Deniable Fully Homomorphic Encryption

Shweta Agrawal* Shafi Goldwasser† Saleet Mossel‡

December 19, 2020

Abstract

We introduce the notion of *Deniable Fully Homomorphic Encryption* and provide constructions based on the circular-secure Learning With Errors polynomial hardness assumption. Deniable fully homomorphic encryption offers a compelling upgrade of deniable public key encryption suitable for the motivating applications of deniability, such as prevention of vote-buying in electronic voting schemes where encrypted votes can be tallied without decryption, or storing encrypted data in the cloud, to be processed securely, in a deniable way. Our constructions enjoy *deniability compactness*, namely both the size of the public key and the size of the ciphertext of our schemes can be bounded by a fixed polynomial, independent of the level of deniability (or faking probability) achieved by the scheme. Additionally, our constructions support large message spaces and are well suited to an 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 leads to a very efficient online phase, whose running time is independent of the faking probability, whereas the offline encryption run-time grows with the inverse of the faking probability.

In contrast, all prior constructions even in the context of deniable *public key encryption* without homomorphic properties, encoded large messages bit by bit, where the ciphertext for each bit grew inversely with the faking probability. Indeed, all previous constructions from polynomial hardness assumptions have both the public key and ciphertext size that grows with the inverse of the faking probability achieved by the scheme. This limitation dates back to the seminal work of Canetti, Dwork, Naor and Ostrovsky (CRYPTO 1997) which introduced the notion of deniable encryption, and has been inherited by all subsequent work (excepting one by Sahai and Waters (STOC 2013) which is based on indistinguishability obfuscation[1]). Indeed Canetti *et al.* argued that this dependence "seems inherent". Our constructions imply deniable public key encryption with deniability compactness, showing that this dependence is not inherent. However, the running time of our encryption algorithm *does* depend on the inverse of the faking probability, thus falling short of achieving simultaneously negligible deniability and polynomial encryption time.

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] Note that indistinguishability obfuscation (iO) is not a polynomial hardness assumption. iO is a non-falsifiable assumption, and *all* provably secure constructions of iO (including recent breakthroughs) from polynomial assumptions require a subexponential loss – please see [GPSZ17] for a discussion.

1 Introduction

Deniable encryption [CDNO97] is a paradoxical primitive that allows an encryptor to lie about which message she encrypted, if later coerced to "open" a published ciphertext. In more detail, suppose an encryptor has published a ciphertext ct, and is later forced by an adversary to reveal the randomness and message chosen during encryption. Deniable encryption allows her to reveal *fake* random coins, which convincingly explain the published ciphertext ct as the encryption of *any message chosen at the time of coercion*. Aside from being a fundamental primitive, deniable encryption has many important applications. Consider the scenario of vote buying in elections: if the voter encrypts her vote using deniable encryption, then upon being forced to open the ciphertext, she can claim she encrypted an alternate message. This capability makes vote selling ineffective, since there is no way for the seller to verify the compliance of the voter. Additionally, it encourages honest voting, since the voter may be reassured that she cannot be forced to reveal her choice. Another early motivation for deniable encryption is to store encrypted data in a deniable way [CDNO97].

In this work, we introduce the notion of *deniable fully homomorphic encryption* and provide the first constructions based on the circular-secure Learning With Errors assumption. In deniable FHE, the encryptor can produce ciphertexts that not only support homomorphism, but can additionally be opened to fake messages under coercion. Evidently, for all the applications of deniable public key encryption, adding the capability of homomorphism is a much needed upgrade – several modern e-voting protocols use FHE [CGGI16, Men09], and present-day encrypted data is often stored on a server which assists the data owner with computing "blind-folded" via FHE [Gen09].

Our constructions enjoy *deniability compactness*, namely the public key and 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 faking probability) achieved by the scheme. Additionally, our constructions support large message spaces and are well suited to the 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. In contrast, all prior constructions encoded large messages bit by bit, where the ciphertext for each bit grew inversely with the faking probability.

Furthermore, even in the context of deniable *public key encryption*, all previous constructions from polynomial hardness assumptions[2] have public key and ciphertext size that grows with the inverse of the faking probability achieved by the scheme. This limitation dates back to the seminal work of Canetti, Dwork, Naor and Ostrovsky (CRYPTO 1997) which introduced the notion of deniable encryption, and has been inherited by all subsequent work from polynomial assumptions. Indeed, Canetti *et al.* argued that this dependence "seems inherent". Our constructions imply deniable public key encryption with deniability compactness, showing that this dependence is not inherent. However, the running time of our encryption algorithm *does* depend on the inverse of the faking probability, thus falling short of achieving negligible deniability.

1.1 Prior Work.

The notion of deniable encryption was introduced by Canetti *et al.* (henceforth CDNO) who provided elegant constructions based on the construct of so called "translucent sets", which in turn can be constructed from trapdoor permutations. They also provided other extensions – the notion of

[2] Note that this excludes indistinguishability obfuscation, which is an inherently subexponential assumption, please see [GPSZ17] for a discussion.

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 produce a fake secret key that decrypts the message to an alternate one[3]. Moreover, in the weak model, [CDNO97] 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, we refer the reader to [OPW11] for a discussion.

Interestingly, Canetti *et al.* also provided a lower bound that shows that their construction in the "full" (i.e. not weak) model is in some sense optimal. In more detail, 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 public key and ciphertext size on the inverse of the faking probability "seems inherent" [CDNO97]. Thus, to achieve the desired negligible deniability, the public key and ciphertext size of any separable scheme would be required to grow super-polynomially in the security parameter, ruling out a large class of natural constructions.

Other Related Work. Subsequent work explored extensions to CDNO – O'Neill, Peikert and Waters [OPW11] provided the first constructions of non-interactive *bi*-deniable encryption schemes where both the sender and the receiver can fake simultaneously. Their constructions rely generically on simulatable public key encryption (in the weak model) or *bi*-translucent sets (in both the full and weak models, using the CDNO transformation). They showed how to construct bi-translucent sets from a variant of the Learning With Errors assumption (LWE) and also provided the first construction of identity based bi-deniable encryption. Apon, Fan and Liu [AFL16] extended their results to provide the first construction of *attribute based* translucent sets which in turn led to the first construction of deniable attribute based encryption, also from LWE. We note that in the full model, both works [OPW11, AFL16] inherit the faking probability of CDNO, which is inverse polynomial. De Caro, Iovino and O'Neill [DCIO16] 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) [AJ15, BV18][4].

Aside from work extending the functionality of deniable encryption, there was also progress in lower bounds – for receiver deniability, [BNNO11] showed that a non-interactive public-key scheme having key size δ can be fully receiver-deniable only with non-negligible $\Omega(\frac{1}{\delta})$ faking probability while for sender deniability, Dachman-Soled [DS14] 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 [CPP20]. Our focus in this work is the non-interactive setting.

A significant step forward in our understanding of deniable encryption was achieved via the breakthrough work of Sahai and Waters in 2014 [SW14] which provided the first construction achieving negligible deniability assuming indistinguishability obfuscation (iO) and one way functions. However, iO is an inherently sub-exponential assumption [GGH+16], and while exciting as a feasibility result, does not provide a satisfying solution to the question of efficient constructions of deniable encryption from standard assumptions.

We note that barring the iO based construction, all constructions of (fully) sender deniable

[3] When unspecified, deniability refers to "sender" deniability, as in this work.

[4] We note that functional encryption for circuits is a polynomial hardness assumption.

encryption from standard assumptions (that we are aware of) have used the translucent set paradigm dating from 1997, and suffer from a public key and ciphertext size that is inversely proportional to the faking probability. Evidently, for any faking probability that could be considered reasonable in practice, this implies a prohibitively large blow on efficiency, relegating these constructions to being interesting "only in theory". For a primitive as fundamental and interesting as deniable encryption, this state of affairs is very dissatisfying.

1.2 Our Results.

Deniable FHE. We introduce the notion of (public key, sender) *deniable fully homomorphic encryption* which consists 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 faking 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 2.10.

We extend this definition to the so-called *weak* model (Definition 2.13) – 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 [CDNO97], 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 , i.e. $\mathsf{DEnc}(\mathsf{pk}, m; r) = \mathsf{Enc}(\mathsf{pk}, m^*; r^*)$.

We provide the first constructions of deniable FHE in both the full and weak model, based on the circular-secure Learning With Errors (LWE) assumption [Gen09, BV14, BGV14, GSW13]. In more detail, we construct:

1. A *weakly* deniable FHE scheme for bits with negligible faking 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 faking probability (Section 4.2). We also extend this scheme to support larger (polynomial sized) message spaces (Section 6). Both our fully deniable FHE schemes have compact public key and ciphertext, i.e. with size

independent of the faking probability, but with encryption running time that grows with the inverse of the faking probability.

3. *Plan-ahead* deniable FHE schemes which support exponentially large message spaces (Section 6.1). Plan-ahead deniable encryption [CDNO97] 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, 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.

Compact Deniable PKE from FHE. As discussed above, our construction of fully deniable FHE implies, as a special case, a deniable public key encryption scheme, where the size of the public key and ciphertext are *compact* and independent of the faking 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 faking probability. Moreover, we show that this dependence is inherent, since our constructions can be shown to be separable in the sense of CDNO and hence subject to the lower bound. 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 faking 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 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 [SW14], 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 [Gen09]. 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 a ciphertext – roughly speaking, the FHE ciphertext contains some noise terms which are necessary for security but must be kept bounded below some threshold for correctness. This noise grows as the ciphertext participates in homomorphic evaluations, and must be somehow reduced after it reaches a threshold to support further homomorphic computation.

The neat observation that allows to use bootstrapping for reducing noise is that the decryption circuit of an FHE scheme is quite shallow, so if we run the decryption circuit homomorphically on some FHE ciphertext ct using the encryption of the FHE secret key $\mathsf{ct}_{\mathsf{sk}}$, the noise contained in ct is removed by decryption, and the noise in output ciphertext ct' can be bound depending on the depth of the decryption circuit and the noise in $\mathsf{ct}_{\mathsf{sk}}$.

However, 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 discuss this assumption below). Then, an approach to encrypt in the weak model is as follows.

Let the bootstrapping procedure be denoted by Btsp. 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:

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

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:

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

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 faking 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 faking probability. We proceed to describe the main ideas in the construction.

Let δ be the inverse of the desired faking 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

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

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 $\mathsf{Btsp}(R_j)$ 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. 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 [BGV14] 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. 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 [SV10, BGV14, BDGM19, GH19], and bringing deniable FHE closer to practice.

Let $\mathcal{M}$ be the message space of an FHE scheme Fhe such that $|\mathcal{M}| = \text{poly}(\lambda)$. 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 Sections 5 and 6 for details.

Plan-Ahead Deniability. Plan-ahead deniable encryption [CDNO97] 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 Section 6.1 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 according to the encryption algorithm. Next, we may compute the homomorphic evaluation of the bootstrapping circuit on the $\delta - 1$ random elements, i.e. $\mathsf{Btsp}(R_i)$ for $i \in [\delta - 1]$. 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: $\mathsf{ct} = \mathsf{ct}_{\mathsf{offline}} \oplus_2 \mathsf{Btsp}(R_\delta)$, where

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

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 faking 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 faking 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.

Lower Bound. Canetti *et al.* [CDNO97] showed that no one round (sender) deniable scheme which satisfies a certain structural property called "separability", can enjoy negligible faking probability, (denoted by $\frac{1}{\delta}$). In the original as well as subsequent work, this led to schemes with public key and ciphertext size which grows linearly in δ, which the authors remark "seems to be inherent" [CDNO97, Section 4]. As discussed above, our constructions achieve public key and ciphertext size that is independent of δ, showing that this dependence is not inherent. Nevertheless, the *running time* of the encryption algorithm is linear in δ, and this dependence *is* inherent. Perhaps somewhat surprisingly, despite the assumption of circular-secure LWE, our schemes can be shown to be separable in the sense of [CDNO97] and therefore subject to the dependence of the running time on δ. Please see Section 7 for details.

1.4 Perspective and Open Problems.

Barriers from non-Obfuscation Assumptions. Despite achieving compact public key and ciphertext, our constructions do not achieve negligible deniability. We briefly discuss here the technical barriers encountered in achieving negligible deniability using non-obfuscation assumptions.

At a high level, the Sahai-Waters construction [SW14] based on iO, works by obfuscating the encryption algorithm as well as the faking algorithm. Recall that in deniable encryption, the faking algorithm is required to output the fake randomness (rand^*, say) that is used to explain a ciphertext ct^* as encrypting a fake message m^*. The obfuscated explain algorithm simply takes in a ciphertext and message pair (ct^*, m^*) and outputs a pseudorandom encoding of these as rand^*. The encrypt algorithm, upon receiving a message m and randomness rand^*, first checks for a "hidden sparse trigger", namely whether rand^* is an encoding of some pair (ct^*, m^*) that was output by the faking algorithm. It also checks whether $m = m^*$, and outputs ct^* if these two conditions hold. If not, and rand^* looks like genuine randomness, it proceeds to encrypt as usual.

Showing that the above idea can be made to work by relying only on iO (and one way functions), and not virtual black box obfuscation, must overcome several hurdles and requires multiple innovative techniques, but these are not relevant for the present discussion. Here, we only draw attention to two relevant facts: first, since the encryption and faking algorithms are obfuscated, they can share secrets such as PRF keys. In contrast, without using obfuscation, the encryption and faking algorithms are public and cannot share any secrets, making any co-ordination between them significantly harder. Second, in the above construction, the tuple $(\mathsf{ct}^*, \mathsf{rand}^*, m^*)$ are not required to satisfy any structural/algebraic relation of well-formedness. In more detail, the ciphertext ct^* and message m^* need not be related in any way, and the the only property that rand^* must satisfy is to "tie together" this unrelated ciphertext and message pair via a pseudorandom encoding. These unrelated objects can be made to appear related by triggering a trapdoor mode which is hidden in the encryption procedure by the amazing power of obfuscation.

Without relying on obfuscation, it is significantly more difficult to design an encryption algorithm with compact randomness so that the structural relationship of well-formedness holds for a single ciphertext with respect to multiple messages. All constructions of non-interactive sender deniable encryption in the full model known from 1997 to date, achieve this by relying on the trick of providing multiple elements in the ciphertext, both pseudorandom and random, and encoding the message bit in the parity of the number of pseudorandom elements as described previously. However, since the ability to pretend is only in one direction, namely, pseudorandom to random, this approach is inherently restricted to polynomial deniability as formalized by the "separability" argument of [CDNO97].

Using fully homomorphic encryption, bootstrapping let's us obliviously sample ciphertexts, and FHE evaluation enables us to "compactify" the final ciphertext. However, we remain stuck with being able to only cheat in a single direction – we can pretend that FHE ciphertexts are random but not the other way around. This leads to the unfortunate fact that the number of pseudorandom elements is always smaller in the fake algorithm than in the real algorithm, and this can be formalized into an attack. Thus, as we show in Section 7, our constructions of deniable FHE are also separable in the sense of CDNO and hence the limitation of polynomial (and not negligible) deniability is inherent. Overcoming these barriers, using our techniques or otherwise, is a fascinating open problem.

On Receiver Deniability. We briefly discuss here the prospect of constructing receiver deniable FHE. The notion of receiver deniablity allows the receiver to decrypt a received ciphertext to an alternate message by using a fake secret key, which is *derived specifically for that particular "challenge" ciphertext*. However, the fake secret key must nevertheless correctly decrypt all other ciphertexts to their honest messages. It is easy to see that inability to do so leads to a distinguishing attack – the adversary may itself encrypt messages of its choice and use the fake secret key to open them.

Due to these requirements on the fake secret key, it is unclear whether the notion of receiver deniability is meaningful in the context of FHE. To see the conundrum, consider an adversary, who receives a challenge ciphertext ct^* along with a fake secret key sk^* which falsely decrypts ct^* to m^*. Since ct^* is an FHE ciphertext, it could be the result of evaluating some circuit on some other FHE ciphertexts, or it could be the input ciphertext used in homomorphic operations to generate other evaluated ciphertexts. Let us say that ct^* participates in multiple homomorphic evaluations, say of circuits $\mathcal{C}_1, \dots, \mathcal{C}_n$ to yield outputs $\mathsf{ct}_1, \dots, \mathsf{ct}_n$. Then, given the input and output ciphertexts, the adversary can decrypt these and test whether the circuits $\mathcal{C}_1, \dots, \mathcal{C}_n$ applied to the input messages yield the output messages. To avoid a distinguishing attack, the fake key sk^* should decrypt the input and output ciphertexts to messages consistent with the fake message m^*, which implies that i) these ciphertexts cannot be decrypted honestly in general, violating one of the conditions discussed above, and ii) even if we modify the condition and allow faking for some non-challenge ciphertexts, it may not be possible to find fake messages consistent with the multiple, arbitrary dependencies imposed by the circuits.

Hence, to define receiver deniability in the context of FHE, it appears necessary to restrict an adversary's view to exclude all ciphertexts which are related to the challenge ciphertext ct^* . However, this restriction seems hard to justify in practice. For example, it appears infeasible to control which ciphertexts are obtained by an adversary when encrypted data is stored on the cloud. Due to these difficulties, we do not consider receiver deniabile FHE in this work.

2 Preliminaries

In this section, we define the notation and preliminaries that we require in this work.

Notation. Let $\mathcal{A}(x; r)$ denote the randomized algorithm $\mathcal{A}$ run on input x, using randomness r. We let $\overline{m}$ denote the complement of bit m. We denote by $[n]$ the set $\{1, ..., n\}$. If X is a random variable, a probability distribution, or a randomized algorithm we let $x \leftarrow X$ denote the process of sampling x according to X. If $\mathcal{X}$ is a set, we let $x \leftarrow \mathcal{X}$ denote the process of sampling x uniformly at random from $\mathcal{X}$.

We say a function $f(\lambda)$ is *negligible* if it is $O(\lambda^{-c})$ for all $c > 0$, and we use $\mathsf{negl}(\lambda)$ to denote a negligible function of λ. We say $f(\lambda)$ is *polynomial* if it is $O(\lambda^c)$ for some constant $c > 0$, and we use $\mathrm{poly}(\lambda)$ to denote a (positive) polynomial function of λ. We say that an event occurs with *overwhelming* probability in λ if it occurs with probability $1 - \mathsf{negl}(\lambda)$. Where evident from context, we sometimes use f to denote $f(\lambda)$.

Definition 2.1 (Statistical Distance)**.** Let P and Q be two distributions over a finite set $\mathcal{U}$. The statistical distance is define as

$$\mathsf{SD}(P, Q) := \frac{1}{2} \sum_{x \in \mathcal{U}} |P(x) - Q(x)|.$$

2.1 Fully Homomorphic Encryption

Definition 2.2 (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 ciphertexts 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 2.3 (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 2.4 (Bootstrapping Procedure)**.** [Gen09] 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 $\mathsf{Btsp} : \mathcal{R}^{\ell_c} \to \mathcal{R}^{\ell_c}$, as

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

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 2.5 (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 2.6 (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 2.7 (Multiplexer)**.** A multiplexer is a deterministic procedure that selects between several inputs using "selector" bits. In more detail, on input $x_0, \ldots, x_k$, and $b_1, \ldots, b_t$ where $k < 2^t$, and $b_i \in \{0, 1\}$, outputs x_j where $j = \sum_{i=1}^{t} 2^{i-1} b_i$. Let Fhe be a public-key FHE scheme for message space $\mathcal{M}$, we denote by Mux the homomorphic evaluation of the multiplexer (data selector) circuit. That is for $k < 2^t$,

$$\mathsf{Mux}(\mathsf{ct}_0, \ldots, \mathsf{ct}_k, \mathsf{ct}'_1, \ldots, \mathsf{ct}'_t) = \mathsf{ct},$$

ct is a valid encryption of the selected message x_j where $j = \sum_{i=1}^{t} 2^{i-1} b_i$, ct'_i is a valid encryption of $b_i \in \{0, 1\}$ for $i \in [t]$, and ct_i is a valid encryption of $x_i \in \mathcal{M}$ for $i \in [k]$.

Definition 2.8 (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 2.9 (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 2.10 (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:

- $\mathsf{Gen}(1^\lambda) \to (\mathsf{dpk}, \mathsf{dsk})$: on input the unary representation of the security parameter λ, generates a public-key dpk and a secret-key dsk.
- $\mathsf{Enc}(\mathsf{dpk}, m; r) \to \mathsf{ct}$: on input a public-key dpk and a message $m \in \mathcal{M}$, uses ℓ-bit string randomness r, outputs a ciphertexts dct.
- $\mathsf{Eval}(\mathsf{dpk}, \mathcal{C}, \mathsf{dct}_1, \ldots, \mathsf{dct}_k) \to \mathsf{dct}$: 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 2.2. 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 2.11. 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 $(\mathsf{dpk}, \mathsf{dsk}) \leftarrow \mathsf{Gen}(1^\lambda)$.

Remark 2.12. The above definition can be modified to capture a compact deniable public key encryption scheme by removing the evaluation algorithm required by FHE.

Definition 2.13 (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 2.10, 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 2.2.

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

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

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 Btsp, defined in 2.4, 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 $\mathsf{Btsp}(\mathcal{R}^{\ell_c})$ is computationally indistinguishable from $\mathcal{R}^{\ell_c}$, where Btsp is the bootstrapping procedure as in Definition 2.4.

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 Btsp is always a valid ciphertext (Definition 2.5).

4. *Circular Secure.* The scheme Fhe is circular secure as in Definition 2.3.

5. *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 Section 8 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 $0 \in \mathcal{S}$.

3.1 Instantiation

For concreteness, we instantiate our special FHE scheme with (a modified version of) the scheme by Brakerski, Gentry and Vaikuntanathan [BGV14] (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 4) 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 [BGV14, 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 5). Establishing this property requires slight modifications to the BGV scheme[5]. 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 [BGV14, 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[6]. 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

[5] We note that these properties are also satisfied by several other FHE schemes, for instance [BV14, Bra12, GSW13].

[6] 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.

$\mathsf{ct} = \mathbf{m} + \mathbf{A}^\top \mathbf{r}$. 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 [BGV14, 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 $R \leftarrow \mathcal{R}^{n+1}$, 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 5. Below, we will describe the modification to BGV to achieve the strong version of property 5. In Section 8, 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 2.5), 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 [Gen09].

Let us assume that the decryption error is bounded above by $B - 1$, for some $B = \text{poly}(\lambda)$. We note that this bound holds true for the current setting of parameters in [BGV14]. Then, it follows that the output of step 1 of decryption, for any well formed ciphertext can be bounded from above by B. 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 [BGV14] 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 Section 8, we describe how to relax this requirement.

Next, we discuss how to modify the BGV scheme supporting larger (polynomial) message spaces, as discussed in [BGV14, 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 5 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 $P(m) := \Pr[\mathsf{Fhe.Dec}(\mathsf{sk}, x) = m]$ 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 Section 8 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 [BGV14, 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 2.13, and compact deniable FHE, as in Definition 2.10. 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 Btsp denotes the bootstrapping procedure described in Definition 2.4 and $\oplus_2$ denotes the homomorphic evaluation of addition mod 2 described in Definition 2.6.

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 $\mathsf{dpk} := (\mathsf{pk}, \mathsf{ct}_{\mathsf{sk}}), \mathsf{dsk} := \mathsf{sk}$

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

1. Parse $\mathsf{dpk} := (\mathsf{pk}, \mathsf{ct}_{\mathsf{sk}})$ and $\mathsf{rand} = (r_1, r_2, r_3, R_4)$, 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 $\mathsf{ct}_0 = \mathsf{Btsp}(R_1) \oplus_2 \mathsf{Btsp}(R_2)$ and $\mathsf{ct}_1 = \mathsf{Btsp}(R_4) \oplus_2 \mathsf{Btsp}(R_3)$.
4. Output $\mathsf{dct} = \mathsf{ct}_m$.

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

1. Parse $\mathsf{dpk} := (\mathsf{pk}, \mathsf{ct}_{\mathsf{sk}})$ and $\mathsf{rand} = (R_1, R_2, R_3, r_4)$, 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 $\mathsf{ct}_0 = \mathsf{Btsp}(R_1) \oplus_2 \mathsf{Btsp}(R_2)$ and $\mathsf{ct}_1 = \mathsf{Btsp}(R_3) \oplus_2 \mathsf{Btsp}(R_4)$.
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 $\mathsf{Fhe.Dec}(\mathsf{sk}, \mathsf{ct})$.

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

1. Parse $\mathsf{dpk} := (\mathsf{pk}, \mathsf{ct}_{\mathsf{sk}})$ and $\mathsf{rand} = (r_1, r_2, r_3, R_4)$, 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 $\mathsf{rand}^* = (R_1^*, R_2^*, R_3^*, r_4^*)$

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 Btsp 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 4 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 $\mathsf{rand} \in \{0,1\}^{3\ell+\ell'_c}$ as $\mathsf{rand} = (r_1, r_2, r_3, R_4)$. 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 5 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 $\mathsf{Btsp}(R_4)$ is a valid encryption of 0 with overwhelming probability.

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

Next we prove correctness of $\mathsf{wDFhe.Enc}$. Parse $\mathsf{rand} \in \{0,1\}^{\ell+3\ell'_c}$ as $\mathsf{rand} = (R_1, R_2, R_3, r_4)$. 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 5 that $\mathsf{Btsp}(R_i)$ for $i \in [3]$ are valid encryptions of 0 with overwhelming probability.

Thus, again by correctness of FHE evaluation, we have that $\mathsf{ct}_0 = \mathsf{Btsp}(R_1) \oplus_2 \mathsf{Btsp}(R_2)$ is a valid encryption of 0 and $\mathsf{ct}_1 = \mathsf{Btsp}(R_3) \oplus_2 \mathsf{Btsp}(R_4)$ 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 $(\mathsf{dpk}, m^*, \mathsf{rand}, \mathsf{DEnc}(\mathsf{dpk}, m; \mathsf{rand}'))$ in the case of faking.

1. Select uniformly at random $\mathsf{rand}' \leftarrow \{0,1\}^{3\ell} \times \mathcal{R}^{\ell_c}$.
2. Parse $\mathsf{rand}' := (r_1, r_2, r_3, R_4)$, 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 $\mathsf{rand}^* = \mathsf{wDFhe.Fake}(\mathsf{dpk}, m, \mathsf{rand}', m^*)$.
5. By the faking algorithm $\mathsf{rand}^* = \left(R_1^*, R_2^*, R_3^*, r_4^*\right)$ 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:

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

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

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

where $\mathsf{ct}^* := \mathsf{ct}_{m^*}$, $\mathsf{ct}_0 = \mathsf{Btsp}(R_1^*) \oplus_2 \mathsf{Btsp}(R_2^*)$, $\mathsf{ct}_1 = \mathsf{Btsp}(R_3^*) \oplus_2 \mathsf{Btsp}(R_4^*)$ 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 $\mathsf{rand} \leftarrow \mathcal{R}^{3\ell_c} \times \{0,1\}^\ell$, so the output distribution is:

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

where $\mathsf{ct}^* := \mathsf{ct}_{m^*}$, $\mathsf{ct}_0 = \mathsf{Btsp}(R_1) \oplus_2 \mathsf{Btsp}(R_2)$, $\mathsf{ct}_1 = \mathsf{Btsp}(R_3) \oplus_2 \mathsf{Btsp}(R_4)$ 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[7] 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 Appendix A. Recall that Btsp denotes the bootstrapping procedure described in Definition 2.4 and $\oplus_2$ denotes the homomorphic evaluation of addition mod 2 described in Definition 2.6). 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 rand as follows:
   (a) Select uniformly $x_1, \ldots, x_n \in \{0,1\}$ such that $\sum_{i=1}^n x_i = m \pmod 2$.
   (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 $\mathsf{dct} = \oplus_2(\mathsf{Btsp}(R_1), \ldots, \mathsf{Btsp}(R_n))$

$\mathsf{DFhe.Eval}(\mathsf{dpk}, \mathcal{C}, \mathsf{dct}_1, \ldots, \mathsf{dct}_k)$: 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 $\mathsf{dct} = \mathsf{Fhe.Eval}(\mathsf{pk}, \mathcal{C}, \mathsf{ct}_1, \ldots, \mathsf{ct}_k)$.

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

$\mathsf{DFhe.Dec}(\mathsf{dsk}, \mathsf{dct})$: 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 $\mathsf{Fhe.Dec}(\mathsf{sk}, \mathsf{ct})$.

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

1. If $m = m^*$, output $\mathsf{rand}^* = \mathsf{rand}$.
2. Parse $\mathsf{dpk} := (\mathsf{pk}, \mathsf{ct}_{\mathsf{sk}})$ and $\mathsf{rand} = (x_1, \ldots, x_n, \rho_1, \ldots, \rho_n)$, 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 $\mathsf{rand}^* = (x^*_1, \ldots, x^*_n, \rho^*_i, \ldots, \rho^*_n)$.

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 $\mathsf{Btsp}(R_i)$ are valid encryptions of 1 for all $i \in [n]$ such that $x_i = 1$.

2. By properties 3 and 5 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 $\mathsf{Btsp}(R_i)$ 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 $\mathsf{Btsp}(R_i)$ 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 rand^* is equal to the original randomness rand. Thus in this case, $m = m^*$, the distributions are identical:

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

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 $(\mathsf{dpk}, m^*, \mathsf{rand}^*, \mathsf{DFhe.Enc}(\mathsf{dpk}, m; \mathsf{rand}))$ in the case of faking, where $\mathsf{rand}^* \leftarrow \mathsf{DFhe.Fake}(\mathsf{dpk}, m, \mathsf{rand}; m^*)$.

1. Select uniform $\mathsf{rand} := (x_1, \ldots, x_n, \rho_1, \ldots, \rho_n)$, 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 $\mathsf{rand}^* = \mathsf{DFhe.Fake}(\mathsf{dpk}, m, \mathsf{rand}, m^*)$, that is $\mathsf{rand}^* = (x_1^*, \ldots, x_n^*, \rho_1^*, \ldots, \rho_n^*)$ 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 $(\mathsf{dpk}, m^*, \mathsf{rand}', \mathsf{DFhe.Enc}(\mathsf{dpk}, m; \mathsf{rand}))$, where $\mathsf{rand}' = (x_1', \ldots, x_n', \rho_1', \ldots, \rho_n')$ 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 $(\mathsf{dpk}, m^*, \mathsf{rand}, \mathsf{DFhe.Enc}(\mathsf{dpk}, m^*; \mathsf{rand}))$, where $\mathsf{rand} = (x_1, \ldots, x_n, \rho_1, \ldots, \rho_n)$ 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[8] message space $\mathcal{M}$, as in Definition 2.13. Let $\mathsf{Fhe} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ be a *special* public-key fully homomorphic encryption for the message space $\mathcal{M}$ with ciphertext space $\mathcal{R}^{\ell_c}$, as in Definition 3.1, and $\mathsf{Btsp}(x)$ be the bootstrapping procedure, described in Definition 2.4. We denote by $\mathcal{S}$ a strict subset of the message space to which decryption of random elements is biased,[9] by $\mathbf{1}_{\overline{\mathcal{S}}}$ the indicator function for the set $\overline{\mathcal{S}} = \mathcal{M} \setminus \mathcal{S}$, described in Definition 2.9, 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 2.6 and select denotes the selector circuit described in Definition 2.8.

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

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

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 $\mathsf{dpk} := (\mathsf{pk}, \mathsf{ct}_{\mathsf{sk}}), \mathsf{dsk} := \mathsf{sk}$

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

1. Parse the input.

   $\mathsf{dpk} := (\mathsf{pk}, \mathsf{ct}_{\mathsf{sk}})$, $\quad \mathsf{rand} = (r_1, \ldots, r_\mu, (r_{1,1}, r_{1,2}, r_{1,3}, \hat{R}_{1,4}), \ldots, (r_{\mu,1}, r_{\mu,2}, r_{\mu,3}, \hat{R}_{\mu,4}))$ 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

   $$\mathsf{ct}^i_0 = \mathsf{Btsp}(R_{i,1}) \oplus_2 \mathsf{Btsp}(R_{i,2}), \quad \mathsf{ct}^i_1 = \mathsf{Btsp}(R_{i,4}) \oplus_2 \mathsf{Btsp}(R_{i,3})$$

   (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 $\mathsf{ct}^{\mathsf{sel}}_i = \mathsf{ct}^i_0$; else if $i = k$, set $\mathsf{ct}^{\mathsf{sel}}_i = \mathsf{ct}^i_1$.
4. Evaluate selector circuit on ciphertexts.

   Compute and output $\mathsf{dct} = \mathsf{select}(\mathsf{ct}_1, \ldots, \mathsf{ct}_\mu, \mathsf{ct}^{\mathsf{sel}}_1, \ldots, \mathsf{ct}^{\mathsf{sel}}_\mu)$, that is $\mathsf{dct} = \sum_{i \in [\mu]} \left(\mathsf{ct}^{\mathsf{sel}}_i \otimes \mathsf{ct}_i \right)$.

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

1. Parse the input.

   $\mathsf{dpk} := (\mathsf{pk}, \mathsf{ct}_{\mathsf{sk}})$, $\quad \mathsf{rand} = (r_1, \ldots, r_\mu, (\hat{R}_{1,1}, \hat{R}_{1,2}, \hat{R}_{1,3}, r_{1,4}), \ldots, (\hat{R}_{\mu,1}, \hat{R}_{\mu,2}, \hat{R}_{\mu,3}, r_{\mu,4}))$ 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

$$\mathsf{ct}_0^i = \mathsf{Btsp}(R_{i,1}) \oplus_2 \mathsf{Btsp}(R_{i,2}), \quad \mathsf{ct}_1^i = \mathsf{Btsp}(R_{i,3}) \oplus_2 \mathsf{Btsp}(R_{i,4}).$$

(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 $\mathsf{dpk} = (\mathsf{pk}, \mathsf{ct}_{\mathsf{sk}})$, 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 $\mathsf{Fhe.Dec}(\mathsf{sk}, \mathsf{ct})$.

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

1. Parse $\mathsf{dpk} := (\mathsf{pk}, \mathsf{ct}_{\mathsf{sk}})$, and $\mathsf{rand} := (r_1, \ldots, r_\mu, (r_{1,1}, r_{1,2}, r_{1,3}, \hat{R}_{1,4}), \ldots, (r_{\mu,1}, r_{\mu,2}, r_{\mu,3}, \hat{R}_{\mu,4}))$, 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 $\mathsf{rand}^* = (r_1^*, \ldots, r_\mu^*, (\hat{R}_{1,1}^*, \hat{R}_{1,2}^*, \hat{R}_{1,3}^*, r_{1,4}^*), \ldots, (\hat{R}_{\mu,1}^*, \hat{R}_{\mu,2}^*, \hat{R}_{\mu,3}^*, r_{\mu,4}^*))$

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 $|\mathsf{rand}| = \mu\ell + 2\log_2(\mu)\ell_c'$.

We now prove the scheme satisfies correctness, compactness, CPA security and weak deniability. As in Section 4.1, compactness and security follow from those of the underlying FHE scheme. We argue correctness and weak deniability next.

Correctness. We start by proving correctness of the deniable encryption algorithm $\mathsf{wDFhe.DEnc}$. Parse $\mathsf{dpk} := (\mathsf{pk}, \mathsf{ct_{sk}}), \mathsf{dsk} := \mathsf{sk}$, and $\mathsf{rand} \in \{0,1\}^{\mu(4\ell+\ell'_c)}$ as

$$\mathsf{rand} = (r_1, \ldots, r_\mu, (r_{1,1}, r_{1,2}, r_{1,3}, \hat{R}_{1,4}), \ldots, (r_{\mu,1}, r_{\mu,2}, r_{\mu,3}, \hat{R}_{\mu,4})),$$

where $|r_i| = |r_{i,j}| = \ell$ and $|\hat{R}_{i,4}| = \ell'_c$ for $i \in [\mu], j \in [3]$. Observe that:

1. Since $\mathsf{ct}_i = \mathsf{Fhe.Enc}(\mathsf{pk}, m_i; r_i)$ for $i \in [\mu]$, we have by correctness of the underlying scheme Fhe, that ct_i is a valid encryption of m_i for every $i \in [\mu]$.

2. Since $\hat{R}_{i,j} = \mathsf{Fhe.Enc}(\mathsf{pk}, s; r_{i,j})$, we have by correctness of the underlying scheme Fhe, that $\hat{R}_{i,j}$ is a valid encryption of s for $s \notin \mathcal{S}, i \in [\mu]$, and $j \in [3]$.

3. By correctness of the underlying scheme Fhe, we have that $R_{i,j} = \mathsf{Fhe.Eval}(\mathsf{pk}, \mathbf{1}_{\overline{\mathcal{S}}}, \hat{R}_{i,j})$ is a valid encryption of 1 for every $i \in [\mu], j \in [3]$. Thus, also $\mathsf{Btsp}(R_{i,j})$ is a valid encryption of 1.

4. By correctness of the underlying scheme Fhe and the properties 3 and 5 which state that FHE decryption always outputs a valid ciphertext for some message $m \in \mathcal{M}$, and $m \in \mathcal{S}$ with overwhelming probability when decryption is invoked with a truly random input, we have that $R_{i,4} = \mathsf{Fhe.Eval}(\mathsf{pk}, \mathbf{1}_{\overline{\mathcal{S}}}, \hat{R}_{i,4})$ is a valid encryption of 0 with overwhelming probability for every $i \in [\mu]$. Thus, for every $i \in [\mu]$, $\mathsf{Btsp}(R_{i,4})$ is also a valid encryption of 0 with similar probability.

Now, by correctness of FHE evaluation, we have that $\mathsf{ct}_0^i = \mathsf{Btsp}(R_{i,1}) \oplus_2 \mathsf{Btsp}(R_{i,2})$ is a valid encryption of 0 and $\mathsf{ct}_1^i = \mathsf{Btsp}(R_{i,4}) \oplus_2 \mathsf{Btsp}(R_{i,3})$ is a valid encryption of 1 for every $i \in [\mu]$. Thus, for every $i \in [\mu], i \neq k$, $\mathsf{ct}_i^{\mathsf{sel}}$ is a valid encryption of 0, and for $i = k$, $\mathsf{ct}_k^{\mathsf{sel}}$ is a valid encryption of 1. This implies that the output $\mathsf{dct} = \sum_{i \in [\mu]} \mathsf{ct}_i^{\mathsf{sel}} \otimes \mathsf{ct}_i$ is a valid encryption of the message m_k.

Next, we prove correctness of the encryption algorithm $\mathsf{wDFhe.Enc}$. Parse $\mathsf{dpk} := (\mathsf{pk}, \mathsf{ct_{sk}})$, $\mathsf{dsk} := \mathsf{sk}$, and $\mathsf{rand} \in \{0,1\}^{\mu(2\ell+3\ell'_c)}$ as

$$\mathsf{rand} = (r_1, \ldots, r_\mu, (\hat{R}_{1,1}, \hat{R}_{1,2}, \hat{R}_{1,3}, r_{1,4}), \ldots, (\hat{R}_{\mu,1}, \hat{R}_{\mu,2}, \hat{R}_{\mu,3}, r_{\mu,4})),$$

where where $|r_i| = |r_{i,4}| = \ell$ and $|\hat{R}_{i,j}| = \ell'_c$ for $i \in [\mu], j \in [3]$. Observe that, as in the proof of correctness for DEnc, we have that:

1. For every $i \in [\mu]$, $\mathsf{ct}_i = \mathsf{Fhe.Enc}(\mathsf{pk}, m_i; r_i)$ is a valid encryption of m_i.

2. For every $i \in [\mu]$, $\hat{R}_{i,4} = \mathsf{Fhe.Enc}(\mathsf{pk}, s; r_{i,4})$ is a valid encryption of s for $s \notin \mathcal{S}$.

3. For every $i \in [\mu]$, $R_{i,4} = \mathsf{Fhe.Eval}(\mathsf{pk}, \mathbf{1}_{\overline{\mathcal{S}}}, \hat{R}_{i,4})$ is a valid encryption of 1.

4. For every $i \in [\mu], j \in [3]$, $R_{i,j} = \mathsf{Fhe.Eval}(\mathsf{pk}, \mathbf{1}_{\overline{\mathcal{S}}}, \hat{R}_{i,j})$ is a valid encryption of 0 with overwhelming probability. Thus, $\mathsf{Btsp}(R_{i,j})$ is also a valid encryption of 0 with overwhelming probability.

Now, by correctness of FHE evaluation, we have that $\mathsf{ct}_0^i = \mathsf{Btsp}(R_{i,1}) \oplus_2 \mathsf{Btsp}(R_{i,2})$ is a valid encryption of 0 and $\mathsf{ct}_1^i = \mathsf{Btsp}(R_{i,3}) \oplus_2 \mathsf{Btsp}(R_{i,4})$ is a valid encryption of 1 for every $i \in [\mu]$. Thus, for every $i \in [\mu], i \neq k$, $\mathsf{ct}_i^{\mathsf{sel}}$ is a valid encryption of 0, and for $i = k$, $\mathsf{ct}_k^{\mathsf{sel}}$ is a valid encryption of 1. This implies that the output $\mathsf{dct} = \sum_{i \in [\mu]} \mathsf{ct}_i^{\mathsf{sel}} \otimes \mathsf{ct}_i$ is a valid encryption of the message m_k.

Weak Deniability. Next, we prove weak deniability of the construction. Fix a security parameter λ, an original message $m_k \in \mathcal{M}$ and a fake message $m_{k^*} \in \mathcal{M}$. 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 $(\mathsf{dpk}, m_{k^*}, \mathsf{rand}^*, \mathsf{DEnc}(\mathsf{dpk}, m_k; \mathsf{rand}'))$ in the case of faking.

1. Select uniformly at random $\mathsf{rand}' \leftarrow \{0,1\}^{4\mu\ell} \times \mathcal{R}^{\mu\ell_c}$.
2. Parse $\mathsf{rand}' := (r_1, \ldots, r_\mu, (r_{1,1}, r_{1,2}, r_{1,3}, \hat{R}_{1,4}), \ldots, (r_{\mu,1}, r_{\mu,2}, r_{\mu,3}, \hat{R}_{\mu,4}))$, where $|r_i| = |r_{i,j}| = \ell$ and $|\hat{R}_{i,4}| = \ell'_c$ for $i \in [\mu], j \in [3]$.
3. For $i \in [\mu], j \in [3]$, set $\hat{R}_{i,j} = \mathsf{Fhe.Enc}(pk, s; r_{i,j})$.
4. Let $\mathsf{rand}^* = \mathsf{wDFhe.Fake}(\mathsf{dpk}, m_k, \mathsf{rand}', m_{k^*})$
5. By the faking algorithm $\mathsf{rand}^* = (r_1^*, \ldots, r_\mu^*, (\hat{R}_{1,1}^*, \hat{R}_{1,2}^*, \hat{R}_{1,3}^*, r_{1,4}^*), \ldots, (\hat{R}_{\mu,1}^*, \hat{R}_{\mu,2}^*, \hat{R}_{\mu,3}^*, r_{\mu,4}^*))$ which is computed as follows:
   (a) For every $i \in [\mu]$, set $r_i^* = r_i$.
   (b) For every $i \in [\mu] \setminus \{k, k^*\}, j \in [4]$, set $\hat{R}_{1,j}^* = \hat{R}_{1,j}, r_{i,4}^* = r_{i,4}$.
   (c) For every $i \in \{k, k^*\}$, set
      i. Case $k = k^*$:
      $$\hat{R}_{k,1}^* = \hat{R}_{k,1}, \quad \hat{R}_{k,2}^* = \hat{R}_{k,2}, \quad \hat{R}_{k,3}^* = \hat{R}_{k,4}, \quad r_{k,4}^* = r_{k,3}.$$
      By property 2 which asserts that ciphertexts are pseudorandom, we can explain $\hat{R}_{i,1}^*$ and $\hat{R}_{i,2}^*$ as uniform from the ciphertexts space $\mathcal{R}^{\ell_c}$. Here $\hat{R}_{i,3}^*$ is already a uniform element in $\mathcal{R}^{\ell_c}$, and $r_{i,4}^*$ is a uniform ℓ bit string. Hence, we can explain $\mathsf{rand}^* \leftarrow \{0,1\}^{2\mu\ell} \times \mathcal{R}^{3\mu\ell_c}$.
      ii. Case $k \neq k^*$:
      $$\hat{R}_{i,1}^* = \hat{R}_{i,4}, \quad \hat{R}_{i,2}^* = \hat{R}_{i,3}, \quad \hat{R}_{i,3}^* = \hat{R}_{i,1}, \quad r_{i,4}^* = r_{i,2}.$$
      As above, we can explain $\hat{R}_{i,2}^*$ and $\hat{R}_{i,3}^*$ as uniform element in $\mathcal{R}^{\ell_c}$, and $\hat{R}_{i,1}^*$, and $r_{i,4}^*$ are already uniform. Hence, we can explain $\mathsf{rand}^* \leftarrow \{0,1\}^{2\mu\ell} \times \mathcal{R}^{3\mu\ell_c}$.
6. The output of this hybrid is:
   $$(\mathsf{dpk}, m_{k^*}, \mathsf{rand}^*, \mathsf{ct}^* = \mathsf{wDFhe.DEnc}(\mathsf{dpk}, m_k; \mathsf{rand}')).$$
   Observe that $\mathsf{ct}^* = \mathsf{wDFhe.Enc}(\mathsf{dpk}, m_{k^*}; \mathsf{rand}^*)$. Thus the output of this hybrid can be written as:
   $$(\mathsf{dpk}, m_{k^*}, \mathsf{rand}^*, \mathsf{ct}^* = \mathsf{wDFhe.Enc}(\mathsf{dpk}, m_{k^*}; \mathsf{rand}^*)).$$

Honest Case. Next, note that in the honest case $\mathsf{rand} \leftarrow \{0,1\}^{2\mu\ell} \times \mathcal{R}^{3\mu\ell_c}$, so the output distribution is:
$$(\mathsf{dpk}, m_{k^*}, \mathsf{rand}, \mathsf{ct}^* = \mathsf{wDFhe.Enc}(\mathsf{dpk}, m_{k^*}; \mathsf{rand})).$$

Hence, the two distributions are indistinguishable.

6 Fully Deniable FHE with Large Message Space

In this section, we construct a compact public-key $1/\delta$-deniable[10] fully homomorphic encryption scheme for polynomial sized message space $\mathcal{M}$, as in Definition 2.10. Let $\mathsf{Fhe} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec})$ be a *special* fully homomorphic encryption scheme for the message space $\mathcal{M}$ with ciphertext $\mathcal{R}^{\ell_c}$, as in Definition 3.1. Again, we let $\mathsf{Btsp}(x)$ be the bootstrapping procedure, described in Definition 2.4. We denote by $\mathcal{S}$ a strict subset of the message space to which decryption of random element is biased,[11] by $\mathbf{1}_{\overline{\mathcal{S}}}$ the indicator function for the set $\overline{\mathcal{S}} := \mathcal{M} \setminus \mathcal{S}$, described in Definition 2.9, and by $s \in \overline{\mathcal{S}}$ a fixed element in $\overline{\mathcal{S}}$. Recall that $\oplus_2$ denotes the homomorphic evaluation of addition mod 2 described in Definition 2.6) and select denotes the selector circuit described in Definition 2.8. We let $n = \delta^2$

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) compact $1/\delta$-deniable fully homomorphic encryption scheme for message space $\mathcal{M}$ $\mathsf{DFhe} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Dec}, \mathsf{Fake})$ is described as follows:

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

$\mathsf{DFhe.Enc}(\mathsf{dpk}, m_k)$: Upon input the public-key dpk and a message $m_k \in \mathcal{M}$, do the following:

1. <u>Parse the input.</u>
   $\mathsf{dpk} := (\mathsf{pk}, \mathsf{ct_{sk}})$.
2. <u>Select randomness</u>.
   Select rand as follow:
   (a) Select uniform ℓ-bit strings r_i for $i \in [\mu]$.
   (b) For each $i \in [\mu]$ do:
      i. If $i \neq k$: select uniformly $x_{i,1}, \ldots, x_{i,n} \in \{0,1\}$ s.t. $\sum_{j=1}^{n} x_{i,j} = 0 \pmod 2$.
      ii. Else if $i = k$: select uniformly $x_{k,1}, \ldots, x_{k,n} \in \{0,1\}$ s.t. $\sum_{j=1}^{n} x_{k,j} = 1 \pmod 2$.
      iii. For every $j \in [n]$: if $x_{i,j} = 1$, then select $r_{i,j} \leftarrow \{0,1\}^\ell$; else if $x_{i,j} = 0$, select $\hat{R}_{i,j} \leftarrow \mathcal{R}^{\ell_c}$.
3. <u>Generate ciphertexts for every possible message.</u>
   Let $\mathsf{ct}_i = \mathsf{Fhe.Enc}(\mathsf{pk}, m_i; r_i)$ for $i \in [\mu]$.
4. <u>Generate ciphertext for "selector" bits.</u>
   (a) For each $i \in [\mu], j \in [n]$ such that $x_{i,j} = 1$, let $\hat{R}_{i,j} = \mathsf{Fhe.Enc}(\mathsf{pk}, s; r_{i,j})$.
   (b) We compute ciphertexts for selector bits for each $i \in [\mu]$ as follows.
      i. For each $j \in [n]$, set $R_{i,j} = \mathsf{Fhe.Eval}(\mathsf{pk}, \mathbf{1}_{\overline{\mathcal{S}}}, \hat{R}_{i,j})$.

[10] We remind the reader that $\delta = \delta(\lambda)$, but we drop the λ for readability.
[11] Note that this exists from property 5 of the special Fhe.

ii. Let $\mathsf{ct}_i^{\mathsf{sel}} = \oplus_2 (\mathsf{Btsp}(R_{i,1}), \dots, \mathsf{Btsp}(R_{i,n}))$.

5. Evaluate selector circuit on ciphertexts.
Compute and output $\mathsf{dct} = \mathsf{select}(\mathsf{ct}_1, \dots, \mathsf{ct}_\mu, \mathsf{ct}_i^{\mathsf{sel}}, \dots, \mathsf{ct}_i^{\mathsf{sel}})$, that is $\mathsf{dct} = \sum_{i\in[\mu]} \left(\mathsf{ct}_i^{\mathsf{sel}} \otimes \mathsf{ct}_i\right)$.

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

$\mathsf{DFhe.Dec}(\mathsf{dsk}, \mathsf{dct})$: 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 $\mathsf{Fhe.Dec}(\mathsf{sk}, \mathsf{ct})$.

$\mathsf{DFhe.Fake}(\mathsf{dpk}, m_k, \mathsf{rand}, m_{k^*})$: Upon input the public key dpk, the original message $m_k \in \mathcal{M}$, the randomness rand, and the faking messages m_{k^*} do the following:

1. If $k = k^*$, output rand.
2. Parse $\mathsf{dsk} := (\mathsf{pk}, \mathsf{ct}_{\mathsf{sk}})$, and the randomness as:
$\mathsf{rand} = (r_1, \dots, r_\mu, (x_{1,1}, \dots, x_{1,n}, \rho_{1,1}, \dots, \rho_{1,n}), \dots, (x_{\mu,1}, \dots, x_{\mu,n}, \rho_{\mu,1}, \dots, \rho_{\mu,n}))$, where $|r_i| = \ell$, and $x_{i,j} \in \{0,1\}$, if $x_{i,j} = 1$ then $|\rho_{i,j}| = \ell$; else if $x_{i,j} = 0$, $|\rho_{i,j}| = \ell'_c$ for every $i \in [\mu], j \in [n]$.
3. Set $r_i^* = r_i$ for all $i \in [\mu]$.
4. For every $i \in [\mu] \setminus \{k, k^*\}$, set $x_{i,j}^* = x_{i,j}$ and $\rho_{i,j}^* = \rho_{i,j}$ for all $j \in [n]$.
5. For $i \in \{k, k^*\}$ do:
   (a) Select uniform $j_i^* \in [n]$ such that $x_{i,j_i^*} = 1$
      i. Set $x_{i,j_i^*}^* = 0$, and $\rho_{i,j_i^*}^* = \mathsf{Fhe.Enc}(\mathsf{pk}, s; \rho_{i,j_i^*})$.
      ii. For every $j \in [n] \setminus \{j_i^*\}$: set $x_{i,j}^* = x_{i,j}$ and $\rho_{i,j}^* = \rho_{i,j}$.
6. Output $\mathsf{rand}^* = (r_1^*, \dots, r_\mu^*, (x_{1,1}^*, \dots, x_{1,n}^*, \rho_{1,1}^*, \dots, \rho_{1,n}^*), \dots, (x_{\mu,1}^*, \dots, x_{\mu,n}^*, \rho_{\mu,1}^*, \dots, \rho_{\mu,n}^*))$.

Remark 6.1. We observe that by using the circuit Mux instead of the circuit select, we can use smaller randomness - in particular, we can achieve $|\mathsf{rand}| = \mu\ell + \delta^2 \log_2(\mu)(1 + \ell'_c)$.[12]

Online-Offline Encryption. It is easy to see that the only step in the encryption algorithm whose running time depends on the detection probability is the step that computes the ciphertexts for the selector bits, namely step 4. Since for every valid ciphertext, the number of selector bits encoding 0 are $|\mathcal{M}| - 1$ and there is a single selector bit encoding 1, these bits can be computed offline. Moreover, the ciphertexts encoding every possible message in $\mathcal{M}$ can also be constructed offline. Only the final step of evaluating the selector circuit based on the selected message, i.e. step 5 needs be performed after the message becomes available in the online phase. The running time of this step depends on $|\mathcal{M}|$ but not on the detection probability of the scheme.

We also remark that the dependence of the online computation time on $|\mathcal{M}|$ may be mitigated by evaluating the selector circuit with *all* selector bits set to 0 in the offline phase, and storing the selector ciphertexts. An extra selector ciphertext ct^1 for the bit 1 is also computed and stored, to be used in the online phase. Then, in the online phase when the message is known, the precomputed

[12] Here we assume w.l.o.g that $\ell \le \ell'_c$, namely the randomness used by the Fhe encryption algorithm is at most the size of the output ciphertext.

sum can be adjusted by subtracting out the incorrect term and adding in the correct one. In more detail, in the offline phase, the encryptor can perform the homomorphic evaluation of the function $\sum_{m_i \in \mathcal{M}} 0 \cdot m_i$, namely with all the selector bits set to 0, and store the selector bit ciphertexts in a table. In the online phase, when the message m_k (say) is known, it can subtract the wrongly deselected term $\mathsf{ct}_k^0 \cdot \mathsf{ct}_k$ and add the term $\mathsf{ct}^1 \cdot \mathsf{ct}_k$ to obtain the correct ciphertext. Note that here, ct^1 is the extra selector bit computed in the offline phase.

Compactness and Security. As in Section 4.1, compactness and security follow from those of the underlying FHE scheme. We argue correctness, polynomial deniability, and deniability compactness next.

Correctness. Parse $\mathsf{dpk} := (\mathsf{pk}, \mathsf{ct}_{\mathsf{sk}}), \mathsf{dsk} := \mathsf{sk}$, and rand as
$\mathsf{rand} = (r_1, \ldots, r_\mu, (x_{1,1}, \ldots, x_{1,n}, \rho_{1,1}, \ldots, \rho_{1,n}), \ldots, (x_{\mu,1}, \ldots, x_{\mu,n}, \rho_{\mu,1}, \ldots, \rho_{\mu,n}))$,
where $|r_i| = \ell$, $x_{i,j} \in \{0,1\}$, if $x_{i,j} = 0$, then $|\rho_{i,j}| = \ell'_c$; else if $x_{i,j} = 1$, then $|\rho_{i,j}| = \ell$; for $i \in [\mu], j \in [n]$.

Observe that:

1. Since $\mathsf{ct}_i = \mathsf{Fhe}(\mathsf{pk}, m_i; r_i)$ for $i \in [\mu]$, we have by correctness of the underlying scheme Fhe, that ct_i is a valid encryption of m_i for every $i \in [\mu]$.

2. For every $i \in [\mu]$, $j \in [n]$ such that $x_{i,j} = 1$, since $\hat{R}_{i,j} = \mathsf{Fhe.Enc}(\mathsf{pk}, s; r_{i,j})$, we have by correctness of the underlying scheme Fhe that $\hat{R}_{i,j}$ is a valid encryption of s.

3. For every $i \in [\mu]$, $j \in [n]$ such that $x_{i,j} = 1$, by correctness of the underlying Fhe, we have that $R_{i,j} = \mathsf{Fhe.Eval}(\mathsf{pk}, \mathbf{1}_{\overline{\mathcal{S}}}, \hat{R}_{i,j})$ is a valid encryption of 1. Thus, also $\mathsf{Btsp}(R_{i,j})$ is a valid encryption of 1.

4. For every $i \in [\mu]$, $j \in [n]$ such that $x_{i,j} = 0$, by correctness of the underlying Fhe and the properties 3 and 5 which state that FHE decryption always outputs a valid ciphertext for some message $m \in \mathcal{M}$, and $m \in \mathcal{S}$ with overwhelming probability when decryption is invoked with a truly random input, we have that $R_{i,j} = \mathsf{Fhe.Eval}(\mathsf{pk}, \mathbf{1}_{\overline{\mathcal{S}}}, \hat{R}_{i,j})$ is a valid encryption of 0 with overwhelming probability. Thus, also $\mathsf{Btsp}(R_{i,j})$ is a valid encryption of 0 with overwhelming probability.

5. For every $i \in [\mu], j \in [n]$, since $\mathsf{Btsp}(R_{i,j})$ is a valid encryption of $x_{i,j}$, we have by correctness of the underlying Fhe that $\mathsf{ct}_i^{\mathsf{sel}}$ is a valid encryption of $\sum_{j \in [n]} x_{i,j} \pmod 2$ which is 0 for every $i \neq k$, and 1 for $i = k$.

Hence, the output $\mathsf{dct} = \sum_{i \in [\mu]} \mathsf{ct}_i^{\mathsf{sel}} \otimes \mathsf{ct}_i$, is a valid encryption of the message m_k.

Deniability. Next, we prove polynomial deniability of the construction. Fix a security parameter λ, an original message $m_k \in \mathcal{M}$, and a faking message $m_{k^*} \in \mathcal{M}$. 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_k and the fake message m_{k^*} are the same, the fake randomness rand^* is equal to the original randomness rand. Thus in this case, $k = k^*$, the distributions are identical:

$$(\mathsf{dpk}, m_{k^*}, \mathsf{rand}, \mathsf{DFhe.Enc}(\mathsf{dpk}, m_{k^*}; \mathsf{rand})) = (\mathsf{dpk}, m_{k^*}, \mathsf{rand}^*, \mathsf{DFhe.Enc}(\mathsf{dpk}, m_k; \mathsf{rand}))$$

When the original message m_k and the faked message m_{k^*} are not the same, observe that the only difference between the randomnesses rand and rand^* is in the randomness used in the encryption algorithm for encrypting the k and k^* selector bits, i.e. in computing $\mathsf{ct}_k^{\mathsf{sel}}$ and $\mathsf{ct}_{k^*}^{\mathsf{sel}}$ which is sampled independent of the messages m_k, m_k^*. Moreover, all other randomness is selected independent of the randomness used for $\mathsf{ct}_k^{\mathsf{sel}}$ and $\mathsf{ct}_{k^*}^{\mathsf{sel}}$. Thus, in the proof below we will only write the parts of the distribution that involve the randomness used for encrypting the k and k^* selector bits, that is:

$$(x_{k,1}, \ldots, x_{k,n}, \rho_{k,1}, \ldots, \rho_{k,n}, x_{k^*,1}, \ldots, x_{k^*,n}, \rho_{k^*,1}, \ldots, \rho_{k^*,n}).$$

The proof is very similar to the proof of polynomial deniability for our public-key compact deniable fully homomorphic encryption scheme for bits described in Section 4.2. We provide it here for the sake of completeness.

Faking Case. First consider the distribution in the case of faking, where

$(x^*_{k,1}, \ldots, x^*_{k,n}, \rho^*_{k,1}, \ldots, \rho^*_{k,n}, x^*_{k^*,1}, \ldots, x^*_{k^*,n}, \rho^*_{k^*,1}, \ldots, \rho^*_{k^*,n})$ is sampled as follows:

1. For $i \in [n]$, select uniformly at random $x^*_{k,i}, x^*_{k^*,i} \leftarrow \{0,1\}$ such that $\sum_{i\in[n]} x^*_{k,i} = 1 \pmod 2; \quad \sum_{i\in[n]} x^*_{k^*,i} = 0 \pmod 2$.
2. Select uniform indexes $i_k^*, i_{k^*}^* \in [n]$ such that $x^*_{k,i_k^*} = 1$ and $x^*_{k^*,i_{k^*}^*} = 1$.
3. Set $x^*_{k,i_k^*} = 0$ and $x^*_{k^*,i_{k^*}^*} = 0$.
4. Set $\rho^*_{k,i_k^*} = \mathsf{Fhe.Enc}(\mathsf{pk}, s; \rho_{k,i_k^*})$ and $\rho^*_{k^*,i_{k^*}^*} = \mathsf{Fhe.Enc}(\mathsf{pk}, s; \rho_{k^*,i_{k^*}^*})$ where $\rho_{k,i_k^*}, \rho_{k^*,i_{k^*}^*}$ are random ℓ bit strings.
5. For $j \in \{k, k^*\}, i \in [n], i \neq i_j^*$, if $x^*_{j,i} = 1$, select $\rho^*_{j,i} \leftarrow \{0,1\}^\ell$.
6. For $j \in \{k, k^*\}, i \in [n], i \neq i_j^*$, if $x^*_{j,i} = 0$, select $\rho^*_{j,i} \leftarrow \mathcal{R}^{\ell_c}$.

Intermediate Case. By property 2 of the special FHE, which asserts that ciphertexts are pseudorandom, we can explain $\rho^*_{k,i_k^*} = \mathsf{Fhe.Enc}(\mathsf{pk}, s; \rho_{k,i_k^*})$ and $\rho^*_{k^*,i_{k^*}^*} = \mathsf{Fhe.Enc}(\mathsf{pk}, s; \rho_{k^*,i_{k^*}^*})$ as uniform elements from the ciphertexts space $\mathcal{R}^{\ell_c}$.

In this hybrid,

$(x'_{k,1}, \ldots, x'_{k,n}, \rho'_{k,1}, \ldots, \rho'_{k,n}, x'_{k^*,1}, \ldots, x'_{k^*,n}, \rho'_{k^*,1}, \ldots, \rho'_{k^*,n})$ is sampled as follows:

1. For $i \in [n]$, select uniformly at random $x'_{k,i}, x'_{k^*,i} \leftarrow \{0,1\}$ for $i \in [n]$ such that $\sum_{i\in[n]} x'_{k,i} = 1 \pmod 2; \quad \sum_{i\in[n]} x'_{k^*,i} = 0 \pmod 2$.
2. Select uniform indexes $i_k^*, i_{k^*}^* \in [n]$ such that $x'_{k,i_k^*} = 1$ and $x'_{k^*,i_{k^*}^*} = 1$.
3. Set $x'_{k,i_k^*} = 0$ and $x'_{k^*,i_{k^*}^*} = 0$.
4. For $j \in \{k, k^*\}, i \in [n]$, if $x'_{j,i} = 1$, select $\rho^*_{j,i} \leftarrow \{0,1\}^\ell$.
5. For $j \in \{k, k^*\}, i \in [n]$, if $x'_{j,i} = 0$, select $\rho^*_{j,i} \leftarrow \mathcal{R}^{\ell_c}$.

Honest Case. In this hybrid,

$(x_{k,1}, \ldots, x_{k,n}, \rho_{k,1}, \ldots, \rho_{k,n}, x_{k^*,1}, \ldots, x_{k^*,n}, \rho_{k^*,1}, \ldots, \rho_{k^*,n})$ is sampled as follows:

1. For $i \in [n]$, select uniformly at random $x_{k,i}, x_{k^*,i} \leftarrow \{0,1\}$ for $i \in [n]$ such that $\sum_{i\in[n]} x_{k,i} = 1 \pmod 2; \quad \sum_{i\in[n]} x_{k^*,i} = 0 \pmod 2$.

2. For $j \in \{k, k^*\}, i \in [n]$, if $x_i = 1$, select $\rho_i \leftarrow \{0,1\}^\ell$
3. For $j \in \{k, k^*\}, i \in [n]$, if $x_i = 0$, select $\rho_i \leftarrow \mathcal{R}^{\ell_c}$

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

Deniability Compactness. Fix a security parameter λ, and a message $m \in \mathcal{M}$. Observe that the output of the encryption algorithm is a ciphertext of the underlying Fhe scheme, namely $\mathsf{DFhe.Enc}(\mathsf{dpk}, m) \in \mathcal{R}^{\ell_c}$ where $(\mathsf{dpk}, \mathsf{dsk}) \leftarrow \mathsf{DFhe.Gen}(1^\lambda)$. Hence, it follows from the compactness of Fhe that the ciphertext also satisfies deniability compactness.

6.1 Plan Ahead Deniability.

Plan-ahead deniable encryption [CDNO97] requires the sender to choose all possible fake messages at the time of encryption. For the plan-ahead setting, we can instantiate the underlying FHE to support message spaces of exponential size. Intuitively, without the plan-ahead restriction, the above construction 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, then it is easy to check that exact same construction as above is a plan-ahead deniable encryption scheme, provided we can instantiate the special FHE to have an exponentially large message space. As discussed in Section 3, to support message spaces of exponential size [BGV14, Section 5], i.e. $|\mathcal{M}| = 2^\lambda$, we can set $\mathcal{S} = \mathcal{M} \setminus \{1\}$. This ensures that $\mathcal{S}$ has an efficient representation and that the output is biased to $\mathcal{S}$ with overwhelming probability, as desired.

7 Lower Bound for Deniable Schemes

As discussed in Section 1, Canetti *et al.* [CDNO97] (denoted by CDNO) showed that no one round (sender) deniable scheme which satisfies a certain structural property called "separability", can enjoy negligible detection probability, say $\frac{1}{\delta}$. While our constructions (Sections 4.2 and 6) achieve deniability compactness, where the size of the public key and ciphertext do not depend on δ, we show here that these schemes are *separable* in the sense of CDNO and hence the dependence of the encryption running time on δ is inherent. This implies that our schemes cannot achieve negligible deniability without incurring super-polynomial running time.

Separable Schemes. In a separable scheme, the decryption key is a trapdoor that allows the holder to distinguish a pseudorandom element from random. In CDNO, the ciphertext consists of a sequence of elements $R_1, \ldots, R_n$ where R_i for $i \in [n]$ may be random or pseudorandom, and distinguishing between the two cases is hard given public information. To encrypt a bit b, the encryptor samples uniform random bits $x_1, \ldots, x_n$ such that $\sum_{i \in [n]} x_i = b \pmod 2$. It then computes n elements $R_1, \ldots, R_n$ of which, R_i is pseudorandom when $x_i = 1$, and R_i is random when $x_i = 0$. To fake, it samples a random $j \in [n]$ such that $x_j = 1$, sets $x^*_j = 0$, and $x^*_i = x_i$ for every $i \neq j, i \in [n]$. It pretends that R_j is chosen uniformly at random – this flips the parity of $\sum_{i \in [n]} x^*_i \pmod 2$ and hence the presumed encoded message.

CDNO provide an attack against any separable scheme which claims to enjoy negligible detection probability. The attack is based on the observation that the faking algorithm always *decreases* the number of claimed pseudorandom elements – in particular, one may pretend that pseudorandom is random, but one cannot pretend in the opposite direction. Hence, for any bit b, the adversary can compute the expected number of elements in $[n]$ which ought to be pseudorandom. If the claimed number of pseudorandom elements is below the expected value, the adversary decides that the sender is lying. They show that this strategy succeeds with probability $\Omega(\frac{1}{n})$.

Separability of Our Schemes. Our schemes can be seen as following a similar philosophy of separability as above, but with compactification of the public key and ciphertext using FHE. For concreteness, let us consider the construction from Section 4.2 that achieves polynomial deniability for bits in the full model. Here, to encrypt a bit b, the encryptor samples uniform random bits $x_1, \ldots, x_n$ such that $\sum_{i \in [n]} x_i = b \pmod 2$. It then computes n elements $R_1, \ldots, R_n$ 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

$$\mathsf{ct} = \mathsf{Btsp}(R_1) \oplus_2 \mathsf{Btsp}(R_2) \oplus_2 \ \ldots \ \oplus_2 \mathsf{Btsp}(R_n)$$

To fake, it samples a random $j \in [n]$ such that $x_j = 1$, sets $x_j^* = 0$, and $x_i^* = x_i$ for every $i \neq j, i \in [n]$. It pretends that R_j is chosen uniformly at random, implying that $\mathsf{Btsp}(R_j)$ encodes 0 with overwhelming probability.

For applying the lower bound of CDNO, it suffices to observe that to fake a bit, the encryptor must again always *decrease* the number of claimed pseudorandom elements by 1. As in CDNO, one may pretend that pseudorandom is random, but it is infeasible to pretend in the opposite direction. Hence, for any bit b, the adversary can compute the expected number of elements in $[n]$ which should be pseudorandom, and decide that the encryptor is cheating if the claimed number of pseudorandom elements is below the expected value. The success probability of the adversary is $\Omega(\frac{1}{n})$ by exactly the same argument as in CDNO. We recap the argument below.

Definition 7.1 (Separable Scheme)**.** [CDNO97] A $\frac{1}{\delta}$-deniable public key encryption scheme is n-separable if there exists an efficient deterministic algorithm Cnt which given an input rand (interpreted as the claimed random input of the sender), outputs a number $\mathsf{Cnt}(\mathsf{rand}) \in \{1, \ldots, n\}$ (interpreted as the number of pseudorandom elements used by the encryption algorithm to generate the ciphertext). Additionally:

1. For a value rand, let rand^b be the random variable denoting the output of the faking algorithm $\mathsf{Fake}(\mathsf{dpk}, \bar{b}, \mathsf{rand}, b)$ and $\mathbb{E}(\mathsf{Cnt}(\mathsf{rand}^b))$ denote the expected value of $\mathsf{Cnt}(\mathsf{rand}^b)$. Then for any value rand such that $\mathsf{Cnt}(\mathsf{rand}) \geq 1$, either $\mathbb{E}(\mathsf{Cnt}(\mathsf{rand}^0)) \leq \mathsf{Cnt}(\mathsf{rand}) - 1$ or $\mathbb{E}(\mathsf{Cnt}(\mathsf{rand}^1)) \leq \mathsf{Cnt}(\mathsf{rand}) - 1$.

2. If $\mathsf{Cnt}(\mathsf{rand}) = 0$, then the faking algorithm aborts and outputs "cheating impossible".

It is easy to see that our schemes in Sections 4.2 and 6 are n-separable. The value $\mathsf{Cnt}(\mathsf{rand})$, i.e. the number of pseudorandom elements in rand can be easily computed as the number of 1's in $x_1, \ldots, x_n$. Moreover, the faking algorithm always decreases the number of pseudorandom elements used during encryption hence condition 1 is satisfied. If the number of pseudorandom elements used is 0, the fake algorithm outputs "cheating impossible" and aborts (please see step 3 of Fake in Section 4.2), hence condition 2 is satisfied.

CDNO prove the following theorem:

Theorem 7.2. *[CDNO97, Claim 8] For any n-separable scheme with $\frac{1}{\delta}$ deniability, it holds that $2n \geq \delta$.*

The proof follows by demonstrating an adversary $\mathcal{A}$ who can distinguish between the real and fake distributions of randomness when the bit $\bar{b}$ is encrypted (or claimed encrypted). We have by the definition of separable that $\mathbb{E}(\mathsf{Cnt}(\mathsf{rand})) - \mathbb{E}(\mathsf{Cnt}(\mathsf{rand}^0)) \geq \frac{1}{2}$ or $\mathbb{E}(\mathsf{Cnt}(\mathsf{rand})) - \mathbb{E}(\mathsf{Cnt}(\mathsf{rand}^1)) \geq \frac{1}{2}$. Let D denote the distribution of $\mathsf{Cnt}(\mathsf{rand})$ when rand is chosen randomly, and D_b denote the distribution of $\mathsf{Cnt}(\mathsf{rand}^b)$. Then we have that

$$\mathsf{SD}(D, D_0) > \frac{1}{2n} \quad \text{or} \quad \mathsf{SD}(D, D_1) > \frac{1}{2n}$$

The distinguisher $\mathcal{A}$ is now straightforward – it leverages the above statistical distance between the real and fake distributions to distinguish successfully with probability at least $\frac{1}{2n}$. We refer the reader to [CDNO97] for more details.

8 Weakening the Condition on Special FHE

In this section, we describe how to adapt our constructions to rely on the weak version of property 5 of special FHE. To aid understanding, we recap the strong and weak version of the property below:

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 $\mathcal{S} = \{0\}$.

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 $0 \in \mathcal{S}$.

Modifying Our Constructions. Let us consider the case of binary message spaces. Let us say that decryption of a random element R from the ciphertext space yields 0 with only non-negligible probability. Thus, $\mathsf{Btsp}(R)$ is an encryption of 0 also with non-negligible probability. Intuitively, we may amplify this probability by sampling many random elements, bootstrapping them, and setting R as the homomorphic AND function on these. In more detail, denote by $1/p = 1/\operatorname{poly}(\lambda)$

the probability in which $\mathsf{Btsp}(R)$ is a valid encryption of 0, when R is a uniform element form the ciphertext space, i.e. $R \leftarrow \mathcal{R}^{\ell_c}$. If we sample $k = \lambda \cdot p^2$ random elements then the probability for homomorphic AND of k such elements to be a valid encryption of 1 is $(1 - 1/p)^k \leq e^{-k/p} = e^{-\lambda \cdot p}$, which is negligible[13]. Thus, the homomorphic AND ciphertext will be an encryption of 0 with overwhelming probability as desired.

For concreteness, we describe the encryption algorithm in Section 4.2. Assume that for a uniformly random $R \leftarrow \mathcal{R}^{\ell_c}$, $\Pr[\mathsf{Dec}(\mathsf{sk}, R) = 0] = 1/p$. Then, the new encryption algorithm is described as follows:

$\underline{\mathsf{DFhe.Enc}(\mathsf{dpk}, m):}$ 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 uniformly $x_1, \ldots, x_n \in \{0, 1\}$ such that $\sum_{i=1}^{n} x_i = m \pmod 2$.
3. For $i \in [n]$, if $x_i = 0$, select R_i as in Figure 8.1. Observe that in Section 4.2, we select $R_i \leftarrow \mathcal{R}^{\ell_c}$ when $x_i = 0$.
4. For $i \in [n]$ such that $x_i = 1$, select R_i as in Figure 8.2. Observe that in Section 4.2, we select $R_i = \mathsf{Fhe.Enc}(\mathsf{pk}, 1; r_i)$ when $x_i = 1$, where $r_i \leftarrow \{0, 1\}^{\ell}$.
5. Output $\mathsf{dct} = \oplus_2(\mathsf{Btsp}(R_1), \ldots, \mathsf{Btsp}(R_n))$

Sampling R_i for $x_i = 0$

Sample R_i as follows:

1. Sample $A_1, \ldots, A_k$ randomly in $\mathcal{R}^{\ell_c}$.
2. For $j \in [k]$, let $T_j = \mathsf{Btsp}(A_j)$.
3. Set $R_i = \mathsf{Fhe.Eval}(\mathsf{pk}, \mathsf{AND}, T_1, \ldots, T_k)$.

Figure 8.1: Algorithm to sample R_i when $x_i = 0$.

Sampling R_i for $x_i = 1$

Sample R_i as follows:

1. Sample $r_j \leftarrow \{0, 1\}^{\ell}$ for $j \in [k]$.
2. Compute $A_1, \ldots, A_k$ as FHE encryptions of 1, that is $A_j \leftarrow \mathsf{Enc}(\mathsf{pk}, 1; r_j)$ for $j \in [k]$.
3. Set $T_j = \mathsf{Btsp}(A_j)$ for $j \in [k]$.
4. Set $R_i = \mathsf{Fhe.Eval}(\mathsf{pk}, \mathsf{AND}, T_1, \ldots, T_k)$.

Figure 8.2: Algorithm to sample R_i when $x_i = 1$.

[13]Recall, for any real numbers x, r with $r > 0$, one has $(1 + x)^r \leq e^{rx}$.

Correctness. Observe that when $x_i = 0$, R_i, and hence $\mathsf{Btsp}(R_i)$ will be an encryption of 0 with overwhelming $(1 - \mathsf{negl})$ probability. Similarly, when $x_i = 1$, R_i and hence $\mathsf{Btsp}(R_i)$ will always be an encryption of 1. The remainder of the correctness follows exactly as in Section 4.2.

Faking. Faking is performed exactly as in Section 4.2, except that each R_i is now replaced by the corresponding vector $A_{i,1}, \ldots, A_{i,k}$. If R_i is explained as random (resp. pseudorandom) in the fake algorithm of Section 4.2, then the corresponding tuple is explained as random (resp. pseudorandom) in the current construction.

Acknowledgment

We are grateful to Daniele Micciancio for very insightful discussions about bootstrapping, and helpful comments that helped us improve the quality of this writeup.

References

[AFL16] Daniel Apon, Xiong Fan, and Feng-Hao Liu. Deniable attribute based encryption for branching programs from lwe. In *Theory of Cryptography Conference*, pages 299–329. Springer, 2016.

[AJ15] Prabhanjan Ananth and Abhishek Jain. Indistinguishability obfuscation from compact functional encryption. In *Annual Cryptology Conference*, pages 308–326. Springer, 2015.

[BDGM19] Zvika Brakerski, Nico Döttling, Sanjam Garg, and Giulio Malavolta. Leveraging linear decryption: Rate-1 fully-homomorphic encryption and time-lock puzzles. In *Theory of Cryptography Conference*, pages 407–437. Springer, 2019.

[BGV14] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. *ACM Transactions on Computation Theory (TOCT)*, 6(3):1–36, 2014.

[BNNO11] Rikke Bendlin, Jesper Buus Nielsen, Peter Sebastian Nordholt, and Claudio Orlandi. Lower and upper bounds for deniable public-key encryption. In *Asiacrypt*. Springer, 2011.

[Bra12] Zvika Brakerski. Fully homomorphic encryption without modulus switching from classical gapsvp. In *Annual Cryptology Conference*, pages 868–886. Springer, 2012.

[BV14] Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) lwe. *SIAM Journal on Computing*, pages 831–871, 2014.

[BV18] Nir Bitansky and Vinod Vaikuntanathan. Indistinguishability obfuscation from functional encryption. *Journal of the ACM (JACM)*, 65(6):1–37, 2018.

[CDNO97] Rein Canetti, Cynthia Dwork, Moni Naor, and Rafail Ostrovsky. Deniable encryption. In *Annual International Cryptology Conference*, pages 90–104. Springer, 1997.

[CGGI16] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. A homomorphic lwe based e-voting scheme. In *Post-Quantum Cryptography*, pages 245–265. Springer, 2016.

[CPP20] Ran Canetti, Sunoo Park, and Oxana Poburinnaya. Fully deniable interactive encryption. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology - CRYPTO 2020 - 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17-21, 2020, Proceedings, Part I*, volume 12170 of *Lecture Notes in Computer Science*, pages 807–835. Springer, 2020.

[DCIO16] Angelo De Caro, Vincenzo Iovino, and Adam O'Neill. Deniable functional encryption. In *Public-Key Cryptography–PKC 2016*, pages 196–222. Springer, 2016.

[DS14] Dana Dachman-Soled. On minimal assumptions for sender-deniable public key encryption. In *International Workshop on Public Key Cryptography*, pages 574–591. Springer, 2014.

[Gen09] Craig Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, 2009. crypto.stanford.edu/craig.

[GGH+16] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. *SIAM Journal on Computing*, 45(3):882–929, 2016.

[GH19] Craig Gentry and Shai Halevi. Compressible fhe with applications to pir. In *Theory of Cryptography Conference*, pages 438–464. Springer, 2019.

[GPSZ17] Sanjam Garg, Omkant Pandey, Akshayaram Srinivasan, and Mark Zhandry. Breaking the sub-exponential barrier in obfustopia. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 156–181. Springer, 2017.

[GSW13] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In *Annual Cryptology Conference*, pages 75–92. Springer, 2013.

[Men09] Bo 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*, 4(5):370–377, 2009.

[OPW11] Adam O'Neill, Chris Peikert, and Brent Waters. Bi-deniable public-key encryption. In *Annual Cryptology Conference*, pages 525–542. Springer, 2011.

[SV10] Nigel P Smart and Frederik Vercauteren. Fully homomorphic encryption with relatively small key and ciphertext sizes. In *International Workshop on Public Key Cryptography*, pages 420–443. Springer, 2010.

[SW14] Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 475–484, 2014.

A Another Compact FHE for bits

We provide another construction for compact deniable FHE, as in Definition 2.10. 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, $\ell'_c = \lceil \ell_c \log_2(|\mathcal{R}|) \rceil$).

Our alternate compact public-key deniable 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 as follows:

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

$\mathsf{DFhe.Enc}(\mathsf{dpk}, m)$: Upon input the public-key dpk, the message bit m, do the following:

1. Parse $\mathsf{dpk} := (\mathsf{pk}, \mathsf{ct_{sk}})$
2. Select $\mathsf{rand} = (k, r_1, \ldots, r_k, R_{k+1}, \ldots R_n) \in \{0,1\}^{\log_2(n)} \times \{0,1\}^{k\ell} \times \mathcal{R}^{(n-k)\ell_c}$ as follows:
   (a) If $m = 0$, select $k \leftarrow \{0, 2, \ldots, n-1\}$, else if $m = 1$ select $k \leftarrow \{1, 3, \ldots, n\}$ where n is an odd integer.
   (b) Select $r_i \leftarrow \{0,1\}^\ell$, for $i \in [k]$.
   (c) Select and $R_i \leftarrow \mathcal{R}^{\ell_c}$ for $i \in [n] \setminus [k]$.
3. Let $R_i = \mathsf{Fhe.Enc}(\mathsf{pk}, 1; r_i)$ for $i \in [k]$.
4. Output $\mathsf{dct} = \oplus_2(\mathsf{Btsp}(R_1), \ldots, \mathsf{Btsp}(R_n))$

$\mathsf{DFhe.Eval}(\mathsf{dpk}, \mathcal{C}, \mathsf{dct}_1, \ldots, \mathsf{dct}_k)$: Upon input the public key $\mathsf{dpk} = (\mathsf{pk}, \mathsf{ct_{sk}})$, 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 $\mathsf{Fhe.Dec}(\mathsf{sk}, \mathsf{ct})$.

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

1. If $m = m^*$, output $\mathsf{rand}^* = \mathsf{rand}$.
2. Parse $\mathsf{dpk} := (\mathsf{pk}, \mathsf{ct_{sk}})$ and $\mathsf{rand} = (k, r_1, \ldots, r_k, R_{k+1}, \ldots, R_n)$, where $|k| = \log_2(n)$, $|r_i| = \ell$ for $i \in [k]$, $|R_i| = \ell'_c$ for $i \in [n] \setminus [k]$.
3. Set $k^* = k - 1$. If $k = 0$, output "cheating impossible" and abort.
4. Set $r^*_i = r_i$ for $i \in [k^*]$.
5. Set $R^*_{k^*+1} = \mathsf{Fhe.Enc}(\mathsf{pk}, 1; r_k)$.
6. Set $R^*_i = R_i$ for $i \in [n] \setminus [k]$.

7. Let $\mathsf{rand}^* = (k^*, r_1^*, \ldots, r_{k^*}^*, R_{k^*+1}^*, \ldots, R_n^*)$, that is

$$\mathsf{rand}^* = (k-1, r_1, \ldots, r_{k-1}, \mathsf{Fhe.Enc}(\mathsf{pk}, 1; r_k), R_{k+1}, \ldots, R_n).$$

8. Output rand^*.

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 $i \in [k]$, we have by correctness of the underlying Fhe that $R_1, \ldots, R_k$, and hence $\mathsf{Btsp}(R_1), \ldots, \mathsf{Btsp}(R_k)$ are valid encryptions of 1.
2. By properties 3 and 5 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 $\mathsf{Btsp}(R_{k+1}), \ldots, \mathsf{Btsp}(R_n)$ are valid encryptions of 0 with overwhelming probability.

Hence, when k is odd (respectively even), the (FHE evaluation of) addition mod 2 of $\mathsf{Btsp}(R_i)$ for $i \in [n]$ yields an encryption of 1 (respectively 0). Hence, the scheme encodes the message bit correctly.

Polynomial Deniability. When the original message m and the fake message m^* are the same, the faked randomness rand^* is equal to the original randomness rand. Thus in this case, $m = m^*$, the distribution are identical $(\mathsf{dpk}, m, \mathsf{rand}) = (\mathsf{dpk}, m^*, \mathsf{rand}^*)$. When the original message m and the fake message m^* are not the same, we distinguish two cases:

1. When the original message $m = 0$, we have in the real randomness rand, $k \leftarrow \{0, 2, \ldots, n-1\}$, where n is an odd integer. In the faking algorithm, we have in the faked randomness rand^*, $k^* = k - 1$, that is $k^* \leftarrow \{-1, 1, \ldots, n-2\}$.
2. When the original message $m = 1$, we have in the real randomness rand, $k \leftarrow \{1, 3 \ldots, n\}$, where n is an odd integer. In the faking algorithm, we have in the faked randomness rand^*, $k^* = k - 1$, that is $k^* \leftarrow \{0, 2, \ldots, n-1\}$.

Observe that when the original message is $m = 1$, we have that k^* in the faking randomness rand^* is sampled from the exact same distribution as in the real randomness rand when encrypting the message $m = 0$. Moreover, by property 2 which asserts that ciphertexts are pseudorandom, we can explain R_k^* as uniform element in $\mathcal{R}^{\ell_c}$. Hence, the output of the faking algorithm in this case $(m = 0, m^* = 1)$ will be indistinguishable from real randomness.

When the original message is $m = 0$, the statistical distance between the distribution of sampling k^* in the faking algorithm (namely, sampling $k \leftarrow \{2, 4, \ldots, n-1\}$ and setting $k^* = k - 1$) and the distribution of sampling k when encrypting the message $m = 1$ (namely, sampling $k \leftarrow \{1, 3, \ldots, n\}$) is $\frac{2}{n+1}$. To see this, let P and Q be two distribution over a finite set $\mathcal{U} = \{-1, 1, 3, \ldots, n\}$, where $P(x) = \frac{2}{n+1} = \frac{1}{|\mathcal{U}| \setminus \{-1\}}$, $P(-1) = 0$, i.e. P is the uniform distribution over the set $\{1, 3, \ldots, n\}$, and $Q(x) = \frac{2}{n+1}$ for all $x \in \mathcal{U} \setminus \{n\}$, and $Q(n) = 0$ (Q is the uniform distribution over the set $\{-1, 1, \ldots, n-2\}$). The statistical distance between these distributions is $\mathsf{SD}(P, Q) = \frac{1}{2}\left(\frac{2}{n+1} + \frac{2}{n+1} + 0\right) = \frac{2}{n+1}$. Note that when $k = 0$ ($k^* = -1$) "cheating is impossible", which happened with $\frac{2}{n+1}$ probability. The probability we select $k = 0$ from the set of $\{0, 2, \ldots, n-1\}$, is $\frac{1}{|\{0,2,\ldots,n-1\}|} = \frac{2}{n+1}$.