Factoring and Pairings are not Necessary for iO: Circular-Secure LWE Suffices

Zvika Brakerski$^{\star 1}$, Nico Döttling^{2}, Sanjam Garg$^{\star\star 3}$, and Giulio Malavolta^{4}

[1]Weizmann Institute of Science
[2]CISPA Helmoltz Center for Information Security
[3]UC Berkeley
[4]Max Planck Institute for Security and Privacy

Abstract. We construct indistinguishability obfuscation (iO) solely under circular-security properties of encryption schemes based on the Learning with Errors (LWE) problem. Circular-security assumptions were used before to construct (non-leveled) fully-homomorphic encryption (FHE), but our assumption is stronger and requires circular randomness-leakage-resilience. In contrast with prior works, this assumption can be conjectured to be post-quantum secure; yielding the first provably secure iO construction that is (plausibly) post-quantum secure.

Our work follows the high-level outline of the recent work of Gay and Pass [STOC 2021], who showed a way to remove the heuristic step from the homomorphic-encryption based iO approach of Brakerski, Döttling, Garg, and Malavolta [EUROCRYPT 2020]. They thus obtain a construction proved secure under circular security assumption of natural homomorphic encryption schemes — specifically, they use homomorphic encryption schemes based on LWE and DCR, respectively. In this work we show how to remove the DCR assumption and remain with a scheme based on the circular security of LWE alone. Along the way we relax some of the requirements in the Gay-Pass blueprint and thus obtain a scheme that is secure under a relaxed assumption. Specifically, we do not require security in the presence of a key-cycle, but rather only in the presence of a key-randomness cycle.

An additional contribution of our work is to point out a problem in one of the building blocks used by many iO candidates, including *all* existing provable post-quantum candidates. Namely, in the transformation from exponentially-efficient iO (XiO) from Lin, Pass, Seth and Telang [PKC 2016]. We show why their transformation inherently falls short of achieving the desired goal, and then rectify this situation by showing that *shallow* XiO (i.e. one where the obfuscator is depth-bounded) does translate to iO using LWE.

1 Introduction

The goal of program obfuscation [Had00, BGI$^+$01] is to transform an arbitrary circuit Π into an unintelligible but functionally equivalent circuit $\tilde{\Pi}$. The aforementioned works showed that strong simulation-based notions of obfuscation were impossible for general purpose functionalities. However, the seemingly weaker *indistinguishability obfuscation* (iO) was not ruled out by prior work (and has in fact been shown to be the same as the best possible notion of obfuscation [GR07]). In broad terms, iO requires that if two circuits Π_0 and Π_1 are two implementations of the same function, then their obfuscations are computationally indistinguishable.

* Supported by the Binational Science Foundation (Grant No. 2016726), and by the European Union Horizon 2020 Research and Innovation Program via ERC Project REACT (Grant 756482) and via Project PROMETHEUS (Grant 780701).

** Supported in part from AFOSR Award FA9550-19-1-0200, AFOSR YIP Award, NSF CNS Award 1936826, DARPA and SPAWAR under contract N66001-15-C-4065, DARPA/ARL SAFEWARE Award W911NF15C0210, a Hellman Award and research grants by the Okawa Foundation, Visa Inc., and Center for Long-Term Cybersecurity (CLTC, UC Berkeley). The views expressed are those of the authors and do not reflect the official policy or position of the funding agencies.

Garg et al. [GGH13a, GGH$^+$13b] presented the first candidate for general purpose iO, paving the way for numerous other candidates based on a variety of mathematical structures. Although iO appears to be a weak notion of security, it has been shown to be sufficient for numerous cryptographic applications, including ones that were previously not known to exist under other assumptions (see [SW14, GGHR14, BZ14] for examples). The first realizations of obfuscation relied an a new algebraic object called multilinear maps [GGH13a, CLT13, GGH15], which had only recently been constructed. Furthermore, the security of these objects relied on new (and poorly understood) computational intractability assumptions, or more commonly on plain heuristics. In fact, several attacks on multilinear map candidates [CHL$^+$15, HJ16] and on obfuscation constructions based on multilinear maps [MSZ16, CGH17] were demonstrated. To defend against these attacks, several safeguards have been (e.g., [GMM$^+$16, CVW18, MZ18, BGMZ18, DGG$^+$18]) proposed. Even with these heuristic safeguards, all but the schemes based on the Gentry et al. [GGH15] multilinear maps are known to broken against quantum adversaries.

Towards the goal of avoiding heuristics and obtaining provably secure constructions, substantial effort was made towards obtaining iO while minimizing (with the ultimate goal of removing) the use of multilinear maps [Lin16, LV16, AS17, Lin17, LT17]. These efforts culminated in replacing the use of multilinear maps with just bilinear maps [Agr19, JLMS19, AJL$^+$19], together with an additional pseudorandom generators of constant locality over the integers with polynomial stretch. Very recently this last limitation was removed by Jain, Lin and Sahai [JLS20]. Specifically, they obtained iO based on the combined (sub-exponential) hardness of the Learning with Errors problem (LWE), a large-modulus variant of the Learning Parity with Noise problem (LPN), the existence of a pseudorandom generator in NC_0, and in addition the hardness of the external Diffie-Hellman problem in bilinear groups (SXDH). We note that the use of the pairings makes these construction insecure against quantum adversaries.

A different approach towards provably secure iO, which is more relevant to this work, was presented by Brakerski et al. [BDGM20]. They showed an iO candidate that is based on combining certain natural *homomorphic* encryption schemes. However, their construction was *heuristic* in the sense that the security argument could only be presented in the random oracle model. In a recent work, Gay and Pass [GP20] showed a way to remove the heuristic step and instead rely on a concrete assumption. Their construction is proved secure under the circular security of natural homomorphic encryption schemes — specifically, they use homomorphic encryption schemes based on LWE and Decisional Composite Residuosity (DCR, also known as Paillier's assumption). In terms of assumptions, their construction assumes sub-exponential security of (i) the Learning with Error (LWE) assumption, (ii) the Decisional Composite Residuosity (DCR) assumption, and (iii) a new notion of security that they call "shielded randomness leakage" (SRL). The latter essentially requires that a fully homomorphic encryption scheme (specifically the GSW encryption scheme [GSW13]) remains secure even in the presence of a key-cycle with the Damgård-Jurik encryption scheme [DJ01]. Moreover, the notion of security is not the standard semantic security, but rather a new notion of security with respect to leakage of ciphertext randomness. We note that this construction is insecure against quantum attackers because of the use of the Damgård-Jurik encryption scheme [DJ01].[1] In this work, we ask:

Can we realize provably secure constructions of iO with (plausible) post-quantum security?

1.1 Our results

We obtain a general purpose iO construction based solely on the circular security of LWE-based encryption schemes. On a technical level, we achieve this by introducing a "packed" variant of the dual-Regev LWE-based encryption scheme, and showing novel ways of manipulating ciphertexts of this variant in conjunction with ciphertexts of an FHE scheme. This allows us to remove the need for DCR-based encryption from the construction of [BDGM20, GP20]. Furthermore, our technique allows us to relax the SRL security property that is required, so that we no longer need to require SRL security with respect to a key-cycle, but rather only with respect to a key-randomness cycle. We put forth this potentially weaker assumption as an object for further study.

[1] Later, [GP20] updated their manuscript to also include a solution based on LWE. See Section 1.3 for additional discussion.

More concretely, the circular security assumption made in [GP20], and thus also in this work, is that a scheme (in particular a leveled FHE scheme) maintains this property even in the presence of some leakage on the randomness of the ciphertext. In [GP20] it is shown that standard GSW encryption [GSW13] satisfies SRL security (under the LWE assumption), and the additional assumption is therefore that SRL security is maintained in the presence of a key-randomness cycle, connecting GSW to another encryption scheme. While this assumption falls into the category of "circular security assumptions", similarly to the ones that underlie bootstrapping in FHE, the concrete assumption is quite different. While in the FHE setting it was only assumed that (standard) CPA security is preserved given a key cycle, here we assume that the stronger SRL property remains intact.

Let us now state our results somewhat more precisely.

Theorem 1 (Informal). *Assume the (sub-exponential) hardness of the LWE problem, and the SRL security of GSW in the presence of a randomness-key cycle with a packed variant of dual-Regev, then there exists indistinguishability obfuscation for all circuits.*

We note that if we further assume that circular security also maintains post-quantum security, then our assumption becomes post-quantum secure; yielding the first provably secure iO construction that is post-quantum secure.

Shallow XiO. As an additional contribution, we identify a gap in the transformation of "exponentially efficient iO" (XiO), a notion introduced by Lin, Pass, Seth and Telang [LPST16a] that was used almost universally in prior work. We show that this transformation has an inherent problem that does not allow to recover the result as stated. This gap affects most known iO constructions and, in particular, *all post-quantum provably secure candidates*. We rectify this situation by showing that a fairly simple technical modification (i.e. constraining the compiler to be *shallow*) allows us to recover the prior results. Along the way, we develop a framework for analyzing composition of compressing encodings, which can be a useful perspective for future research in this area.

1.2 Technical Overview

We now provide a technical outline of our construction and its properties.

Obfuscation via Homomorphic Encryption. The connection between (fully) homomorphic encryption and obfuscation is fairly straightforward. Given a program Π to be obfuscated, we can provide a ciphertext c_Π which encrypts Π under an FHE scheme. This will allow to use homomorphism to derive $c_x = \mathsf{Enc}(\Pi(x))$ for all x. Now all that is needed is a way to decrypt c_x in a way that does not reveal any information on Π. Early works (e.g. [GGH+13b] and followups) attempted to use this approach and provide a "defective" version of the secret key of the FHE scheme, but a different approach was suggested in [BDGM20].

Specifically, [BDGM20] considered a homomorphic evaluation that takes c_Π to c_{TT}, an encryption of the *entire truth table* of Π, i.e. to an encryption of a multi-bit value. By relying on prior generic transformations [LPST16a], they showed that one can reduce the task of constructing general-purpose obfuscation to the task of computing a "decryption" hint for c_{TT} with the following properties:

- Succinctness: The size of the decryption hint must be sublinear in the size of the truth table $|\mathsf{TT}|$.
- Simulatability: The decryption hint should not reveal any additional information besides the truth table TT.

The reason why this is helpful is that some so-called "packed-encryption" schemes have the property that a short ciphertext-dependent decryption hint suffices in order to decrypt the ciphertext, in a way that does not seem to leak the secret key of the scheme itself. While standard FHE schemes do not natively support packed encryption, it was shown in [BDGM19] that it is possible to use the so-called key-switching technique to switch from an FHE scheme into a packed-encryption scheme.

Alas, when instantiating the components of the [BDGM20] approach in its simplistic form described above, the decryption hint leaks information that renders the scheme insecure. To counter this issue, [BDGM20] proposed to inject another source of randomness: By adding freshly sampled ciphertexts of the packed-encryption scheme (which in their case was instantiated with the Damgård-Jurik scheme [DJ01]) one can smudge the leakage of the decryption hint. However the size of these fresh ciphertext would largely exceed the size of the truth table TT. Therefore, [BDGM20] proposed to heuristically sample them from a random oracle, leveraging the fact that the ciphertexts of [DJ01] are *dense*, i.e. a uniformly sampled string lies in the support of the encryption algorithm with all but negligible probability. This led to a candidate, but without a proof of security.

A Provably Secure Scheme. In a recent work, Gay and Pass [GP20] observed that for the purpose of constructing obfuscation, it suffices to consider schemes in the common random string (CRS) model where, importantly, the size of the CRS can exceed the size of the truth table. This allowed them to place the Damgård-Jurik ciphertexts in the CRS and therefore avoid relying on random-oracle-like heuristics.

They propose a new method to prove the security of this approach: Leveraging the structural property of the GSW scheme [GSW13]. They showed that adding a GSW encryption of 0 to the evaluated FHE ciphertext (before key-switching to Damgård-Jurik) allows one to program the FHE ciphertext in the security proof. To sample these GSW encryptions of 0, they propose to draw the random coins $\mathbf{r}^*$ again from the CRS and let the evaluator recompute the correct ciphertext $\mathsf{GSW.Enc}(0; \mathbf{r}^*)$.

Taken together, these new ideas allow them to prove their construction secure against the shielded randomness leakage (SRL) security of the resulting FHE scheme. Loosely speaking, SRL security requires that semantic security of an encryption scheme is retained in the presence of an oracle that leaks the randomness $\mathbf{r}_f$ of the homomorphic evaluation of the function f over the challenge ciphertext. However the randomness $\mathbf{r}_f$ is not revealed in plain to the adversary, instead it is "shielded" by the random coins of a fresh GSW ciphertext $c = \mathsf{GSW.Enc}(0; \mathbf{r}^*)$. That is, the adversary is given $(\mathbf{r}_f - \mathbf{r}^*, c)$. In fact, the adversary can obtain polynomially-many samples from this distribution, for any function f, conditioned on the fact that the adversary knows the output of $f(m^*)$, where m^* is the hidden message.

To gain confidence in the veracity of the assumption, [GP20] show that the GSW encryption scheme satisfies SRL security if the (plain) LWE assumption holds. However, their obfuscation scheme requires one to publish a key cycle of GSW and Damgård-Jurik (i.e. an encryption of the GSW secrey key under Damgård-Jurik and vice versa). Thus their final assumption is that SRL security is retained in the presence of such a key cycle.

Obfuscation from Circular-Secure LWE. We wish to remove the need for the Damgård-Jurik encryption scheme from the above construction paradigm. The major obstacle to overcome consists is designing an LWE-based encryption scheme that simultaneously satisfies three properties.

– Linear Homomorphism: In order to key switch the GSW ciphertext into this form, the scheme must satisfy some weak notion of homomorphism. Specifically, it must support the homomorphic evaluation of linear functions.
– Succinct Randomness: The scheme must allow us to encrypt a long message string with a short randomness, that can then function as the decryption hint.
– Dense Ciphertexts: A uniformly sampled string must lie in the support of the encryption algorithm with all but negligible probability. This will allow us to parse the CRS as a collection of ciphertexts.[2]

Unfortunately all natural lattice-based candidates seem to fail to satisfy all of these properties. In particular, for all LWE-based schemes linear homomorphism seems to be at odds with dense ciphertexts: To ensure that the noise accumulated during the homomorphic evaluation does not impact the decryption correctness, one needs to ensure a gap between the noise bound and the modulus. More concretely, ciphertext are typically of the form $(\mathsf{a}, \mathsf{a} \cdot \mathsf{s} + e + q/2 \cdot \mathsf{m}) \in \mathbb{Z}_q^{n+1}$ where $e \ll q$, which makes them inherently sparse.

Our Solution: A Packed Variant of Dual-Regev that is also Dense-Friendly. We show that the above requirements can be relaxed. Our starting point is devising a "packed" version of the dual-Regev

[2] Note that for the purpose of constructing the obfuscator, one could make do with a common reference string which can have an arbitrary distribution. However, the string needs to be parsed as a ciphertext with respect to *all* public-keys. Requiring dense ciphertexts is a simple requirement that implies this property.

encryption scheme [GPV08]. This scheme will not have dense ciphertexts so it does not fit the requirements from previous works. However, we will show how we can define, for the same scheme, a family of ciphertexts which are both "almost dense" and can inter-operate with the non-dense scheme, so as to allow to construct the obfuscator.

Let us start with our packed dual-Regev scheme. To pack a k-bit plaintext $\mathbf{m} \in \{0,1\}^k$ in a dual-Regev ciphertext we construct the public key as a matrix $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$, which is statistically close to uniform but is sampled together with a trapdoor τ (whose role will be explained below), and another uniformly sampled matrix $\mathbf{B} \in \mathbb{Z}_q^{k \times n}$. The encryption algorithm computes a the ciphertext as

$$(\mathbf{A} \cdot \mathbf{r} + \mathbf{e}_0, \mathbf{B} \cdot \mathbf{r} + q/2 \cdot \mathbf{m} + \mathbf{e})$$

where $\mathbf{r} \leftarrow_{\$} \mathbb{Z}_q^n$ is the encryption randomness and the vectors $\mathbf{e}_0$ and $\mathbf{e}$ are the encryption noises, where the norm of both vectors is bounded by some $B \ll q$. The property of the trapdoor τ is that it allows to recover $\mathbf{r}$ from $\mathbf{A} \cdot \mathbf{r} + \mathbf{e}_0$. The (semantic) security of the scheme follows directly by definition of LWE. To decrypt, therefore, one can first use the trapdoor τ to recover $\mathbf{r}$ from the first m elements of the ciphertext, and then recompute the mask $\mathbf{B} \cdot \mathbf{r}$ and recover each individual bit by rounding to the closest multiple of $q/2$. Setting the parameters appropriately, we can guarantee that the decryption is always successful. One important property of this scheme is that the random coins $\mathbf{r} \in \mathbb{Z}_q^n$ are sufficient to recover the entire message and furthermore the size of $\mathbf{r}$ is succinct (in particular independent of k).

In terms of homomorphism, the scheme is straightforwardly additively homomorphic. Furthermore, it supports key switching from any scheme with almost-linear decryption as per [BDGM19].[3] In particular it is possible to take a (long) message encrypted under an FHE scheme such as GSW and convert it to an encryption of the same message under packed dual-Regev, using precomputed key-switching parameters.[4]

As explained above, this scheme does not have dense ciphertexts. At this point we make two crucial observations that will allow us to bypass this hurdle.

(1) In order to construct the obfuscator using the [BDGM20] approach, dense ciphertexts only need to enjoy a very limited form of homomorphism, they only need to support a single addition with a non-dense ciphertext.

This is essentially because the obfuscator has the following outline. It starts by considering the dense ciphertext from the CRS (or oracle in the case of the original [BDGM20]), and homomorphically bootstraps it into a non-dense FHE ciphertext by evaluating the decryption circuit. Let $\mathbf{m}$ be the (random) message that is induced by the process. Then, the FHE encryption of $\mathbf{m}$ is processed in order to create a non-dense *packed* encryption of $\mathbf{m} \oplus \mathsf{TT}$, where TT is the truth table of the program to be obfuscated (or, more accurately, a chunk of this truth table, partitioning into chunks is required in order to allow reusability of the keys). Then a single homomorphic addition between the dense and non-dense ciphertext would imply a packed encryption of the truth table. All of this can be performed by the evaluator of the obfuscated program, so all that is needed is the decryption hint for this final ciphertext, that would allow to recover TT.

We note importantly, that in prior approaches (including the [GP20] blueprint) the aforementioned bootstrapping creates a key cycle, since a packed ciphertext is bootstrapped into an FHE ciphertext, which is afterwards key-switched into a packed ciphertext. However, we notice that it suffices to provide an encryption of the (succinct) *randomness* of the dense ciphertext in order to apply bootstrapping, thus leading to a relaxed key-randomness circular assumption. Interestingly, this observation is not very useful for actual dense ciphertexts (since finding the randomness would require using the key), however, our relaxed notion of density described below will allow to apply it and thus relax the circularity notion as well.

(2) A notion of *almost-everywhere* density suffices. A ciphertext distribution is almost-everywhere dense if it is dense except for a non-dense part whose length is independent of k (the message length).

[3] This is done using the by-now-standard technique of encrypting powers-of-two of the elements of the secret key of the latter scheme, so that it is possible to evaluate any inner product homomorphically.

[4] We note that the key switching parameters are quite long so it is required for our method that they are reusable.

The reason that this is sufficient is that the non-dense part of the ciphertext, which we refer to as the *header*, can be generated by the obfuscator and provided to the evaluator as a part of the obfuscated program. Since the header is short, and in particular the message length k can be selected to be much longer than the header, the effect on the length of the obfuscated program will be minimal. As hinted above, since the obfuscator generates the header, it in particular also samples the randomness for the final almost-everywhere dense ciphertext. This means that the obfuscator can generate the bootstrapping parameters using this randomness without requiring a key cycle.

Dense Encryption Mode. With these observations in mind we describe an alternative encryption mode ($\mathsf{DenseEnc}$) for the packed variant of dual-Regev where the bulk of the ciphertext is dense. On input a message $\mathbf{m} \in \{0,1\}^k$, the encryption algorithm in dense mode computes the following ciphertext

$$(\mathbf{A} \cdot \mathbf{r} + \mathbf{e}_0, \mathbf{B} \cdot \mathbf{r} + q/2 \cdot \mathbf{m} + \mathbf{u})$$

where $\mathbf{r}$ and $\mathbf{e}_0$ are sampled as before and $\mathbf{u} \leftarrow_{\$} [-q/4, +q/4]^k$. For convenience, we are going to split the ciphertexts into two blocks: The header $\mathbf{h}_0 \in \mathbb{Z}_q^m$ and the message carrier $(h_1, \ldots, h_k) \in \mathbb{Z}_q^k$. Foremost, observe that the decryption algorithm as described before still returns the correct message with probability 1, since it recovers the same $\mathbf{r}$ from $\mathbf{h}_0$. Furthermore, note that (for a fixed header) all vectors $(h_1, \ldots, h_k) \in \mathbb{Z}_q^k$ are in the support of the encryption algorithm. Since $k \gg m$, most of the elements of the ciphertext in the alternative encryption mode are dense.

One can verify that the aforementioned limited form of homomorphism indeed holds, namely that

$$\mathsf{dR.Enc}(\mathbf{m}) + \mathsf{dR.DenseEnc}(\mathbf{m}') \in \mathsf{dR.DenseEnc}(\mathbf{m} \oplus \mathbf{m}').$$

This is the case since

$$\begin{aligned}
&(\mathbf{A} \cdot \mathbf{r} + \mathbf{e}_0, \mathbf{B} \cdot \mathbf{r} + q/2 \cdot \mathbf{m} + \mathbf{e}) + (\mathbf{A} \cdot \mathbf{r}' + \mathbf{e}_0', \mathbf{B} \cdot \mathbf{r}' + q/2 \cdot \mathbf{m}' + \mathbf{u}) \\
&= (\mathbf{A} \cdot (\mathbf{r} + \mathbf{r}') + \mathbf{e}_0 + \mathbf{e}_0', \mathbf{B} \cdot (\mathbf{r} + \mathbf{r}') + q/2 \cdot (\mathbf{m} \oplus \mathbf{m}') + \mathbf{e} + \mathbf{u}) \\
&= (\mathbf{A} \cdot \tilde{\mathbf{r}} + \tilde{\mathbf{e}}_0, \mathbf{B} \cdot \tilde{\mathbf{r}} + q/2 \cdot (\mathbf{m} \oplus \mathbf{m}') + \tilde{\mathbf{u}})
\end{aligned}$$

where $\tilde{\mathbf{u}} = \mathbf{e} + \mathbf{u} \in [-q/4, +q/4]^k$ with all but negligible probability over the random choice of $\mathbf{u}$, for an appropriate choice of the parameters.

Doing Away with the Header. We notice that given our two observations above, the goal of the header in the obfuscation scheme is quite minimal. The header is not needed for homomorphism, and is only needed for the purpose of extracting the randomness $\mathbf{r}$ at decryption time. We then observe that decrypting packed ciphertext is done in two contexts in the scheme. The first is when we bootstrap the almost-everywhere dense ciphertext into an FHE ciphertext, and the other is when the evaluator of the obfuscated program recovers TT from the final ciphertext. For the latter there is no need for a header since the decryption hint, i.e. the respective $\mathbf{r}$ value, is provided within the obfuscated program. For the former we do not need a header of a specific structure, but rather simply an encryption of $\mathbf{r}$ that allows bootstrapping the almost-dense ciphertext. It therefore suffices to provide $\mathsf{GSW.Enc}(\mathbf{r})$ directly, which makes the header completely redundant.

On the Assumption. Equipped with the newly developed packed version of dual-Regev we can follow the [BDGM20, GP20] approach, with the aforementioned modifications, to construct the obfuscator. The resulting construction can be shown secure against the assumption that the SRL security of GSW is retained in the presence of a key cycle with the packed dual-Regev encryption scheme as presented above.

We then observe that it suffices to assume SRL security with respect to key-randomness cycles, rather than key cycles. We note that this assumption is no-stronger than key-cycle SRL since given a key-cycle it is possible to homomorphically generate a key-randomness cycle, but the converse is not known to be true.

Adding this to our observation about the redundancy of the header, the assumption we require is that SRL security is retained in the presence of a key-randomness cycle between GSW and packed dual-Regev, i.e.

$$(\mathsf{GSW.Enc}(\mathbf{r}), \mathsf{dR.Enc}(\mathsf{sk}_{\mathsf{GSW}}; \mathbf{r}))\,.$$

Since dual-Regev is randomness recoverable, this assumption is syntactically weaker than SRL security in the presence of a key-cycle: Given a GSW encryption of the dual-Regev secret key, one can homomorphically compute the randomness recovery circuit to obtain a GSW encryption of the randomness $\mathbf{r}$.

1.3 Related and Follow-up Work

Subsequently to the posting of this manuscript online (but concurrently and independently) [GP20] updated their manuscript to include a solution based on LWE in the place of DCR. They do not make the observations that a relaxed notion of density suffices (and is preferable) and thus they explicitly construct an encryption scheme with dense ciphertexts based on the (primal) Regev encryption scheme. The resulting scheme is more involved and in particular requires the two-key circular SRL security of GSW and (primal) Regev rather than the relaxed key-randomness circularity notion.

Wee and Wichs [WW20], again concurrently, presented another instantiation of the [BDGM20] approach which is arguably post-quantum secure. They rely on an indistinguishability assumption between two distributions and not directly on circular security. However, the underlying machinery developed shares many similarities with our approach. Specifically, while we essentially rely on randomness that is embedded in the CRS by interpreting it as an obliviously sampled ciphertext (which thus corresponds to one encrypted with fresh randomness), their approach is to use a pseudorandom function to transform the CRS into a randomizer for the output hint.

A follow-up work by Hopkins, Jain, and Lin [HJL21] shows counterexamples to SRL security for general functions in the presence of a 2-key cycle, as stated in [GP20], and the conjecture from [WW20]. We stress that their findings do not imply an attack against the corresponding obfuscation scheme of [WW20] and [GP20] (as also pointed out by the authors in [HJL21]). Rather, their results show that the veracity of SRL security depends on the concrete circuit representation of the functions under consideration. As a consequence of their findings, we updated the statement of our assumption (and adapted the analysis of our scheme) with a refined version, that further restricts the power of the adversary and more tightly characterize the security of our construction. However, the iO construction is unchanged from previous versions of this work.

A few remarks about the susceptibility of our scheme to the [HJL21] attack are in order. In short, the attack exploits the randomness homomorphism of GSW to compute a biased leakage. The SRL function consists of a bootstrapping followed by a modular reduction (modulo 2). On the other hand, our admissible class of leakage functions consists of linear functions (modulo q) followed by a rounding (i.e. outputting the most significant bit). We are not aware of a method to establish the same correlations exploited by [HJL21] without violating the admissibility criteria for the leakage functions. Thus, we conjecture that SRL security with respect to such leakage function holds for all natural FHE candidates (see Section 3.1 for further details).

2 Preliminaries

We denote by $\lambda \in \mathbb{N}$ the security parameter. We say that a function negl is negligible if it vanishes faster than any inverse polynomial. Given a set S, we denote by $s \leftarrow_{\$} S$ the uniform sampling from S. We say that an algorithm is PPT if it can be implemented by a probabilistic Turing machine M running in time $\mathsf{poly}(\lambda)$. The execution of a Turing machine M on input x and with random coins fixed to r is denoted by $M(x; r)$. We say that two distributions (D_0, D_1) are computationally (statistically, resp.) indistinguishable if for all PPT (unbounded, resp.) distinguishers, the probability to tell D_0 an D_1 apart is negligible. Matrices are denoted by $\mathbf{M}$ and vectors are denoted by $\mathbf{v}$. For convenience, we define $\mathbf{Bit}(\cdot)$ as the bit decomposition operation. We denote the infinity norm of a vector $\mathbf{v}$ by $\|\mathbf{v}\|_\infty$. We recall the smudging lemma.

Lemma 1 (Smudging). *Let $B_1 = B_1(\lambda)$ and $B_2 = B_2(\lambda)$ be positive integers and let $e_1 \in [-B_1, B_1]$ be a fixed integer. Let $e_2 \leftarrow_{\$} [-B_2, B_2]$ chosen uniformly at random. Then the distribution of e_2 is statistically indistinguishable to that of $e_2 + e_1$ as long as $B_1/B_2 = \mathsf{negl}(\lambda)$.*

2.1 Indistinguishability Obfuscation

We recall the notion of indistinguishability obfuscation (iO) from [BGI+01].

Definition 1 (Indistinguishability Obfuscation). *A PPT machine* iO *is an indistinguishability obfuscator for a circuit class* $\{\mathfrak{C}_\lambda\}_{\lambda\in\mathbb{N}}$ *if the following conditions are satisfied:*
(Functionality) For all $\lambda \in \mathbb{N}$*, all circuit* $\Pi \in \mathfrak{C}_\lambda$*, all inputs* x *it holds that:* $\tilde{\Pi}(x) = \Pi(x)$*, where* $\tilde{\Pi} \leftarrow_{\$} \mathsf{iO}(\Pi)$*.*
(Indistinguishability) For all $\lambda \in \mathbb{N}$*, all pairs of circuit* $(\Pi_0, \Pi_1) \in \mathfrak{C}_\lambda$ *such that* $|\Pi_0| = |\Pi_1|$ *and* $\Pi_0(x) = \Pi_1(x)$ *on all inputs* x*, it holds that the following distributions are computationally indistinguishable:* $\mathsf{iO}(\Pi_0) \approx \mathsf{iO}(\Pi_1)$*.*

Shallow XiO. In this work we construct a weaker version of iO called (shallow) XiO, which however is sufficient (along with the LWE assumption) to construct fully-fledged iO. Loosely speaking, a shallow XiO is a indistinguishability obfuscator (with pre-processing) for $\mathsf{P}^{\mathsf{log}}/\mathsf{poly}$ with non-trivial efficiency. Here $\mathsf{P}^{\mathsf{log}}/\mathsf{poly}$ denotes the class of polynomial-size circuits with inputs of length $\eta = O(\log(\lambda))$ and by non-trivial efficiency we mean that the size of the obfuscated circuit is bounded by $\mathsf{poly}(\lambda, |\Pi|) \cdot 2^{\eta\cdot(1-\varepsilon)}$, for some constant $\varepsilon > 0$. The runtime of the obfuscator can be any polynomial in λ, $|\Pi|$, and 2^η, except that its depth should not depend on 2^η. Furthermore, we allow the obfuscator to access a large uniform random string (the pre-processing) of size even larger than the truth table of the circuit. For a formal statement, we refer the reader to Appendix C.

2.2 The GSW Fully-Homomorphic Encryption

In the following we briefly recall the encryption scheme by Gentry, Sahai, and Waters [GSW13] (henceforth, GSW). We denote by $n = n(\lambda)$ the lattice dimension and by $q = q(\lambda)$ the modulus (which we assume for simplicity to be even). Throughout the rest of this paper, we set $m = (n+1)(\log(q)+1)$ and $d = d(\lambda)$ as a bound on the depth of the arithmetic circuit to be evaluated.

$\underline{\mathsf{KeyGen}(1^\lambda)\text{:}}$ Sample a uniform matrix $\mathbf{A} \leftarrow_{\$} \mathbb{Z}_q^{n\times m}$ and a vector $\mathbf{s} \leftarrow_{\$} \chi^n$. Set the public key to $(\mathbf{A}, \mathbf{b} = \mathbf{s}^T\mathbf{A} + \mathbf{e}^T)$, where $\mathbf{e} \leftarrow_{\$} \chi^m$. The secret key is set to $(-\mathbf{s}, 1)$.

$\underline{\mathsf{Enc}(\mathsf{pk}, \mathsf{m})\text{:}}$ On input a message $\mathsf{m} \in \{0,1\}$, sample a uniform $\mathbf{R} \leftarrow_{\$} \{0,1\}^{m\times m}$ and compute

$$\mathbf{C} = (\mathbf{A}, \mathbf{b}) \cdot \mathbf{R} + \mathsf{m} \cdot \mathbf{G}$$

where $\mathbf{G} = (1, 2, \dots, 2^{\log(q)-1})^T \otimes I_{(n+1)}$ and $I_{(n+1)} \in \{0,1\}^{(n+1)\times(n+1)}$ denotes the identity matrix.

$\underline{\mathsf{Eval}(\mathsf{pk}, \Pi, (c_1, \dots, c_\mu))\text{:}}$ There exists a (deterministic) polynomial-time algorithm that allows one to compute any d-bounded depth arithmetic circuit $\Pi : \{0,1\}^n \to \{0,1\}$ homomorphically over a vector of ciphertexts $(c_1, \dots, c_\mu)$. For details about this algorithm, we refer the reader to [GSW13]. For the purpose of this work, the only relevant information is that the evaluated ciphertext $\mathbf{c}_\Pi \in \mathbb{Z}_q^{(n+1)}$ is an $(n+1)$-dimensional vector. For multiple bits of output, the resulting ciphertext is defined to be the concatenation of the single-bit ciphertexts.

$\underline{\mathsf{Dec}(\mathsf{sk}, \mathbf{c})\text{:}}$ We assume without loss of generality that the input ciphertext $\mathbf{c} \in \mathbb{Z}_q^{(n+1)}$ is the output of the evaluation algorithm. Such a ciphertext defines a linear function $\boldsymbol{\ell}_{\mathbf{c}}$ such that

$$\boldsymbol{\ell}_{\mathbf{c}}(\mathsf{sk}) = q/2 \cdot \mathsf{m} + e$$

where $|e| \le \hat{B} = (m+1)^d mB$. The message m is recovered by returning the most significant bit of the output.

Note that the decryption routine of GSW consists of the application of a linear function, followed by a rounding and we refer to this property as to *almost-linear* decryption. In a slight abuse of notation, we sometimes write $\mathsf{KeyGen}(1^\lambda; q)$ to denote the above key generation algorithm with a fixed modulus q.

Alternate Encryption. For convenience we also define a modified encryption algorithm, where the output ciphertexts consists of a single column vector. An additional difference is that we sample the randomness with norm $\tilde{B} = 2^\lambda \cdot \hat{B}$.

$\underline{\mathsf{ColEnc}(\mathsf{pk}, \mathsf{m})}$**:** On input a message m, sample a uniform $\mathbf{r} \leftarrow_{\$} [-\tilde{B}, +\tilde{B}]^m$ and compute

$$\mathbf{c} = (\mathbf{A}, \mathbf{b}) \cdot \mathbf{r} + (0^n, q/2) \cdot \mathsf{m}.$$

This algorithm is going instrumental for our scheme, although ciphertexts in this form no longer support the homomorphic evaluation of arbitrary circuits. The multi-bit version of such an algorithm is defined accordingly to output the concatenation of independently sampled ciphertexts. We now recall a useful Lemma from [GP20].

Lemma 2 (GSW Smudging). *Let $\tilde{B} = 2^\lambda \cdot \hat{B}$. For all $\lambda \in \mathbb{N}$, for all $(\mathsf{sk}, \mathsf{pk})$ in the support of $\mathsf{KeyGen}(1^\lambda)$, for all messages $\mathbf{m} = (\mathsf{m}_1, \ldots, \mathsf{m}_\mu)$, for all depth-$d$ circuit $(\Pi_1, \ldots, \Pi_\tau)$, the following distributions are statistically indistinguishable*

$$\begin{pmatrix} c_1, \ldots, c_\mu, \mathbf{r}_1^*, \ldots, \mathbf{r}_\tau^*, \\ \mathsf{Eval}(\mathsf{pk}, \Pi_1, (c_1, \ldots, c_\mu)) + \mathsf{ColEnc}(\mathsf{pk}, 0; \mathbf{r}_1^*), \ldots, \\ \mathsf{Eval}(\mathsf{pk}, \Pi_\tau, (c_1, \ldots, c_\mu)) + \mathsf{ColEnc}(\mathsf{pk}, 0; \mathbf{r}_\tau^*) \end{pmatrix}$$
$$\approx \begin{pmatrix} c_1, \ldots, c_\mu, \mathbf{r}_1^* - \mathsf{RandEval}(\mathsf{pk}, \Pi_1, \mathbf{m}, (\mathbf{R}_1, \ldots, \mathbf{R}_\mu)), \ldots, \\ \mathbf{r}_\tau^* - \mathsf{RandEval}(\mathsf{pk}, \Pi_\tau, \mathbf{m}, (\mathbf{R}_1, \ldots, \mathbf{R}_\mu)), \\ \mathsf{ColEnc}(\mathsf{pk}, \Pi_1(\mathsf{m}_1, \ldots, \mathsf{m}_\mu); \mathbf{r}_1^*), \ldots, \mathsf{ColEnc}(\mathsf{pk}, \Pi_\tau(\mathsf{m}_1, \ldots, \mathsf{m}_\mu); \mathbf{r}_\tau^*) \end{pmatrix}$$

where $c_i \leftarrow_{\$} \mathsf{Enc}(\mathsf{pk}, \mathsf{m}_i; \mathbf{R}_i)$, $\mathbf{r}_i^* \leftarrow_{\$} [-\tilde{B}, +\tilde{B}]^m$, and $\mathbf{R}_i \leftarrow_{\$} \{0,1\}^{m \times m}$.

Randomness Homomorphism. We recall a useful property of the GSW scheme, namely that one can alternatively evaluate functions directly over the randomness of a ciphertext to obtain the same result. More formally, we say that a homomorphic encryption scheme $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ has randomness homomorphism for the circuit class $\{\mathfrak{C}_\lambda\}_{\lambda \in \mathbb{N}}$ if there exists an efficient algorithm $\mathsf{RandEval}$ such that for all $\Pi \in \mathfrak{C}_\lambda$, all $(\mathsf{sk}, \mathsf{pk})$ in the support of KeyGen, all vectors of messages $\mathbf{m} = (\mathsf{m}_1, \ldots, \mathsf{m}_\mu)$ and $\mathbf{R} = (\mathbf{R}_1, \ldots, \mathbf{R}_\mu)$, all ciphertexts $(c_1, \ldots, c_\mu)$ in the support of $(\mathsf{Enc}(\mathsf{pk}, \mathsf{m}_1; \mathbf{R}_1), \ldots, \mathsf{Enc}(\mathsf{pk}, \mathsf{m}_\mu; \mathbf{R}_\mu))$ it holds that

$$\mathsf{Eval}(\mathsf{pk}, \Pi, (c_1, \ldots, c_\mu)) = \mathsf{ColEnc}(\mathsf{pk}, \Pi(\mathbf{m}); \mathsf{RandEval}(\mathsf{pk}, \Pi, \mathbf{m}, \mathbf{R})).$$

Circuit Privacy. It is well known that the GSW encryption scheme satisfies the following notion of circuit privacy [BdMW16, DS16, OPP14] (with a randomized evaluation algorithm).

Definition 2 (Circuit Privacy). *For all $\lambda \in \mathbb{N}$, all all $\Pi \in \mathfrak{C}_\lambda$, all $(\mathsf{sk}, \mathsf{pk})$ in the support of KeyGen, and all messages m, it holds that the following distributions are statistically indistinguishable*

$$(\mathsf{pk}, \mathsf{Enc}(\mathsf{pk}, \Pi(\mathsf{m}))) \approx (\mathsf{pk}, \mathsf{Eval}(\mathsf{pk}, \Pi, \mathsf{Enc}(\mathsf{pk}, \mathsf{m}); r)).$$

where $r \leftarrow_{\$} \{0,1\}^\lambda$.

3 Packed Encryption from LWE

In the following we describe a packed version of the dual-Regev encryption scheme [GPV08]. We denote by $n = n(\lambda)$ the lattice dimension, by $q = q(\lambda)$ the modulus (which we assume for simplicity to be a power of 2), and by $k = k(\lambda)$ the expansion factor. We require the existence of a public-key encryption scheme $(\mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$.

$\underline{\mathsf{KeyGen}(1^\lambda, 1^k)}$**:** Sample a uniform $k \times n$ matrix $\mathbf{B} \leftarrow_{\$} \mathbb{Z}_q^{k \times n}$ and a key pair of a public-key encryption scheme $(\mathsf{sk}_{\mathsf{PKE}}, \mathsf{pk}_{\mathsf{PKE}}) \leftarrow_{\$} \mathsf{PKE.KeyGen}(1^\lambda)$. The public key consists of $(\mathbf{B}, \mathsf{pk}_{\mathsf{PKE}})$ and the secret key is set to $\mathsf{sk}_{\mathsf{PKE}}$.

$\underline{\mathsf{Enc}(\mathsf{pk}, \mathbf{m})}$**:** To encrypt a k-bit message $\mathbf{m} \in \{0,1\}^k$, sample a uniform randomness vector $\mathbf{r} \leftarrow_{\$} \mathbb{Z}_q^n$ a noise vector $\mathbf{e} \leftarrow_{\$} \chi^k$ and return the ciphertext

$$\mathbf{c} = (\mathsf{PKE.Enc}(\mathsf{pk}_{\mathsf{PKE}}, \mathbf{r}), \mathbf{B} \cdot \mathbf{r} + q/2 \cdot \mathbf{m} + \mathbf{e}).$$

$\underline{\mathsf{Dec}(\mathsf{sk}, \mathbf{c})\mathbf{:}}$ Parse $\mathbf{c}$ as $(c_{\mathsf{PKE}}, c_1, \ldots, c_k)$ and recover the random coins by decrypting $\mathbf{r} = \mathsf{PKE.Dec}(\mathsf{sk}_{\mathsf{PKE}}, c_{\mathsf{PKE}})$. Let $\mathbf{b}_i$ be the i-th row of $\mathbf{B}$. For $i = 1 \ldots k$, compute $\mathsf{m}_i = \mathsf{Round}(c_i - \mathbf{b}_i \cdot \mathbf{r})$, where Round rounds to the nearest multiple of $q/2$, i.e. it returns 1 if the input is closer to $q/2$ and 0 otherwise. Output $\mathbf{m} = (\mathsf{m}_1, \ldots, \mathsf{m}_k)$.

Clearly, the scheme is perfectly correct since

$$\begin{aligned}
&(\mathsf{Round}(c_1 - \mathbf{b}_1 \cdot \mathbf{r}), \ldots, \mathsf{Round}(c_k - \mathbf{b}_k \cdot \mathbf{r}))\\
&= (\mathsf{Round}(q/2 \cdot \mathsf{m}_1 + e_1), \ldots, \mathsf{Round}(q/2 \cdot \mathsf{m}_k + e_k))\\
&= (\mathsf{Round}(q/2 \cdot \mathsf{m}_1), \ldots, \mathsf{Round}(q/2 \cdot \mathsf{m}_k))\\
&= (\mathsf{m}_1, \ldots, \mathsf{m}_k)\\
&= \mathbf{m}.
\end{aligned}$$

Extended Encryption. It is not hard to see that the scheme presented above is (bounded) additively homomorphic over $\mathbb{Z}_2^k$. To lift the class of computable functions to all linear functions over $\mathbb{Z}_q^k$, we adopt the standard trick of encrypting the message multiplied by all powers of two $(1, 2, \ldots, 2^{\log(q)})$. For convenience, we define the following augmented encryption algorithm.

$\underline{\mathsf{ExtEnc}(\mathsf{pk}, \mathbf{m})\mathbf{:}}$ On input an ℓ-dimensional message $\mathbf{m} \in \mathbb{Z}_q^\ell$, let $\mathbf{g} = (1, 2, \ldots, 2^{\log(q)-1})^T$ and define

$$\mathbf{M} = \begin{bmatrix} \mathsf{m}_1 \cdot \mathbf{g} & \mathsf{m}_2 \cdot \mathbf{g} & \ldots & 0^{\log(q)} \\ 0^{\log(q)} & 0^{\log(q)} & \ldots & 0^{\log(q)} \\ \vdots & \vdots & \ddots & \vdots \\ 0^{\log(q)} & 0^{\log(q)} & \ldots & \mathsf{m}_\ell \cdot \mathbf{g} \end{bmatrix} \in \mathbb{Z}_q^{k \times \ell \cdot k \cdot \log(q)}.$$

Sample a uniform randomness matrix $\mathbf{R} \leftarrow_\$ \mathbb{Z}_q^{n \times \ell \cdot k \cdot \log(q)}$ and a uniform noise matrix $\mathbf{E} \leftarrow_\$ \chi^{k \times \ell \cdot k \cdot \log(q)}$. Compute

$$\mathbf{C} = \mathbf{B} \cdot \mathbf{R} + \mathbf{M} + \mathbf{E}$$

and return the ciphertext $(\mathsf{PKE.Enc}(\mathsf{pk}_{\mathsf{PKE}}, \mathbf{R}), \mathbf{C})$.

Decryption works, as before, by recovering $\mathbf{R}$ from the public-key encryption scheme and then decrypting $\mathbf{m}$ component-wise.

Almost-Everywhere Dense Encryption. For convenience, we also define an alternative encryption algorithm in the following. Note that the encryption algorithm does not take as input any message, instead it encrypts a uniform k-bit binary vector. Syntactically, this is the equivalent of a key-encapsulation mechanism.

$\underline{\mathsf{DenseEnc}(\mathsf{pk})\mathbf{:}}$ Sample a uniform randomness vector $\mathbf{r} \leftarrow_\$ \mathbb{Z}_q^n$ and return the ciphertext

$$\mathbf{c} = (c_{\mathsf{PKE}}, c_1, \ldots, c_k) = (\mathsf{PKE.Enc}(\mathsf{pk}_{\mathsf{PKE}}, \mathbf{r}), \mathbf{B} \cdot \mathbf{r} + \mathbf{u}).$$

where $\mathbf{u} \leftarrow_\$ \mathbb{Z}_q^k$.

We highlight two facts about this algorithm that are going to be important for our later construction: (i) The decryption algorithm works for both Enc and $\mathsf{DenseEnc}$ algorithms, where the plaintext of $\mathsf{DenseEnc}$ corresponds to $(\mathsf{Round}(u_1), \ldots, \mathsf{Round}(u_k))$. In fact, the scheme satisfies perfect correctness in both cases. (ii) The domain of the elements $(c_1, \ldots, c_k)$ is *dense*, i.e. the support of the scheme spans the entire vector space $\mathbb{Z}_q^k$. Since the element c_{PKE} is small (i.e. independent of k) for an appropriate choice of the public-key encryption scheme, we refer to such a property as *almost-everywhere* density.

Semantic Security. We argue that the scheme satisfies a strong form of semantic security, i.e. the honestly computed ciphertexts are computationally indistinguishable from uniform vectors in $\mathbb{Z}_q^k$. Semantic security for the extended encryption ExtEnc and the dense encryption $\mathsf{DenseEnc}$ follows along the same lines.

Theorem 2 (Semantic Security). *If* $(\mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ *is semantically secure and the LWE assumption holds, then for all* $\lambda \in \mathbb{N}$ *and all messages* m *it holds that the following distributions are computationally indistinguishable*

$$(\mathsf{pk}, \mathsf{Enc}(\mathsf{pk}, \mathsf{m})) \approx (\mathsf{pk}, \mathsf{PKE.Enc}(\mathsf{pk}_{\mathsf{PKE}}, \mathbf{z}), \mathbf{u}).$$

where $(\mathsf{sk}, \mathsf{pk}) \leftarrow_{\$} \mathsf{KeyGen}(1^{\lambda}, 1^{k})$*,* $\mathbf{z} \leftarrow_{\$} \mathbb{Z}_q^n$*, and* $\mathbf{u} \leftarrow_{\$} \mathbb{Z}_q^k$*.*

Proof. The security of the scheme follows routinely by an invocation of semantic security of the public-key encryption scheme and an invocation of the LWE assumption.

3.1 Key-Randomness SRL Security

We state a version of SRL security [GP20] tailored for our specific instance and adapted to the randomness-key circularity assumption (rather than the 2-key circularity, as stated in [GP20]).

Definition 3 (Key-Randomness SRL Security). *Let* $(\mathsf{GSW.KeyGen}, \mathsf{GSW.Enc}, \mathsf{GSW.Eval}, \mathsf{GSW.Dec})$ *be the GSW encryption scheme and* $(\mathsf{dR.KeyGen}, \mathsf{dR.Enc}, \mathsf{dR.Eval}, \mathsf{dR.Dec})$ *be the packed dual-Regev encryption scheme. Fix messages* $(\mathsf{m}_0, \mathsf{m}_1)$*, polynomials* $\tau = \tau(\lambda)$ *and* $k = k(\lambda)$ *and an adversary* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$*. Consider the following experiment.*

$\mathsf{Exp}_{\mathsf{SRL}}^{(b)}(\mathcal{A})$ *:*

- *Sample* $(\bar{\mathsf{sk}}, (\bar{\mathsf{pk}}, \mathbf{B})) \leftarrow_{\$} \mathsf{dR.KeyGen}(1^{\lambda}, 1^{k})$ *and* $(\mathsf{sk}, \mathsf{pk}) \leftarrow_{\$} \mathsf{GSW.KeyGen}(1^{\lambda})$
- *Compute* $\mathbf{c} \leftarrow_{\$} \mathsf{GSW.Enc}(\mathsf{pk}, \mathsf{m}_b)$
- $\{\mathbf{P}_i, \mathbf{p}_i\}_{i=1\dots\tau} = \mathcal{A}_1(\mathsf{pk}, \mathbf{B}, \mathbf{c})$
- *Compute a key-randomness cycle* $\left(\bar{c}_{\mathsf{sk}}, \bar{\mathbf{C}}_{\mathsf{sk}}\right) = \mathsf{dR.ExtEnc}(\bar{\mathsf{pk}}, \mathsf{sk}; \mathbf{S})$ *and* $\mathbf{c}_{\mathbf{S}} \leftarrow_{\$} \mathsf{GSW.Enc}(\mathsf{pk}, \mathbf{S}; \mathbf{R}_{\mathbf{S}})$
- *For all* $i = 1 \dots \tau$*:*
    - *Sample* $\mathbf{C}_i^* = \mathsf{ColEnc}(0; \mathbf{r}_i^*)$ *where* $\mathbf{r}_i^* \leftarrow_{\$} [-\tilde{B}, +\tilde{B}]^{m \cdot k}$
    - *Sample* $\mathbf{t}_i \leftarrow_{\$} \mathbb{Z}_q^n$
    - *Sample* $\mathbf{c}_{i,\mathbf{r}} = \mathsf{GSW.Enc}(\mathsf{pk}, \mathbf{S} \cdot \mathsf{Bit}(\ell_i) + \mathbf{t}_i; \mathbf{R}_i) \leftarrow_{\$} \mathsf{GSW.Eval}(\mathsf{pk}, \cdot \mathsf{Bit}(\ell_i) + \mathbf{t}_i, \mathbf{c}_{\mathbf{S}})$
- *Output* $\mathcal{A}_2(\bar{\mathbf{C}}_{\mathsf{sk}}, \{\mathbf{c}_{i,\mathbf{r}}, \mathbf{C}_i^*, \mathbf{u}_i, \mathbf{t}_i, \mathbf{r}_{\psi,i} - \mathbf{r}_i^*\}_{i=1\dots\tau})$

Here, letting ℓ_i *be the linear function associated with* $\mathbf{C}_i^* + \mathbf{P}_i + q/2 \cdot \mathbf{v}_i$*, we set*

$$\mathbf{r}_{\psi,i} = \mathsf{RandEval}(\mathsf{pk}, \psi_i, \mathbf{S} \cdot \mathsf{Bit}(\ell_i) + \mathbf{t}_i, \mathbf{R}_i) \text{ and}$$
$$\psi_i(\mathbf{Z}) = \mathsf{Round}\left(\mathbf{B} \cdot \mathbf{t}_i + q/2 \cdot \mathbf{p}_i + \mathbf{w}_i - \bar{\mathbf{C}}_{\mathsf{sk}} \cdot \mathsf{Bit}(\ell_i) - \mathbf{B} \cdot \mathbf{Z}\right)$$

where $\mathbf{w}_i \leftarrow_{\$} [-q/4, q/4]^k$*,* $\mathbf{v}_i \leftarrow_{\$} \{0,1\}^k$*, and* $\mathbf{u}_i = \mathbf{w}_i + q/2 \cdot \mathbf{v}_i$*, for all* $i = 1 \dots \tau$*. We say that an adversary* $\mathcal{A}$ *is admissible if for all* $i = 1 \dots \tau$ *it holds that* $\mathbf{P}_i \in \mathsf{GSW.Enc}(\mathsf{pk}, \mathbf{p}_i)$*. The KR-SRL assumption conjectures that it holds for all admissible PPT adversaries* $\mathcal{A}$*, all messages* $(\mathsf{m}_0, \mathsf{m}_1)$ *and all polynomials* $\tau = \tau(\lambda)$ *and* $k = k(\lambda)$ *that*

$$|\Pr[\mathsf{Exp}_{\mathsf{SRL}}^{(1)}(\mathcal{A}) = 1] - \Pr[\mathsf{Exp}_{\mathsf{SRL}}^{(0)}(\mathcal{A}) = 1]| \leq \mathsf{negl}(\lambda)\,.$$

Since the SRL leakage depends on the specific circuit representation of the functions $(\psi_1 \dots \psi_\tau)$, we propose a natural implementation for a class of functions that suffices to capture all possible leakage functions. Specifically, observe that ψ_i consist of a linear function (computed over $\mathbb{Z}_q$) followed by a rounding to the nearest multiple of $q/2$. Since all inputs are bit-wise encrypted the computation of modular additions (and multiplication by constants) is done via a canonical boolean circuit (see [BV11] for a concrete example) and the rounding is obtained by simply returning the ciphertext containing the most significant bit of the output.

4 Constructing (Shallow) XiO

In the following we present the construction of shallow XiO from the GSW scheme $(\mathsf{GSW.KeyGen}, \mathsf{GSW.Enc}, \mathsf{GSW.Eval}, \mathsf{GSW.Dec})$ and the packed version of the dual-Regev encryption $(\mathsf{dR.KeyGen}, \mathsf{dR.Enc}, \mathsf{dR.Eval}, \mathsf{dR.Dec})$ as described in Section 3.

4.1 Construction

The scheme assumes a long uniform string that is, for convenience, split in two chunks:

- A sequence of randomization vectors $(\mathbf{r}^*_1, \ldots, \mathbf{r}^*_{2^{\eta-\log(k)}})$ for the GSW scheme $\mathsf{GSW.PubCoin}$, where each $\mathbf{r}^*_i = (\mathbf{r}^*_{i,1}, \ldots, \mathbf{r}^*_{i,k}) \in [-\tilde{B}, +\tilde{B}]^{m\cdot k}$.
- A sequence of dense ciphertexts $(h_1, \ldots, h_{2^{\eta-\log(k)}})$ for packed dual-Regev scheme $\mathsf{dR.PubCoin}$, where each $h_i = (h_{i,1}, \ldots, h_{i,k}) \in \mathbb{Z}_q^k$.

On input the security parameter 1^λ and the circuit $\Pi : \{0,1\}^\eta \to \{0,1\}$, the obfuscator proceeds as follows.

Setting the Public Keys: Sample a dual-Regev key pair $(\bar{\mathsf{sk}}, \bar{\mathsf{pk}}) \leftarrow_{\$} \mathsf{dR.KeyGen}(1^\lambda, 1^k)$ and GSW key pair $(\mathsf{sk}, \mathsf{pk}) \leftarrow_{\$} \mathsf{GSW.KeyGen}(1^\lambda; q)$, where q is the modulus defined by the dual-Regev scheme. Compute a bit-by-bit GSW encryption $\mathbf{c}_\Pi \leftarrow_{\$} \mathsf{GSWEnc}(\mathsf{pk}, \Pi)$ of the binary representation of the circuit Π.

Compute a Key Encryption: Compute a dual-Regev extended encryption of the GSW secret key

$$\left(\bar{c}_{\mathsf{sk}}, \bar{\mathbf{C}}_{\mathsf{sk}}\right) = \mathsf{dR.ExtEnc}(\bar{\mathsf{pk}}, \mathsf{sk}; \mathbf{S}).$$

where $\mathsf{sk} \in \mathbb{Z}_q^{n+1}$ and $\mathbf{S} \leftarrow_{\$} \mathbb{Z}_q^{n\times\log(q)\cdot k\cdot(n+1)}$.

Decryption Hints: For all indices $i \in \{0,1\}^{\eta-\log(k)}$, do the following.

Evaluate the Circuit: Let $\Phi_{i,j} : \{0,1\}^{|\Pi|} \to \{0,1\}$ be the universal circuit that, on input a circuit description Π, returns the j-th bit of the i-th block (where each block consists of k bits) of the corresponding truth table. Compute

$$\mathbf{C}_i = \begin{bmatrix} \mathsf{GSW.Eval}(\mathsf{pk}, \Phi_{i,1}, \mathbf{c}_\Pi) \\ \ldots \\ \mathsf{GSW.Eval}(\mathsf{pk}, \Phi_{i,k}, \mathbf{c}_\Pi) \end{bmatrix} \in \mathbb{Z}_q^{k\times(n+1)}.$$

Compute the Low-Order Bits: Sample a uniform $\mathbf{r}_i \leftarrow_{\$} \mathbb{Z}_q^n$ and compute $\mathbf{c}_{i,\mathbf{r}} \leftarrow_{\$} \mathsf{GSW.Enc}(\mathsf{pk}, \mathbf{r}_i)$. Parse the i-th block of $\mathsf{dR.PubCoin}$ as

$$(h_{i,1}, \ldots, h_{i,k}) = \mathbf{B} \cdot \mathbf{r}_i + (u_{i,1}, \ldots, u_{i,k}) \in \mathbb{Z}_q^k$$

for some $(u_{i,1}, \ldots, u_{i,k}) \in \mathbb{Z}_q^k$. Let $\Psi_{i,j} : \{0,1\}^\lambda \to \{0,1\}$ be the circuit that, on input $\mathbf{r}_i$, computes the decryption of the j-th bit encrypted in $(h_{i,1}, \ldots, h_{i,k})$. I.e. it computes $\mathsf{Round}((h_{i,1}, \ldots, h_{i,k}) - \mathbf{B} \cdot \mathbf{r}_i)$. Compute homomorphically the matrix of ciphertexts

$$\mathbf{C}_{i,\mathsf{Round}} = \begin{bmatrix} \mathsf{GSW.Eval}(\mathsf{pk}, \Psi_{i,1}, \mathbf{c}_{i,\mathbf{r}}) \\ \ldots \\ \mathsf{GSW.Eval}(\mathsf{pk}, \Psi_{i,k}, \mathbf{c}_{i,\mathbf{r}}) \end{bmatrix} \in \mathbb{Z}_q^{k\times(n+1)}.$$

Rerandomize the Ciphertext: Parse the i-th block of $\mathsf{GSW.PubCoin}$ as $(\mathbf{r}^*_{i,1}, \ldots, \mathbf{r}^*_{i,k}) \in [-\tilde{B}, +\tilde{B}]^{m\cdot k}$ and compute

$$\mathbf{C}'_{i,\mathsf{Round}} = \mathbf{C}_{i,\mathsf{Round}} + \begin{bmatrix} \mathsf{GSW.ColEnc}(\mathsf{pk}, 0; \mathbf{r}^*_{i,1}) \\ \ldots \\ \mathsf{GSW.ColEnc}(\mathsf{pk}, 0; \mathbf{r}^*_{i,k}) \end{bmatrix} \in \mathbb{Z}_q^{k\times(n+1)}.$$

Proxy Re-Encrypt: Define $\mathbf{D}_i$ as the vector of GSW ciphertexts resulting from the homomorphic sum of $\mathbf{C}'_{i,\mathsf{Round}}$ and $\mathbf{C}_i$, i.e. $\mathbf{D}_i = \mathbf{C}'_{i,\mathsf{Round}} + \mathbf{C}_i$. Observe that $\mathbf{D}_i$ consists of k GSW ciphertexts and let $\boldsymbol{\ell}_{i,j} \in \mathbb{Z}_q^{(n+1)}$ be the linear function associated with the decryption of the j-th ciphertext. Define $\boldsymbol{\ell}_i = (\boldsymbol{\ell}_{i,1}, \ldots, \boldsymbol{\ell}_{i,k})$ and compute

$$\bar{\mathbf{c}}_i = \bar{\mathbf{C}}_{\mathsf{sk}} \cdot \mathsf{Bit}(\boldsymbol{\ell}_i) + (h_{i,1}, \ldots, h_{i,k}) \in \mathbb{Z}_q^k$$

where the function $\mathsf{Bit} : \mathbb{Z}_q^{k\cdot(n+1)} \to \{0,1\}^{\log(q)\cdot k\cdot(n+1)}$ is the bit decomposition operator.

Release Hint: Compute the i-th decryption hint as

$$\boldsymbol{\rho}_i = \mathbf{S} \cdot \mathsf{Bit}(\boldsymbol{\ell}_i) + \mathbf{r}_i \in \mathbb{Z}_q^n.$$

Output: The obfuscated circuit consists of the public keys $(\mathsf{pk}, \bar{\mathsf{pk}})$, the matrix $\bar{\mathbf{C}}_{\mathsf{sk}}$, the GSW encryption of the circuit $\mathbf{c}_\Pi$, the encryption headers $(\mathbf{c}_{1,\mathbf{r}}, \dots, \mathbf{c}_{2^{\eta-\log(k)},\mathbf{r}})$, and the decryption hints $(\boldsymbol{\rho}_1, \dots, \boldsymbol{\rho}_{2^{\eta-\log(k)}})$.

To evaluate the obfuscated circuit on input x, let i be the index of the block of the truth table of Π that contains $\Pi(x)$. The evaluator computes $\bar{\mathbf{c}}_i$ as specified above (note that all the operations are public, given the information included in the obfuscated circuit) and recovers $\Pi^{(i)}$ (the i-th block of the truth table of Π) by computing

$$\Pi^{(i)} = \mathsf{Round}(\bar{\mathbf{c}}_i - \mathbf{B} \cdot \boldsymbol{\rho}_i)$$

where $\mathsf{Round} : \mathbb{Z}_q^k \to \{0,1\}^k$ rounds the input to the nearest multiple of $q/2$.

Correctness. To see why the evaluation algorithm is correct, recall that

$$\bar{\mathbf{c}}_i = \bar{\mathbf{C}}_{\mathsf{sk}} \cdot \mathsf{Bit}(\boldsymbol{\ell}_i) + (h_{i,1}, \dots, h_{i,k}).$$

First observe that $(\mathbf{r}_i, h_{i,1}, \dots, h_{i,k})$ define a ciphertext in the support of the algorithm $\mathsf{dR.DenseEnc}(\bar{\mathsf{pk}})$, which we rewrite as

$$\begin{aligned}\mathsf{dR.DenseEnc}(\bar{\mathsf{pk}}) &= (\mathsf{PKE.Enc}(\mathsf{pk}_{\mathsf{PKE}}, \mathbf{r}_i), \mathbf{B} \cdot \mathbf{r}_i + (u_{i,1}, \dots, u_{i,k})) \\ &= (\mathsf{PKE.Enc}(\mathsf{pk}_{\mathsf{PKE}}, \mathbf{r}_i), \mathbf{B} \cdot \mathbf{r}_i + \mathbf{u}_i)\,.\end{aligned}$$

Thus $\mathbf{C}'_{i,\mathsf{Round}}$ and $\mathbf{C}_i$ are in the support of

$$\begin{bmatrix} \mathsf{GSW.ColEnc}(\mathsf{pk}, \mathsf{Round}(u_1)) \\ \dots \\ \mathsf{GSW.ColEnc}(\mathsf{pk}, \mathsf{Round}(u_k)) \end{bmatrix} \text{ and } \begin{bmatrix} \mathsf{GSW.ColEnc}(\mathsf{pk}, \Pi_1^{(i)}) \\ \dots \\ \mathsf{GSW.ColEnc}(\mathsf{pk}, \Pi_k^{(i)}) \end{bmatrix}$$

respectively, by the evaluation correctness of the GSW scheme and by Lemma 1. Furthermore, recall that $\mathbf{D}_i = \mathbf{C}'_{i,\mathsf{Round}} + \mathbf{C}_i$. By an invocation of Lemma 1, we have that $\mathbf{D}_i$ is in the support of

$$\begin{aligned}&\begin{bmatrix} \mathsf{GSW.ColEnc}(\mathsf{pk}, \mathsf{Round}(u_1)) \\ \dots \\ \mathsf{GSW.ColEnc}(\mathsf{pk}, \mathsf{Round}(u_k)) \end{bmatrix} + \begin{bmatrix} \mathsf{GSW.ColEnc}(\mathsf{pk}, \Pi_1^{(i)}) \\ \dots \\ \mathsf{GSW.ColEnc}(\mathsf{pk}, \Pi_k^{(i)}) \end{bmatrix} \\ &\approx \begin{bmatrix} \mathsf{GSW.ColEnc}(\mathsf{pk}, \mathsf{Round}(u_1) \oplus \Pi_1^{(i)}) \\ \dots \\ \mathsf{GSW.ColEnc}(\mathsf{pk}, \mathsf{Round}(u_k) \oplus \Pi_k^{(i)}) \end{bmatrix}\end{aligned}$$

with all but negligible probability. By the almost-linear decryption of GSW, it follows that

$$\bar{\mathbf{C}}_{\mathsf{sk}} \cdot \mathsf{Bit}(\boldsymbol{\ell}_i) = \mathbf{B} \cdot \tilde{\mathbf{s}}_i + \boldsymbol{\xi}_i + \boldsymbol{\zeta}_i + q/2 \cdot \left(\mathsf{Round}(u_1) \oplus \Pi_1^{(i)}, \dots, \mathsf{Round}(u_k) \oplus \Pi_k^{(i)}\right)$$

where $\boldsymbol{\xi}_i$ is the decryption noise of the packed dual-Regev scheme (i.e. the subset sum of the noise terms of $\bar{\mathbf{C}}_{\mathsf{sk}}$) and $\boldsymbol{\zeta}_i$ is the decryption noise of the GSW ciphertext. It follows that $\|\boldsymbol{\xi}_i\|_\infty \leq B \cdot \log(q) \cdot k \cdot (n+1)$ and, by Lemma 1, $\|\boldsymbol{\zeta}_i\|_\infty \leq \tilde{B}$ with all but negligible probability. Note that, by linearity we have that $\tilde{\mathbf{s}}_i = \mathbf{S} \cdot \mathsf{Bit}(\boldsymbol{\ell}_i)$. Consequently, it holds that

$$\begin{aligned}\bar{\mathbf{c}}_i &= \mathbf{B} \cdot \tilde{\mathbf{s}}_i + \boldsymbol{\xi}_i + \boldsymbol{\zeta}_i + q/2 \cdot \left(\mathsf{Round}(u_1) \oplus \Pi_1^{(i)}, \dots, \mathsf{Round}(u_k) \oplus \Pi_k^{(i)}\right) + \mathbf{B} \cdot \mathbf{r}_i + \mathbf{u}_i \\ &= \mathbf{B} \cdot (\tilde{\mathbf{s}}_i + \mathbf{r}_i) + \boldsymbol{\xi}_i + \boldsymbol{\zeta}_i + q/2 \cdot \left(\mathsf{Round}(u_1) \oplus \Pi_1^{(i)}, \dots, \mathsf{Round}(u_k) \oplus \Pi_k^{(i)}\right) + \mathbf{u}_i \\ &= \mathbf{B} \cdot (\tilde{\mathbf{s}}_i + \mathbf{r}_i) + q/2 \cdot \Pi^{(i)} + \mathbf{v}_i \\ &= \mathbf{B} \cdot \boldsymbol{\rho}_i + q/2 \cdot \Pi^{(i)} + \mathbf{v}_i\end{aligned}$$

where $\mathbf{v}_i = \mathbf{u}_i + q/2 \cdot \mathsf{Round}(\mathbf{u}_i) + \boldsymbol{\xi}_i + \boldsymbol{\zeta}_i$ and $\|\mathbf{v}_i\|_\infty < q/4$ with all but negligible probability, over the random choice of $\mathbf{u}_i$. This is because $\mathbf{D}_i$ is statistically close to a fresh GSW encryption of $(\mathsf{Round}(u_1), \ldots, \mathsf{Round}(u_k)) \oplus \mathit{\Pi}^{(i)}$, by Lemma 2. Therefore we have that

$$\begin{aligned}\mathsf{Round}\left(\mathbf{c}_i - \mathbf{B} \cdot \boldsymbol{\rho}_i\right) &= \mathsf{Round}\left(\mathbf{B} \cdot \boldsymbol{\rho}_i + q/2 \cdot \mathit{\Pi}^{(i)} + \mathbf{v}_i - \mathbf{B} \cdot \boldsymbol{\rho}_i\right) \\ &= \mathsf{Round}\left(q/2 \cdot \mathit{\Pi}^{(i)} + \mathbf{v}_i\right) \\ &= \mathsf{Round}\left(q/2 \cdot \mathit{\Pi}^{(i)}\right) \\ &= \mathit{\Pi}^{(i)}\end{aligned}$$

with the same probability. Due to space constraints, we defer the analysis of our scheme to Appendix B.

References

[Agr19] Shweta Agrawal. Indistinguishability obfuscation without multilinear maps: New methods for bootstrapping and instantiation. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2019, Part I*, volume 11476 of *Lecture Notes in Computer Science*, pages 191–225, Darmstadt, Germany, May 19–23, 2019. Springer, Heidelberg, Germany.

[AJ15] Prabhanjan Ananth and Abhishek Jain. Indistinguishability obfuscation from compact functional encryption. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *Advances in Cryptology – CRYPTO 2015, Part I*, volume 9215 of *Lecture Notes in Computer Science*, pages 308–326, Santa Barbara, CA, USA, August 16–20, 2015. Springer, Heidelberg, Germany.

[AJL$^+$19] Prabhanjan Ananth, Aayush Jain, Huijia Lin, Christian Matt, and Amit Sahai. Indistinguishability obfuscation without multilinear maps: New paradigms via low degree weak pseudorandomness and security amplification. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019, Part III*, volume 11694 of *Lecture Notes in Computer Science*, pages 284–332, Santa Barbara, CA, USA, August 18–22, 2019. Springer, Heidelberg, Germany.

[AS17] Prabhanjan Ananth and Amit Sahai. Projective arithmetic functional encryption and indistinguishability obfuscation from degree-5 multilinear maps. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *Advances in Cryptology – EUROCRYPT 2017, Part I*, volume 10210 of *Lecture Notes in Computer Science*, pages 152–181, Paris, France, April 30 – May 4, 2017. Springer, Heidelberg, Germany.

[BDGM19] Zvika Brakerski, Nico Döttling, Sanjam Garg, and Giulio Malavolta. Leveraging linear decryption: Rate-1 fully-homomorphic encryption and time-lock puzzles. In Dennis Hofheinz and Alon Rosen, editors, *TCC 2019: 17th Theory of Cryptography Conference, Part II*, volume 11892 of *Lecture Notes in Computer Science*, pages 407–437, Nuremberg, Germany, December 1–5, 2019. Springer, Heidelberg, Germany.

[BDGM20] Zvika Brakerski, Nico Döttling, Sanjam Garg, and Giulio Malavolta. Candidate iO from homomorphic encryption schemes. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology – EUROCRYPT 2020, Part I*, volume 12105 of *Lecture Notes in Computer Science*, pages 79–109, Zagreb, Croatia, May 10–14, 2020. Springer, Heidelberg, Germany.

[BdMW16] Florian Bourse, Rafaël del Pino, Michele Minelli, and Hoeteck Wee. FHE circuit privacy almost for free. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016, Part II*, volume 9815 of *Lecture Notes in Computer Science*, pages 62–89, Santa Barbara, CA, USA, August 14–18, 2016. Springer, Heidelberg, Germany.

[BGI$^+$01] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 1–18, Santa Barbara, CA, USA, August 19–23, 2001. Springer, Heidelberg, Germany.

[BGI14] Elette Boyle, Shafi Goldwasser, and Ioana Ivan. Functional signatures and pseudorandom functions. In Hugo Krawczyk, editor, *PKC 2014: 17th International Conference on Theory and Practice of Public Key Cryptography*, volume 8383 of *Lecture Notes in Computer Science*, pages 501–519, Buenos Aires, Argentina, March 26–28, 2014. Springer, Heidelberg, Germany.

[BGMZ18] James Bartusek, Jiaxin Guan, Fermi Ma, and Mark Zhandry. Return of GGH15: Provable security against zeroizing attacks. In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018: 16th Theory of Cryptography Conference, Part II*, volume 11240 of *Lecture Notes in Computer Science*, pages 544–574, Panaji, India, November 11–14, 2018. Springer, Heidelberg, Germany.

[BV11] Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. In Rafail Ostrovsky, editor, *52nd Annual Symposium on Foundations of Computer Science*, pages 97–106, Palm Springs, CA, USA, October 22–25, 2011. IEEE Computer Society Press.

[BV15] Nir Bitansky and Vinod Vaikuntanathan. Indistinguishability obfuscation from functional encryption. In Venkatesan Guruswami, editor, *56th Annual Symposium on Foundations of Computer Science*, pages 171–190, Berkeley, CA, USA, October 17–20, 2015. IEEE Computer Society Press.

[BZ14] Dan Boneh and Mark Zhandry. Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology – CRYPTO 2014, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 480–499, Santa Barbara, CA, USA, August 17–21, 2014. Springer, Heidelberg, Germany.

[CGH17] Yilei Chen, Craig Gentry, and Shai Halevi. Cryptanalyses of candidate branching program obfuscators. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *Advances in Cryptology – EUROCRYPT 2017, Part III*, volume 10212 of *Lecture Notes in Computer Science*, pages 278–307, Paris, France, April 30 – May 4, 2017. Springer, Heidelberg, Germany.

[CHL$^+$15] Jung Hee Cheon, Kyoohyung Han, Changmin Lee, Hansol Ryu, and Damien Stehlé. Cryptanalysis of the multilinear map over the integers. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015, Part I*, volume 9056 of *Lecture Notes in Computer Science*, pages 3–12, Sofia, Bulgaria, April 26–30, 2015. Springer, Heidelberg, Germany.

[CLT13] Jean-Sébastien Coron, Tancrède Lepoint, and Mehdi Tibouchi. Practical multilinear maps over the integers. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part I*, volume 8042 of *Lecture Notes in Computer Science*, pages 476–493, Santa Barbara, CA, USA, August 18–22, 2013. Springer, Heidelberg, Germany.

[CVW18] Yilei Chen, Vinod Vaikuntanathan, and Hoeteck Wee. GGH15 beyond permutation branching programs: Proofs, attacks, and candidates. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018, Part II*, volume 10992 of *Lecture Notes in Computer Science*, pages 577–607, Santa Barbara, CA, USA, August 19–23, 2018. Springer, Heidelberg, Germany.

[DGG$^+$18] Nico Döttling, Sanjam Garg, Divya Gupta, Peihan Miao, and Pratyay Mukherjee. Obfuscation from low noise multilinear maps. In Debrup Chakraborty and Tetsu Iwata, editors, *Progress in Cryptology - INDOCRYPT 2018: 19th International Conference in Cryptology in India*, volume 11356 of *Lecture Notes in Computer Science*, pages 329–352, New Delhi, India, December 9–12, 2018. Springer, Heidelberg, Germany.

[DJ01] Ivan Damgård and Mats Jurik. A generalisation, a simplification and some applications of Paillier's probabilistic public-key system. In Kwangjo Kim, editor, *PKC 2001: 4th International Workshop on Theory and Practice in Public Key Cryptography*, volume 1992 of *Lecture Notes in Computer Science*, pages 119–136, Cheju Island, South Korea, February 13–15, 2001. Springer, Heidelberg, Germany.

[DS16] Léo Ducas and Damien Stehlé. Sanitization of FHE ciphertexts. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016, Part I*, volume 9665 of *Lecture Notes in Computer Science*, pages 294–310, Vienna, Austria, May 8–12, 2016. Springer, Heidelberg, Germany.

[GGH13a] Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology – EUROCRYPT 2013*, volume 7881 of *Lecture Notes in Computer Science*, pages 1–17, Athens, Greece, May 26–30, 2013. Springer, Heidelberg, Germany.

[GGH$^+$13b] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *54th Annual Symposium on Foundations of Computer Science*, pages 40–49, Berkeley, CA, USA, October 26–29, 2013. IEEE Computer Society Press.

[GGH15] Craig Gentry, Sergey Gorbunov, and Shai Halevi. Graph-induced multilinear maps from lattices. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015: 12th Theory of Cryptography Conference, Part II*, volume 9015 of *Lecture Notes in Computer Science*, pages 498–527, Warsaw, Poland, March 23–25, 2015. Springer, Heidelberg, Germany.

[GGHR14] Sanjam Garg, Craig Gentry, Shai Halevi, and Mariana Raykova. Two-round secure MPC from indistinguishability obfuscation. In Yehuda Lindell, editor, *TCC 2014: 11th Theory of Cryptography Conference*, volume 8349 of *Lecture Notes in Computer Science*, pages 74–94, San Diego, CA, USA, February 24–26, 2014. Springer, Heidelberg, Germany.

[GGM84] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions (extended abstract). In *25th Annual Symposium on Foundations of Computer Science*, pages 464–479, Singer Island, Florida, October 24–26, 1984. IEEE Computer Society Press.

[GKP$^+$13] Shafi Goldwasser, Yael Tauman Kalai, Raluca A. Popa, Vinod Vaikuntanathan, and Nickolai Zeldovich. Reusable garbled circuits and succinct functional encryption. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th Annual ACM Symposium on Theory of Computing*, pages 555–564, Palo Alto, CA, USA, June 1–4, 2013. ACM Press.

[GM82] Shafi Goldwasser and Silvio Micali. Probabilistic encryption and how to play mental poker keeping secret all partial information. In *14th Annual ACM Symposium on Theory of Computing*, pages 365–377, San Francisco, CA, USA, May 5–7, 1982. ACM Press.

[GMM$^+$16] Sanjam Garg, Eric Miles, Pratyay Mukherjee, Amit Sahai, Akshayaram Srinivasan, and Mark Zhandry. Secure obfuscation in a weak multilinear map model. In Martin Hirt and Adam D. Smith, editors, *TCC 2016-B: 14th Theory of Cryptography Conference, Part II*, volume 9986 of *Lecture Notes in Computer Science*, pages 241–268, Beijing, China, October 31 – November 3, 2016. Springer, Heidelberg, Germany.

[Gol01] Oded Goldreich. *The Foundations of Cryptography - Volume 1: Basic Techniques*. Cambridge University Press, 2001.

[GP20] Romain Gay and Rafael Pass. Indistinguishability obfuscation from circular security. Cryptology ePrint Archive, Report 2020/1010, 2020.

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th Annual ACM Symposium on Theory of Computing*, pages 197–206, Victoria, BC, Canada, May 17–20, 2008. ACM Press.

[GR07] Shafi Goldwasser and Guy N. Rothblum. On best-possible obfuscation. In Salil P. Vadhan, editor, *TCC 2007: 4th Theory of Cryptography Conference*, volume 4392 of *Lecture Notes in Computer Science*, pages 194–213, Amsterdam, The Netherlands, February 21–24, 2007. Springer, Heidelberg, Germany.

[GSW13] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part I*, volume 8042 of *Lecture Notes in Computer Science*, pages 75–92, Santa Barbara, CA, USA, August 18–22, 2013. Springer, Heidelberg, Germany.

[Had00] Satoshi Hada. Zero-knowledge and code obfuscation. In Tatsuaki Okamoto, editor, *Advances in Cryptology – ASIACRYPT 2000*, volume 1976 of *Lecture Notes in Computer Science*, pages 443–457, Kyoto, Japan, December 3–7, 2000. Springer, Heidelberg, Germany.

[HJ16] Yupu Hu and Huiwen Jia. Cryptanalysis of GGH map. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016, Part I*, volume 9665 of *Lecture Notes in Computer Science*, pages 537–565, Vienna, Austria, May 8–12, 2016. Springer, Heidelberg, Germany.

[HJL21] Sam Hopkins, Aayush Jain, and Huijia Lin. Counterexamples to new circular security assumptions underlying io. Springer-Verlag, 2021.

[JLMS19] Aayush Jain, Huijia Lin, Christian Matt, and Amit Sahai. How to leverage hardness of constant-degree expanding polynomials overa $\mathbb{R}$ to build $i\mathcal{O}$. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2019, Part I*, volume 11476 of *Lecture Notes in Computer Science*, pages 251–281, Darmstadt, Germany, May 19–23, 2019. Springer, Heidelberg, Germany.

[JLS20] Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from well-founded assumptions. Cryptology ePrint Archive, Report 2020/1003, 2020.

[Lin16] Huijia Lin. Indistinguishability obfuscation from constant-degree graded encoding schemes. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016, Part I*, volume 9665 of *Lecture Notes in Computer Science*, pages 28–57, Vienna, Austria, May 8–12, 2016. Springer, Heidelberg, Germany.

[Lin17] Huijia Lin. Indistinguishability obfuscation from SXDH on 5-linear maps and locality-5 PRGs. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017, Part I*, volume 10401 of *Lecture Notes in Computer Science*, pages 599–629, Santa Barbara, CA, USA, August 20–24, 2017. Springer, Heidelberg, Germany.

[LPST16a] Huijia Lin, Rafael Pass, Karn Seth, and Sidharth Telang. Indistinguishability obfuscation with non-trivial efficiency. In Chen-Mou Cheng, Kai-Min Chung, Giuseppe Persiano, and Bo-Yin Yang, editors, *PKC 2016: 19th International Conference on Theory and Practice of Public Key Cryptography, Part II*, volume 9615 of *Lecture Notes in Computer Science*, pages 447–462, Taipei, Taiwan, March 6–9, 2016. Springer, Heidelberg, Germany.

[LPST16b] Huijia Lin, Rafael Pass, Karn Seth, and Sidharth Telang. Output-compressing randomized encodings and applications. In Eyal Kushilevitz and Tal Malkin, editors, *TCC 2016-A: 13th Theory of Cryptography Conference, Part I*, volume 9562 of *Lecture Notes in Computer Science*, pages 96–124, Tel Aviv, Israel, January 10–13, 2016. Springer, Heidelberg, Germany.

[LT17] Huijia Lin and Stefano Tessaro. Indistinguishability obfuscation from trilinear maps and block-wise local PRGs. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017, Part I*, volume 10401 of *Lecture Notes in Computer Science*, pages 630–660, Santa Barbara, CA, USA, August 20–24, 2017. Springer, Heidelberg, Germany.

[LV16] Huijia Lin and Vinod Vaikuntanathan. Indistinguishability obfuscation from DDH-like assumptions on constant-degree graded encodings. In Irit Dinur, editor, *57th Annual Symposium on Foundations of Computer Science*, pages 11–20, New Brunswick, NJ, USA, October 9–11, 2016. IEEE Computer Society Press.

[MSZ16] Eric Miles, Amit Sahai, and Mark Zhandry. Annihilation attacks for multilinear maps: Cryptanalysis of indistinguishability obfuscation over GGH13. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016, Part II*, volume 9815 of *Lecture Notes in Computer Science*, pages 629–658, Santa Barbara, CA, USA, August 14–18, 2016. Springer, Heidelberg, Germany.

[MZ18] Fermi Ma and Mark Zhandry. The MMap strikes back: Obfuscation and new multilinear maps immune to CLT13 zeroizing attacks. In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018: 16th Theory of Cryptography Conference, Part II*, volume 11240 of *Lecture Notes in Computer Science*, pages 513–543, Panaji, India, November 11–14, 2018. Springer, Heidelberg, Germany.

[OPP14] Rafail Ostrovsky, Anat Paskin-Cherniavsky, and Beni Paskin-Cherniavsky. Maliciously circuit-private FHE. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology – CRYPTO 2014, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 536–553, Santa Barbara, CA, USA, August 17–21, 2014. Springer, Heidelberg, Germany.

[PRS17] Chris Peikert, Oded Regev, and Noah Stephens-Davidowitz. Pseudorandomness of ring-LWE for any ring and modulus. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *49th Annual ACM Symposium on Theory of Computing*, pages 461–473, Montreal, QC, Canada, June 19–23, 2017. ACM Press.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th Annual ACM Symposium on Theory of Computing*, pages 84–93, Baltimore, MA, USA, May 22–24, 2005. ACM Press.

[SW14] Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In David B. Shmoys, editor, *46th Annual ACM Symposium on Theory of Computing*, pages 475–484, New York, NY, USA, May 31 – June 3, 2014. ACM Press.

[WW20] Hoeteck Wee and Daniel Wichs. Candidate obfuscation via oblivious lwe sampling. Cryptology ePrint Archive, Report 2020/1042, 2020.

A More Preliminaries

A.1 Learning with Errors

We recall the definition of the learning with errors (LWE) problem [Reg05].

Definition 4 (Learning with Errors). *The LWE problem is parametrized by a modulus q, positive integers (n, m) and an error distribution χ. The LWE problem is hard if the following distributions are computationally indistinguishable:*

$$(\mathbf{A}, \mathbf{A} \cdot \mathbf{s} + \mathbf{e}) \approx (\mathbf{A}, \mathbf{u})$$

where $\mathbf{A} \leftarrow_{\$} \mathbb{Z}_q^{m \times n}$, $\mathbf{s} \leftarrow_{\$} \mathbb{Z}_q^n$, $\mathbf{u} \leftarrow_{\$} \mathbb{Z}_q^m$, and $\mathbf{e} \leftarrow_{\$} \chi^m$.

As shown in [Reg05, PRS17], for *any* sufficiently large modulus q the LWE problem where χ is a discrete Gaussian distribution with parameter $\sigma = \alpha q \geq 2\sqrt{n}$ (i.e. the distribution over $\mathbb{Z}$ where the probability of x is proportional to $e^{-\pi(|x|/\sigma)^2}$), is at least as hard as approximating the shortest independent vector problem (SIVP) to within a factor of $\gamma = \tilde{O}(n/\alpha)$ in *worst case* dimension n lattices. We refer to $\alpha = \sigma/q$ as the *modulus-to-noise* ratio, and by the above this quantity controls the hardness of the LWE instantiation. Hereby, LWE with polynomial α is (presumably) harder than LWE with super-polynomial or sub-exponential α. We can truncate the discrete Gaussian distribution χ to $\sigma \cdot \omega(\sqrt{\log(\lambda)})$ while only introducing a negligible error. Consequently, we typically omit the actual distribution χ but only use the fact that it can be bounded by a (small) value B.

A.2 Public-Key Encryption

We recall the definition of public key encryption in the following.

Definition 5 (Public-Key Encryption). *A public-key encryption scheme consists of the following efficient algorithms.*

$\underline{\mathsf{KeyGen}(1^\lambda)}$**:** *On input the security parameter 1^λ, the key generation algorithm returns a key pair $(\mathsf{sk}, \mathsf{pk})$.*
$\underline{\mathsf{Enc}(\mathsf{pk}, \mathsf{m})}$**:** *On input a public key pk and a message m, the encryption algorithm returns a ciphertext c.*
$\underline{\mathsf{Dec}(\mathsf{sk}, c)}$**:** *On input the secret key sk and a ciphertext c, the decryption algorithm returns a message m.*

Correctness and Semantic Security. We recall the standard notions of correctness and semantic security [GM82] for public-key encryption.

Definition 6 (Correctness). *A public-key encryption scheme $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ is correct if for all $\lambda \in \mathbb{N}$, all messages m, all $(\mathsf{sk}, \mathsf{pk})$ in the support of $\mathsf{KeyGen}(1^\lambda)$, and all c in the support of $\mathsf{Enc}(\mathsf{pk}, \mathsf{m})$ it holds that $\mathsf{Dec}(\mathsf{sk}, c) = \mathsf{m}$.*

Definition 7 (Semantic Security). *A public-key encryption scheme $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ is semantically secure if for all $\lambda \in \mathbb{N}$, all pairs of message $(\mathsf{m}_0, \mathsf{m}_1)$, it holds that the following distributions are computationally indistinguishable*

$$(\mathsf{pk}, \mathsf{Enc}(\mathsf{pk}, \mathsf{m}_0)) \approx (\mathsf{pk}, \mathsf{Enc}(\mathsf{pk}, \mathsf{m}_1))$$

where $(\mathsf{sk}, \mathsf{pk}) \leftarrow_{\$} \mathsf{KeyGen}(1^\lambda)$.

Homomorphic Encryption. We say that a public-key encryption scheme $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ is homomorphic for the circuit class $\{\mathfrak{C}_\lambda\}_{\lambda\in\mathbb{N}}$ if there exists an efficient deterministic algorithm Eval such that for all $\Pi \in \mathfrak{C}_\lambda$, all $(\mathsf{sk}, \mathsf{pk})$ in the support of KeyGen, all vectors of messages $(\mathsf{m}_1, \ldots, \mathsf{m}_\mu)$, all ciphertexts $(c_1, \ldots, c_\mu)$ in the support of $(\mathsf{Enc}(\mathsf{pk}, \mathsf{m}_1), \ldots, \mathsf{Enc}(\mathsf{pk}, \mathsf{m}_\mu))$ it holds that

$$\mathsf{Dec}(\mathsf{sk}, \mathsf{Eval}(\mathsf{pk}, \Pi, (c_1, \ldots, c_\mu))) = \Pi(\mathsf{m}_1, \ldots, \mathsf{m}_\mu).$$

Furthermore, we say that a scheme is *fully-homomorphic* if it is homomorphic for all polynomial-size circuits.

Randomness-Key Circularity. We say that two encryption schemes $(\mathsf{KeyGen}_0, \mathsf{Enc}_0, \mathsf{Dec}_0)$ and $(\mathsf{KeyGen}_1, \mathsf{Enc}_1, \mathsf{Dec}_1)$ form a key cycle if the distinguisher is given a cross-encryption of the secret keys and the randomness used for the latter, i.e. $\mathsf{Enc}(\mathsf{pk}_1, \mathsf{sk}_0; s_1)$ and $\mathsf{Enc}(\mathsf{pk}_0, s_1)$. We say that the scheme is randomness-key circular secure if semantic security is retained in the presence of such a cycle.

Definition 8 (Randomness-Key Circular Security). *A pair of public-key encryption schemes $(\mathsf{KeyGen}_0, \mathsf{Enc}_0, \mathsf{Dec}_0)$ and $(\mathsf{KeyGen}_1, \mathsf{Enc}_1, \mathsf{Dec}_1)$ is randomness-key circular secure if for all $\lambda \in \mathbb{N}$, all pairs of message $(\mathsf{m}_0, \mathsf{m}_1)$, it holds that the following distributions are computationally indistinguishable*

$$\begin{aligned}&(\mathsf{pk}_0, \mathsf{pk}_1, \mathsf{Enc}_0(\mathsf{pk}_0, \mathsf{sk}_1; r), \mathsf{Enc}_1(\mathsf{pk}_1, r), \mathsf{Enc}_0(\mathsf{pk}_0, \mathsf{m}_0))\\ \approx\ &(\mathsf{pk}_0, \mathsf{pk}_1, \mathsf{Enc}_0(\mathsf{pk}_0, \mathsf{sk}_1; r), \mathsf{Enc}_1(\mathsf{pk}_1, r), \mathsf{Enc}_0(\mathsf{pk}_0, \mathsf{m}_1))\end{aligned}$$

where $(\mathsf{sk}_0, \mathsf{pk}_0) \leftarrow_{\$} \mathsf{KeyGen}_0(1^\lambda)$, $(\mathsf{sk}_1, \mathsf{pk}_1) \leftarrow_{\$} \mathsf{KeyGen}_1(1^\lambda)$, and $r \leftarrow_{\$} \{0,1\}^\lambda$.

B Analysis

In the following we show that our shallow XiO scheme satisfies the notion of security for indistinguishability obfuscation.

Theorem 3 (Shallow XiO Security). *If the GSW scheme $(\mathsf{GSW.KeyGen}, \mathsf{GSW.Enc}, \mathsf{GSW.Eval}, \mathsf{GSW.Dec})$ and the packed dual-Regev scheme $(\mathsf{dR.KeyGen}, \mathsf{dR.Enc}, \mathsf{dR.Eval}, \mathsf{dR.Dec})$ are key-randomness SRL secure, then the XiO scheme as described above is secure.*

Proof. We prove the scheme via a series of hybrid experiments.

- Hybrid $\mathcal{H}_0$: This is the original obfuscation of the circuit Π_0.
- Hybrid $\mathcal{H}_1$: This hybrid is identical to the previous one, except that for all $i \in \{0,1\}^{\eta-\log(k)}$ we sample $\mathbf{C}'_{i,\mathsf{Round}}$ as
$$\mathbf{C}'_{i,\mathsf{Round}} = \begin{bmatrix} \mathsf{GSW.ColEnc}(\mathsf{pk}, \mathsf{Round}(u_{i,1}); \mathbf{r}^*_{i,1}) \\ \dots \\ \mathsf{GSW.ColEnc}(\mathsf{pk}, \mathsf{Round}(u_{i,k}); \mathbf{r}^*_{i,k}) \end{bmatrix}$$
where $\mathbf{r}^*_{i,j} \leftarrow\!\!\$\, [-\tilde{B}, +\tilde{B}]^m$. Let $(\mathbf{r}_{\Psi,i,1}, \dots, \mathbf{r}_{\Psi,i,k})$ be the random coins of $\mathbf{C}_{i,\mathsf{Round}}$ (as computed in the original protocol). We additionally set the i-the block of the $\mathsf{GSW.PubCoin}$ to $(\mathbf{r}^*_{i,1} - \mathbf{r}_{\Psi,i,1}, \dots, \mathbf{r}^*_{i,k} - \mathbf{r}_{\Psi,i,k})$. Statistical indistinguishability with respect to the previous hybrid follows from an invocation of Lemma 2.
- Hybrid $\mathcal{H}_2$: This hybrid is identical to the previous one, except that for all $i \in \{0,1\}^{\eta-\log(k)}$ we set
$$(h_{i,1}, \dots, h_{i,k}) = \mathbf{B} \cdot \mathbf{r}_i + (u_{i,1}, \dots, u_{i,k})$$
where $(u_{i,1}, \dots, u_{i,k}) \leftarrow\!\!\$\, \mathbb{Z}_q^k$. The only difference with respect to the previous hybrid is that the obfuscator knows the values $(u_{i,1}, \dots, u_{i,k})$ ahead of time. However, the two distributions are identical to the eyes of the distinguisher and therefore the change here is only syntactical.
- Hybrid $\mathcal{H}_3$: In this hybrid we generate, for all $i \in \{0,1\}^{\eta-\log(k)}$, $\bar{\mathbf{c}}_i$ as follows
$$\bar{\mathbf{c}}_i = \mathbf{B} \cdot \mathbf{t}_i + \boldsymbol{\xi}_i + \boldsymbol{\zeta}_i + q/2 \cdot \left(\mathsf{Round}(u_1) \oplus \Pi_1^{(i)}, \dots, \mathsf{Round}(u_k) \oplus \Pi_k^{(i)}\right) + \mathbf{u}_i$$
where $\mathbf{t}_i \leftarrow\!\!\$\, \mathbb{Z}_q^n$. Here $\boldsymbol{\xi}_i$ and $\boldsymbol{\zeta}_i$ denote the decryption noises of $\bar{\mathbf{C}}_{\mathsf{sk}}$ and $\mathbf{D}_i$, respectively. Furthermore, we set $(h_{i,1}, \dots, h_{i,k}) = \bar{\mathbf{c}}_i - \bar{\mathbf{C}}_{\mathsf{sk}} \cdot \mathsf{Bit}(\boldsymbol{\ell}_i)$ and $\mathbf{c}_{i,\mathbf{r}} = \mathsf{GSW.Enc}(\mathsf{pk}, \mathbf{t}_i - \mathbf{S} \cdot \mathsf{Bit}(\boldsymbol{\ell}_i))$. Note that $\mathbf{C}'_{i,\mathsf{Round}}$ is fixed (and in particular is independent of $\mathbf{c}_{i,\mathbf{r}}$) and thus the above variables are always well defined. In fact, observe that
$$\begin{aligned} &(h_{i,1}, \dots, h_{i,k}) \\ &= \bar{\mathbf{c}}_i - \bar{\mathbf{C}}_{\mathsf{sk}} \cdot \mathsf{Bit}(\boldsymbol{\ell}_i) \\ &= \bar{\mathbf{c}}_i - \mathbf{B} \cdot (\mathbf{S} \cdot \mathsf{Bit}(\boldsymbol{\ell}_i)) - \boldsymbol{\xi}_i - \boldsymbol{\zeta}_i - q/2 \cdot \left(\mathsf{Round}(u_1) \oplus \Pi_1^{(i)}, \dots, \mathsf{Round}(u_k) \oplus \Pi_k^{(i)}\right) \\ &= \mathbf{B} \cdot (\mathbf{t}_i - \mathbf{S} \cdot \mathsf{Bit}(\boldsymbol{\ell}_i)) + \mathbf{u}_i \end{aligned}$$
which is exactly the same distribution as in the previous hybrid. Furthermore, note that we now have $\boldsymbol{\rho}_i = \boldsymbol{t}_i$. Thus the change introduced here is only syntactical.
- Hybrid $\mathcal{H}_4$: In this hybrid we generate, for all $i \in \{0,1\}^{\eta-\log(k)}$, $\bar{\mathbf{c}}_i$ as a fresh encryption of $\Pi^{(i)}$, i.e.
$$\bar{\mathbf{c}}_i = \mathbf{B} \cdot \mathbf{t}_i + q/2 \cdot \Pi^{(i)} + \mathbf{w}_i$$
where $\mathbf{t}_i \leftarrow\!\!\$\, \mathbb{Z}_q^n$ and $\mathbf{w}_i \leftarrow\!\!\$\, [-q/4, +q/4]^k$. Note that we can bound $\|\boldsymbol{\xi}_i\|_\infty \leq B \cdot \log(q) \cdot k \cdot (n+1)$. Furthermore, we have that $\|\boldsymbol{\zeta}_i\|_\infty \leq \tilde{B}$ with all but negligible probability over the random choice of $\mathbf{r}^*_i$, by Lemma 1. Statistical indistinguishability follows from another application of Lemma 1.
- Hybrid $\mathcal{H}_5$: Here we define, for all $i \in \{0,1\}^{\eta-\log(k)}$, the following circuit $\Gamma_i : \mathbb{Z}_q^{n \times (n+1) \cdot k \cdot \log(q)} \to \mathbb{Z}_q^n$ to return
$$\Gamma_i(\mathbf{X}) = \mathbf{t}_i - \mathbf{X} \cdot \mathsf{Bit}(\boldsymbol{\ell}_i).$$
Then we set $\mathbf{c}_{i,\mathbf{r}} = \mathsf{GSW.Eval}(\mathsf{pk}, \Gamma_i, \mathsf{GSW.Enc}(\mathsf{pk}, \mathbf{S}))$. Recall that $\mathbf{S}$ are the random coins used in the encryption of sk. To see why the hybrids are statistically close, note that
$$\mathsf{GSW.Eval}(\mathsf{pk}, \Gamma_i, \mathsf{GSW.Enc}(\mathsf{pk}, \mathbf{S})) \approx \mathsf{GSW.Enc}(\mathsf{pk}, \mathbf{t}_i - \mathbf{S} \cdot \mathsf{Bit}(\boldsymbol{\ell}_i))$$
by the statistical circuit privacy of GSW.

- Hybrid $\mathcal{H}_6$: In this hybrid we compute $\mathbf{c}_\Pi$ as an encryption of Π_1 instead of Π_0. Note that we no longer use the secret key of either of the encryption schemes to obfuscate the circuit, besides the encrypted key-randomness cycle. The random coins $\mathsf{GSW.PubCoin}$ can be modelled as an SRL leakage. More in details, we are going to reduce the security of the XiO scheme to the key-randomness SRL security of GSW and packed dual-Regev. Since the distinguisher is expecting an obfuscation of either Π_0 or Π_1, it suffices to describe how to construct the corresponding XiO given the information provided by the challenger.
  The reduction receives from the challenger the public keys $(\mathsf{pk}, \mathbf{B})$ and the GSW encryption $\mathbf{c}_\Pi$ (of either Π_0 or Π_1). The reduction sets $(\mathsf{pk}, \mathbf{B})$ to be the public keys of the XiO and $\mathbf{c}_\Pi$ to be the encryption of the obfuscated circuit. Then it homomorphically computes $(\mathbf{P}_1, \dots, \mathbf{P}_{2^{\eta-\log(k)}})$ as the encryptions of the blocks of the truth table $(\Pi^{(1)}, \dots, \Pi^{(2^{\eta-\log(k)})})$, by evaluating the universal circuit on $\mathbf{c}_\Pi$, and it returns them along with their plaintexts to the challenger. In response, the reduction receives $(\mathbf{c}_{1,\mathbf{r}}, \dots, \mathbf{c}_{2^{\eta-\log(k)},\mathbf{r}})$, $(\mathbf{C}_1^*, \dots, \mathbf{C}^*_{2^{\eta-\log(k)}})$, $\bar{\mathbf{C}}_{\mathsf{sk}}$, $(\mathbf{u}_1, \dots, \mathbf{u}_{2^{\eta-\log(k)}})$, $(\mathbf{t}_1, \dots, \mathbf{t}_{2^{\eta-\log(k)}})$, and $(\hat{\mathbf{r}}_1, \dots, \hat{\mathbf{r}}_{2^{\eta-\log(k)}})$. For all $i = 1 \dots 2^{\eta-\log(k)}$, the reduction proceeds as follows:
  - Parse $\mathbf{u}_i = \mathbf{w}_i + q/2 \cdot \mathbf{v}_i$.
  - Compute $\mathbf{D}_i = \mathbf{P}_i + \mathbf{C}_i^* + q/2 \cdot \mathbf{v}_i$ and let ℓ_i be the corresponding linear function.
  - Compute $h_i = \mathbf{B} \cdot \mathbf{t}_i + q/2 \cdot \Pi^{(i)} + \mathbf{w}_i - \bar{\mathbf{C}}_{\mathsf{sk}} \cdot \mathsf{Bit}(\boldsymbol{\ell}_i)$.

  The rest of the obfuscated circuit is computed by simply copying the appropriate variable provided by the challenger. A routine calculation shows that the view produced by the reduction is identical to $\mathcal{H}_5$ if $\mathbf{c}_\Pi$ encrypts Π_0 and identical to $\mathcal{H}_6$ if $\mathbf{c}_\Pi$ encrypts Π_1. Furthermore, note that the reduction is admissible by the evaluation correctness of GSW. Thus a distinguisher between the two hybrids contradicts the key-randomness SRL security.
- Hybrids $\mathcal{H}_7 \dots \mathcal{H}_{11}$: In this series of hybrids we undo all changes that we did in $\mathcal{H}_5 \dots \mathcal{H}_1$. The statistical indistinguishability follows by the same arguments as before.

Note that $\mathcal{H}_{11}$ is the original obfuscation of Π_1. This concludes our proof.

B.1 Parameters

The analysis of our scheme sets two constraints on the noise growth of the encryption schemes. The first application of the smudging Lemma requires that the noise bound $\tilde{B}$ is exponentially larger than the noise bound on evaluated GSW ciphertexts $\hat{B}$, i.e. (1) $\tilde{B} \geq 2^\lambda \cdot \hat{B}$. The second application requires that (2) $q/4 \geq 2^\lambda \cdot \tilde{B}$ and (3) $q/4 \geq 2^\lambda \cdot B \cdot \log(q) \cdot k \cdot (n+1)$, to ensure that the term u_i (minus its most significant bit) properly floods the noise terms of the GSW and dual-Regev ciphertexts. Note that, for an appropriate choice of parameters of the GSW scheme, condition (3) is already implied by conditions (1) and (2) and therefore all we need to ensure is that these two conditions are satisfiable. It is not hard to see that the above pair of constraints can be satisfied by setting the modulo-to-noise ratio of the LWE assumption to be super-polynomial.

To make the scheme compressing, we set $k = 2^{\eta/4}$. We analyze the size of each component of the obfuscated circuit in the following:

- The size of the public keys $(\mathsf{pk}, \bar{\mathsf{pk}})$ is linear in k and thus can be bounded by $2^{\eta/4} \cdot \mathsf{poly}(\lambda)$.
- The size of the encrypted key $\bar{\mathbf{C}}_{\mathsf{sk}}$ can be bounded by $k^2 \cdot \mathsf{poly}(\lambda) = 2^{\eta/2} \cdot \mathsf{poly}(\lambda)$.
- The GSW encryption of the circuit $\mathbf{c}_\Pi$ is of size $|\Pi| \cdot \mathsf{poly}(\lambda)$.
- The size of an encrypted header $\mathbf{c}_{i,\mathbf{r}}$ is $\mathsf{poly}(\lambda)$ (and in particular independent of k) and the size of each decryption hint $\boldsymbol{\rho}_i$ is also $\mathsf{poly}(\lambda)$ (by the randomness succinctness of dual-Regev). It follows that the total size of the encryption headers and the decryption hints can be bounded by $2^{\eta-\log(k)} \cdot \mathsf{poly}(\lambda) = 2^{3\eta/4} \cdot \mathsf{poly}(\lambda)$.

The summation of the above terms is sublinear in $\mathsf{poly}(|\Pi|, \lambda) \cdot 2^{\eta(1-\varepsilon)}$ (for $\varepsilon > 0$) and therefore the XiO scheme satisfies non-trivial efficiency. Note that we omitted the public parameters $\mathsf{GSW.PubCoin}$ and $\mathsf{dR.PubCoin}$ from the analysis since they are uniformly at random and thus can be computed in the preprocessing stage of the obfuscator.

Finally, we argue that the scheme is *shallow*. The size of the ciphertext $\mathbf{c}_\Pi$ depends only on the size of the circuit $|\Pi|$ and the security parameter, which is also a trivial depth on the computation needed to output $\mathbf{c}_\Pi$. This is also the case for each encryption header $\mathbf{c}_{i,\mathbf{r}}$ and each decryption hint $\boldsymbol{\rho}_i$, which can be all computed in parallel. Thus, all is left to argue is that the public keys $(\mathsf{pk}, \bar{\mathsf{pk}})$ and the encrypted key $\bar{\mathbf{C}}_{\mathsf{sk}}$ can be computed in parallel with depth independent of k. This is easily verified by observing that in the scheme in Section 3 both the key generation and the encryption algorithm can be implemented in depth independent of k.

C From Shallow XiO to iO

C.1 Background

We give some background on the notion of exponentially efficient iO (XiO), introduced by Lin, Pass, Seth and Telang [LPST16a]. They showed that this seemingly very weak notion of iO (to be explained below) in fact suffices to construct fully-fledged iO, relying on a prior work of the same authors [LPST16b]. We refer to the collection of both works as LPST. The transformation uses a construction of a so-called functional encryption scheme (FE) by Goldwasser et al. [GKP$^+$13] which is based on the widely studied Learning with Errors (LWE) assumption [Reg05].

The idea of LPST is quite simple and elegant. Functional encryption is closely related to program obfuscation. Specifically, this is an encryption scheme where it is possible to generate a functional secret key sk_f for a function f, so that decrypting a ciphertext $\mathsf{Enc}(x)$ using sk_f will output $f(x)$, and furthermore the pair $\mathsf{sk}_f, \mathsf{Enc}(x)$ does not reveal any information about x beyond the value $f(x)$. It has been established [BV15, AJ15, LPST16b] that if we can construct an FE scheme which supports functions f that contain the scheme's own encryption algorithm, executed twice (in parallel), plus some additional calculations, then a construction of indistinguishability obfuscation follows. The [GKP$^+$13] scheme does not have this property. Its encryption complexity and output length scales roughly with $\ell \cdot \mathsf{poly}(d)$, where ℓ is the output length of f and d is the depth of f. Namely, the blowup of $2 \cdot \mathsf{poly}(d)$ in the output length prevents us from instantiating the obfuscation construction with this scheme.

The idea in [LPST16a] is to modify this scheme as follows. First, instead of issuing a key sk_f for a multi output-bit function f, a key $\mathsf{sk}_{f'}$ for a single output-bit f' is issued. The function f' takes as input a pair (x, i) and outputs a single bit, such that $f'(x, i) = (f(x))_i$, i.e. $f'(x, i)$ is the i-th bit of $f(x)$. Furthermore, the encryption process is modified such that the encryptor does not compute $\mathsf{Enc}(x)$, but instead it considers the circuit C that takes an index i as input, and outputs a ciphertext $\mathsf{Enc}(x, i)$ encrypting the pair (x, i). The description of this circuit is independent of ℓ, d, and only its input length depends on them. In terms of functionality, instead of outputting $\mathsf{Enc}(x, i)$, the encryption algorithm can produce C itself, thus deferring the process of generating $\mathsf{Enc}(x, i)$ to the time of decryption. Namely, the decryptor will first compute $C(i)$ for all i in order to obtain $\mathsf{Enc}(x, i)$ and then proceed as in the original scheme. Namely, given the ciphertexts $\mathsf{Enc}(x, 1), \ldots, \mathsf{Enc}(x, \ell)$ and the secret key $\mathsf{sk}_{f'}$, the decryptor can recover $f(x)$ as $f(x) = (f'(x, 1), \ldots, f'(x, \ell))$. In terms of security, the description of C reveals x and therefore security is not preserved. LPST thus propose to output $\hat{C}$, an obfuscation C, which on one hand will hide x and on the other will allow to produce $\mathsf{Enc}(x, i)$ by computing all $\hat{C}(i)$ just as with the original C.

LPST thus propose a new notion of obfuscation which is suitable for the above task. Note that we only need to obfuscate circuits whose entire truth table is of length $\ell \cdot \mathsf{poly}(d)$ which is polynomial. Therefore we can afford an obfuscator that runs in time polynomial in the *truth-table size* of the circuit to be obfuscated, and the resulting obfuscated program needs to be sufficiently short, LPST require that that this scales with $\ell_C^{1-\epsilon}$, where $\ell_C = \ell \cdot \mathsf{poly}(d)$ is the truth-table size of C and $\epsilon > 0$ is an arbitrary constant. They defined this notion as "exponentially-efficient iO" or XiO. Indeed, setting $\hat{C}$ to be the XiO of C will allow to compress the output size to be $\ell^{1-\epsilon} \cdot \mathsf{poly}(d, |C|) \ll \ell$ (typically $\ell \gg |C|$ so $\ell^{1-\epsilon} \ll \ell \mathsf{poly}(|C|)$). Let us from here on denote by $\widehat{\mathsf{Enc}}(x)$ the modified encryption procedure, that applies the XiO on C and outputs $\hat{C}$ as output. Let us denote by $\hat{f}$ the derived function f which is required to execute $\widehat{\mathsf{Enc}}$ twice plus some additional computations.

Indeed, now $\hat{f}$ has output length that is smaller than ℓ, so we may hope that it is possible to use our new scheme in order to derive an iO construction. However, there is still the obstacle of depth since the depth of $\hat{f}$ might be larger than the depth d that is supported by the FE scheme. Indeed, importantly for this work, we consider a case where the XiO obfuscator requires large depth, specifically depth that is also proportional to $\mathsf{poly}(\ell, |C|)$. Such parameters are allowed by the XiO definition of [LPST16a]. Consequently, in this case, the encryption $\widehat{\mathsf{Enc}}(x)$ and the function $\hat{f}$ will also inherit the $\mathsf{poly}(\ell, |C|)$ depth. This value is much larger than the depth d supported by the FE scheme and consequently this solution is insufficient.

LPST were aware of this difficulty, and they propose a solution. This is described in the paragraph above [LPST16b, Theorem 5], which is also quoted in [LPST16a, Theorem 5]. LPST argue that prior works showed that the depth limitation can be avoided by using the notion of garbled circuits (see [LPST16a, Theorem 4]). To the best of our knowledge, the result in [LPST16a, Theorem 4] is currently not known to exist and indeed is a very important open problem in the design of functional encryption schemes. While LPST do not provide a rigorous proof for this claim, we can only assume that their intention was as follows. Instead of considering the function $\hat{f}$, to consider its so-called "garbling". Without getting into details let us only mention that the garbling of *any* circuit is a *shallow* circuit whose depth is independent of the depth of the original circuit, and garbling also preserves the privacy properties of the original function. However, crucially, the *output length* of the garbling procedure of $\hat{f}$ is proportional to $|\hat{f}| = \mathsf{poly}(\ell, |C|)$. Therefore, while garbling indeed takes care of the depth problem, it re-creates the output length problem. In fact, we will show next that composing general XiO, FE and garbled circuits (GC) in any order cannot be used to make both the output length of the new f less than ℓ and the depth be less than d. In fact the sum of these two values is always greater than $\ell + d$.

C.2 Why Composition of Existing Primitives Cannot Work

We propose a method to analyze LPST-style transformations, where a composition of primitives is used to achieve succinctness. We will also use this method to analyze our own version of this transformation. We identify 3 properties of circuits that are of interest in the context of the FE to iO transformation. For a circuit C and input x, we let ℓ denote the length of $C(x)$, i.e. the output length of C. Now, rather than considering C as a single circuit, we consider the circuit that computes a single bit out of the output $C(x)$. Namely, the circuit that takes $i \in [\ell]$ as input, and outputs the ith bit of $C(x)$. Let d and s be the depth and circuit-size, respectively, of this bit-wise circuit (note that by definition it always holds $s \geq d$).

We can now view all primitives of interest, namely functional encryption, XiO and garbled circuits as methods of encoding a circuit-input pair (C, x) with length, depth and size (ℓ, d, s), via some encoding that on one hand allows to produce the value $C(x)$ and on the other does not allow to learn any information about x other than $C(x)$[5]. We can thus consider, for each such encoding method, how the (ℓ, d, s) values of the encoding function relates to the (ℓ, d, s) of the original C, x. We call this the *scaling characteristic* of the encoding, and it quantifies the complexity of encoding a pair of circuit C and input x compared to the complexity of just evaluating $C(x)$. We would like to stress that only the complexity of encoding determines the scaling characteristic, *not the complexity of evaluating such an encoding.*

The [GKP$^+$13] FE scheme translates into an encoding with scaling characteristic

$$\mathsf{SC}_{\mathsf{FE}}(\ell, d, s) = (\ell \cdot \mathsf{poly}(d), \mathsf{polylog}(s), \mathsf{polylog}(s)).$$

An XiO scheme is an encoding with scaling characteristic

$$\mathsf{SC}_{\mathrm{XiO}}(\ell, d, s) = (\ell^{1-\epsilon}, \mathsf{poly}(\ell, s), \mathsf{poly}(\ell, s)).$$

The aforementioned garbled-circuit (whose exact properties are not important for this discussion) can be considered as an encoding with scaling characteristic *at least* [6]

$$\mathsf{SC}_{\mathsf{GC}}(\ell, d, s) = (\ell + d, 1, 1).$$

[5] In a well-defined sense which we do not elaborate on at this point.

[6] All of the expressions for scaling characteristics hide polynomials in the so-called security-parameter. However for the purpose of the negative result, it only makes the result stronger to omit them.

The LPST approach is to compose these encodings in some order, where the goal is to arrive at a *compressing* encoding, i.e. one where all ℓ, d, s of the encoded function are smaller than the respective parameters of the original circuit. However, when considering these transformations in terms of their scaling characteristics, it is straightforward to see that this cannot be the case using the above components, since all of them have the invariant that the amount $\ell + d$ only increases after the encoding, and therefore either ℓ or d will be greater post-encoding than its pre-encoding value. In fact, for the purpose of the LPST transformation, there is no advantage in using the [GKP$^+$13] FE scheme compared to use of garbled-circuit based FE which can be constructed under much milder assumption (indeed, looking at the LPST paper, the unique properties of [GKP$^+$13] are never used).

Note that our analysis holds even in the case where composition preserves security "for free". In actuality, there are often additional overhead required in order to perform the composition in a way where security of the composed primitive is preserved, which could make the parameters of the composed primitive even *worse*. This makes our analysis fairly robust to the specific notion of security used, and it still holds even if all components were to satisfy the strongest possible security notion (such as simulation-based security).

We therefore conclude that beyond citing a result that is currently out of reach in [LPST16a, Theorem 4], it appears to be inherently impossible to apply the approach via a simple composition of currently existing primitives in order to obtain a "bootstrappable" FE scheme.

C.3 Bitwise Complexity and Scaling Characteristic

We now describe a useful measure for several cryptographic primitive of interest that characterizes the amount of compression that they provide. Such compression be expressed via a quantity we call the scaling characteristic SC, which describes how the output length, computation depth and computational complexity per output bit of the encoded computation scales with that of the unencoded version. To ease notation, we often drop the security parameter from the description below. All algorithms referred to below are potentially randomized, unless explicitly mentioned otherwise.

Bit-Wise Complexity of a Computation. Let C be some boolean circuit. We let ℓ be its output length. Let x be an input for C. We consider the bit-wise depth and size of the execution of C on x, as the depth and size of the circuit that takes as input i and outputs the ith output bit of $C(x)$. We let $\mathsf{BWComp}(C, x)$ be a function whose inputs are C, x and its output is the tuple (ℓ, d, s) as described above.

Randomized Encoding. A randomized encoding (RE) scheme for a circuit C in the common reference string model is a tuple of algorithms $\mathrm{RE} = (\mathrm{RE.Init}, \mathrm{RE.Encode}, \mathrm{RE.Eval})$, such that for every input x of a suitable length it holds that $\mathrm{RE.Eval}(z, \mathrm{RE.Encode}_{C,z}(x)) = C(x)$ where $z = \mathrm{RE.Init}_C()$, and such that for all x_0, x_1, if $C(x_0) = C(x_1)$ then $(z, \mathrm{RE.Encode}_{C,z}(x_0))$ and $(z, \mathrm{RE.Encode}_{C,z}(x_1))$ are computationally indistinguishable[7]. We often consider a family of RE schemes $\mathcal{RE}$ which contains an encoding for every circuit C in some class of circuits $\mathcal{C}$. The scaling characteristic of a family $\mathcal{RE}$ is a function $\mathsf{SC}_{\mathcal{RE}} : \mathbb{N}^3 \to \mathbb{N}^3$ defined as follows. Let $\mathcal{C}_{|(\ell,d,s)} = \{C \in \mathcal{C} : \mathsf{BWComp}(C) = (\ell, d, s)\}$.

$$\mathsf{SC}_{\mathcal{RE}}(\ell, d, s) = \mathsf{max}_{C \in \mathcal{C}_{|(\ell,d,s)}, z, x} \mathsf{BWComp}(\mathrm{RE.Encode}_{C,z}, x) \ ,$$

where the maximum is computed over each coordinate independently. We note that the scaling characteristic also implicitly depends on the security parameter.

Indistinguishability Obfuscation, XiO, Shallow XiO. We note that iO for a family $\mathcal{C}$ can be viewed as a RE for the truth-table function $(\cdot)$, which takes as input a circuit C and outputs the entire truth table of C. Note that in this case ℓ is the size of the domain of C, and d, s are simply the depth and size of C. Furthermore, there is no common reference string in this case (i.e. the RE.Init algorithm produces an empty output). We define the scaling characteristic of an iO scheme in light of this perspective. Namely, for a candidate construction iO, its scaling characteristic is defined as its expansion when viewed as RE for $_{\ell,\mathtt{d},\mathtt{s}}$,

[7] While the standard security notion for garbled circuits is simulation based, it is sufficient for our context to only consider the weaker indistinguishability notion considered e.g. in [LPST16b].

which is the function restricted to circuits C of domain size ℓ, depth d and size s. Using this definition, the standard sought-after notion of iO is one where

$$\mathsf{SC}_{\mathrm{iO}}(\ell, d, s) = (\mathsf{poly}(s), \mathsf{poly}(s), \mathsf{poly}(s)).$$

In the literature this is referred to simply as "iO" but we call this notion "succinct iO" to distinguish from the next notion we consider. An exponentially efficient iO (XiO) [LPST16a] is an iO scheme for which there exists $\epsilon > 0$ s.t.

$$\mathsf{SC}_{\mathrm{XiO}}(\ell, d, s) = (\ell^{1-\epsilon} \cdot \mathsf{poly}(s), \mathsf{poly}(s, \ell), \mathsf{poly}(s, \ell)).$$

For the purpose of this work, we consider the notion of *Shallow XiO* (ShXiO), which is an XiO scheme where the compressed depth does not depend on ℓ, namely

$$\mathsf{SC}_{\mathrm{ShXiO}}(\ell, d, s) = (\ell^{1-\epsilon} \cdot \mathsf{poly}(s), \mathsf{poly}(s), \mathsf{poly}(s, \ell)).$$

Functional Encryption. A functional encryption scheme (FE) for a function class $\mathcal{C}$ is a tuple of algorithms $\mathsf{FE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ such that: $\mathsf{Setup}()$ outputs a public key pk and a master secret key msk, $\mathsf{KeyGen}(\mathsf{msk}, C)$ outputs a secret key sk_C where C is a circuit from some class $\mathcal{C}$, $\mathsf{Enc}(\mathsf{pk}, x)$ outputs a ciphertext c and it holds that $\mathsf{Dec}(\mathsf{KeyGen}(\mathsf{msk}, C), \mathsf{Enc}(\mathsf{pk}, x)) = C(x)$ for all $C \in \mathcal{C}$ and appropriate x. We will use a very rudimentary security requirement which nonetheless suffices for our purposes. The requirement is that for all (C, x_0, x_1), if $C(x_0) = C(x_1)$ then for properly generated $\mathsf{pk}, \mathsf{msk}$ it holds that $(\mathsf{pk}, \mathsf{KeyGen}(\mathsf{msk}, C), \mathsf{Enc}(\mathsf{pk}, x_0))$ and $(\mathsf{pk}, \mathsf{KeyGen}(\mathsf{msk}, C), \mathsf{Enc}(\mathsf{pk}, x_1))$ are computationally indistinguishable.

We now note that an FE scheme also induces a RE scheme by setting $\mathrm{RE.Init}_C()$ to be the algorithm that generates $z = (\mathsf{pk}, \mathsf{KeyGen}(\mathsf{msk}, C))$ and $\mathrm{RE.Encode}_{C,z}(x) = \mathsf{Enc}(\mathsf{pk}, x)$. The scaling characteristic of a family of FE schemes $\mathcal{FE}$ is defined as the scaling characteristic of the RE family induced by it as above. To be more explicit:

$$\mathsf{SC}_{\mathcal{FE}}(\ell, d, s) = \mathsf{min}_{\mathrm{FE} \in \mathcal{FE}} \mathsf{max}_x \mathsf{BWComp}(\mathrm{FE.Enc}, x) \ ,$$

where the minimization is over all $\mathrm{FE} \in \mathcal{FE}$ that support the circuit class $\mathcal{C}_{|(\ell,d,s)}$.

Goldwasser et al. [GKP$^+$13] provided a construction of a functional encryption scheme with the following properties.

Theorem 4 ([GKP$^+$13]). *Assuming the hardness of the LWE problem [Reg05] with sub-exponential modulus to noise ratio, there exists a FE family $\mathcal{FE}$, that contains a scheme $\mathrm{FE}_{\ell,d,s}$ for any (ℓ, d, s), with scaling characteristic*

$$\mathsf{SC}_{\mathcal{FE}}(\ell, d, s) = (\ell \cdot \mathsf{poly}(d)), \mathsf{polylog}(s), \mathsf{polylog}(s)) \ .$$

Furthermore, if the LWE problem above is sub-exponentially hard to solve, then the security of the FE *scheme holds against sub-exponential adversaries.*

The following corollary establishes the required parameters from a FE scheme in order to imply the existence of a succinct iO scheme. For a definition of one-way functions see, e.g., [Gol01].

Corollary 1 ([BV15]). *Assume the existence of one-way functions secure against sub-exponential adversaries. Then there exists a fixed polynomial p such that the following holds. If for every sufficiently large polynomial ℓ in the security parameter, there exist polynomials d, s such that there exists a subexponentially secure FE family with scaling characteristic*

$$\mathsf{SC}_{\mathcal{FE}}(\ell, d, s) \leq (\ell/2, d - p(\lambda), s/p(\lambda)) \tag{1}$$

then there also exists a succinct iO scheme.

Proof. To show the corollary, we quote the following result that was proven in [BV15]. They showed that iO follows from the existence of a sub-exponentially secure FE scheme that can evaluate the following function:

$$f_{\mathrm{FE}}(k,x,\beta,c) = \begin{cases} (\mathrm{FE.Enc}((k,x\|0);F_k(x\|0)),\mathrm{FE.Enc}((k,x\|1);F_k(x\|1))) & \text{if } \beta = 0 \\ c & \text{if } \beta = 1 \end{cases} \tag{2}$$

where F is a sub-exponentially secure pseudorandom function family which is implied by the existence of sub-exponentially secure one-way functions, and can be computed in complexity $\mathsf{poly}(\lambda)$ for some fixed polynomial. See [Gol01] for definition and the construction from [GGM84].

It therefore follows from the aforementioned [BV15] result that if we have a family $\mathcal{FE}$ is as in Eq. (1) then $\mathcal{FE}$ contains a scheme FE that can evaluate its own f_{FE}. This is observed by taking $p(\cdot)$ to be the circuit size of computing the pseudorandom function, and noticing that if the scaling characteristic of FE is bounded by $(\ell/2, d - p(\lambda), s/p(\lambda))$, as guaranteed in Eq. (1), then the bitwise complexity of f_{FE} is at most (ℓ, d, s) which allows for it to be evaluated by the scheme FE.

C.4 Shallow XiO Implies Bootstrappable FE

We recall the notion of *puncturable pseudorandom function* [BGI14]. In a nutshell, a puncturable pseudorandom function is an efficiently computable keyed function $\mathsf{PRF}_K(x)$, with an additional (efficient) puncturing algorithm which takes as input a key K and an input x^* and produces a punctured key K_{-x^*}. The punctured key K_{-x^*} allows to evaluate $\mathsf{PRF}_K(\cdot)$ on every input *except* x^*, i.e. it holds for all $x \neq x^*$ that $\mathsf{PRF}_K(x) = \mathsf{PRF}_{K_{-x^*}}(x)$. In terms of security, we require that $\mathsf{PRF}_K(x^*)$ is pseudorandom (i.e. computationally indistinguishable from a fresh uniformly random value) even given K_{-x^*}. We can now state our positive result, namely that shallow XiO and and FE scheme as provided by Theorem 4 yield iO.

Theorem 5. *Assume the sub-exponential hardness of the LWE assumption as per Theorem 4. Then the existence of* ShXiO *implies the existence of succinct* iO*.*

Proof. The construction is essentially the same as in [LPST16a], however relying on ShXiO instead of XiO. That is, we construct a weakly compact FE scheme from the succinct FE scheme given in Theorem 4 and ShXiO. We can then then apply Corollary 1 to obtain an iO scheme. Let PRF be a puncturable pseudorandom function and $\mathsf{sFE} = (\mathsf{sFE.Setup}, \mathsf{sFE.KeyGen}, \mathsf{sFE.Enc}, \mathsf{sFE.Dec})$ be a succinct functional encryption scheme as per Theorem 4. Further, let ShXiO be a shallow XiO scheme. Consider the following functional encryption scheme $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$.

- $\mathsf{Setup}(1^\lambda)$: This is identical to $\mathsf{sFE.Setup}(1^\lambda)$
- $\mathsf{KeyGen}(\mathsf{msk}, C)$: Let C' be a circuit which on input (m,i) outputs $C_i(m)$, i.e. the i-th bit of $C(m)$ (Note that C_i has the same depth as C). Compute and output $\mathsf{sk}_C \leftarrow \mathsf{sFE.KeyGen}(\mathsf{msk}, C')$.
- $\mathsf{Enc}(\mathsf{pk}, m)$:
  - Choose a uniformly random PRF key K
  - Let $G[pk, K, m]$ be a circuit which on input $(i, i') \in [\ell] \times [\ell']$ computes

    $$G[pk, K, m](i) = \mathsf{sFE.Enc}(\mathsf{pk}, (m,i); \mathsf{PRF}_K(i))[i'],$$

    i.e. it encrypts (m, i) under public key pk using randomness $\mathsf{PRF}_K(i)$, and outputs the i'-th bit of the ciphertext, where we let ℓ' denote the ciphertext length of the scheme.
  - Compute and output $\mathit{\Pi} \leftarrow \mathrm{ShXiO}(1^\lambda, G[\mathsf{pk}, K, m])$
- $\mathsf{Dec}(\mathsf{sk}_C, \mathit{\Pi})$: For all $i \in [\ell]$ output $y_i \leftarrow \mathsf{sFE.Dec}(\mathsf{sk}_C, \mathit{\Pi}(i,1)\| \dots \|\mathit{\Pi}(i,\ell'))$.

The security proof proceeds analogous to [LPST16a], and we only sketch the idea here. Fix two messages m_0 and m_1 for which it holds that $C(m_0) = C(m_1)$, and consider the hybrid circuits $G_j[pk, K_{-j}, m_0, m_1, c_j]$. G_j has the public key pk, a key K_{-j} punctured at input j, both messages m_0 and m_1 as well as a ciphertext $c_j = \mathsf{sFE.Enc}(\mathsf{pk}, (m_1, j); \mathsf{PRF}_K(j))$ hardwired. On input $(i, i') \in [\ell] \times [\ell']$, the circuit $G_j[pk, K_{-j}, m_0, m_1, c_j]$ does the following

- If $i < j$: Compute $c_i \leftarrow \mathsf{sFE.Enc}(\mathsf{pk}, (m_1, i); \mathsf{PRF}_{K_{-j}}(i))$ and output $c_i[i']$
- If $i > j$: Compute $c_i \leftarrow \mathsf{sFE.Enc}(\mathsf{pk}, (m_0, i); \mathsf{PRF}_{K_{-j}}(i))$ and output $c_i[i']$
- If $i = j$: Output $c_j[i']$ (hardwired)

First note that $G_0[pk, K_{-0}, m_0, m_1, c_0]$ is functionally equivalent to $G[pk, K, m_0]$, whereas the circuit $G_\ell[pk, K_{-\ell}, m_0, m_1, c_\ell]$ is functionally equivalent to $G[pk, K, m_1]$. We can now define a sequence of hybrid experiments $\mathcal{H}_0, \ldots, \mathcal{H}_\ell$, where in hybrid $\mathcal{H}_j$ the challenge ciphertext Π is computed by $\Pi \leftarrow \text{ShXiO}(1^\lambda, G_j[pk, K_{-j}, m_0, m_1, c_j])$. To see that for each $j \in [\ell]$ the hybrids $\mathcal{H}_{j-1}$ and $\mathcal{H}_j$ are indistinguishable, note the following:

- $G_{j-1}[pk, K_{-(j-1)}, m_0, m_1, c_{j-1}]$ and $G_j[pk, K_{-j}, m_0, m_1, c'_j]$, where c'_j is computed by $c'_j = \mathsf{sFE.Enc}(\mathsf{pk}, (m_0, j); \mathsf{PRF}_K(j))$, are functionally equivalent, thus XiO security of ShXiO allows us to argue that Π can be computed as $\Pi \leftarrow \text{ShXiO}(1^\lambda, G_j[pk, K_{-j}, m_0, m_1, c'_j])$, while this modification is not detectable by the adversary.
- Next, using puncturing security of PRF, we can compute c'_j by $c'_j \leftarrow \mathsf{sFE.Enc}(\mathsf{pk}, (m, i); r)$, where $r \leftarrow_\$ \{0,1\}^\lambda$ are fresh uniformly random coins.
- Using security of sFE, we can now argue that $c'_j \leftarrow \mathsf{sFE.Enc}(\mathsf{pk}, (m_0, i); r)$ can be replaced by $c_j \leftarrow \mathsf{sFE.Enc}(\mathsf{pk}, (m_1, i); r)$.
- Again using puncturing security, we can argue that the ciphertext c_j can be computed via $c_j \leftarrow \mathsf{sFE.Enc}(\mathsf{pk}, (m_1, i); \mathsf{PRF}_{K_{-j}}(j))$. This is hybrid $\mathcal{H}_j$.

Hence, we can argue that $\mathcal{H}_0$ and $\mathcal{H}_\ell$ are computationally indistinguishable under the said assumptions. This concludes the sketch of the security proof.

We will now turn to the compactness analysis of FE and determine the relations between the parameters. We will forgo tightness of parameters in order to reduce the number of different variables in the proof. Consider a polynomial $q(x) = x^a$ with the following properties:

1. Recalling Corollary 1 and in particular the polynomial p defined therein, let q be s.t. $q \geq p$ for every input.
2. The scaling characteristic of ShXiO is at most
$$\mathsf{SC}_{\text{ShXiO}}(\ell, d, s) = (\ell^{1-\epsilon} q(s), q(s), q(\ell \cdot s)).$$
3. For the sake of this analysis, it will be convenient to consider the ℓ-wise parallel composition of sFE, call it sFE'. By Theorem 4 the scaling characteristic of sFE' is at most
$$\mathsf{SC}_{\mathsf{sFE}'}(\ell, d, s) = (\ell \cdot q(d), \log(q(s)), \log(q(s))).$$

Let ℓ be some polynomial in λ such that $\ell^{\epsilon/a} \geq p^2(\lambda)$ and also $\ell^{0.1} > p^2(\lambda)$. We instantiate sFE' with parameters $(\ell_0, d_0, s_0) = (\ell, \ell^{\epsilon/a}, q(\ell^{1.1}))$, and consider its composition with ShXiO as in the scheme FE above. First let us compute the scaling characteristic of sFE':

$$\begin{aligned}\mathsf{SC}_{\mathsf{sFE}'}(\ell_0, d_0, s_0) = (\ell_0 q(d_0), \mathsf{polylog}(s_0), \mathsf{polylog}(s_0)) &= (\ell q(\ell^{\epsilon/a}), \mathsf{polylog}(\ell), \mathsf{polylog}(\ell)) \\ &= (\ell^{1+\epsilon}, \mathsf{polylog}(\ell), \mathsf{polylog}(\ell)) \ .\end{aligned}$$

We then apply $\mathsf{SC}_{\text{ShXiO}}$ to obtain the total scaling characteristic of the FE:

$$\mathsf{SC}_{\mathsf{FE}}(\ell_0, d_0, s_0) = (\ell^{(1+\epsilon)(1-\epsilon)}, q(\mathsf{polylog}(\ell)), q(\ell\mathsf{polylog}(\ell))) = (\ell^{1-\epsilon^2}, \mathsf{polylog}(\ell), q(\ell\mathsf{polylog}(\ell))) \ .$$

Asymptotically, for large enough λ, it holds that $\ell^{\epsilon^2}(\lambda) > 2$, $\ell^{\epsilon/a}(\lambda) \geq p(\lambda) + \mathsf{polylog}(\ell)$, $\ell^{0.1}(\lambda) > p(\lambda)\mathsf{polylog}(\ell)$ (for any polylogarithmic function). We therefore conclude that the overall scaling characteristic is at most $(\ell_0/2, d_0 - p(\lambda), s_0/p(\lambda))$, and thus indeed for all ℓ we have polynomials $d = d_0, s = s_0$ as defined above that satisfy the requirement to apply Corollary 1.