Deniable Fully Homomorphic Encryption from Learning With Errors

Shweta Agrawal*, Shafi Goldwasser**, and Saleet Mossel***

Abstract. We define and construct *Deniable Fully Homomorphic Encryption* based on the Learning With Errors (LWE) polynomial hardness assumption. Deniable FHE enables storing encrypted data in the cloud to be processed securely without decryption, maintaining deniability of the encrypted data, as well the prevention of vote-buying in electronic voting schemes where encrypted votes can be tallied without decryption.

Our constructions achieve *compactness* independently of the level of deniability- both the size of the public key and the size of the ciphertexts are bounded by a fixed polynomial, independent of the detection probability achieved by the scheme. This is in contrast to all previous constructions of deniable encryption schemes (even without requiring homomorphisms) which are based on polynomial hardness assumptions, originating with the seminal work of Canetti, Dwork, Naor and Ostrovsky (CRYPTO 1997) in which the ciphertext size grows with the inverse of the detection probability. Canetti *et al.* argued that this dependence "seems inherent", but our constructions illustrate this is not the case. We note that the Sahai-Waters (STOC 2014) construction of deniable encryption from indistinguishability obfuscation achieves compactness and can be easily modified to achieve deniable FHE as well, but it requires multiple, stronger sub-exponential hardness assumptions, which are furthermore not post-quantum secure. In contrast, our constructions rely only on the LWE polynomial hardness assumption, as currently required for FHE even without deniability.

The running time of our encryption algorithm depends on the inverse of the detection probability, thus the scheme falls short of achieving simultaneously compactness, negligible deniability probability *and* polynomial encryption time. Yet, we believe that achieving compactness is a fundamental step on the way to achieving all properties simultaneously as has been the historical journey for other primitives such as functional encryption. Our constructions support large message spaces, whereas previous constructions were bit by bit, and can be run in online-offline model of encryption, where the bulk of computation is independent of the message and may be performed in an offline pre-processing phase. This results in an efficient online phase whose running time is independent of the detection probability. At the heart of our constructions is a new way to use bootstrapping to obliviously generate FHE ciphertexts so that it supports faking under coercion.

* IIT Madras, shweta.a@cse.iitm.ac.in.

** Simons Institute of TOC at UC Berkeley, shafi.goldwasser@gmail.com.

*** MIT, saleet@mit.edu

1 Introduction

Deniable (public-key) encryption, which was introduced in a seminal work by Canetti, Dwork, Naor and Ostrovsky (CRYPTO 1997) [13], is a seemingly paradoxical primitive that enables a user, who may be coerced to reveal the plaintexts corresponding to her public ciphertexts, to successfully lie about which messages she encrypted.

In particular, suppose Alice encrypted a message m with ciphertext ct which she deposits in the cloud for the purpose of cloud computing, and is later forced by the government to reveal the randomness she used and the message encrypted. Deniable encryption allows her to chose a different message m' at coercion time and reveal *fake* random coins, which convincingly explain ct as the encryption of m'. Clearly, deniability is a property which may be highly desirable when one uses a public resource such as cloud computing which expose him to possible coercion. Another use case is preventing vote buying in electronic elections: if the voter encrypts her vote using deniable encryption, then she can claim she encrypted an alternate message when forced to reveal her vote, deeming vote selling ineffective and encouraging honest voting since the voter cannot be forced to reveal her choice.

In this work, we introduce the notion of deniable *fully homomorphic encryption* (FHE) and provide the first constructions based on the Learning With Errors polynomial hardness assumption. In deniable FHE, the encryptor can produce ciphertexts that can be opened to fake messages under coercion, and additionally support fully homomorphic computations and achieve security as in (by now) classical FHE. We emphasize that for all the applications of deniable public key encryption mentioned above, the capability of homomorphism is an important implicit requirement – indeed, several modern e-voting protocols use FHE [15, 27], and present-day encrypted data is often stored on a cloud server which assists the data owner with computing "blind-folded" via FHE [21].

We proceed to describe important prior work before we proceeding to describe our results in detail.

1.1 Prior Work on deniability.

Canetti et al. (CDNO) [13] provided elegant constructions of deniable encryption based on the construct of so called "translucent sets", which in turn can be constructed from trapdoor permutations. A major disadvantage of the CDNO construction was lack of compactness – the ciphertext size grows with the inverse of the detection probability achieved by the scheme. Furthermore, it encodes large messages bit by bit, where the ciphertext for each bit grows inversely with the detection probability. CDNO provided a lower bound that shows that their construction is in some sense optimal. They identified a structural property of encryption, which they term as *separability* and argued that as long as a construction is separable, the dependence of the ciphertext size with the inverse of the detection probability "seems inherent"[13].

A significant step forward in our understanding of deniable encryption and compactness was achieved via the work of Sahai and Waters in 2014 [29] which provided the first construction achieving negligible deniability assuming indistinguishability obfuscation (iO) and one way functions. However, iO seems to be an inherently sub-exponential assumption [19, 20], and while exciting as a feasibility result, does not provide a satisfying solution to the question of deniable encryption from standard polynomial hardness assumptions.

CDNO also suggested the notion of *weak* deniability where the encryptor can lie not only about the random coins used to generate the ciphertext, but also the *algorithm* used to encrypt the message and the notion of *receiver* deniability, where the receiver can also produce a fake secret key that decrypts the message to an alternate one. In the weak model, [13] showed that compact public key and ciphertext as well as negligible deniability are possible. However, whether the weak model is meaningful for practical applications has been the subject of some debate – as discussed in [28], a common objection to the weak model is "since there are alternative deniable algorithms that are strictly more powerful than the normal ones, why would anyone ever run the normal algorithms? And given this situation, why would a coercer ever accept a transcript corresponding to the normal algorithms?". We refer the reader to [28] for a detailed discussion.

Other extensions to deniable encryption were also explored – O'Neill, Peikert and Waters [28] provided the first constructions of non-interactive *bi*-deniable encryption schemes where both the sender and the receiver can fake simultaneously as well as the first construction of identity based bi-deniable encryption. Apon, Fan and Liu [4] extended their results to provide the first construction deniable *attribute based encryption*. However, in the full model, both works [28, 4] inherit the detection probability of CDNO, which is inverse polynomial. Additional prior work not directly related to the current work is discussed in Section 1.5.

Summarizing, barring the iO based construction which seems to require a sub-exponential hardness assumption, all proposals for (fully) sender deniable encryption schemes from standard assumptions suffer from ciphertext size that is inversely proportional to the detection probability. This implies a prohibitively large blow on efficiency. For a primitive as fundamental and interesting as deniable encryption, this state of affairs is very dissatisfying.

1.2 Our Results.

In this work, we introduce the notion of deniable *fully homomorphic encryption* (FHE) and provide the first constructions of deniable FHE based on the Learning With Errors (LWE) assumption. Our constructions enjoy deniability compactness - the public key as well as the ciphertext of our schemes have size that can be bounded by a fixed polynomial, and are, in particular, independent of the level of deniability (or detection probability) achieved by the scheme. Our constructions support large messages paces, whereas all prior constructions encoded large messages bit by bit. On the down side, our encryption time depends on the inverse of the detection probability, thus the scheme falls short of achieving

simultaneously compactness, negligible deniability and polynomial encryption time. Luckily, the scheme can be run in online-offline model of encryption, where the bulk of computation, which grows with the inverse of the detection probability, is independent of the message and may be performed in an offline pre-processing phase. The running time of the online phase, is independent of the detection probability.

We believe that achieving compact ciphertext even at the price of large encryption time is a fundamental step forward – indeed, note that for the related primitive of functional encryption (FE), compact ciphertext was later found to imply compact running time [26] by additionally assuming LWE via the "succinct" FE of Goldwasser et al. [24]. While this implication does not hold true for our work at present, it is a tantalizing possibility for future work.

We now proceed to on expound on the particulars of our results.

Deniable FHE. A (public key, sender) *deniable fully homomorphic encryption* consists of a tuple of algorithms $\mathsf{DFhe} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec}, \mathsf{Fake})$ where Gen, Enc and Dec are the standard key-generation, encryption and decryption algorithms, Eval is an algorithm that takes as input the public key, a circuit $\mathcal{C}$ and a tuple of ciphertexts $\mathsf{ct}_1, \ldots, \mathsf{ct}_n$ encrypting $x_1, \ldots, x_n$ respectively, and outputs a ciphertext ct^* which encrypts $\mathcal{C}(x_1, \ldots, x_n)$, and Fake is a faking algorithm, which takes as input the public key, an original message m, randomness r, and a fake message m^* and outputs a fake randomness r^* so that the encryption of message m using randomness r produces the same ciphertext as the encryption of message m^* using randomness r^*, i.e. $\mathsf{Enc}(\mathsf{pk}, m; r) = \mathsf{Enc}(\mathsf{pk}, m^*; r^*)$. The detection probability is the probability with which an adversary can distinguish r from r^*, and we denote it by $1/\delta = 1/\delta(\lambda)$ where λ is the security parameter. Our notion of deniable FHE is formalized in Definition 2.8.

We naturally extend this definition to the *weak* model (Definition 2.11) – a weakly deniable FHE is defined as $\mathsf{wDFhe} = (\mathsf{Gen}, \mathsf{DEnc}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec}, \mathsf{Fake})$ which is distinct from "fully" deniable FHE in that there are two distinct algorithms for encryption, namely Enc and DEnc. Here, as in [13], leveraging the additional secret "deniable" encryption algorithm DEnc, allows for better constructions as discussed below (in particular, those that achieve negligible deniability in polynomial time).

In more detail, Enc is an "honest" encryption algorithm and is used by the encryptor when it does *not* wish to fake a ciphertext, and DEnc is a "deniable" encryption algorithm, which is used when the encryptor wishes to retain the ability of faking a ciphertext in the future. Let us say the encryptor wishes to compute an encryption of m which it may later want to explain differently. Then it produces a ciphertext ct^* by running the algorithm DEnc with message m using randomness r. To explain ct^* as encrypting an arbitrary fake message m^* at a later time, the encryptor produces random coins r^* using the Fake algorithm, so that the ciphertext output by the *honest* encryption algorithm Enc on m^* using r^* equals the ciphertext ct^* which was produced using the deniable encryption algorithm DEnc, i.e. $\mathsf{DEnc}(\mathsf{pk}, m; r) = \mathsf{Enc}(\mathsf{pk}, m^*; r^*)$.

Next, we describe our constructions. We provide:

1. A *weakly* deniable FHE scheme for bits with negligible detection probability (Section 4.1). We extend this scheme to support larger (polynomial sized) message spaces (Section 5).
2. A *fully* deniable FHE scheme for bits with inverse polynomial detection probability (Section 4.2). We also extend this scheme to support larger (polynomial sized) message spaces (see the full version[1]). Both our fully deniable FHE schemes have compact public key and ciphertext, i.e. with size independent of the detection probability, but with encryption running time that grows with the inverse of the detection probability.
3. *Plan-ahead* deniable FHE schemes which support exponentially large message spaces (see the full version[1]). Plan-ahead deniable encryption [13] requires the encryptor to choose all (polynomially many) possible fake messages at the time of encryption. Later, when the encryptor desires to explain a ciphertext, it can only provide convincing fake randomness for one of the fake messages chosen during encryption.

Fake Evaluation. We note that our notions of deniable FHE also allow, in some cases, to explain *evaluated* ciphertexts as encoding a fake message. For instance, in the case that Eval is a deterministic algorithm, suppose that ct^* was computed by homomorphically evaluating a polynomial sized circuit $\mathcal{C}$ on ciphertexts $\mathsf{ct}_1, \ldots, \mathsf{ct}_n$ which encode messages $x_1, \ldots, x_n$ respectively. Suppose an encryptor wishes to explain ct^* as an encryption of an arbitrary message $m^* \neq \mathcal{C}(x_1, \ldots, x_n)$, and $\mathcal{C}$ supports inversion, i.e. given a value m^*, it is possible to efficiently sample $x'_1, \ldots x'_n$ such that $\mathcal{C}(x'_1, \ldots, x'_n) = m^*$. Then, the encryptor may simply explain ct_i as an encryption of x'_i for $i \in [n]$ and exhibit that the homomorphic evaluation procedure for $\mathcal{C}$ results in ct^*. This convinces the adversary that ct^* encodes m^*, as desired. We note that for several applications of interest, the circuit $\mathcal{C}$ can indeed be invertible – for instance, $\mathcal{C}$ may represent the vote counting circuit, which is simply addition and hence easily invertible.

On the Underlying Assumptions. We remark that the Sahai-Waters construction of public key deniable encryption from indistinguishability obfuscation (iO) [29] can be modified in a natural way to construct deniable fully homomorphic encryption. This provides an appealing feasibility result for deniable fully homomorphic encryption with negligible deniability, but rely on the strong hammer of indistinguishability obfuscation. While (concurrent) exciting recent work [25] has based indistinguishability obfuscation on well-founded assumptions, this construction relies on the subexponential hardness of four different assumptions, including assumptions on bilinear maps which are known to be insecure in the post-quantum regime. It is also well known that existing reductions to indistinguishability obfuscation [29] run into subexponential barrier due to the number of hybrids used in the security reductions – this results a subexponential assumption, please see [20] for a discussion.

The focus of our work is to rely on *minimal* assumptions. The primitive of levelled (respectively, pure) fully homomorphic encryption may be based

on the polynomial hardness of the Learning With Errors (respectively, with circular security) assumption, with polynomial approximation factors [12]. Our constructions show that we can achieve (polynomially) deniable FHE without making any additional assumptions.

Compact Deniable PKE from FHE. Homomorphism aside, as discussed above, our construction implies, as a special case, a *compact* deniable public key encryption scheme, where the size of the public key and ciphertext are independent of the detection probability, which can be made an arbitrarily small inverse polynomial. However, as discussed above, the running time of our encryption algorithm *does* grow linearly with the inverse of the detection probability. This dependence again seems inherent, since our constructions can be shown to be separable in the sense of CDNO and hence subject to the lower bound (see the full version [1]). We discuss in Section 1.4 the technical barriers in circumventing this lower bound from non-obfuscation assumptions.

Online-Offline Encryption. Our constructions of deniable FHE also enjoy a desirable *online-offline* property, which allows the encryptor to do the bulk of the work in an offline phase that is independent of the message to be encrypted. In more detail, our encryption algorithm can be divided into two parts – an offline, message independent part which runs in time $O(\delta)$ (recall that $\frac{1}{\delta}$ is the detection probability), and an online phase which is efficient and independent of δ. We believe this feature makes these schemes especially attractive for practice since it mitigates the disadvantage of the large running time of encryption.

1.3 Our Techniques.

The primary technical challenge in (full) deniable encryption is satisfying the many constraints imposed by the faking algorithm: the adversary knows the encryption algorithm and must be shown correctly distributed randomness that explains a given challenge ciphertext to a fake message. Excepting the construction based on obfuscation [29], all prior work addressed this challenge by setting the ciphertext to be a long sequence of elements that are either random or pseudorandom, and encoding the message bit in the parity of the number of pseudorandom elements. To fake, the encryptor pretends that one of the pseudorandom elements is in fact random, thus flipping the parity of the number of pseudorandom elements, and hence the encoded message. To construct a deniable fully homomorphic encryption scheme, the first challenge that arises is that an FHE ciphertext is highly structured, and this is necessary if it has to support homomorphic evaluation. Moreover, valid FHE ciphertexts are sparse in the ciphertext space, so randomly sampled elements are unlikely to be well-formed ciphertexts. Hence, if the encryptor for deniable FHE constructs all components of the ciphertext by running the FHE encryption algorithm i.e. $\mathsf{Fhe.Enc}(\mathsf{pk}, m; r)$, then it is forced to open the FHE ciphertexts to provide r honestly – the structure of ciphertexts does not support lying about any of the encoded bits. The encryptor is thus faced

with the incongruous task of producing highly structured ciphertexts without running the FHE encryption algorithm.

The Magic of Bootstrapping. To overcome this hurdle, we leverage the clever idea of "bootstrapping" proposed by Gentry [21]. At a high level, bootstrapping is the procedure of homomorphically computing the decryption circuit of a given scheme, say Fhe, on a ciphertext of the same scheme, using an encryption of the scheme's secret key, denoted by $\mathsf{ct}_{\mathsf{sk}}$. This procedure assumes circular security, namely that semantic security of Fhe holds even when the adversary is provided an encryption of the scheme's own secret key. The original motivation for bootstrapping was to reduce the "noise" level in FHE ciphertext – since the decryption circuit of an FHE scheme is quite shallow, running the decryption circuit homomorphically on some FHE ciphertext ct using the encryption of the FHE secret key $\mathsf{ct}_{\mathsf{sk}}$, removes the noise contained in ct via decryption, and the noise in output ciphertext ct' can be bound depending on the depth of the decryption circuit and the noise in $\mathsf{ct}_{\mathsf{sk}}$. To date, all constructions of "pure" FHE, namely, FHE that supports unbounded depth circuits, must assume circular security of the underlying "somewhat homomomorphic" encryption scheme, and hence of the underlying Learning With Errors (LWE) assumption. Since circular security is required anyway for the construction of pure FHE, we assume it in our construction of deniable (pure) FHE, and in the exposition below for simplicity. For the case of "levelled" FHE, which assumes a bound on the depth of supported circuits, and which can be built from standard LWE, this requirement can be removed as discussed in the full version [1].

Aside from noise reduction, an additional attractive feature of bootstrapping is that it suggests a way to *obliviously* generate FHE ciphertexts. Suppose our FHE scheme's decryption algorithm always outputs a valid message regardless of whether the ciphertext is well-formed or not. Then, by running the bootstrapping procedure on a random element from the ciphertext space, we obtain a well formed, valid FHE ciphertext for an unknown bit, by correctness of FHE evaluation. Moreover, if we run the bootstrapping procedure on a valid FHE ciphertext of any bit, the ciphertext output by bootstrapping still encodes the same bit, by correctness of FHE decryption and evaluation. If FHE ciphertexts are indistinguishable from random (which they usually are), then the encryptor may cheat about which of the two types of inputs was provided to the bootstrapping procedure and thereby lie about the encoded bit in the bootstrapped ciphertext.

While this feels like progress, it is still unclear how to encrypt a single bit of one's choosing using obliviously generated ciphertexts of unknown bits and honestly generated ciphertexts of known bits.

Deniable FHE in the Weak Model. As a warm-up, let us consider the weak model of deniability, where the encryptor can lie not only about the randomness used in encryption but also the algorithm used. Let us suppose for the moment that we may engineer the bootstrapping procedure so that an obliviously generated FHE ciphertext is biased and encodes the bit 0 with overwhelming probability (we

will weaken this assumption later). Then, an approach to encrypt in the weak model is as follows.

Let the bootstrapping procedure be denoted by boot. In the honest mode, the encryptor encrypts bit 0 by choosing R_1 and R_2 randomly from the ciphertext space, converting these to well formed FHE ciphertexts via the bootstrapping procedure, and finally computing the homomorphic XOR operation (denoted by $\oplus_2$) on these FHE ciphertexts. Thus, we have:

$$\mathsf{ct}_0 = \mathsf{boot}(R_1) \oplus_2 \mathsf{boot}(R_2)$$

Since we assumed that random elements are bootstrapped to encode 0 with overwhelming probability, the ciphertext ct_0 encodes 0 due to correctness of the FHE evaluation procedure. To encrypt bit 1, the encryptor chooses R_3 randomly from the ciphertext space, and computes R_4 as an honest encryption of 1 using the FHE encryption algorithm. It then sets:

$$\mathsf{ct}_1 = \mathsf{boot}(R_3) \oplus_2 \mathsf{boot}(R_4)$$

It is easy to see that correctness is preserved by the same arguments as above.

In the deniable or fake encryption algorithm, the sender changes the way it encrypts 0. Instead of choosing R_1 and R_2 uniformly at random, it now computes both R_1 and R_2 as well formed FHE ciphertexts of 1. Bootstrapping preserves the message bit and homomorphic evaluation of addition modulo 2 ensures that ct_0 is a valid encryption of 0. The bit 1 is encrypted as before. However, if asked to explain, the encryptor can pretend that ct_0 is in fact an encryption of 1 by claiming that R_1 is chosen uniformly and by explaining R_2 as an encryption of 1. Since R_1 is an FHE ciphertext, the adversary cannot tell the difference as long as FHE ciphertext is pseudorandom. Similarly, if asked to explain ct_1 as an encryption of 0, she explains R_4 as a randomly chosen element in the ciphertext space. Thus, we obtain a construction of weakly deniable FHE for bits which achieves negligible detection probability. For more details, please see Section 4.1.

Deniable FHE in the Full Model. In the full model, the encryptor is not allowed to cheat about the algorithm it used for encryption, hence we may not take advantage of different ways of sampling randomness in the real and deniable encryption algorithms – there is only one encryption algorithm. In this model, we obtain FHE with polynomial deniability but with compact public key and ciphertext, that is, the size of the public key and ciphertext are independent of the detection probability. We proceed to describe the main ideas in the construction.

Let δ be the inverse of the desired detection probability. To encrypt a bit b, the encryptor samples uniform random bits $x_1, \ldots, x_\delta$ such that $\sum_{i \in [\delta]} x_i = b \pmod 2$. It then computes δ elements $R_1, \ldots, R_\delta$ of which, R_i is computed as an FHE encryption of 1 when $x_i = 1$, and R_i is sampled uniformly at random when $x_i = 0$. Finally, it outputs

$$\mathsf{ct} = \mathsf{boot}(R_1) \oplus_2 \mathsf{boot}(R_2) \oplus_2 \ \ldots \ \oplus_2 \mathsf{boot}(R_\delta)$$

To fake, it samples a random $j \in [\delta]$ such that $x_j = 1$, sets $x_j^* = 0$, and $x_i^* = x_i$ for every $i \neq j, i \in [\delta]$. It pretends that R_j is chosen uniformly at random, implying that $\mathsf{boot}(R_j)$ encodes 0 with overwhelming probability. It is easy to see that this flips the message bit that was chosen during encryption. Moreover, the statistical distance between honest randomness and fake randomness is $O(\frac{1}{\delta})$ and we achieve polynomial deniability, so long as the encryption time is polynomial. Please see Section 4.2 for more details.

Special FHE. The above informal description brushes several important details under the rug. For instance, we assumed various properties about the underlying FHE scheme which are not true in general. The most problematic assumption we made is that the FHE bootstrapping procedure can be engineered so that it outputs an encryption of 0 for a random input with overwhelming probability.

Some thought reveals that existing FHE schemes do not satisfy this property. Fortunately however, we show that some constructions can be modified to do so. For concreteness, we describe how to modify the FHE scheme by Brakerski, Gentry and Vaikuntanathan [10] to get the "special FHE" that we require. At a high level, decryption in the BGV cryptosystem is a two step procedure, where the first step computes the inner product of the ciphertext and the secret key over the ambient ring, and the second step computes the least significant bit of the result, which is then output. One can check that for any well formed ciphertext in this scheme, regardless of whether it encodes 0 or 1, the first step of the decryption procedure always yields a "small" element. On the other hand, for a random element in the ciphertext space, the first step of decryption yields a random element, i.e. it is small with low probability. Thus, we may modify the BGV decryption algorithm so that after computing the inner product in the first step, it checks whether the output is small, and outputs 0 if not. This does not change decryption for well formed ciphertexts but by a suitable setting of parameters, it biases the output of decryption to 0 for random inputs. In fact, we can make do with a weaker requirement on bias, namely that the bootstrapping procedure outputs an encryption of 0 for a random input with only non-negligible (not overwhelming) probability. However this makes the scheme more complicated, so we do not discuss it here. Please see the full version [1] for details. We also require some additional properties from our special FHE, which we define and establish in Section 3.

Large Messages. In all prior constructions of deniable encryption, larger messages were encoded bit by bit, where the ciphertext for a single bit is itself quite substantial ($O(\delta)$) as discussed above. To further improve efficiency, we again leverage the power of FHE. This enables our schemes to support large message spaces natively, thereby inheriting the significant advances in FHE schemes with large information rate [30, 10, 9, 22], and bringing deniable FHE closer to practice.

Let $\mathcal{M}$ be the message space of an FHE scheme Fhe such that $|\mathcal{M}| = \mathrm{poly}(\lambda)$. Further, let us assume that Fhe satisfies the special properties discussed above (formalized in Section 3). Then, to compute a ciphertext for a message $m_k \in \mathcal{M}$, we express m_k as the output of a "selector" function which computes the inner

product of the k^{th} unit vector with a vector of all messages in $\mathcal{M}$. In more detail, we express

$$m_k = 1 \cdot m_k + \sum_{m_i \in \mathcal{M}, i \neq k} 0 \cdot m_i$$

Here, the bits 0 or 1 are referred to as "selector" bits for obvious reasons. Our main observation is that the deniable encryption scheme for bits can now be used to add deniability to ciphertexts of selector bits and thereby to the overall ciphertext.

In more detail, assume that the sender selects message m_k at the time of encryption. To compute a ciphertext of m_k, she computes FHE ciphertexts ct_i for all $m_i \in \mathcal{M}$ and selector bit ciphertexts $\mathsf{ct}_i^{\mathsf{sel}}$ for $i \in [|\mathcal{M}|]$ where $\mathsf{ct}_i^{\mathsf{sel}}$ encodes 0 if $i \neq k$ and 1 otherwise. We use deniable encryption to compute the ciphertexts of selector bits as described above; thus, each selector bit is computed using multiple elements $\{R_i\}$ where $i \in [\delta]$. She then homomorphically computes the selector function described above to obtain a ciphertext ct^* encoding m_k. Under coercion, she may explain ct^* as encoding of any message m_i, even for $i \neq k$, by explaining the corresponding selector bits differently, i.e. by explaining $\mathsf{ct}_i^{\mathsf{sel}}$ as an encryption of 1 and $\mathsf{ct}_k^{\mathsf{sel}}$ as an encryption of 0.

We note that the above description is oversimplified and glosses over many technical details – for instance, the deniable FHE scheme for bits assumes that decryption of a random element in the ciphertext space is biased to 0 with overwhelming probability, which is no longer the case for FHE with large message spaces. However, this and other issues can be addressed, and we get schemes in both the weak and full models – please see Section 5 and the full version [1] for details.

Plan-Ahead Deniability. Plan-ahead deniable encryption [13] requires the sender to choose all possible fake messages at the time of encryption itself. For plan-ahead fully homomorphic encryption, it becomes possible to instantiate the underlying FHE to have super-polynomial message space. Intuitively, without the plan-ahead restriction, the construction discussed above fails for exponentially large message spaces, since it is not possible to "select" between exponentially many options in polynomial time. However, if the number of possible fake messages is fixed to some polynomial in advance, as is the case for plan-ahead deniability, then the same construction as above works, as long as we can establish the "special" properties of the FHE. We discuss how this can be achieved, please see the full version [1] for details.

Online-Offline Encryption. We now describe how our encryption algorithms lend themselves naturally to the online-offline model, where a bulk of the computation required for encryption is performed before the message is available. Consider the encryption algorithm for bits in the full model. Observe that sampling δ random bits $x_1, \ldots, x_\delta$ such that $\sum_{i \in [\delta]} x_i = b \pmod 2$ is the same as sampling $\delta - 1$ random bits $x_1, \ldots, x_{\delta-1}$ and setting $x_\delta = b + \sum_{i \in [\delta-1]} x_i \pmod 2$. In the offline phase, we may select $\delta - 1$ bits $x_1, \ldots, x_{\delta-1}$ at random as well as the

corresponding $\delta - 1$ elements R_i based on the bit x_i as specified in the encryption algorithm. Next, we homomorphically evaluate the bootstrapping circuit on the $\delta - 1$ random elements, i.e. $\mathsf{boot}(R_i)$ for $i \in [\delta - 1]$ and then compute:

$$\mathsf{ct}_{\mathsf{offline}} = \mathsf{boot}(R_1) \oplus_2 \mathsf{boot}(R_2) \oplus_2 \ldots \oplus_2 \mathsf{boot}(R_{\delta-1}).$$

Now, in the online phase we can simply select the last bit and corresponding randomness R_δ according to the message b being encrypted, compute the homomorphic bootstrapping algorithm on R_δ, and evaluate the homomorphic addition mod 2 as: $\mathsf{ct} = \mathsf{ct}_{\mathsf{offline}} \oplus_2 \mathsf{boot}(R_\delta)$. Thus, the online encryption time is independent of δ.

Next, consider the encryption scheme for large message spaces. Even here, note that the dependence of the encryption running time on the detection probability comes from the construction of selector bits. Since the construction of any ciphertext involves $|\mathcal{M}| - 1$ encryptions of 0 and a single encryption of 1, the encryptions of these selector bits can be computed in an offline pre-processing phase. The encryptions of all possible messages in the message space can also be performed offline. Then, in the online phase, given message m_k, the encryptor needs only to perform the homomorphic evaluation of the selector function to compute the final ciphertext. This leads to an online encryption time which grows with $|\mathcal{M}|$ but not with the inverse of the detection probability.

The online processing time may be optimized further as follows – now, additionally in the offline phase, let the encryptor perform the homomorphic evaluation of the selector function with *all* the selector bits set to 0, i.e. $\sum_{m_i \in \mathcal{M}} 0 \cdot m_i$. It stores the ciphertexts for all possible messages $m \in \mathcal{M}$, the ciphertexts of the computed selector bits which are set to 0 as well as a ciphertext ct^1 for an extra selector bit which is set to 1. In the online phase, when m_k is known, it subtracts the "wrong" term $\mathsf{ct}_k^0 \cdot \mathsf{ct}_k$ and adds the term $\mathsf{ct}^1 \cdot \mathsf{ct}_k$ to the evaluated ciphertext to obtain the correct ciphertext. Thus, the online phase can be performed in time independent of both $|\mathcal{M}|$ as well as δ.

Removing the Circularity Assumption for Levelled FHE. Above, our usage of the bootstrapping procedure implies the assumption of circular secure homomorphic encryption, hence circular secure LWE. Since circular security is required anyway for all known constructions of pure FHE (we refer the reader to [8] for a discussion), this assumption currently comes "for free" in the construction of deniable pure FHE. However, for levelled FHE, which only supports circuits of bounded depth and can be constructed from standard LWE [11, 10, 23], the assumption of circularity is not implied. In this setting, our construction can be easily adapted to make do without the circularity assumption, as observed by [3]. The idea is simple – instead of assuming that the encryption of a scheme's secret key under it's own public key is secure, we can instead rely on two encryption schemes and assume that the secret key of first scheme sk_1 can be securely encrypted using the public key of the second scheme pk_2. Let us denote this ciphertext by $\mathsf{ct}_{\mathsf{sk}_1}$. Now, the obliviously sampled ciphertexts can be seen as encrypted under pk_1 and the ciphertext $\mathsf{ct}_{\mathsf{sk}_1}$ may be used to translate these to valid ciphertexts under

pk_2 via a variant of the bootstrapping procedure discussed above. In more detail, the modified bootstrapping procedure computes the homomomorphic evaluation procedure of the second scheme using as inputs the ciphertext $\mathsf{ct}_{\mathsf{sk}_1}$ and the decryption circuit of the first scheme to produce valid ciphertexts under the second scheme. We refer the reader to the full version [1] for more details.

1.4 Perspective: FHE as a Tool

As discussed above, bootstrapping enables us to obliviously sample FHE ciphertexts, and homomorphic evaluation enables us to "compactify" the final ciphertext – this makes FHE a useful tool even in the context of deniable *public key encryption* (PKE). One of the main insights of our work is that *evaluation* compactness in FHE can be leveraged to achieve *deniability* compactness in PKE. All constructions of non-interactive sender deniable encryption in the full model known from 1997 to date (excepting the one based on iO [29]), must provide multiple elements in the ciphertext, both pseudorandom and random, and encode the message bit in the parity of the number of pseudorandom elements leading to ciphertext size that grows inversely with detection probability. We can avoid this dependence using FHE.

Can FHE also help achieve compact runtime of encryption? If so, this would lead to negligibly deniable PKE from LWE, resolving the long-standing open problem of deniable PKE from a standard, polynomial hardness assumption, with the post-quantum advantage as the "icing on the cake". While this exciting possibility cannot be ruled out, a thorny technical barrier that arises is the hardness of inverting the bootstrapping procedure. Intuitively, deniable encryption requires *invertible biased oblivious sampling* – the encryption procedure must obliviously sample a ciphertext (biased to encoding 0, say) and the faking procedure must invert a given ciphertext, encoding either 0 or 1, to produce a well distributed randomness. In hindsight, even the iO based construction of Sahai and Waters [29] can be viewed as a construction of invertible oblivious sampling – indeed, similar techniques have been used to construct invertible sampling [17].

Using our current techniques, bootstrapping enables us to perform oblivious sampling, but not inversion. Due to this limitation, we are restricted to cheating only in one direction – we can pretend that a ciphertext of 1 encodes 0 but not the other way around. This leads to the attack discussed in the full version [1], which curtails the scheme to polynomial deniability. However if, given $y = \mathsf{boot}(R)$, we could compute well-distributed R' such that $\mathsf{boot}(R') = y \oplus_2 1$, where $\oplus_2 1$ denotes homomorphic XOR of the bit 1, then we would gain the ability to cheat in both directions and obtain negligibly deniable PKE. We remark that while boot is a one way function, infeasibility of inversion does not apply since we have potentially useful side information about the preimage – we must find the preimage of $y \oplus_2 1$ and know the preimage to y. Unfortunately, we currently do not know how to leverage this information. Nevertheless, we view ciphertext compactness as a useful stepping stone to full runtime compactness from LWE, and hope it can lead to progress towards a full solution. Please see the full

version [1] for a more in-depth discussion on the barriers in achieving negligible deniability.

In the full version[1], we discuss the notion of receiver deniable FHE.

1.5 Other Related Work

De Caro, Iovino and O'Neill [18] studied the notion of *receiver deniable functional encryption*, but instantiating these constructions requires the assumption of full fledged functional encryption, which in turn is known to imply indistinguishability obfuscation (iO) [2, 6].

Aside from work extending the functionality of deniable encryption, there was also progress in lower bounds – for receiver deniability, [5] showed that a non-interactive public-key scheme having key size δ can be fully receiver-deniable only with non-negligible $\Omega(\frac{1}{\delta})$ detection probability while for sender deniability, Dachman-Soled [16] showed that there is no black-box construction of sender-deniable public key encryption with super-polynomial deniability from simulatable public key encryption. There has also been work on *interactive* deniable encryption where the sender and receiver are allowed to participate in an interactive protocol – in this setting, negligible bi-deniability in the full model has been achieved based on subexponentially secure indistinguishability obfuscation and one-way functions [14]. Our focus in this work is the non-interactive setting.

2 Preliminaries

In this section, we define the notation and preliminaries that we require in this work. Some standard notions are moved to the full version [1] due to space constraints.

2.1 Fully Homomorphic Encryption

Definition 2.1 (Fully Homomorphic Encryption). *A public-key fully homomorphic encryption scheme for a message space $\mathcal{M}$ consists of PPT algorithms $\mathsf{Fhe} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ with the following syntax:*

- $\mathsf{Gen}(1^\lambda) \to (\mathsf{pk}, \mathsf{sk})$*: on input the unary representation of the security parameter λ, generates a public-key pk and a secret-key sk.*
- $\mathsf{Enc}(\mathsf{pk}, m) \to \mathsf{ct}$*: on input a public-key pk and a message $m \in \mathcal{M}$, outputs a ciphertext ct.*
- $\mathsf{Eval}(\mathsf{pk}, \mathcal{C}, \mathsf{ct}_1, \ldots, \mathsf{ct}_k) \to \mathsf{ct}$*: on input a public-key pk, a circuit $\mathcal{C} : \mathcal{M}^k \to \mathcal{M}$, and a tuple of ciphertexts $\mathsf{ct}_1, \ldots, \mathsf{ct}_k$, outputs a ciphertext ct.*
- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \to m$*: on input a secret-key sk and a ciphertext ct, outputs a message $m \in \mathcal{M}$.*

The scheme should satisfies the following properties:

Correctness. *A scheme* Fhe *is correct if for every security parameter* λ*, polynomial-time circuit* $\mathcal{C} : \mathcal{M}^k \to \mathcal{M}$*, and messages* $m_i \in \mathcal{M}$ *for* $i \in [k]$*:*

$$\Pr[\mathsf{Dec}(\mathsf{sk}, \mathsf{Eval}(\mathsf{pk}, \mathcal{C}, \mathsf{ct}_1, \ldots, \mathsf{ct}_k)) = \mathcal{C}(m_1, \ldots, m_k)] = 1 - \mathsf{negl}(\lambda)$$

where $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$*, and* $\mathsf{ct}_i \leftarrow \mathsf{Enc}(\mathsf{pk}, m_i)$ *for* $i \in [k]$*.*

Compactness. *A scheme* Fhe *is compact if there exists a polynomial* $\mathrm{poly}(\cdot)$ *such that for all security parameter* λ*, polynomial-time circuit* $\mathcal{C} : \mathcal{M}^k \to \mathcal{M}$*, and messages* $m_i \in \mathcal{M}$ *for* $i \in [k]$*:*

$$\Pr\left[|\mathsf{Eval}\left(\mathsf{pk}, \mathcal{C}, \mathsf{ct}_1, \ldots, \mathsf{ct}_k\right)| \leq \mathrm{poly}(\lambda)\right] = 1$$

where $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$*, and* $\mathsf{ct}_i \leftarrow \mathsf{Enc}(\mathsf{pk}, m_i)$ *for* $i \in [k]$*.*

CPA Security. *A scheme* Fhe *is IND-CPA secure if for all PPT adversary* $\mathcal{A}$*:*

$$\left|\Pr\left[\mathsf{FheGame}^0_{\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{FheGame}^1_{\mathcal{A}}(\lambda) = 1\right]\right| \leq \mathsf{negl}(\lambda)$$

where $\mathsf{FheGame}^b_{\mathcal{A}}(\lambda)$ *is a game between an adversary and a challenger with a challenge bit* b *defined as follows:*

- *Sample* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$*, and send* pk *to* $\mathcal{A}$*.*
- *The adversary chooses* $m_0, m_1 \in \mathcal{M}$*.*
- *Compute* $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, m_b)$*, and send* ct *to* $\mathcal{A}$*.*
- *The adversary* $\mathcal{A}$ *outputs a bit* b' *which we define as the output of the game.*

Definition 2.2 (Circular Security). *A public-key encryption scheme with key generation algorithm* Gen *and encryption algorithm* Enc *is circular secure if for every PPT adversary* $\mathcal{A}$*:*

$$\left|\Pr\left[\mathsf{CircGame}^0_{\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{CircGame}^1_{\mathcal{A}}(\lambda) = 1\right]\right| \leq \mathsf{negl}(\lambda)$$

where $\mathsf{CircGame}^b_{\mathcal{A}}(\lambda)$ *is a game between an adversary and a challenger with a challenge bit* b *defined as follows:*

- *Sample* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$*, compute* $\mathsf{ct}_{\mathsf{sk}} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{sk})$*, and give* $(\mathsf{pk}, \mathsf{ct}_{\mathsf{sk}})$ *to* $\mathcal{A}$*.*
- *The adversary chooses* $m_0, m_1 \in \mathcal{M}$*.*
- *Compute* $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, m_b)$*, and give* ct *to* $\mathcal{A}$*.*
- *The adversary* $\mathcal{A}$ *outputs a bit* b' *which we define as the output of the game.*

Definition 2.3 (Bootstrapping Procedure). *[21] Let* $\mathsf{Fhe} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ *be a public-key FHE scheme for a message space* $\mathcal{M}$ *with ciphertext space* $\mathcal{R}^{\ell_c}$*. We define the bootstrapping procedure, denoted by* $\mathsf{boot} : \mathcal{R}^{\ell_c} \to \mathcal{R}^{\ell_c}$*, as*

$$\mathsf{boot}(x) = \mathsf{Fhe.Eval}(\mathsf{pk}, \mathsf{Dec}_x, \mathsf{ct}_{\mathsf{sk}})$$

where $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Fhe.Gen}(1^\lambda)$*,* $\mathsf{ct}_{\mathsf{sk}} \leftarrow \mathsf{Fhe.Enc}(\mathsf{pk}, \mathsf{sk})$*, and* $\mathsf{Dec}_x(\mathsf{sk}) = \mathsf{Fhe.Dec}(\mathsf{sk}, x)$*. Above, when* $\mathsf{sk} \notin \mathcal{M}$*, we assume that* sk *may be represented as a vector of elements in* $\mathcal{M}$*, which would make* $\mathsf{ct}_{\mathsf{sk}}$ *a vector of ciphertexts.*

Definition 2.4 (Valid Ciphertext). *We say that an* Fhe *ciphertext* ct *is a* valid *ciphertext of m, if either*

$$\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, m),$$

or for any polynomial-sized circuit $\mathcal{C}$, we have that:

$$\Pr[\mathsf{Dec}(\mathsf{sk}, \mathsf{Eval}(\mathsf{pk}, \mathcal{C}, \mathsf{ct})) = \mathcal{C}(m)] = 1 - \mathsf{negl}(\lambda),$$

where $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$ *and λ is the security parameter.*

Some Useful Functions. In this paragraph, we define notation for some functions that will prove useful in our constructions.

Definition 2.5 (Addition Modulo 2). *We denote by $\oplus_2$ the homomorphic evaluation of addition modulo 2 circuit, that is for $k \geq 2$, $\oplus_2(\mathsf{ct}_1, \ldots, \mathsf{ct}_k) = \mathsf{ct}$,* ct *is a valid encryption of $\sum_{i=1}^{k} x_i \pmod 2$ where $x_i \in \{0,1\}$ and ct_i is a valid encryption of x_i for $i \in [k]$.*

For ease of readability, we will often denote $\oplus_2(\mathsf{ct}_1, \ldots, \mathsf{ct}_k)$ by $\mathsf{ct}_1 \oplus_2 \mathsf{ct}_2 \ldots \oplus_2 \mathsf{ct}_k$.

Definition 2.6 (Selector). *Let $b_i \in \{0,1\}$ such that for all $i \in [k], i \neq j$, $b_i = 0$, and $b_j = 1$ for some fixed $j \in [k]$. For all $i \in [k]$, let $x_i \in \mathcal{M}$. We define a selector function as $\sum_{i \in [k]} b_i x_i = x_j$.*

We denote the homomorphic evaluation of this function by

$$\sum_{i \in [k]} \mathsf{ct}_i^{\mathsf{sel}} \otimes \mathsf{ct}_i = \mathsf{ct},$$

where ct *is a valid encryption of the selected message x_j, $\mathsf{ct}_i^{\mathsf{sel}}$ is a valid encryption of b_i and ct_i is a valid encryption of x_i for all $i \in [k]$.*

Definition 2.7 (Indicator Function). *The indicator function for the set $\mathcal{X}$, denoted by $\mathbf{1}_{\mathcal{X}}(\cdot)$, defined as*

$$\mathbf{1}_{\mathcal{X}}(x) = \begin{cases} 1 & x \in \mathcal{X} \\ 0 & x \notin \mathcal{X} \end{cases}.$$

2.2 Deniable Homomorphic Encryption

Definition 2.8 (Compact Deniable FHE.). *A compact public-key deniable fully homomorphic encryption scheme for message space $\mathcal{M}$ consists of PPT algorithms* $\mathsf{DFhe} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec}, \mathsf{Fake})$ *with the following syntax:*

- $\mathsf{Gen}(1^\lambda) \rightarrow (\mathsf{dpk}, \mathsf{dsk})$*: on input the unary representation of the security parameter λ, generates a public-key* dpk *and a secret-key* dsk*.*
- $\mathsf{Enc}(\mathsf{dpk}, m; r) \rightarrow \mathsf{ct}$*: on input a public-key* dpk *and a message $m \in \mathcal{M}$, uses ℓ-bit string randomness r, outputs a ciphertexts* dct*.*

– $\mathsf{Eval}(\mathsf{dpk}, \mathcal{C}, \mathsf{dct}_1, \ldots, \mathsf{dct}_k) \to \mathsf{dct}$*: on input a public-key* dpk*, a circuit* $\mathcal{C} : \mathcal{M}^k \to \mathcal{M}$*, and a tuple of ciphertexts* $\mathsf{dct}_1, \ldots, \mathsf{dct}_k$*, outputs a ciphertext* dct*.*
– $\mathsf{Dec}(\mathsf{dsk}, \mathsf{dct}) \to m$*: on input a secret-key* dsk *and a ciphertext* dct*, outputs a message* $m \in \mathcal{M}$*.*
– $\mathsf{Fake}(\mathsf{dpk}, m, r, m^*) \to r^*$*: on input a public-key* dpk*, an original message* $m \in \mathcal{M}$*, an* ℓ*-bit string randomness* r*, and a fake message* $m^* \in \mathcal{M}$*, output an* ℓ*-bit string randomness* r^**.*

The scheme should satisfies the following properties:

Correctness, Compactness & CPA Security. *A scheme* DFhe *is correct, compact and secure if the scheme* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ *satisfies the standard notions of correctness, compactness and IND-CPA security properties of fully homomorphic encryption, as in Definition 2.1. We remark that a scheme cannot simultaneously satisfy perfect correctness and deniability, so negligible decryption error in correctness is inherent.*

Deniability. *A scheme* DFhe *is* $\delta(\lambda)$*-deniable if for all PPT adversary* $\mathcal{A}$*:*

$$\left|\Pr\left[\mathsf{DnblGame}^0_{\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{DnblGame}^1_{\mathcal{A}}(\lambda) = 1\right]\right| \leq \delta(\lambda)$$

where $\mathsf{DnblGame}^b_{\mathcal{A}}(\lambda)$ *is a game between an adversary and a challenger with a challenge bit* b *defined as follows:*
– *Sample* $(\mathsf{dpk}, \mathsf{dsk}) \leftarrow \mathsf{Gen}(1^\lambda)$*, and send* dpk *to* $\mathcal{A}$*.*
– *The adversary chooses* $m, m^* \in \mathcal{M}$*.*
– *Sample* $r \leftarrow \{0,1\}^\ell$*, and* $r^* \leftarrow \mathsf{Fake}(\mathsf{dpk}, m, r, m^*)$*; if* $b = 0$ *give* $(m^*, r, \mathsf{Enc}(\mathsf{dpk}, m^*; r))$ *to* $\mathcal{A}$*, else if* $b = 1$*, give* $(m^*, r^*, \mathsf{Enc}(\mathsf{dpk}, m; r))$ *to* $\mathcal{A}$*.*
– *The adversary* $\mathcal{A}$ *outputs a bit* b' *which we define as the output of the game.*

Remark 2.9. We note that in our constructions, the length of randomness used during encryption may depend on the message being encrypted. This does not affect deniability, because the length of the randomness is only revealed together with the encrypted message. For ease of exposition, we do not introduce additional notation to capture this nuance.

Deniability Compactness. *A* $\delta(\lambda)$*-deniable scheme* DFhe *is deniability compact if there exists a a polynomial* $\mathrm{poly}(\cdot)$ *such that for all security parameters* λ*, and message* $m \in \mathcal{M}$*:*

$$\Pr[|\mathsf{Enc}(\mathsf{dpk}, m)| \leq \mathrm{poly}(\lambda)] = 1$$

where $(\mathsf{dpk}, \mathsf{dsk}) \leftarrow \mathsf{Gen}(1^\lambda)$*, regardless of the encryption running time.*

Remark 2.10. The above definition can be modified to capture a compact deniable public key encryption scheme by removing the evaluation algorithm required by FHE.

Definition 2.11 (Weak Deniable FHE). *A public-key weak deniable fully homomorphic encryption scheme for message space $\mathcal{M}$ consists of PPT algorithms* $\mathsf{wDFhe} = (\mathsf{Gen}, \mathsf{DEnc}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec}, \mathsf{Fake})$ *where* $\mathsf{Gen}, \mathsf{Eval}$, *and* Dec *have the same syntax as in Definition 2.8, and* $\mathsf{DEnc}, \mathsf{Enc}$ *and* Fake *have the following syntax:*

- $\mathsf{DEnc}(\mathsf{dpk}, m; r) \to \mathsf{ct}$*: on input a public-key* dpk *and a message $m \in \mathcal{M}$, uses ℓ-bit string randomness r, outputs a ciphertexts* dct.
- $\mathsf{Enc}(\mathsf{dpk}, m; r) \to \mathsf{ct}$*: on input a public-key* dpk *and a message $m \in \mathcal{M}$, uses ℓ^*-bit string randomness r, outputs a ciphertexts* dct.
- $\mathsf{Fake}(\mathsf{dpk}, m, r, m^*) \to r^*$*: on input a public-key* dpk*, an original message $m \in \mathcal{M}$, an ℓ-bit string randomness r, and a faking message $m^* \in \mathcal{M}$, output an ℓ^*-bit string randomness r^*.*

The scheme should satisfies the following properties:

Correctness, Compactness & CPA Security. *A scheme* wDFhe *is correct, compact and secure if both schemes* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$*, and* $(\mathsf{Gen}, \mathsf{DEnc}, \mathsf{Eval}, \mathsf{Dec})$ *satisfy the standard notions of correctness, compactness and IND-CPA security properties of fully homomorphic encryption, as in Definition 2.1.*

Weak Deniability. *A scheme* wDFhe *is weakly-deniable if for all PPT adversaries $\mathcal{A}$:*

$$\left|\Pr\left[\mathsf{wDnblGame}^0_{\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{wDnblGame}^1_{\mathcal{A}}(\lambda) = 1\right]\right| \leq \mathsf{negl}(\lambda)$$

where $\mathsf{wDnblGame}^b_{\mathcal{A}}(\lambda)$ *is a game between an adversary and a challenger with a challenge bit b defined as follows:*

- *Sample* $(\mathsf{dpk}, \mathsf{dsk}) \leftarrow \mathsf{Gen}(1^\lambda)$*, and send* dpk *to $\mathcal{A}$.*
- *The adversary $\mathcal{A}$ chooses $m, m^* \in \mathcal{M}$.*
- *Sample $r \leftarrow \{0,1\}^{\ell^*}$, $r' \leftarrow \{0,1\}^{\ell}$, and $r^* \leftarrow \mathsf{Fake}(\mathsf{dpk}, m, r', m^*)$; if $b = 0$ return $(m^*, r, \mathsf{Enc}(\mathsf{dpk}, m^*; r))$ else if $b = 1$ return $(m^*, r^*, \mathsf{DEnc}(\mathsf{dpk}, m; r'))$ to $\mathcal{A}$.*
- *The adversary $\mathcal{A}$ outputs a bit b' which we define as the output of the game.*

3 Special Homomorphic Encryption

Our constructions rely on a fully homomorphic encryption scheme which satisfies some special properties. We define these and instantiate it below.

Definition 3.1 (Special FHE). *A special public-key FHE scheme for a message space $\mathcal{M}$ with ciphertext space $\mathcal{R}^{\ell_c}$ is a public-key FHE scheme,* $\mathsf{Fhe} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$*, with the following additional properties:*

1. *Deterministic Algorithms. The evaluation and decryption algorithms,* Eval *and* Dec *respectively, are deterministic. In particular, this implies the bootstrapping procedure* boot*, defined in 2.3, is deterministic.*

2. *Pseudorandom Ciphertext. The distribution* $\mathsf{Fhe.Enc}(\mathsf{pk}, m; U^\ell)$ *is computationally indistinguishable from* $\mathcal{R}^{\ell_c}$*, where* U^ℓ *is the uniform distribution over* ℓ*-bit strings,* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Fhe.Gen}(1^\lambda)$*, and* $m \in \mathcal{M}$*. Moreover, the distribution* $\mathsf{boot}(\mathcal{R}^{\ell_c})$ *is computationally indistinguishable from* $\mathcal{R}^{\ell_c}$*, where* boot *is the bootstrapping procedure as in Definition 2.3.*
3. *Decryption Outputs Valid Message. The decryption algorithm,* $\mathsf{Fhe.Dec}$*, always outputs a message from the message space* $\mathcal{M}$*. Namely, for any* $x \in \mathcal{R}^{\ell_c}$*,* $\mathsf{Fhe.Dec}(\mathsf{sk}, x) \in \mathcal{M}$ *where* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Fhe.Gen}(1^\lambda)$*. In particular, this implies that the output of the bootstrapping procedure* boot *is always a valid ciphertext (Definition 2.4).*
4. *Biased Decryption on Random Input (Strong Version). The decryption algorithm* $\mathsf{Fhe.Dec}$*, when invoked with a random element in the ciphertext space* $x \leftarrow \mathcal{R}^{\ell_c}$*, outputs a message from a fixed (strict) subset of the message space* $\mathcal{S} \subset \mathcal{M}$ *with overwhelming probability.*
   Formally, we require that there exists a strict subset of the message space, $\mathcal{S} \subset \mathcal{M}$*, such that*

$$P(\mathcal{S}) := \sum_{m \in S} P(m) \geq 1 - \mathsf{negl}(\lambda)$$

   where $P : \mathcal{M} \to \mathbb{R}$ *is defined as* $P(m) := \Pr\left[\mathsf{Fhe.Dec}\left(\mathsf{sk}, x\right) = m\right]$ *where* $x \leftarrow \mathcal{R}^{\ell_c}$ *and* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Fhe.Gen}(1^\lambda)$*. Moreover, we require that* $0 \in \mathcal{S}$*. Thus, if the message space is binary, then* $\mathcal{S} = \{0\}$*.*
   We remark that the above property, while sufficient, is not strictly necessary for our constructions. However, for ease of exposition, our constructions assume the "strong version" stated above. In the full version [1] we describe how to modify our constructions to instead use the weaker version below.
   Biased Decryption on Random Input (Weak Version). This version weakens overwhelming to noticeable in the above definition, i.e. using the notation above, we require:

$$P(\mathcal{S}) := \sum_{m \in S} P(m) \geq 1/\operatorname{poly}(\lambda)$$

   As before, we require that $0 \in \mathcal{S}$*.*
5. *Circular Secure. The scheme* Fhe *is circular secure as in Definition 2.2. As discussed in Section 1, this condition may be removed at the cost of making the construction more complicated, please see the full version [1] for details. Since this condition is anyway required for the construction of pure FHE, we assume it for ease of exposition.*

3.1 Instantiation

For concreteness, we instantiate our special FHE scheme with (a modified version of) the scheme by Brakerski, Gentry and Vaikuntanathan [10] (henceforth BGV), which is based on the hardness of the learning with errors (LWE) problem. To begin, note that BGV already satisfies the property that the algorithms for

evaluation and decryption are deterministic (property 1), the property that the ciphertext is pseudorandom (property 2) as well as the property that decryption always outputs valid message (property 3). The property of circular security (property 5) does not provably hold in BGV, or indeed any existing FHE scheme, but is widely *assumed* to hold for BGV. In particular, the authors already assume it for optimized versions of their main construction (which does not require this assumption)– please see [10, Section 5] for a discussion. We also remark that circular security is assumed by all "pure" FHE schemes, namely, schemes that can support homomorphic evaluation of circuits of arbitrary polynomial depth. We require circular security for a different reason – to support the bootstrapping operation, which allows us to obliviously sample FHE ciphertexts. Thus, it remains to establish the property that decryption of a (truly) random element from the ciphertext space outputs a biased message from the message space (property 4). Establishing this property requires slight modifications to the BGV scheme[1]. Next, we describe these modifications for the case when the $\mathcal{M}$ is binary, of polynomial size and of super-polynomial size.

Recap of BGV. Let us consider the BGV construction for binary messages [10, Section 4]. We begin by providing a brief recap of the features of BGV that we require. We use the same notation as in their paper for ease of verification. Let $\mathcal{R}$ be a ring and $|\mathcal{R}| = q$. Recall that the key generation algorithm of BGV samples a vector $\mathbf{s}' \in \mathcal{R}^n$ such that all the entries of $\mathbf{s}'$ are "small" with high probability (details of the distribution are not relevant here) and outputs $\mathsf{sk} = \mathbf{s} = (1, \mathbf{s}')$. The public key is constructed by sampling a uniform random matrix $\mathbf{A}' \leftarrow \mathcal{R}^{N \times n}$, an error vector $\mathbf{e} \in \mathcal{R}^N$ from a special "error" distribution, and setting $\mathbf{b} = \mathbf{A}'\mathbf{s}' + 2 \cdot \mathbf{e}$. Denote by $\mathbf{A}$ the $(n+1)$ column matrix consisting of $\mathbf{b}$ followed by the n columns of $-\mathbf{A}'$. Observe that $\mathbf{A} \cdot \mathbf{s} = 2\mathbf{e}$. The public key contains $\mathbf{A}$ in addition to some other elements which are not relevant for our discussion[2]. To encrypt a message bit m, set $\mathbf{m} = (m, 0, 0, \ldots, 0) \in \{0,1\}^{n+1}$, sample $\mathbf{r} \leftarrow \{0,1\}^N$ and output $\mathsf{ct} = \mathbf{m} + \mathbf{A}^\top \mathbf{r}$. To decrypt, compute and output $[[\langle \mathsf{ct}, \mathsf{sk} \rangle]_q]_2$, where $\langle \cdot\ , \cdot \rangle$ denotes inner product over the ring, and $[\cdot]_p$ denotes reduction modulo p. The above construction can be adapted to support larger message spaces. A simple extension is to choose the message from $\mathbb{Z}_p$ for a polynomial sized prime p and multiply the error with p instead of 2. This, and other extensions are discussed in detail in [10, Section 5].

Creating a Bias. Observe that the decryption algorithm, given a ciphertext ct and secret sk, outputs the decrypted message bit as $[[\langle \mathsf{ct}, \mathsf{sk} \rangle]_q]_2$ regardless of the distribution of ct. Thus, even if ct is a random element from the ciphertext space $\mathcal{R}^{n+1}$ which may not be well formed, it still outputs a valid message from the message space. However, it is easy to see that for a random element $R \leftarrow \mathcal{R}^{n+1}$,

[1] We note that these properties are also satisfied by several other FHE schemes, for instance [11, 7, 23].

[2] Since we assume circular security which BGV do not, we can simplify their scheme – in particular, we not need fresh keys for each level of the circuit as they do.

the output of $[[\langle R, \mathsf{sk}\rangle]_q]_2$ is a uniformly distributed random bit, whereas we require the decryption algorithm to output a biased bit to satisfy property 4. Below, we will describe the modification to BGV to achieve the strong version of property 4. In the full version [1], we describe how we can instead rely on the weak version of the property, which is satisfied by BGV unmodified.

To create a bias, an idea is to build in an additional step in the decryption algorithm, which first checks whether the input ciphertext ct is well-formed. If so, it proceeds with legitimate decryption, i.e. computes $[[\langle \mathsf{ct}, \mathsf{sk}\rangle]_q]_2$. If not, it simply outputs 0. Since well-formed ciphertexts in the BGV FHE are sparse in the ciphertext space $\mathcal{R}^{n+1}$, this ensures that a randomly chosen element from the ciphertext space is decrypted to 0 with high probability.

It remains to identify an efficient check for the well-formedness of the ciphertext. Towards this, we observe that for any valid ciphertext (Definition 2.4), the inner product $[\langle \mathsf{ct}, \mathsf{sk}\rangle]_q = m + 2e$ where m is the encrypted bit and e is some error whose norm may be bounded using bounds on the norms of the secret key $\mathbf{s}$, the randomness $\mathbf{r}$, the error term in the public key $\mathbf{e}$ and the depth of the circuit – of which the norms of all aforementioned elements were chosen to be sufficiently "small" and the depth of the circuit can be bounded by the depth of the bootstrapping circuit [21].

Let us assume that the decryption error is bounded above by $B - 1$, for some $B = \mathrm{poly}(\lambda)$. We note that this bound holds true for the current setting of parameters in [10]. Then, it follows that the output of step 1 of decryption can be bounded from above by B (for any well formed ciphertext). On the other hand, the output of $[\langle R, \mathsf{sk}\rangle]_q$ for a random element R will also be uniformly distributed, and hence will have norm $\leq B$ only with probability $O(\frac{B}{q})$. If we set q to be super-polynomial in the security parameter, then this term is negligible. Thus, we may modify the BGV decryption algorithm so that after computing $[\langle \mathsf{ct}, \mathsf{sk}\rangle]_q$, it checks whether the output is $\leq B$, and outputs 0 if not. This biases the output of decryption to 0 for random inputs – in more detail, decryption of a random element yields 0 with probability $1 - \mathsf{negl}(\lambda)$ as desired. With this modification, we ensured that BGV satisfies all the properties required by special FHE. We refer the reader to [10] for more details about the full construction of FHE.

In the above description, we chose the ring modulus q to be super-polynomial in the security parameter to obtain the desired bias. However, this large modulus is unnecessary and affects the efficiency of the scheme negatively. In the full version [1], we describe how to relax this requirement.

Next, we discuss how to modify the BGV scheme supporting larger (polynomial) message spaces, as discussed in [10, Section 5]. As in the case of binary messages (discussed above), we have that without performing any modifications, the BGV decryption algorithm, if executed on a random element in the ciphertext space, outputs a uniformly distributed message from the message space.

It remains to establish property 4 which requires that there exists a strict subset of the message space, $\mathcal{S} \subset \mathcal{M}$, such that

$$P(\mathcal{S}) := \sum_{m \in S} P(m) \geq 1 - \mathsf{negl}(\lambda)$$

where $P : \mathcal{M} \to \mathbb{R}$ is defined as $P(m) := \Pr\left[\mathsf{Fhe.Dec}\left(\mathsf{sk}, x\right) = m\right]$ where $x \leftarrow \mathcal{R}^{\ell_c}$ and $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Fhe.Gen}(1^\lambda)$.

Let $\mathcal{S}$ be an arbitrary subset of $\mathcal{M}$ that contains 0. For the binary message case above, we described a trick that ensures that random elements are decrypted to 0 with overwhelming probability. The same trick may be generalized to larger message spaces. If the modulus q is superpolynomial, and the message space is polynomial (say of size p), then the first step of decryption yields $[\langle \mathsf{ct}, \mathsf{sk} \rangle]_q = m + p \cdot e$ for well-formed ciphertexts, and a random element in $\mathcal{R}$ otherwise. Again, this term can be bounded by some polynomial B and the decryption procedure can be modified to output 0 (or any element from the set $\mathcal{S}$) if the output of step 1 is greater than B. By the same reasoning as above, this biases the output to $\mathcal{S}$ with overwhelming probability as long as q is super-polynomial. Please see the full version [1] to avoid the restriction of super-polynomial q.

Finally, we remark that BGV also includes variants where the message space is super-polynomial in size [10, Section 5.4]. In this case, biasing the output to a fixed set $\mathcal{S}$ is simple: we can just set $\mathcal{S} = \mathcal{M} \setminus \{1\}$. Moreover $\mathcal{S}$ has efficient representation since it can simply be represented by its complement, which is of small size and it is clear that the decryption output of a random element is biased to $\mathcal{S}$ with overwhelming probability.

4 Deniable Encryption for Bits

In this section, we provide our constructions for weak deniable FHE, as in Definition 2.11, and compact deniable FHE, as in Definition 2.8. Let $\mathsf{Fhe} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ be a *special* public-key FHE scheme for the message space $\mathcal{M} = \{0, 1\}$ with ciphertext space $\mathcal{R}^{\ell_c}$, as in Definition 3.1. For reading convenience, we denote by lowercase r, the ℓ-bit string randomness that is input to an $\mathsf{Fhe.Enc}$ algorithm, and by uppercase R, the elements in $\mathcal{R}^{\ell_c}$, where $\mathcal{R}^{\ell_c}$ is the co-domain of the algorithm $\mathsf{Fhe.Enc}$. We denote by ℓ'_c the bit length of elements in $\mathcal{R}^{\ell_c}$ (that is, $\ell'_c = \lceil \ell_c \log_2(|\mathcal{R}|) \rceil$). Recall that boot denotes the bootstrapping procedure described in Definition 2.3 and $\oplus_2$ denotes the homomorphic evaluation of addition mod 2 described in Definition 2.5.

4.1 Weakly Deniable FHE for Bits

Our public-key weak deniable fully homomorphic encryption scheme for message space $\mathcal{M} = \{0, 1\}$, $\mathsf{wDFhe} = (\mathsf{Gen}, \mathsf{DEnc}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec}, \mathsf{Fake})$, is described as follows:

$\mathsf{wDFhe.Gen}(1^\lambda)$: Upon input the unary representation of the security parameter λ, do the following:

1. Sample $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Fhe.Gen}(1^\lambda)$, and $\mathsf{ct}_{\mathsf{sk}} \leftarrow \mathsf{Fhe.Enc}(\mathsf{pk}, \mathsf{sk})$.
2. Outputs $\mathsf{dpk} := (\mathsf{pk}, \mathsf{ct}_{\mathsf{sk}}), \mathsf{dsk} := \mathsf{sk}$

$\mathsf{wDFhe.DEnc}(\mathsf{dpk}, m; \mathsf{r})$: Upon input the public key dpk, a message bit m and $(3\ell + \ell'_c)$-bit string randomness r, do the following:

1. Parse $\mathsf{dpk} := (\mathsf{pk}, \mathsf{ct}_{\mathsf{sk}})$ and $\mathsf{r} = (r_1, r_2, r_3, R_4)$, where $|r_i| = \ell$ for $i \in [3]$ and $|R_4| = \ell'_c$.
2. For $i \in [3]$, set $R_i = \mathsf{Fhe.Enc}(\mathsf{pk}, 1; r_i)$.
3. Let $\mathsf{ct}_0 = \mathsf{boot}(R_1) \oplus_2 \mathsf{boot}(R_2)$ and $\mathsf{ct}_1 = \mathsf{boot}(R_4) \oplus_2 \mathsf{boot}(R_3)$.
4. Output $\mathsf{dct} = \mathsf{ct}_m$.

$\mathsf{wDFhe.Enc}(\mathsf{dpk}, m; \mathsf{r})$: Upon input the public-key dpk, the message bit m, and the $(\ell + 3\ell'_c)$-bit string randomness r, do the following:

1. Parse $\mathsf{dpk} := (\mathsf{pk}, \mathsf{ct}_{\mathsf{sk}})$ and $\mathsf{r} = (R_1, R_2, R_3, r_4)$, where $|R_i| = \ell'_c$ for $i \in [3]$ and $|r_4| = \ell$.
2. Set $R_4 = \mathsf{Fhe.Enc}(\mathsf{pk}, 1; r_4)$.
3. Let $\mathsf{ct}_0 = \mathsf{boot}(R_1) \oplus_2 \mathsf{boot}(R_2)$ and $\mathsf{ct}_1 = \mathsf{boot}(R_3) \oplus_2 \mathsf{boot}(R_4)$.
4. Output $\mathsf{dct} = \mathsf{ct}_m$.

$\mathsf{wDFhe.Eval}(\mathsf{dpk}, \mathcal{C}, \mathsf{dct}_1, \ldots, \mathsf{dct}_k)$: Upon input the public key $\mathsf{dpk} = (\mathsf{pk}, \mathsf{ct}_{\mathsf{sk}})$, the circuit $\mathcal{C}$ and the ciphertexts $\mathsf{dct}_1, \ldots, \mathsf{dct}_k$, interpret dct_i as Fhe ciphertext ct_i for $i \in [k]$, and output $\mathsf{dct} = \mathsf{Fhe.Eval}(\mathsf{pk}, \mathcal{C}, \mathsf{ct}_1, \ldots, \mathsf{ct}_k)$.

$\mathsf{wDFhe.Dec}(\mathsf{dsk}, \mathsf{dct})$: Upon input the secret key dsk and the ciphertext dct, interpret dsk and dct as Fhe secret key sk and Fhe ciphertext ct and output $\mathsf{Fhe.Dec}(\mathsf{sk}, \mathsf{ct})$.

$\mathsf{wDFhe.Fake}(\mathsf{dpk}, m, \mathsf{r}, m^*)$: Upon input the public key dpk, the original message bit m, $(3\ell + \ell'_c)$-bit string randomness r, and the faking message bit m^*, do the following:

1. Parse $\mathsf{dpk} := (\mathsf{pk}, \mathsf{ct}_{\mathsf{sk}})$ and $\mathsf{r} = (r_1, r_2, r_3, R_4)$, where $|r_i| = \ell$ for $i \in [3]$ and $|R_4| = \ell'_c$.
2. For $i \in [3]$, set $R_i = \mathsf{Fhe.Enc}(\mathsf{pk}, 1; r_i)$.
3. If $m = m^*$, then set $R^*_1 = R_1$, $R^*_2 = R_2$, $R^*_3 = R_4$, and $r^*_4 = r_3$.
4. Else if $m \neq m^*$, then set $R^*_1 = R_4$, $R^*_2 = R_3$, $R^*_3 = R_1$, and $r^*_4 = r_2$.
5. Output $\mathsf{r}^* = (R^*_1, R^*_2, R^*_3, r^*_4)$

We now prove the scheme satisfies correctness, compactness, CPA security and weak deniability.

Compactness and Security. Observe that the output of both $\mathsf{wDFhe.DEnc}$ and $\mathsf{wDFhe.Enc}$ is a valid ciphertext of the underlying Fhe scheme. This is due to property 3 of the special FHE which states that the FHE decryption algorithm always outputs a valid bit, and due to the correctness of FHE evaluation which implies correctness of bootstrapping. Together, these two properties ensure that boot always outputs a valid ciphertext. Moreover, correctness of homomorphic evaluation implies that the addition mod 2 operation is performed correctly, so that the output of $\mathsf{wDFhe.DEnc}$ and $\mathsf{wDFhe.Enc}$ is a valid ciphertext of FHE.

Since the underlying FHE scheme satisfies compactness, it holds that the ciphertext output by $\mathsf{wDFhe.DEnc}$ and $\mathsf{wDFhe.Enc}$ is also compact. Similarly,

due to property 5 which states that the scheme is circular secure, and since the ciphertext of the underlying FHE satisfies semantic security, so does the ciphertext output by $\mathsf{wDFhe.DEnc}$ and $\mathsf{wDFhe.Enc}$. Thus, both schemes are compact and secure as the underlying FHE scheme is.

Correctness. We start by proving correctness of the deniable encryption algorithm $\mathsf{wDFhe.DEnc}$. Parse $\mathsf{r} \in \{0,1\}^{3\ell+\ell'_c}$ as $\mathsf{r} = (r_1, r_2, r_3, R_4)$. Observe that:

1. Since $R_i = \mathsf{Fhe.Enc}(\mathsf{pk}, 1; r_i)$ for $i \in [3]$, we have by correctness of the underlying Fhe, that R_1, R_2 and R_3 are valid encryptions of 1.
2. By properties 3 and 4 which state that FHE decryption always outputs a bit and this bit is biased to 0 with overwhelming probability when decryption is invoked with a truly random input, we have that $\mathsf{boot}(R_4)$ is a valid encryption of 0 with overwhelming probability.

Now, by correctness of FHE evaluation, we have that $\mathsf{ct}_0 = \mathsf{boot}(R_1) \oplus_2 \mathsf{boot}(R_2)$ is a valid encryption of 0 and $\mathsf{ct}_1 = \mathsf{boot}(R_4) \oplus_2 \mathsf{boot}(R_3)$ is a valid encryption of 1.

Next we prove correctness of $\mathsf{wDFhe.Enc}$. Parse $\mathsf{r} \in \{0,1\}^{\ell+3\ell'_c}$ as $\mathsf{r} = (R_1, R_2, R_3, r_4)$. Observe that:

1. Since $R_4 = \mathsf{Fhe.Enc}(\mathsf{pk}, 1; r_4)$, we have that R_4 is a valid encryption of 1.
2. As above, we have by properties 3 and 4 that $\mathsf{boot}(R_i)$ for $i \in [3]$ are valid encryptions of 0 with overwhelming probability.

Thus, again by correctness of FHE evaluation, we have that $\mathsf{ct}_0 = \mathsf{boot}(R_1) \oplus_2 \mathsf{boot}(R_2)$ is a valid encryption of 0 and $\mathsf{ct}_1 = \mathsf{boot}(R_3) \oplus_2 \mathsf{boot}(R_4)$ is a valid encryption of 1.

Weak-Deniability. Next, we prove weak deniability of the construction. Fix a security parameter λ, an original message $m \in \{0,1\}$, and a faking message $m^* \in \{0,1\}$. Let $(\mathsf{dpk}, \mathsf{dsk}) \leftarrow \mathsf{wDFhe.Gen}(1^\lambda)$, and parse $\mathsf{dpk} := (\mathsf{pk}, \mathsf{ct}_{\mathsf{sk}}), \mathsf{dsk} := \mathsf{sk}$.

Faking Case. First consider the distribution of $(\mathsf{dpk}, m^*, \mathsf{r}, \mathsf{DEnc}(\mathsf{dpk}, m; \mathsf{r}'))$ in the case of faking.

1. Select uniformly at random $\mathsf{r}' \leftarrow \{0,1\}^{3\ell} \times \mathcal{R}^{\ell_c}$.
2. Parse $\mathsf{r}' := (r_1, r_2, r_3, R_4)$, where $|r_i| = \ell$ for $i \in [3]$ and $|R_4| = \ell'_c$.
3. For $i \in [3]$, set $R_i = \mathsf{Fhe.Enc}(\mathsf{pk}, 1; r_i)$.
4. Let $\mathsf{r}^* = \mathsf{wDFhe.Fake}(\mathsf{dpk}, m, \mathsf{r}', m^*)$.
5. By the faking algorithm $\mathsf{r}^* = \left(R_1^*, R_2^*, R_3^*, r_4^*\right)$ which is computed as follows:
   (a) *Case* $m = m^*$:

$$R_1^* = R_1, \quad R_2^* = R_2, \quad R_3^* = R_4, \quad r_4^* = r_3.$$

By property 2 which asserts that ciphertexts are pseudorandom, we can explain R_1^* and R_2^* as uniform from the ciphertexts space $\mathcal{R}^{\ell_c}$. Here, $R_3^* = R_4$ is already a uniform element in $\mathcal{R}^{\ell_c}$, and $r_4^* = r_3$ is a uniform ℓ bit string.

(b) *Case* $m \neq m^*$:

$$R_1^* = R_4, \quad R_2^* = R_3, \quad R_3^* = R_1, \quad r_4^* = r_2.$$

As above, we can explain R_2^* and R_3^* as uniform elements in $\mathcal{R}^{\ell_c}$, and $R_1^* = R_4$ and $r_4^* = r_2$ are already uniform.

6. The output of this hybrid is:

$$\left(\mathsf{dpk}, m^*, \mathsf{r}^* = (R_1^*, R_2^*, R_3^*, r_4^*), \mathsf{ct}^* = \mathsf{wDFhe.DEnc}(\mathsf{dpk}, m; \mathsf{r}')\right)$$

where $\mathsf{ct}^* := \mathsf{ct}_m$, $\mathsf{ct}_0 = \mathsf{boot}(R_1) \oplus_2 \mathsf{boot}(R_2)$ and $\mathsf{ct}_1 = \mathsf{boot}(R_4) \oplus_2 \mathsf{boot}(R_3)$.
Observe that $\mathsf{ct}^* = \mathsf{wDFhe.Enc}(\mathsf{dpk}, m^*; \mathsf{r}^*)$. Thus, the output of this hybrid can be written as:

$$\left(\mathsf{dpk}, m^*, \mathsf{r}^* = (R_1^*, R_2^*, R_3^*, r_4^*), \mathsf{ct}^* = \mathsf{wDFhe.Enc}(\mathsf{dpk}, m^*; \mathsf{r}^*)\right)$$

where $\mathsf{ct}^* := \mathsf{ct}_{m^*}$, $\mathsf{ct}_0 = \mathsf{boot}(R_1^*) \oplus_2 \mathsf{boot}(R_2^*)$, $\mathsf{ct}_1 = \mathsf{boot}(R_3^*) \oplus_2 \mathsf{boot}(R_4^*)$ and R_1^*, R_2^*, R_3^* and r_4^* are explained as uniform in $\mathcal{R}^{3\ell_c} \times \{0,1\}^{\ell}$.

Honest Case. Next, note that in the honest case $\mathsf{r} \leftarrow \mathcal{R}^{3\ell_c} \times \{0,1\}^{\ell}$, so the output distribution is:

$$\left(\mathsf{dpk}, m^*, \mathsf{r} = (R_1, R_2, R_3, r_4), \mathsf{ct}^* = \mathsf{wDFhe.Enc}(\mathsf{dpk}, m^*; \mathsf{r})\right)$$

where $\mathsf{ct}^* := \mathsf{ct}_{m^*}$, $\mathsf{ct}_0 = \mathsf{boot}(R_1) \oplus_2 \mathsf{boot}(R_2)$, $\mathsf{ct}_1 = \mathsf{boot}(R_3) \oplus_2 \mathsf{boot}(R_4)$ and R_1, R_2, R_3 and r_4 are sampled uniformly. Hence, the two distributions are indistinguishable.

4.2 Fully Deniable FHE for Bits

Our compact public-key $1/\delta$-deniable[3] fully homomorphic encryption scheme for message space $\mathcal{M} = \{0,1\}$, $\mathsf{DFhe} = (\mathsf{Gen}, \mathsf{DEnc}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec}, \mathsf{Fake})$, is described below. We also provide an alternate construction with slightly different parameters in the full version [1]. Recall that boot denotes the bootstrapping procedure described in Definition 2.3 and $\oplus_2$ denotes the homomorphic evaluation of addition mod 2 described in Definition 2.5). We let $n = \delta^2$.

$\mathsf{DFhe.Gen}(1^\lambda)$: Upon input the unary representation of the security parameter λ, do the following:
1. Sample $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Fhe.Gen}(1^\lambda)$, and $\mathsf{ct}_{\mathsf{sk}} \leftarrow \mathsf{Fhe.Enc}(\mathsf{pk}, \mathsf{sk})$.
2. Outputs $\mathsf{dpk} := (\mathsf{pk}, \mathsf{ct}_{\mathsf{sk}}), \mathsf{dsk} := \mathsf{sk}$.

$\mathsf{DFhe.Enc}(\mathsf{dpk}, m)$: Upon input the public-key dpk, the message bit m, do the following:
1. Parse $\mathsf{dpk} := (\mathsf{pk}, \mathsf{ct}_{\mathsf{sk}})$
2. Select r as follows:
   (a) Select uniformly $x_1, \ldots, x_n \in \{0,1\}$ such that $\sum_{i=1}^{n} x_i = m \pmod 2$.

[3] We remind the reader that $\delta = \delta(\lambda)$, but we drop the λ for readability.

(b) For $i \in [n]$: if $x_i = 1$, then select $r_i \leftarrow \{0,1\}^\ell$; else if $x_i = 0$, select $R_i \leftarrow \mathcal{R}^{\ell_c}$.

3. For $i \in [n]$ such that $x_i = 1$, set $R_i = \mathsf{Fhe.Enc}(\mathsf{pk}, 1; r_i)$.
4. Output $\mathsf{dct} = \oplus_2(\mathsf{boot}(R_1), \ldots, \mathsf{boot}(R_n))$

$\mathsf{DFhe.Eval}(\mathsf{dpk}, \mathcal{C}, \mathsf{dct}_1, \ldots, \mathsf{dct}_k)$: Upon input the public key $\mathsf{dpk} = (\mathsf{pk}, \mathsf{ct}_{\mathsf{sk}})$, the circuit $\mathcal{C}$ and the ciphertexts $\mathsf{dct}_1, \ldots, \mathsf{dct}_k$, interpret dct_i as Fhe ciphertext ct_i for $i \in [k]$, and output $\mathsf{dct} = \mathsf{Fhe.Eval}(\mathsf{pk}, \mathcal{C}, \mathsf{ct}_1, \ldots, \mathsf{ct}_k)$.

$\mathsf{DFhe.Dec}(\mathsf{dsk}, \mathsf{dct})$: Upon input the secret key dsk and the ciphertext dct, interpret dsk and dct as Fhe secret key sk and Fhe ciphertext ct and output $\mathsf{Fhe.Dec}(\mathsf{sk}, \mathsf{ct})$.

$\mathsf{DFhe.Fake}(\mathsf{dpk}, m, \mathsf{r}, m^*)$: Upon input the public key dpk, the original message bit m, randomness r, and the fake message m^* do the following:

1. If $m = m^*$, output $\mathsf{r}^* = \mathsf{r}$.
2. Parse $\mathsf{dpk} := (\mathsf{pk}, \mathsf{ct}_{\mathsf{sk}})$ and $\mathsf{r} = (x_1, \ldots, x_n, \rho_1, \ldots, \rho_n)$, where $x_1, \ldots, x_n \in \{0,1\}$, and for each $i \in [n]$, if $x_i = 1$, then $|\rho_i| = \ell$; else if $x_i = 0$, $|\rho_i| = \ell'_c$.
3. Select uniform $i^* \in [n]$ such that $x_{i^*} = 1$. If there is no such i^*, output "cheating impossible"; else:
   (a) Set $x^*_{i^*} = 0$ and $\rho^*_{i^*} = \mathsf{Fhe.Enc}(\mathsf{pk}, 1; \rho_{i^*})$;
   (b) For $i \in [n] \setminus \{i^*\}$, set $x^*_i = x_i$ and $\rho^*_i = \rho_i$.
4. Output $\mathsf{r}^* = (x^*_1, \ldots, x^*_n, \rho^*_1, \ldots, \rho^*_n)$.

We now prove the scheme satisfies correctness, compactness, CPA security and poly deniability. Compactness and security follow exactly as in Section 4.1.

Correctness. To argue correctness, we note that:

1. Since $R_i = \mathsf{Fhe.Enc}(\mathsf{pk}, 1; r_i)$ for i such that $x_i = 1$, we have by correctness of the underlying Fhe that R_i, and hence $\mathsf{boot}(R_i)$ are valid encryptions of 1 for all $i \in [n]$ such that $x_i = 1$.
2. By properties 3 and 4 which state that FHE decryption always outputs a bit and this bit is biased to 0 with overwhelming probability when decryption is invoked with a truly random input, we have that $\mathsf{boot}(R_i)$ for i such that $x_i = 0$ is valid encryption of 0 with overwhelming probability.

Hence, since $\sum_{i=1}^n x_i = m \pmod 2$, the (FHE evaluation of) addition mod 2 of $\mathsf{boot}(R_i)$ for $i \in [n]$ yields an encryption of m. Hence, the scheme encodes the message bit correctly.

Deniability. Next, we prove $1/\delta$-deniability of the construction. Fix a security parameter λ, an original message $m \in \{0,1\}$, and a faking message $m^* \in \{0,1\}$. Let $(\mathsf{dpk}, \mathsf{dsk}) \leftarrow \mathsf{DFhe.Gen}(1^\lambda)$, and parse $\mathsf{dpk} := (\mathsf{pk}, \mathsf{ct}_{\mathsf{sk}}), \mathsf{dsk} := \mathsf{sk}$. When the original message m and the fake message m^* are the same, the faked randomness r^* is equal to the original randomness r. Thus in this case, $m = m^*$, the distributions are identical:

$$(\mathsf{dpk}, m^*, \mathsf{r}, \mathsf{DFhe.Enc}(\mathsf{dpk}, m^*; \mathsf{r})) = (\mathsf{dpk}, m^*, \mathsf{r}^*, \mathsf{DFhe.Enc}(\mathsf{dpk}, m; \mathsf{r})).$$

When the original message m and the fake message m^* are not the same, observe that "cheating impossible" will be output only in case that $x_i = 0$ for all $i \in [n]$, which occurs with probability 2^{-n}. Assuming we are not in this case, the output distribution is:

Faking Case. First consider the distribution of $(\mathsf{dpk}, m^*, \mathsf{r}^*, \mathsf{DFhe.Enc}(\mathsf{dpk}, m; \mathsf{r}))$ in the case of faking, where $\mathsf{r}^* \leftarrow \mathsf{DFhe.Fake}(\mathsf{dpk}, m, \mathsf{r}; m^*)$.

1. Select uniform $\mathsf{r} := (x_1, \ldots, x_n, \rho_1, \ldots, \rho_n)$, by,
   (a) Select $x_i \leftarrow \{0,1\}$ for $i \in [n]$ such that $\sum_{i\in[n]} x_i = m \pmod 2$
   (b) For $i \in [n]$, if $x_i = 1$, select $\rho_i \leftarrow \{0,1\}^\ell$
   (c) For $i \in [n]$, if $x_i = 0$, select $\rho_i \leftarrow \mathcal{R}^{\ell_c}$
2. Let $\mathsf{r}^* = \mathsf{DFhe.Fake}(\mathsf{dpk}, m, \mathsf{r}, m^*)$, that is $\mathsf{r}^* = (x_1^*, \ldots, x_n^*, \rho_1^*, \ldots, \rho_n^*)$ which is computed as follows:
   (a) Select a uniform index $i^* \in [n]$ such that $x_{i^*} = 1$, i.e. $i^* \leftarrow \{i | x_i = 1\}$.
   (b) For $i \in [n], i \neq i^*$, set $x_i^* = x_i$ and $\rho_i^* = \rho_i$.
   (c) Set $x_{i^*} = 0$, and $\rho_{i^*}^* = \mathsf{Fhe.Enc}(\mathsf{pk}, 1; \rho_{i^*})$.

Intermediate Case. By property 2 of the special FHE, which asserts that ciphertexts are pseudorandom, we can explain $\rho_{i^*}^* = \mathsf{Fhe.Enc}(\mathsf{pk}, 1; \rho_{i^*})$ as uniform element from the ciphertexts space $\mathcal{R}^{\ell_c}$. The distribution of this hybrid is $(\mathsf{dpk}, m^*, \mathsf{r}', \mathsf{DFhe.Enc}(\mathsf{dpk}, m; \mathsf{r}))$, where $\mathsf{r}' = (x_1', \ldots, x_n', \rho_1', \ldots, \rho_n')$ is sampled as follows:

1. Select $x_i \leftarrow \{0,1\}$ for $i \in [n]$ such that $\sum_{i\in[n]} x_i = m \pmod 2$
2. Select a uniform index $i' \in [n]$ such that $x_{i'} = 1$ (i.e. $i' \leftarrow \{i | x_i = 1\}$), and set $x_{i'}' = 0$, and for all $i \in [n] \setminus \{i'\}$ set $x_i' = x_i$.
3. For $i \in [n]$, if $x_i' = 1$, select $\rho_i' \leftarrow \{0,1\}^\ell$
4. For $i \in [n]$, if $x_i' = 0$, select $\rho_i' \leftarrow \mathcal{R}^{\ell_c}$

Honest Case. Note that in the honest case the distribution is $(\mathsf{dpk}, m^*, \mathsf{r}, \mathsf{DFhe.Enc}(\mathsf{dpk}, m^*; \mathsf{r}))$, where $\mathsf{r} = (x_1, \ldots, x_n, \rho_1, \ldots, \rho_n)$ is sampled as follows:

1. Select $x_i \leftarrow \{0,1\}$ for $i \in [n]$ such that $\sum_{i\in[n]} x_i = m^* \pmod 2$.
2. For $i \in [n]$, if $x_i = 1$, select $\rho_i' \leftarrow \{0,1\}^\ell$
3. For $i \in [n]$, if $x_i = 0$, select $\rho_i' \leftarrow \mathcal{R}^{\ell_c}$

The statistical distance between the two distributions used to sample $(x_1, \ldots, x_n)$, in the honest case and in the intermediate/faking case, is $\frac{1}{\sqrt{n}}$. Hence, any PPT adversary $\mathcal{A}$ can win the $\mathsf{DnblGame}_{\mathcal{A}}^b(\lambda)$ game with probability at most $\frac{1}{\sqrt{n}}$, which is $\frac{1}{\delta}$ by our choice of n.

5 Weakly Deniable FHE with Large Message Space

In this section, we provide our construction for weak deniable FHE for polynomial size[4] message space $\mathcal{M}$, as in Definition 2.11. Let $\mathsf{Fhe} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ be a *special* public-key fully homomorphic encryption for the message space $\mathcal{M}$

[4] Polynomial in the security parameter. That is $|\mathcal{M}| = \mathrm{poly}(\lambda)$.

with ciphertext space $\mathcal{R}^{\ell_c}$, as in Definition 3.1, and $\mathsf{boot}(x)$ be the bootstrapping procedure, described in Definition 2.3. We denote by $\mathcal{S}$ a strict subset of the message space to which decryption of random elements is biased,[5] by $\mathbf{1}_{\overline{\mathcal{S}}}$ the indicator function for the set $\overline{\mathcal{S}} = \mathcal{M} \setminus \mathcal{S}$, described in Definition 2.7, and by s a fixed element in $\overline{\mathcal{S}}$. Recall that $\oplus_2$ denotes the homomorphic evaluation of addition mod 2 described in Definition 2.5 and select denotes the selector circuit described in Definition 2.6.

For reading convenience, we denote by lowercase r, the ℓ-bit string randomness that is input to an $\mathsf{Fhe.Enc}$ algorithm, and by upper case R, the elements in $\mathcal{R}^{\ell_c}$, where $\mathcal{R}^{\ell_c}$ is the co-domain of the FHE encryption algorithm. We denote by ℓ'_c the bit length of elements in $\mathcal{R}^{\ell_c}$ (that is, $\ell'_c = \lceil \ell_c \log_2(|R|) \rceil$). We index the messages in the message space as $\mathcal{M} = \{m_0, \ldots, m_\mu\}$.

Our (public-key) weakly deniable fully homomorphic encryption scheme for message space $\mathcal{M}$ $\mathsf{wDFhe} = (\mathsf{Gen}, \mathsf{DEnc}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec}, \mathsf{Fake})$ is described as follows:

$\mathsf{wDFhe.Gen}(1^\lambda)$: Upon input the unary representation of the security parameter λ, do the following:

1. Sample $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Fhe.Gen}(1^\lambda)$, and $\mathsf{ct}_{\mathsf{sk}} \leftarrow \mathsf{Fhe.Enc}(\mathsf{pk}, \mathsf{sk})$.
2. Outputs $\mathsf{dpk} := (\mathsf{pk}, \mathsf{ct}_{\mathsf{sk}}), \mathsf{dsk} := \mathsf{sk}$

$\mathsf{wDFhe.DEnc}(\mathsf{dpk}, m_k; \mathsf{r})$: Upon input the public key dpk, a message $m_k \in \mathcal{M}$ and $((4\ell + \ell'_c)\mu)$-bit string randomness r, do the following:

1. Parse the input.
   $\mathsf{dpk} := (\mathsf{pk}, \mathsf{ct}_{\mathsf{sk}}), \quad \mathsf{r} = (r_1, \ldots, r_\mu, (r_{1,1}, r_{1,2}, r_{1,3}, \hat{R}_{1,4}), \ldots, (r_{\mu,1}, r_{\mu,2}, r_{\mu,3}, \hat{R}_{\mu,4}))$
   where $|r_i| = |r_{i,j}| = \ell$ and $|\hat{R}_{i,4}| = \ell'_c$ for $i \in [\mu], j \in [3]$.
2. Generate ciphertexts for every possible message.
   For $i \in [\mu]$, set $\mathsf{ct}_i = \mathsf{Fhe.Enc}(\mathsf{pk}, m_i; r_i)$.
3. Generate ciphertexts for "selector" bits.
   (a) For every $i \in [\mu], j \in [3]$, set $\hat{R}_{i,j} = \mathsf{Fhe.Enc}(\mathsf{pk}, s; r_{i,j})$.
   (b) For every $i \in [\mu], j \in [4]$, set $R_{i,j} = \mathsf{Fhe.Eval}(\mathsf{pk}, \mathbf{1}_{\overline{\mathcal{S}}}, \hat{R}_{i,j})$.
   (c) We compute ciphertexts for selector bits 0 and 1 for every index as follows. For $i \in [\mu]$, compute
   $$\mathsf{ct}_0^i = \mathsf{boot}(R_{i,1}) \oplus_2 \mathsf{boot}(R_{i,2}), \quad \mathsf{ct}_1^i = \mathsf{boot}(R_{i,4}) \oplus_2 \mathsf{boot}(R_{i,3})$$
   (d) We let the k^{th} message to be selected by setting it's selector bit to 1, and all others to 0 as follows. For every $i \in [\mu]$ if $i \neq k$, set $\mathsf{ct}_i^{\mathsf{sel}} = \mathsf{ct}_0^i$; else if $i = k$, set $\mathsf{ct}_i^{\mathsf{sel}} = \mathsf{ct}_1^i$.
4. Evaluate selector circuit on ciphertexts.
   Compute and output $\mathsf{dct} = \mathsf{select}(\mathsf{ct}_1, \ldots, \mathsf{ct}_\mu, \mathsf{ct}_1^{\mathsf{sel}}, \ldots, \mathsf{ct}_\mu^{\mathsf{sel}})$, that is $\mathsf{dct} = \sum_{i \in [\mu]} (\mathsf{ct}_i^{\mathsf{sel}} \otimes \mathsf{ct}_i)$.

$\mathsf{wDFhe.Enc}(\mathsf{dpk}, m_k; \mathsf{r})$: Upon input public-key dpk, a message $m_k \in \mathcal{M}$, and $((2\ell + 3\ell'_c)\mu)$-bit string randomness r, do the following:

[5] Note that this exists from property 4 of the special Fhe.

1. Parse the input.
   $\mathsf{dpk} := (\mathsf{pk}, \mathsf{ct_{sk}})$, $\quad \mathsf{r} = (r_1, \ldots, r_\mu, (\hat{R}_{1,1}, \hat{R}_{1,2}, \hat{R}_{1,3}, r_{1,4}), \ldots, (\hat{R}_{\mu,1}, \hat{R}_{\mu,2}, \hat{R}_{\mu,3}, r_{\mu,4}))$
   where $|r_i| = |r_{i,4}| = \ell$ and $|\hat{R}_{i,j}| = \ell'_c$ for $i \in [\mu], j \in [3]$.
2. Generate ciphertexts for every possible message.
   For $i \in [\mu]$, set $\mathsf{ct}_i = \mathsf{Fhe.Enc}(\mathsf{pk}, m_i; r_i)$.
3. Generate ciphertexts for "selector" bits.
   (a) For every $i \in [\mu]$, set $\hat{R}_{i,4} = \mathsf{Fhe.Enc}(\mathsf{pk}, s; r_{i,4})$.
   (b) For every $i \in [\mu], j \in [4]$, set $R_{i,j} = \mathsf{Fhe.Eval}(\mathsf{pk}, \mathbf{1}_{\overline{\mathcal{S}}}, \hat{R}_{i,j})$.
   (c) We compute ciphertexts for selector bits 0 and 1 for every index as follows.
   For $i \in [\mu]$, compute
   $$\mathsf{ct}_0^i = \mathsf{boot}(R_{i,1}) \oplus_2 \mathsf{boot}(R_{i,2}), \quad \mathsf{ct}_1^i = \mathsf{boot}(R_{i,3}) \oplus_2 \mathsf{boot}(R_{i,4}).$$
   (d) We let the k^{th} message to be selected by setting it's selector bit to 1, and all others to 0 as follows. For every $i \in [\mu]$ if $i \neq k$, set $\mathsf{ct}_i^{\mathsf{sel}} = \mathsf{ct}_0^i$; else if $i = k$, set $\mathsf{ct}_i^{\mathsf{sel}} = \mathsf{ct}_1^i$.
4. Evaluate selector circuit on ciphertexts.
   Compute and output $\mathsf{dct} = \mathsf{select}(\mathsf{ct}_1, \ldots, \mathsf{ct}_\mu, \mathsf{ct}_1^{\mathsf{sel}}, \ldots, \mathsf{ct}_\mu^{\mathsf{sel}})$, that is $\sum_{i \in [\mu]} \left(\mathsf{ct}_i^{\mathsf{sel}} \otimes \mathsf{ct}_i\right)$.

$\mathsf{wDFhe.Eval}(\mathsf{dpk}, \mathcal{C}, \mathsf{dct}_1, \ldots, \mathsf{dct}_k)$: Upon input the public key $\mathsf{dpk} = (\mathsf{pk}, \mathsf{ct_{sk}})$, the circuit $\mathcal{C}$ and the ciphertexts $\mathsf{dct}_1, \ldots, \mathsf{dct}_k$, interpret dct_i as Fhe ciphertext ct_i for $i \in [k]$, and output $\mathsf{dct} = \mathsf{Fhe.Eval}(\mathsf{pk}, \mathcal{C}, \mathsf{ct}_1, \ldots, \mathsf{ct}_k)$.

$\mathsf{wDFhe.Dec}(\mathsf{dsk}, \mathsf{dct})$: Upon input the secret key dsk and the ciphertext dct, interpret dsk and dct as Fhe secret key sk and Fhe ciphertext ct and output $\mathsf{Fhe.Dec}(\mathsf{sk}, \mathsf{ct})$.

$\mathsf{wDFhe.Fake}(\mathsf{dpk}, m_k, \mathsf{r}, m_{k^*})$: Upon input the public key dpk, the original message $m_k \in \mathcal{M}$, $((4\ell + \ell_c)\mu)$-bit string randomness r and the fake message m_{k^*}, do the following:

1. Parse $\mathsf{dpk} := (\mathsf{pk}, \mathsf{ct_{sk}})$, and
   $\mathsf{r} := (r_1, \ldots, r_\mu, (r_{1,1}, r_{1,2}, r_{1,3}, \hat{R}_{1,4}), \ldots, (r_{\mu,1}, r_{\mu,2}, r_{\mu,3}, \hat{R}_{\mu,4}))$, where $|r_i| = |r_{i,j}| = \ell$ and $|\hat{R}_{i,4}| = \ell'_c$ for $i \in [\mu], j \in [3]$.
2. For all $i \in [\mu]$, set $r_i^* = r_i$.
3. For every $i \in [\mu], j \in [3]$, set $\hat{R}_{i,j} = \mathsf{Fhe.Enc}(\mathsf{pk}, s; r_{i,j})$.
4. For every $i \in [\mu] \setminus \{k, k^*\}$ set
   $$\hat{R}_{i,1}^* = \hat{R}_{i,1}, \quad \hat{R}_{i,2}^* = \hat{R}_{i,2}, \quad \hat{R}_{i,3}^* = \hat{R}_{i,3}, \quad r_{i,4}^* = r_{i,4}.$$
5. If $k = k^*$, then set
   $$\hat{R}_{k,1}^* = \hat{R}_{k,1}, \quad \hat{R}_{k,2}^* = \hat{R}_{k,2}, \quad \hat{R}_{k,3}^* = \hat{R}_{k,4}, \quad r_{k,4}^* = r_{k,3};$$
   Else if $k \neq k^*$, for every $i \in \{k, k^*\}$ set
   $$\hat{R}_{i,1}^* = \hat{R}_{i,4}, \quad \hat{R}_{i,2}^* = \hat{R}_{i,3}, \quad \hat{R}_{i,3}^* = \hat{R}_{i,1}, \quad r_{i,4}^* = r_{i,2}.$$

6. Output $\mathsf{r}^* = (r_1^*, \ldots, r_\mu^*, (\hat{R}_{1,1}^*, \hat{R}_{1,2}^*, \hat{R}_{1,3}^*, r_{1,4}^*), \ldots, (\hat{R}_{\mu,1}^*, \hat{R}_{\mu,2}^*, \hat{R}_{\mu,3}^*, r_{\mu,4}^*))$

Remark 5.1. We observe that by using the circuit Mux instead of the circuit select, we can use smaller randomness – in particular, we can achieve $|\mathsf{r}| = \mu\ell + 2\log_2(\mu)\ell_c'$.

In the full version [1], we prove the scheme satisfies correctness, compactness, CPA security and weak deniability. Due to space constraints, we provide our construction of compact public-key $1/\delta$-deniable fully homomorphic encryption scheme for polynomial sized message space in the full model in the full version of this paper [1].

Acknowledgments

We are grateful to Daniele Micciancio for very insightful discussions about bootstrapping, and helpful comments that helped us improve the quality of this writeup. We thank Vinod Vaikuntanathan and Aayush Jain for suggesting the use of a key-chain rather than key-cycle to get rid of circular security for the case of levelled FHE. Research of the first author is supported by the DST "Swarnajayanti" fellowship, an Indo-French CEFIPRA project and the CCD Centre of Excellence. Part of the research corresponding to this work was conducted while visiting the Simons Institute for the Theory of Computing. Research of the second author is supported in part by DARPA under Agreement No. HR00112020023. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Government or DARPA.

References

1. S. Agrawal, S. Goldwasser, and S. Mossel. Deniable fully homomorphic encryption from lwe. Cryptology ePrint Archive, Report 2020/1588, 2020. https://eprint.iacr.org/2020/1588.
2. P. Ananth and A. Jain. Indistinguishability obfuscation from compact functional encryption. In *Annual Cryptology Conference*, 2015.
3. Anonymous. Removing circularity for levelled fhe. Personal Communication, 2020.
4. D. Apon, X. Fan, and F.-H. Liu. Deniable attribute based encryption for branching programs from lwe. In *Theory of Cryptography Conference*, 2016.
5. R. Bendlin, J. B. Nielsen, P. S. Nordholt, and C. Orlandi. Lower and upper bounds for deniable public-key encryption. In *Asiacrypt*, 2011.
6. N. Bitansky and V. Vaikuntanathan. Indistinguishability obfuscation from functional encryption. *Journal of the ACM (JACM)*, 2018.
7. Z. Brakerski. Fully homomorphic encryption without modulus switching from classical gapsvp. In *Annual Cryptology Conference*, 2012.
8. Z. Brakerski. Fundamentals of fully homomorphic encryption. In *Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali*. 2019.

9. Z. Brakerski, N. Döttling, S. Garg, and G. Malavolta. Leveraging linear decryption: Rate-1 fully-homomorphic encryption and time-lock puzzles. In *Theory of Cryptography Conference*, 2019.
10. Z. Brakerski, C. Gentry, and V. Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. *ACM Transactions on Computation Theory (TOCT)*, 2014.
11. Z. Brakerski and V. Vaikuntanathan. Efficient fully homomorphic encryption from (standard) lwe. *SIAM Journal on Computing*, 2014.
12. Z. Brakerski and V. Vaikuntanathan. Lattice-based fhe as secure as pke. In *ITCS*, 2014.
13. R. Canetti, C. Dwork, M. Naor, and R. Ostrovsky. Deniable encryption. In *Crypto*, 1997.
14. R. Canetti, S. Park, and O. Poburinnaya. Fully deniable interactive encryption. In D. Micciancio and T. Ristenpart, editors, *Crypto*, 2020.
15. I. Chillotti, N. Gama, M. Georgieva, and M. Izabachène. A homomorphic lwe based e-voting scheme. In *Post-Quantum Cryptography*, 2016.
16. D. Dachman-Soled. On minimal assumptions for sender-deniable public key encryption. In *PKC*, 2014.
17. D. Dachman-Soled, J. Katz, and V. Rao. Adaptively secure, universally composable, multiparty computation in constant rounds. In *TCC*, 2015.
18. A. De Caro, V. Iovino, and A. O'Neill. Deniable functional encryption. In *PKC*, 2016.
19. S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. *SIAM Journal on Computing*, 2016.
20. S. Garg, O. Pandey, A. Srinivasan, and M. Zhandry. Breaking the sub-exponential barrier in obfustopia. In *Eurocrypt*, 2017.
21. C. Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, 2009. crypto.stanford.edu/craig.
22. C. Gentry and S. Halevi. Compressible fhe with applications to pir. In *TCC*, 2019.
23. C. Gentry, A. Sahai, and B. Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In *Annual Cryptology Conference*, 2013.
24. S. Goldwasser, Y. Kalai, R. A. Popa, V. Vaikuntanathan, and N. Zeldovich. Reusable garbled circuits and succinct functional encryption. In *STOC*, 2013.
25. A. Jain, H. Lin, and A. Sahai. Indistinguishability obfuscation from well-founded assumptions. In *STOC*, 2021.
26. H. Lin, R. Pass, K. Seth, and S. Telang. Indistinguishability obfuscation with non-trivial efficiency. In *PKC*, 2016.
27. B. Meng. A secure internet voting protocol based on non-interactive deniable authentication protocol and proof protocol that two ciphertexts are encryption of the same plaintext. *J. Networks*, 2009.
28. A. O'Neill, C. Peikert, and B. Waters. Bi-deniable public-key encryption. In *Crypto*, 2011.
29. A. Sahai and B. Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In *STOC*, 2014.
30. N. P. Smart and F. Vercauteren. Fully homomorphic encryption with relatively small key and ciphertext sizes. In *PKC*, 2010.