Candidate iO From Homomorphic Encryption Schemes

Zvika Brakerski*[1], Nico Döttling[2], Sanjam Garg†[3], and Giulio Malavolta[4]

[1]Weizmann Institute of Science
[2]CISPA Helmoltz Center for Information Security
[3]UC Berkeley
[4]UC Berkeley & Carnegie Mellon University

Abstract

We propose a new approach to construct general-purpose indistinguishability obfuscation (iO). Our construction is obtained via a new intermediate primitive that we call *split fully-homomorphic encryption* (split FHE), which we show to be sufficient for constructing iO. Specifically, split FHE is FHE where decryption takes the following two-step syntactic form: (i) A *secret* decryption step uses the secret key and produces a *hint* which is (asymptotically) shorter than the length of the encrypted message, and (ii) a *public* decryption step that only requires the ciphertext and the previously generated hint (and not the entire secret key), and recovers the encrypted message. In terms of security, the hints for a set of ciphertexts should not allow one to violate semantic security for any other ciphertexts.

Next, we show a *generic candidate* construction of split FHE based on three building blocks: (i) A standard FHE scheme with linear decrypt-and-multiply (which can be instantiated with essentially all LWE-based constructions), (ii) a linearly homomorphic encryption scheme with short decryption hints (such as the Damgård-Jurik encryption scheme, based on the DCR problem), and (iii) a cryptographic hash function (which can be based on a variety of standard assumptions). Our approach is *heuristic* in the sense that our construction is not provably secure and makes implicit assumptions about the interplay between these underlying primitives. We show evidence that this construction is secure by providing an argument in an appropriately defined oracle model.

We view our construction as a big departure from the state-of-the-art constructions, and it is in fact quite simple.

1 Introduction

The goal of program obfuscation is to transform an arbitrary circuit C into an unintelligible but functionally equivalent circuit $\tilde{C}$. The notion of program obfuscation was first studied by Hada [Had00] and Barak et al. [BGI+01]. However, these works showed that natural notions of obfuscation are impossible to realize for general functionalities. Specifically, Barak et al. [BGI+01] defined a very natural notion of security for program obfuscation called virtual black-box (VBB) security, which requires that an obfuscated program does not revel anything

*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, 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.

beyond what could be learned from just the input-output behavior of the original program. In the same work, they showed that this notion of program obfuscation is impossible to achieve for arbitrary circuits.

In light of this impossibility result, much of the work on obfuscation focused on realizing obfuscation for special functionalities. However, this changed with the work of Garg et al. [GGH$^+$13b] that proposed the first candidate indistinguishability obfuscation (iO) construction based on multilinear maps [GGH13a]. Furthermore, Garg et al. [GGH$^+$13b] showed powerful applications of iO to tasks such as functional encryption. Loosely speaking, iO requires that the obfuscations of two circuits C_0 and C_1 that have identical input output behavior are computationally indistinguishable. Subsequently, significant work on using program obfuscation (e.g., [SW14, GGHR14, BZ14]) has shown that most cryptographic applications of interest can be realized using iO (and one-way functions), or that iO is virtually *crypto-complete*.

Given its importance, significant effort has been poured into realizing secure obfuscation candidates. The first approach to obfuscation relied on using new candidate constructions of multilinear maps [GGH13a, CLT13, GGH15], an algebraic object that significantly expands the structure available for cryptographic construction. Unfortunately, all multilinear map constructions so far have relied on ad-hoc and new computational intractability assumptions. Furthermore, attacks [CHL$^+$15, HJ16] on the multilinear map candidates and attacks [MSZ16, CGH17] on several of the multilinear map based iO constructions [GGH$^+$13b, BR14, BGK$^+$14] were later found. In light of these attacks, follow up works (e.g., [GMM$^+$16]) offered constructions that defended against these attacks by giving constructions in the so-called weak multilinear map model [MSZ16]. Several of these weak multilinear map model based iO constructions are still conjectured to be secure, however, the *break-and-repair* cycle of their development has left cryptographers wary, and rightly so.

Around the time when attacks on multilinear map candidates were at an all time high, cryptographers started exploring new approaches to iO without using multiliear maps (or reducing their usage). Toward this goal, Bitansky and Vaikunthanathan [BV15] and Ananth and Jain [AJ15] showed that iO could be realized assuming just functional encryption. In another approach, instead of trying to remove multilinear maps completely, Lin [Lin16] and Lin and Vaikuntanathan [LV16] attempted to reduce their usage, i.e., they proposed iO constructions using only constant degree multilinear maps. With the goal of ultimately basing iO constructions on standard assumptions on bilinear maps, cryptographers started developing new ideas for realizing iO candidates from smaller constant degree multilinear maps [AS17, Lin17]. Recently, Lin and Tessaro [LT17a] described a candidate iO construction from degree-L multilinear maps for any $L \geq 2$ and additionally assuming PRGs with certain special locality properties. Unfortunately, it was shown the needed PRGs for the case of $L = 2$ are insecure (in fact it was proved that they cannot exist) [LV17, BBKK17]. Thus, still leaving a gap between bilinear maps and iO constructions which could now be based on trilinear maps [LT17b]. Very recent works [Agr19, JLMS19, AJL$^+$19] (and cryptanalysis [BHJ$^+$19]), develop new ideas to resolve these problems and realize constructions based on bilinear maps. However, these bilinear map based constructions, which are still conjectured to be secure, additionally rely on certain pseudorandom objects with novel security properties. Finally, we note that all the other (perhaps less popular) approaches to iO (e.g., [GJK18]) also start from new computational hardness assumptions.

Given the prior work, it is plausible that new sources of hardness are necessary for realizing iO candidates. Thus, this break-and-repair cycle would be necessary as we understand the underlying new assumptions better. In fact, there is some evidence that iO constructions based on simpler primitives [GMM17a, GMM17b] are hard to realize. Making progress on this dilemma is the focus of this work.

1.1 Our Results

We propose a new approach to construct general-purpose indistinguishability obfuscation. Our approach is *heuristic* but *without* using any new sources of computational hardness. In other words, our constructions use well-studied cryptographic primitives in a generic way to realize obfuscation, while still being heuristic in the sense that our constructions are not provably secure and make implicit assumptions about the interplay of the underlying primitives. The primitives we use can themselves be securely realized based on standard assumptions, namely the hardness of the learning with errors (LWE) and the decisional composite residues (DCR) problem. At a high level, our heuristics are similar in flavor to (i) the random oracle heuristic that is often used in cryptographic constructions [BR93] and (ii) the circular security heuristic that has been widely used in the construction of fully-homomorphic encryption schemes (FHE) [Gen09].

Split-FHE. The starting point of our work is the fact that iO can *provably* be based on *split FHE*, a new primitive that we introduce in this work. A split FHE is an FHE scheme that allows for certain special properties of the decryption algorithm. Specifically, we consider FHE schemes for which the decryption algorithm can be split into two subroutines:

- $\rho \leftarrow \mathsf{PDec}(\mathsf{sk}, c)$: A *private* procedure that takes the FHE secret key and a ciphertext as input and produces a decryption hint ρ, of size much smaller than the message encrypted in c.
- $m \leftarrow \mathsf{Rec}(\rho, c)$: A *public* procedure that takes as input the decryption hint ρ (generated by PDec) and the ciphertext c and recovers the full plaintext.

The security for a split FHE scheme requires that, for all pairs of messages (m_0, m_1) and all circuits C such that $C(m_0) = C(m_1)$, the encryption of m_0 is computationally indistinguishable from the encryption of m_1, even given the decryption hint for the ciphertext evaluated on C.

We show that split FHE alone suffices to construct exponentially-efficient iO [LPST16], which in turn allows us to build fully-fledged iO. Concretely, we prove the following theorem.

Theorem 1.1 (Informal) *Assuming sub-exponentially hard LWE and the existence of sub-exponentially secure split FHE, then there exists indistinguishability obfuscation for all circuits.*

A Generic Candidate. Next, we show a *generic candidate* construction of split FHE based on three building blocks: (i) a standard FHE scheme with linear decrypt-and-multiply (which can be instantiated with essentially all LWE-based constructions), (ii) a linearly homomorphic encryption scheme with short decryption hints (such as the Damgård-Jurik encryption scheme [DJ01], based on the DCR problem), and (iii) a cryptographic hash functions. The security of the scheme can be based on a new conjecture on the interplay of these primitives, which we view as a natural strengthening of circular security. In this sense, it is aligned with Gentry's heuristic step in the FHE bootstrapping theorem [Gen09]. Additionally, our use of the cryptographic hash function has similarities to the other heuristic uses of hash functions, e.g., in the Fiat-Shamir transformation [FS87].

We expect that there will exist instantiations of the underlying primitives (though contrived) for which this construction is insecure. For example, if the underlying schemes are not circular secure to begin with, then the resulting split FHE would also be insecure. However, for natural instantiations of these primitives, security can be conjectured.

Evidence of security. In order to build confidence in our construction, we show evidence that the above-mentioned conjecture on the interplay between the security holds in an appropriate oracle model, inspired by the random oracle model. Thus, pushing all the heuristic aspects of the construction to an oracle. In fact, we show that security can be proved in this oracle model.

An alternate way to think of this result is that we construct split FHE based on a obfuscation for a specific program (representing the oracle), for which we can offer a relatively simple and natural heuristic implementation.

Conceptual Simplicity. Another positive feature of our construction is its conceptual simplicity, which makes it much easier to analyze and thus have confidence in. Finally, we remark that our construction is a big departure from the previously-mentioned multilinear maps based and local PRG based iO constructions and will be accessible to readers without first understanding prior iO constructions.

1.2 Technical Overview

In the following we give an informal overview of the techniques we develop in this work and we refer the reader to the technical sections for more precise statements.

1.2.1 Chimeric FHE

Our starting point is the hybrid FHE scheme recently introduced by Brakerski et al. [BDGM19], which we recall in the following. The objective of their work is to build an FHE scheme with best possible rate (in an asymptotic sense) by leveraging the fact that most LWE-based FHE scheme admit an efficient *linear noisy decryption.* Specifically, given an FHE ciphertext c and an LWE secret key $(s_1, \ldots, s_n)$ one can rewrite the decryption operation as a linear function $L_c(\cdot)$ such that

$$L_c(s_1, \ldots, s_n) = \mathsf{ECC}(m) + e$$

where e is a B-bounded noise term and ECC is some encoding of the plaintext (in their scheme m is packed in the high-order bits so that it does not interfere with the noise term). The idea then is to encrypt the secret key $(s_1, \ldots, s_n)$ under a (high-rate) linearly homomorphic encryption (LHE) scheme, which allows one to compress evaluated FHE ciphertext by computing $L_c(\cdot)$ homomorphically.

One interesting property of this approach is that it is completely parametric in the choice of the schemes, as long as they satisfy some simple structural requirements: More concretely, one can use *any* LHE scheme as long as its plaintext domain matches the LWE modulus of the FHE scheme. As an example, one can set the LHE to be the Damgård-Jurik encryption scheme [DJ01, Pai99], which we briefly recall in the following. The public key of the scheme consists of a large composite $N = pq$ and an integer ζ, and the encryption algorithm a message m computes

$$c = r^{N^\zeta} \cdot (1 + N)^m \mod N^{\zeta+1}$$

for some uniform $r \leftarrow\$ \mathbb{Z}_N$. Note that the corresponding plaintext space is $\mathbb{Z}_{N^\zeta}$ and therefore the rate of the scheme approaches 1 as ζ grows. Furthermore, we observe that the scheme has one additional property that we refer to as *split decryption.* A scheme has split decryption if the decryption algorithm can be divided into a private and a public subroutine:

- The *private* procedure takes as input a ciphertext c and the secret key $\phi(N)$ and computes a *decryption hint*

$$\rho = c^{N^{-\zeta}} \mod N$$

using the extended Euclidean algorithm. It is crucial to observe that $\rho \in \mathbb{Z}_N$ is potentially much smaller than the plaintext m.

- The *public* procedure takes as input a ciphertext c and the decryption hint ρ and recovers the plaintext by computing

$$(1+N)^m = c/\rho^{N^\zeta} \mod N^{\zeta+1}$$

and decoding m in polynomial time using the binomial theorem.

In a nutshell, the subgroup homomorphism allows one to compute a compressed version of the randomness, which can be then publicly stretched and used to unmask the plaintext. This means that m can be fully recovered by communicating a small hint of size fixed and, in particular, independent of $|m|$. As we are going to discuss later, this property is going to be our main leverage to build general-purpose obfuscation.

Temporarily glossing over the security implications, we point out that the hybrid scheme of Brakerski et al. [BDGM19] already suffices to construct an FHE scheme with split decryption (in short, split FHE): Simply instantiate the LHE scheme with Damgård-Jurik and convert evaluated FHE ciphertexts before decryption using the algorithm described above.

1.2.2 Security for Split FHE

We now delve into the desired security property for a split FHE scheme. On a high level, we would like to ensure that the decryption hint does not reveal any additional information, beyond the plaintext of the corresponding ciphertext. It is instructive to observe that if we do not insist on this property, then every FHE scheme has a trivial split decryption procedure which simply outputs the secret key. We formalize this intuition as an indistinguishability definition that, roughly speaking, demands that for all plaintext pairs (m_0, m_1) and every set of circuits $(C_1, \dots, C_\beta)$ such that $C_i(m_0) = C_i(m_1)$, then the encryption of m_0 and m_1 are computationally indistinguishable, even given the decryption hints ρ_i of the evaluated ciphertexts. The condition $C_i(m_0) = C_i(m_1)$ rules out trivial attacks where the distinguisher just checks the output of the evaluation. Here $\beta = \beta(\lambda)$ is an arbitrary (but a priori bounded) polynomial in the security parameter.

Unfortunately, our candidate as described above falls short in satisfying this security notion: The central problem is that our split decryption procedure reveals the complete plaintext encoded in the Damgård-Jurik ciphertext. This means that the distinguisher learns arbitrarily many relations of the form

$$L_{c_i}(s_1, \dots, s_n) = \mathsf{ECC}(C_i(m_b)) + e_i$$

where c_i is the evaluated ciphertext and L_{c_i} is a publicly known linear function. Collecting a large enough sample allows the distinguisher to recompute the FHE secret key $(s_1, \dots, s_n)$ via, e.g., Gaussian elimination. A standard approach to obviate this problem is to smudge the noise e_i with some mask r_i uniformly sampled from an exponentially larger domain. Thus, a natural solution would be to compute a randomizing ciphertext $d_i = \mathsf{DJ.Enc}(\mathsf{pk}_{\mathsf{DJ}}, r_i)$ and output the decryption hint for

$$c_i \cdot d_i = \mathsf{DJ.Enc}(\mathsf{pk}_{\mathsf{DJ}}, \mathsf{ECC}(C_i(m_b)) + e_i + r_i) \approx \mathsf{DJ.Enc}(\mathsf{pk}_{\mathsf{DJ}}, \mathsf{ECC}(C_i(m_b)) + r_i)$$

where r_i is sampled from a domain exponentially larger than the noise bound B but small enough to allow one to decode $\mathsf{ECC}(C_i(m_b))$. While it is possible to show that this approach indeed satisfies the security notion outlined above, it introduces an overhead in the size of the hint, which now consists of the pair (ρ_i, d_i). Note that we cannot allow the distinguisher to recompute d_i locally as it is crucial that r_i remains hidden, so we have no other choice but append it to the decryption hint. However the decryption hint is now of size $O(|c_i|)$, which does not satisfy our compactness requirement and makes our efforts purposeless (one can just set the decryption hint to be $C_i(m_b)$ and achieve better efficiency).

Although we appear to have encountered a roadblock, a closer look reveals that we still gained something from this approach: The ciphertext d_i encodes a (somewhat small) random value and in particular is completely independent from c_i. Furthermore, the decryption hint of $c_i \cdot d_i$ can be computed using the secret key alone. Assume for the moment that we had access to an oracle Sample that outputs uniform Damgård-Jurik encryption of bounded random values, then our idea is to delegate the sampling of d_i to Sample. This allows us to bypass the main obstacle: We do not need to include d_i in the decryption hint as it can be recomputed by querying Sample. One can think of this approach as a more structured version of the Fiat-Shamir transform [FS87], which allows us to state the following theorem.

Theorem 1.2 (Informal) *Assuming the hardness of LWE and DCR, then there exists a split FHE scheme in the* Sample*-hybrid model.*

Looking ahead to our end goal, another interpretation of this theorem is as a universality result: Assuming the hardness of LWE and DCR, we can bootstrap an obfuscator for a specific circuit (i.e., the one that samples a uniform Damgård-Jurik encryption of a bounded random value) to an obfuscator for all circuits.

1.2.3 Instantiating the Oracle

The most compelling question which arises from our main theorem is whether there exist plausible instantiations for the oracle Sample. A first (flawed) attempt is to devise an oblivious sampling procedure for Damgård-Jurik ciphertext using a random oracle: Note that Damgård-Jurik ciphertexts live in a dense domain $\mathbb{Z}_{N^{\zeta+1}}$ and indeed sampling a random integer $c_i \leftarrow\!\$\, \mathbb{Z}_{N^{\zeta+1}}$ maps to a well-formed ciphertext with all but negligible probability. However, since c_i is uniform in the ciphertext domain, then so is the underlying plaintext $r_i \in \mathbb{Z}_{N^\zeta}$. This makes c_i unusable for our purposes since we require r_i to be bounded by some value $\tilde{q}$, which is exponentially smaller than N^ζ. If we were to sample r_i this way, then it would completely mask the term $\mathsf{ECC}(C_i(m_b))$, thus making the plaintext impossible to decode.

Ideally, we would like to restrict the oblivious sampling to ciphertexts encrypting $\tilde{q}$-bounded messages. Unfortunately, we are not aware of the existence of any such algorithm. Instead, our idea is to still sample c_i uniformly over the complete ciphertext domain and remove the high-order bits of r_i *homomorphically*: This can be done by including an FHE encryption of the Damgård-Jurik secret key, then homomorphically evaluating the circuit that decrypts c_i and computes $-\lfloor r_i/\tilde{q} \rfloor \cdot \tilde{q}$. The evaluated ciphertext is then converted again to the Damgård-Jurik domain using the linear noisy decryption of the FHE scheme. At this point, one can obtain a well-formed encryption of a $\tilde{q}$-bounded value by computing

$$\begin{aligned}\mathsf{DJ.Enc}(\mathsf{pk}_{\mathsf{DJ}}, -\lfloor r_i/\tilde{q} \rfloor \cdot \tilde{q} + e) \cdot c_i &= \mathsf{DJ.Enc}(\mathsf{pk}_{\mathsf{DJ}}, -\lfloor r_i/\tilde{q} \rfloor \cdot \tilde{q} + e + r_i) \\ &= \mathsf{DJ.Enc}(\mathsf{pk}_{\mathsf{DJ}}, (r_i \mod \tilde{q}) + e)\end{aligned}$$

where the term $(r_i \mod \tilde{q}) + e$ is $\tilde{q}$-bounded with all but negligible probability by setting $\tilde{q} \gg B$. While this approach brings us tantalizingly close to a provably secure scheme, a careful analysis highlights two lingering conjectures.

(1) *Circular Security:* Adding and FHE encryption of the Damgård-Jurik secret key introduces a circular dependency in the security of the two schemes (recall that our construction

already encodes a Damgård-Jurik encryption of the FHE secret key). While circular security falls outside of the realm of provable statements, it is widely accepted as a mild assumption and it is known to be achieved by most natural encryption schemes [BHHI10]. We stress that circular security is also inherent in the the bootstrapping theorem of Gentry [Gen09], the only known method to construct fully (as opposed to levelled) homomorphic encryption from LWE.

(2) *Correlations:* While the homomorphically evaluated circuit essentially ignores the low-order bits of r_i, the corresponding decryption noise e might still depend on $(r_i \mod \tilde{q})$ in some intricate way. This might introduce some correlation and bias the distribution of the term $(r_i \mod \tilde{q}) + e$ with respect to a uniform $u \leftarrow_{\$} \mathbb{Z}_{\tilde{q}}$. However, the noise function is typically highly non-linear and therefore appears to be difficult to exploit. We also point out that the distinguisher has no control over the choice of e, which exclusively depends on an honest execution of the homomorphic evaluation algorithm. We therefore conjecture that the distribution of $(r_i \mod \tilde{q}) + e$ is computationally indistinguishable from u.

In light of the above insights, we put forward the conjecture that the proposed algorithm already gives us a secure implementation of the oracle Sample. We view this as a natural strengthening of Gentry's heuristic for the bootstrapping theorem, which is justified by our more ambitious objective. As the conjecture pertains to standard cryptographic material (FHE and Damgård-Jurik encryption) we believe that any further insight on its veracity would substantially improve our understanding on these important and well-studied building blocks.

Finally, we mention that many heuristics can be used to weaken the correlation between the decryption noise e and the low-order bits $(r_i \mod \tilde{q})$, such as repeated applications of FHE bootstrapping [DS16]. We also propose a different heuristic approach to remove correlations based on binary extractors and we refer the reader to the technical sections for further details.

1.2.4 From Split FHE to iO

What is left to be shown is that split FHE does indeed suffice to construct program obfuscation. With this goal in mind, we recall a surprising result by Lin et al. [LPST16] which states that, under the assumption that the LWE problem is sub-exponentially hard, iO for all circuits is implied by an obfuscator for circuits with logarithmic-size inputs with non-trivial efficiency. Here non-trivial efficiency means that the size of the obfuscated circuit $\tilde{C}$ with input domain $\{0,1\}^\eta$ is at most $\mathsf{poly}(\lambda, |C|) \cdot 2^{\eta \cdot (1-\varepsilon)}$, for some constant $\epsilon > 0$. This means that it suffices to show that split FHE implies the existence of an obfuscator (for circuits with polynomial-size input domain) with non-trivial efficiency.

The transformation is deceptively simple (and similar to [BNPW16]): The obfuscator computes a split FHE encryption of the circuit C and partitions the input domains in $2^{\eta/2}$ disjoint sets $(P_1, \ldots, P_{2^{\eta/2}})$ of equal size. Then, for each partition P_i, the algorithm homomorphically evaluates the universal circuit that evaluates C on all inputs in P_i and returns the concatenation of all outputs. Finally it returns the decryption hint ρ_i corresponding to the evaluated ciphertext. The obfuscated circuit consists of the public-key of the split FHE scheme, the encryption of C, and all of the decryption hints $(\rho_1, \ldots, \rho_{2^{\eta/2}})$. Note that the obfuscated circuit can be evaluated efficiently: On input x, let P_x be the partition that contains x, then the evaluator recomputes the homomorphic evaluation (which is a deterministic operation) of C on P_x and recovers the output using the decryption hint ρ_x. As for non-trivial efficiency, since the size of each decryption hint is that of a fixed polynomial $\mathsf{poly}(\lambda)$, the total size of the obfuscated circuit is bounded by $\mathsf{poly}(\lambda, |C|) \cdot 2^{\eta/2}$, as desired.

1.2.5 Other Applications

To demonstrate that the scope of our split FHE scheme goes beyond program obfuscation, we outline two additional applications. In both cases we only rely on standard assumptions, i.e., we do not need to introduce any new conjecture.

Two-Party Computation with Pre-Processing. We obtain a (semi-honest) two-party computation scheme for any circuit $C : \{0,1\}^\ell \to \{0,1\}^k$ with an input- and circuit-independent pre-processing where the communication complexity of the pre-processing phase is $\mathsf{poly}(\lambda, k)$, whereas the communication complexity of the online phase is $\mathsf{poly}(\lambda) + \ell$. This improves over garbled circuit-based approaches that require a pre-processing at least linear in $|C|$. The protocol works as follows: In the pre-processing phase Alice and Bob exchange their (independently sampled) public-keys for a split FHE scheme and Alice computes a randomizing ciphertext (in the scheme defined above this corresponds to a Damgård-Jurik encryption of a bounded random value), which is sent to Bob. In the online phase, Alice and Bob exchange their inputs encrypted under their own public keys (to achieve best-possible rate this can be done using hybrid encryption) and homomorphically compute the multi-key evaluation of f over both inputs. Note that multi-key evaluation is generically possible for the case of two parties by nesting the two split FHE evaluations. Then Alice consumes the randomizing ciphertext computed in the pre-processing and sends a partial decryption of the evaluated ciphertext in the form of a decryption hint. Bob can then locally complete the partial decryption using its own secret key and recover the output.

Rate-1 Reusable Garbled Circuits. The work of Goldwasser et al. [GKP$^+$13] showed, assuming the hardness of the LWE problem, how to construct reusable garbled circuits where the size of the input encodings is $\mathsf{poly}(\lambda, d, \ell \cdot k)$, where $C : \{0,1\}^\ell \to \{0,1\}^k$ and d is the depth of C. Using split FHE, we obtain a scheme with rate-1 encodings, i.e., of size $\mathsf{poly}(\lambda, d, \ell) + k$. This is done by using their scheme to garble the circuit that computes C homomorphically over the input encrypted under a split FHE scheme an returns the decryption hint of the evaluated ciphertext. This effectively removes the dependency of the underlying reusable garbled circuit on the output size k. However, we also need to include in the input encoding the answer of the Sample oracle (a Damgård-Jurik ciphertext, for the scheme describe above), which reintroduces an additive overhead in k.

1.3 Related Work

In the following we discuss more in depth the relation of our approach when compared with recent candidate constructions of iO from lattices and bilinear maps [Agr19, JLMS19, AJL$^+$19]. Very informally, this line of works leverages weak pseudorandom generators (PRG) to mask the noise of the LWE decryption. However, the output domain of such a PRG is only polynomially large: This is because of the usage of bilinear groups, where the plaintext space is polynomially bounded (decryption requires one to solve a discrete logarithm). This is especially problematic because statistical/computational indistinguishability cannot hold in this regime of parameters. To circumvent this problem, all papers in this line of work assume a strict bound on the distinguisher's success probability (e.g., 0.99) and then rely on amplification techniques. This however requires one to construct a weak PRG where the advantage of any PPT distinguisher is non-negligible but at the same time bounded by < 0.99.

On the other hand, we rely on the Damgård-Jurik encryption scheme, where the message domain is exponential. This allows us to sample the smudging factor from a distribution that is exponentially larger than the noise bound, which is necessary in order to argue about statistical indistinguishability. Thus in our settings, conjecturing that the advantage of the distinguisher is negligible is, at least in principle, plausible.

2 Preliminaries

We denote by $\lambda \in \mathbb{N}$ the security parameter. We say that a function $\mathsf{negl}(\cdot)$ is negligible if it vanishes faster than any 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 machine running in time $\mathsf{poly}(\lambda)$. We abbreviate the set $\{1, \ldots, n\}$ as $[n]$. Matrices are denoted by $\mathbf{M}$ and vectors are denoted by $\mathbf{v}$. We recall the smudging lemma [AIK11, AJL$^+$12].

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

2.1 Linear Algebra

We will need the following fact from linear algebra over rings, which holds immediately for fields but is non-trivial for the case of rings.

Lemma 2 *Let q be an arbitrary integer modulus and n be an integer. Let $\mathbf{t} \leftarrow\!\$\, \mathbb{Z}_q^n$ be chosen uniformly at random. Now let $\mathbf{a} \leftarrow\!\$\, \mathbb{Z}_q^n$ be distributed uniformly random. Then it holds that*

$$\Pr_{\mathbf{t}}[\langle \mathbf{a}, \mathbf{t} \rangle \textit{ distributed uniformly in } \mathbb{Z}_q] > 1 - \log(q) \cdot 2^{-n}.$$

In other words, except with probability $\log(q) \cdot 2^{-n}$ over the choice of $\mathbf{t}$ the inner product $\langle \mathbf{a}, \mathbf{t} \rangle$ will be distributed uniformly random given that $\mathbf{a}$ is uniform.

We provide the proof of Lemma 2 for completeness. The proof assumes some basic notions of algebra which we omit introducing here.

Proof: For a fixed $\mathbf{t}$, $\langle \mathbf{a}, \mathbf{t} \rangle$ will be uniform for a uniform $\mathbf{a} \leftarrow\!\$\, \mathbb{Z}_q^n$ given that the linear form $\Phi_{\mathbf{t}} : \mathbb{Z}_1^n \to \mathbb{Z}_q$ given by $\mathbf{x} \mapsto \langle \mathbf{t}, \mathbf{x} \rangle$ has range $\mathbb{Z}_q$. To see this, note that by linearity every $y \in \mathbb{Z}_q$ has the same number of preimages under $\Phi_{\mathbf{t}}$. Thus, $\Phi_{\mathbf{t}}$ maps a uniform distribution to a uniform distribution.

We will thus establish that the linear form $\Phi_{\mathbf{t}} : \mathbb{Z}_1^n \to \mathbb{Z}_q$ given by $\mathbf{x} \mapsto \langle \mathbf{t}, \mathbf{x} \rangle$ has range $\mathbb{Z}_q$, except with negligible probability over the choice of $\mathbf{t} \leftarrow\!\$\, \mathbb{Z}_q^n$. This function is not full range, if and only if there exists a non-trivial ideal $J \subseteq \mathbb{Z}_q$ such that for all i we have $\mathbf{t}_i \in J$. To see this, note that if all $\mathbf{t}_i \in J$, then $\langle \mathbf{a}, \mathbf{t} \rangle \in J$ and therefore $\Phi_{\mathbf{t}}$ is not full range. On the other hand, if no such J exists, then we can construct an $\mathbf{a}^* \in \mathbb{Z}_q^n$ via Chinese Remaindering such that $\langle \mathbf{a}^*, \mathbf{t} \rangle = 1$ and therefore $\Phi_{\mathbf{t}}$ is full range.

Thus, it suffices to show the above property for all maximal ideals in $\mathbb{Z}_q$, which are the $p_s\mathbb{Z}_q$, where the p_s are the prime-factors of q. As $p_s \geq 2$ we can upper-bound the number of maximal ideals in $\mathbb{Z}_q$ by $\log(q)$. Fix a a maximal ideal $J = p\mathbb{Z}_q$ and let $\mathbf{t} = (t_1, \ldots, t_n) \leftarrow\!\$\, \mathbb{Z}_q^n$ be chosen uniformly at random. It holds for a single uniformly random $t \leftarrow\!\$\, \mathbb{Z}_q$ that $\Pr[t \in J] = 1/p \leq 1/2$. Since the $(t_1, \ldots, t_n)$ are independent, it holds that $\Pr[\forall i : \mathbf{t}_i \in J] \leq 2^{-n}$. Finally, as there are at most $\log(q)$ maximal ideals J a union-bound yields $\Pr[\exists J \forall i : \mathbf{t}_i \in J] \leq \log(q) \cdot 2^{-n}$. We conclude that $\Phi_{\mathbf{t}}$ has range $\mathbb{Z}_q$, except with probability $\log(q) \cdot 2^{-n}$ over the choice of $\mathbf{t}$. □

2.2 Indistinguishability Obfuscation

We recall the notion of indistinguishability obfuscation (iO) from [GGH$^+$13b].

Definition 2.1 (Indistinguishability Obfuscation) *A PPT machine* iO *is an indistinguishability obfuscator for a circuit class* $\{\mathcal{C}_\lambda\}_{\lambda\in\mathbb{N}}$ *if the following conditions are satisfied:*
(Functionality) For all $\lambda \in \mathbb{N}$*, all circuit* $C \in \mathcal{C}_\lambda$*, all inputs* x *it holds that*

$$\Pr\left[\tilde{C}(x) = C(x) \middle| \tilde{C} \leftarrow \mathsf{iO}(C)\right] = 1.$$

(Indistinguishability) For all polynomial-size distinguishers $\mathcal{D}$ *there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all* $\lambda \in \mathbb{N}$*, all pairs of circuit* $(C_0, C_1) \in \mathcal{C}_\lambda$ *such that* $|C_0| = |C_1|$ *and* $C_0(x) = C_1(x)$ *on all inputs* x*, it holds that*

$$\left|\Pr\left[1 = \mathcal{D}(\mathsf{iO}(C_0))\right] - \Pr\left[1 = \mathcal{D}(\mathsf{iO}(C_1))\right]\right| = \mathsf{negl}(\lambda).$$

2.3 Learning with Errors

We recall the (decisional) learning with errors (LWE) problem as introduced by Regev [Reg05].

Definition 2.2 (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 for all polynomial-size distinguishers* $\mathcal{D}$ *there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all* $\lambda \in \mathbb{N}$ *it holds that*

$$\left|\Pr\left[1 = \mathcal{D}(\mathbf{A}, \mathbf{s}^\top \cdot \mathbf{A} + \mathbf{e})\right] - \Pr\left[1 = \mathcal{D}(\mathbf{A}, \mathbf{u})\right]\right| = \mathsf{negl}(\lambda).$$

where $\mathbf{A}$ *is chosen uniformly from* $\mathbb{Z}_q^{n\times m}$*,* $\mathbf{s}$ *is chosen uniformly from* $\mathbb{Z}_q^n$*,* $\mathbf{u}$ *is chosen uniformly from* $\mathbb{Z}_q^m$ *and* $\mathbf{e}$ *is chosen from* χ^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 omit the actual distribution χ but only use the fact that it can be bounded by a (small) value B.

2.4 Decisional Composite Residuosity

In the following we recall the decisional composite residuosity (DCR) assumption over $\mathbb{Z}^*_{N^{\zeta+1}}$ [Pai99, DJ01]. Let $N = pq$, where p and q are primes, be a uniformly sampled Blum integer and let ζ be a fixed non-negative integer. Observe that the multiplicative group $\mathbb{Z}^*_{N^{\zeta+1}}$ can be rewritten as the product of the subgroup $\mathbb{H}_N = \{(1+N)^i : i \in [N^\zeta]\}$, generated by $(1+N)$, and the group of N^ζ-th residues $\mathbb{NR}_N = \{x^{N^\zeta} : x \in \mathbb{Z}^*_N\}$ of order $\varphi(N)$, where $\varphi(\cdot)$ denotes Euler's totient function.

Definition 2.3 (Decisional Composite Residuosity) *Let $N = pq$, where p and q are primes, be a uniformly sampled Blum integer and let ζ be a fixed non-negative integer. The DCR problem is hard if for all polynomial-size distinguishers $\mathcal{D}$ there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all $\lambda \in \mathbb{N}$ it holds that*

$$|\Pr[1 = \mathcal{D}(r)] - \Pr[1 = \mathcal{D}(u)]| = \mathsf{negl}(\lambda).$$

where $r \leftarrow\$ \mathbb{NR}_N$ and $u \leftarrow\$ \mathbb{Z}^*_{N^{\zeta+1}}$.

3 Homomorphic Encryption

We recall the definition of homomorphic encryption in the following.

Definition 3.1 (Homomorphic Encryption) *A homomorphic encryption scheme consists of the following efficient algorithms.*

$\mathsf{KeyGen}(1^\lambda)$: *On input the security parameter 1^λ, the key generation algorithm returns a key pair* $(\mathsf{sk}, \mathsf{pk})$.

$\mathsf{Enc}(\mathsf{pk}, m)$: *On input a public key* pk *and a message m, the encryption algorithm returns a ciphertext c.*

$\mathsf{Eval}(\mathsf{pk}, C, (c_1, \ldots, c_\ell))$: *On input the public key* pk*, an ℓ-inputs circuit C, and a vector of ciphertexts $(c_1, \ldots, c_\ell)$, the evaluation algorithm returns an evaluated ciphertext c.*

$\mathsf{Dec}(\mathsf{sk}, c)$: *On input the secret key* sk *and a ciphertext c, the decryption algorithm returns a message m.*

We say that a scheme is fully-homomorphic (FHE) if it is homomorphic for all (unbounded) polynomial-size circuits. If the maximum size of the circuit that can be evaluated is bounded in the public parameters, then we call such a scheme a levelled FHE. We also consider a restricted class of homomorphism that supports linear functions and we refer to such a scheme as linearly-homomorphic encryption (LHE). We characterize correctness of a single evaluation, which suffices for our purposes. This can be extended to the more general notion of multi-hop correctness [GHV10] if the condition specified below is required to hold for arbitrary compositions of circuits.

Definition 3.2 (Correctness) *A homomorphic encryption scheme* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ *is correct if for all $\lambda \in \mathbb{N}$, all ℓ-inputs circuits C, all inputs $(m_1, \ldots, m_\ell)$, all* $(\mathsf{sk}, \mathsf{pk})$ *in the support of* $\mathsf{KeyGen}(1^\lambda)$*, and all c_i in the support of* $\mathsf{Enc}(\mathsf{pk}, m_i)$ *it holds that*

$$\Pr[\mathsf{Dec}(\mathsf{sk}, \mathsf{Eval}(\mathsf{pk}, C, (c_1, \ldots, c_\ell))) = C(m_1, \ldots, m_\ell)] = 1.$$

We require a scheme to be compact in the sense that the size of the ciphertext should not grow with the size of the evaluated circuit.

Definition 3.3 (Compactness) *A homomorphic encryption scheme* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ *is compact if there exists a polynomial* $\mathsf{poly}(\cdot)$ *such that for all $\lambda \in \mathbb{N}$, all ℓ-inputs circuits C in the supported family, all inputs $(m_1, \ldots, m_\ell)$, all* $(\mathsf{sk}, \mathsf{pk})$ *in the support of* $\mathsf{KeyGen}(1^\lambda)$*, and all c_i in the support of* $\mathsf{Enc}(\mathsf{pk}, m_i)$ *it holds that*

$$|\mathsf{Eval}(\mathsf{pk}, C, (c_1, \ldots, c_\ell))| = \mathsf{poly}(\lambda) \cdot |C(m_1, \ldots, m_\ell)|.$$

We define a weak notion of security (implied by the standard semantic security [GM82]) which is going to be more convenient to work with.

Definition 3.4 (Semantic Security) *A homomorphic encryption scheme* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ *is semantically secure if for all polynomial-size distinguishers* $\mathcal{D}$ *there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all* $\lambda \in \mathbb{N}$*, all pairs of message* (m_0, m_1)*, it holds that*

$$|\Pr[1 = \mathcal{D}(\mathsf{pk}, \mathsf{Enc}(\mathsf{pk}, m_0))] - \Pr[1 = \mathcal{D}(\mathsf{pk}, \mathsf{Enc}(\mathsf{pk}, m_1))]| = \mathsf{negl}(\lambda)$$

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

3.1 Linear Decrypt-and-Multiply

We consider schemes with a fine-grained correctness property. Specifically, we require that the decryption consists of the application of a linear function in the secret key, followed by some publicly computable function. Furthermore, we require that such a procedure allows us to specify an arbitrary constant ω that is multiplied to the resulting plaintext. We refer to such schemes as linear decrypt-and-multiply schemes. This property was introduced in an oral presentation by Micciancio [Mic19] and recently formalized by Brakerski et al. [BDGM19]. We stress that all major candidate FHE constructions satisfy (or can be adapted to) such a constraint, e.g., [GSW13, AP14, BV14]. We recall the definition in the following.

Definition 3.5 (Decrypt-and-Multiply) *We call a homomorphic encryption scheme* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ *a decrypt-and-multiply scheme, if there exists bounds* $B = B(\lambda)$ *and* $Q = Q(\lambda)$ *and an algorithm* $\mathsf{Dec\&Mult}$ *such that the following holds. For every* $q \geq Q$*, all* $(\mathsf{sk}, \mathsf{pk})$ *in the support of* $\mathsf{KeyGen}(1^\lambda, q)$*, every* ℓ*-inputs circuit* C*, all inputs* $(m_1, \dots, m_\ell)$*, all* c_i *in the support of* $\mathsf{Enc}(\mathsf{pk}, m_i)$ *and every* $\omega \in \mathbb{Z}_q$ *that*

$$\mathsf{Dec\&Mult}(\mathsf{sk}, \mathsf{Eval}(\mathsf{pk}, C, (c_1, \dots, c_\ell)), \omega) = \omega \cdot C(m_1, \dots, m_\ell) + e \mod q$$

where $\mathsf{Dec\&Mult}$ *is a linear function in* sk *over* $\mathbb{Z}_q$ *and* $|e| \leq B$ *with all but negligible probability.*

In our construction, we will need some additional structure for the modulus q. Fortunately, most LWE-based FHE schemes can be instantiated with an arbitrary q that does not depend on any secret input but only on the security parameter. Moreover, LWE-based FHE schemes can be instantiated with any (sufficiently large) modulus q without affecting the worst-case hardness of the underlying LWE problem [PRS17]. In an abuse of notation, we often write $\mathsf{KeyGen}(1^\lambda; q)$ to fix the modulus q in the key generation algorithm. In favor of a simpler analysis, we assume that e is always non-negative. Note that this is without loss of generality as it can be always guaranteed by adding B to the result of $\mathsf{Dec\&Mult}$ and setting a slightly looser bound $B = 2B$.

3.2 Split Decryption

We define the notion of homomorphic encryption with split decryption, which is going to be central in our work. Loosely speaking, a scheme has split decryption if the decryption algorithm consists of two subroutines: A private algorithm (that depends on the secret key) that on input a ciphertext c computes a *small* hint ρ, and a publicly computable algorithm that takes as input ρ and c and returns the corresponding plaintext. We henceforth refer to such schemes as *split* homomorphic encryption. We introduce the syntax in the following.

Definition 3.6 (Split Decryption) *A homomorphic encryption scheme* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ *has split decryption if the decryption algorithm* Dec *consist of the following two subroutines.*

$\mathsf{PDec}(\mathsf{sk}, c)$: *On input the secret key* sk *and a ciphertext* c*, the partial decryption algorithm returns a decryption hint* ρ*.*

$\mathsf{Rec}(\rho, c)$: *On input the hint* ρ *and a ciphertext* c*, the recovery algorithm returns a message* m*.*

The notion of correctness is extended canonically.

Definition 3.7 (Split Correctness) *A homomorphic encryption scheme with split decryption* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{PDec}, \mathsf{Rec})$ *is correct if for all* $\lambda \in \mathbb{N}$*, all* ℓ*-inputs circuits* C *in the supported family, all inputs* $(m_1, \ldots, m_\ell)$*, all* $(\mathsf{sk}, \mathsf{pk})$ *in the support of* $\mathsf{KeyGen}(1^\lambda)$*, and all* c_i *in the support of* $\mathsf{Enc}(\mathsf{pk}, m_i)$ *it holds that*

$$\Pr\left[\mathsf{Rec}(\mathsf{PDec}(\mathsf{sk}, c), c) = C(m_1, \ldots, m_\ell)\right] = 1$$

where $c = \mathsf{Eval}(\mathsf{pk}, C, (c_1, \ldots, c_\ell))$*.*

Beyond the standard compactness for homomorphic encryption, a scheme with split decryption must satisfy the additional property that the size of the decryption hint ρ is independent (or, more generally, sublinear) of the size of the message. Furthermore, the size of the public key and of a fresh encryption of a message m should depend polynomially in the security parameter and otherwise be linear in the size of the output. These are the properties that make split decryption non-trivial and that are going to be our main leverage to bootstrap this primitive into more powerful machinery. We formally characterize these requirements below.

Definition 3.8 (Split Compactness) *A homomorphic encryption scheme with split decryption* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{PDec}, \mathsf{Rec})$ *is compact if there exists a polynomial* $\mathsf{poly}(\cdot)$ *and such that for all* $\lambda \in \mathbb{N}$*, all* ℓ*-inputs circuits* C *in the supported family, all inputs* $(m_1, \ldots, m_\ell)$*, all* $(\mathsf{sk}, \mathsf{pk})$ *in the support of* $\mathsf{KeyGen}(1^\lambda)$*, and all* c_i *in the support of* $\mathsf{Enc}(\mathsf{pk}, m_i)$ *it holds that*

- $|\mathsf{pk}| \leq \mathsf{poly}(\lambda) \cdot |C(m_1, \ldots, m_\ell)|$,
- $|c_i| \leq \mathsf{poly}(\lambda, |m_i|) \cdot |C(m_1, \ldots, m_\ell)|$, *and*
- $|\rho| \leq \mathsf{poly}(\lambda)$

where $\rho = \mathsf{PDec}(\mathsf{sk}, \mathsf{Eval}(\mathsf{pk}, C, (c_1, \ldots, c_\ell)))$*.*

Finally the notion of semantic security for split schemes requires that the decryption hint ρ for a certain ciphertext does not reveal any information beyond the corresponding plaintext. Note that we define a very weak notion where the above must hold only for a bounded number of ciphertexts, and the inputs are fixed prior to the public parameters of the scheme.

Definition 3.9 (Split Security) *A homomorphic encryption scheme with split decryption* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{PDec}, \mathsf{Rec})$ *is secure if for all polynomial-size distinguishers* $\mathcal{D}$ *there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all* $\lambda \in \mathbb{N}$*, all polynomials* $\beta = \beta(\lambda)$*, all pairs of messages* (m_0, m_1)*, all vectors of circuits* $(C_1, \ldots, C_\beta)$ *such that, for all* $i \in [\beta]$*,* $C_i(m_0) = C_i(m_1)$ *it holds that*

$$\left|\Pr\left[1 = \mathcal{D}(\mathsf{pk}, c_0, \rho_{(1,0)}, \ldots, \rho_{(\beta,0)})\right] - \Pr\left[1 = \mathcal{D}(\mathsf{pk}, c_1, \rho_{(1,1)}, \ldots, \rho_{(\beta,1)})\right]\right| = \mathsf{negl}(\lambda)$$

where $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$*, for all* $b \in \{0, 1\}$ *define* $c_b \leftarrow \mathsf{Enc}(\mathsf{pk}, m_b)$ *and, for all* $i \in [\beta]$ *and all* $b \in \{0, 1\}$*, define* $\rho_{(i,b)} \leftarrow \mathsf{PDec}(\mathsf{sk}, \mathsf{Eval}(\mathsf{pk}, C_i, c_b))$*.*

We also present the stronger definition of simulation security for decryption hints, which requires the existence of a simulator that can compute the decryption hint without the knowledge of the secret key. This notion is not achievable in the standard model for split FHE scheme with short hints, but it may be achievable in the presence of a (programmable) oracle. We show that it is nevertheless useful.

Definition 3.10 (Simulatable Hints) *A homomorphic encryption scheme with split decryption* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{PDec}, \mathsf{Rec})$ *has simulatable hints if there exists a polynomial-size simulator* Sim *such that for all* $\lambda \in \mathbb{N}$*, all* $(\mathsf{sk}, \mathsf{pk})$ *in the support of* $\mathsf{KeyGen}(1^\lambda)$*, all messages* m*, and all ciphertexts* c *in the support of* $\mathsf{Enc}(\mathsf{pk}, m)$ *it holds that*

$$\mathsf{PDec}(\mathsf{sk}, c) \equiv \mathsf{Sim}(1^\lambda, \mathsf{pk}, c, m).$$

3.3 Damgård-Jurik Encryption

In the following we recall a variant of the Damgård-Jurik encryption linearly homomorphic encryption scheme [DJ01]. We present a variant of the scheme that satisfies the notion of split correctness, which is going to be instrumental for our purposes. The scheme is parametrized by a non-negative integer ζ that we assume is given as input to all algorithms.

$\mathsf{DJ.KeyGen}(1^\lambda)$: On input the security parameter 1^λ, sample a uniform Blum integer $N = pq$, where p and q are λ-bits primes. Set $\mathsf{pk} = (N, \zeta)$ and $\mathsf{sk} = \varphi(N)$.

$\mathsf{DJ.Enc}(\mathsf{pk}, m)$: On input a message $m \in \mathbb{Z}_{N^\zeta}$, sample a random $r \leftarrow\$ \mathbb{Z}_N$ and compute

$$c = r^{N^\zeta} \cdot (1+N)^m \mod N^{\zeta+1}.$$

$\mathsf{DJ.Eval}(\mathsf{pk}, f, (c_1, \dots, c_\ell))$: On input a vector of ciphertexts $(c_1, \dots, c_\ell)$ and a linear function $f = (\alpha_1, \dots, \alpha_\ell) \in \mathbb{Z}^\ell_{N^\zeta}$, compute

$$c = \prod_{i=1}^{\ell} c_i^{\alpha_1} \mod N^{\zeta+1}.$$

$\mathsf{DJ.PDec}(\mathsf{sk}, c)$: On input a ciphertext c, set $s = c \mod N$. Then compute $N^{-\zeta}$ such that $N^\zeta \cdot N^{-\zeta} = 1 \mod \varphi(N)$ using the extended Euclidean algorithm. Return

$$\rho = s^{N^{-\zeta}} \mod N.$$

$\mathsf{DJ.Rec}(\rho, c)$: On input a hint ρ and a ciphertext c, compute

$$(1+N)^m = c/\rho^{N^\zeta} \mod N^{\zeta+1}$$

and recover m using the polynomial-time algorithm described in [DJ01].

It is well known that the scheme satisfies (standard) semantic security assuming the intractability of the decisional composite residuosity (DCR) problem. To prove correctness, we are going to use the fact that

$$x^{N^\zeta} \mod N^{\zeta+1} = (x \mod N)^{N^\zeta} \mod N^{\zeta+1} \tag{1}$$

for all non-negative integers (x, ζ). We refer the reader to [MT19] for a proof of this equality. Recall that $c = r^{N^\zeta} \cdot (1+N)^m$ and that

$$\begin{aligned}
\rho &= (c \mod N)^{N^{-\zeta}} \mod N \\
&= \left(r^{N^\zeta} \cdot (1+N)^m \mod N\right)^{N^{-\zeta}} \mod N \\
&= \left(r^{N^\zeta} \mod N\right)^{N^{-\zeta}} \mod N.
\end{aligned}$$

Therefore we have that

$$\begin{aligned}
\rho^{N^\zeta} \mod N^{\zeta+1} &= \left(\left(r^{N^\zeta} \mod N\right)^{N^{-\zeta}} \mod N\right)^{N^\zeta} \mod N^{\zeta+1} \\
&= \left(r^{N^\zeta} \mod N\right)^{N^{-\zeta} \cdot N^\zeta} \mod N^{\zeta+1} \\
&= r^{N^\zeta} \mod N^{\zeta+1}
\end{aligned}$$

by an application of Equation (1). Taking the inverse on both sides of the equation above we obtain

$$\begin{aligned}
c/\rho^{N^\zeta} \mod N^{\zeta+1} &= c/r^{N^\zeta} \mod N^{\zeta+1} \\
&= r^{N^\zeta} \cdot (1+N)^m / r^{N^\zeta} \mod N^{\zeta+1} \\
&= (1+N)^m \mod N^{\zeta+1}
\end{aligned}$$

as desired for correctness. Although such a scheme does not immediately give us a *secure* split LHE, we highlight a few salient properties that we are going to leverage in our main constructions.

Split Compactness: The hint $\rho \in \mathbb{Z}_N$ consists of $\lceil \log(N) \rceil$ bits and in particular is independent of the size of the message space $\mathbb{Z}_{N^\zeta}$, as the integer ζ can be set to be arbitrarily large (within the range of polynomials in λ).

Simulatable Hints: Given a ciphertext c and a plaintext value m, the simulator can efficiently compute a ciphertext $\tilde{c}$ such that the homomorphic sum of c and $\tilde{c}$ results in a uniform encryption of m and the corresponding decryption hint can be computed given only the random coins used to generate $\tilde{c}$. Concretely, let

$$\tilde{c} = \frac{r^{N^\zeta} \cdot (1+N)^m}{c} \mod N^{\zeta+1}$$

for some $r \leftarrow_{\$} \mathbb{Z}_N$, then $\rho = r$.

Dense Ciphertexts: Sampling a random integer in $\mathbb{Z}_{N^{\zeta+1}}$ gives a well-formed ciphertext with all but negligible probability. This is because the group order $\varphi(N) \cdot N^\zeta$ is close to $N^{\zeta+1}$, i.e., $\frac{\varphi(N) \cdot N^\zeta}{N^{\zeta+1}} = \frac{\varphi(N)}{N} = 1 - \mathsf{negl}(\lambda)$.

4 Split Fully-Homomorphic Encryption

In the following we present our construction for FHE with split decryption. We first present a generic construction in the presence of a (a structured version of) a random oracle, then we propose concrete instantiations for the building blocks and plausible candidates for the implementation of the oracle.

4.1 Generic Construction

Our scheme assumes the existence of the following primitives:

- A fully-homomorphic encryption scheme $\mathsf{FHE} = (\mathsf{FHE.KeyGen}, \mathsf{FHE.Enc}, \mathsf{FHE.Eval}, \mathsf{FHE.Dec})$ with linear decrypt-and-multiply and with noise bound B.
- A linearly homomorphic encryption $\mathsf{LHE} = (\mathsf{LHE.KeyGen}, \mathsf{LHE.Enc}, \mathsf{LHE.Eval}, \mathsf{LHE.PDec}, \mathsf{LHE.Rec})$ with simulatable decryption hints.

If the underlying FHE scheme is leveled then so is going to be the resulting split FHE. Conversely, if the FHE scheme supports the evaluation of unbounded circuits, then so does the resulting split FHE construction. The scheme is described below, assuming that all algorithms have oracle access to the Sample interface.

$\mathsf{KeyGen}(1^\lambda)$: On input the security parameter 1^λ, sample a key pair $(\mathsf{sk}_{\mathsf{LHE}}, \mathsf{pk}_{\mathsf{LHE}}) \leftarrow \mathsf{LHE.KeyGen}(1^\lambda)$. Let $\mathbb{Z}_q$ be the plaintext space defined by LHE, then sample $(\mathsf{sk}_{\mathsf{FHE}}, \mathsf{pk}_{\mathsf{FHE}}) \leftarrow \mathsf{FHE.KeyGen}(1^\lambda; q)$. Let $\mathsf{sk}_{\mathsf{FHE}} = (s_1, \ldots, s_n) \in \mathbb{Z}_q^n$, then return

$$\mathsf{sk} = \mathsf{sk}_{\mathsf{LHE}} \quad \text{and} \quad \mathsf{pk} = \left(\mathsf{pk}_{\mathsf{FHE}}, \mathsf{pk}_{\mathsf{LHE}}, c_{(\mathsf{LHE},1)}, \ldots, c_{(\mathsf{LHE},n)}\right)$$

where, for all $i \in [n]$, we define $c_{(\mathsf{LHE},i)} \leftarrow \mathsf{LHE.Enc}(\mathsf{pk}_{\mathsf{LHE}}, s_i)$.

$\mathsf{Enc}(\mathsf{pk}, m)$: On input a message m return

$$c \leftarrow \mathsf{FHE.Enc}(\mathsf{pk}_{\mathsf{FHE}}, m).$$

$\mathsf{Eval}^{\mathsf{Sample}}(\mathsf{pk}, C, (c_1, \ldots, c_\ell))$: On input a circuit C with ℓ bits of input and k bits of output and a vector of ciphertexts $(c_1, \ldots, c_\ell)$, let, for all $j \in [k]$, C_j be the circuit that returns the j-th bit of the output of C, then compute

$$d_j \leftarrow \mathsf{FHE.Eval}(\mathsf{pk}_{\mathsf{FHE}}, C_j, (c_1, \ldots, c_\ell)).$$

Define the following linear function over $\mathbb{Z}_q$:

$$g(x_1, \ldots, x_n) = \sum_{j=1}^{k} \mathsf{Dec\&Mult}\left((x_1, \ldots, x_n), d_j, 2^{\lceil \log(\tilde{q}+(k+1)B)\rceil + j}\right).$$

Compute $d \leftarrow \mathsf{LHE.Eval}(\mathsf{pk}_{\mathsf{LHE}}, g, (c_{(\mathsf{LHE},1)}, \ldots, c_{(\mathsf{LHE},n)}))$, query $a \leftarrow \mathsf{Sample}(\mathsf{pk}, d)$ and return

$$c \leftarrow \mathsf{LHE.Eval}\left(\mathsf{pk}_{\mathsf{LHE}}, \sum, (d, a)\right)$$

where $\sum$ denotes the homomorphic summation.

$\mathsf{PDec}(\mathsf{sk}, c)$: On input an evaluated ciphertext c return

$$\rho \leftarrow \mathsf{LHE.PDec}(\mathsf{sk}_{\mathsf{LHE}}, c).$$

$\mathsf{Rec}(\rho, c)$: On input an evaluated ciphertext c, compute

$$\tilde{m} \leftarrow \mathsf{LHE.Rec}(\rho, c)$$

and return the binary representation of $\tilde{m}$ without its $\lceil \log(\tilde{q} + (k+1)B)\rceil$ least significant bits.

What is left to be shown is the exact specification of the oracle $\mathsf{Sample}(\mathsf{pk}, x)$, which we describe in the following. The oracle is deterministic and it is accessible by all parties, thus on input the same x, the oracle will always return the same output. The oracle is parametrized by an integer $\tilde{q}$, which we are going to define later.

$\mathsf{Sample}(\mathsf{pk}, x)$: On input a string $x \in \{0,1\}^*$ return a uniformly distributed ciphertext

$$\mathsf{LHE.Enc}(\mathsf{pk}_{\mathsf{LHE}}, r)$$

where $r \leftarrow_{\$} \mathbb{Z}_{\tilde{q}}$.

We formally analyze our scheme in the following. During the analysis, we set the parameters on demand and we show afterwards that our choices lead to a satisfiable set of constraints for which the underlying computational problems are still conjectured to be hard. The following theorem establishes correctness.

Theorem 4.1 (Split Correctness) *Let* $q \geq 2^k + 2^{\lceil \log(\tilde{q}+kB) \rceil}$. *Let* FHE *be a correct fully-homomorphic encryption scheme with linear decrypt-and-multiply and let* LHE *be a split correct linearly-homomorphic encryption scheme. Then the scheme as described above satisfies split correctness.*

Proof: Let us rewrite

$$\tilde{m} = \mathsf{LHE.Rec}(\rho, c) = \mathsf{LHE.Rec}(\mathsf{LHE.PDec}(\mathsf{sk}_{\mathsf{LHE}}, c), c)$$

where $c = \mathsf{LHE.Eval}\left(\mathsf{pk}_{\mathsf{LHE}}, \sum, (d, a)\right)$. We first expand the d term as

$$\begin{aligned} d &= \mathsf{LHE.Eval}(\mathsf{pk}_{\mathsf{LHE}}, g, (c_{(\mathsf{LHE},1)}, \dots, c_{(\mathsf{LHE},n)})) \\ &= \mathsf{LHE.Eval}(\mathsf{pk}_{\mathsf{LHE}}, g, (\mathsf{LHE.Enc}(\mathsf{pk}_{\mathsf{LHE}}, s_1), \dots, \mathsf{LHE.Enc}(\mathsf{pk}_{\mathsf{LHE}}, s_n))) \\ &= \mathsf{LHE.Enc}\left(\mathsf{pk}_{\mathsf{LHE}}, \sum_{j=1}^{k} \mathsf{Dec\&Mult}\left((s_1, \dots, s_n), d_j, 2^{\lceil \log(\tilde{q}+(k+1)B) \rceil + j}\right)\right) \end{aligned}$$

by the correctness of the LHE scheme, where

$$d_j = \mathsf{FHE.Eval}(\mathsf{pk}_{\mathsf{FHE}}, C_j, (c_1, \dots, c_\ell))$$

and $c_i = \mathsf{FHE.Enc}(\mathsf{pk}_{\mathsf{FHE}}, m_i)$. Thus by the decrypt-and-multiply correctness of the FHE scheme we can rewrite

$$\begin{aligned} d &= \mathsf{LHE.Enc}\left(\mathsf{pk}_{\mathsf{LHE}}, \sum_{j=1}^{k} 2^{\lceil \log(\tilde{q}+(k+1)B) \rceil + j} \cdot C_j(m_1, \dots, m_\ell) + e_j\right) \\ &= \mathsf{LHE.Enc}\left(\mathsf{pk}_{\mathsf{LHE}}, \sum_{j=1}^{k} 2^{\lceil \log(\tilde{q}+(k+1)B) \rceil + j} \cdot C_j(m_1, \dots, m_\ell) + \underbrace{\sum_{j=1}^{k} e_j}_{\tilde{e}}\right). \end{aligned}$$

For the a variable we have that $a = \mathsf{LHE.Enc}(\mathsf{pk}_{\mathsf{LHE}}, r)$, for some uniform $r \leftarrow\!\!\$\, \mathbb{Z}_{\tilde{q}}$, by definition of the oracle Sample. Thus

$$\begin{aligned} c &= \mathsf{LHE.Eval}\left(\mathsf{pk}_{\mathsf{LHE}}, \sum, (d, a)\right) \\ &= \mathsf{LHE.Enc}\left(\mathsf{pk}_{\mathsf{LHE}}, \sum_{j=1}^{k} 2^{\lceil \log(\tilde{q}+(k+1)B)\rceil + j} \cdot C_j(m_1, \dots, m_\ell) + \tilde{e} + r\right) \end{aligned}$$

by the correctness of the LHE scheme. Note that the sum $\tilde{e}$ is bounded from above by $k \cdot B$, whereas the term $\tilde{r}$ is trivially bounded from above by $\tilde{q}$. This implies that the output of the circuit is encoded in the higher order bits of $\tilde{m}$ with probability 1, for a large enough q. □

We then argue about the split security of the scheme. We remark that we analyze security in the presence of an oracle and we refer the reader to Section 4.3 and Section 4.4 for concrete instantiations.

Theorem 4.2 (Split Security) *Let $\tilde{q} \geq 2^\lambda \cdot k \cdot B$. Let* FHE *be a semantically secure fully-homomorphic encryption scheme and let* LHE *be a semantically secure linearly homomorphic encryption scheme with simulatable decryption hints. Then the scheme as described above satisfies split security in the* Sample*-hybrid model.*

Proof: Let $(m_0, m_1, C_1, \dots, C_\beta)$ be the inputs specified by the adversary at the beginning of the experiment. Consider the following series of hybrids.

Hybrid $\mathcal{H}_0$: Is defined as the original experiment. Denote the distribution induced by the random coins of the challenger by

$$(\mathsf{pk}, c = \mathsf{FHE.Enc}(\mathsf{pk}_{\mathsf{FHE}}, m_b), \rho_1, \dots, \rho_\beta)$$

where

$$\mathsf{pk} = (\mathsf{pk}_{\mathsf{FHE}}, \mathsf{pk}_{\mathsf{LHE}}, \mathsf{LHE.Enc}(\mathsf{pk}_{\mathsf{LHE}}, s_1), \dots, \mathsf{LHE.Enc}(\mathsf{pk}_{\mathsf{LHE}}, s_n))$$

and ρ_i is computed as $\mathsf{PDec}(\mathsf{sk}, \mathsf{Eval}(\mathsf{pk}, C_i, c))$.

Hybrids $\mathcal{H}_1 \dots \mathcal{H}_\beta$: Let $d^{(i)}$ be the variable d defined during the execution of $\mathsf{Eval}(\mathsf{pk}, C_i, c)$. The i-th hybrid $\mathcal{H}_i$ is defined to be identical to $\mathcal{H}_{i-1}$, except that the oracle $\mathsf{Sample}(\mathsf{pk}, \cdot)$ on input $d^{(i)}$ is programmed to output some a such that the resulting c is of the form

$$c = \mathsf{LHE.Enc}\left(\mathsf{pk}_{\mathsf{LHE}}, \mathsf{ECC}(C_i(m_b)) + \tilde{e} + r\right)$$

where ECC is the high-order bits encoding defined in the evaluation algorithm, $\tilde{e}$ is the sum of the decryption noises of the ciphertexts $(d^{(1)}, \dots, d^{(k)})$, as defined in the evaluation algorithm, and $r \leftarrow\!\!\$\, \mathbb{Z}_{\tilde{q}}$. Then $\tilde{\rho}_i$ is defined to be the decryption hint of c, computed using the random coins of the simulated a.

First observe that $\tilde{e}$ is efficiently computable given the secret key of the FHE scheme and therefore $\tilde{\rho}_i$ is also computable in polynomial time. It is important to observe that the distribution of c is identical to the previous hybrid and the difference lies only in the way $\tilde{\rho}_i$ is computed. Since the LHE scheme has simulatable hints, it follows that the distribution of $\mathcal{H}_i$ is identical to that of $\mathcal{H}_{i-1}$ and the change described here is only syntactical. That is,

$$\begin{aligned} &(\mathsf{pk}, \mathsf{FHE.Enc}(\mathsf{pk}_{\mathsf{FHE}}, m_b), \tilde{\rho}_1, \dots, \tilde{\rho}_{i-1}, \rho_i, \rho_{i+1}, \dots, \rho_\beta) \\ &= (\mathsf{pk}, \mathsf{FHE.Enc}(\mathsf{pk}_{\mathsf{FHE}}, m_b), \tilde{\rho}_1, \dots, \tilde{\rho}_{i-1}, \tilde{\rho}_i, \rho_{i+1}, \dots, \rho_\beta) \,. \end{aligned}$$

Hybrids $\mathcal{H}_{\beta+1} \dots \mathcal{H}_{2\beta}$: The $(\beta + i)$-th hybrid is defined to be identical to the previous ones except that the a variable corresponding to the i-th call in the evaluation algorithm is programmed such that

$$c = \mathsf{LHE.Enc}\left(\mathsf{pk}_{\mathsf{LHE}}, \mathsf{ECC}(C_i(m_b)) + \tilde{r}\right).$$

I.e., the noise term $\tilde{e}$ is omitted from the computation. Thus the only difference with respect to the previous hybrid is whether the noise term $\tilde{e}$ is included in the ciphertext or not. Since $\tilde{e}$ is bounded from above by $k \cdot B$ and $\tilde{q} \geq 2^\lambda \cdot k \cdot B$, by Lemma 1 the distribution induced by this hybrid is statistically indistinguishable from that of the previous one.

Hybrids $\mathcal{H}_{2\beta+1} \dots \mathcal{H}_{2\beta+n}$: The $(2\beta + i)$-th hybrid is defined as the previous one, except that the ciphertext $c_{(\mathsf{LHE},i)}$ in the public parameters is computed as the encryption of 0. Note that the secret key of the LHE scheme is no longer used in the computation of $(\tilde{\rho}_1, \dots, \tilde{\rho}_\beta)$ and therefore indistinguishability follows from an invocation of the semantic security of the LHE scheme. Specifically, the following distributions are computationally indistinguishable

$$\begin{pmatrix} \mathsf{LHE.Enc}(\mathsf{pk}_{\mathsf{LHE}}, 0), \dots, \mathsf{LHE.Enc}(\mathsf{pk}_{\mathsf{LHE}}, 0), \mathsf{LHE.Enc}(\mathsf{pk}_{\mathsf{LHE}}, s_i), \\ \mathsf{LHE.Enc}(\mathsf{pk}_{\mathsf{LHE}}, s_{i+1}), \dots, \mathsf{LHE.Enc}(\mathsf{pk}_{\mathsf{LHE}}, s_n) \end{pmatrix}$$
$$\approx \begin{pmatrix} \mathsf{LHE.Enc}(\mathsf{pk}_{\mathsf{LHE}}, 0), \dots, \mathsf{LHE.Enc}(\mathsf{pk}_{\mathsf{LHE}}, 0), \mathsf{LHE.Enc}(\mathsf{pk}_{\mathsf{LHE}}, 0), \\ \mathsf{LHE.Enc}(\mathsf{pk}_{\mathsf{LHE}}, s_{i+1}), \dots, \mathsf{LHE.Enc}(\mathsf{pk}_{\mathsf{LHE}}, s_n) \end{pmatrix}.$$

Hybrid $\mathcal{H}^{(b)}_{2\beta+n}$: We define the hybrid $\mathcal{H}^{(b)}_{2\beta+n}$ as $\mathcal{H}_{2\beta+n}$ with the challenger bit fixed to b. Note that the distribution induced by these hybrids is

$$(\mathsf{pk}, c = \mathsf{FHE.Enc}(\mathsf{pk}_{\mathsf{FHE}}, m_b), \tilde{\rho}_1, \dots, \tilde{\rho}_\beta)$$

where

$$\mathsf{pk} = (\mathsf{pk}_{\mathsf{FHE}}, \mathsf{pk}_{\mathsf{LHE}}, \mathsf{LHE.Enc}(\mathsf{pk}_{\mathsf{LHE}}, 0), \dots, \mathsf{LHE.Enc}(\mathsf{pk}_{\mathsf{LHE}}, 0)).$$

Observe that the secret key of the FHE scheme is no longer encoded in the public parameters and is not needed to compute $(\tilde{\rho}_1, \dots, \tilde{\rho}_\beta)$ either. It follows that any advantage that the adversary has in distinguishing $\mathcal{H}^{(0)}_{3\beta+n}$ from $\mathcal{H}^{(1)}_{3\beta+n}$ cannot be greater than the advantage in distinguishing $\mathsf{FHE.Enc}(\mathsf{pk}_{\mathsf{FHE}}, m_0)$ from $\mathsf{FHE.Enc}(\mathsf{pk}_{\mathsf{FHE}}, m_1)$. Thus, computational indistinguishability follows from an invocation of the semantic security of the FHE scheme. This concludes our proof. □

4.2 Instantiating the Oracle

To complete the description of our scheme, we discuss a few candidate instantiations for the oracle Sample. We require the underlying LHE scheme to have a dense ciphertext domain (which is the case for the Damgård-Jurik encryption scheme). Both of our proposals require to augment the public key of the scheme with an FHE encryption of the LHE secret key $c_{\mathsf{FHE}} \leftarrow \mathsf{FHE.Enc}(\mathsf{pk}_{\mathsf{FHE}}, \mathsf{sk}_{\mathsf{LHE}})$ and introduce new circularity assumptions between the FHE and the LHE schemes.

An alternate way to think of the oracle in Theorem 4.2 is to see it as an obfuscation for a special program, which is sufficient for realizing split FHE. The candidate constructions that we provide below can be seen as a natural and simple obfuscation of this special program.

4.2.1 A Simple Candidate

Let $\mathfrak{C}$ be the ciphertext domain of LHE. Throughout the following description, we fix the random coins of the algorithm (whenever needed) by drawing them from the evaluation of a cryptographic hash function Hash over the input. The intuition for our candidate is very simple: We sample an LHE ciphertext sampled using a random oracle (which is the reason why we need dense ciphertexts) and then we cancel out the high-order bits of the underlying plaintext by homomorphically decrypting the random ciphertext, isolating the chunk of information that we are interested in, and finally key-switch into the LHE scheme. The formal description is elaborated below.

$\mathsf{Sample}(\mathsf{pk}, x)$: On input a string $x \in \{0,1\}^*$ sample $y \leftarrow_{\$} \mathfrak{C}$, (using $\mathsf{Hash}(x)$ as the random coins) then compute

$$\tilde{y} \leftarrow \mathsf{FHE.Eval}\left(\mathsf{pk}_{\mathsf{FHE}}, -\lfloor \mathsf{LHE.Dec}(\cdot, y)/\tilde{q} \rfloor \cdot \tilde{q}, c_{\mathsf{FHE}}\right).$$

Define the following linear function over $\mathbb{Z}_q$:

$$h(x_0, x_1, \ldots, x_n) = x_0 + \mathsf{Dec\&Mult}\left((x_1, \ldots, x_n), \tilde{y}, 1\right).$$

Return

$$z \leftarrow \mathsf{LHE.Eval}\left(\mathsf{pk}_{\mathsf{LHE}}, h, (y, c_{(\mathsf{LHE},1)}, \ldots, c_{(\mathsf{LHE},n)})\right).$$

Observe that y is an element in the ciphertext domain of LHE and it is of the form $y = \mathsf{LHE.Enc}(\mathsf{pk}_{\mathsf{LHE}}, m)$, for some $m \in \mathbb{Z}_q$, since LHE has a dense ciphertext domain. Furthermore, by the correctness of the FHE and the LHE scheme, we have that

$$\begin{aligned}\tilde{y} &= \mathsf{FHE.Eval}\left(\mathsf{pk}_{\mathsf{FHE}}, -\lfloor \mathsf{LHE.Dec}(\cdot, y)/\tilde{q} \rfloor \cdot \tilde{q}, c_{\mathsf{FHE}}\right)\\ &= \mathsf{FHE.Eval}\left(\mathsf{pk}_{\mathsf{FHE}}, -\lfloor \mathsf{LHE.Dec}(\cdot, y)/\tilde{q} \rfloor \cdot \tilde{q}, \mathsf{FHE.Enc}(\mathsf{pk}_{\mathsf{FHE}}, \mathsf{sk}_{\mathsf{LHE}})\right)\\ &= \mathsf{FHE.Enc}\left(\mathsf{pk}_{\mathsf{FHE}}, -\lfloor \mathsf{LHE.Dec}(\mathsf{sk}_{\mathsf{LHE}}, y)/\tilde{q} \rfloor \cdot \tilde{q}\right)\\ &= \mathsf{FHE.Enc}\left(\mathsf{pk}_{\mathsf{FHE}}, -\lfloor m/\tilde{q} \rfloor \cdot \tilde{q}\right).\end{aligned}$$

Let $q \geq 2^\lambda \cdot \tilde{q}$ (this additional constraint is compatible with the parameters settings defined above), then we have that

$$\begin{aligned}z &= \mathsf{LHE.Eval}\left(\mathsf{pk}_{\mathsf{LHE}}, h, (y, c_{(\mathsf{LHE},1)}, \ldots, c_{(\mathsf{LHE},n)})\right)\\ &= \mathsf{LHE.Enc}\left(\mathsf{pk}_{\mathsf{LHE}}, m + \mathsf{Dec\&Mult}\left((s_1, \ldots, s_n), \tilde{y}, 1\right)\right)\\ &= \mathsf{LHE.Enc}\left(\mathsf{pk}_{\mathsf{LHE}}, m - \lfloor m/\tilde{q} \rfloor \cdot \tilde{q} + e\right)\\ &= \mathsf{LHE.Enc}\left(\mathsf{pk}_{\mathsf{LHE}}, (m \mod \tilde{q}) + e\right)\end{aligned}$$

Where $m \mod q$ is distributed uniformly over $\mathbb{Z}_{\tilde{q}}$ except for the event where $m \in \{q - (q \mod \tilde{q}), \ldots, q\}$, which happens only with negligible probability. It follows that the output of the oracle is syntactically well formed. However, a closer look to the oracle instantiation reveals two lingering assumptions.

(1) *Circular Security:* The addition of $c_{\mathsf{FHE}} = \mathsf{FHE.Enc}(\mathsf{pk}_{\mathsf{FHE}}, \mathsf{sk}_{\mathsf{LHE}})$ introduces a circular dependency in the security of the LHE and FHE schemes (recall that our split FHE construction includes in the public key an encryption of $\mathsf{sk}_{\mathsf{FHE}}$ under $\mathsf{pk}_{\mathsf{LHE}}$). Circular

security is however widely considered to be a very mild assumption and currently is the only known approach to construct plain (as opposed to levelled) FHE from LWE via the bootstrapping theorem [Gen09].

(2) *Correlations:* Although $\tilde{y}$ is an FHE encryption of the correct value, it is not necessarily uniformly distributed, conditioned on y. In particular the randomness of $\tilde{y}$ may depend in some intricate way on the low-order bits of m. For the specific case of LWE-based schemes, the *noise* term might carry some information about $m \mod \tilde{q}$, which could introduce some harmful correlation. However, the noise function is typically highly non-linear and therefore appears to be difficult to exploit. We also stress that we only consider honest executions of the $\mathsf{FHE.Eval}$ algorithm.

While (1) can be regarded as a standard assumption, we view (2) as a natural conjecture which we believe holds true for any natural/known candidate instantiation of the FHE and LHE schemes. In light of these considerations, we conjecture that the implementation as describe above already leads to a secure split FHE scheme.

4.2.2 Towards Removing Correlations

A natural approach towards removing the correlation of the LHE and FHE ciphertexts is that of ciphertext sanitization [DS16]: One could expect that repeatedly bootstrapping the FHE ciphertext would decorrelate the noise from the companion LHE ciphertext. Unfortunately our settings are different than those typically considered in the literature, in the sense that the santiziation procedure must be carried out by the distinguisher and therefore cannot use private random coins. Although it appears hard to formally analyze the effectiveness of these methods in our settings, we expect that these techniques might (at least heuristically) help to obliterate harmful correlations. In this work we take a different route and we suggest a simple heuristic method to prevent correlations. In a nutshell, the idea is to sample a set of random plaintexts and define the random string as the sum of a uniformly sampled subset S of these plaintext. The key observation is that subset sum is a linear operation and therefore can be performed directly in the LHE scheme, which implies that the *leakage* of the FHE scheme cannot depend on S. As for the previous construction, our instantiation contains and a ciphertext $c_{\mathsf{FHE}} = \mathsf{FHE.Enc}(\mathsf{pk}_{\mathsf{FHE}}, \mathsf{sk}_{\mathsf{LHE}})$. The scheme is parametrized by some $\sigma \in \mathsf{poly}(\lambda)$, which defines the size of the set S. In the following description we present the algorithm as randomized, although this simplification can be easily bypassed with standard techniques (e.g., computing the random coins using a cryptographic hash $\mathsf{Hash}(x)$).

$\mathsf{Sample}(\mathsf{pk}, x)$: On input a string $x \in \{0,1\}^*$ sample a random set $S \leftarrow_{\$} \{0,1\}^{\sigma}$. Then, for all $i \in [\sigma]$, do the following:

- If $S_i = 1$, sample a uniform $y_i \leftarrow_{\$} \mathfrak{C}$.
- If $S_i = 0$, sample a uniform encryption $y_i \leftarrow_{\$} \mathsf{LHE.Enc}(\mathsf{pk}_{\mathsf{LHE}}, m_i)$, for a random known m_i.

Then compute

$$\tilde{y} \leftarrow \mathsf{FHE.Eval}\left(\mathsf{pk}_{\mathsf{FHE}}, -\sum_{i=1}^{\sigma} \lfloor \mathsf{LHE.Dec}(\cdot, y_i)/\tilde{q} \rceil \cdot \tilde{q}, c_{\mathsf{FHE}}\right).$$

Let h be the following linear function over $\mathbb{Z}_q$:

$$h(w_1, \ldots, w_{|S|}, x_1, \ldots, x_n) = \sum_{i \in S} w_i + \sum_{i \notin S} \lfloor m_i/\tilde{q} \rfloor \cdot \tilde{q} + \mathsf{Dec\&Mult}\left((x_1, \ldots, x_n), \tilde{y}, 1\right).$$

Return

$$z \leftarrow \mathsf{LHE.Eval}\left(\mathsf{pk}_{\mathsf{LHE}}, h, \left(\{y_i\}_{i \in S}, c_{(\mathsf{LHE},1)}, \ldots, c_{(\mathsf{LHE},n)}\right)\right).$$

To see why the implementation is syntactically correct, observe that

$$\begin{aligned}
\tilde{y} &= \mathsf{FHE.Eval}\left(\mathsf{pk}_{\mathsf{FHE}}, -\sum_{i=1}^{\sigma} \lfloor \mathsf{LHE.Dec}(\cdot, y_i)/\tilde{q} \rfloor \cdot \tilde{q}, c_{\mathsf{FHE}}\right) \\
&= \mathsf{FHE.Enc}\left(\mathsf{pk}_{\mathsf{FHE}}, -\sum_{i=1}^{\sigma} \lfloor \mathsf{LHE.Dec}(\mathsf{sk}_{\mathsf{LHE}}, y_i)/\tilde{q} \rfloor \cdot \tilde{q}\right) \\
&= \mathsf{FHE.Enc}\left(\mathsf{pk}_{\mathsf{FHE}}, -\sum_{i=1}^{\sigma} \lfloor m_i/\tilde{q} \rfloor \cdot \tilde{q}\right)
\end{aligned}$$

by the evaluation correctness of the FHE scheme. Invoking the correctness of the LHE scheme we have that

$$\begin{aligned}
z &= \mathsf{LHE.Eval}\left(\mathsf{pk}_{\mathsf{LHE}}, h, \left(\{y_i\}_{i \in S}, c_{(\mathsf{LHE},1)}, \ldots, c_{(\mathsf{LHE},n)}\right)\right) \\
&= \mathsf{LHE.Eval}\left(\mathsf{pk}_{\mathsf{LHE}}, h, \left(\{\mathsf{LHE.Enc}(\mathsf{pk}_{\mathsf{LHE}}, m_i)\}_{i \in S}, c_{(\mathsf{LHE},1)}, \ldots, c_{(\mathsf{LHE},n)}\right)\right) \\
&= \mathsf{LHE.Enc}\left(\mathsf{pk}_{\mathsf{LHE}}, \sum_{i \in S} m_i + \sum_{i \notin S} \lfloor m_i/\tilde{q} \rfloor \cdot \tilde{q} - \sum_{i=1}^{\sigma} \lfloor m_i/\tilde{q} \rfloor \cdot \tilde{q} + e\right) \\
&= \mathsf{LHE.Enc}\left(\mathsf{pk}_{\mathsf{LHE}}, \sum_{i \in S} (m_i \mod \tilde{q}) + \sum_{i=1}^{\sigma} \lfloor m_i/\tilde{q} \rfloor \cdot \tilde{q} - \sum_{i=1}^{\sigma} \lfloor m_i/\tilde{q} \rfloor \cdot \tilde{q} + e\right) \\
&= \mathsf{LHE.Enc}\left(\mathsf{pk}_{\mathsf{LHE}}, \underbrace{\sum_{i \in S} (m_i \mod \tilde{q})}_{\tilde{m}} + e\right)
\end{aligned}$$

which is exactly what we want, except that $\tilde{m}$ is slightly larger than $\tilde{q}$, by a factor of at most σ. This can still be used in our main construction by adjusting the error correcting code accordingly. The intuition why we believe that this variant is secure is that the leakage in the FHE randomness cannot depend on the set S, since the distributions of all y_i are statistically close (recall that LHE has dense ciphertexts). Thus, S (which is chosen uniformly) resembles the behavior of a binary extractor on $(m_i \mod \tilde{q})$. Nevertheless, proving a formal statement remains an interesting open question.

4.3 Damgård-Jurik Instantiation

When instantiating the LHE scheme with the Damgård-Jurik encryption scheme (as described in Section 3.3) and the FHE scheme with any LWE-based scheme with linear decrypt-and-multiply (e.g., the scheme proposed in [GSW13]) we obtain a split FHE which satisfies the notion of split compactness: The hint ρ is of size $N = \mathsf{poly}(\lambda)$ and in particular is arbitrarily smaller than the size of the plaintext space $q = N^\zeta$. For essentially any choice of the LWE-based FHE scheme with modulus q, the size of the public key and fresh ciphertexts depends polynomially in λ and linearly in $\log(q) = \log(N^\zeta)$, which gives us the desired bound. The analysis above sets the following additional constraints:

- $q \geq 2^k + 2^{\lceil \log(\tilde{q}+(k+1)B)\rceil}$ and
- $\tilde{q} \geq 2^\lambda \cdot (k+1) \cdot B$

which are always satisfied for $q = N^\zeta$, by setting the integer ζ to be large enough. Note that this choice of parameters fixes the modulus of the FHE with linear decrypt-and-multiply to $\mathbb{Z}_{N^\zeta}$, which is super-polynomially larger than the noise bound B. Finally, the LWE parameter n is free and can be set to any value for which the corresponding problem (with super-polynomial modulus-to-noise ratio) is conjectured to be hard.

4.4 Lattice-Based Instantiation

In the following we describe a split FHE construction based exclusively on LWE. Our scheme assumes the existence of a fully-homomorphic encryption scheme $\mathsf{FHE} = (\mathsf{FHE.KeyGen}, \mathsf{FHE.Enc}, \mathsf{FHE.Eval}, \mathsf{FHE.Dec})$ with linear decrypt-and-multiply with noise bound B and (for simplicity prime) modulus q. To simplify the exposition we also assume the existence of a public-key encryption scheme $\mathsf{PKE} = (\mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$. This is without loss of generality since any FHE scheme is also a public-key encryption scheme.

In favor of a simpler exposition, we present a direct construction of FHE with split decryption, instead of instantiating each building block individually. We stress that, in contrast with the instantiation based on the Damgård-Jurik encryption scheme (Section 4.3), this scheme does not satisfy the syntactical requirements to apply the generic transformations (described in Section 4.2) to lift the scheme to the plain model. Nevertheless, such a scheme is still useful a building block to construct rate-1 reusable garbled circuits (Section 6).

$\mathsf{KeyGen}(1^\lambda)$: On input the security parameter 1^λ sample $(\mathsf{sk}_{\mathsf{FHE}}, \mathsf{pk}_{\mathsf{FHE}}) \leftarrow \mathsf{FHE.KeyGen}(1^\lambda)$ and parse $\mathsf{sk}_{\mathsf{FHE}} = (s_1, \dots, s_n)$. Sample $\mathbf{A} \leftarrow_\$ \mathbb{Z}_q^{k\times n}$, $\mathbf{R} \leftarrow \mathbb{Z}_q^{n\times nk\lceil\log(q)\rceil}$, and compute

$$\mathbf{B} = \mathbf{AR} + \mathbf{E} + (s_1, \dots, s_n) \otimes \mathbf{G}$$

where $\mathbf{E}$ is a noise matrix and $\mathbf{G}$ is the gadget matrix, both of proper dimensions. Then sample $(\mathsf{sk}_{\mathsf{PKE}}, \mathsf{pk}_{\mathsf{PKE}}) \leftarrow \mathsf{PKE.KeyGen}(1^\lambda)$ and set

$$\mathsf{sk} = (\mathsf{sk}_{\mathsf{PKE}}, \mathbf{R}) \text{ and } \mathsf{pk} = (\mathsf{pk}_{\mathsf{PKE}}, \mathsf{pk}_{\mathsf{FHE}}, \mathbf{A}, \mathbf{B}).$$

$\mathsf{Enc}(\mathsf{pk}, m)$: On input a message m return

$$c \leftarrow \mathsf{FHE.Enc}(\mathsf{pk}_{\mathsf{FHE}}, m).$$

$\mathsf{Eval}^{\mathsf{Sample}}(\mathsf{pk}, C, (c_1, \dots, c_\ell))$: On input a circuit C with ℓ bits of input and k bits of output and a vector of ciphertexts $(c_1, \dots, c_\ell)$, let, for all $j \in [k]$, C_j be the circuit that returns the j-th bit of the output of C, then compute

$$\tilde{c}_j \leftarrow \mathsf{FHE.Eval}(\mathsf{pk}_{\mathsf{FHE}}, C_j, (c_1, \dots, c_\ell)).$$

Query the sampling oracle $(\tau, c_\tau) \leftarrow \mathsf{Sample}(\tilde{c}_1, \dots, \tilde{c}_k)$ and return $(\tilde{c}_1, \dots, \tilde{c}_k, c_\tau, \tau)$.

$\mathsf{PDec}(\mathsf{sk}, c)$: On input an evaluated ciphertext $c = (\tilde{c}_1, \dots, \tilde{c}_k, c_\tau, \tau)$ compute

$$\mathbf{t} \leftarrow \mathsf{PKE.Dec}(\mathsf{sk}_{\mathsf{PKE}}, c_\tau)$$

and return

$$\rho = \mathbf{R}\mathbf{G}^{-1}(L_c) - \mathbf{t}$$

where L_c is the vector concatenation corresponding to the coefficients of the linear function $\mathsf{Dec\&Mult}\left((\cdots), (\tilde{c}_1, \dots, \tilde{c}_k), \lceil q/2 \rceil\right)$.

$\mathsf{Rec}(\rho, c)$: On input an evaluated ciphertext $c = (\tilde{c}_1, \dots, \tilde{c}_k, c_\tau, \tau)$ and a decryption hint ρ return

$$\tilde{m} = \mathsf{MSB}\left(\mathbf{B}\mathbf{G}^{-1}(L_c) - \mathbf{A}\rho\right) \oplus \tau$$

where L_c is defined as above.

The sampling oracle is defined below. For simplicity we describe the oracle as randomized and we implement the deterministic variant by fixing the random coins using, e.g., a hash function.

$\mathsf{Sample}(\mathsf{pk}, x)$: On input a string $x \in \{0,1\}^*$ sample a uniform $\mathbf{t} \leftarrow_{\$} \mathbb{Z}_q^n$ and compute encryption $c_\tau \leftarrow \mathsf{PKE.Enc}(\mathsf{pk}_{\mathsf{PKE}}, \mathbf{t})$. Let

$$\tau = \mathsf{MSB}(\mathbf{A}\mathbf{t})$$

where MSB returns the most significant bit of each element of the input vector. Return (τ, c_τ).

The correctness of the scheme follows since

$$\begin{aligned}
\mathsf{MSB}\left(\mathbf{B}\mathbf{G}^{-1}(L_c) - \mathbf{A}\rho\right) &= \mathsf{MSB}\left(\left(\mathbf{A}\mathbf{R} + \mathbf{E} + (s_1, \dots, s_n) \otimes \mathbf{G}\right)\mathbf{G}^{-1}(L_c) - \mathbf{A}\rho\right) \\
&= \mathsf{MSB}\left(\mathbf{A}\mathbf{R}\mathbf{G}^{-1}(L_c) + \mathbf{E}\mathbf{G}^{-1}(L_c) + L_c(s_1, \dots, s_n) - \mathbf{A}\rho\right) \\
&= \mathsf{MSB}\left(\mathbf{A}\mathbf{R}\mathbf{G}^{-1}(L_c) + \mathbf{e} + L_c(s_1, \dots, s_n) - \mathbf{A}\left(\mathbf{R}\mathbf{G}^{-1}(L_c) - \mathbf{t}\right)\right) \\
&= \mathsf{MSB}\left(L_c(s_1, \dots, s_n) + \mathbf{A}\mathbf{t} + \mathbf{e}\right) \\
&= \mathsf{MSB}\left(\mathsf{Dec\&Mult}\left((s_1, \dots, s_n), (\tilde{c}_1, \dots, \tilde{c}_k), \lceil q/2 \rceil\right) + \mathbf{A}\mathbf{t} + \mathbf{e}\right)
\end{aligned}$$

where

$$\begin{aligned}
\tilde{c}_j &= \mathsf{FHE.Eval}(\mathsf{pk}_{\mathsf{FHE}}, C_j, (c_1, \dots, c_\ell)) \\
&= \mathsf{FHE.Eval}(\mathsf{pk}_{\mathsf{FHE}}, C_j, (\mathsf{FHE.Enc}(\mathsf{pk}_{\mathsf{FHE}}, m_1), \dots, \mathsf{FHE.Enc}(\mathsf{pk}_{\mathsf{FHE}}, m_\ell))) \\
&= \mathsf{FHE.Enc}(\mathsf{pk}, C_j(m_1, \dots, m_\ell))
\end{aligned}$$

thus

$$\begin{aligned}
\mathsf{MSB}\left(\mathbf{B}\mathbf{G}^{-1}(L_c) - \mathbf{A}\rho\right) \oplus \tau &= \mathsf{MSB}\left(\lceil q/2 \rceil \cdot C(m_1, \dots, m_\ell) + \tilde{\mathbf{e}} + \mathbf{A}\mathbf{t} + \mathbf{e}\right) \oplus \tau && (2) \\
&= C(m_1, \dots, m_\ell) \oplus \mathsf{MSB}\left(\mathbf{A}\mathbf{t} + \mathbf{e} + \tilde{\mathbf{e}}\right) \oplus \tau && (3) \\
&= C(m_1, \dots, m_\ell) \oplus \mathsf{MSB}\left(\mathbf{A}\mathbf{t}\right) \oplus \mathsf{MSB}\left(\mathbf{A}\mathbf{t}\right) && (4) \\
&= C(m_1, \dots, m_\ell) && (5)
\end{aligned}$$

with all but negligible probability over the random choice of $\mathbf{t}$. To establish this, we need to show that equality (4) holds, except with negligible probability over the choice of $\mathbf{t}$ and $\mathbf{A}$.

Observe that $\|\tilde{\mathbf{e}} + \mathbf{e}\|_\infty \leq B \cdot (n \cdot k \cdot \lceil \log(q) \rceil + 1)$. For a given $z \in \mathbb{Z}_q$ say that z is bad if $|z - q/4| < B \cdot (n \cdot k \cdot \lceil \log(q) \rceil + 1)$ or $|z + q/4| < B \cdot (n \cdot k \cdot \lceil \log(q) \rceil + 1)$. By choosing q sufficiently large, e.g. by $q/4 > 2^\lambda \cdot B \cdot (n \cdot k \cdot \lceil \log(q) \rceil + 1)$, we get that the probability that a uniformly random $z \leftarrow\!\!\$\, \mathbb{Z}_q$ is bad is negligible. Further say that a vector $\mathbf{z} \in \mathbb{Z}_q^k$ is bad if any of its components is bad.

By Lemma 2 we can fix a $\mathbf{t} \in \mathbb{Z}_q^n$ such if $\mathbf{a} \leftarrow\!\!\$\, \mathbb{Z}_q^n$ is chosen uniformly random, then $\langle \mathbf{a}, \mathbf{t} \rangle$ is distributed uniformly random, as a uniformly random $\mathbf{t} \in \mathbb{Z}_q^n$ has this property except with probability $\log(q) \cdot 2^{-n}$. Let $\mathbf{a}_i, \ldots, \mathbf{a}_k$ be the rows of $\mathbf{A}$. Since the $\mathbf{a}_i$ are uniformly random, so are the $\langle \mathbf{a}_i, \mathbf{t} \rangle$. Thus it holds for every i that $\Pr[\langle \mathbf{a}_i, \mathbf{t} \rangle \text{ bad}] < \mathsf{negl}(\lambda)$. By a union bound we immediately get that $\Pr_{\mathbf{A}}[\mathbf{At} \text{ bad}] = \Pr_{\mathbf{A}}[\exists i : \langle \mathbf{a}_i, \mathbf{t} \rangle \text{ bad}] < k \cdot \mathsf{negl}(\lambda) = \mathsf{negl}(\lambda)$. Putting everything together, we get that $\Pr_{\mathbf{A},\mathbf{t}}[\mathbf{At} \text{ bad}] < \mathsf{negl}(\lambda) + \log(q) \cdot 2^{-n} = \mathsf{negl}(\lambda)$. We can conclude that $\mathsf{MSB}(\mathbf{At}) = \mathsf{MSB}(\mathbf{At})$, except with negligible choice over $\mathbf{t}$ and $\mathbf{A}$.

We now proceed to argue about the security of our scheme.

Theorem 4.3 (Split Security) *Let* FHE *be a semantically secure fully-homomorphic encryption scheme with simulable hints and let* PKE *be a semantically secure public-key encryption scheme. If the LWE problem is hard, then the scheme as described above satisfies split security in the* Sample*-hybrid model.*

Proof: Let $(m_0, m_1, C_1, \ldots, C_\beta)$ be the inputs specified by the adversary at the beginning of the experiment. Consider the following series of hybrids.

Hybrid $\mathcal{H}_0$: Is defined as the original experiment.

Hybrids $\mathcal{H}_1 \ldots \mathcal{H}_\beta$: We define the hybrid $\mathcal{H}_i$ to be identical to the previous one except that the i-th hint ρ_i is sampled uniformly from $\mathbb{Z}_q^n$ and the corresponding query to the Sample oracle is answered by computing $c_\tau \leftarrow \mathsf{PKE.Enc}(\mathsf{pk}_{\mathsf{PKE}}, 0)$ and setting

$$\tau = \mathsf{MSB}(\mathbf{B}\mathbf{G}^{-1}(L_c) - \mathbf{A}\rho) \oplus C(m_1, \ldots, m_\ell).$$

It is not hard to see that the distribution induced by this hybrid is computationally indistinguishable from the previous one, by a reduction against the semantic security of PKE.

Hybrid $\mathcal{H}_{\beta+1}$: This hybrid is defined to be exactly as the previous one except that the element $\mathbf{B}$ is chosen uniformly over $\mathbb{Z}_q^{k \times nk\lceil \log(q) \rceil}$. Indistinguishability follows from a standard hybrid argument against the LWE assumption.

Hybrid $\mathcal{H}_{\beta+1}^{(b)}$: We define the hybrid $\mathcal{H}_{\beta+1}^{(b)}$ as $\mathcal{H}_{\beta+1}$ with the challenger bit fixed to b. By the semantic security of the FHE scheme it follows that $\mathcal{H}_\beta^{(0)}$ and $\mathcal{H}_\beta^{(1)}$ are computationally indistinguishable. This concludes our proof. □

Note that the above prove implicitly defines a simulator (in the Sample-hybrid model) for decryption hints: Sample $\rho \leftarrow\!\!\$\, \mathbb{Z}_q^n$ and program the corresponding query to the Sample oracle to $c_\tau \leftarrow \mathsf{PKE.Enc}(\mathsf{pk}_{\mathsf{PKE}}, 0)$ and

$$\tau = \mathsf{MSB}(\mathbf{B}\mathbf{G}^{-1}(L_c) - \mathbf{A}\rho) \oplus m.$$

where L_c is the linear function defined by the input ciphertext c and m is the message given as input to the simulator. This implies that the scheme as described above satisfies (computational) hint simulatability in the Sample-hybrid model.

Finally we observe that the scheme satisfies split compactness as the size of the decryption hints is $O(n \log(q)) = \mathsf{poly}(\lambda)$ and in particular is independent of the output size k.

5 Split Fully-Homomorphic Encryption Implies Obfuscation

In order to construct fully-fledged iO from split FHE, we rely on a theorem from Lin et al. [LPST16], which we recall in the following. Roughly speaking, the theorem states that, under the assumption that the LWE problem is sub-exponentially hard, it suffices to consider circuits with a polynomial-size input domain and obfuscators that output obfuscated circuits of size slightly sublinear in size of the truth table of the circuit.

Theorem 5.1 ([LPST16]) *Assuming sub-exponentially hard LWE, if there exists a sub-exponentially secure indistinguishability obfuscator for* $\mathsf{P}^{\mathsf{log}}/\mathsf{poly}$ *with non-trivial efficiency, then there exists an indistinguishability obfuscator for* P/poly *with sub-exponential security.*

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 $\mathsf{poly}(\lambda, |C|) \cdot 2^{\eta \cdot (1-\varepsilon)}$, for some constant $\varepsilon > 0$. Note that the above theorem poses no restriction on the runtime of the obfuscator, which can be as large as $\mathsf{poly}(\lambda, |C|) \cdot 2^{\eta}$.

In the following we show how to construct an obfuscator for $\mathsf{P}^{\mathsf{log}}/\mathsf{poly}$ with non-trivial efficiency. We assume only the existence of a (levelled) split FHE scheme $\mathsf{sFHE} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{PDec}, \mathsf{Rec})$.

$\mathsf{iO}(C)$: On input the description of a circuit C, sample a pair $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ and compute $c \leftarrow \mathsf{Enc}(\mathsf{pk}, C)$. For all $i \in \left[2^{\eta/2}\right]$ define the universal circuit $\mathfrak{U}_i$ as

$$\mathfrak{U}_i(C) = C\left((i-1) \cdot 2^{\eta/2}\right) \| \dots \| C\left(i \cdot 2^{\eta/2} - 1\right).$$

Then compute $c_i \leftarrow \mathsf{Eval}(\mathsf{pk}, \mathfrak{U}_i, c)$ and $\rho_i \leftarrow \mathsf{PDec}(\mathsf{sk}, c_i)$. The obfuscated circuit is defined to be $(\mathsf{pk}, c, \rho_1, \dots, \rho_{2^{\eta/2}})$.

First we discuss how to evaluate an obfuscated circuit: On input some $x \in \{0,1\}^\eta$, parse it as an integer and round it to the nearest multiple of $2^{\eta/2}$ (let such integer be $\bar{x}$) such that $\bar{x} \leq x$. Then compute $c_{\bar{x}} \leftarrow \mathsf{Eval}(\mathsf{pk}, \mathfrak{U}_{\bar{x}}, c)$ and $m \leftarrow \mathsf{Rec}(\rho_{\bar{x}}, c_{\bar{x}})$. Read the output as the $(x - \bar{x})$-th bit of m.

5.1 Analysis

Note that the runtime of the obfuscator is dominated by $2^{\eta/2}$ evaluations of the split FHE ciphertext, where each subroutine homomorphically evaluates the circuit C $2^{\eta/2}$-many times. Thus the total runtime of the obfuscator is in the order of $\mathsf{poly}(\lambda, |C|) \cdot 2^{\eta}$. We now argue that our obfuscator has non trivial efficiency in terms of output size. We analyze the size of each component of the obfuscated circuit:

- By the compactness of the split FHE scheme, the public key pk grows linearly with the size of the output domain, i.e., $2^{\eta/2}$, and polynomially in the security parameter.
- The ciphertext c grows linearly with the size of the encrypted message and therefore, by the compactness of the split FHE scheme, bounded by $\mathsf{poly}(\lambda, |C|) \cdot 2^{\eta/2}$.
- Each decryption hint ρ_i is of size $\mathsf{poly}(\lambda)$, since the underlying split FHE is compact. As an obfuscated circuit consists of $2^{\eta/2}$-many decryption hints, the size of the vector $(\rho_1, \ldots, \rho_{2^{\eta/2}})$ is $\mathsf{poly}(\lambda) \cdot 2^{\eta/2}$.

It follows that the total size of the obfuscated circuit is bounded from above by $\mathsf{poly}(\lambda, |C|) \cdot 2^{\eta/2}$. What is left to be shown is that our obfuscator satisfies the notion of indistinguishability obfuscation.

Theorem 5.2 (Indistinguishability Obfuscation) *Let* sFHE *be a sub-exponentially secure levelled split FHE scheme. Then the scheme as described above is a sub-exponentially secure indistinguishability obfuscator.*

Proof: By the perfect correctness of the split FHE scheme it follows that the obfuscated circuit is functionally equivalent to the plain circuit. Indistinguishability follows immediately from the split security of sFHE: If the split FHE is secure against a distinguisher running in sub-exponential time, then so is iO. □

6 Rate-1 Reusable Garbled Circuits

In this section we recall the definition of reusable garbled circuits (rGC) and discuss how to construct a scheme with rate-1 input encodings (in the output length).

6.1 Definition

We briefly recall the syntax of garbling schemes as defined by Yao [Yao86].

Definition 6.1 (Garbling Scheme) *A garbling scheme consists of the following efficient algorithms*

$\mathsf{GC.Garble}(1^\lambda, C)$: *On input the security parameter* 1^λ *and a circuit* C*, the garbling algorithm returns a garbled circuit* $\tilde{C}$ *and a secret key* sk.

$\mathsf{GC.Encode}(\mathsf{sk}, x)$: *On input the secret key* sk *and an input* x*, the encoding algorithm returns an encoding* c.

$\mathsf{GC.Eval}(\tilde{C}, c)$: *On input a garbled circuit* $\tilde{C}$ *and an encoding* c*, the evaluation algorithm returns an output* y.

We define correctness for a garbling scheme.

Definition 6.2 (Correctness) *A garbling scheme* $(\mathsf{GC.Garble}, \mathsf{GC.Encode}, \mathsf{GC.Eval})$ *is correct if for all* $\lambda \in \mathbb{N}$*, all circuits* C*, all inputs* x*, and all pairs* $(\tilde{C}, \mathsf{sk})$ *in the support of* $\mathsf{GC.Garble}(1^\lambda, C)$ *it holds that*

$$\Pr[\mathsf{GC.Eval}(\tilde{C}, \mathsf{GC.Encode}(\mathsf{sk}, x)) = C(x)] = 1.$$

We recall the notion of reusable security from [GKP+13], that requires that the encodings of inputs x with respect to a garbled circuit C reveal nothing beyond $C(x)$.

Definition 6.3 (Reusable Security) *A garbling scheme* $(\mathsf{GC.Garble}, \mathsf{GC.Encode}, \mathsf{GC.Eval})$ *is reusably secure if there exists a PPT simulator* $\mathsf{Sim} = (\mathsf{Sim}_0, \mathsf{Sim}_1)$ *such that for all PPT attackers* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that*

$$\left| \begin{array}{r} \Pr\left[1 \leftarrow \mathcal{A}_2^{\mathsf{GC.Encode}(\mathsf{sk},\cdot)}(\tilde{C}, \mathsf{st}) \middle| \begin{array}{l} (C, \mathsf{st}) \leftarrow \mathcal{A}_1(1^\lambda) \\ (\tilde{C}, \mathsf{sk}) \leftarrow \mathsf{GC.Garble}(1^\lambda, C) \end{array} \right] - \\ \Pr\left[1 \leftarrow \mathcal{A}_2^{\mathcal{O}_{C,\tilde{\mathsf{st}}}(\cdot)}(\tilde{C}, \mathsf{st}) \middle| \begin{array}{l} (C, \mathsf{st}) \leftarrow \mathcal{A}_1(1^\lambda) \\ (\tilde{C}, \tilde{\mathsf{st}}) \leftarrow \mathsf{Sim}_0(1^\lambda, 1^{|C|}) \end{array} \right] \end{array} \right| = \mathsf{negl}(\lambda)$$

where $\mathcal{O}_{C,\tilde{\mathsf{st}}}$*, on input* x*, runs* $(c, \tilde{\mathsf{st}}') \leftarrow \mathsf{Sim}_1(1^{|x|}, \tilde{\mathsf{st}}, C(x))$*, sets* $\tilde{\mathsf{st}} = \tilde{\mathsf{st}}'$ *and returns* c*.*

Theorem 6.4 ([GKP⁺13]) *Assuming sub-exponentially hard LWE, there exists a reusable garbled circuit scheme with input encodings of size* $\mathsf{poly}(\lambda, d, |x|, |y|)$*, where* d *is the depth of the circuit.*

6.2 Rate-1 Construction

In the following we present our rGC scheme with rate-1 encodings. We assume the existence of the following building blocks.

- A one-time secure split FHE scheme $\mathsf{FHE} = (\mathsf{FHE.KeyGen}, \mathsf{FHE.Enc}, \mathsf{FHE.Eval}, \mathsf{FHE.Sample}, \mathsf{FHE.PDec}, \mathsf{FHE.Rec})$ with simulatable decryption hints.
- A reusable garbled circuit $\mathsf{rGC} = (\mathsf{GC.Garble}, \mathsf{GC.Encode}, \mathsf{GC.Eval})$.

We stress that the former can be constructed using the schemes presented in Section 4.1 and Section 4.4, without making any additional assumption. As a main corollary, we obtain that assuming the hardness of the LWE problem is sufficient.

$\mathsf{Garble}(1^\lambda, C)$: On input the security parameter 1^λ and a circuit C, sample a key pair $(\mathsf{sk}_{\mathsf{FHE}}, \mathsf{pk}_{\mathsf{FHE}}) \leftarrow \mathsf{FHE.KeyGen}(1^\lambda)$ and compute $(\tilde{C}, \mathsf{sk}_{\mathsf{rGC}}) \leftarrow \mathsf{GC.Garble}(1^\lambda, \Gamma)$ where Γ is the following circuit.

> $\Gamma(x, r, s)$: Compute $\tilde{c} \leftarrow \mathsf{FHE.Enc}(\mathsf{pk}_{\mathsf{FHE}}, x; r)$, $a \leftarrow \mathsf{Sample}(\mathsf{pk}_{\mathsf{FHE}}, \tilde{c}; s)$, and $c \leftarrow \mathsf{FHE.Eval}^a(\mathsf{pk}_{\mathsf{FHE}}, C, \tilde{c})$, where the answer a to the oracle query is hardwired in the evaluation circuit. Return $\rho \leftarrow \mathsf{FHE.PDec}(\mathsf{sk}_{\mathsf{FHE}}, c)$.

Return $(\tilde{C}, \mathsf{pk}_{\mathsf{FHE}})$ as the garbled circuit and $(\mathsf{sk}_{\mathsf{rGC}}, \mathsf{pk}_{\mathsf{FHE}})$ as the secret key.

$\mathsf{Encode}(\mathsf{sk}, x)$: On input the secret key $(\mathsf{sk}_{\mathsf{rGC}}, \mathsf{pk}_{\mathsf{FHE}})$ and an input x, sample two random strings $(r, s) \leftarrow_{\$} \{0,1\}^{2\lambda}$ and compute the ciphertext $\tilde{c} \leftarrow \mathsf{FHE.Enc}(\mathsf{pk}_{\mathsf{FHE}}, x; r)$, the oracle answer $a \leftarrow \mathsf{Sample}(\mathsf{pk}_{\mathsf{FHE}}, \tilde{c}; s)$ and the encoding $e \leftarrow \mathsf{GC.Encode}(\mathsf{sk}_{\mathsf{rGC}}, (x, r, s))$. The input encoding consists of the tuple $(\tilde{c}, a, e)$.

$\mathsf{Eval}(\tilde{C}, c)$: On input a garbled circuit $(\tilde{C}, \mathsf{pk}_{\mathsf{FHE}})$ and an encoding $(\tilde{c}, a, e)$, evaluate $\rho \leftarrow \mathsf{GC.Eval}(\tilde{C}, e)$ and compute $c \leftarrow \mathsf{FHE.Eval}^a(\mathsf{pk}_{\mathsf{FHE}}, C, \tilde{c})$. Return $\mathsf{FHE.Rec}(\rho, c)$.

To see why the scheme satisfies correctness, observe that

$$\begin{aligned}\mathsf{FHE.Rec}(\rho, c) &= \mathsf{FHE.Rec}(\mathsf{GC.Eval}(\tilde{C}, e), c)\\ &= \mathsf{FHE.Rec}(\mathsf{GC.Eval}(\tilde{C}, \mathsf{GC.Encode}(\mathsf{sk}_{\mathsf{rGC}}, (x, r, s))), c)\end{aligned}$$

where $(\tilde{C}, \mathsf{sk}_{\mathsf{rGC}}) \leftarrow \mathsf{GC.Garble}(1^\lambda, \Gamma)$. By definition of Γ and by the correctness of the garbling scheme, we have that

$$\mathsf{FHE.Rec}(\rho, c) = \mathsf{FHE.Rec}(\mathsf{FHE.PDec}(\mathsf{sk}_{\mathsf{FHE}}, c), c)$$

where

$$\begin{aligned}c &= \mathsf{FHE.Eval}^a(\mathsf{pk}_{\mathsf{FHE}}, C, \tilde{c})\\ &= \mathsf{FHE.Eval}^a(\mathsf{pk}_{\mathsf{FHE}}, C, \mathsf{FHE.Enc}(\mathsf{pk}_{\mathsf{FHE}}, x))\\ &= \mathsf{FHE.Eval}^a(\mathsf{pk}_{\mathsf{FHE}}, C, \mathsf{FHE.Enc}(\mathsf{pk}_{\mathsf{FHE}}, x))\\ &= \mathsf{FHE.Enc}(\mathsf{pk}_{\mathsf{FHE}}, C(x))\end{aligned}$$

and therefore

$$\mathsf{FHE.Rec}(\rho, c) = C(x).$$

We analyze the security of our scheme in the following theorem.

Theorem 6.5 (Reusable Security) *Let* FHE *be a split FHE scheme with simulatable decryption hints and let* rGC *be a reusably secure garbling scheme. Then the scheme as described above is reusably secure.*

Proof: The proof follows from a standard hybrid argument over the security of the reusable garbling scheme and by an invocation of the simulatability of the decryption hints of the split FHE scheme. □

All is left to be shown is that the scheme has rate-1 encodings (in the output length), for an appropriate instantiation of the underlying building blocks. The input encoding consists of three components:

(1) A ciphertext $\tilde{c}$ of the split FHE scheme encrypting the input x, whose size can be bounded by $\mathsf{poly}(\lambda, |x|)$.

(2) The answer a to a query to the Sample oracle, whose size is (for the choice of parameters discussed in, e.g., Section 4.3) bounded by $|y| + \mathsf{poly}(\lambda)$.

(3) The encoding e for the garbling of the circuit Γ. Note that the size of the output of Γ depends only on the security parameter (and in particular is independent of $|y|$), thus the size of e can be bounded by $\mathsf{poly}(\lambda, d, |x|)$, where d is the depth of the circuit C.

It follows that the total size of the input encoding is bounded by $\mathsf{poly}(\lambda, d, |x|) + |y|$.

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.

[AIK11] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. How to garble arithmetic circuits. In Rafail Ostrovsky, editor, *52nd Annual Symposium on Foundations of Computer Science*, pages 120–129, Palm Springs, CA, USA, October 22–25, 2011. IEEE Computer Society Press.

[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$^+$12] Gilad Asharov, Abhishek Jain, Adriana López-Alt, Eran Tromer, Vinod Vaikuntanathan, and Daniel Wichs. Multiparty computation with low communication, computation and interaction via threshold FHE. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 483–501, Cambridge, UK, April 15–19, 2012. 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.

[AP14] Jacob Alperin-Sheriff and Chris Peikert. Faster bootstrapping with polynomial error. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology – CRYPTO 2014, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 297–314, Santa Barbara, CA, USA, August 17–21, 2014. 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.

[BBKK17] Boaz Barak, Zvika Brakerski, Ilan Komargodski, and Pravesh K. Kothari. Limits on low-degree pseudorandom generators (or: Sum-of-squares meets program obfuscation). Cryptology ePrint Archive, Report 2017/312, 2017. `http://eprint.iacr.org/2017/312`.

[BDGM19] Zvika Brakerski, Nico Döttling, Sanjam Garg, and Giulio Malavolta. Leveraging linear decryption: Rate-1 fully-homomorphic encryption and time-lock puzzles. Cryptology ePrint Archive, Report 2019/720, 2019.

[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.

[BGK$^+$14] Boaz Barak, Sanjam Garg, Yael Tauman Kalai, Omer Paneth, and Amit Sahai. Protecting obfuscation against algebraic attacks. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology – EUROCRYPT 2014*, volume 8441 of *Lecture Notes in Computer Science*, pages 221–238, Copenhagen, Denmark, May 11–15, 2014. Springer, Heidelberg, Germany.

[BHHI10] Boaz Barak, Iftach Haitner, Dennis Hofheinz, and Yuval Ishai. Bounded key-dependent message security. In Henri Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 423–444, French Riviera, May 30 – June 3, 2010. Springer, Heidelberg, Germany.

[BHJ$^+$19] Boaz Barak, Samuel B. Hopkins, Aayush Jain, Pravesh Kothari, and Amit Sahai. Sum-of-squares meets program obfuscation, revisited. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2019, Part I*, volume 11476 of *Lecture Notes in Computer Science*, pages 226–250, Darmstadt, Germany, May 19–23, 2019. Springer, Heidelberg, Germany.

[BNPW16] Nir Bitansky, Ryo Nishimaki, Alain Passelègue, and Daniel Wichs. From cryptomania to obfustopia through secret-key functional encryption. 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 391–418, Beijing, China, October 31 – November 3, 2016. Springer, Heidelberg, Germany.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In Dorothy E. Denning, Raymond Pyle, Ravi Ganesan, Ravi S. Sandhu, and Victoria Ashby, editors, *ACM CCS 93: 1st Conference on Computer and Communications Security*, pages 62–73, Fairfax, Virginia, USA, November 3–5, 1993. ACM Press.

[BR14] Zvika Brakerski and Guy N. Rothblum. Virtual black-box obfuscation for all circuits via generic graded encoding. In Yehuda Lindell, editor, *TCC 2014: 11th Theory of Cryptography Conference*, volume 8349 of *Lecture Notes in Computer Science*, pages 1–25, San Diego, CA, USA, February 24–26, 2014. Springer, Heidelberg, Germany.

[BV14] Zvika Brakerski and Vinod Vaikuntanathan. Lattice-based FHE as secure as PKE. In Moni Naor, editor, *ITCS 2014: 5th Conference on Innovations in Theoretical Computer Science*, pages 1–12, Princeton, NJ, USA, January 12–14, 2014. Association for Computing Machinery.

[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.

[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.

[FS87] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *Advances in Cryptology – CRYPTO'86*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194, Santa Barbara, CA, USA, August 1987. Springer, Heidelberg, Germany.

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In Michael Mitzenmacher, editor, *41st Annual ACM Symposium on Theory of Computing*, pages 169–178, Bethesda, MD, USA, May 31 – June 2, 2009. ACM Press.

[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.

[GHV10] Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. i-Hop homomorphic encryption and rerandomizable Yao circuits. In Tal Rabin, editor, *Advances in Cryptology – CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 155–172, Santa Barbara, CA, USA, August 15–19, 2010. Springer, Heidelberg, Germany.

[GJK18] Craig Gentry, Charanjit S. Jutla, and Daniel Kane. Obfuscation using tensor products. Cryptology ePrint Archive, Report 2018/756, 2018.

[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.

[GMM17a] Sanjam Garg, Mohammad Mahmoody, and Ameer Mohammed. Lower bounds on obfuscation from all-or-nothing encryption primitives. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017, Part I*, volume 10401 of *Lecture Notes in Computer Science*, pages 661–695, Santa Barbara, CA, USA, August 20–24, 2017. Springer, Heidelberg, Germany.

[GMM17b] Sanjam Garg, Mohammad Mahmoody, and Ameer Mohammed. When does functional encryption imply obfuscation? In Yael Kalai and Leonid Reyzin, editors, *TCC 2017: 15th Theory of Cryptography Conference, Part I*, volume 10677 of *Lecture Notes in Computer Science*, pages 82–115, Baltimore, MD, USA, November 12–15, 2017. 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.

[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.

[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.

[LPST16] 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.

[LT17a] Huijia Lin and Stefano Tessaro. Indistinguishability obfuscation from bilinear maps and block-wise local prgs. Cryptology ePrint Archive, Report 2017/250, Version 20170320:142653, 2017.

[LT17b] 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.

[LV17] Alex Lombardi and Vinod Vaikuntanathan. Limits on the locality of pseudorandom generators and applications to indistinguishability obfuscation. In Yael Kalai and Leonid Reyzin, editors, *TCC 2017: 15th Theory of Cryptography Conference, Part I*, volume 10677 of *Lecture Notes in Computer Science*, pages 119–137, Baltimore, MD, USA, November 12–15, 2017. Springer, Heidelberg, Germany.

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

[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.

[MT19] Giulio Malavolta and Sri Aravinda Krishnan Thyagarajan. Homomorphic time-lock puzzles and applications. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019, Part I*, volume 11692 of *Lecture Notes in Computer Science*, pages 620–649, Santa Barbara, CA, USA, August 18–22, 2019. Springer, Heidelberg, Germany.

[Pai99] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In Jacques Stern, editor, *Advances in Cryptology – EUROCRYPT'99*, volume 1592 of *Lecture Notes in Computer Science*, pages 223–238, Prague, Czech Republic, May 2–6, 1999. 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.

[Yao86] Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *27th Annual Symposium on Foundations of Computer Science*, pages 162–167, Toronto, Ontario, Canada, October 27–29, 1986. IEEE Computer Society Press.