{U}, T_{\mathbf{U}}) \leftarrow \mathsf{TrapGen}(1^\lambda, q, d) : (\mathbf{U}, \mathcal{A}(\mathbf{U}))\}$. For instance, the algorithm $\mathcal{A}$ enumerates all matrices $\mathbf{R} \in [-1, 1]^{\widetilde{m} \times w}$, then chooses one uniformly at random among those which match $\mathbf{U}$. Thus, Proposition 3 implies the lemma. $\square$

Theorem 3.6 (SRL security)**.** *For any polynomials $d(\cdot)$, the GSW FHE for depth-d circuits with batch correctness presented above is SRL-secure, under the LWE assumption.*

Proof: We proceed via a hybrid argument using the following ensembles for all $b \in \{0, 1\}$.

- $\underline{\mathcal{D}_\lambda^b}$: is the ensemble given in Definition 3.3.

- $\underline{\mathcal{H}_\lambda^{b.1}}$: is as $\mathcal{D}_\lambda^b$ except the LWE sample $\mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top$ from the public key is switched to a uniformly random vector using the LWE assumption. That is, the public key is computed as follows: $\mathbf{A} \leftarrow_{\mathsf{R}} \mathbb{Z}_N^{\kappa \times m}$, $\mathbf{v} \leftarrow_{\mathsf{R}} \mathbb{Z}_N^m$, $\mathbf{U} = \begin{pmatrix} \mathbf{A} \\ \mathbf{v}^\top \end{pmatrix}$; the gadget matrix $\mathbf{G}$ is computed as in $\mathcal{D}_\lambda^b$, and $\mathsf{pk} = (B, \mathbf{U}, \mathbf{G})$. The secret key is also computed as in $\mathcal{D}_\lambda^b$ (but now it is uncorrelated with pk), namely: $\mathbf{s} \leftarrow_{\mathsf{R}} \chi^\kappa$, $\mathsf{sk} = (-\mathbf{s}, 1) \otimes \mathbf{g} \in \mathbb{Z}_N^w$. The challenge ciphertext is computed as $\mathsf{ct} = \mathsf{Enc}_{\mathsf{pk}}(\mathbf{m}^\star; \mathbf{r})$ and the oracle $\mathcal{O}_{\mathsf{SRL}}(\mathbf{m}^\star, \mathbf{r})$ behaves as in $\mathcal{D}_\lambda^b$. From the LWE assumption, we have:

$$\mathcal{D}_\lambda^b \approx_c \mathcal{H}_\lambda^{b.1}.$$

- $\underline{\mathcal{H}_\lambda^{b.2}}$: is as $\mathcal{H}_\lambda^{b.1}$ except the matrix $\mathbf{U}$ from the public key is sampled from $(\mathbf{U}, T_{\mathbf{U}}) \leftarrow \mathsf{TrapGen}(1^\lambda, q, \kappa + 1)$. By Property 3, this is statistically close to generating a uniformly random $\mathbf{U} \leftarrow_{\mathsf{R}} \mathbb{Z}_N^{(\kappa+1) \times m}$ as done in $\mathcal{H}_\lambda^{b.1}$. The rest of the adversary view can be generated from $\mathbf{U}$, thus, we have:

$$\mathcal{H}_\lambda^{b.1} \approx_s \mathcal{H}_\lambda^{b.2}.$$

- $\underline{\mathcal{H}_\lambda^{b.3}}$: is as $\mathcal{H}_\lambda^{b.2}$ except we use the oracle $\widetilde{\mathcal{O}}_{\mathsf{SRL}}$ instead of $\mathcal{O}_{\mathsf{SRL}}$:

```
$\underline{\widetilde{\mathcal{O}}_{\mathsf{SRL}}(\mathbf{m}^\star, \mathsf{ct})}$:
$\mathbf{r}^\star \leftarrow_{\mathsf{R}} \mathcal{R}^\star$, $\mathsf{ct}^\star = \mathsf{Enc}^\star_{\mathsf{pk}}(0; \mathbf{r}^\star)$
$(f, \alpha) \leftarrow \mathcal{A}(\mathsf{ct}^\star)$
$\mathsf{BD}(\mathsf{ct}'_f) = \mathsf{Eval}'(\mathsf{pk}, f, 0, \mathsf{ct})$. Parse $\mathsf{ct}'_f = \big(\mathbf{A}\mathbf{r}_f, \mathbf{v}^\top \mathbf{r}_f + f(\mathbf{m}^\star)\big) \in \mathbb{Z}_N^{\kappa+1}$.
Compute $\mathbf{t}_f = (\mathbf{A}\mathbf{r}_f, \mathbf{v}^\top \mathbf{r}_f) \in \mathbb{Z}_N^{\kappa+1}$, and $\widetilde{\mathbf{r}}_f \leftarrow \mathsf{PreImSamp}(\mathbf{U}, T_{\mathbf{U}}, \mathbf{t}_f, \widetilde{B})$.
If $f(\mathbf{m}^\star) = \alpha$, and $f$ is of depth $d$, then $\mathsf{leak} = \mathbf{r}^\star - \widetilde{\mathbf{r}}_f \in \mathcal{R}^\star$.
Otherwise, $\mathsf{leak} = \perp$. Return $\mathsf{leak}$.
```

Note that the oracle $\widetilde{\mathcal{O}}_{\mathsf{SRL}}$ only takes as input the message $\mathbf{m}^\star \in \{0, 1\}^*$ and the challenge ciphertext $\mathsf{ct} = \mathsf{Enc}_{\mathsf{pk}}(\mathbf{m}^\star; \mathbf{r})$, but not the randomness $\mathbf{r}$ itself. Instead of computing the evaluated randomness $\mathbf{r}_f = \mathsf{Eval}_{\mathsf{rand}}(\mathsf{pk}, f, \mathbf{m}^\star, \mathbf{r})$, it computes a small $\widetilde{\mathbf{r}}_f$ that is consistent with ct_f, that is, such that $\mathsf{ct}_f = (\mathbf{A}\widetilde{\mathbf{r}}_f, \mathbf{v}^\top \widetilde{\mathbf{r}}_f + f(\mathbf{m}))$. Clearly, the distributions: $(\mathsf{ct}, \mathbf{r}_f)$, which corresponds to $\mathcal{H}_\lambda^{b.2}$ and $(\mathsf{ct}, \widetilde{\mathbf{r}}_f)$, which corresponds to $\mathcal{H}_\lambda^{b.3}$ are distinct — for one thing, the first distribution has less entropy than the second distribution where $\widetilde{\mathbf{r}}_f$ is sampled freshly. However, the value $\widetilde{\mathbf{r}}_f$ is shielded by the noisy randomness $\mathbf{r}^\star \leftarrow_{\mathsf{R}} \mathcal{R}^\star$. Because it is of much larger magnitude than $\mathbf{r}_f$ and $\widetilde{\mathbf{r}}_f$, the latter can smudge the difference $\delta_f = \mathbf{r}_f - \widetilde{\mathbf{r}}_f$, which would successfully transition from $\mathcal{H}_\lambda^{b.2}$ to $\mathcal{H}_\lambda^{b.3}$. To effectively hide δ_f, we need to make sure $\mathbf{r}^\star \in \mathbb{Z}^m$ itself is hidden. Partial information is revealed in $\mathsf{ct}^\star = \mathsf{Enc}^\star_{\mathsf{pk}}(0; \mathbf{r}^\star)$, of the form $\mathbf{U}\mathbf{r}^\star \in \mathbb{Z}_N^{\kappa+1}$. Intuitively, the component of $\mathbf{r}^\star$ along $\mathbf{U}$ is revealed by

$\mathsf{ct}^\star$, but the remaining entropy of $\mathbf{r}^\star$ is hidden; in particular, its component along $\mathbf{U}^\perp$, the orthogonal space of $\mathbf{U}$, is hidden. Because $\widetilde{\mathbf{r}}_f$ is consistent with ct_f, we have $\mathbf{U}\delta_f = \mathbf{0}$; that is, δ_f is orthogonal to $\mathbf{U}$. The orthogonal component of $\mathbf{r}^\star$ can simply smudge δ_f. This argument is formalized in Lemma 3.2. Overall, we have:

$$\mathcal{H}_\lambda^{b.2} \approx_s \mathcal{H}_\lambda^{b.3}.$$

To complete the proof of Theorem 3.6, we show that $\mathcal{H}_\lambda^{0.3} \approx_s \mathcal{H}_\lambda^{1.3}$, which gives overall:

$$\mathcal{D}_\lambda^0 \approx_c \mathcal{H}_\lambda^{0.1} \approx_s \mathcal{H}_\lambda^{0.2} \approx_s \mathcal{H}_\lambda^{0.3} \approx_s \mathcal{H}_\lambda^{1.3} \approx_s \mathcal{H}_\lambda^{1.2} \approx_s \mathcal{H}_\lambda^{1.1} \approx_c \mathcal{D}_\lambda^1.$$

We look at the challenge ciphertext in $\mathcal{H}_\lambda^{0.3}$. It of the form $\mathsf{ct} = (\mathsf{ct}_i)_{i\in[|\mathbf{m}^\star|]}$ where for all $i \in [|\mathbf{m}^\star|]$, $\mathsf{ct}_i = \mathbf{U}\mathbf{R}_i + m_i^\star \mathbf{G} \in \mathbb{Z}_N^{(\kappa+1)\times w}$, where the randomness $\mathbf{R}_i \leftarrow_{\mathsf{R}} [-1,1]^{m\times w}$, and $m_i^\star$ denotes the i'th bit of the message $\mathbf{m}^\star$. The only information revealed about $\mathbf{R}_i$ is $\mathbf{U}\mathbf{R}_i$ — in particular the oracle $\widetilde{\mathcal{O}}_{\mathsf{SRL}}$ does not require to know $\mathbf{R}_i$. Thus, we can use Lemma 3.1, which directly implies that the following two distributions have statistical distance at most $2^{-\lambda}$:

$$\Big\{(\mathbf{U}, T_{\mathbf{U}}) \leftarrow \mathsf{TrapGen}(1^\lambda, N, \kappa+1), \mathbf{R}_i \leftarrow_{\mathsf{R}} [-1,1]^{m\times w} : \Big(\mathbf{U}, T_{\mathbf{U}}, \big(\mathbf{U}\mathbf{R}_i + m_i^0\mathbf{G}\big)_{i\in[s]}\Big)\Big\}$$
$$\Big\{(\mathbf{U}, T_{\mathbf{U}}) \leftarrow \mathsf{TrapGen}(1^\lambda, N, \kappa+1), \mathbf{R}_i \leftarrow_{\mathsf{R}} [-1,1]^{m\times w} : \Big(\mathbf{U}, T_{\mathbf{U}}, \big(\mathbf{U}\mathbf{R}_i + m_i^1\mathbf{G}\big)_{i\in[s]}\Big)\Big\},$$

where $s = |\mathbf{m}^0| = |\mathbf{m}^1|$. The first distribution corresponds to $\mathcal{H}_\lambda^{3.0}$, whereas the second corresponds to $\mathcal{H}_\lambda^{3.1}$. □

Lemma 3.2. *The following ensembles are statistically close:* $\mathcal{H}_\lambda^{b.2} \approx_s \mathcal{H}_\lambda^{b.3}$.

Proof: We introduce intermediate ensembles $\{\mathcal{H}_\lambda^{b.2.i}\}_{\lambda\in\mathbb{N}}$ for $i = 1,2$ which are defined as follows.

Ensemble $\mathcal{H}_\lambda^{b.2.1}$ is as $\mathcal{H}_\lambda^{b.2}$ except is uses the following oracle $\mathcal{O}_{\mathsf{SRL}}^1$ instead of $\mathcal{O}_{\mathsf{SRL}}$.

```
$\underline{\mathcal{O}_{\mathsf{SRL}}^1(\mathbf{m}^\star, \mathbf{r})}$:
$\mathbf{r}^\star \leftarrow_{\mathsf{R}} \mathcal{R}^\star$, $\widetilde{\mathbf{r}}_0 \leftarrow \mathsf{PreImSamp}(\mathbf{U}, T_{\mathbf{U}}, \mathbf{0}, \widetilde{B})$, $\mathsf{ct}^\star = \mathsf{Enc}_{\mathsf{pk}}^\star(0; \mathbf{r}^\star - \widetilde{\mathbf{r}}_0)$
$(f, \alpha) \leftarrow \mathcal{A}(\mathsf{ct}^\star)$
$\mathbf{r}_f = \mathsf{Eval}_{\mathsf{rand}}(\mathsf{pk}, f, \mathbf{r}, \mathbf{m}^\star)$.
If $f(\mathbf{m}^\star) = \alpha$ and $f$ is of depth $d$, then $\mathsf{leak} = \mathbf{r}^\star - \widetilde{\mathbf{r}}_0 - \mathbf{r}_f \in \mathcal{R}^\star$.
Otherwise, $\mathsf{leak} = \perp$. Return $\mathsf{leak}$.
```

We first prove that:

$$\mathcal{H}_\lambda^{b.2} \approx_s \mathcal{H}_\lambda^{b.2.1}.$$

The only difference between these ensembles is that $\mathcal{O}_{\mathsf{SRL}}^1$ adds a pre-image of $\mathbf{0} \in \mathbb{Z}_N^m$ to the shielded randomness, that is, it uses $\mathbf{r}^\star - \widetilde{\mathbf{r}}_0$ with $\mathbf{r}^\star \leftarrow_{\mathsf{R}} \mathcal{R}^\star$ and $\widetilde{\mathbf{r}}_0 \leftarrow \mathsf{PreImSamp}(\mathbf{U}, T_{\mathbf{U}}, \mathbf{0}, \widetilde{B})$ instead of $\mathbf{r}^\star$.

By Property 4, $\widetilde{\mathbf{r}}_0 \in \mathbb{Z}_N^m$ is such that $\|\widetilde{\mathbf{r}}_0\|_\infty < \widetilde{B}2^{\lambda/2}$. Thus, by Lemma 2.2 (smudging), the following distributions have statistical distance at most $2^{-\lambda/2}$:

$$\{\mathbf{r}^\star \leftarrow_{\mathsf{R}} [-\widetilde{B}2^\lambda, \widetilde{B}2^\lambda]^m : \mathbf{r}^\star\} \text{ and } \{\mathbf{r}^\star \leftarrow_{\mathsf{R}} [-\widetilde{B}2^\lambda, \widetilde{B}2^\lambda]^m : \mathbf{r}^\star - \widetilde{\mathbf{r}}_0\}.$$

The leftmost ensemble corresponds to $\mathcal{H}_\lambda^{b.2}$, whereas the rightmost ensemble corresponds to $\mathcal{H}_\lambda^{b.2.1}$. This completes the proof that $\mathcal{H}_\lambda^{b.2} \approx_s \mathcal{H}_\lambda^{b.2.1}$.

Now, we introduce another intermediate ensemble, $\mathcal{H}_\lambda^{b.2.2}$, which is defined as $\mathcal{H}_\lambda^{b.2.1}$ except is uses the following oracle $\mathcal{O}_{\mathsf{SRL}}^2$ instead of $\mathcal{O}_{\mathsf{SRL}}^1$.

```
$\underline{\mathcal{O}^2_{\mathsf{SRL}}(\mathbf{m}^\star, \mathbf{r})}$:
$\mathbf{r}^\star \leftarrow_{\mathsf{R}} \mathcal{R}^\star$, $\mathsf{ct}^\star = \mathsf{Enc}^\star_{\mathsf{pk}}(0; \mathbf{r}^\star)$
$(f, \alpha) \leftarrow \mathcal{A}(\mathsf{ct}^\star)$
$\mathbf{r}_f = \mathsf{Eval}_{\mathsf{rand}}(\mathsf{pk}, f, \mathbf{r}, \mathbf{m}^\star)$, $\widetilde{\mathbf{r}_0} \leftarrow \mathsf{PreImSamp}(\mathbf{U}, T_{\mathbf{U}}, \mathbf{0}, \widetilde{B})$.
If $f(\mathbf{m}^\star) = \alpha$ and $f$ is of depth $d$, then $\mathsf{leak} = \mathbf{r}^\star - \widetilde{\mathbf{r}_0} - \mathbf{r}_f \in \mathcal{R}^\star$.
Otherwise, $\mathsf{leak} = \perp$. Return $\mathsf{leak}$.
```

by Property 4, we have, $\mathbf{U}\widetilde{\mathbf{r}_0} = \mathbf{0}$. This implies $\mathsf{Enc}^\star_{\mathsf{pk}}(0; \mathbf{r}^\star - \widetilde{\mathbf{r}_0}) = \mathsf{Enc}^\star_{\mathsf{pk}}(0; \mathbf{r}^\star)$. Thus, we have:

$$\mathcal{H}^{b.2.1}_\lambda = \mathcal{H}^{b.2.2}_\lambda.$$

To conclude the proof of this lemma, we now prove that:

$$\mathcal{H}^{b.2.2}_\lambda \approx_s \mathcal{H}^{b.3}_\lambda.$$

To do so, we note that $\mathbf{r}_f \in \mathbb{Z}_N^m$ is such that $\|\mathbf{r}_f\|_\infty < \widetilde{B}$. Moreover, it is independent of the vector $\widetilde{\mathbf{r}_0} \leftarrow \mathsf{PreImSamp}(\mathbf{U}, T_{\mathbf{U}}, \mathbf{0}, \widetilde{B})$. Therefore, we can use Proposition 5, which states that the following distributions have statistical distance at most $2^{-\lambda/2}$:

$$\{\widetilde{\mathbf{r}_0} \leftarrow_{\mathsf{R}} \mathsf{PreImSamp}(\mathbf{U}, T_{\mathbf{U}}, \mathbf{0}, \widetilde{B}) : \widetilde{\mathbf{r}_0} + \mathbf{r}_f \in \mathbb{Z}_N^m\} \text{ and } \{\widetilde{\mathbf{r}_f} \leftarrow \mathsf{PreImSamp}(\mathbf{U}, T_{\mathbf{U}}, \mathbf{U}\mathbf{r}_f, \widetilde{B}) : \widetilde{\mathbf{r}_f}\}.$$

The leftmost distribution corresponds to $\mathcal{H}^{b.2.2}_\lambda$, whereas the rightmost distribution corresponds to $\mathcal{H}^{b.3}_\lambda$. □

4 Hintable Linearly Homomorphic Encryption

BDGM [BDGM20a] introduced the notion of "hintable" Linearly Homomorphic Encryption (LHE). Roughly speaking, an LHE scheme is said to be hintable if there is a secret-key algorithm that given a ciphertext, produces a "short" decryption hint. The latter can be used to decrypt the ciphertext is was generated from, without the secret key. It is also possible to generate a hint from a ciphertext only knowing the random coins used to produce that ciphertext (but without knowledge of the secret key), and the hints generated in these two ways should be statistically close. For our purposes (and as explained in the introduction), we will need to consider a notion of a hintable LHE satisfying a "weak circuit privacy" notion.

Additionally, we here generalize the notion of a hintable LHE to also consider "packed" LHE, where we can encrypt a vector of messages. Additionally, (just as we did for FHE), we will consider LHE with two encryption modes: a "normal" and an "extra noisy" mode. Linear functions can be evaluated on normal encryptions; furthermore *one* addition with a noisy encryption can be performed. More additions with noisy encryptions would lead to ill-formed ciphertexts that cannot be decrypted properly. (We introduce these extra generalizations to be able to obtain an instantiation based on LWE; these extra generalizations are not needed to capture DJ).

More precisely, we consider the notion of an (ℓ_1, ℓ_2, h)-hintable packed LHE which enables operating over a plaintext space of length $\ell_2(\lambda)$ vectors over $\mathbb{Z}_N$ for some modulus $|N| \geq \ell_1(\lambda)$, and release hints of size $h(\lambda)$. We will be interested in schemes where either ℓ_1 or ℓ_2 can be made arbitrarily big, while keeping h the same (i.e, the hint will become significantly shorter than a single group element, or it will be significantly smaller than the packing capacity).

We will present two constructions satifying the notion of a hintable packed LHE. The first one is the Damgård-Jurik [DJ01] encryption scheme which is proven secure under the DCR assumption:

this construction considers the setting where $\ell_2(\lambda) = 1$ (i.e., there is no packing, and instead the group elements are directly much larger than the size of the hint). The second construction will instead be a tweaked version of Packed Regev [Reg05, PVW08] which is secure under the LWE assumption; in this construction, $\ell_1(\lambda)$ is small (comparable to the hint size), but instead $\ell_2(\lambda)$ can be made arbitrarily large (i.e., we can pack a large number of elements into a ciphertext and still keep the hint size small).

4.1 Definition of Hintable LHE

We proceed to the formal defnition.

Definition 4.1 (hintable LHE). *An* (ℓ_1, ℓ_2, h)-Hintable Packed LHE *comprises the following PPT algorithms:*

- $\mathsf{CRSgen}(1^\lambda)$*: given as input the security parameter* $\lambda \in \mathbb{N}$*, it outputs* crs*.*
- $\mathsf{Gen}(\mathsf{crs})$*: given as input* crs*, it outputs the pair* $(\mathsf{pk}, \mathsf{sk})$ *that defines the message space* $\mathbb{Z}_N^{\ell_2(\lambda)}$*, where* $N \geq 2^{\ell_1(\lambda)}$*.*
- $\mathsf{Enc}_{\mathsf{pk}}(\mathbf{x})$*: given as input the public key* pk *and a vector in* $\mathbf{x} \in \mathbb{Z}_N^{\nu}$*, it outputs a ciphertext* ct*.*
- $\mathsf{Enc}^\star_{\mathsf{pk}}(\mathbf{m})$*: given as input the public key* pk *and a message in* $\mathbf{m} \in \mathbb{Z}_N^{\ell_2(\lambda)}$*, it outputs a noisy ciphertext* $\mathsf{ct}^\star$*.*
- $\mathsf{Eval}(\mathsf{pk}, \mathsf{ct}, \mathsf{ct}^\star, \mathbf{y})$*: given as input the public key* pk*, ciphertexts* ct*, a noisy ciphertext* $\mathsf{ct}^\star$ *and a function* $\mathbf{y} \in [-1, 1]^{\nu\ell_2}$*, it outputs an evaluated ciphertext* $\mathsf{ct}_{\mathbf{y}}$*.*
- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}^\star)$*: given as input the secret key and a (noisy or evaluated) ciphertext* $\mathsf{ct}^\star$*, it outputs a plaintext.*
- $\mathsf{SecHint}(\mathsf{sk}, \mathsf{ct}^\star)$*: given as input the secret key* sk *and a (noisy or evaluated) ciphertext* $\mathsf{ct}^\star$*, it outputs a decryption hint* ρ*.*
- $\mathsf{PubHint}(\mathsf{pk}, r)$*: given as input the public key and some random coins* $r \in \mathcal{R}^\star$*, where* $\mathcal{R}^\star$ *denotes the randomness space of* $\mathsf{Enc}^\star$*, it outputs a hint* ρ*.*
- $\mathsf{Rec}(\mathsf{pk}, \mathsf{ct}^\star, \rho)$*: given as input a (noisy or evaluated) ciphertext and a decryption hint* ρ*, it outputs a plaintext.*

These PPT algorithms additionally need to satisfy the properties listed below.

Property 4.1 (Correctness). *For all* $\lambda \in \mathbb{N}$*, all* crs *in the support of* $\mathsf{CRSgen}(1^\lambda)$*, all pairs* $(\mathsf{pk}, \mathsf{sk})$ *in the support of* $\mathsf{Gen}(\mathsf{crs})$ *that define the message space* $\mathbb{Z}_N^{\ell_2(\lambda)}$ *where* $N \geq 2^{\ell_1(\lambda)}$*, all messages* $\mathbf{m} \in \mathbb{Z}_N^{\ell_2(\lambda)}$*, all ciphertexts* $\mathsf{ct}^\star$ *in the support of* $\mathsf{Enc}^\star_{\mathsf{pk}}(\mathbf{m})$*, we have* $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}^\star) = \mathbf{m}$.

Property 4.2 (Linear Homomorphism). *For all polynomials* $\nu(\cdot)$*, all* $\lambda \in \mathbb{N}$*, all* crs *in the support of* $\mathsf{CRSgen}(1^\lambda)$*, all pairs* $(\mathsf{pk}, \mathsf{sk})$ *in the support of* $\mathsf{Gen}(\mathsf{crs})$ *that define the message space* $\mathbb{Z}_N^{\ell_2(\lambda)}$*, all vectors* $\mathbf{x} \in \mathbb{Z}_N^{\nu(\lambda)}$*, all ciphertexts* ct *in the support of* $\mathsf{Enc}_{\mathsf{pk}}(\mathbf{x})$*, all messages* $\mathbf{m} \in \mathbb{Z}_N^{\ell_2(\lambda)}$*, all ciphertexts* $\mathsf{ct}^\star$ *in the support of* $\mathsf{Enc}^\star_{\mathsf{pk}}(\mathbf{x}^\star)$*, all vectors* $\mathbf{y} = (\mathbf{y}_1, \ldots, \mathbf{y}_{\ell_2}) \in \{0,1\}^{\nu(\lambda)\ell_2}$*, the algorithm* Eval *deterministically outputs an evaluated ciphertext* $\mathsf{ct}_{\mathbf{y}} = \mathsf{Eval}(\mathsf{pk}, \mathsf{ct}, \mathsf{ct}^\star, \mathbf{y})$ *that is in the support of* $\mathsf{Enc}^\star_{\mathsf{pk}}(m_1 + \mathbf{x}^\top \mathbf{y}_1, \ldots, m_{\ell_2} + \mathbf{x}^\top \mathbf{y}_{\ell_2})$*.*

Property 4.3 (Correctness of secret hints)**.** *For all* $\lambda \in \mathbb{N}$*, all* crs *in the support of* $\mathsf{CRSgen}(1^\lambda)$*, all pairs* $(\mathsf{pk}, \mathsf{sk})$ *in the support of* $\mathsf{Gen}(\mathsf{crs})$ *that define the message space* $\mathbb{Z}_N^{\ell_2(\lambda)}$*, for all messages* $\mathbf{m} \in \mathbb{Z}_N^{\ell_2(\lambda)}$*, we have:* $\Pr\left[\mathsf{ct}^\star \leftarrow \mathsf{Enc}^\star_{\mathsf{pk}}(\mathbf{m}), \rho \leftarrow \mathsf{SecHint}(\mathsf{sk}, \mathsf{ct}^\star) : \mathsf{Rec}(\mathsf{pk}, \mathsf{ct}^\star, \rho) = \mathbf{m}\right] \in 1 - 2^{-\Omega(\lambda)}$*, where the probability is taken over the random coins of* $\mathsf{Enc}^\star$ *and* $\mathsf{SecHint}$*.*

Property 4.4 (Correctness of public hints)**.** *For all* $\lambda \in \mathbb{N}$*, all* crs *in the support of* $\mathsf{CRSgen}(1^\lambda)$*, all messages* $\mathbf{m} \in \mathbb{Z}_N^{\ell_2(\lambda)}$*, the following distributions have statistical distance at most* $2^{-\Omega(\lambda)}$*:*

$$\{(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda), \mathsf{ct}^\star \leftarrow \mathsf{Enc}^\star_{\mathsf{pk}}(\mathbf{m}), \rho \leftarrow \mathsf{SecHint}(\mathsf{sk}, \mathsf{ct}) : (\mathsf{pk}, \mathsf{ct}^\star, \rho)\}$$
$$\{(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda), r \leftarrow_{\mathsf{R}} \mathcal{R}^\star, \rho \leftarrow \mathsf{PubHint}(\mathsf{pk}, r), \mathsf{ct}^\star = \mathsf{Enc}^\star_{\mathsf{pk}}(\mathbf{m}; r) : (\mathsf{pk}, \mathsf{ct}^\star, \rho)\}$$

Property 4.5 (h-succinctness of hints)**.** *For all* $\lambda \in \mathbb{N}$*, all* crs *in the support of* $\mathsf{CRSgen}(1^\lambda)$*, all pairs* $(\mathsf{pk}, \mathsf{sk})$ *in the support of* $\mathsf{Gen}(\mathsf{crs})$ *that define the message space* $\mathbb{Z}_N^{\ell_2(\lambda)}$*, all messages* $\mathbf{x} \in \mathbb{Z}_N^{\ell_2(\lambda)}$*, all ciphertexts* $\mathsf{ct}^\star$ *in the support of* $\mathsf{Enc}^\star_{\mathsf{pk}}(\mathbf{x})$*, all hints* ρ *in the support of* $\mathsf{SecHint}(\mathsf{sk}, \mathsf{ct}^\star)$ *are of size at most* $h(\lambda)$*.*

Property 4.6 (Weak circuit privacy)**.** *For all polynomials* $\nu(\cdot)$*, all* $\lambda \in \mathbb{N}$*, all* crs *in the support of* $\mathsf{CRSgen}(1^\lambda)$*, all pairs* $(\mathsf{pk}, \mathsf{sk})$ *in the support of* $\mathsf{Gen}(\mathsf{crs})$ *that define the message space* $\mathbb{Z}_N^{\ell_2(\lambda)}$*, all messages* $\mathbf{m} \in \mathbb{Z}_N^{\ell_2}$*, all vectors* $\mathbf{x} \in \mathbb{Z}_N^{\nu(\lambda)}$*, all vectors* $\mathbf{y} \in [-1, 1]^{\nu(\lambda)\ell_2(\lambda)}$*, the following distributions have statistical distance at most* $2^{-\Omega(\lambda)}$*:*

$$\{\mathsf{ct} \leftarrow \mathsf{Enc}_{\mathsf{pk}}(\mathbf{x}), \mathsf{ct}^\star \leftarrow \mathsf{Enc}^\star_{\mathsf{pk}}(\mathbf{m}), \mathsf{ct}_{\mathbf{y}} = \mathsf{Eval}(\mathsf{pk}, \mathsf{ct}, \mathsf{ct}^\star, \mathbf{y}) : (\mathsf{pk}, \mathsf{crs}, \mathsf{ct}, \mathsf{ct}^\star, \mathsf{ct}_{\mathbf{y}})\}$$
$$\{\mathsf{ct} \leftarrow \mathsf{Enc}_{\mathsf{pk}}(\mathbf{x}), \mathsf{ct}_{\mathbf{y}} \leftarrow \mathsf{Enc}^\star_{\mathsf{pk}}(\mu), \mathsf{ct}^\star = \mathsf{Eval}(\mathsf{pk}, \mathsf{ct}, \mathsf{ct}_{\mathbf{y}}, -\mathbf{y}) : (\mathsf{crs}, \mathsf{pk}, \mathsf{ct}, \mathsf{ct}^\star, \mathsf{ct}_{\mathbf{y}})\},$$

where $\mu = (m_1 + \mathbf{x}^\top \mathbf{y}_1, \ldots, m_{\ell_2} + \mathbf{x}^\top \mathbf{y}_{\ell_2}) \in \mathbb{Z}_N^{\ell_2}$*.*

Property 4.7 (Density of the noisy ciphertexts)**.** *There exists a polynomial* $s(\cdot)$ *and a poly-time determinstic function* $\mathsf{CTsample}$ *such that the following distributions have statistical distance at most* $2^{-\Omega(\lambda)}$*:*

$$\left\{\mathsf{crs} \leftarrow \mathsf{CRSgen}(1^\lambda), (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(\mathsf{crs}), \mathbf{r} \leftarrow_{\mathsf{R}} \{0,1\}^{s(\lambda)} : \mathsf{CTsample}(\mathbf{r})\right\}.$$
$$\left\{\mathsf{crs} \leftarrow \mathsf{CRSgen}(1^\lambda), (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(\mathsf{crs}), \mathbf{m} \leftarrow_{\mathsf{R}} \mathbb{Z}_N^{\ell_2(\lambda)}, \mathsf{ct}^\star \leftarrow \mathsf{Enc}^\star_{\mathsf{pk}}(\mathbf{m}) : \mathsf{ct}^\star\right\}.$$

4.2 Hintable LHE from DCR

We consider the Damgård Jurik Linearly Homomorphic Encryption scheme from [DJ01], which generalizes Paillier's encryption scheme [Pai99] to larger message spaces, whose security relies on the Decisional Composite Residuosity (DCR) assumption:

Definition 4.2 (Decisional Composite Residuosity (DCR) assumption [Pai99])**.** *There exists a PPT algorithm* $\mathsf{RSAsample}$ *that on input a security parameter* λ*, outputs a pair* $(N, \phi(N))$ *where* N *is a* 2λ*-bits integer,* ϕ *denotes Euler's totient function; such that* $\gcd(\phi(N), N) = 1$ *and such that for all polynomial* $\zeta(\cdot)$*, the following ensembles are computationally indistinguishable:*

$$\{\mathcal{D}^0_\lambda\}_{\lambda \in \mathbb{N}} = \left\{(N, \phi(N)) \leftarrow \mathsf{RSAsample}(1^\lambda); r \leftarrow_{\mathsf{R}} \mathbb{Z}_M : r^{N^{\zeta(\lambda)}} \in \mathbb{Z}_{N^{\zeta(\lambda)+1}}\right\}_{\lambda \in \mathbb{N}}$$
$$\{\mathcal{D}^0_\lambda\}_{\lambda \in \mathbb{N}} = \left\{(N, \phi(N)) \leftarrow \mathsf{RSAsample}(1^\lambda); u \leftarrow_{\mathsf{R}} \mathbb{Z}_{N^{\zeta(\lambda)+1}} : u \in \mathbb{Z}_{N^{\zeta(\lambda)+1}}\right\}_{\lambda \in \mathbb{N}}$$

As explained in [DJ01] in further details, the algorithm $\mathsf{RSAsample}(1^\lambda)$ samples two safe primes p, q of λ bits each, and compute the RSA modulus $N = pq$.

We will show how the DJ encryption scheme satisfies our notion of a hintable packed LHE (actually even without any packing):

Theorem 4.3. *Assuming the DCR assumption, for every polynomial ℓ_1, there exists a secure (ℓ_1, ℓ_2, h)-hintable packed LHE, where $h(\lambda) = 2\lambda$ and $\ell_2(\lambda) = 1$.*

4.2.1 The DJ scheme

Given a polynomial $\ell_1(\cdot)$, we recall the LHE from [DJ01] when operating on plaintext of size ℓ_1 (recall that $\ell_2 = 1$ so there is no packing). That is, the scheme is parameterized by a polynomial ℓ_1, and we call it DJ_{ℓ_1}. For this scheme, the hint is simply the randomness used to encrypt a message, and noisy and normal encryptions behave in the same way:

- $\underline{\mathsf{Gen}(\mathsf{crs})}$:

Given $\mathsf{crs} = 1^\lambda$, it uses the sampling algorithm from Definition 4.2, $(M, \phi(M)) \leftarrow \mathsf{RSAsample}(1^\lambda)$, where $M \in \mathbb{N}$ is a 2λ-bit modulus, ϕ denotes Euler's totient function, and we have $\gcd(\phi(M), M) = 1$. Then it chooses a polynomial $\zeta(\cdot)$ such that $2^{\ell_1(\lambda)+2\lambda} > N \geq 2^{\ell_1(\lambda)}$, where $N = M^{\zeta(\lambda)}$. For simplicity of the notations, we write $\zeta = \zeta(\lambda)$. It sets $\mathsf{pk} = (N, \zeta)$, $\mathsf{sk} = \phi(M)$ and outputs $(\mathsf{pk}, \mathsf{sk})$. The plaintext space is $\mathbb{Z}_N = \mathbb{Z}_{M^\zeta}$. The randomness space for Enc is $\mathbb{Z}_M^*$, the ciphertext space is $\mathbb{Z}_{M^\zeta}^*$, and the function space is $\mathbb{Z}_N$, that is $t = N$.

- $\underline{\mathsf{Enc}_{\mathsf{pk}}(\mathbf{x})}$:

Given the public pk, a vector $\mathbf{x} \in \mathbb{Z}_N^\nu$, for all $i \in [\nu]$, it samples $r_i \leftarrow_{\mathsf{R}} \mathbb{Z}_M^*$ and compute $\mathsf{ct}_i = r_i^{M^\zeta} \cdot (1 + M)^{x_i} \in \mathbb{Z}_{M^{\zeta+1}}^*$. It outputs the ciphertext $\mathsf{ct} = (\mathsf{ct}_1, \ldots, \mathsf{ct}_\nu)$.

- $\underline{\mathsf{Enc}^\star_{\mathsf{pk}}(m)}$:

Given the public pk, a message $m \in \mathbb{Z}_N$, it samples $r \leftarrow_{\mathsf{R}} \mathbb{Z}_M^*$ and outputs the noisy ciphertext $\mathsf{ct}^\star = r^{M^\zeta} \cdot (1 + M)^m \in \mathbb{Z}_{M^{\zeta+1}}^*$.

- $\underline{\mathsf{Eval}(\mathsf{pk}, \mathsf{ct}, \mathsf{ct}^\star, \mathbf{y})}$:

Given as input the public key pk, ciphertext $\mathsf{ct} \in \mathbb{Z}_{M^{\zeta+1}}^{*\nu}$, noisy ciphertext $\mathsf{ct}^\star \in \mathbb{Z}_{M^{\zeta+1}}^*$, and a vector $\mathbf{y} \in [-1, 1]^\nu$, it outputs the evaluated ciphertext $\mathsf{ct}^\star \cdot \prod_{i\in[\nu]} \mathsf{ct}_i^{y_i} \in \mathbb{Z}_{M^{\zeta+1}}^*$, where $\cdot$ denotes the integer multiplication in $\mathbb{Z}_{M^{\zeta+1}}^*$.

- $\underline{\mathsf{SecHint}(\mathsf{sk}, \mathsf{ct}^\star)}$:

Given the secret key sk and a noisy ciphertext $\mathsf{ct}^\star \in \mathbb{Z}_{M^{\zeta+1}}^*$, it computes $d = \mathsf{ct} \mod M$. Since $\gcd(M^\zeta, \phi(M)) = 1$, it can compute $M^{-\zeta} \in \mathbb{Z}$ such that $M^\zeta \cdot M^{-\zeta} = 1 \mod \phi(M)$. It outputs the hint $d^{M^{-\zeta}} \in \mathbb{Z}_M^*$.

- $\underline{\mathsf{PubHint}(\mathsf{pk}, r)}$:

Given the public key pk and some randomness $r \in Z_M^*$, it outputs the hint $\rho = r$.

- $\underline{\mathsf{Rec}(\mathsf{pk}, \mathsf{ct}^\star, \rho)}$:

Given the public key pk, a noisy ciphertext $\mathsf{ct}^\star$, and a hint $\rho \in \mathbb{Z}_M^*$, it computes $d = \mathsf{ct} \cdot r^{-M^\zeta} \in \mathbb{Z}_{M^{\zeta+1}}^*$, where ρ^{-M^ζ} is the inverse of ρ^{M^ζ} in $\mathbb{Z}_{M^{\zeta+1}}^*$. Then, it applies Paillier's decryption recursively to obtain $x \in \mathbb{Z}_N$. It outputs $x \in \mathbb{Z}_N$.

• $\underline{\mathsf{Dec}(\mathsf{pk}, \mathsf{ct}^\star)}$:
Given the secret key sk, a noisy ciphertext $\mathsf{ct}^\star$, it runs $\mathsf{Ext}(\mathsf{sk}, \mathsf{ct}^\star)$ to recover the randomness $r \in \mathbb{Z}_M^*$, then outputs $\mathsf{Rec}(\mathsf{pk}, \mathsf{ct}^\star, r)$.

The proof of Theorem 4.3 follows from the propositions and theorem below (which demonstrate that DJ satisfies the desired properties of a hintable packed LHE, as well as security).

Proposition 6 (Linear Homomorphism)**.** *The LHE presented above satisfies Property 4.2 (linear homomorphism).*

Proof: For all $i \in [\nu]$, let ct_i be a ciphertext in the support of $\mathsf{Enc}_{\mathsf{pk}}(x_i)$, that is, of the form $\mathsf{ct}_i = (r_i)^{M^\zeta} \cdot (1+M)^{x_i} \in \mathbb{Z}^*_{M^{\zeta+1}}$, and let $\mathsf{ct}^\star$ be a ciphertext in the support of $\mathsf{Enc}^\star_{\mathsf{pk}}(x^\star)$, that is, of the form $\mathsf{ct}^\star = r^{M^\zeta} \cdot (1+M)^{x^\star} \in \mathbb{Z}^*_{M^{\zeta+1}}$. For all $\mathbf{y} \in \{0,1\}^\nu$, the evaluated ciphertext $\mathsf{ct}_{\mathbf{y}}$ is of the form: $\left(r \prod_{i\in[\nu]} r_i^{y_i}\right)^{M^\zeta} \cdot (1+M)^{x^\star + \sum_{i\in[\nu]} x_i y_i} \in \mathbb{Z}^*_{M^{\zeta+1}}$, which is in the support of $\mathsf{Enc}^\star_{\mathsf{pk}}(x^\star + \sum_{i\in[\nu]} x_i y_i)$. □

Proposition 7 (Batch evaluation)**.** *The LHE presented above satisfies Property* **??** *(batch evaluation).*

Proof: This immediately follow from the linear homomorphism property. □

Proposition 8 (Correctness of secret hints)**.** *The LHE presented above satisfies Property 4.3 (correctness of hints).*

Proof: Let $\mathsf{ct}^\star = r^{M^\zeta} \cdot (1+M)^{x^\star} \in \mathbb{Z}^*_{M^{\zeta+1}}$, where $x^\star \in \mathbb{Z}_{M^\zeta}$ and $r \in \mathbb{Z}^*_M$. We have $\mathsf{ct}^\star \bmod M = r^{M^\zeta \bmod \phi(M)} \in \mathbb{Z}^*_M$. Since $\gcd(\phi(M), M^\zeta) = 1$, there is $M^{-\zeta} \in \mathbb{Z}$ such that $M^\zeta \cdot M^{-\zeta} = 1 \bmod \phi(M)$. Thus, the algorithm $\mathsf{SecHint}(\mathsf{sk}, \mathsf{ct})$ outputs $r \in \mathbb{Z}^*_M$. That is, the hint output by $\mathsf{SecHint}$ is the randomness used to produce $\mathsf{ct}^\star$.

The algorithm $\mathsf{Rec}(\mathsf{pk}, \mathsf{ct}^\star, r)$ computes $d = \mathsf{ct}^\star \cdot r^{-M^\zeta} = (1+M)^{x^\star} \in \mathbb{Z}^*_{M^{\zeta+1}}$ where r^{-M^ζ} is the inverse of r^{M^ζ} in $\mathbb{Z}^*_{M^{\zeta+1}}$. Then it applies Paillier's decryption recursively to obtain $x^\star \in \mathbb{Z}^*_{M^\zeta}$. It outputs $x^\star$. □

Proposition 9 (Correctness of the public hints)**.** *The LHE presented above satisfies Property 4.4 (correctness of public hints).*

Proof: As seen in the proof of Proposition 8, the hint recovered by $\mathsf{SecHint}$ on input the secret key sk and a ciphertext $\mathsf{ct}^\star$ is the randomness r used by $\mathsf{Enc}^\star$ to produce $\mathsf{ct}^\star$, which is the output also what $\mathsf{PubHint}(\mathsf{pk}, r)$ outputs. □

Proposition 10 (Correctness)**.** *The LHE presented above satisfies Property 4.1 (correctness).*

Proof: Correctness follows from the correctness of the secret hints (Property 4.3). □

Proposition 11 (h-succinctness of hints)**.** *The LHE presented above satisfies $h(\lambda) = 2\lambda$-succinctness.*

Proof: For all $\lambda \in \mathbb{N}$, for all pairs $(\mathsf{pk}, \mathsf{sk})$ in the support of $\mathsf{Gen}(1^\lambda)$ that define the message space $\mathbb{Z}_N$, for all $x^\star \in \mathbb{Z}_N$, all ciphertext $\mathsf{ct}^\star$ in the support of $\mathsf{Enc}^\star_{\mathsf{pk}}(x^\star)$, we have: all hints ρ in the support of $\mathsf{SecHint}(\mathsf{sk}, \mathsf{ct}^\star)$ are in $\mathbb{Z}^*_M$, where M is an RSA modulus of size at most 2λ bits. □

Proposition 12 (Weak circuit privacy)**.** *The LHE presented above satisfies Property 4.6 (weak circuit privacy).*

Proof: For any message $\mathbf{x} = (x_1, \ldots, x_\nu) \in \mathbb{Z}_N^\nu$, $x^\star \in \mathbb{Z}_N$, $\mathbf{y} \in \{0,1\}^\nu$, we aim at proving that following distributions are identical:

$$\mathcal{D}_0 : \left\{ \begin{array}{l} \forall i \in [\nu], r_i \leftarrow_{\mathsf{R}} \mathbb{Z}_M^*, \mathsf{ct}_i = r_i^{M^\zeta} \cdot (1+M)^{x_i}, r \leftarrow_{\mathsf{R}} \mathbb{Z}_M^*, \mathsf{ct}^\star = r^{M^\zeta} \cdot (1+M)^{x^\star} \\ \mathsf{ct}_{\mathbf{y}} = (r \prod_i r_i^{y_i})^{M^\zeta} \cdot (1+M)^{x^\star + \mathbf{x}^\top \mathbf{y}} : ((\mathsf{ct}_i)_i, \mathsf{ct}^\star, \mathsf{ct}_{\mathbf{y}}) \end{array} \right\}$$

$$\mathcal{D}_1 : \left\{ \begin{array}{l} \forall i \in [\nu], r_i \leftarrow_{\mathsf{R}} \mathbb{Z}_M^*, \mathsf{ct}_i = r_i^{M^\zeta} \cdot (1+M)^{x_i}, r \leftarrow_{\mathsf{R}} \mathbb{Z}_M^*, \mathsf{ct}^\star = (r/\prod_i r_i^{y_i})^{M^\zeta} \cdot (1+M)^{x^\star} \\ \mathsf{ct}_{\mathbf{y}} = r^{M^\zeta} \cdot (1+M)^{x^\star + \mathbf{x}^\top \mathbf{y}} : ((\mathsf{ct}_i)_i, \mathsf{ct}^\star, \mathsf{ct}_{\mathbf{y}}) \end{array} \right\}.$$

This relies on the fact that for all $i \in [\nu]$, all $r_i \in \mathbb{Z}_M^*$, all $y_i \in \mathbb{Z}_{M^\zeta}$, the following distributions are identical: $\mathcal{D}'_0 = \{r \leftarrow_{\mathsf{R}} \mathbb{Z}_M^* : ((r_i)_i, r, r \cdot \prod_i r_i^{y_i})\}$ and $\mathcal{D}'_1 = \{r \leftarrow_{\mathsf{R}} \mathbb{Z}_M^* : ((r_i)_i, r/(\prod_i r_i^{y_i}), r)\}$. The distribution $\mathcal{D}'_0$ corresponds to $\mathcal{D}_0$, whereas $\mathcal{D}'_1$ corresponds to $\mathcal{D}_1$. □

Proposition 13 (Density of the noisy ciphertexts)**.** *The LHE presented above satisfies Property 4.7 (density of the noisy ciphertexts).*

Proof: One can sample a uniform random value $u \leftarrow_{\mathsf{R}} \mathbb{Z}_{M^{\zeta+1}}$ from $\lceil \log(M^{\zeta+1}) \rceil$ random bits. The random value $u \in \mathbb{Z}_{M^{\zeta+1}}$ can be written $u = r^{M^\zeta} \cdot (1+M)^x$ where $x \in \mathbb{Z}_{M^\zeta}$ and $r \in \mathbb{Z}_M^*$ with probability $1 - \frac{\phi(N)}{N} > 1 - \frac{3}{2^\lambda}$ over the choice of $u \leftarrow_{\mathsf{R}} \mathbb{Z}_{M^{\zeta+1}}$. □

Theorem 4.4 (Security [DJ01])**.** *Assuming the DCR assumption (see Definition 4.2), the DJ scheme is secure.*

4.3 Hintable LHE from LWE

We present a packed version of Regev encryption scheme [Reg05], and demonstrate that it can be used to satisfy our notion of a (ℓ_1, ℓ_2, h)-hintable packed LHE, where ℓ_1 and ℓ_2 can be any polynomial, and $h(\lambda)$ is a polynomial that that is larger than ℓ_1 but is independent of ℓ_2. That is, the hint is large in comparison to individual group elements, but we can pack in an arbitrary polynomial number of elements and still use the same hint size.

As explained in the introduction, our construction is slightly different, but similar in spirit, to the Packed Regev from [PVW08]. Just as in [PVW08], the idea is to individually encrypt the ℓ_2 different components of the message vector but reusing the *same* randomness $\mathbf{r}$ (but different parts of the secret key) for the components. In contrast to [PVW08], as we do not want the length of the randomness to grow with ℓ_2, to prove security of the scheme, we add an extra smudging noise term $\mathbf{e}'$ to each encrypted component.

The hint for an encrypted message is a short pre-image of the ciphertext hear $\mathbf{Ar}$, where $\mathbf{r}$ is the randomness of the encryption. To enable efficiently recovering this hint, we will generate the lattice given in the public key together with a standard lattice trapdoor that enables sampling random short pre-images as in [Ajt96, GPV08, AP09, MP12].

To satisfy the properties of density and weak circuit privacy, we will rely on a extra noisy Packed Regev encryption which proceeds just like the normal one but uses a much larger amount of randomness (so that it covers the whole set of strings for density, and so that it smudges the noises of evaluated ciphertexts for weak circuit privacy).

We will show how the Packed Regev encryption scheme satisfies our notion of a hintable packed LHE.

Theorem 4.5. *Assuming the LWE assumption, for all polynomials ℓ_1, there exists some polynomial h such that for all polynomials ℓ_2, there exists a secure (ℓ_1, ℓ_2, h)-hintable packed LHE.*

4.4 The Packed Regev Scheme

Our hintable Packed Regev scheme makes use of the following lattice trapdoor mechanism: Prior works [Ajt96, GPV08, AP09, MP12] show that there exist PPT algorithm $\mathsf{TrapGen}$ and $\mathsf{PreImSamp}$, an ensemble of efficiently sampleable distributions over $\mathbb{Z}$, $\{\mathcal{D}_\lambda\}_{\lambda\in\mathbb{N}}$, such that the following holds.

- $\underline{\mathsf{TrapGen}(1^\lambda, q, d)}$:

Given as input the security parameter $\lambda \in \mathbb{N}$, a modulus $q \in \mathbb{N}$, a dimension $d \in \mathbb{N}$, it outputs $(\mathbf{A}, T_{\mathbf{A}})$, where $\mathbf{A} \in \mathbb{Z}_q^{d\times m}$, $m \in \Theta(d\log(q))$ and $T_{\mathbf{A}}$ is a trapdoor. The matrix $\mathbf{A}$ is statistically close to uniform, that is, for all $\lambda, q, d \in \mathbb{N}$, the following distributions have statistical distance at most $2^{-\Omega(\lambda)}$: $\{(\mathbf{A}, T_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^\lambda, d, q) : \mathbf{A}\}$ and $\{\mathbf{A} \leftarrow_{\mathsf{R}} \mathbb{Z}_q^{\kappa\times m} : \mathbf{A}\}$.

- $\underline{\mathsf{PreImSamp}(\mathbf{A}, T_{\mathbf{A}}, \mathbf{t})}$:

Given as input the matrix $\mathbf{A}$, the trapdoor $T_{\mathbf{A}}$, a target vector $\mathbf{t} \in \mathbb{Z}_q^d$, it outputs $\mathbf{r} \in \mathbb{Z}_q^m$. For all $\lambda, d, q \in \mathbb{N}$, all $(\mathbf{A}, T_{\mathbf{A}})$ in the support of $\mathsf{TrapGen}(1^\lambda, d, q)$, all $\mathbf{t} \in \mathbb{Z}_q^d$, $\mathsf{PreImSamp}(\mathbf{A}, T_{\mathbf{A}}, \mathbf{t})$ outputs $\mathbf{r} \in \mathbb{Z}_q^d$ such that $\mathbf{Ar} = \mathbf{t} \in \mathbb{Z}_q^d$ and $\|\mathbf{r}\|_\infty < 2^{\lambda/2}$ with probability $1 - 2^{-\Omega(\lambda)}$ over its random coin.

We require the output $\mathbf{r}$ to follow some distribution that does not depend on the actual trapdoor $T_{\mathbf{A}}$, namely, for all λ, d, q, the following distributions have statistical distance at most $2^{-\Omega(\lambda)}$:

$$\{(\mathbf{A}, T_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^\lambda, d, q), \mathbf{r} \leftarrow_{\mathsf{R}} \mathcal{D}_\lambda^m, \mathbf{r}' \leftarrow_{\mathsf{R}} \mathsf{PreImSamp}(\mathbf{A}, T_{\mathbf{A}}, \mathbf{Ar}) : (\mathbf{r}', \mathbf{Ar})\}$$
$$\{(\mathbf{A}, T_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^\lambda, d, q), \mathbf{r} \leftarrow \mathcal{D}_\lambda^m : (\mathbf{r}, \mathbf{Ar})\}.$$

For our purposes, we want the distributions $\mathcal{D}_\lambda$ to be of smudging size. That is, for all polynomials $p(\cdot)$, all $\lambda, q \in \mathbb{N}$, $\gamma \in \mathbb{Z}_q$ such that $|\gamma| < p(\lambda)$, the following distributions have statistical distance at most $2^{-\Omega(\lambda)}$:

$$\{r \leftarrow_{\mathsf{R}} \mathcal{D}_\lambda : r \in \mathbb{Z}_q\}$$
$$\{r \leftarrow_{\mathsf{R}} \mathcal{D}_\lambda : r + \gamma \in \mathbb{Z}_q\}.$$

Finally, by the left over hash lemma, since m is large enough and $\mathcal{D}$ has enough entropy, for all $\lambda, d, q \in \mathbb{N}$, the following distributions have statistical distance at most $2^{-\Omega(\lambda)}$: $\{\mathbf{r} \leftarrow \mathcal{D}_\lambda^m, (\mathbf{A}, T_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^\lambda, d, q) : (\mathbf{A}, \mathbf{Ar})\}$ and $\{\mathbf{r} \leftarrow \mathcal{D}_\lambda^m, (\mathbf{A}, T_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^\lambda, d, q), \mathbf{u} \leftarrow_{\mathsf{R}} \mathbb{Z}_q^d : (\mathbf{A}, \mathbf{u})\}$.

We now proceed to describing the Packed Regev scheme, which is parameterized by polynomials ℓ_1 and ℓ_2. We denote the scheme by $\text{P-Regev}_{\ell_1,\ell_2}$.

- $\underline{\mathsf{Gen}(\mathsf{crs})}$:

Given as input $\mathsf{crs} = 1^\lambda$, it chooses polynomials $\kappa(\cdot)$, $B(\cdot)$, a B-bounded ensemble of efficiently sampleable distributions $\chi = \{\chi_\lambda\}_{\lambda\in\mathbb{N}}$ over $\mathbb{Z}$, a sequence $q = \{q_\lambda\}_{\lambda\in\mathbb{N}}$ where for each $\lambda \in \mathbb{N}$, q_λ has a polynomially bounded bit size, such that LWE holds with respect to q, κ, χ. The existence of such parameters is ensured by Theorem 3.5. Concretely, Gen takes $\kappa(\lambda) = (\ell_1(\lambda) + \lambda)^{1/c}$, $B(\lambda) = \kappa(\lambda)$, and $q_\lambda = 2^\lambda N_\lambda B(\lambda)$, where $N_\lambda = 2^{\ell_1(\lambda)}$ and $c \in (0, 1)$ is the constant from Theorem 3.5. For simplicity we write $q = q_\lambda$, $B = B(\lambda)$, $N = N_\lambda$, $\chi = \chi_\lambda$, and $\kappa = \kappa(\lambda)$. It sets $m = 2\kappa\log(q) + 2\lambda$. It samples $(\mathbf{A}, T_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^\lambda, q, \kappa)$, $\mathbf{S} \leftarrow_{\mathsf{R}} \mathbb{Z}_q^{\ell_2\times\kappa}$, $\mathbf{E} \leftarrow \chi^{\ell_2\times m}$, and sets $\mathsf{pk} = (N, \mathbf{A}, \mathbf{SA} + \mathbf{E}) \in \mathbb{N} \times \mathbb{Z}_q^{\kappa\times m} \times \mathbb{Z}_q^{\ell_2\times m}$, $\mathsf{sk} = (\mathbf{S}, T_{\mathbf{A}}, \mathsf{pk})$. It outputs $(\mathsf{pk}, \mathsf{sk})$.

- $\underline{\mathsf{Enc}_{\mathsf{pk}}(\mathbf{x} \in \mathbb{Z}_N^{\nu})}$:
Given the public pk, a vector $\mathbf{x} \in \mathbb{Z}_N^{\nu}$, it samples $\mathbf{R} \leftarrow [-1,1]^{m\times\nu\ell_2}$, $\mathbf{E}' \leftarrow_{\mathsf{R}} [-2^{\lambda/2}, 2^{\lambda/2}]^{\ell_2\times\nu\ell_2}$ and outputs the ciphertext $\mathsf{ct} = \Big(\mathbf{AR}, (\mathbf{SA}+\mathbf{E})\mathbf{R} + \mathbf{E}' + q/N\cdot\mathbf{x}^\top \otimes \mathrm{Id}_{\ell_2}\Big) \in \mathbb{Z}_q^{(\kappa+\ell_2)\times\nu\ell_2}$, where $\mathrm{Id}_{\ell_2} \in \mathbb{Z}_N^{\ell_2\times\ell_2}$ denotes the identity matrix, and $\mathbf{x}^\top \otimes \mathrm{Id}_{\ell_2} \in \mathbb{Z}_N^{\ell_2\times\nu\ell_2}$.

- $\underline{\mathsf{Enc}^\star_{\mathsf{pk}}(\mathbf{m} \in \mathbb{Z}_N^{\ell_2})}$:
Given the public pk, a message $\mathbf{m} \in \mathbb{Z}_q^{\ell_2}$, it samples $\mathbf{r}^\star \leftarrow_{\mathsf{R}} \mathcal{D}^m$, where $\mathcal{D}$ is the efficiently sampleable distribution over $\mathbb{Z}$ related to $\mathsf{TrapGen}$ and $\mathsf{PreImSamp}$; $\mathbf{e}^\star \leftarrow_{\mathsf{R}} \big(-\frac{q}{2N}, \frac{q}{2N}\big]^{\ell_2}$, and outputs the noisy ciphertext $\mathsf{ct}^\star = \Big(\mathbf{Ar}^\star, (\mathbf{SA}+\mathbf{E})\mathbf{r}^\star + \mathbf{e}^\star + q/N\cdot\mathbf{m}\Big) \in \mathbb{Z}_q^{\kappa+\ell_2}$.

- $\underline{\mathsf{Eval}(\mathsf{pk}, \mathsf{ct}, \mathsf{ct}^\star, \mathbf{y})}$:
Given the public pk, ciphertext $\mathsf{ct} \in \mathbb{Z}_q^{(\kappa+\ell_2)\times\nu\ell_2}$, noisy ciphertext $\mathsf{ct}^\star \in \mathbb{Z}_q^{\kappa+\ell_2}$ and a vector $\mathbf{y} \in [-1,1]^{\nu\ell_2}$, it outputs the evaluated ciphertext $\mathsf{ct}\mathbf{y} + \mathsf{ct}^\star \in \mathbb{Z}_q^{\kappa+\ell_2}$.

- $\underline{\mathsf{SecHint}(\mathsf{sk}, \mathsf{ct}^\star)}$:
Given as input the secret key sk and a (noisy or evaluated) ciphertext $\mathsf{ct}^\star \in \mathbb{Z}_q^{\kappa+\ell_2}$ of the form $(\mathbf{t}, \mathbf{z})$ where $\mathbf{t} \in \mathbb{Z}_q^{\kappa}$ and $\mathbf{z} \in \mathbb{Z}_q^{\ell_2}$, it samples $\rho \leftarrow \mathsf{PreImSamp}(\mathbf{A}, T_{\mathbf{A}}, \mathbf{t})$ and outputs the hint $\rho \in \mathbb{Z}_q^m$.

- $\underline{\mathsf{PubHint}(\mathsf{pk}, r)}$:
Given as input the public key pk and the random coins $r = (\mathbf{r}^\star, \mathbf{e}^\star)$ where $\mathbf{r}^\star \in \mathcal{D}_\lambda^m$, $\mathbf{e}^\star \in \big(-\frac{q}{2N}, \frac{q}{2N}\big]^{\ell_2}$ used to produce a noisy ciphertext, it outputs $\mathbf{r}^\star \in \mathbb{Z}_q^m$.

- $\underline{\mathsf{Rec}(\mathsf{pk}, \mathsf{ct}^\star, \rho)}$:
Given as input the public key pk, a (noisy or evaluated) ciphertext $\mathsf{ct}^\star \in \mathbb{Z}_q^{\kappa+\ell_2}$ of the form $\mathsf{ct} = (\mathbf{t}, \mathbf{z})$ where $\mathbf{t} \in \mathbb{Z}_q^{\kappa}$, $\mathbf{z} \in \mathbb{Z}_q^{\ell_2}$ and a hint $\rho \in \mathbb{Z}_q^m$, it computes $\mathbf{d} = \mathbf{z} - (\mathbf{SA}+\mathbf{E})\rho \in \mathbb{Z}_q^{\ell_2}$ and outputs $\lfloor N/q\cdot\mathbf{d}\rceil \in \mathbb{Z}_N^{\ell_2}$.

- $\underline{\mathsf{Dec}_{\mathsf{sk}}(\mathsf{ct}^\star)}$:
Given as input the secret key sk and a (noisy or evaluated) ciphertext $\mathsf{ct}^\star$, it runs $\mathsf{SecHint}(\mathsf{sk}, \mathsf{ct}^\star)$ to obtain the hint ρ, and outputs $\mathsf{Rec}(\mathsf{pk}, \mathsf{ct}^\star, \rho)$.

The proof of Theorem 4.5 follows from the propositions and theorem below (which demonstrate that Packed Regev Scheme satisfies the desired properties of a hintable packed LHE, as well as security).

Proposition 14 (Linear Homomorphism)**.** *The LHE presented above satisfies Property 4.2 (linear homomorphism).*

Proof: The ciphertext produced by $\mathsf{Enc}_{\mathsf{pk}}(\mathbf{x})$ has the form $\mathsf{ct} = \big(\mathbf{AR}, (\mathbf{SA}+\mathbf{E})\,\mathbf{R} + \mathbf{E}' + q/N\cdot\mathbf{x}^\top \otimes \mathrm{Id}_{\ell_2}\big) \in \mathbb{Z}_q^{(\kappa+\ell_2)\times\nu\ell_2}$, and $\mathsf{ct}^\star = (\mathbf{Ar}^\star, (\mathbf{SA}+\mathbf{E})\,\mathbf{r}^\star + \mathbf{e}^\star + \mathbf{x}^\star)$. For all $\mathbf{y} \in [-1,1]^{\nu\ell_2}$, $\mathsf{Eval}(\mathsf{pk}, \mathsf{ct}, \mathsf{ct}^\star, \mathbf{y})$ outputs the evaluated ciphertext $\mathsf{ct}_{\mathbf{y}} = \Big(\mathbf{A}(\mathbf{Ry}+\mathbf{r}^\star), (\mathbf{SA}+\mathbf{E})(\mathbf{Ry}+\mathbf{r}^\star) + \mathbf{E}'\mathbf{y} + \mathbf{e}^\star + q/N\cdot\mu\Big) \in \mathbb{Z}_q^{\kappa+\ell_2}$ where $\mu = (m_1 + \mathbf{x}^\top\mathbf{y}_1, \ldots, m_{\ell_2} + \mathbf{x}^\top\mathbf{y}_{\ell_2}) \in \mathbb{Z}_N^{\ell_2}$ which is in the support of $\mathsf{Enc}^\star_{\mathsf{pk}}(\mu)$. □

Proposition 15 (Correctness of secret hints)**.** *The LHE presented above satisfies Property 4.3 (correctness of secret hints).*

Proof: For all messages $\mathbf{m} \in \mathbb{Z}_N^{\ell_2}$, $\mathsf{Enc}^\star\mathsf{pk}(\mathbf{m})$ is of the form $\mathsf{ct}^\star = (\mathbf{t}, \mathbf{z}) \in \mathbb{Z}_N^{\kappa \times \ell_2}$, where $\mathbf{t} = \mathbf{A}\mathbf{r}^\star$ and $\mathbf{z} = (\mathbf{SA} + \mathbf{E})\,\mathbf{r}^\star + \mathbf{e}^\star + \mathbf{x}^\star)$.

The algorithm $\mathsf{SecHint}$ computes $\rho \leftarrow \mathsf{PreImSamp}(\mathbf{A}, T_{\mathbf{A}}, \mathbf{t})$. By the properties of $\mathsf{PreImSamp}$, $\rho \in \mathbb{Z}_q^m$ is such that $\|\rho\|_\infty \leq 2^{\lambda/2}$ with all but $2^{-\Omega(\lambda)}$ probability over the random coins of $\mathsf{PreImSamp}$, and $\mathbf{A}\rho = \mathbf{t}$.

Next, the algorithm $\mathsf{SecHint}$ computes $\mathbf{d} = \mathbf{z} - (\mathbf{SA} + \mathbf{E})\,\rho = q/N \cdot \mathbf{m} + \mathsf{noise} + \mathbf{e}^\star$, where $\mathsf{noise} = \mathbf{E}(\mathbf{r}^\star - \rho)$. With probability $1 - 2^{-\Omega(\lambda)}$ over the choice of $\mathbf{E} \leftarrow \chi^{\ell_2 \times m}$, $\mathbf{r}^\star \leftarrow \mathcal{D}_\lambda^m$ and the random coins used to produce ρ, we have $\|\mathsf{noise}\|_\infty \leq 2^{1+\lambda/2}Bm \in q/N \cdot 2^{-\Omega(\lambda)}$. Thus, with probability $1 - 2^{-\Omega(\lambda)}$ over the choice of $\mathbf{e}^\star \leftarrow_{\mathsf{R}} \left(-\frac{q}{2N}, \frac{q}{2N}\right]^{\ell_2}$, we have $\lfloor N/q(\mathsf{noise} + \mathbf{e}^\star)\rceil = 0^{\ell_2} \in \mathbb{Z}_N^{\ell_2}$. Consequently, we have $\lfloor N/q \cdot \mathbf{d}\rceil = \mu \in \mathbb{Z}_N^{\ell_2}$. □

Proposition 16 (Correctness of public hints)**.** *The LHE presented above satisfies Property 4.4 (correctness of public hints).*

Proof: We aim at proving that for all $\lambda \in \mathbb{N}$, all messages $\mathbf{m} \in \mathbb{Z}_N^{\ell_2}$, the following distributions have statistical distance at most $2^{-\Omega(\lambda)}$:

$$\mathcal{D}_0 = \left\{ \begin{array}{l} \big(\mathsf{pk} = (\mathbf{A}, \mathbf{SA} + \mathbf{E}), \mathsf{sk} = (\mathbf{S}, T_{\mathbf{A}})\big) \leftarrow \mathsf{Gen}(1^\lambda), \mathbf{r}^\star \leftarrow \mathcal{D}_\lambda^m, \mathbf{e}^\star \leftarrow_{\mathsf{R}} (-\frac{q}{2N}, \frac{q}{2N}]^{\ell_2} \\ \rho \leftarrow \mathsf{PreImSamp}(\mathbf{A}, T_{\mathbf{A}}, \mathbf{A}\mathbf{r}^\star) : (\mathbf{A}\mathbf{r}^\star, (\mathbf{SA} + \mathbf{E})\mathbf{r}^\star + \mathbf{e}^\star + q/N \cdot \mathbf{m}, \rho) \end{array} \right\}$$

$$\mathcal{D}_1 = \left\{ \begin{array}{l} \big(\mathsf{pk} = (\mathbf{A}, \mathbf{SA} + \mathbf{E}), \mathsf{sk} = (\mathbf{S}, T_{\mathbf{A}})\big) \leftarrow \mathsf{Gen}(1^\lambda), \mathbf{r}^\star \leftarrow \mathcal{D}_\lambda^m \\ \mathbf{e}^\star \leftarrow_{\mathsf{R}} (-\frac{q}{2N}, \frac{q}{2N}]^{\ell_2} : (\mathbf{A}\mathbf{r}^\star, (\mathbf{SA} + \mathbf{E})\mathbf{r}^\star + \mathbf{e}^\star + q/N \cdot \mathbf{m}, \mathbf{r}^\star) \end{array} \right\}$$

By Lemma 2.2 (smudging), distribution $\mathcal{D}_0$ has statistical distance at most $2^{-\Omega(\lambda)}$ with $\mathcal{D}_0' = \{(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda), \mathbf{r}^\star \leftarrow \mathcal{D}_\lambda^m, \mathbf{e}^\star \leftarrow_{\mathsf{R}} \left(-\frac{q}{2N}, \frac{q}{2N}\right]^{\ell_2}, \rho \leftarrow \mathsf{PreImSamp}(\mathbf{A}, T_{\mathbf{A}}, \mathbf{A}\mathbf{r}^\star) : \big(\mathbf{A}\mathbf{r}^\star, \mathbf{SA}\mathbf{r}^\star + \mathbf{e}^\star + q/N \cdot \mathbf{m}, \rho\big)\}$. By the property of $\mathsf{PreImSamp}$, $\mathcal{D}_0'$ has statistical distance at most $2^{-\Omega(\lambda)}$ with $\mathcal{D}_1' = \{(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda), \mathbf{r}^\star \leftarrow \mathcal{D}_\lambda^m, \mathbf{e}^\star \leftarrow_{\mathsf{R}} \left(-\frac{q}{2N}, \frac{q}{2N}\right]^{\ell_2} : \big(\mathbf{A}\mathbf{r}^\star, \mathbf{SA}\mathbf{r}^\star + \mathbf{e}^\star + q/N \cdot \mathbf{m}, \mathbf{r}^\star\big)\}$. Finally, using smudging again, we have $\mathcal{D}_1' \approx_s \mathcal{D}_1$, with statistical distance at most $2^{-\Omega(\lambda)}$. □

Proposition 17 (Correctness)**.** *The LHE presented above satisfies Property 4.1 (correctness).*

Proof: Correctness follows from the correctness of the secret hints (Property 4.3), since $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}^\star)$ simply applies $\rho \leftarrow \mathsf{SecHint}(\mathsf{sk}, \mathsf{ct}^\star)$ to compute a hint, then returns $\mathsf{Rec}(\mathsf{pk}, \mathsf{ct}^\star, \rho)$. □

Proposition 18 (Succinctness of hints)**.** *The LHE presented above satisfies $h(\lambda) = (\lambda/2 + 1)m$ succinctness, where $m = 2\kappa \log(q) + 2\lambda$.*

Proof: Hints are of the form $\mathbf{r}^\star \in [-2^{\lambda/2}, 2^{\lambda/2}]^m$. □

Proposition 19 (Density of the noisy ciphertexts)**.** *The LHE presented above satisfies Property 4.7 (density of the noisy ciphertexts).*

Proof: For all $\mathbf{m} \in \mathbb{Z}_q^{\ell_2}$, a noisy ciphertext $\mathsf{ct}^\star \leftarrow_{\mathsf{R}} \mathsf{Enc}_{\mathsf{pk}}^\star(\mathbf{m})$ is of the form $\mathsf{ct}^\star = (\mathbf{A}\mathbf{r}^\star, (\mathbf{SA} + \mathbf{E})\mathbf{r}^\star + \mathbf{e}^\star + q/N \cdot \mathbf{m}) \in \mathbb{Z}_q^{\kappa + \ell_2}$. When $\mathbf{m} \leftarrow_{\mathsf{R}} \mathbb{Z}_N^{\ell_2}$, the second part of the ciphertext is uniformly random over $\mathbb{Z}_q^{\ell_2}$, since $\mathbf{e}^\star \leftarrow_{\mathsf{R}} \left(-\frac{q}{2N}, \frac{q}{2N}\right]^{\ell_2}$. That is, for all $\lambda \in \mathbb{N}$, for all pairs $(\mathsf{pk}, \mathsf{sk})$ in the support of $\mathsf{Gen}(1^\lambda)$, the following distributions are identical: $\{\mathbf{m} \leftarrow_{\mathsf{R}} \mathbb{Z}_N^{\ell_2}, \mathsf{ct}^\star \leftarrow \mathsf{Enc}_{\mathsf{pk}}^\star(\mathbf{m}) : \mathsf{ct}^\star\}$ and $\{\mathbf{r}^\star \leftarrow_{\mathsf{R}} \mathcal{D}_\lambda^m, \mathbf{w} \leftarrow_{\mathsf{R}} \mathbb{Z}_q^{\ell_2} : (\mathbf{A}\mathbf{r}^\star, \mathbf{w})\}$. We conclude the proof using the properties of $\mathsf{PreImSamp}$, which imply that the following distributions have statistical distance at most $2^{-\Omega(\lambda)}$: $\{\mathbf{A} \leftarrow_{\mathsf{R}} \mathbb{Z}_q^{\kappa \times m}, \mathbf{r}^\star \leftarrow_{\mathsf{R}} \mathcal{D}_\lambda^m : (\mathbf{A}, \mathbf{A}\mathbf{r}^\star)\}$ and $\{\mathbf{A} \leftarrow_{\mathsf{R}} \mathbb{Z}_q^{\kappa \times m}, \mathbf{u} \leftarrow_{\mathsf{R}} \mathbb{Z}_q^\kappa : (\mathbf{A}, \mathbf{u})\}$. □

Proposition 20 (Weak circuit privacy)**.** *The LHE presented above satisfies Property 4.6 (weak circuit privacy).*

Proof: For all $\mathbf{x} \in \mathbb{Z}_N^{\nu}$, $\mathbf{y} \in [-1,1]^{\nu\ell_2}$, we aim at proving the following distributions have statistical distance at most $2^{-\Omega(\lambda)}$:

$$\mathcal{D}_0 = \left\{ \; (\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda), \mathsf{ct} \leftarrow \mathsf{Enc}_{\mathsf{pk}}(\mathbf{x}), \mathsf{ct}^\star \leftarrow \mathsf{Enc}^\star_{\mathsf{pk}}(\mathbf{m}), \mathsf{ct}_{\mathbf{y}} = \mathsf{Eval}(\mathsf{pk}, \mathsf{ct}, \mathsf{ct}^\star, \mathbf{y}) : (\mathsf{ct}, \mathsf{ct}^\star, \mathsf{ct}_{\mathbf{y}}) \; \right\}$$

$$\mathcal{D}_1 = \left\{ \begin{array}{l} (\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda), \mathsf{ct} \leftarrow \mathsf{Enc}_{\mathsf{pk}}(\mathbf{x}) \\ \mathsf{ct}_{\mathbf{y}} \leftarrow \mathsf{Enc}^\star_{\mathsf{pk}}(\mu), \mathsf{ct}^\star = \mathsf{Eval}(\mathsf{pk}, \mathsf{ct}, \mathsf{ct}_{\mathbf{y}}, -\mathbf{y}) : (\mathsf{ct}, \mathsf{ct}^\star, \mathsf{ct}_{\mathbf{y}}) \end{array} \right\},$$

where $\mu = (m_1 + \mathbf{x}^\top \mathbf{y}_1, \ldots, m_{\ell_2} + \mathbf{x}^\top \mathbf{y}_{\ell_2}) \in \mathbb{Z}_N^{\ell_2}$.

In distribution $\mathcal{D}_0$, we have:

- $\mathsf{ct} = \big(\mathbf{AR}, (\mathbf{SA}+\mathbf{E})\,\mathbf{R} + \mathbf{E}' + q/N \cdot \mathbf{x}^\top \otimes \mathrm{Id}_{\ell_2}\big)$,
- $\mathsf{ct}^\star = (\mathbf{A}\mathbf{r}^\star, (\mathbf{SA}+\mathbf{E})\,\mathbf{r}^\star + \mathbf{e}^\star + q/N \cdot \mathbf{x}^\star)$,
- $\mathsf{ct}_{\mathbf{y}} = (\mathbf{A}(\mathbf{Ry} + \mathbf{r}^\star), (\mathbf{SA}+\mathbf{E})\,(\mathbf{Ry}+\mathbf{r}^\star) + \mathbf{E}'\mathbf{y} + \mathbf{e}^\star + q/N \cdot \mu)$.

We show that it has statistical distance $2^{-\Omega(\lambda)}$ from $\mathcal{D}_1$ by a series of claims.

Claim 1. For all $\mathbf{y} \in [-1,1]^{\nu\ell_2}$, the following distributions have statistical distance $2^{-\Omega(\lambda)}$:

$$\Big\{\mathbf{r}^\star \leftarrow_{\mathsf{R}} \mathcal{D}_\lambda^m, \mathbf{R} \leftarrow_{\mathsf{R}} [-1,1]^{m\times\nu\ell_2} : (\mathbf{R}, \mathbf{r}^\star, \mathbf{r}^\star + \mathbf{Ry})\Big\}$$
$$\Big\{\mathbf{r}^\star \leftarrow_{\mathsf{R}} \mathcal{D}_\lambda^m, \mathbf{R} \leftarrow_{\mathsf{R}} [-1,1]^{m\times\nu\ell_2} : (\mathbf{R}, \mathbf{r}^\star - \mathbf{Ry}, \mathbf{r}^\star)\Big\},$$

by the properties of $\mathsf{PreImSamp}$.

Claim 2. For all $\mathbf{y} \in [-1,1]^{\nu\ell_2}$, the following distributions have statistical distance $2^{-\Omega(\lambda)}$:

$$\Big\{\mathbf{e}^\star \leftarrow_{\mathsf{R}} \big(-\frac{q}{2N}, \frac{q}{2N}\big]^{\ell_2}, \mathbf{E}' \leftarrow [-2^{\lambda/2}, 2^{\lambda/2}]^{\ell_2\times\nu\ell_2} : (\mathbf{E}', \mathbf{e}^\star, \mathbf{e}^\star + \mathbf{E}'\mathbf{y})\Big\}$$
$$\Big\{\mathbf{e}^\star \leftarrow_{\mathsf{R}} \big(-\frac{q}{2N}, \frac{q}{2N}\big]^{\ell_2}, \mathbf{E}' \leftarrow [-2^{\lambda/2}, 2^{\lambda/2}]^{\ell_2\times\nu\ell_2} : (\mathbf{E}', \mathbf{e}^\star - \mathbf{E}'\mathbf{y}, \mathbf{e}^\star)\Big\}.$$

by Lemma 2.2 (smudging).

Claim 1 and Claim 2 imply the following claim.

Claim 3. For all $\mathbf{y} \in [-1,1]^{\nu\ell_2}$, the following distributions have statistical distance $2^{-\Omega(\lambda)}$:

$$\mathcal{D}'_0 = \left\{ \begin{array}{l} \mathbf{e}^\star \leftarrow_{\mathsf{R}} \big(-\frac{q}{2N}, \frac{q}{2N}\big]^{\ell_2}, \mathbf{E}' \leftarrow [-2^{\lambda/2}, 2^{\lambda/2}]^{\ell_2\times\nu\ell_2}, \mathbf{r}^\star \leftarrow_{\mathsf{R}} \mathcal{D}_\lambda^m \\ \mathbf{R} \leftarrow_{\mathsf{R}} [-1,1]^{m\times\nu\ell_2} : \Big(\mathbf{R}, \mathbf{r}^\star, \mathbf{r}^\star + \mathbf{Ry}, \mathbf{E}', \mathbf{e}^\star, \mathbf{e}^\star + \mathbf{E}'\mathbf{y}\Big) \end{array} \right\}$$

$$\mathcal{D}'_1 = \left\{ \begin{array}{l} \mathbf{e}^\star \leftarrow_{\mathsf{R}} \big(-\frac{q}{2N}, \frac{q}{2N}\big]^{\ell_2}, \mathbf{E}' \leftarrow [-2^{\lambda/2}, 2^{\lambda/2}]^{\ell_2\times\nu\ell_2}, \mathbf{r}^\star \leftarrow_{\mathsf{R}} \mathcal{D}_\lambda^m \\ \mathbf{R} \leftarrow_{\mathsf{R}} [-1,1]^{m\times\nu\ell_2} : \Big(\mathbf{R}, \mathbf{r}^\star - \mathbf{Ry}, \mathbf{r}^\star, \mathbf{E}', \mathbf{e}^\star - \mathbf{E}'\mathbf{y}, \mathbf{e}^\star\Big) \end{array} \right\}.$$

Now we prove the following claim, that will conclude the proof of Proposition 20.

Claim 4. There exists a (possibly inefficient) simulator $\mathcal{S}$ that given as input $\mathbf{v} \in \left(\mathbb{Z}_q^{m\times(\nu\ell_2+2)} \times \mathbb{Z}_q^{\ell_2\times(\nu\ell_2+2)}\right)$, outputs a tuple $(\mathsf{ct}, \mathsf{ct}^\star, \mathsf{ct}_\mathbf{y}) \in \mathbb{Z}_q^{(\kappa+\ell_2)\times\nu\ell_2} \times \left(\mathbb{Z}_q^{\kappa+\ell_2}\right)^2$, such that when $\mathcal{S}$ is fed with an input from distribution $\mathcal{D}'_0$, it produces an output following the distribution $\mathcal{D}_0$, whereas when fed with an input coming from distribution $\mathcal{D}'_1$, it produces an output following the distribution $\mathcal{D}_1$.

Given as input $\left(\mathbf{R}, \mathbf{r}_1, \mathbf{r}_2, \mathbf{E}', \mathbf{e}_1, \mathbf{e}_2\right)$, $\mathcal{S}$ computes:

- $\mathsf{ct} = \left(\mathbf{AR}, (\mathbf{SA}+\mathbf{E})\,\mathbf{R} + \mathbf{E}' + q/N \cdot \mathbf{x}^\top \otimes \mathrm{Id}_{\ell_2}\right)$,
- $\mathsf{ct}^\star = (\mathbf{A}\mathbf{r}_1, (\mathbf{SA}+\mathbf{E})\,\mathbf{r}_1 + \mathbf{e}_1 + q/N \cdot \mathbf{m})$,
- $\mathsf{ct}_\mathbf{y} = (\mathbf{A}\mathbf{r}_2, (\mathbf{SA}+\mathbf{E})\,\mathbf{r}_2 + \mathbf{e}_2 + q/N \cdot \mu)$.

$\square$

Theorem 4.6 (Security)**.** *The LHE presented is semantically secure under the LWE assumption.*

Proof: In a nutshell, Regev encryption uses the LWE assumption to switch the LWE samples from the public key $\mathbf{SA}+\mathbf{E}$ to uniformly random, then use the left over hash lemma to extract the entropy from the randomness used to encrypt the messages. The problem here is that for succinctness, we use small randomness, and large messages: the randomness does not hold enough entropy to hide the messages. Instead, we use the randomness and extra smudging noise to generate fresh LWE samples, and then hide the messages.

We now provide the formal proof. We use the following hybrid games, for all $b \in \{0,1\}$.

• $\underline{\mathcal{D}^b_\lambda}$: as per Definition 2.9. Namely, this game generates $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$ where $\mathsf{pk} = (\mathbf{A}, \mathbf{SA}+\mathbf{E})$, $\mathsf{ct} = (\mathbf{Ar}, (\mathbf{SA}+\mathbf{E})\mathbf{r} + \mathbf{e}' + q/N \cdot \mathbf{m}^b) \in \mathbb{Z}_q^{\kappa+\ell_2}$. It returns $(\mathsf{pk}, \mathsf{ct})$ to the adversary.

• $\underline{\mathcal{H}^{b.1}_\lambda}$: this game generates $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$ where $\mathsf{pk} = (\mathbf{A}, \mathbf{SA}+\mathbf{E})$, $\mathsf{ct} = (\mathbf{Ar}, \mathbf{SAr} + \mathbf{e}' + q/N \cdot \mathbf{m}^b)$. Recall that $\mathbf{E}$ is polynomially bounded, $\mathbf{r} \leftarrow_{\mathsf{R}} [-1,1]^m$ for a polynomial m and $\mathbf{e}' \leftarrow_{\mathsf{R}} [-2^{\lambda/2}, 2^{\lambda/2}]^{\ell_2}$. Thus, we can use Lemma 2.2 (smudging) to argue that $(\mathbf{E}, \mathbf{r}, \mathbf{e}' + \mathbf{Er}) \approx_s (\mathbf{E}, \mathbf{r}, \mathbf{e}')$ with statistical distance $2^{-\Omega(\lambda)}$. The first distribution corresponds to $\mathcal{D}^b_\lambda$, whereas the second distribution corresponds to $\mathcal{H}^{b.1}_\lambda$.

• $\underline{\mathcal{H}^{b.2}_\lambda}$: this game generates $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$ where $\mathsf{pk} = (\mathbf{A}, \mathbf{SA}+\mathbf{E})$, $\mathsf{ct} = (\mathbf{u}, \mathbf{Su} + \mathbf{e}' + q/N \cdot \mathbf{m}^b)$, $\mathbf{u} \leftarrow_{\mathsf{R}} \mathbb{Z}_q^\kappa$. The fact that $\mathcal{H}^{b.1}_\lambda \approx_s \mathcal{H}^{b.2}_\lambda$ with statistical distance $2^{-\Omega(\lambda)}$ follows readily from the left over hash lemma, which states that the following distributions have statistical distance $2^{-\Omega(\lambda)}$:

$$\{\mathbf{A} \leftarrow_{\mathsf{R}} \mathbb{Z}_q^{\kappa\times m}, \mathbf{r} \leftarrow_{\mathsf{R}} [-1,1]^m : (\mathbf{A}, \mathbf{Ar})\}$$
$$\{\mathbf{A} \leftarrow_{\mathsf{R}} \mathbb{Z}_q^{\kappa\times m}, \mathbf{u} \leftarrow_{\mathsf{R}} \mathbb{Z}_q^\kappa : (\mathbf{A}, \mathbf{u})\}.$$

Now we prove that $\mathcal{H}^{0.2}_\lambda \approx_c \mathcal{H}^{1.2}_\lambda$. This holds by the properties of $\mathsf{TrapGen}$, which states that $\mathbf{A}$ is statistically close to uniform over $\mathbb{Z}_q^{\kappa\times m}$. Then, we rely on the LWE assumption, which implies that $(\mathbf{A}, \mathbf{SA}+\mathbf{E}, \mathbf{u}, \mathbf{Su}+\mathbf{e}') \approx_c (\mathbf{A}, \mathbf{V}, \mathbf{u}, \mathbf{w}) \approx (\mathbf{A}, \mathbf{SA}+\mathbf{E}, \mathbf{u}, \mathbf{w})$ where $\mathbf{w} \leftarrow_{\mathsf{R}} \mathbb{Z}_q^{\ell_2}$, and $\mathbf{V} \leftarrow_{\mathsf{R}} \mathbb{Z}_q^{\ell_2\times m}$.

Summarizing, we have shown that: $\mathcal{D}^0_\lambda \approx_s \mathcal{H}^{0.1}_\lambda \approx_s \mathcal{H}^{0.2}_\lambda \approx_c \mathcal{H}^{1.2}_\lambda \approx_s \mathcal{H}^{1.1}_\lambda \approx_s \mathcal{D}^1_\lambda$. $\square$

5 Constructing XiO for $\mathsf{P}^{\log}/\mathsf{poly}$

We present a modular construction of XiO $\mathsf{P}^{\log}/\mathsf{poly}$ from the GSW FHE scheme for circuits of depth δ, denoted by GSW_δ, for sufficiently sufficiently large δ, and any (ℓ_1, ℓ_2, h)-hintable packed LHE for sufficiently small h and sufficiently large ℓ_1 and ℓ_2 —recall that h measures the LHE succinctness, while ℓ_1, ℓ_2 measure the plaintext size, or "batching capacity" of the scheme; ℓ_1 intuitively represents how many bits can be packed in a scalar, i.e. how large is the modulus in use, whereas ℓ_2 measures how many such scalars are recovered when decrypting one ciphertext. We now state our formal theorem.

Theorem 5.1. *Assume that for all polynomials δ, b, all constants $\varepsilon \in (0,1)$, there exists a polynomial h s.t. for all polynomials n, there exist ℓ_1, ℓ_2 and an (ℓ_1, ℓ_2, h)-hintable packed LHE denoted by $\mathcal{LHE}_{b,\varepsilon,n}$ s.t. for all $\lambda \in \mathbb{N}$:*

- $\ell_1(\lambda) \geq b(\lambda)$
- $\big(\ell_1(\lambda) - b(\lambda)\big) \cdot \ell_2(\lambda) > n(\lambda)^{\varepsilon}$
- *$\mathcal{LHE}_{b,\varepsilon,n}$ and the GSW FHE scheme for depth $\delta(\lambda)$ circuits are circularly SRL-secure.*

Then XiO for $\mathsf{P}^{\log}/\mathsf{poly}$ exists.

To enable a modular proof, we abstract out 2 additional properties of the GSW FHE that we will rely on.

5.1 Additional properties for GSW

Weak Circuit Privacy of GSW As mentionned in the introduction, we will rely on the fact that the GSW encryption scheme also satisfies a notion of "weak circuit privacy" similar to the one defined for LHE. More precisely, we show that GSW satisfies a property that involves a public-key algorithm that re-randomizes evaluated ciphertext so that they look like fresh ciphertexts from the support of the noise encryption algorithm $\mathsf{Enc}^\star$. Namely, we show that there exists a PPT algorithm ReRand that takes as input the public key pk, an evaluated ciphertext ct, some random coins $\mathbf{r}^\star \in \mathcal{R}^\star$, and outputs an evaluated ciphertext ct computed as described below.

• $\underline{\mathsf{ReRand}(\mathsf{pk}, \mathsf{ct}; \mathbf{r}^\star)}$:
Given $\mathsf{pk} = (B, \mathbf{U}, \mathbf{G})$, $\mathsf{ct} \in \{0,1\}^w$ and $\mathbf{r}^\star \leftarrow_{\mathsf{R}} [-\widetilde{B}2^\lambda, \widetilde{B}2^\lambda]^m$, it computes $\mathsf{ct}' \in \mathbb{Z}_N^{\kappa+1}$ whose binary decomposition is ct, computes $\widetilde{\mathsf{ct}} = \mathsf{ct}' + \mathbf{U}\mathbf{r}^\star \in \mathbb{Z}_N^{\kappa+1}$, and outputs the re-randomized ciphertext $\mathsf{BD}(\widetilde{\mathsf{ct}}) \in \{0,1\}^w$.

Theorem 5.2 (weak circuit privacy). *For all polynomials ν, q, all $\lambda \in \mathbb{N}$, all crs containing a modulus N of $\theta \in \Omega(\lambda \cdot d(\lambda))$ bits, all pairs $(\mathsf{pk}, \mathsf{sk})$ in the support of $\mathsf{Gen}(\mathsf{crs})$, all messages $m_1, \ldots, m_{\nu(\lambda)}$, all depth-$d$ circuits $f_1, \ldots, f_q : \{0,1\}^\nu \to \mathbb{Z}_N$, the following distributions have statistical distance at most $2^{-\lambda}$:*

$$\mathcal{D}_0 : \left\{ \begin{array}{l} \forall i \in [\nu], \mathbf{R}_i \leftarrow_{\mathsf{R}} [-1,1]^{m \times w}, \mathsf{ct}_i \leftarrow \mathsf{Enc}_{\mathsf{pk}}(m_i; \mathbf{R}_i) \\ \forall j \in [q], \mathsf{ct}_{f_j} = \mathsf{Eval}'(\mathsf{pk}, f_j, 1, \mathsf{ct}_1, \ldots, \mathsf{ct}_\nu), \mathbf{r}_j^\star \leftarrow_{\mathsf{R}} \mathcal{R}^\star \\ \mathsf{ct}_{f_j}^\star = \mathsf{ReRand}(\mathsf{pk}, \mathsf{ct}_{f_j}; \mathbf{r}^\star) : \Big((\mathsf{ct}_i)_{i \in [\nu]}, (\mathbf{r}_j^\star, \mathsf{ct}_{f_j}^\star)_{j \in [q]} \Big) \end{array} \right\}$$

$$\mathcal{D}_1 : \left\{ \begin{array}{l} \forall i \in [\nu] : \mathbf{R}_i \leftarrow_{\mathsf{R}} [-1,1]^{m \times w}, \mathsf{ct}_i \leftarrow \mathsf{Enc}_{\mathsf{pk}}(m_i; \mathbf{R}_i) \\ \forall j \in [q], \mathbf{r}_j^\star \leftarrow_{\mathsf{R}} \mathcal{R}^\star, \mathsf{ct}_{f_j}^\star = \mathsf{Enc}_{\mathsf{pk}}^\star(f_j(\mathbf{m}); \mathbf{r}^\star) \\ \mathbf{r}_{f_j} = \mathsf{Eval}_{\mathsf{rand}}\left(\mathsf{pk}, f_j, (\mathbf{R}_i)_{i \in [\nu]}, (m_i)_{i \in [\nu]}\right) : \Big((\mathsf{ct}_i)_{i \in [\nu]}, (\mathbf{r}_j^\star - \mathbf{r}_{f_j}, \mathsf{ct}_{f_j}^\star)_{j \in [q]} \Big) \end{array} \right\}$$

Proof: By batch correctness of the scheme, for all $j \in [q]$, the evaluated ciphertext is of the form $\mathsf{ct}_{f_j} = \big(\mathbf{A}\mathbf{r}_{f_j}, (\mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top)\mathbf{r}_{f_j} + f_j(\mathbf{m})\big) \in \mathbb{Z}_N^{\kappa+1}$, and the re-randomized ciphertext is of the form $\mathsf{ct}_{f_j}^\star =$

$\big(\mathbf{A}(\mathbf{r}_{f_j}+\mathbf{r}^\star_j), (\mathbf{s}^\top\mathbf{A}+\mathbf{e}^\top)(\mathbf{r}_{f_j}+\mathbf{r}^\star_j)+f_j(\mathbf{m})\big) \in \mathbb{Z}_N^{\kappa+1}$, where $\|\mathbf{r}_{f_j}\|_\infty < \widetilde{B}$ and $\mathbf{r}^\star_j \leftarrow_{\mathsf{R}} [-\widetilde{B}2^\lambda, \widetilde{B}2^\lambda]^m$. By Lemma 2.2 (smudging), the following distributions have statistical distance at most $2^{-\lambda}$:

$$\left(\mathbf{r}^\star_j\right)_{j\in[q]} \approx_s \left(\mathbf{r}^\star_j - \mathbf{r}_{f_j}\right)_{j\in[q]}.$$

The leftmost distribution corresponds to $\mathcal{D}_0$, whereas the rightmost distribution corresponds to $\mathcal{D}_1$. ☐

We also show that a fresh noisy encryption satisfies the following weak correctness notion.

Proposition 21 (approximate noisy correctness)**.** *For all $\lambda \in \mathbb{N}$, all* crs *containing a modulus N of $\theta \in \Omega(\lambda \cdot d(\lambda)$ bits, all* $(\mathsf{pk}, \mathsf{sk})$ *in the support of* $\mathsf{Gen}(\mathsf{crs})$*, all messages $\mathbf{m} \in \{0,1\}^\nu$, all ciphertexts* ct *in the support of* $\mathsf{Encpk}(\mathbf{m})$*, all circuits $f : \{0,1\}^\nu \to \mathbb{Z}_N$ of depth d, all* ct'_f *in the support of* $\mathsf{ReRand}(\mathsf{pk}, \mathsf{ct}_f)$ *where* $\mathsf{ct}_f = \mathsf{Eval}'(\mathsf{pk}, f, 1, \mathsf{ct})$*, we have:*

$$\mathsf{sk}^\top \mathsf{ct}'_f = f(\mathbf{m}) + \mathsf{noise} \in \mathbb{Z}_N,$$

with $|\mathsf{noise}| < B(2^\lambda + 1)$*.*

Proof: The ciphertext ct_f is the binary decomposition of $\big(\mathbf{A}(\mathbf{r}^\star+\mathbf{r}_f), (\mathbf{s}^\top\mathbf{A}+\mathbf{e}^\top)(\mathbf{r}^\star+\mathbf{r}_f)+f(\mathbf{m})\big) \in \mathbb{Z}_N^{\kappa+1}$, where $|\mathbf{e}^\top\mathbf{r}_f| < B$ by batch correctness, and $|\mathbf{e}^\top\mathbf{r}^\star| < 2^\lambda\widetilde{B}B'm = 2^\lambda B$. ☐

5.2 XiO Construction

We directly dive into the formal description of the construction, see the introduction for a detailed overview.

We present a modular construction of XiO for the class of circuits $\mathcal{C}_{\log(n),s,d}$ for polynomials n, s, d, from the following building blocks:

- the GSW FHE scheme for depth δ circuits, denoted by $\mathrm{GSW}_\delta = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Enc}^\star, \mathsf{Eval}, \mathsf{ReRand})$, presented in Section 3.2.2. The depth δ is chosen sufficiently large to handle the homomorphic evaluations of the circuits described below.
- an (ℓ_1, ℓ_2, h)-Hintable Packed LHE, denoted by $\mathcal{LHE}_{b,n,\varepsilon} = (\overline{\mathsf{Gen}}, \overline{\mathsf{Enc}}, \overline{\mathsf{Enc}}^\star, \overline{\mathsf{Dec}}, \overline{\mathsf{Eval}}, \overline{\mathsf{SecHint}}, \overline{\mathsf{PubHint}}, \overline{\mathsf{Rec}})$ where h is independent of n to ensure succinctness; $\ell_1(\lambda) \geq b(\lambda) + 2\lambda$, where 2^b is a bound on the noise obtained when FHE evaluating circuits of depth at most δ[6]; moreover $\big(\ell_1(\lambda) - b(\lambda) - 2\lambda\big) \cdot \ell_2(\lambda) \geq n^\varepsilon(\lambda)$, where $\varepsilon \in (0,1)$ is defined below (see the paragraph about succinctness).

Notations. For every program Π with $\log(n)$ bits inputs, every $\varepsilon \in (0,1)$, the truth table can be written as $(\Pi_i)_{i\in[n^{1-\varepsilon}]}$, where each chunk Π_i contains n^ε bits. The chunks Π_i themselves can be subdivided into sub-chunks $\Pi_i = (\Pi_{i,j})_{j\in[\ell_2]}$, where each sub-chunk $\Pi_{i,j}$ contains n^ε/ℓ_2 bits. For all $i \in [n^{1-\varepsilon}]$ and $j \in [\ell_2]$, we denote by $C_{i,j}$ the circuit that takes as input a program Π of size s and outputs $\Pi_{i,j}$.

The construction: We proceed to the construction.

- $\underline{\mathsf{Gen}_{\mathsf{Obf}}(1^\lambda):}$

Set the parameters:

[6]To make sure these parameters are instantiable, we require that LHE decryption is of poly-logarithmic depth, which ensures that δ and therefore B only depend poly-logarithmically on ℓ_1.

- Choose a constant $0 < \varepsilon < 1$ that is small enough so as to ensure succinctness of the scheme (see paragraph succinctness below).
- Let $|\overline{\mathsf{ct}}|(\cdot)$, $|\overline{\mathbf{r}^\star}|(\cdot)$ be polynomials such that for every $\lambda \in \mathbb{N}$, every $(\overline{\mathsf{pk}}, \overline{\mathsf{sk}})$ in the support of $\overline{\mathsf{Gen}}(1^\lambda)$ that defines the message space $\mathbb{Z}_N^{\ell_2}$ and the noisy randomness space $\mathcal{R}^\star$, every message $\mathbf{m} \in \mathbb{Z}_N^{\ell_2}$, every ciphertext in the support of $\overline{\mathsf{Enc}}_{\overline{\mathsf{pk}}}(\mathbf{m})$ has a bit size at most $|\overline{\mathsf{ct}}|(\lambda)$ and every $\mathbf{r}^\star \in \mathcal{R}^\star$ has bit size at most $|\mathbf{r}^\star|(\lambda)$.
- $\mathsf{FHE.PubCoin} \leftarrow_{\mathsf{R}} \{0,1\}^{n^{1-\varepsilon} \cdot \ell_2 \cdot |\mathbf{r}^\star|}$, $\mathsf{LHE.PubCoin} \leftarrow_{\mathsf{R}} \{0,1\}^{n^{1-\varepsilon} \cdot |\overline{\mathsf{ct}}|}$.

Return $\mathsf{pp} = (\mathsf{FHE.PubCoin}, \mathsf{LHE.PubCoin})$.

- $\underline{\mathsf{Obf}(\mathsf{pp}, 1^n, \Pi)}$:

Sample the following parameters:

- $(\overline{\mathsf{pk}}, \overline{\mathsf{sk}}) \leftarrow \overline{\mathsf{Gen}}(1^\lambda)$ that defines the message space $\mathbb{Z}_N^{\ell_2}$.
- $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(\overline{\mathsf{pk}})$ that defines the noisy randomness space $\mathcal{R}^\star$, where $\mathsf{sk} \in \mathbb{Z}_N^w$, and pk contains the noise bound B; we write $b = \lceil \log(B) \rceil$.

Compute the following ciphertexts:

- $\mathsf{ct}_1 \leftarrow \mathsf{Enc}_{\mathsf{pk}}(\Pi)$.
- $\mathsf{ct}_2 \leftarrow \mathsf{Enc}_{\mathsf{pk}}(\overline{\mathsf{sk}})$.
- $\overline{\mathsf{ct}} \leftarrow \overline{\mathsf{Enc}}_{\overline{\mathsf{pk}}}(\mathsf{sk})$.

For all $i \in [n^{1-\varepsilon}]$, $j \in [\ell_2]$, compute the following:

- $\mathsf{ct}_{i,j} = \mathsf{Eval}'(\mathsf{pk}, C_{i,j}, b + 2\lambda, \mathsf{ct}_1) \in \{0,1\}^w$, where the circuit $C_{i,j}$ is defined above. The homomorphic evaluation is performed with scaling factor $b + 2\lambda$.
- $\mathsf{ct}_{\mathsf{MSB},i,j} = \mathsf{Eval}'(\mathsf{pk}, f_{i,j}, 0, \mathsf{ct}_2) \in \{0,1\}^w$, where the circuit $f_{i,j}$ takes as input a bit string $\mathbf{a} \in \{0,1\}^{|\overline{\mathsf{sk}}|}$, which it uses as an LHE secret key to compute $\mathbf{v}_i = \overline{\mathsf{Dec}}_{\mathbf{a}}(\mathsf{LHE.PubCoin}_i)$, where $\mathsf{LHE.PubCoin}_i$ is interpreted as an LHE ciphertext $\overline{\mathsf{Enc}}_{\overline{\mathsf{pk}}}(\mathbf{u}_i)$, with $\mathbf{u}_i \in \mathbb{Z}_N^{\ell_2}$, by density of the LHE ciphertext space. I If the decryption is successful, then it computes $v_{i,j} \in \mathbb{Z}_N$, the j'th coordinate of $\mathbf{v}_i \in \mathbb{Z}_N^{\ell_2}$ and outputs the most significant bits of $v_{i,j}$, of the form: $\mathsf{MSB}(v_{i,j}) = v_{i,j} - \mathsf{LSB}(v_{i,j}) \in \mathbb{Z}^{\ell_2}$, where the (shifted) least significant bits are of the form: $\mathsf{LSB}(v_{i,j}) = v_{i,j} \mod B2^{2\lambda} - B2^{2\lambda}/2 \in \mathbb{Z}_N$. The homomorphic evaluation is performed with scaling factor 1.
- Parse $\mathsf{FHE.PubCoin}_{i,j} = \mathbf{r}^\star_{i,j} \in \mathcal{R}^\star$ and compute $\mathsf{ct}'_{\mathsf{MSB},i,j} = \mathsf{ReRand}(\mathsf{pk}, \mathsf{ct}_{\mathsf{MSB},i,j}; \mathbf{r}^\star_{i,j}) \in \mathbb{Z}_N^{\kappa+1}$.
- Compute $\mathsf{ct}_i = (\mathsf{ct}_{i,j})_{j \in [\ell_2]} \in \{0,1\}^{w\ell_2}$, $\mathsf{ct}'_{\mathsf{MSB},i} = (\mathsf{ct}'_{\mathsf{MSB},i,j})_{j \in [\ell_2]} \in \{0,1\}^{w\ell_2}$.
- Compute $\overline{\mathsf{ct}}_i = \overline{\mathsf{Eval}}(\overline{\mathsf{pk}}, \overline{\mathsf{ct}}, \mathsf{LHE.PubCoin}_i, \mathsf{ct}_i - \mathsf{ct}'_{\mathsf{MSB},i})$.
- Compute $\rho_i \leftarrow \overline{\mathsf{SecHint}}(\overline{\mathsf{sk}}, \overline{\mathsf{ct}}_i)$.

Return $\widetilde{\Pi} = (\mathsf{pk}, \overline{\mathsf{pk}}, \mathsf{ct}_1, \mathsf{ct}_2, \overline{\mathsf{ct}}, \{\rho_i\}_{i \in [n^{1-\varepsilon}]})$.

- $\underline{\mathsf{Eval}(\mathsf{pp}, \widetilde{\Pi}, \mathbf{x})}$:

- Let $i \in [n^{1-\varepsilon}]$ such that $\Pi(\mathbf{x})$ belongs to the i'th chunk of the truth table of Π. Compute $\overline{\mathsf{ct}}_i$ as described above.

- Recover $m_i \leftarrow \overline{\mathsf{Rec}}(\overline{\mathsf{pk}}, \overline{\mathsf{ct}}_i, \rho_i)$.

- Compute $m'_i = \lfloor 2^{-2\lambda}/B \cdot m_i \rceil$, which contains $\Pi(\mathbf{x})$.

We now proceed to prove Theorem 5.1.

Succinctness. By h-succinctness of $\mathcal{LHE}_{n,q}$, for all $i \in [n^{1-\varepsilon}]$, we have $|\rho_i| \leq h(\lambda)$ for a polynomial h that is independent of n and d. The rest of the obfuscated circuit $\widetilde{\Pi}$ is of size $\mathsf{poly}(\lambda, n^{\varepsilon}, d)$. Overall, there exists a constant $c \in \mathbb{N}$ (independent of ε) and polynomials p such that $|\widetilde{\Pi}| \in n^{c\varepsilon}(\lambda) \cdot p(\lambda) + n^{1-\varepsilon}(\lambda) \cdot h(\lambda)$. For succinctness, we pick an appropriately small $0 < \varepsilon < 1/c$.

Correctness.

- By the batch correctness of the GSW scheme (Proposition 1), for all $i \in [n^{1-\varepsilon}]$ and $j \in [\ell_2]$, we have:
$$\mathsf{sk}^\top \mathsf{ct}_{i,j} = 2^{2\lambda} B \cdot \Pi_{i,j} + \mathsf{noise}_{i,j} \in \mathbb{Z}_N,$$
where $|\mathsf{noise}_{i,j}| < B$.

- By the density of the noisy ciphertexts of $\mathcal{LHE}_n$, for all $i \in [n^{1-\varepsilon}]$, we have $\mathsf{LHE.PubCoin}_i = \overline{\mathsf{Enc}}_{\overline{\mathsf{pk}}}(\mathbf{u}_i)$ with $\mathbf{u}_i \in \mathbb{Z}_N^{\ell_2}$.

- By the approximate noisy correctness of the GSW scheme (Proposition 21), for all $i \in [n^{1-\varepsilon}]$ and $j \in [\ell_2]$, we have:
$$\mathsf{sk}^\top \mathsf{ct}'_{\mathsf{MSB},i,j} = \mathsf{MSB}(u_{i,j}) + \mathsf{noise}_{\mathsf{MSB},i,j} \in \mathbb{Z}_N,$$
where $|\mathsf{noise}_{\mathsf{MSB},i,j}| < (2^\lambda + 1)B$.

- By linear homomorphism of $\mathcal{LHE}_{n,d}$ (Property 4.2), the ciphertext $\overline{\mathsf{ct}}_i$ is in the support of $\overline{\mathsf{Enc}}_{\overline{\mathsf{pk}}}(\mathbf{m}_i)$, $\mathbf{m}_i = (m_{i,j})_{j \in [\ell_2]}$ of the form:
$$m_{i,j} = B 2^{2\lambda} \cdot \Pi_{i,j} + \mathsf{LSB}(u_{i,j}) + \mathsf{noise}_{i,j} + \mathsf{noise}_{\mathsf{MSB},i,j} \in \mathbb{Z}_N.$$

- By correctness of secret hints of $\mathcal{LHE}_{n,d}$ (Property 4.3), the evaluator of the obfuscated circuit recovers the message $\mathbf{m}_i \in \mathbb{Z}_N^{\ell_2}$.

- With probability $1 - 2^{-\Omega(\lambda)}$ over the choice of $\mathbf{u}_i \leftarrow_{\mathsf{R}} \mathbb{Z}_N^{\ell_2}$, we have for all $j \in [\ell_2]$, $|\mathsf{LSB}(u_{i,j}) + \mathsf{noise}_{i,j} + \mathsf{noise}_{\mathsf{MSB},i}| < B 2^{2\lambda}/2$. Thus, $m'_{i,j} = \lfloor m_{i,j} 2^{-2\lambda}/B \rceil = \Pi_{i,j}$ for all $j \in [\ell_2]$, and the evaluator outputs $\Pi(\mathbf{x})$.

IND security. We now prove that the XiO scheme presented in Section 5 is IND-secure, provided the GSW FHE for depth δ circuits and $\mathcal{LHE}_{n,d}$ are 2-circular SRL secure (as per Definition 3.3).

We proceed via a hybrid argument using the following ensembles for all $b \in \{0, 1\}$.

- $\underline{\mathcal{D}_\lambda^b}$: this is the ensemble from Definition 2.4. For completeness, we describe it here.

- Generation of pp: for all $i \in [n^{1-\varepsilon}]$, $\mathsf{LHE.PubCoin}_i \leftarrow_{\mathsf{R}} \{0,1\}^{|\overline{\mathsf{ct}}|}$, for all $j \in [\ell_2]$, $\mathsf{FHE.PubCoin}_{i,j} \leftarrow_{\mathsf{R}} \mathcal{R}^\star$, where $\mathcal{R}^\star$ denotes the noisy randomness space of $\mathcal{FHE}$. Return $\mathsf{pp} = \Big((\mathsf{LHE.PubCoin}_i)_{i\in[n^{1-\varepsilon}]}, (\mathsf{FHE.PubCoin}_{i,j})_{i\in[n^{1-\varepsilon}],j\in[\ell_2]}\Big)$.

- Generation of $\widetilde{\Pi}_b$: $(\overline{\mathsf{pk}}, \overline{\mathsf{sk}}) \leftarrow \overline{\mathsf{Gen}}(1^\lambda)$, $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(\overline{\mathsf{pk}})$, $\mathsf{ct}_1 \leftarrow \mathsf{Enc}_{\mathsf{pk}}(\Pi)$, $\mathsf{ct}_2 \leftarrow \mathsf{Enc}_{\mathsf{pk}}(\overline{\mathsf{sk}})$, $\overline{\mathsf{ct}} \leftarrow \overline{\mathsf{Enc}}_{\overline{\mathsf{pk}}}(\mathsf{sk})$. For all $i \in [n^{1-\varepsilon}]$, $j \in [\ell_2]$, compute the following:
  - $\mathsf{ct}_{i,j} = \mathsf{Eval}'(\mathsf{pk}, C_{i,j}, b + 2\lambda, \mathsf{ct}_1) \in \{0,1\}^w$;
  - $\mathsf{ct}_{\mathsf{MSB},i,j} = \mathsf{Eval}'(\mathsf{pk}, f_{i,j}, 0, \mathsf{ct}_2) \in \{0,1\}^w$;
  - Parse $\mathsf{FHE.PubCoin}_{i,j} = \mathbf{r}^\star_{i,j} \in \mathcal{R}^\star$ and compute $\mathsf{ct}'_{\mathsf{MSB},i,j} = \mathsf{ReRand}(\mathsf{pk}, \mathsf{ct}_{\mathsf{MSB},i,j}; \mathbf{r}^\star_{i,j}) \in \{0,1\}^w$.
  - Compute $\mathsf{ct}_i = (\mathsf{ct}_{i,j})_{j\in[\ell_2]} \in \{0,1\}^{w\ell_2}$, $\mathsf{ct}'_{\mathsf{MSB},i} = (\mathsf{ct}'_{\mathsf{MSB},i,j})_{j\in[\ell_2]} \in \{0,1\}^{w\ell_2}$.
  - $\overline{\mathsf{ct}}_i = \overline{\mathsf{Eval}}(\overline{\mathsf{pk}}, \overline{\mathsf{ct}}, \mathsf{LHE.PubCoin}_i, \mathsf{ct}_i - \mathsf{ct}'_{\mathsf{MSB},i})$;
  - $\rho_i \leftarrow \overline{\mathsf{SecHint}}(\overline{\mathsf{sk}}, \overline{\mathsf{ct}}_i)$.

  Return $\widetilde{\Pi}_b = \big(\mathsf{pk}, \overline{\mathsf{pk}}, \mathsf{ct}_1, \mathsf{ct}_2, \overline{\mathsf{ct}}, (\rho_i)_{i\in[n^{1-\varepsilon}]}\big)$.

• $\underline{\mathcal{H}^{b.1}_\lambda}$: the ensemble samples $\mathsf{LHE.PubCoin}$ as in $\mathcal{D}^b_\lambda$, but does not sample $\mathsf{FHE.PubCoin}$ just yet; it then generates $\widetilde{\Pi}_b$ as in $\mathcal{D}^b_\lambda$ up until the point that the $\mathsf{ct}_{\mathsf{MSB},i,j}$ get re-randomized into $\mathsf{ct}'_{\mathsf{MSB},i,j}$ via ReRand. Next, instead of performing the re-randomization, it samples $\mathsf{ct}'_{\mathsf{MSB},i,j}$ as a *fresh* extra noisy encryption of $\mathsf{MSB}(u_{i,j})$ using randomness $\mathbf{r}^\star_{i,j} \leftarrow_{\mathsf{R}} \mathcal{R}^\star$, and setting $\mathsf{FHE.PubCoin}_{i,j}$ to be $\mathbf{r}^\star_{i,j} - \mathbf{r}_{f_{i,j}}$, where $\mathbf{r}_{f_{i,j}}$ denotes the evaluated randomness computed via $\mathsf{Eval}_{\mathsf{rand}}$. Afterwards, the experiment continues exactly the same way as in $\mathcal{D}^b_\lambda$.

We use the weak circuit privacy of $\mathcal{FHE}$ (Theorem 5.2) we have:

$$\mathcal{D}^b_\lambda \approx_s \mathcal{H}^{b.1}_\lambda.$$

The latter states that for all $(\overline{\mathsf{pk}}, \overline{\mathsf{sk}})$ in the support of $\overline{\mathsf{Gen}}(1^\lambda)$, all $(\mathsf{pk}, \mathsf{sk})$ in the support of $\mathsf{Gen}(\overline{\mathsf{pk}})$, for all depth d-circuits and in particular the functions $f_{i,j}$ defined previously, these two distributions have statistical distance at most $2^{-\Omega(\lambda)}$:

$$\mathcal{D}_0 : \left\{ \begin{array}{l} r \leftarrow_{\mathsf{R}} \big([-1,1]^{m\times w}\big)^{|\overline{\mathsf{sk}}|}, \mathsf{ct} = \mathsf{Enc}_{\mathsf{pk}}(\overline{\mathsf{sk}}; r), \forall i \in [n^{1-\varepsilon}], j \in [\ell_2], \mathsf{ct}_{f_{i,j}} = \mathsf{Eval}'(\mathsf{pk}, f_{i,j}, 0, \mathsf{ct}) \\ \mathbf{r}^\star_{i,j} \leftarrow_{\mathsf{R}} \mathcal{R}^\star, \mathsf{ct}^\star_{f_{i,j}} = \mathsf{ReRand}(\mathsf{pk}, \mathsf{ct}_{f_{i,j}}; \mathbf{r}^\star_{i,j}) : \left(\overline{\mathsf{pk}}, \mathsf{pk}, \mathsf{ct}, \left(\mathbf{r}^\star_{i,j}, \mathsf{ct}^\star_{f_{i,j}}\right)_{i\in[n^{1-\varepsilon}],j\in[\ell_2]}\right) \end{array} \right\}$$

$$\mathcal{D}_1 : \left\{ \begin{array}{l} r \leftarrow_{\mathsf{R}} \big([-1,1]^{m\times w}\big)^{|\overline{\mathsf{sk}}|}, \mathsf{ct} = \mathsf{Enc}_{\mathsf{pk}}(\overline{\mathsf{sk}}; r), \forall i \in [n^{1-\varepsilon}], j \in [\ell_2], \mathbf{r}^\star_{i,j} \leftarrow_{\mathsf{R}} \mathcal{R}^\star \\ \mathsf{ct}^\star_{f_{i,j}} = \mathsf{Enc}^\star_{\mathsf{pk}}(\mathsf{MSB}(u_{i,j}); \mathbf{r}^\star_{i,j}) \\ \mathbf{r}_{f_{i,j}} = \mathsf{Eval}_{\mathsf{rand}}\left(\mathsf{pk}, f_{i,j}, r, \overline{\mathsf{sk}}\right) : \left(\overline{\mathsf{pk}}, \mathsf{pk}, \mathsf{ct}, \left(\mathbf{r}^\star_{i,j} - \mathbf{r}_{f_{i,j}}, \mathsf{ct}^\star_{f_{i,j}}\right)_{i\in[n^{1-\varepsilon}],j\in[\ell_2]}\right) \end{array} \right\}.$$

We design an inefficient simulator $\mathcal{S}$ that given a tuple $\big(\overline{\mathsf{pk}}, \mathsf{pk}, \mathsf{ct}, (\mathbf{r}_{i,j}, \mathsf{ct}_{i,j})_{i\in[n^{1-\varepsilon}],j\in[\ell_2]}\big)$, simulates the adversary view in the XiO security experiment. That is, we show that when fed with an input distributed according to $\mathcal{D}_0$, $\mathcal{S}$ simulates the experiment $\mathcal{D}^b_\lambda$, whereas it simulates the experiment $\mathcal{H}^{b.1}_\lambda$ when fed with an input distributed according to $\mathcal{D}_1$.

Given $\big(\overline{\mathsf{pk}}, \mathsf{pk}, \mathsf{ct}, (\mathbf{r}_{i,j}, \mathsf{ct}_{i,j})_{i\in[n^{1-\varepsilon}],j\in[\ell_2]}\big)$, $\mathcal{S}$ (inefficiently) recovers $\overline{\mathsf{sk}}$ from $\overline{\mathsf{pk}}$, sk from pk, and the randomness r from ct (more precisely $\mathcal{S}$ samples some uniformly random $\overline{\mathsf{sk}}$, sk, r among those that match $\overline{\mathsf{pk}}$, pk and ct). It samples $\mathsf{LHE.PubCoin} \leftarrow_{\mathsf{R}} \{0,1\}^{n^{1-\varepsilon}\cdot|\overline{\mathsf{ct}}|}$, and for all $i \in [n^{1-\varepsilon}]$, $j \in [\ell_2]$, sets

$\mathsf{FHE.PubCoin}_{i,j} = \mathbf{r}_{i,j}$, and $\mathsf{pp} = (\mathsf{LHE.PubCoin}, (\mathsf{FHE.PubCoin}_{i,j})_{i,j})$. It computes $\mathsf{ct}_1 \leftarrow \mathsf{Enc}_{\mathsf{pk}}(\Pi_b)$, $\overline{\mathsf{ct}} \leftarrow \overline{\mathsf{Enc}}_{\overline{\mathsf{pk}}}(\mathsf{sk})$.

For all $i \in [n^{1-\varepsilon}]$, $j \in [\ell_2]$, $\mathcal{S}$ computes the following:

- $\mathsf{ct}_{i,j} = \mathsf{Eval}'(\mathsf{pk}, C_{i,j}, b + 2\lambda, \mathsf{ct}_1) \in \{0,1\}^w$;
- $\mathsf{ct}'_{\mathsf{MSB},i,j} = \mathsf{ct}_{i,j} \in \{0,1\}^w$.
- Compute $\mathsf{ct}_i = (\mathsf{ct}_{i,j})_{j\in[\ell_2]} \in \{0,1\}^{w\ell_2}$, $\mathsf{ct}'_{\mathsf{MSB},i} = (\mathsf{ct}'_{\mathsf{MSB},i,j})_{j\in[\ell_2]} \in \{0,1\}^{w\ell_2}$.
- $\overline{\mathsf{ct}}_i = \overline{\mathsf{Eval}}(\overline{\mathsf{pk}}, \overline{\mathsf{ct}}, \mathsf{LHE.PubCoin}_i, \mathsf{ct}_i - \mathsf{ct}'_{\mathsf{MSB},i})$;
- $\rho_i \leftarrow \overline{\mathsf{SecHint}}(\overline{\mathsf{sk}}, \overline{\mathsf{ct}}_i)$.

The simulator sets $\widetilde{\Pi}_b = \big(\mathsf{pk}, \overline{\mathsf{pk}}, \mathsf{ct}_1, \mathsf{ct}_2, \overline{\mathsf{ct}}, (\rho_i)_{i\in[n^{1-\varepsilon}]}\big)$, and returns $(\mathsf{pp}, \widetilde{\Pi_b})$. It is clear from the description of the simulator $\mathcal{S}$ that when the latter is fed with an input distributed according to $\mathcal{D}_0$, it simulates $\mathcal{D}^b_\lambda$, whereas it simulates $\mathcal{H}^{b.1}_\lambda$ when fed with an input distributed according to $\mathcal{D}_1$.

• $\underline{\mathcal{H}^{b.2}_\lambda}$: this ensemble is the same as $\mathcal{H}^{b.1}_\lambda$, except that instead of sampling $\mathsf{LHE.PubCoin}_i$ as random strings, they are sampled as fresh LHE ciphertexts of random plaintexts, that is, of the form $\mathsf{LHE.PubCoin}_i \leftarrow \overline{\mathsf{Enc}}_{\overline{\mathsf{pk}}}(\mathbf{u}_i)$ for $\mathbf{u}_i \leftarrow_{\mathsf{R}} \mathbb{Z}_N^{\ell_2}$. By density of the ciphertexts of $\mathcal{LHE}$ Property 4.7, we have:

$$\mathcal{H}^{b.1}_\lambda \approx_s \mathcal{H}^{b.2}_\lambda,$$

with statistical distance $2^{-\Omega(\lambda)}$.

• $\underline{\mathcal{H}^{b.3}_\lambda}$: this ensemble is the same as $\mathcal{H}^{b.2}_\lambda$, except it generates the ciphertexts $\overline{\mathsf{ct}}_i$ as fresh noisy LHE encryptions of the messages $\mathbf{m}_i = \mathsf{sk}^\top(\mathsf{ct}_i - \mathsf{ct}'_{\mathsf{MSB},i}) + \mathbf{u}_i \in \mathbb{Z}_N^{\ell_2}$, and $\mathsf{LHE.PubCoin}_i$ is instead computed homomorphically by subtracting the LHE encryption of the message $\widetilde{\mathbf{m}_i} = \mathsf{sk}^\top(\mathsf{ct}_i + \mathsf{ct}'_{\mathsf{MSB},i}) \in \mathbb{Z}_N^{\ell_2}$ from the fresh noisy encryption of $\mathbf{m}_i$. That is, for all $i \in [n^{1-\varepsilon}]$, $\overline{\mathsf{ct}}_i \leftarrow \overline{\mathsf{Enc}^\star_{\mathsf{pk}}}(\mathbf{m}_i)$, $\mathsf{LHE.PubCoin}_i = \overline{\mathsf{Eval}}\left(\overline{\mathsf{pk}}, \overline{\mathsf{ct}}, \overline{\mathsf{ct}}_i, -\mathsf{ct}_i + \mathsf{ct}'_{\mathsf{MSB},i}\right)$.

Note that it is possible to define this hybrid since $\mathsf{ct}'_{\mathsf{MSB},i}$ remains exactly the same no matter what $\mathsf{LHE.PubCoin}_i$ is. This was not true in $\mathcal{D}^b_\lambda$, and we introduced $\mathcal{H}^{b.1}_\lambda$ to break this dependency.

We use the weak circuit privacy property of $\mathcal{LHE}_{n,d}$ (Property 4.6), to show that:

$$\mathcal{H}^{b.2}_\lambda \approx_s \mathcal{H}^{b.3}_\lambda,$$

with statistical distance $2^{-\Omega(\lambda)}$.

The latter states that for all $(\overline{\mathsf{pk}}, \overline{\mathsf{sk}})$ in the support of $\overline{\mathsf{Gen}}(1^\lambda)$, all vectors $\mathbf{x} \in \mathbb{Z}_N^w$ and in particular $\mathbf{x} = \mathsf{sk} \in \mathbb{Z}_N^w$, all $\mathbf{u}_i \in \mathbb{Z}_N^{\ell_2}$, all functions $\mathbf{y}^i = (\mathbf{y}^i_1, \ldots, \mathbf{y}^i_{\ell_2}) \in [-1,1]^{w\ell_2}$ and in particular the vector $\mathsf{ct}_i - \mathsf{ct}'_{\mathsf{MSB},i} \in [-1,1]^{w\ell_2}$ defined previously for all $i \in [n^{1-\varepsilon}]$, the following distributions have statistical distance $2^{-\Omega(\lambda)}$:

$$\mathcal{D}_0 = \left\{ \begin{array}{l} \overline{\mathsf{ct}} \leftarrow \overline{\mathsf{Enc}}_{\overline{\mathsf{pk}}}(\mathsf{sk}), \forall i \in [n^{1-\varepsilon}], \mathsf{LHE.PubCoin}_i \leftarrow \overline{\mathsf{Enc}^\star_{\mathsf{pk}}}(\mathbf{u}_i) \\ \overline{\mathsf{ct}}_i = \overline{\mathsf{Eval}}(\overline{\mathsf{pk}}, \overline{\mathsf{ct}}, \mathsf{LHE.PubCoin}_i, \mathsf{ct}_i - \mathsf{ct}'_{\mathsf{MSB},i}) : \big(\overline{\mathsf{pk}}, \overline{\mathsf{ct}}, (\mathsf{LHE.PubCoin}_i, \overline{\mathsf{ct}}_i)_{i\in[n^{1-\varepsilon}]}\big) \end{array} \right\}$$

$$\mathcal{D}_1 = \left\{ \begin{array}{l} \overline{\mathsf{ct}} \leftarrow \overline{\mathsf{Enc}}_{\overline{\mathsf{pk}}}(\mathsf{sk}), \forall i \in [n^{1-\varepsilon}], \overline{\mathsf{ct}}_i \leftarrow \overline{\mathsf{Enc}^\star_{\mathsf{pk}}}(\mathbf{m}_i) \\ \mathsf{LHE.PubCoin}_i = \overline{\mathsf{Eval}}(\overline{\mathsf{pk}}, \overline{\mathsf{ct}}, \overline{\mathsf{ct}}_i, -\mathsf{ct}_i + \mathsf{ct}'_{\mathsf{MSB},i}) : \big(\overline{\mathsf{pk}}, \overline{\mathsf{ct}}, (\mathsf{LHE.PubCoin}_i, \overline{\mathsf{ct}}_i)_{i\in[n^{1-\varepsilon}]}\big) \end{array} \right\},$$

where for all $i \in [n^{1-\varepsilon}]$, $\mathbf{m}_i = \mathsf{sk}^\top(\mathsf{ct}_i - \mathsf{ct}'_{\mathsf{MSB},i}) + \mathbf{u}_i \in \mathbb{Z}_N^{\ell_2}$.

We design an inefficient simulator $\mathcal{S}$ that given a tuple $\left(\overline{\mathsf{pk}}, \overline{\mathsf{ct}}, (\widetilde{\mathsf{ct}}_i, \overline{\mathsf{ct}}_i)_{i\in[n^{1-\varepsilon}]}\right)$, simulates the adversary view in the XiO security experiment. That is, we show that when fed with an input distributed according to $\mathcal{D}_0$, $\mathcal{S}$ simulates the experiment $\mathcal{H}_\lambda^{b.1}$, whereas it simulates the experiment $\mathcal{H}_\lambda^{b.2}$ when fed with an input distributed according to $\mathcal{D}_1$.

Given $\left(\overline{\mathsf{pk}}, \overline{\mathsf{ct}}, (\widetilde{\mathsf{ct}}_i, \overline{\mathsf{ct}}_i)_{i\in[n^{1-\varepsilon}]}\right)$, $\mathcal{S}$ (inefficiently) recovers $\overline{\mathsf{sk}}$ from $\overline{\mathsf{pk}}$, sk from $\overline{\mathsf{ct}}$, pk from sk and $\mathbf{u}_i$ from $\widetilde{\mathsf{ct}}_i$ for all $i \in [n^{1-\varepsilon}]$. It generates $\mathsf{ct}_1 \leftarrow \mathsf{Enc}_{\mathsf{pk}}(\Pi_b)$, $r \leftarrow_{\mathsf{R}} \left([-1,1]^{m\times w}\right)^{|\overline{\mathsf{sk}}|}$, $\mathsf{ct}_2 = \mathsf{Enc}_{\mathsf{pk}}(\overline{\mathsf{sk}}; r)$, for all $i \in [n^{1-\varepsilon}]$, $j \in [\ell_2]$, $\mathsf{ct}_{i,j} = \mathsf{Eval}'(\mathsf{pk}, C_{i,j}, b + 2\lambda, \mathsf{ct}_1)$, $\mathbf{r}^\star_{i,j} \leftarrow_{\mathsf{R}} \mathcal{R}^\star$, where $\mathcal{R}^\star$ denotes the noisy randomness space of $\mathcal{FHE}_d$, $\mathsf{ct}'_{\mathsf{MSB},i,j} = \mathsf{Enc}^\star_{\mathsf{pk}}(\mathsf{MSB}(u_{i,j}); \mathbf{r}^\star_{i,j})$, $\mathbf{r}_{f_{i,j}} = \mathsf{Eval}_{\mathsf{rand}}(\mathsf{pk}, f_{i,j}, r, \overline{\mathsf{sk}})$, $\mathsf{FHE.PubCoin}_{i,j} = \mathbf{r}^\star_{i,j} - \mathbf{r}_{f_{i,j}}$, $\rho_i \leftarrow \overline{\mathsf{SecHint}}(\overline{\mathsf{sk}}, \overline{\mathsf{ct}}_i)$, $\mathsf{LHE.PubCoin}_i = \widetilde{\mathsf{ct}}_i$.

It returns $\mathsf{pp} = \left((\mathsf{LHE.PubCoin}_i)_i, (\mathsf{FHE.PubCoin}_{i,j})_{i,j}\right)$ and $\widetilde{\Pi_b} = \left(\mathsf{pk}, \overline{\mathsf{pk}}, \mathsf{ct}_1, \mathsf{ct}_2, \overline{\mathsf{ct}}, (\rho_i)_{i\in[n^{1-\varepsilon}]}\right)$. It is clear from the description of the simulator $\mathcal{S}$ that when the latter is fed with an input distributed according to $\mathcal{D}_0$, it simulates $\mathcal{H}_\lambda^{b.1}$, whereas it simulates $\mathcal{H}_\lambda^{b.2}$ when fed with an input distributed according to $\mathcal{D}_1$.

- $\underline{\mathcal{H}_\lambda^{b.4}}$: it is the same ensemble as $\mathcal{H}_\lambda^{b.3}$, except the hints ρ_i for $i \in [n^{1-\varepsilon}]$ are computed using $\overline{\mathsf{PubHint}}(\overline{\mathsf{pk}}, r_i)$, where r_i denotes the randomness used to produces the ciphertexts $\overline{\mathsf{ct}}_i$; instead of $\overline{\mathsf{SecHint}}(\overline{\mathsf{sk}}, \overline{\mathsf{ct}}_i)$. By Property 4.4, we have $\mathcal{H}_\lambda^{b.3} \approx_s \mathcal{H}_\lambda^{b.4}$, with statistical distance $2^{-\Omega(\lambda)}$. Note that in $\mathcal{H}_\lambda^{b.4}$, we no longer use the LHE secret key $\overline{\mathsf{sk}}$.

- $\underline{\mathcal{H}_\lambda^{b.5}}$: it is the same ensemble as $\mathcal{H}_\lambda^{b.4}$, except that $\overline{\mathsf{ct}}_i$ is generated as a fresh encryption of a message $\mathbf{m}_i \in \mathbb{Z}_N^{\ell_2}$ of the form $(m_{i,1}, \ldots, m_{i,\ell_2})$ where for all $j \in [\ell_2]$, we have $m_{i,j} = B2^{2\lambda} \cdot \Pi^b_{i,j} + \mathsf{LSB}(u_{i,j}) \in \mathbb{Z}_N$. Recall that $\mathsf{LSB}(u_{i,j}) = u_{i,j} \mod B2^{2\lambda} - B2^{2\lambda}/2 \in \mathbb{Z}_N$. This is instead of having $m_{i,j} = \mathsf{sk}^\top(\mathsf{ct}_{i,j} + \mathsf{ct}'_{\mathsf{MSB},i,j}) + u_{i,j} = B2^{2\lambda} \cdot \Pi^b_{i,j} + \mathsf{LSB}(u_{i,j}) + \mathsf{noise}_{i,j} + \mathsf{noise}_{\mathsf{MSB},i} \in \mathbb{Z}_N$, where $\mathsf{noise}_{i,j} = \mathbf{e}^\top \mathbf{r}_{C_{i,j}} \in \mathbb{Z}_N$ and $\mathsf{noise}_{\mathsf{MSB},i} = \mathbf{e}^\top \mathbf{r}^\star_{i,j} \in \mathbb{Z}_N$, $\mathbf{r}_{C_{i,j}}$ is the randomness obtained when evaluating the circuit $C_{i,j}$ on the FHE ciphertext ct_1, $\mathbf{r}^\star_{i,j} \leftarrow_{\mathsf{R}} \mathcal{R}^\star$, and $\mathbf{e} \leftarrow \chi^m$ is used to generate pk.

Note in particular that $\mathsf{noise}_{i,j}$ and $\mathsf{noise}_{\mathsf{MSB},i}$ are deterministic functions of pk, $\mathbf{r}^\star_{i,j}$ and the randomness $\left([-1,1]^{m\times w}\right)^s$ used to produce ct_1.

We show that $\mathcal{H}_\lambda^{b.3} \approx_s \mathcal{H}_\lambda^{b.4}$ with statistical distance $2^{-\Omega(\lambda)}$. To do so, we exhibit two distributions $\mathcal{D}_0$ and $\mathcal{D}_1$, together with a (possibly inefficient) simulator $\mathcal{S}$, such that (1) $\mathcal{D}_0$ and $\mathcal{D}_1$ have statistical distance $2^{-\Omega(\lambda)}$, and (2) for all $\beta \in \{0,1\}$, when fed with an input from distribution $\mathcal{D}_\beta$, $\mathcal{S}$ produces the adversary view as in hybrid $\mathcal{H}_\lambda^{b.4+\beta}$.

The distributions are defined as follows (the differences are highlighted in red):

$$\mathcal{D}_0 = \left\{ \begin{array}{c} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(\mathsf{crs}), r \leftarrow_{\mathsf{R}} \left([-1,1]^{m\times w}\right)^s, \forall i \in [n^{1-\varepsilon}], j \in [\ell_2], \mathbf{r}^\star_{i,j} \leftarrow_{\mathsf{R}} \mathcal{R}^\star \\ \gamma_{i,j} \leftarrow_{\mathsf{R}} \left(-B2^{2\lambda}/2, B2^{2\lambda}/2\right] : \left(\mathsf{pk}, r, \left(\gamma_{i,j}, \mathbf{r}^\star_{i,j}\right)_{i\in[n^{1-\varepsilon}], j\in[\ell_2]}\right) \end{array} \right\}$$

$$\mathcal{D}_1 = \left\{ \begin{array}{c} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(\mathsf{crs}), r \leftarrow_{\mathsf{R}} \left([-1,1]^{m\times w}\right)^s, \forall i \in [n^{1-\varepsilon}], j \in [\ell_2], \mathbf{r}^\star_{i,j} \leftarrow_{\mathsf{R}} \mathcal{R}^\star \\ \gamma_{i,j} \leftarrow_{\mathsf{R}} \left(-B2^{2\lambda}/2, B2^{2\lambda}/2\right] : \left(\mathsf{pk}, r, \left(\gamma_{i,j} + \mathsf{noise}_{i,j} + \mathsf{noise}_{\mathsf{MSB},i,j}, \mathbf{r}^\star_{i,j}\right)_{i\in[n^{1-\varepsilon}], j\in[\ell_2]}\right) \end{array} \right\},$$

where $\mathsf{noise}_{i,j}$ $\mathsf{noise}_{\mathsf{MSB},i,j}$ are functions of pk, r and $\mathbf{r}^\star_{i,j}$ defined as below, namely, $\mathsf{noise}_{i,j} = \mathbf{e}^\top \mathbf{r}_{C_{i,j}}$ and $\mathsf{noise}_{\mathsf{MSB},i,j} = \mathbf{e}^\top \mathbf{r}^\star_{i,j}$.

We show that these distributions have statistical distance $2^{-\Omega(\lambda)}$. The only difference is that in $\mathcal{D}_1$, an extra noise $\mathsf{noise}_{i,j} + \mathsf{noise}_{\mathsf{MSB},i,j}$ is added to the random value $s_{i,j}$. This noise is small, indeed

$|\mathsf{noise}_{i,j} + \mathsf{noise}_{\mathsf{MSB},i,j}| \leq B(2^\lambda + 1)$ (see the correctness section for more details). Moreover, $\gamma_{i,j}$ is sampled uniformly at random over $\big(- B2^{2\lambda}/2, B2^{2\lambda}/2\big]$, independently of the other values output by the distributions. Thus, we can use the value $\gamma_{i,j}$ to smudge the noise $\mathsf{noise}_{i,j} + \mathsf{noise}_{\mathsf{MSB},i,j}$. That is, by Lemma 2.2 (smudging), the statistical distance of the two distributions is $2^{-\Omega(\lambda)}$.

Now, we proceed to describe the simulator $\mathcal{S}$. Given as input the tuple $\Big(\mathsf{pk}, r, (v_{i,j}, \mathbf{r}^\star_{i,j})_{i\in[n^{1-\varepsilon}],j\in[\ell_2]}\Big)$, the simulator (inefficiently) recovers sk from pk, samples $(\overline{\mathsf{pk}}, \overline{\mathsf{sk}}) \leftarrow \overline{\mathsf{Gen}}(1^\lambda)$, generates $\overline{\mathsf{ct}} \leftarrow \overline{\mathsf{Enc}}_{\overline{\mathsf{pk}}}(\mathsf{sk})$, $\mathsf{ct}_1 = \mathsf{Enc}_{\mathsf{pk}}(\Pi_b; r)$. It samples $r' \leftarrow_{\mathsf{R}} \big([-1,1]^{m\times w}\big)^{|\overline{\mathsf{sk}}|}$, computes $\mathsf{ct}_2 = \mathsf{Enc}_{\mathsf{pk}}\big(\overline{\mathsf{sk}}; r'\big)$.

For all $i \in [n^{1-\varepsilon}]$, $j \in [\ell_2]$, samples $\omega_{i,j} \leftarrow_{\mathsf{R}} \mathbb{Z}_{N/(2^{2\lambda}B)}$, and sets $m_{i,j} = B2^{2\lambda} \cdot \Pi^b_{i,j} + v_{i,j} + \omega_{i,j} \in \mathbb{Z}_N$. Note that $(\gamma_{i,j}, \omega_{i,j})$ is identically distributed to $(\mathsf{LSB}(u_{i,j}), \mathsf{MSB}(u_{i,j}))$ for $u_{i,j} \leftarrow_{\mathsf{R}} \mathbb{Z}_N$.

It sets $\mathbf{m}_i = (m_{i,1}, \ldots, m_{i,\ell_2}) \in \mathbb{Z}_N^{\ell_2}$, samples $r_i \leftarrow_{\mathsf{R}} \overline{\mathcal{R}}$, where $\overline{\mathcal{R}}$ denotes the randomness space of $\overline{\mathsf{Enc}}$, computes $\overline{\mathsf{ct}}_i = \overline{\mathsf{Enc}}_{\overline{\mathsf{pk}}}(\mathbf{m}_i; r_i)$ and $\rho_i \leftarrow \overline{\mathsf{PubHint}}(\overline{\mathsf{pk}}, r_i)$. It computes $\mathsf{ct}'_{\mathsf{MSB},i,j} = \mathsf{Enc}^\star_{\mathsf{pk}}(\omega_{i,j}; \mathbf{r}^\star_{i,j})$, $\mathsf{ct}_{i,j} = \mathsf{Eval}'(\mathsf{pk}, C_{i,j}, b + 2\lambda, \mathsf{ct}_1)$, $\mathsf{ct}_i = (\mathsf{ct}_{i,j})_{j\in[\ell_2]} \in \{0,1\}^{w\ell_2}$, $\mathsf{ct}'_{\mathsf{MSB},i} = (\mathsf{ct}'_{\mathsf{MSB},i,j})_{j\in[\ell_2]} \in \{0,1\}^{w\ell_2}$, and $\mathsf{LHE.PubCoin}_i = \overline{\mathsf{Eval}}(\overline{\mathsf{pk}}, \overline{\mathsf{ct}}, \overline{\mathsf{ct}}_i, -\mathsf{ct}_i + \mathsf{ct}'_{\mathsf{MSB},i})$. It computes $\mathbf{r}_{f_{i,j}} = \mathsf{Eval}_{\mathsf{rand}}(\mathsf{pk}, f_{i,j}, \mathsf{ct}_2)$ where the functions $f_{i,j}$ are defined as before, and sets $\mathsf{FHE.PubCoin}_{i,j} = \mathbf{r}^\star_{i,j} - \mathbf{r}_{f_{i,j}} \in \mathbb{Z}_N^m$.

It returns $\mathsf{pp} = (\mathsf{LHE.PubCoin}_i, \mathsf{FHE.PubCoin}_{i,j})_{i\in[n^{1-\varepsilon}]}$, and $\widetilde{\Pi}_b = \big(\mathsf{pk}, \overline{\mathsf{pk}}, \mathsf{ct}_1, \mathsf{ct}_2, \overline{\mathsf{ct}}, (\rho_i)_{i\in[n^{1-\varepsilon}]}\big)$. When $\mathcal{S}$ is fed with distribution $\mathcal{D}_0$, it simulates the hybrid $\mathcal{H}^{b.4}_\lambda$, whereas it simulates the hybrid $\mathcal{H}^{b.5}_\lambda$ when fed with distribution $\mathcal{D}_1$. Thus, we have:

$$\mathcal{H}^{b.4}_\lambda \approx_s \mathcal{H}^{b.5}_\lambda.$$

- $\underline{\mathcal{H}^{0.5}_\lambda \approx_c \mathcal{H}^{1.5}_\lambda}$: To complete the proof, we show that $\mathcal{H}^{0.5}_\lambda$ is computationally indistinguishable from $\mathcal{H}^{1.5}_\lambda$. These two ensembles are the same except the former obfuscates the program Π^0, whereas the latter obfuscates Π^1. Note that other than the encrypted key cycle, we never use the FHE secret key, and due to Hybrid $\mathcal{H}^{b.4}_\lambda$, we no longer use the LHE secret key. The coins $\mathsf{FHE.PubCoin}$ exactly correspond to an SRL leakage on the FHE ciphertext ct_2 (and note that in the experiment we do know the output $\alpha_{i,j}$ of the function $f_{i,j}$ that is applied to the plaintexts encrypted in $\mathsf{ct}_1, \mathsf{ct}_2$—namely, it is $\mathsf{MSB}(u_{i,j})$ where $u_{i,j}$ is a random element of $\mathbb{Z}_N$ selected in the experiment, see Hybrid $\mathcal{H}^{b.2}_\lambda$). Thus, indistinguishability of $\mathcal{H}^{0.5}_\lambda$ and $\mathcal{H}^{1.5}_\lambda$ follows from 2-circular SRL-security of $\mathcal{FHE}_d$ and $\mathcal{LHE}_{n,d}$.

Namely, we provide a reduction from the 2-circular SRL security with respect to $\mathcal{FHE}_d$ and $\mathcal{LHE}_{n,d}$ (as per Definition 3.3) to distinguishing $\mathcal{H}^{0.5}_\lambda$ and $\mathcal{H}^{1.5}_\lambda$. Given the public keys pk, $\overline{\mathsf{pk}}$ and the ciphertexts $\mathsf{ct} = (\mathsf{ct}_1 \| \mathsf{ct}_2)$, $\overline{\mathsf{ct}}$ provided by the 2-circular security experiment, where ct_1 encrypts Π^0 or Π^1, ct_2 encrypts $\overline{\mathsf{sk}}$, and $\overline{\mathsf{ct}}$ encrypts sk, the reduction samples $u_{i,j} \leftarrow_{\mathsf{R}} \mathbb{Z}_N$ for all $i \in [n^{1-\varepsilon}]$, $j \in [\ell_2]$, and generates the following:

- Generation of pp:
  - To generate $\mathsf{LHE.PubCoin}_i$:
    * It samples $r_i \leftarrow_{\mathsf{R}} \overline{\mathcal{R}}$, where $\overline{\mathcal{R}}$ denotes the randomness space of $\overline{\mathsf{Enc}}$, and computes $\overline{\mathsf{ct}}_i = \overline{\mathsf{Enc}}_{\overline{\mathsf{pk}}}(\mathbf{m}_i; r_i)$, where for all $j \in [\ell_2]$, the j'th coordinate of $\mathbf{m}_i$ is of the form: $m_{i,j} = B2^{2\lambda} \cdot \Pi^b_{i,j} + \mathsf{LSB}(u_{i,j}) \in \mathbb{Z}_N$. Note that this does not require to know the bit b, since $\Pi^0_{i,j} = \Pi^1_{i,j}$ for all $i \in [n^{1-\varepsilon}]$, $j \in [\ell_2]$, because the program Π^0 and Π^1 are functionally equivalent.
    * It computes $\rho_i \leftarrow \overline{\mathsf{PubHint}}(\overline{\mathsf{pk}}, r_i)$.
    * It computes $\mathsf{ct}_{i,j} = \mathsf{Eval}'(\mathsf{pk}, C_{i,j}, b + 2\lambda, \mathsf{ct}_1)$.
    * Then, it queries its $\mathcal{O}_{\mathsf{SRL}}$ oracle, to obtain a fresh, extra noisy encryption $\mathsf{Enc}^\star_{\mathsf{pk}}(0; \mathbf{r}^\star_{i,j})$. It leaves the oracle $\mathcal{O}_{\mathsf{SRL}}$ pending.

    * It adds the vector $(\mathbf{0}, \mathsf{MSB}(u_{i,j})) \in \mathbb{Z}_N^{\kappa+1}$ to the vector whose binary decomposition is $\mathsf{Enc}^\star_{\mathsf{pk}}(0; \mathbf{r}^\star_{i,j})$, which yields a vector $\mathsf{ct}^\star_{i,j} \in \mathbb{Z}_N^{\kappa+1}$ whose binary decomposition is $\mathsf{Enc}^\star_{\mathsf{pk}}(\mathsf{MSB}(u_{i,j}); \mathbf{r}^\star_{i,j})$. Then, it computes $\mathsf{ct}'_{\mathsf{MSB},i,j} = \mathsf{BD}(\mathsf{ct}^\star_{i,j}) \in \{0,1\}^w$.
    * It computes $\mathsf{ct}_i = (\mathsf{ct}_{i,j})_{j \in [\ell_2]} \in \{0,1\}^{w\ell_2}$ and $\mathsf{ct}'_{\mathsf{MSB},i} = (\mathsf{ct}'_{\mathsf{MSB},i,j})_{j \in [\ell_2]} \in \{0,1\}^{w\ell_2}$.
    * Finally, it computes $\mathsf{LHE.PubCoin}_i = \overline{\mathsf{Eval}}(\overline{\mathsf{pk}}, \overline{\mathsf{ct}}, \overline{\mathsf{ct}}_i, -\mathsf{ct}_i - \mathsf{ct}'_{\mathsf{MSB},i})$.
  - To generate $\mathsf{FHE.PubCoin}_{i,j}$: it answers the pending oracle $\mathcal{O}_{\mathsf{SRL}}$ with the function $f_{i,j}$ and the value $\alpha = \mathsf{MSB}(u_{i,j})$. The oracle $\mathcal{O}_{\mathsf{SRL}}$ returns the leakage $\mathbf{r}^\star_{i,j} - \mathbf{r}_{f_{i,j}} \in \mathcal{R}^\star$. The reduction sets $\mathsf{FHE.PubCoin}_{i,j} = \mathbf{r}^\star_{i,j} - \mathbf{r}_{f_{i,j}}$.

  It returns $\mathsf{pp} = \big((\mathsf{LHE.PubCoin}_i)_{i \in [n^{1-\varepsilon}]}, (\mathsf{FHE.PubCoin}_{i,j})_{i \in [n^{1-\varepsilon}], j \in [\ell_2]}\big)$.

- Generation of $\widetilde{\Pi}_b$: it returns $(\mathsf{pk}, \overline{\mathsf{pk}}, \overline{\mathsf{ct}}, \mathsf{ct}, (\rho_i)_{i \in [n^{1-\varepsilon}]})$ computed as described above.

When ct_1 encrypts Π^0, the reduction simulates $\mathcal{H}^{0.5}_\lambda$, whereas it simulates $\mathcal{H}^{1.5}_\lambda$ when ct_2 encrypts Π^1.

Overall, we have shown that:

$$\mathcal{D}^0_\lambda \approx_s \mathcal{H}^{0.1}_\lambda \approx_s \mathcal{H}^{0.2}_\lambda \approx_s \mathcal{H}^{0.3}_\lambda \approx_s \mathcal{H}^{0.4}_\lambda \approx_s \mathcal{H}^{0.5}_\lambda \approx_c \mathcal{H}^{1.5}_\lambda \approx_s \mathcal{H}^{1.4}_\lambda \approx_s \mathcal{H}^{1.3}_\lambda \approx_s \mathcal{H}^{1.2}_\lambda \approx_s \mathcal{H}^{1.1}_\lambda \approx_s \mathcal{D}^1_\lambda.$$

6 Concluding the Main Theorem

6.1 Instantiation with DJ LHE

Now we instantiate our modular XiO construction with the DJ LHE presented in Section 4.2 that is parameterized by a polynomial ℓ_1; we denote it by DJ_{ℓ_1}. For all polynomials δ, we denote by GSW_δ the GSW FHE scheme for depth δ circuits.

By combining Theorem 4.4 (i.e. security of DJ_{ℓ_1} for all polynomials ℓ_1 under DCR) and Theorem 3.6 (i.e. SRL-security of the GSW_δ for all polynomials δ under LWE), and noting that both of these results directly upgrade to subexponential security if we assume subexponential security of the underlying assumptions, we get:

Lemma 6.1. *Assume the (subexponential) DCR and (subexponential) LWE assumptions hold. Then, assuming that for all polynomials δ and ℓ_1, the (subexponential) $\mathsf{2CIRC}^{\mathcal{O}_{\mathsf{SRL}}}$ assumption w.r.t. GSW_δ and DJ_{ℓ_1} holds implies that for all polynomials δ and ℓ_1, (subexponential) 2-circular SRL security holds w.r.t. GSW_δ and DJ_{ℓ_1}.*

By combining Lemma 6.1 above with Theorem 4.3, which states that for all polynomials ℓ_1, DJ_{ℓ_1} is an (ℓ_1, ℓ_2, h)-hintable packed LHE with $\ell_2(\lambda) = 1$ and $h(\lambda) = 2\lambda$, together with Theorem 5.1, and noting that Theorem 5.1 yields a subexponentially-secure $Xi\mathcal{O}$ assuming the subexponential security of the underlying building block (i.e., subexponential 2-circular SRL security of GSW_δ and DJ_{ℓ_1}), we get:

Theorem 6.1. *Assume the (subexponential) DCR and (subexponential) LWE assumptions hold. Then, assuming that for all polynomials δ, ℓ_1, the (subexponential) $\mathsf{2CIRC}^{\mathcal{O}_{\mathsf{SRL}}}$ assumption holds w.r.t. GSW_δ and DJ_{ℓ_1} implies the existence of $Xi\mathcal{O}$ for $\mathsf{P}^{\log}/\mathsf{poly}$.*

Finally, combining Theorem 6.1 with Theorem 2.5 (i.e., $i\mathcal{O}$ from $Xi\mathcal{O}$ and LWE) yields out one of our two main theorem:

Theorem 6.2. *Assume the subexponential DCR and subexponential LWE assumptions hold. Then, assuming that for all polynomials δ and ℓ_1 the subexponential $\mathsf{2CIRC}^{\mathcal{O}_{\mathsf{SRL}}}$ assumption w.r.t. GSW_δ and DJ_{ℓ_1} implies the existence of $i\mathcal{O}$ for P/poly.*

6.2 Instantiation with Packed Regev LHE

We now state the results when instantiating our modular XiO construction with the Packed Regev LHE, presented in Section 4.3. This construction is parameterized by polynomials ℓ_1 and ℓ_2, and it is denoted by $\text{P-Regev}_{\ell_1,\ell_2}$.

By combining Theorem 4.6 (i.e. security of $\text{P-Regev}_{\ell_1,\ell_2}$ for all polynomials ℓ_1 and ℓ_2 under LWE) and Theorem 3.6 (i.e. SRL-security of the GSW_δ for all polynomials δ under LWE), we get:

Lemma 6.2. *Assume (subexponential) LWE assumptions hold. Then, assuming for all polynomials δ, ℓ_1 and ℓ_2, the (subexponential) $\mathsf{2CIRC}^{\mathcal{O}_{\mathsf{SRL}}}$ assumption w.r.t.* GSW_δ *and* $\text{P-Regev}_{\ell_1,\ell_2}$. *Then for all polynomials δ, ℓ_1 and ℓ_2, (subexponential) 2-circular SRL security holds w.r.t.* GSW_δ *and* P-Regev_{ℓ_2}.

By combining Lemma 6.2 above with Theorem 4.5, which states that for all polynomials ℓ_1, there exists a polynomial h such that for all polynomials ℓ_2, $\text{P-Regev}_{\ell_1,\ell_2}$ is an (ℓ_1, ℓ_2, h)-hintable packed LHE, together with Theorem 5.1, we get:

Theorem 6.3. *Assume the (subexponential) LWE assumption holds. Then, assuming that for all polynomials δ, ℓ_1 and ℓ_2 the (subexponential) $\mathsf{2CIRC}^{\mathcal{O}_{\mathsf{SRL}}}$ assumption holds w.r.t.* GSW_δ *and* $\text{P-Regev}_{\ell_1,\ell_2}$ *implies the existence of $Xi\mathcal{O}$ for $\mathsf{P}^{\log}/\mathsf{poly}$.*

Finally, combining Theorem 6.3 with Theorem 2.5 (i.e., $i\mathcal{O}$ from $Xi\mathcal{O}$ and LWE) yields out one of our two main theorem:

Theorem 6.4. *Assume the subexponential LWE assumptions hold. Then, assuming that for all polynomials δ, ℓ_1 and ℓ_2, the subexponential $\mathsf{2CIRC}^{\mathcal{O}_{\mathsf{SRL}}}$ assumption w.r.t.* GSW_δ *and* $\text{P-Regev}_{\ell_1,\ell_2}$ *implies the existence of $i\mathcal{O}$ for P/poly.*

References

[ABBC10] T. Acar, M. Belenkiy, M. Bellare, and D. Cash. Cryptographic agility and its relation to circular encryption. In *EUROCRYPT 2010*, *LNCS* 6110, pages 403–422. Springer, Heidelberg, May / June 2010.

[Agr19] S. Agrawal. Indistinguishability obfuscation without multilinear maps: New methods for bootstrapping and instantiation. In *EUROCRYPT 2019, Part I*, *LNCS* 11476, pages 191–225. Springer, Heidelberg, May 2019.

[AJ15] P. Ananth and A. Jain. Indistinguishability obfuscation from compact functional encryption. In *CRYPTO 2015, Part I*, *LNCS* 9215, pages 308–326. Springer, Heidelberg, August 2015.

[AJKS18] P. Ananth, A. Jain, D. Khurana, and A. Sahai. Indistinguishability obfuscation without multilinear maps: iO from LWE, bilinear maps, and weak pseudorandomness. Cryptology ePrint Archive, Report 2018/615, 2018. https://eprint.iacr.org/2018/615.

[AJL+19] P. Ananth, A. Jain, H. Lin, C. Matt, and A. Sahai. Indistinguishability obfuscation without multilinear maps: New paradigms via low degree weak pseudorandomness and security amplification. Cryptology ePrint Archive, Report 2019/643, 2019. https://eprint.iacr.org/2019/643.

[Ajt96] M. Ajtai. Generating hard instances of lattice problems (extended abstract). In *28th ACM STOC*, pages 99–108. ACM Press, May 1996.

[AP09] J. Alwen and C. Peikert. Generating Shorter Bases for Hard Random Lattices. In *26th International Symposium on Theoretical Aspects of Computer Science STACS 2009*, Proceedings of the 26th Annual Symposium on the Theoretical Aspects of Computer Science, pages 75–86, Freiburg, Germany, February 2009. IBFI Schloss Dagstuhl.

[AP20] S. Agrawal and A. Pellet-Mary. Indistinguishability obfuscation without maps: Attacks and fixes for noisy linear FE. In *EUROCRYPT 2020, Part I*, LNCS, pages 110–140. Springer, Heidelberg, May 2020.

[BBKK18] B. Barak, Z. Brakerski, I. Komargodski, and P. K. Kothari. Limits on low-degree pseudorandom generators (or: Sum-of-squares meets program obfuscation). In *EUROCRYPT 2018, Part II*, *LNCS* 10821, pages 649–679. Springer, Heidelberg, April / May 2018.

[BCGI18] E. Boyle, G. Couteau, N. Gilboa, and Y. Ishai. Compressing vector OLE. In *ACM CCS 2018*, pages 896–912. ACM Press, October 2018.

[BCP14] E. Boyle, K.-M. Chung, and R. Pass. On extractability obfuscation. In *TCC*, pages 52–73, 2014.

[BDGM19] Z. Brakerski, N. Döttling, S. Garg, and G. Malavolta. Leveraging linear decryption: Rate-1 fully-homomorphic encryption and time-lock puzzles. In *TCC 2019, Part II*, LNCS, pages 407–437. Springer, Heidelberg, March 2019.

[BDGM20a] Z. Brakerski, N. Döttling, S. Garg, and G. Malavolta. Candidate iO from homomorphic encryption schemes. In *EUROCRYPT 2020, Part I*, LNCS, pages 79–109. Springer, Heidelberg, May 2020.

[BDGM20b] Z. Brakerski, N. Döttling, S. Garg, and G. Malavolta. Factoring and pairings are not necessary for io: Circular-secure lwe suffices. Cryptology ePrint Archive, Report 2020/1024, 2020. https://eprint.iacr.org/2020/1024.

[BGI+01] B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S. P. Vadhan, and K. Yang. On the (im)possibility of obfuscating programs. In *CRYPTO 2001*, *LNCS* 2139, pages 1–18. Springer, Heidelberg, August 2001.

[BGL+15] N. Bitansky, S. Garg, H. Lin, R. Pass, and S. Telang. Succinct randomized encodings and their applications. *IACR Cryptology ePrint Archive*, 2015:356, 2015.

[BHJ+18] B. Barak, S. B. Hopkins, A. Jain, P. Kothari, and A. Sahai. Sum-of-squares meets program obfuscation, revisited. Cryptology ePrint Archive, Report 2018/1237, 2018. https://eprint.iacr.org/2018/1237.

[BHW15] A. Bishop, S. Hohenberger, and B. Waters. New circular security counterexamples from decision linear and learning with errors. In *ASIACRYPT 2015, Part II*, *LNCS* 9453, pages 776–800. Springer, Heidelberg, November / December 2015.

[BP15] N. Bitansky and O. Paneth. ZAPs and non-interactive witness indistinguishability from indistinguishability obfuscation. In *TCC 2015, Part II*, *LNCS* 9015, pages 401–427. Springer, Heidelberg, March 2015.

[BPR15] N. Bitansky, O. Paneth, and A. Rosen. On the cryptographic hardness of finding a Nash equilibrium. In *56th FOCS*, pages 1480–1498. IEEE Computer Society Press, October 2015.

[BPW16] N. Bitansky, O. Paneth, and D. Wichs. Perfect structure on the edge of chaos - trapdoor permutations from indistinguishability obfuscation. In *TCC 2016-A, Part I*, *LNCS* 9562, pages 474–502. Springer, Heidelberg, January 2016.

[BR93] M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM CCS 93*, pages 62–73. ACM Press, November 1993.

[BRS02] J. Black, P. Rogaway, and T. Shrimpton. Encryption-scheme security in the presence of key-dependent messages. Cryptology ePrint Archive, Report 2002/100, 2002. http://eprint.iacr.org/2002/100.

[BV15] N. Bitansky and V. Vaikuntanathan. Indistinguishability obfuscation from functional encryption. In *56th FOCS*, pages 171–190. IEEE Computer Society Press, October 2015.

[BZ14] D. Boneh and M. Zhandry. Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. In *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I*, pages 480–499, 2014.

[CGH98] R. Canetti, O. Goldreich, and S. Halevi. The random oracle methodology, revisited (preliminary version). In *30th ACM STOC*, pages 209–218. ACM Press, May 1998.

[CGH12] D. Cash, M. Green, and S. Hohenberger. New definitions and separations for circular security. In *PKC 2012*, *LNCS* 7293, pages 540–557. Springer, Heidelberg, May 2012.

[CHJV14] R. Canetti, J. Holmgren, A. Jain, and V. Vaikuntanathan. Indistinguishability obfuscation of iterated circuits and RAM programs. Cryptology ePrint Archive, Report 2014/769, 2014. http://eprint.iacr.org/2014/769.

[CHL$^+$15] J. H. Cheon, K. Han, C. Lee, H. Ryu, and D. Stehlé. Cryptanalysis of the multilinear map over the integers. In *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*, pages 3–12, 2015.

[CKP15] R. Canetti, Y. T. Kalai, and O. Paneth. On obfuscation with random oracles. In *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part II*, pages 456–467, 2015.

[CL01] J. Camenisch and A. Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In *EUROCRYPT 2001*, *LNCS* 2045, pages 93–118. Springer, Heidelberg, May 2001.

[CLP15] K.-M. Chung, H. Lin, and R. Pass. Constant-round concurrent zero-knowledge from indistinguishability obfuscation. In *CRYPTO 2015, Part I*, *LNCS* 9215, pages 287–307. Springer, Heidelberg, August 2015.

[CLT13] J.-S. Coron, T. Lepoint, and M. Tibouchi. Practical multilinear maps over the integers. In *CRYPTO 2013, Part I*, *LNCS* 8042, pages 476–493. Springer, Heidelberg, August 2013.

[CLT15] J.-S. Coron, T. Lepoint, and M. Tibouchi. New multilinear maps over the integers. In *CRYPTO 2015, Part I*, *LNCS* 9215, pages 267–286. Springer, Heidelberg, August 2015.

[DJ01] I. Damgård and M. Jurik. A generalisation, a simplification and some applications of Paillier's probabilistic public-key system. In *PKC 2001*, *LNCS* 1992, pages 119–136. Springer, Heidelberg, February 2001.

[Gen09] C. Gentry. Fully homomorphic encryption using ideal lattices. In *41st ACM STOC*, pages 169–178. ACM Press, May / June 2009.

[GGH13a] S. Garg, C. Gentry, and S. Halevi. Candidate multilinear maps from ideal lattices. In *EUROCRYPT 2013*, *LNCS* 7881, pages 1–17. Springer, Heidelberg, May 2013.

[GGH$^+$13b] S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *54th FOCS*, pages 40–49. IEEE Computer Society Press, October 2013.

[GGH15] C. Gentry, S. Gorbunov, and S. Halevi. Graph-induced multilinear maps from lattices. In *TCC 2015, Part II*, *LNCS* 9015, pages 498–527. Springer, Heidelberg, March 2015.

[GGHR14] S. Garg, C. Gentry, S. Halevi, and M. Raykova. Two-round secure MPC from indistinguishability obfuscation. In *Theory of Cryptography - 11th Theory of Cryptography Conference, TCC 2014, San Diego, CA, USA, February 24-26, 2014. Proceedings*, pages 74–94, 2014.

[GH10] M. Green and S. Hohenberger. Cpa and cca-secure encryption systems that are not 2-circular secure, 2010. matthewdgreen@gmail.com 14686 received 16 Mar 2010, last revised 18 Mar 2010.

[GJLS20] R. Gay, A. Jain, H. Lin, and A. Sahai. Indistinguishability obfuscation from simple-to-state hard problems: New assumptions, new techniques, and simplification. Technical report, Cryptology ePrint Archive, Report 2020/764, 2020. https://eprint.iacr.org/2020/764, 2020.

[GK05] S. Goldwasser and Y. T. Kalai. On the impossibility of obfuscation with auxiliary input. In *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2005), 23-25 October 2005, Pittsburgh, PA, USA, Proceedings*, pages 553–562, 2005.

[GKW17] R. Goyal, V. Koppula, and B. Waters. Separating semantic and circular security for symmetric-key bit encryption from the learning with errors assumption. In *EUROCRYPT 2017, Part II*, *LNCS* 10211, pages 528–557. Springer, Heidelberg, April / May 2017.

[GLSW14] C. Gentry, A. Lewko, A. Sahai, and B. Waters. Indistinguishability obfuscation from the multilinear subgroup elimination assumption. Cryptology ePrint Archive, Report 2014/309, 2014.

[GM84] S. Goldwasser and S. Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984.

[GPV08] C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *40th ACM STOC*, pages 197–206. ACM Press, May 2008.

[GSW13] C. Gentry, A. Sahai, and B. Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In *CRYPTO 2013, Part I*, *LNCS* 8042, pages 75–92. Springer, Heidelberg, August 2013.

[ILL89] R. Impagliazzo, L. A. Levin, and M. Luby. Pseudo-random generation from one-way functions (extended abstracts). In *21st ACM STOC*, pages 12–24. ACM Press, May 1989.

[JLMS19] A. Jain, H. Lin, C. Matt, and A. Sahai. How to leverage hardness of constant-degree expanding polynomials overa $\mathbb{R}$ to build $i\mathcal{O}$. In *EUROCRYPT 2019, Part I*, *LNCS* 11476, pages 251–281. Springer, Heidelberg, May 2019.

[JLS20] A. Jain, H. Lin, and A. Sahai. Indistinguishability obfuscation from well-founded assumptions. Cryptology ePrint Archive, Report 2020/1003, 2020. https://eprint.iacr.org/2020/1003.

[JS18] A. Jain and A. Sahai. How to leverage hardness of constant-degree expanding polynomials over $\mathbb{R}$ to build iO. Cryptology ePrint Archive, Report 2018/973, 2018. https://eprint.iacr.org/2018/973.

[KLW15] V. Koppula, A. B. Lewko, and B. Waters. Indistinguishability obfuscation for turing machines with unbounded memory. In *47th ACM STOC*, pages 419–428. ACM Press, June 2015.

[KMN$^+$14] I. Komargodski, T. Moran, M. Naor, R. Pass, A. Rosen, and E. Yogev. One-way functions and (im)perfect obfuscation. In *55th FOCS*, pages 374–383. IEEE Computer Society Press, October 2014.

[KNY14] I. Komargodski, M. Naor, and E. Yogev. Secret-sharing for NP. In *ASIACRYPT 2014, Part II*, *LNCS* 8874, pages 254–273. Springer, Heidelberg, December 2014.

[KRW15] V. Koppula, K. Ramchen, and B. Waters. Separations in circular security for arbitrary length key cycles. In *TCC 2015, Part II*, *LNCS* 9015, pages 378–400. Springer, Heidelberg, March 2015.

[KW16] V. Koppula and B. Waters. Circular security separations for arbitrary length cycles from LWE. In *CRYPTO 2016, Part II*, *LNCS* 9815, pages 681–700. Springer, Heidelberg, August 2016.

[Lin16] H. Lin. Indistinguishability obfuscation from constant-degree graded encoding schemes. In *EUROCRYPT 2016, Part I*, *LNCS* 9665, pages 28–57. Springer, Heidelberg, May 2016.

[Lin17] H. Lin. Indistinguishability obfuscation from SXDH on 5-linear maps and locality-5 PRGs. In *CRYPTO 2017, Part I*, *LNCS* 10401, pages 599–629. Springer, Heidelberg, August 2017.

[LPST16] H. Lin, R. Pass, K. Seth, and S. Telang. Indistinguishability obfuscation with non-trivial efficiency. In *PKC 2016, Part II*, *LNCS* 9615, pages 447–462. Springer, Heidelberg, March 2016.

[LT17] H. Lin and S. Tessaro. Indistinguishability obfuscation from trilinear maps and block-wise local PRGs. In *CRYPTO 2017, Part I*, *LNCS* 10401, pages 630–660. Springer, Heidelberg, August 2017.

[LV16] H. Lin and V. Vaikuntanathan. Indistinguishability obfuscation from DDH-like assumptions on constant-degree graded encodings. In *57th FOCS*, pages 11–20. IEEE Computer Society Press, October 2016.

[MF15] B. Minaud and P.-A. Fouque. Cryptanalysis of the new multilinear map over the integers. Cryptology ePrint Archive, Report 2015/941, 2015. http://eprint.iacr.org/.

[Mic19] D. Micciancio. From linear functions to fully homomorphic encryption. https://bacrypto.github.io/presentations/2018.11.30-micciancio-fhe.pdf. Technical report, 2019.

[MMN15] M. Mahmoody, A. Mohammed, and S. Nematihaji. More on impossibility of virtual black-box obfuscation in idealized models. *IACR Cryptology ePrint Archive*, 2015:632, 2015.

[MO14] A. Marcedone and C. Orlandi. Obfuscation $\Rightarrow$ (IND-CPA security $\not\Rightarrow$ circular security). In *SCN 14*, *LNCS* 8642, pages 77–90. Springer, Heidelberg, September 2014.

[MP12] D. Micciancio and C. Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In *EUROCRYPT 2012*, *LNCS* 7237, pages 700–718. Springer, Heidelberg, April 2012.

[MRH04] U. M. Maurer, R. Renner, and C. Holenstein. Indifferentiability, impossibility results on reductions, and applications to the random oracle methodology. In *TCC 2004*, *LNCS* 2951, pages 21–39. Springer, Heidelberg, February 2004.

[Pai99] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *EUROCRYPT'99*, *LNCS* 1592, pages 223–238. Springer, Heidelberg, May 1999.

[PRS17] C. Peikert, O. Regev, and N. Stephens-Davidowitz. Pseudorandomness of ring-LWE for any ring and modulus. In *49th ACM STOC*, pages 461–473. ACM Press, June 2017.

[Ps16] R. Pass and a. shelat. Impossibility of VBB obfuscation with ideal constant-degree graded encodings. In *TCC 2016-A, Part I*, *LNCS* 9562, pages 3–17. Springer, Heidelberg, January 2016.

[PST14] R. Pass, K. Seth, and S. Telang. Indistinguishability obfuscation from semantically-secure multilinear encodings. In *CRYPTO 2014, Part I*, *LNCS* 8616, pages 500–517. Springer, Heidelberg, August 2014.

[PVW08] C. Peikert, V. Vaikuntanathan, and B. Waters. A framework for efficient and composable oblivious transfer. In *CRYPTO 2008*, *LNCS* 5157, pages 554–571. Springer, Heidelberg, August 2008.

[Reg05] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In *37th ACM STOC*, pages 84–93. ACM Press, May 2005.

[SW14] A. Sahai and B. Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In *46th ACM STOC*, pages 475–484. ACM Press, May / June 2014.

[WW20] H. Wee and D. Wichs. Candidate obfuscation via oblivious lwe sampling. Cryptology ePrint Archive, Report 2020/1042, 2020. https://eprint.iacr.org/2020/1042.