Fully Homomorphic Encryption from the Finite Field Isomorphism Problem

Yarkın Doröz[1] ⋆, Jeffrey Hoffstein[2] ⋆⋆, Jill Pipher[2], Joseph H. Silverman[2] ⋆⋆,
Berk Sunar[1] ⋆, William Whyte[3], and Zhenfei Zhang[3]

[1] Worcester Polytechnic Institute, Worcester MA, USA, {ydoroz,sunar}@wpi.edu
[2] Brown University, Providence RI, USA, {jhoff,jpipher,jhs}@math.brown.edu
[3] OnBoard Security, Wilmington MA, USA, {wwhyte,zzhang}@onboardsecurity.com

Abstract. If q is a prime and n is a positive integer then any two finite fields of order q^n are isomorphic. Elements of these fields can be thought of as polynomials with coefficients chosen modulo q, and a notion of length can be associated to these polynomials. A non-trivial isomorphism between the fields, in general, does not preserve this length, and a short element in one field will usually have an image in the other field with coefficients appearing to be randomly and uniformly distributed modulo q. This key feature allows us to create a new family of cryptographic constructions based on the difficulty of recovering a secret isomorphism between two finite fields. In this paper we describe a fully homomorphic encryption scheme based on this new hard problem.

Keywords: Finite field isomorphism, fully homomorphic encryption, lattice-based cryptography.

1 Introduction

Let q be a prime, let $\mathbb{F}_q$ be the finite field with q elements, and let $\boldsymbol{f}(x) \in \mathbb{F}_q[x]$ and $\boldsymbol{F}(y) \in \mathbb{F}_q[y]$ be irreducible monic polynomials of degree n. Then

$$\mathbb{X} := \mathbb{F}_q[x]/(\boldsymbol{f}(x)) \quad \text{and} \quad \mathbb{Y} := \mathbb{F}_q[y]/(\boldsymbol{F}(y)) \tag{1}$$

are isomorphic fields with q^n elements. Given knowledge of $\boldsymbol{f}(x)$ and $\boldsymbol{F}(y)$, it is easy to write down an explicit isomorphism $\mathbb{X} \to \mathbb{Y}$ and its inverse. We normalize mod q polynomials by choosing their coefficients between $-\frac{1}{2}q$ and $\frac{1}{2}q$, and then we define the size of a polynomial to be the magnitude of its largest coefficient. It is then an observation that, except in trivial cases, the isomorphism $\mathbb{X} \to \mathbb{Y}$ does not respect the Archimedean property of size. Indeed, when $\boldsymbol{f}$ and $\boldsymbol{F}$ are distinct monic irreducible polynomials, we have observed that polynomials

⋆ Funding for this research was in part provided by the US National Science Foundation CNS Award #1561536.

⋆⋆ Fundings for this research were in part provided by the US National Science Foundation CNS Award #1349908 and #1561709.

within a sphere of small radius (with respect to the L^∞ or L^2 norm) in $\mathbb{X}$ appear to be essentially uniformly distributed in $\mathbb{Y}$. We record this observation formally, and construct arguments for its veracity in Section 2.2.1.

Observation 1 *Let $\mathcal{M}_{n,q}$ be the set of all degree n monic irreducible polynomials mod q and fix $1 \leq \beta < q/2$. Sample $\boldsymbol{f} \in \mathbb{F}_q[x]$ and $\boldsymbol{F} \in \mathbb{F}_q[y]$ uniformly from $\mathcal{M}_{n,q}$, and construct $\mathbb{X}$, $\mathbb{Y}$ and the associated isomorphism $\phi : \mathbb{X} \to \mathbb{Y}$ as in (1). Let χ_β be a distribution that produces samples with bounded length less than β. Then the image in $\mathbb{Y}$ of a collection of polynomials in $\mathbb{X}$ sampled from χ_β is computationally hard to distinguish from a collection of polynomials sampled uniformly in $\mathbb{Y}$. By a proper choice of parameters, the ability to distinguish such a collection can be made arbitrarily difficult.*

Remark 1. We will refer to elements of $\mathbb{X}$ or $\mathbb{Y}$ as *short* if they have infinity norm less than β, where generally β will be less than $q/4$.

We will find it essential to choose $\boldsymbol{f}$ from a subset of $\mathcal{M}_{n,q}$ consisting of monic irreducible polynomials of degree n whose coefficients have absolute value less than or equal to 1. Observation 1 appears to remain true, even when restricted to this subset of $\mathcal{M}_{n,q}$, and the security of our proposed homomorphic scheme will rest on:

Observation 2 *Observation 1 remains true if $\boldsymbol{f} \in \mathbb{F}_q[x]$ is chosen from the subset of polynomials in $\mathcal{M}_{n,q}$ whose coefficients have a max absolute value 1.*

In this paper we base two distinct, but related, problems on Observation 2.

Definition 1 (FFI). ***Finite Field Isomorphism Problems*** *Let k be a positive integer. Let $\mathbb{X}, \mathbb{Y}, \phi, \chi_\beta$ be as above. Let $\boldsymbol{a}_1(x), \ldots, \boldsymbol{a}_k(x), \boldsymbol{b}_1(x)$ be samples from χ_β, and $\boldsymbol{A}_i = \phi(\boldsymbol{a}_i)$ and $\boldsymbol{B}_1 = \phi(\boldsymbol{b}_1)$ be the corresponding images. Also sample $\boldsymbol{B}_2(y)$ uniformly from $\mathbb{Y}$.*
Computational FFI problem*: Given $\mathbb{Y}, \boldsymbol{A}_1(y), \ldots, \boldsymbol{A}_k(y)$, recover $\boldsymbol{f}(x)$ and/or $\boldsymbol{a}_1(x), \ldots, \boldsymbol{a}_k(x)$.*
Decisional FFI problem*: Given $\mathbb{Y}, \boldsymbol{A}_1(y), \ldots, \boldsymbol{A}_k(y)$, $\boldsymbol{B}_1$ and $\boldsymbol{B}_2$, with one of $\boldsymbol{B}_1, \boldsymbol{B}_2$ an image of a sample from χ_β, identify the image with a probability greater than $1/2$.*

Clearly, the decisional FFI problem can be solved if the computational FFI problem can be solved, and if Observation 1 is correct, then the decisional FFI problem can be made arbitrarily hard. We will demonstrate that if a certain lattice reduction problem of dimension roughly $2n$ can be solved, then the decisional FFI problem can be solved, and this lattice reduction problem can be made arbitrarily hard. We do not, however, have a reduction showing that ability to solve the decisional problem implies the ability to solve a lattice reduction problem. In other words, the strongest attacks we have found on the decisional problem are via lattice reduction arguments, but we cannot rule out the possibility of other, potentially stronger, attacks.

Our plan is to build a somewhat homomorphic encryption scheme based on the decisional FFI problem. This will have double exponential noise growth, but will also have the advantage of being able to handle a reasonable number of multiplications (and additions) of moderate sized integers. We will then analyze the noise performance, and introduce a bit-decomposition-based noise management scheme that allows us to reduce the noise growth to single exponential. This will yield a bootstrappable, thus a fully homomorphic encryption scheme.

We will encode numbers, i.e messages, as short elements in $\mathbb{X}$, with noise added for semantic security, and view their corresponding images in $\mathbb{Y}$ as ciphertexts. This will create a symmetric encryption algorithm, which will be somewhat homomorphic in the following sense: Polynomials in elements of $\mathbb{X}$ can be evaluated, and lifted to polynomials over $\mathbb{Z}[x]/(\boldsymbol{f}(x))$ as long as their coefficients do not exceed $q/2$ in absolute value. Knowledge of these output polynomials will allow the user with knowledge of $\boldsymbol{f}(x)$ to recover the value of the polynomial over $\mathbb{Z}$, and the output of the computation. The corresponding ciphertext polynomials in $\mathbb{Y}$ can be evaluated by anyone with knowledge of the public key $\boldsymbol{F}(y)$, and substantial reduction modulo q will occur. Decryption will occur by mapping isomorphically back to $\mathbb{X}$, and the correct result will be output as long as the coefficients do not exceed $q/2$ in absolute value.

This is where an important point arises. In 1996, (eventually published in [25]), NTRU introduced the idea that if two short polynomials in $\mathbb{Z}[x]$ are multiplied, and the result is reduced modulo $x^n - 1$, then the reduced product is also (moderately) short. This observation has been used, in the years since then, in a variety of cryptographic constructions. In this paper we make use of a variation on this observation: This property remains true for a considerably larger class of polynomials than $x^n \pm 1$. In particular, if $\boldsymbol{f}(x)$ is chosen to be monic, of degree n, and have coefficients from the set $\{-1, 0, 1\}$, then a short polynomial times a short polynomial remains moderately short when reduced modulo $\boldsymbol{f}(x)$. If parameters are chosen properly, the search space for $\boldsymbol{f}(x)$ can be made arbitrarily large, making it impractical to locate $\boldsymbol{f}(x)$ by a brute force search.

The symmetric system sketched above can be converted into a public key encryption scheme using the standard technique of publishing a list of encryptions of 0 and adding short linear combinations of these encryptions as noise. Its semantic security can be seen to be based on the decisional FFI problem, not on the presumably harder computational FFI problem. It is not immediately obvious that this is the case, as all ciphertexts of messages will be images of short vectors in $\mathbb{X}$, but in the simple instantiation we will present here, it can be shown that this is true. (See Theorem 1 in Section 3.2.4.)

1.1 Subfield Attack

Despite major advances over the past few years the biggest challenge preventing the deployment of FHE schemes in real life applications is efficiency. To address the efficiency bottleneck, many optimizations were proposed including some that take advantage of specialization of the underlying field/ring structure. Such specializations enable efficient batched parallel evaluations, make it possible to choose parameters that support highly efficient number theoretical transforms, and in some cases even reduce the size of evaluation keys.

However, such customizations may potentially introduce weaknesses in the security assumptions of the schemes. A recent family of attacks proposed by Albrecht, Bai and Ducas [29], by Cheon, Jeong and Lee [8], and by Kirchner and Fouque [27] exploit the special structure, namely subfields, in ring based FHE schemes. Furthermore, the attack in [27] also works when the underly ring does not admit subfields. Moving to a subfield with a Norm mapping as in [29], or a Trace mapping as in [8] or the Gentry-Szydlo mapping [22] as in [27] will reduce the dimension of the lattice. Then, via a projection, also named zero-forcing in the original May-Silverman description [30], the Kirchner-Fouque method is able to create a lattice with an even smaller dimension, at the cost of reducing the number of unique shortest vectors in the lattice.

This set of attacks demonstrated that several NTRU based FHEs with medium size parameters are no longer secure. Specifically, if the NTRU scheme is constructed with the DSPR security assumption, which is the case in some of the NTRU based FHE schemes [3, 28], the assumed security level of the scheme can be significantly reduced. While the authors suggest more caution on parameter selection by avoiding specialized fields in this particular case, there could be further attacks that exploit specialized parameters. It has become quite clear that we need more generic constructions that avoid specialized structures as much as possible. Furthermore, we need diversity in the FHE constructions, i.e. FHEs that remain secure even if other conjectured hard problems, e.g. DSPR or Approximate GCD, are shown to be weaker than expected.

These are among the goals of the FHE scheme proposed in this paper: The proposed construction is based on the DFFI problem; a new problem we propose and analyze here for the first time. The proposed construction avoids specializations. The FHE scheme is based on a fixed prime q and a class of short generic private keys $\boldsymbol{f}(x)$ with the property that $\boldsymbol{f}(x)$ is monic, irreducible mod q, and the Galois group of the associated finite field $\mathbb{Z}_q[x]/(\boldsymbol{f}(x))$ is C_n.

With such choice of parameters it is safe to claim that attacks in [29] and [8] no longer apply due to the lack of subfields. In addition, as one shall see in Section 2.4, the unique shortest vectors in this class of lattices are not sparse vectors with many 0s, and they are not cyclic rotations of each other. Therefore, the projection method will not work either. Thus we also assert that attack in [27] is not applicable either.

Remark 2. The security of the finite field homomorphic encryption scheme presented here is based on the decisional problem (DFFI). It may be possible to construct a homomorphic encryption scheme that solely depends on the computational problem, (CFFI), but in the interest of simplicity we will not pursue this here. It is certainly possible to construct a signature scheme, based on the CFFI, and this will appear elsewhere.

1.2 A sketch of the main ideas

Messages, which are integers, will be mapped to elements of $\mathbb{X}$ by some method. These elements will be sparse, low weight polynomials, $m(x)$, of degree at most

$n-1$. For each message encryption, a sparse low weight, e.g. trinary, polynomial $r(x)$ of degree at most $n-1$ will be chosen at random. A polynomial $p(x)$ will be fixed as a public parameter. This polynomial will have coefficients with small infinity norm. Two useful possibilities for $p(x)$ are $p(x) = 2$, and $p(x) = x - 2$. We will illustrate below with the example $p(x) = x - 2$. To encode an integer $1 \leq m < 2^n$, write m in base two as $m = b_0 + 2b_1 + \cdots + 2^{n-1}b_{n-1}$, and represent m by $m(x) = b_0 + b_1x + \cdots + b_{n-1}x^{n-1}$. Thus $m(2) = m$. An encoding of $m(x)$ in $\mathbb{X}$ will be done as follows:

- Choose $r(x)$ at random from a given distribution of sparse, binary or trinary, polynomials of degree less than n.
- The encoded message is $e_m(x) := m(x) + p(x)r(x) \mod f(x)$. As the coefficients of $p(x)$ and $r(x)$ are very small, and $f(x)$ is chosen as described above, the reduction of $m(x)+p(x)r(x) \bmod f(x)$ will have coefficients that remain small relative to q. In other words, the lift of $e_m(x)$ from $\mathbb{X}$ to an element of $\mathbb{Z}[x]/(f(x))$ with coefficients in the interval $(-q/2, q/2]$ will have no reduction modulo q occurring.

Encryption of $e_m(x)$ is done by mapping $e_m(x)$ to its isomorphic image $E_m(y)$ in $\mathbb{Y}$, using the isomorphism $\mathbb{X} \to \mathbb{Y}$ that is known to the encryptor. The somewhat homomorphic property for multiplication is seen as follows: Given $e_{m_1}(x) = m_1(x) + p(x)r_1(x)$ and $e_{m_2}(x) = m_2(x) + p(x)r_2(x)$, the product is given by

$$\begin{aligned}
&e_{m_1}(x)e_{m_2}(x) \\
&\quad = m_1(x)m_2(x) + p(x)r_1(x)m_2(x) + p(x)r_2(x)m_1(x) + p(x)^2r_1(x)r_2(x) \\
&= m_1(x)m_2(x)+p(x)[r_1(x)m_2(x)+r_2(x)m_1(x)+p(x)r_1(x)r_2(x)] \mod (f(x), q).
\end{aligned} \tag{2}$$

The key observation is that since the coefficients of $e_{m_1}(x)$ and $e_{m_2}(x)$ are small compared to q, the product, even after reduction mod $f(x)$, will still have coefficients that are small compared to q. As a result, if the reduced product $e_{m_1}(x)e_{m_2}(x)$ is lifted from $\mathbb{X}$ to $\mathbb{Z}[x]/(f(x))$ with coefficients chosen from the interval $(-q/2, q/2]$, then the coefficients will be the same as if the computation had taken place over $\mathbb{Z}[x]/(f(x))$.

A similar comment applies to $e_{m_1}(x)+e_{m_2}(x)$. Because the mapping between $\mathbb{X}$ and $\mathbb{Y}$ is a field isomorphism, it follows that

$$E_{m_1}(y)E_{m_2}(y) = E_{m_1m_2}(y) \text{ and } E_{m_1}(y) + E_{m_2}(y) = E_{m_1+m_2}(y).$$

This means that a polynomial function of elements of $\mathbb{X}$ can be computed on the isomorphic images of these elements in $\mathbb{Y}$ and the output mapped back to $\mathbb{X}$, and, as long as the coefficients in the corresponding $\mathbb{X}$ computation remain in the interval $(-q/2, q/2]$, the image of the output in $\mathbb{X}$ can be lifted to $\mathbb{Z}[x]/(f(x))$ without any loss of information.

The key question then is how to recover $m(x)$ from a polynomial of the form $m(x) + p(x)r(x)$ in $\mathbb{X}$. After a computation is performed, as seen in (2) above, the output in $\mathbb{X}$ will still have this form, although the coefficients of $m(x)$ and

$r(x)$ may be considerably larger than binary or trinary. As long as they have not passed $q/2$ in absolute value, the lift to $\mathbb{Z}[x]/(f(x))$ will not involve any mod q reduction. The decryption process, then consists of:

- Map the output of the computation in $\mathbb{Y}$ back to $\mathbb{X}$. It will have the form $m'(x) + p(x)r'(x)$, for unknown polynomials $m'(x)$ and $r'(x)$
- This can be further lifted to $\mathbb{Z}[x]$ by viewing of it as $m'(x) + p(x)r('x) + s(x)f(x)$ for some also unknown polynomial $s(x)$
- Compute the resultant of $f(x)$ and $p(x)$. This is the ideal in $\mathbb{Z}[x]$ generated by $p(x)$ and $f(x)$ which, in the case $p(x) = x - 2$, is simply $f(2)$. Also, $m'(x)+p(x)r'(x)+s(x)f(x)$ reduced mod $f(x)$ and $x-2$ is $m(2)$ mod $f(2)$. Thus, as long as m is less than $f(2)$, $m = m(2)$ will be recovered exactly.

The process breaks down when the size of any coefficient of the computation exceeds $q/2$ in absolute value. Note that the collection of all $p(x)r(x)$ in $\mathbb{X}$ is all possible encodings of 0, and their images in $\mathbb{Y}$ are all possible encryptions of 0. As we are in a field, not a ring, the ideal generated by all such $p(x)r(x)$ is, of course, all of $\mathbb{Y}$.

1.3 Related work

The first Fully Homomorphic Encryption (FHE) scheme was constructed by Gentry [17, 19] in 2009, answering a problem that had remained open for over three decades. Gentry's scheme is based on ideal lattices and the security assumptions are based on hard problems in lattices. A key innovation in Gentry's construction is *bootstrapping*, which allows a party to refresh the noise level in a ciphertext without having access to a secret key. Despite its success, bootstrapping has remained the bottleneck in FHE implementations. After Gentry's original scheme, many other constructions based on a variety of hardness assumptions followed that aimed to improve the efficiency of FHE.

One such construction based on the learning-with-errors (LWE) problem was proposed by Brakerski and Vaikuntanathan [6]. The security of the scheme is based on the hardness of short vector problems. The LWE-based construction was later improved by Brakerski, Gentry and Vaikuntanathan (BGV) in [5] using a *modulus switching* technique that slows the noise accumulation drastically. Modulus switching is applied at each multiplicative level, which prevents exponential noise growth. Thereby the noise remains fixed throughout the homomorphic evaluation levels. Later, a new noise management technique was introduced by Brakerski [4], applicable to LWE schemes, that decreases noise growth from quadratic to linear using tensor products. Gentry, Halevi and Smart [20] demonstrated that it is possible to perform deep homomorphic evaluations by providing the first AES evaluation implemented using the BGV scheme embodied in a software library called HElib [23]. The authors optimize the design using the SIMD technique introduced in [31] to batch multiple messages and process parallel AES operations. Another FHE construction based on the assumed hardness of the *Integer Approximate-GCD* problem was proposed by van Dijk et al. [12]. This work was followed by Coron et al. [10], where the public key size was reduced from $\lambda\mathcal{O}(\kappa^{10})$ to $\mathcal{O}(\kappa^7)$ where κ is the security parameter. In [11] the public key size was further reduced from $\mathcal{O}(\kappa^7)$ to $\mathcal{O}(\kappa^5)$ and modulus switching methods were adapted to the integer scheme. Another follow up work by Coron et al. [9] implements a variant of van Dijk et al.'s scheme using the scale invariant property introduced earlier by Brakerski [4].

Another leveled FHE scheme was presented by López-Alt, Tromer, Vaikuntanathan (LTV) in [28]. It is based on a variant of NTRU [25] constructed earlier by Stehlé and Steinfeld [32]. The scheme is a multi-party scheme that is capable of processing homomorphic functions for various users each with their individual keys. The authors use the *relinearization* technique introduced in [6] and also adapt modulus switching to mitigate the noise growth, thus keeping the growth linear in size over the levels. To compute relinearization, the scheme requires evaluation keys, which increases the memory requirement and becomes prohibitive especially in *deep* evaluations. The NTRU variant by Stehlé and Steinfeld [32] was later modified and implemented by Bos et al. in [3]. Their scheme, named YASHE, adopts the tensor product technique in [4] and achieves a scale-invariant scheme with limited noise growth on homomorphic operations. Also, with the use of the tensor product technique, the authors managed to improve the security of the LTV scheme [28] by using much higher levels of noise and thereby removed the Decisional Small Polynomial Ratio (DSPR) assumption. Instead, the scheme relies only on standard lattice reductions as in [32]. However, as the authors also note, the YASHE scheme requires a large evaluation key and a complicated key switching procedure. In [3] the authors introduce a modification (YASHE') to their scheme to eliminate the problems of expensive tensor product calculations and large evaluation keys. However, this modification re-introduces the DSPR assumption. Another modified LTV-FHE implementation, along with AES evaluation, was presented by Doröz et al. in [13]. The security of their scheme depends on the DSPR and R-LWE assumptions as in [28]. Their implementation uses the relinearization and modulus switching methods as in [28] to cope with noise, and it introduced a specialized ring structure to to significantly reduce the evaluation key size. Since both the YASHE' and LTV-FHE schemes rely on the DSPR problem, both are vulnerable to the Subfield Attack [29].

Motivated by the large evaluation key requirements come by complex noise management techniques such as *relinearization, modulus switching, and bootstrapping* employed by earlier FHE schemes Gentry, Sahai and Waters [21] proposed a new scheme based on the approximate eigenvector problem. The system uses matrix additions and multiplications, which makes it asymptotically faster. At first, they constructed the GSW scheme as a *somewhat homomorphic scheme*, since for a depth L circuit with B-bounded parameters, the noise grows with a double exponential B^{2^L}. To convert the scheme into a leveled FHE, they introduced a *Flattening* operation that decomposes the ciphertext entries into bits. The secret key is also kept in a special powers-of-two form. With these modifications, the noise performance is improved significantly. For a depth L circuit with B-bounded secret key entries and 1-bounded (flattened) ciphertexts, the error magnitude is at most $(N+1)^L B$ for $N = \log(q)(n+1)$. However, ciphertexts still require a considerable amount space, roughly $\Theta(n^2 \log(q)^2)$, and as noted by GSW [21], in practice their scheme may not be as efficient as existing leveled schemes. More recently, the Flattening technique was adapted by Doröz and Sunar to NTRU in a new FHE scheme called F-NTRU [14]. Similar to the GSW scheme, F-NTRU does not require evaluation keys or key switching. More significantly, the scheme eliminates the DSPR assumption and relies only on the standard R-LWE assumption which makes it the only NTRU variant FHE scheme immune to the Subfield Attack.

1.4 Paper Organization

In Section 2 we formally introduce the finite field isomorphisms problem, state hardness assumptions, and study lattice and non-lattice techniques to establish the difficulty of the problem against known techniques. We then show how to construct a fully homomorphic public-key encryption scheme in Section 3 by first building a somewhat homomorphic encryption scheme and then by converting it into a bootstrapable scheme via a new bit decomposition based noise management scheme. In Section 4, we conclude our paper.

In the appendices, we discuss how to construct field representations $\mathbb{X}$ and $\mathbb{Y}$ and the necessary isomorphisms $\mathbb{X} \to \mathbb{Y}$ and $\mathbb{Y} \to \mathbb{X}$ (Section A), we give a more detailed noise analysis (Section B), we perform security analysis and give estimates on the parameters (Section C), and we give test results for our observation 2 (Section D).

2 The Finite Field Isomorphism (FFI) Problem

2.1 Preliminaries

We begin by formally introducing some notation that has already been used in the previous section. Additional notation will be introduced at the start of Sections 3. For given degree n monic irreducible polynomials $\boldsymbol{f}(x) \in \mathbb{F}_q[x]$ and $\boldsymbol{F}(y) \in \mathbb{F}_q[y]$, we create two copies of $\mathbb{F}_{q^n}$, which we denote by $\mathbb{X} := \mathbb{F}_q[x]/(\boldsymbol{f}(x))$ and $\mathbb{Y} := \mathbb{F}_q[y]/(\boldsymbol{F}(y))$. In general, polynomials denoted by lower case letters will be polynomials in $\mathbb{X}$, and their isomorphic images in $\mathbb{Y}$ will be denoted with the corresponding capital letters. The vector form of a polynomial is simply the vector consisting of its coefficients. We often identify polynomials and vectors when there is no ambiguity. Consider a polynomial $\boldsymbol{a}(x) = a_0 + a_1 x + \cdots + a_{n-1}x^{n-1} \in \mathbb{X}$. We will informally say that $\boldsymbol{a}(x)$ is *short* if for all i, the congruence class $a_i \bmod q$ reduced into the interval $(-q/2, q/2]$ is small relative to q. An important class of such polynomials are those satisfying $a_i \in \{-1, 0, 1\}$; these are called *trinary polynomials*. We denote by $\|\boldsymbol{a}\| = \|\boldsymbol{a}\|_\infty := \max |a_i|$ and $\|\boldsymbol{a}\|_2 := (a_0^2 + \cdots + a_{n-1}^2)^{1/2}$ the L^∞ and L^2 norms of $\boldsymbol{a}$, respectively, where it is understood that the coefficents of $\boldsymbol{a}$ are always normalized to lie in the interval $(-q/2, q/2]$. Denote by $\mathcal{M}_{n,q}$ the set of all degree n monic irreducible polynomials mod q. When there is no ambiguity, we will suppress the subscripts.

2.2 Discussions and proofs

2.2.1 Arguments for the truth of Observation 1

Lemma 1. *For large n, for any fixed $\boldsymbol{f}(x) \in \mathbb{F}_q[x]$, and any given degree $n-1$ polynomial $\phi(y) \in \mathbb{F}_q[y]$, there will exist, with probability approaching 1, a unique monic irreducible $\boldsymbol{F}(y) \in \mathbb{F}_q[y]$ such that the map $x \to \phi(y)$ induces an isomorphism between $\mathbb{F}_q[x]/(\boldsymbol{f}(x))$ and $\mathbb{F}_q[y]/(\boldsymbol{F}(y))$.*

Proof. As $\mathbb{F}_{q^n}/\mathbb{F}_q$ is Galois, any irreducible polynomial with one root must split completely, implying that $\boldsymbol{f}(x)$ has n distinct roots in $\mathbb{F}_q[y]/(\boldsymbol{F}(y))$, and similarly, that no two monic irreducible polynomials of degree n in $\mathbb{F}_q[x]$ can share a root. Fix a degree n monic irreducible polynomial $\boldsymbol{f}(x) \in \mathbb{F}_q[x]$. By the prime number theorem for function fields, for fixed q and large n, $|\mathcal{M}_{n,q}|$, i.e., the number of distinct irreducible monic polynomials over $\mathbb{F}_q[x]$, is asymptotic to q^n/n; see [26, Chapter 7, Section 2, Corollary 2]. It follows that for any polynomial $\boldsymbol{f} \in \mathcal{M}_{n,q}$ there are asymptotically q^n/n distinct isomorphic images of $\mathbb{F}_q[x]/(\boldsymbol{f}(x))$ and hence also q^n/n potential $\boldsymbol{F}$. Choose at random a degree $n-1$ polynomial $\phi(y) \in \mathbb{F}_q[y]$. There are exactly $(q-1)q^{n-1}$ such polynomials. There are also, asymptotically, a total of $n \times q^n/n = q^n$ isomorphisms between $\mathbb{F}_q[x]/(\boldsymbol{f}(x))$ and all possible $\mathbb{F}_q[y]/(\boldsymbol{F}(y))$, where $\boldsymbol{F}(y)$ varies over all distinct monic irreducible polynomials. These are given by sending x to each of the n distinct roots of each $\boldsymbol{F}(y)$. With probability approaching 1 (for large q), these sets have the same order, and as one is contained in the other, they are asymptotically equal. □

This provides evidence for the truth of Observations 1 for the following reason. Suppose one chooses, independently, a private monic irreducible $\boldsymbol{f}(x)$, and a $\phi(y)$, with the coefficients of $\phi(y)$ chosen randomly and uniformly from $\mathbb{F}_q$. Then with high probability there will be a corresponding (monic, irreducible) $\boldsymbol{F}_1(y)$ and a short polynomial $\boldsymbol{a}(x)$ will be mapped to $\boldsymbol{A}(y) = \boldsymbol{a}(\phi(y))$ reduced modulo $\boldsymbol{F}_1(y)$. As the coefficients of $\phi(y)$ are random and uniformly distributed modulo q it is reasonable to assume that the coefficients of $\boldsymbol{A}(y)$ will be similarly uniformly distributed modulo q. Unfortunately, because of the highly non-linear aspect of this mapping, it appears to be hard to construct a proof of this. The polynomial $\boldsymbol{F}_1(y)$ can be used as the public key. However, it may be convenient to use a polynomial of a simpler form, such as $\boldsymbol{F}_2(y) = y^n - y - 1$ to make computations easier for the public party. In this case the composite isomorphism

$$\mathbb{F}_q[x]/(\boldsymbol{f}(x)) \to \mathbb{F}_q[y]/(\boldsymbol{F}_1(y)) \to \mathbb{F}_q[y]/(\boldsymbol{F}_2(y))$$

can be used for encryption. It is again reasonable to assume, though hard to prove, that the composite mapping continues to cause coefficients of images of short polynomials to be uniformly distributed modulo q.

Remark 3. Because of Observation 2, that non-trivial isomorphisms send short polynomials in $\mathbb{X}$ to uniformly distributed elements of $\mathbb{Y}$, we believe that there are no easy cases of CFFI. Hence, similar to hard lattice problems such as those described in [1], we suspect that there may well be an average-case/worst-case equivalence for the computational finite field isomorphism problem. However, research in this direction is beyond the scope of the present paper and clearly requires considerable further study.

2.2.2 Arguments for the truth of Observation 2

In order to build a multiplicative homomorphic encryption scheme we require that products of short elements in $\mathbb{X}$ are also short. Hence, we cannot simply sample $\boldsymbol{f}(x)$ uniformly from $\mathcal{M}_{n,q}$. Instead, we will sample $\boldsymbol{f}(x)$ uniformly from $\mathcal{M}_{n,q}$ with the requirement that $\|\boldsymbol{f}(x)\|$ is bounded.

In order to estimate the size of the search space for $\boldsymbol{f}(x)$, we will rely on the following very reasonable assumption:

Assumption 1 *Monic irreducible polynomials are uniformly distributed over* $\mathbb{F}_q[x]$.

This assumption implies that Observation 2 is true. It also implies (together with the argument that $|\mathcal{M}_{n,q}|$ is on the order of q^n/n) that for $1 \le \beta \le \frac{1}{2}q$ there are approximately $(2\beta)^n/n$ distinct irreducible monic polynomials $\boldsymbol{a}(x)$ over $\mathbb{F}_q[x]$ satisfying $\|\boldsymbol{a}(x)\| \le \beta$. This quantifies the size of the set of all possible $\boldsymbol{f}$ and enables us to verify that with well chosen parameters it is large enough to be robust against a brute force search.

This shortness of $\boldsymbol{f}(x)$ is exploited via the following useful property:

Property 1. If $\boldsymbol{f}(x)$ is short, and if $\boldsymbol{a}(x)$ and $\boldsymbol{b}(x)$ are short elements of $\mathbb{X}$, then the product $\boldsymbol{a}(x)\boldsymbol{b}(x) \bmod \boldsymbol{f}(x)$ is also a reasonably short element of $\mathbb{X}$.

As remarked earlier, Property 1 has been widely exploited in ideal and lattice-based cryptography, especially with $\boldsymbol{f}(x) = x^n \pm 1$, starting with the original NTRUEncrypt [25].

2.3 An Algorithm to Find an Isomorphism

We explain how to find suitable polynomials $\boldsymbol{f}(x)$ and $\boldsymbol{F}(y)$ and an explicit isomorphism $\mathbb{F}_q[x]/(\boldsymbol{f}(x)) \mapsto \mathbb{F}_q[y]/(\boldsymbol{F}(y))$. We need to find four polynomials $(\boldsymbol{f}, \boldsymbol{F}, \boldsymbol{\phi}, \boldsymbol{\psi})$ satisfying:

- $\boldsymbol{f}(x) \in \mathbb{F}_q[x]$ is irreducible monic of degree n with $\|\boldsymbol{f}(x)\| \le \beta$.
- $\boldsymbol{F}(y) \in \mathbb{F}_q[y]$ is irreducible monic of degree n with random coefficients.
- $\boldsymbol{\phi}(y) \in \mathbb{F}_q[y]$ and $\boldsymbol{\psi}(x) \in \mathbb{F}_q[x]$ have degree less than n.
- $\boldsymbol{F}(y) \bigm| \boldsymbol{f}\bigl(\boldsymbol{\phi}(y)\bigr)$.
- $\boldsymbol{\phi}\bigl(\boldsymbol{\psi}(x)\bigr) \equiv x \pmod{\boldsymbol{f}(x)}$.

The algorithm for finding such an isomorphism is shown in Algorithm 1.

Remark 4. We note again that the secret polynomial $\boldsymbol{f}(x)$ and the public polynomial $\boldsymbol{F}(y)$ are chosen *independently*, so in particular, knowledge of $\boldsymbol{F}(y)$ reveals no information about $\boldsymbol{f}(x)$. In other words, any polynomial satisfying the norm

Algorithm 1 Finite Field Isomorphism Generation

```
1: Sample $\boldsymbol{f}(x)$ and $\boldsymbol{F}(y)$ as required.
2: Find a root of $\boldsymbol{f}(x)$ in the finite field $\mathbb{F}_q[y]/(\boldsymbol{F}(y)) \cong \mathbb{F}_{q^n}$ and lift this root to a
   polynomial $\boldsymbol{\phi}(y) \in \mathbb{F}_q[y]$ of degree less than $n$.
3: Construct the unique polynomial $\boldsymbol{\psi}(x) \in \mathbb{F}_q[x]$ of degree less than $n$ satisfying
   $\boldsymbol{\psi}\big(\boldsymbol{\phi}(y)\big) \equiv y \pmod{\boldsymbol{F}(y)}$.
4: return $\boldsymbol{f}(x)$, $\boldsymbol{F}(y)$, $\boldsymbol{\phi}(y)$ and $\boldsymbol{\psi}(x)$.
```

bound is a potential candidate for $\boldsymbol{f}(x)$. The attacker only begins to acquire information about $\boldsymbol{f}(x)$ when she is given isomorphic images in $\mathbb{Y}$ of (short) polynomials in $\mathbb{X}$. Further, the fact that there are no security issues in the choice of $\boldsymbol{F}(y)$, other than the requirement that it be irreducible in $\mathbb{F}_q[y]$, means that $\boldsymbol{F}(y)$ may be chosen to simplify field operations in the quotient field $\mathbb{F}_q[y]/(\boldsymbol{F}(y))$. For example, one could take $\boldsymbol{F}(y)$ to be a trinomial. The point is that the attacker can always replace your $\boldsymbol{F}(y)$ with her choice of $\boldsymbol{F}'(y)$, since she can easily construct an isomorphism from $\mathbb{F}_q[y]/(\boldsymbol{F}(y))$ to $\mathbb{F}_q[y]/(\boldsymbol{F}'(y))$.

We now discuss the steps in the generation algorithm in more details. In Step 2, we are required to find a root of a polynomial $\boldsymbol{f}(x)$ in a finite field $\mathbb{F}_{q^n}$ that is given explicitly as a quotient $\mathbb{F}_q[y]/(\boldsymbol{F}(y))$. There are fast polynomial-time algorithms for doing this.[4] We note that in our set-up, the polynomial $\boldsymbol{f}(x)$ is irreducible of degree n, so any one of its roots generates the field $\mathbb{F}_{q^n}$, and since any two fields with q^n elements are isomorphic, it follows that $\boldsymbol{f}(x)$ must have a root in $\mathbb{F}_q[y]/(\boldsymbol{F}(y))$. Further, since $\mathbb{F}_{q^n}/\mathbb{F}_q$ is Galois, any irreducible polynomial with one root must split completely, so in fact $\boldsymbol{f}(x)$ has n distinct roots in $\mathbb{F}_q[y]/(\boldsymbol{F}(y))$. We may take $\boldsymbol{\phi}(y) \bmod \boldsymbol{F}(y)$ to be any one of these roots.

In Step 3, we need to construct $\boldsymbol{\psi}(x)$. We describe three ways to do this. All are efficient. Method 2 is always faster than method 1. It is not clear which is the more efficient between methods 2 and 3.

1. One can compute the roots of $\boldsymbol{F}(y)$ in $\mathbb{F}_q[x]/(\boldsymbol{f}(x))$. As above, there will be n distinct roots, and one of them will be the desired $\boldsymbol{\psi}(x)$.
2. One can compute a root of $\boldsymbol{\phi}(y) - x$ in the field $\mathbb{F}_q[x]/(\boldsymbol{f}(x))$.
3. One can use linear algebra as described in Appendix A.

2.4 Known Approaches to Recovering the Secret Isomorphism

In this section, we explore two possible methods to solve the finite field isomorphism problem. Such an isomorphism will be described as an n-by-n matrix M. The first approach is based on lattice reduction. The second approach is a highly non-linear attack of unknown but, we believe, high difficulty.

[4] For example, Pari-GP [33] provides the routine `polrootsff`.

2.4.1 Lattice Attack of (dim $\approx$ 2n) In this subsection we describe a lattice attack that uses a transcript of ciphertexts. We formulate this abstractly by saying that there is an unknown n-by-n matrix M with mod q coefficients, and there are known vectors $\boldsymbol{A}_1, \boldsymbol{A}_2, \ldots, \boldsymbol{A}_k$ with the property that the unknown vectors $M\boldsymbol{A}_i$ mod q are small for all $i = 1, 2, \ldots, k$.

For the computational isomorphism problem we would need to recover the rows of M exactly, and place them in the correct order. However, to solve the decisional problem it would suffice to search for a single row of M. The dimension of an attack lattice can be further reduced. To accomplish this, let $\boldsymbol{m}$ be some (unknown) row of M, say the j^{th} row, and let $b_i = \boldsymbol{m} \cdot \boldsymbol{A}_i$ for $i = 1, 2, \ldots, k$, be the corresponding (unknown) small values of the indicated dot products. Then

$$A = (\boldsymbol{A}_1 \mid \boldsymbol{A}_2 \mid \cdots \mid \boldsymbol{A}_k), a = (\boldsymbol{a}_1 \mid \boldsymbol{a}_2 \mid \ldots \mid \boldsymbol{a}_k), \boldsymbol{b}_j = (b_1, b_2, \ldots, b_k),$$

and we set $D = \begin{pmatrix} A \\ qI \end{pmatrix}$. Thus A and a are two n-by-k matrices, and D is an $(n + k)$-by-k matrix. The vector $\boldsymbol{b}_j$ is a k dimensonal "slice" consisting of the j^{th} coordinates of the $\boldsymbol{a}_i$, which are the inverse images in $\mathbb{X}$ of the $\boldsymbol{A}_i$. Let $\mathcal{L}(D)$ denote the row span of D, so $\dim \mathcal{L}(D) = k$. Then $\mathcal{L}(D)$ contains the short row vector of $\boldsymbol{b}_j$. If we choose k sufficiently large, then the vectors $\boldsymbol{b}_j$ will stand out as unusually short, relative to the Gaussian heuristic, and a successful lattice reduction argument would recover them, or short linear combinations of them. This means that an attacker with sufficient lattice reduction resources could solve the decisional FFI problem, in the following way. Suppose the attacker is provided with a list of $\boldsymbol{A}_i$, images in $\mathbb{Y}$ of short vectors in $\mathbb{X}$, and a vector $\boldsymbol{B}$, which might or might not be the image in $\mathbb{Y}$ of a short vector in $\mathbb{X}$. Considering

$$(\boldsymbol{A}_1 \mid \boldsymbol{A}_2 \mid \cdots \mid \boldsymbol{A}_k \mid \boldsymbol{B}),$$

a successful lattice reduction could produce a slice through the j^{th} coordinates. If each $\boldsymbol{A}_i = (a_{i,1}, a_{i,2}, \ldots, a_{i,n})^T$ then $(a_{1,j}, a_{2,j}, \ldots, a_{k,j}, b_j)$ will be in $\mathcal{L}(D)$. If $\boldsymbol{B}$ is the image of a short vector in $\mathbb{X}$ then $(a_{1,j}, a_{2,j}, \ldots, a_{k,j}, b_j)$ will have all short entries, say, around β in absolute value, and a successful lattice reduction argument should recover it. If $\boldsymbol{B}$ is not the image of a short vector in $\mathbb{X}$ then $(a_{1,j}, a_{2,j}, \ldots, a_{k,j}, b_j)$ will have k short entries and one entry that is random mod q. If the vector, with this new final entry were recovered by lattice reduction, it is highly unlikely that the random length of the final entry would be on the order of β, and, as q will be considerably larger than k, it is also highly unlikely that this output would be shorter than the gaussian heuristic expected vector. This would enable the decision problem to be solved with greater than 50% probability. The technical estimates are given in the remainder of this section.

Since $\|\boldsymbol{a}\| \leq \beta$, the length of the target vector is roughly $\|\boldsymbol{a}\|_2 \asymp \beta\sqrt{k}$. The determinant of $\mathcal{L}(D)$ is the gcd of the k-by-k minors of the matrix D. Each such minor includes at least $k - n$ rows from the bottom part of the matrix, which gives a factor of q^{k-n} to each k-by-k minor. Since the entries of A are more-or-less random, it is likely that $\det \mathcal{L}(D)$ is some small multiple of q^{k-n}. Hence the

Gaussian expected shortest vector in $\mathcal{L}(D)$ has length roughly

$$\gamma\big(\mathcal{L}(D)\big) \asymp \sqrt{\frac{\dim \mathcal{L}(D)}{2\pi e}}\big(\operatorname{Det}\mathcal{L}(D)\big)^{1/\dim \mathcal{L}(D)} = \sqrt{\frac{k}{2\pi e}} \cdot (q^{k-n})^{1/k}.$$

To analyze the hardness of recovering this vector via lattice reductions, we focus on the k-th root of the ratio between the Gaussian expected length and the unique shortest vectors:

$$\left(\frac{q^{\frac{k-n}{k}}}{\beta\sqrt{2\pi e}}\right)^{\frac{1}{k}}.$$

This attack appears to be optimal when $k \approx 2n$. In the meantime, analyses in [16] and [7] suggest that recovering this vector is hard for BKZ 2.0 algorithm when $q^{\frac{1}{4n}}\beta^{-\frac{1}{2n}} \lessapprox 1.005$.

Remark 5. This lattice is a little different from those used in instantiating the unique shortest vector problem, as in our lattice, there are roughly n unique shortest non-zero vectors of similar length. Previous results in [16] and [15] show that the hardness of finding a short vector in q-ary lattices that contain many unique shortest vectors depends not on the gap, but rather on the ratio between the Gaussian heuristic and the actual length of the shortest vector. We conjecture a similar property applies to our lattice.

2.4.2 A Non-Lattice Attack On Small Solutions

There are two pieces of structure lurking within the isomorphism $\mathbb{X} \to \mathbb{Y}$ that are not used in the lattice attack described in Section 2.4.1:

1. The map $\mathbb{X} \to \mathbb{Y}$ is a field isomorphism between two copies of $\mathbb{F}_{q^n}$, not merely an $\mathbb{F}_q$-vector space isomorphism between two copies of $\mathbb{F}_q^n$;
2. The secret polynomial $\boldsymbol{f}(x)$ used to define one of the copies of $\mathbb{F}_{q^n}$ has small coefficients. (And the attacker may, in principle, take $\boldsymbol{F}(y)$ to be any irreducible polynomial that she chooses.)

In this section we explain how to exploit these properties to formulate an attack that requires finding small solutions to systems of higher degree multivariable polynomial equations. We note that solving such systems appears to be exponentially difficult. The polynomials $\boldsymbol{f}(x)$ and $\boldsymbol{F}(y)$ almost, but not quite, determine the polynomials $\boldsymbol{\phi}(y)$ and $\boldsymbol{\psi}(x)$ used to define the isomorphism

$$\mathbb{F}_q[x]/(\boldsymbol{f}(x)) \cong \mathbb{F}_q[y]/(\boldsymbol{F}(y)).$$

More precisely, if $x \to \boldsymbol{\phi}'(y)$ is some other isomorphism, then necessarily

$$\boldsymbol{\phi}'(y) = \boldsymbol{\phi}(y)^{q^t} \pmod{\boldsymbol{F}(y)} \quad \text{for some } 0 \le t < d.$$

This follows immediately from the fact that $\operatorname{Gal}(\mathbb{F}_{q^d}/\mathbb{F}_q)$ is cyclic of order d, generated by the q-power Frobenius map. Alternatively, the possible values for $\boldsymbol{\phi}(y)$

are exactly the roots of $\boldsymbol{f}(x)$ in the field $\mathbb{F}_q[y]/(\boldsymbol{F}(y))$, so in any case there are exactly d possible $\boldsymbol{\phi}(y)$'s. As stated in Remark 4, an attacker knows no useful information about $\boldsymbol{f}(x)$ until she acquires an image, since as already noted, the public value $\boldsymbol{F}(y)$ is chosen independently of $\boldsymbol{f}(x)$. We assume that the attacker is given the value of an arbitrary number of images. As per Definition 1, the attacker is given $\boldsymbol{A}_1, \dots, \boldsymbol{A}_k \in \mathbb{Y}$ with the promise that $\boldsymbol{a}_i, \dots, \boldsymbol{a}_k \in \mathbb{X}$ are small, in other words:

$$\boldsymbol{A}_i(y) = \boldsymbol{a}_i\big(\boldsymbol{\phi}(y)\big) \bmod \boldsymbol{F}(y), \tag{3}$$

where $\boldsymbol{a}_i$ has small coefficients. The equation (3) contain $2n$ quantities that are unknown to the attacker, namely the coefficients of $\boldsymbol{a}$ and $\boldsymbol{\phi}$. Of these, the coefficients of $\boldsymbol{a}$ are small, so she can try to eliminate the coefficients of $\boldsymbol{\phi}$. We note that (3) really gives n equations for the coefficients, since both sides are polynomials of degree $n-1$. Unfortunately, this doesn't quite allow her to eliminate all n of the coefficients of $\boldsymbol{\phi}$. If she uses both $\boldsymbol{A}_1(y)$ and $\boldsymbol{A}_2(y)$, then she obtains $2n$ equations for the $3n$ unknowns consisting of the coefficients of $\boldsymbol{a}_1$, $\boldsymbol{a}_2$, and $\boldsymbol{\phi}$. So using elimination theory (as a practical matter, using Gröbner basis algorithms), she can eliminate the coefficients of $\boldsymbol{\phi}$ and obtain a system of n equations for the $2n$ coefficients of $\boldsymbol{a}_1$ and $\boldsymbol{a}_2$. These are highly non-linear equations over the field $\mathbb{F}_q$, so the attacker is faced with the problem of finding an $\mathbb{F}_q$-point with small coordinates on a high degree n-dimensional subvariety of $\mathbb{F}_q^{2n}$. As far as we are aware, there are no algorithms to solve such problems that are faster than an exhaustive (or possibly collision-based) search. Indeed, there does not appear to be an efficient algorithm to solve the decision problem of whether a small solution exists.

We note that the attacker may continue eliminating variables until eventually arriving at a single equation in $\mathbb{F}_q^{n+1}$. But this is likely to be counter-productive, since it greatly increases the degree of the underlying equation while discarding the information that the eliminated variables are small. Alternatively, the attacker can use one element in $\mathbb{Y}$ and the knowledge that there is a polynomial $\boldsymbol{f}(x)$ with small coefficients that satisfies

$$\boldsymbol{f}\big(\boldsymbol{\phi}(y)\big) = 0 \bmod \boldsymbol{F}(y). \tag{4}$$

Thus (3) and (4) again provide $2n$ equations, this time for the $3n$ coefficients of $\boldsymbol{a}$, $\boldsymbol{f}$, and $\boldsymbol{\phi}$. The first two polynomials have small coefficients, so eliminating the coefficients of $\boldsymbol{\phi}$ again yields an n-dimensional subvariety in $\mathbb{F}_q^{2n}$ on which the attacker must find a small point.

3 Fully Homomorphic Encryption based on DFFI

In this section we use the approach of López-Alt et. al. [28] to show how to turn our scheme into a fully homomorphic encryption scheme. First, we present Gentry's definitions and theorems on fully homomorphic encryption [17, 18]. Later, we show that our scheme satisfies the definitions on somewhat homomorphism, but it does not reach the circuit depth required for evaluating decryption circuit homomorphically. We resolve the issue by turning our scheme into a leveled homomorphic encryption scheme using a technique to reduce the noise growth from doubly exponential to singly exponential. We then describe our leveled homomorphic scheme and show that it is fully homomorphic by showing that it is able to evaluate its decryption circuit homomorphically.

3.1 Fully Homomorphic Encryption Definitions

We give the definitions of fully homomorphic encryption and leveled homomorphic encryption.

Definition 31 ($\mathcal{C}$-Homomorphic Encryption [6]). *Let $\mathcal{C} = \{\mathcal{C}_\kappa\}_{\kappa \in \mathbb{N}}$ be a class of functions with security parameter κ. A scheme $\mathcal{E}$ is $\mathcal{C}$-homomorphic if for any sequence of functions $f_\kappa \in \mathcal{C}_\kappa$ and respective inputs $\mu_1, \ldots, \mu_\ell \in \{0,1\}$ (where $\ell = \ell(\kappa)$), it is true that*

$$\mathrm{PR}[\mathcal{E}.\mathsf{Dec}_{sk}(\mathcal{E}.\mathsf{Eval}_{evk}(f, c_1, \ldots, c_\ell)) \neq f(\mu_1, \ldots, \mu_\ell)] = \mathrm{negl}(\kappa),$$

where (pk, evk, sk)$\leftarrow \mathcal{E}.\mathsf{KeyGen}(1^\kappa)$ and $c_i \leftarrow \mathcal{E}.\mathsf{Enc}_{pk}(\mu_i)$.

Definition 32 (Fully Homomorphic Encryption [28]). *An encryption scheme $\mathcal{E}$ is fully homomorphic if it satisfies the following properties:*

Correctness: *$\mathcal{E}$ is $\mathcal{C}$-homomorphic for the class $\mathcal{C}$ of all circuits.*
Compactness: *The computational complexity of $\mathcal{E}$'s algorithms is polynomial in the security parameter κ, and in the case of the evaluation algorithm, i.e. the size of the circuit.*

Now as given in [28], we continue with the leveled homomorphic encryption definition that is taken from [5]. It is a modified definition of fully homomorphic encryption (Definition 32) into a leveled homomorphic encryption scheme. It removes the requirement that the scheme is able to evaluate all possible circuits and instead imposes a circuit depth D. It requires the scheme to be able to evaluate all circuits (including the decryption circuit) that are depth at most D.

Definition 33 (Leveled Homomorphic Encryption [28]). *Let $\mathcal{C}^{(D)}$ be the class of all circuits of depth at most D (that use some specified complete set of gates). We say that a family of homomorphic encryption schemes $\{\mathcal{E}^{(D)} : D \in \mathbb{Z}^+\}$ is leveled fully homomorphic if, for all $D \in \mathbb{Z}^+$, it satisfies the following properties:*

Correctness: *$\mathcal{E}^{(D)}$ is $\mathcal{C}^{(D)}$-homomorphic.*
Compactness: *The computational complexity of $\mathcal{E}^{(D)}$s algorithms is polynomial in the security parameter κ and D, and in the case of the evaluation algorithm, the size of the circuit. We emphasize that this polynomial must be the same for all D.*

3.2 Somewhat Homomorphic FF-Encrypt Construction

We present a somewhat homomorphic version of our FF-Encrypt construction. We first give the details of our construction, and then we prove that our scheme is able to evaluate homomorphic circuits (multiplications and additions) of bounded depth.

3.2.1 Preliminaries

Here we give some preliminary notation and information that we use for the construction of our homomorphic schemes:

- The error distribution χ is a truncated Gaussian distribution $D_{\mathbb{Z}_r^n}$ with standard deviation r.
- The random polynomials $\boldsymbol{r}(x)$ are ephemeral short noise polynomials that are sampled from χ.
- The message space uses a fixed polynomial $\boldsymbol{p}(x)$, which we take for this instantiation to be the number 2.
- The message $\boldsymbol{m}(x)$ consists of a monomial with a single coefficient that is chosen from $\{0,1\}$.

Polynomial Multiplication Noise in $\mathbb{X}$. The noise of the product of two polynomials is significantly affected by the choice of the polynomial $\boldsymbol{f}(x)$. Two factors that affect noise growth are the choice of the coefficient bound β_f for $\boldsymbol{f}(x)$ and the degree $d := \deg(\boldsymbol{f}'(x))$, where we write $\boldsymbol{f}(x) = x^n + \boldsymbol{f}'(x)$. The noise bound for the product of two β-bounded polynomial $\boldsymbol{a}(x)$ and $\boldsymbol{b}(x)$ for $d < n/2$ satisfies

$$\left\|\boldsymbol{a}(x)\boldsymbol{b}(x) \bmod \boldsymbol{f}(x)\right\|_\infty \leq n[(d+1)^2+1]\beta^2. \tag{5}$$

A detailed noise analysis for general $\boldsymbol{f}(x)$ is given in Appendix B.

3.2.2 Secret-Key Instantiation

The secret key version of our Somewhat Homomorphic Finite Field scheme uses the following four algorithms:

– $\mathsf{SHFF\text{-}SK.Keygen}(1^\kappa)$:
  - Input a security parameter κ.
  - Generate a parameter set $\Xi = \{n, q, \beta\}$ as a function of κ.
  - Use Algorithm 1 (from the FF-Encrypt paper) to generate a finite field homomorphism $\{\boldsymbol{f}, \boldsymbol{F}, \boldsymbol{\psi}, \boldsymbol{\phi}\}$.
  - Output $\{\boldsymbol{f}, \boldsymbol{F}, \boldsymbol{\psi}, \boldsymbol{\phi}\}$. Also output $\boldsymbol{p}(x)$ and $\gamma > 0$.

– $\mathsf{SHFF\text{-}SK.Enc}(\boldsymbol{f}, \boldsymbol{F}, \boldsymbol{\phi}, \boldsymbol{m})$:
  - Encode a plaintext by some method into a short polynomial $\boldsymbol{m}(x) \in \mathbb{X}$;
  - Sample a polynomial $\boldsymbol{r}(x) \in \mathbb{X}$ from the distribution χ_β.
  - Compute $\boldsymbol{C}(y) = \boldsymbol{p}(\boldsymbol{\phi}(y))\boldsymbol{r}(\boldsymbol{\phi}(y)) + \boldsymbol{m}(\boldsymbol{\phi}(y)) \mod \boldsymbol{F}(y)$.
  - Output $\boldsymbol{C}(y)$ as the ciphertext.

– $\mathsf{SHFF\text{-}SK.Dec}(\boldsymbol{f}, \boldsymbol{\psi}, \boldsymbol{C})$:
  - For a ciphertext $\boldsymbol{C}(y)$, compute $\boldsymbol{c}'(x) = \boldsymbol{C}(\boldsymbol{\psi}(x))$.
  - Output $\boldsymbol{m}'(x) = \boldsymbol{c}'(x) \bmod \big(\boldsymbol{p}(x), \boldsymbol{f}(x)\big)$.

– $\mathsf{SHFF\text{-}SK.Eval}(C, \boldsymbol{C}_1, \boldsymbol{C}_2, \ldots, \boldsymbol{C}_\ell)$:
  - The circuit C is represented by two input binary arithmetic circuits with gates $\{+, \times\}$. Then, we can evaluate the circuit C homomorphically, since we can perform homomorphic addition and homomorphic multiplication.

3.2.3 Public-Key Instantiation

The public key version of our Somewhat Homomorphic Finite Field scheme is similar to the secret key instantiation in most aspects. We use a subset sum problem to instatiate the public key version. The scheme uses the following four algorithms:

- $\mathsf{SHFF\text{-}PK.Keygen}(1^{\kappa})$:
  - Perform the key generation as in secret key instantiation $\mathsf{SHFF\text{-}SK.Keygen}(1^{\kappa})$.
  - Choose two integers S, s which $\binom{\mathsf{S}}{\mathsf{s}} > 2^{\kappa}$ for security parameter κ.
  - Set $c_i = \mathsf{SHFF\text{-}SK.Enc}(\boldsymbol{f}, \boldsymbol{F}, \boldsymbol{\phi}, 0)_i$, create an array of zero encryptions $\mathsf{pk} = \mathcal{S} = \{\boldsymbol{C}_0(y), \boldsymbol{C}_1(y), \ldots, \boldsymbol{C}_{\mathsf{S}-1}(y)\}$.
- $\mathsf{SHFF\text{-}PK.Enc}(\mathcal{S}, \boldsymbol{m})$:
  - Choose s random encryptions of zero $\boldsymbol{C}_i(y)$ from $\mathcal{S}$ and compute their summation with message $\boldsymbol{C}(y) = \sum_{i=\mathrm{rand}(\mathsf{S})} \boldsymbol{C}_i(y) + \boldsymbol{M}(y)$ in which $\boldsymbol{M}$ is the representation of the message m in $\mathbb{Y}$.
  - Output $\boldsymbol{C}(y)$ as the ciphertext.
- $\mathsf{SHFF\text{-}PK.Dec}(\boldsymbol{f}, \boldsymbol{\psi}, \boldsymbol{C})$:
  - Compute and output $\mathsf{SHFF\text{-}SK.Dec}(\boldsymbol{f}, \boldsymbol{\psi}, \boldsymbol{C})$.
- $\mathsf{SHFF\text{-}PK.Eval}(C, \boldsymbol{C}_1, \boldsymbol{C}_2, \ldots, \boldsymbol{C}_{\ell})$:
  - Compute and output $\mathsf{SHFF\text{-}SK.Eval}(C, \boldsymbol{C}_1, \boldsymbol{C}_2, \ldots, \boldsymbol{C}_{\ell})$.

The noise and depth performance of this scheme is captured by the following Lemma.

Lemma 2. *The encryption scheme*

$$\mathcal{E}_{\mathsf{SHFF}} = (\mathsf{SHFF.KeyGen}, \mathsf{SHFF.Enc}, \mathsf{SHFF.Dec}, \mathsf{SHFF.Eval})$$

described above is somewhat homomorphic for circuits having depth less than $D < \log\log q - \log(3\log n)$ *where* $q = 2^{n^{\varepsilon}}$ *with* $\varepsilon \in (0, 1)$*, and* χ *is a* β*-bounded Gaussian distribution for random sampling.*

Proof. We denote the encryptions of two messages $\boldsymbol{m}_1$ and $\boldsymbol{m}_2$ by $\boldsymbol{C}_1(y)$ and $\boldsymbol{C}_2(y)$. Then we want the noise of the ciphertexts after an addition or a multiplication to be smaller than $q/2$ so that it can be correctly decrypted.

Addition. Set $\boldsymbol{C}(y) = \boldsymbol{C}_1(y) + \boldsymbol{C}_2(y)$. Dropping y from the notation, we have $\boldsymbol{C} = \left(\sum \boldsymbol{p}(\boldsymbol{\phi})\boldsymbol{r}_1(\boldsymbol{\phi}) + \boldsymbol{m}_1(\boldsymbol{\phi})\right) + \left(\sum \boldsymbol{p}(\boldsymbol{\phi})\boldsymbol{r}_2(\boldsymbol{\phi}) + \boldsymbol{m}_2(\boldsymbol{\phi})\right)$. Apply $\boldsymbol{\psi}(x)$ as the first step of the decryption $\boldsymbol{C}(x) = \left(\sum \boldsymbol{p}(x)\boldsymbol{r}_1(x) + \boldsymbol{m}_1(x)\right) + \left(\sum \boldsymbol{p}(x)\boldsymbol{r}_2(x) + \boldsymbol{m}_2(x)\right)$. Then the infinity norm of $\boldsymbol{C}(x)$ is $\|\boldsymbol{C}(x)\|_{\infty} = 2\mathsf{s}\beta'$.

Multiplication. We compute

$$\begin{aligned}\boldsymbol{C} &= \Big(\sum \boldsymbol{p}(\boldsymbol{\phi})\boldsymbol{r}_1(\boldsymbol{\phi}) + \boldsymbol{m}_1(\boldsymbol{\phi})\Big) \cdot \Big(\sum \boldsymbol{p}(\boldsymbol{\phi})\boldsymbol{r}_2(\boldsymbol{\phi}) + \boldsymbol{m}_2(\boldsymbol{\phi})\Big)\\ &= \sum \boldsymbol{p}(\boldsymbol{\phi})^2\boldsymbol{r}_1(\boldsymbol{\phi})\boldsymbol{r}_2(\boldsymbol{\phi}) + \sum \boldsymbol{p}(\boldsymbol{\phi})\boldsymbol{r}_1(\boldsymbol{\phi})\boldsymbol{m}_2(\boldsymbol{\phi}) + \sum \boldsymbol{p}(\boldsymbol{\phi})\boldsymbol{r}_2(\boldsymbol{\phi})\boldsymbol{m}_1(\boldsymbol{\phi}) + \boldsymbol{m}_1(\boldsymbol{\phi})\boldsymbol{m}_2(\boldsymbol{\phi}).\end{aligned}$$

We calculate the infinity norm of $\boldsymbol{C}(x)$ using Equation 5,

$$\|\boldsymbol{C}(x)\|_{\infty} = n\big((d+1)^2 + 1\big)(\mathsf{s}\beta')^2 + 2\mathsf{s}\beta'.$$

Multiplicative Level $\boldsymbol{D}$. For D-level homomorphic operations, we need to compute the bound of $\left\| \left(\boldsymbol{p}(x)\boldsymbol{r}(x) + \boldsymbol{m}(x)\right)^{2^D} \right\|_\infty$. Since $\boldsymbol{p}(x)\boldsymbol{r}(x) \gg \boldsymbol{m}(x)$, this is essentialy equal to $\left\| \left(\boldsymbol{p}(x)\boldsymbol{r}(x)\right)^{2^D} \right\|_\infty$. This gives an error bound equal to $(nd')^{2^D-1}(\mathsf{s}\beta')^{2^D}$ with $d' = (d+1)^2+1$. We want this noise to be smaller than $q/2$, so we impose the inequality $(nd')^{2^D-1}(\mathsf{s}\beta')^{2^D} < q/2$. Taking the logarithms, we rewrite this as $(2^D-1)\log(nd')+(2^D)\log(\mathsf{s}\beta') < \log q - 1$ Taking logarithm again yields $D+\log(\log(nd')+\log(\mathsf{s}\beta')) < \log(\log q + \log(nd') - 1)$. We can simplify this inequality by noting that $d' \approx n^2/4$, which makes $\log(nd') \approx 3\log(n) > \log(sB')$ and $\log(q) > 3\log(n)$. Omitting small terms, we obtain

$$D < \log\log q - \log(3\log n)$$

Taking $q = 2^{n^\varepsilon}$, our upper bound for the multiplicative depth D is $\mathcal{O}(\varepsilon \log n)$.□

3.2.4 Security

Our construction relies on two security assumptions. The first assumption is the hardness of the Decisional Finite Field Isomorphism problem, which ensures that small norm elements in $\mathbb{X}$ are mapped to random-looking elements in $\mathbb{Y}$. The mapping function is secret, and an attacker has to find some way of identifying images of short objects in $\mathbb{X}$ in order to break the scheme. The second assumption is the difficulty of the subset sum problem that is used to generate encryptions of 0 to add to encryptions of messages. We will choose s ciphertexts from a list length S, so the pair of parameters (S, s) should give reasonable combinatorial security, e.g., $\binom{\mathsf{S}}{\mathsf{s}} > 2^{256}$. Beyond combinatorial security, solving this subset sum problem and identifying an encryption of 0 can be translated into a lattice reduction problem in the rows of an S by $S+n$ matrix, which can be made arbitrarily difficult. In particular $S > 2n$ should suffice. We prove the semantic security via the following theorem.

Theorem 1. *If there is an algorithm $\mathcal{A}$ that breaks the semantic security with parameter $\Xi = \{n, q, \beta\}$ and $\boldsymbol{p}(x) = p$, i.e., if one inputs of any public keys $(\boldsymbol{C}_1, \ldots, \boldsymbol{C}_k)$, a ciphertext $\boldsymbol{D}$ which encrypts a message m of either 0 or 1, and $\mathcal{A}$ outputs the message m with probability $1/2 + \epsilon$ for some non-negligible $\epsilon > 0$, then there exist another algorithm $\mathcal{B}$ that solves the decisional FFI with parameter $\{n, q, \beta/p\}$ with probability $1/2 + \epsilon$.*

Proof. Notice that if the input $(\boldsymbol{C}_1, \ldots, \boldsymbol{C}_k, \boldsymbol{D})$ to algorithm $\mathcal{A}$ is invalid (either $\boldsymbol{D}$ cannot be written as subset sum of $\boldsymbol{C}_i$, or $\boldsymbol{D}$ does not encrypt 0 or 1), it will either output an error or output 0 or 1 with equal probability. On the other hand, if the input is valid, it will output the correct m with probability $1/2 + \epsilon$.

Now we can use $\mathcal{A}$ to build an algorithm $\mathcal{B}$ as follows. Let $\boldsymbol{A}_1, \ldots, \boldsymbol{A}_k, \boldsymbol{B}_1, \boldsymbol{B}_2$ be the input to the decisional FFI problem. Upon receiving those inputs, algorithm $\mathcal{A}$ calls algorithm $\mathcal{B}$ with a "public key" $(p\boldsymbol{B}_1, p\boldsymbol{A}_2, \ldots, p\boldsymbol{A}_k)$ and a ciphertext $\boldsymbol{0}$. Therefore, if $\boldsymbol{B}_1$ has short images in $\mathbb{X}$, then $(p\boldsymbol{B}_1, p\boldsymbol{A}_2, \ldots, p\boldsymbol{A}_k)$ is a legit public key, while if $\boldsymbol{B}_1$ is uniformly sampled in $\mathbb{Z}_q[x]$, then the probability of $(p\boldsymbol{B}_1, p\boldsymbol{A}_2, \ldots, p\boldsymbol{A}_k)$ been a legitimate public key is negligible, roughly $(\frac{\beta}{pq})^n$.

Notice that $\mathbf{0}$ is a subset sum of the "public key" regardless if the "public key" is legitimate or not. So from $\mathcal{A}$'s point of view, $\mathbf{0}$ is a legit ciphertext that encrypts 0 if $\boldsymbol{B}_1$ has a short image. Upon receiving those public key and ciphertext, $\mathcal{A}$ will return 0 with probability $1/2 + \epsilon$ if $\boldsymbol{B}_1$ has a short image. It will return error or random if $\boldsymbol{B}_1$ doesn't. Thus $\mathcal{B}$ solves the decisional FFI with probability $1/2 + \epsilon$.□

For completeness sake, we also show that if one can solve the Decisional FFI, one can also break the semantic security. Given a ciphertext $\boldsymbol{C}$ with an image $\boldsymbol{c} = \boldsymbol{pr} + \ell \boldsymbol{m}$, one can compute $\boldsymbol{p}^{-1}\boldsymbol{C} \bmod q$ (assuming $\boldsymbol{p}$ is an integer, say 2) which has a reverse image $\boldsymbol{r} + \boldsymbol{p}^{-1}\ell\boldsymbol{m}$. If $m = 0$, this quantity will be short. If $m = 1$, this quantity will be of length $\|\boldsymbol{p}^{-1}\ell\ \boldsymbol{r} \bmod q\|$. This is highly probable to be large, as if, say, $\boldsymbol{p} = 2$, then $\|\boldsymbol{p}^{-1} \bmod \boldsymbol{r} \bmod q\|$ will probably be of a size that takes random values mod q as ℓ varies.

3.3 From Somewhat to Fully Homomorphic Encryption

We give the definitions of bootstrappable scheme and weak circular security [17, 18]. Later, we use these two definitions to describe the bootstrapping theorem.

Definition 34 (Bootstrappable Scheme [18]). *Let $\mathcal{E} = (\mathsf{Keygen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ be a $\mathcal{C}$-homomorphic encryption scheme, and let f_{add} and f_{mult} be the augmented decryption functions of the scheme defined as*

$$\begin{aligned} f_{\mathsf{add}}^{c_1,c_2}(\mathsf{sk}) &= \mathsf{Dec}(\mathsf{sk}, c_1) \quad \mathsf{XOR} \quad \mathsf{Dec}(\mathsf{sk}, c_2), \\ f_{\mathsf{mult}}^{c_1,c_2}(\mathsf{sk}) &= \mathsf{Dec}(\mathsf{sk}, c_1) \quad \mathsf{AND} \quad \mathsf{Dec}(\mathsf{sk}, c_2). \end{aligned}$$

Then we say that $\mathcal{E}$ is bootstrappable *if $\{f_{\mathsf{add}}^{c_1,c_2}, f_{\mathsf{mult}}^{c_1,c_2}\}_{c_1,c_2} \subseteq \mathcal{C}$, i.e., if $\mathcal{E}$ can homomorphically evaluate f_{add} and f_{mult}.*

Definition 35 (Weak Circular Security [18]). *A public-key encryption scheme $\mathcal{E} = (\mathsf{Keygen}, \mathsf{Enc}, \mathsf{Dec})$ is* weakly circular secure *if it is IND-CPA secure even for an adversary with auxiliary information containing encryptions of all secret key bits: $\{\mathsf{Enc}(\mathsf{pk}, \mathsf{sk}[i])\}_i$. In other words, no polynomial-time adversary can distinguish an encryption of* 0 *from an encryption of* 1*, even given this additional information.*

Theorem 2. *Let $\mathcal{E}$ be a bootstrappable scheme that is also weakly circular secure. Then there exists a fully homomorphic encryption scheme $\mathcal{E}'$.*

In its current construction, our scheme is not bootstrappable, because it **cannot reach the required multiplicative depth for decryption**. For details on the evaluation of the depth of decryption circuit, see Section 3.3.5. The current scheme is only able to compute circuits with depth $\varepsilon \log(n)$. In order to convert our scheme into a bootstrappable one, in the next section we introduce a multiplication method with better noise management. This helps to significantly improve the depth of the circuits that the scheme can evaluate.

3.3.1 Regular Multiplication

A straightforward multiplication in the SHFF scheme causes the noise to grow doubly exponentially $(nd')^{2^D-1}(\mathsf{s}\beta')^{2^D}$ with respect to the level D. To reduce the growth to singly exponential, we introduce a multiplication technique similar to the flattening in [21]. In rest of this section for notational simplicity, we drop x and y and represent elements of $\mathbb{X}$ with lowercase letters and elements of $\mathbb{Y}$ with uppercase letters, e.g., $\boldsymbol{r} \in \mathbb{X}$ and $\boldsymbol{R} \in \mathbb{Y}$ satisfy $\boldsymbol{r}(\boldsymbol{\phi}(y)) = \boldsymbol{R}(y)$. We first consider the product for two ciphertexts, $\boldsymbol{C}_1 = \sum \boldsymbol{P}\boldsymbol{R}_1 + \boldsymbol{M}_1$ and $\boldsymbol{C}_2 = \sum \boldsymbol{P}\boldsymbol{R}_2 + \boldsymbol{M}_2$. To ease notation we write $\overline{\boldsymbol{R}} = \sum \boldsymbol{R}$. Then $\boldsymbol{C}_1 \cdot \boldsymbol{C}_2 = \boldsymbol{P}^2\overline{\boldsymbol{R}}_1\overline{\boldsymbol{R}}_2 + \boldsymbol{P}\overline{\boldsymbol{R}}_1\boldsymbol{M}_2 + \boldsymbol{P}\overline{\boldsymbol{R}}_2\boldsymbol{M}_1 + \boldsymbol{M}_1\boldsymbol{M}_2$.

Remark 6. Obviously this method creates a significant noise term $\boldsymbol{P}^2\overline{\boldsymbol{R}}_1\overline{\boldsymbol{R}}_2 + \boldsymbol{P}\overline{\boldsymbol{R}}_1\boldsymbol{M}_2 + \boldsymbol{P}\overline{\boldsymbol{R}}_2\boldsymbol{M}_1$. If we map it back to $\mathbb{X}$, the norm of the noise is bounded by $\|\boldsymbol{p}^2\mathsf{s}^2\boldsymbol{r}^2 + 2\boldsymbol{p}\mathsf{s}\boldsymbol{r}\|$ for $m \in \{0, 1\}$.

We look at the steps more closely. If we expand the second ciphertext $\boldsymbol{C}_2(y)$ and and do not expand $\boldsymbol{C}_1(y)$, we obtain $\boldsymbol{C}_1 \cdot \boldsymbol{C}_2 = \boldsymbol{P}\overline{\boldsymbol{R}}_2\boldsymbol{C}_1 + \boldsymbol{C}_1\boldsymbol{M}_2$. Here $\boldsymbol{C}_1\boldsymbol{M}_2$ gives the desired message product, with the side effect that the $\boldsymbol{P}\overline{\boldsymbol{R}}_2\boldsymbol{C}_1$ term adds a significant amount of noise. To curb the noise growth, we have to find a way to evaluate $\boldsymbol{C}_1\boldsymbol{M}_2$ while avoiding $\boldsymbol{P}\overline{\boldsymbol{R}}_2\boldsymbol{C}_1$.

3.3.2 Multiplication with Noise Management

In this section we explain the idea behind computing the ciphertext product while avoiding the noisy $\boldsymbol{P}\overline{\boldsymbol{R}}_2\boldsymbol{C}_1$ term. To achieve this we change the format of the ciphertexts and define two ciphertext operands: the Left-Hand-Side (LHS) and the Right-Hand-Side (RHS).

LHS Operand: The LHS-operand format is simply a matrix formed by bit decomposition of the ciphertext. We write $\hat{\boldsymbol{C}}^{\boldsymbol{m}}_{\mathrm{BD}}$ for the bit decomposition matrix of the ciphertext $\boldsymbol{C} = \boldsymbol{P}\overline{\boldsymbol{R}} + \boldsymbol{M}$ with message $\boldsymbol{m}(x)$. We denote the elements of the matrix by $\boldsymbol{C}_{i,j} = \hat{\boldsymbol{C}}^{\boldsymbol{m}}_{\mathrm{BD}}[i][j]$ for $0 < i < n$ and $0 < j < \ell$. More precisely, in the matrix, the entry $\boldsymbol{C}_{i,j}$ denotes the j^{th} bit of the i^{th} coefficient of $\boldsymbol{C}$. From this point on, we denote matrices by using a hat on top of the letters, e.g., $\hat{\boldsymbol{C}}$ means that it is a matrix.

RHS Operand: We create an n-by-ℓ matrix $\hat{\boldsymbol{C}}$, where each entry is a ciphertext that holds the message $\boldsymbol{m}$ with a specific construction. For simplicity we drop the indices on $\overline{\boldsymbol{R}}$, so each $\overline{\boldsymbol{R}}$ represents a different sample. Then, the entries of the matrix are computed as $\hat{\boldsymbol{C}}^{\boldsymbol{m}}[i][j] = \boldsymbol{P}\overline{\boldsymbol{R}}_{i,j} + 2^i\boldsymbol{\psi}(\boldsymbol{\phi})^j\boldsymbol{M} \quad \text{for} \quad 0 \leq i < n$ and $0 \leq j < \ell$. Note that with each new row, we multiply the message by 2, and for each new column, we increase the power of $\boldsymbol{\psi}(\boldsymbol{\phi})$. Since $y = \boldsymbol{\psi}(\boldsymbol{\phi})$, this matrix is equal to $\hat{\boldsymbol{C}}^{\boldsymbol{m}}[i][j] = \boldsymbol{P}\overline{\boldsymbol{R}}_{i,j} + 2^iy^j\boldsymbol{M} \quad \text{for} \quad 0 \leq i < n$ and $0 \leq j < \ell$.

One-Sided Homomorphic Multiplication: In the first method we use an LHS operand and an RHS operand to create an LHS operand, i.e., $\mathsf{LHS} = \mathsf{LHS} \times \mathsf{RHS}$. The homomorphic product is computed by computing a component-wise product followed by a summation over the products:

$$\langle \hat{\boldsymbol{C}}^{\boldsymbol{m}_1}_{\mathrm{BD}}, \hat{\boldsymbol{C}}^{\boldsymbol{m}_2} \rangle = \sum_{i<n}\sum_{j<\ell} \boldsymbol{C}_{i,j} \cdot \left(\boldsymbol{P}\overline{\boldsymbol{R}}_{i,j} + 2^jy^i\boldsymbol{M}_2\right) = \sum\sum \boldsymbol{P}\overline{\boldsymbol{R}}_{i,j} + \boldsymbol{P}\overline{\boldsymbol{R}}_1\boldsymbol{M}_2 + \boldsymbol{M}_1\boldsymbol{M}_2.$$

If we look more closely, each column in the component-wise product creates an encrypted version of the coefficients of the ciphertext $\boldsymbol{C}_1$. The result of the product is a standard FF-Encrypt ciphertext. To continue using the result, we apply bit decomposition BD to obtain an LHS ciphertext. An LHS operand can be computed from a regular ciphertext on the fly via bit-decomposition. An RHS operand must be constructed before it is given to the cloud/server. This means that the ciphertext size grows by a factor of $n\ell$ for RHS operands only.

Remark 7. Noise growth in multiplications is significantly reduced compared to the earlier method. Using this one-sided multiplication approach and having fresh ciphertexts on the right-hand side, with flattening we obtain a new noise bound of $n\ell\|p\mathsf{s}r\|$. Therefore the noise growth is no longer doubly exponential, and we can support deep evaluations with reasonably sized parameters as long as we restrict evaluations to be one sided evaluations. This may be achieved by expressing the circuit first using NAND gates and then evaluating left to right similar to GSW.

Remark 8. Another significant contribution is that we eliminate polynomial multiplications and only perform polynomial additions. This way, the effect of $\boldsymbol{f}(x)$ is omitted for noise analysis, i.e., it does not have any effect on noise.

Lemma 3. *Let n be the polynomial degree, let $q = 2^{n^{\varepsilon}}$ be the modulus, let $\chi = D_{\mathbb{Z}^n,r}$ be the β-bounded Gaussian distribution, and let D be the multiplicative level. Then, the proposed One-Sided Homomorphic Multiplication algorithm has noise bound $(2^D - 1)(n\ell + 1)\|p\mathsf{s}r\| = O(2^D n \log q)$ for fixed s and β.*

Generic Homomorphic Multiplication: This second method uses two RHS operands to do multiplication and achieves an RHS product as the result of the multiplication, i.e., $\mathsf{RHS} = \mathsf{RHS} \times \mathsf{RHS}$. The multiplication is similar to the multiplication algorithm for LHS and RHS operands. We represent an element (ciphertext) in the RHS operand matrix as $\boldsymbol{C}^{\boldsymbol{m}}[k][l]$ (k^{th} row and l^{th} column). In order to compute all the elements in the matrix we compute the following:

$$\begin{aligned}\boldsymbol{C}^{\boldsymbol{m}_1\cdot\boldsymbol{m}_2}[k][l] = \langle \hat{\boldsymbol{C}}^{\boldsymbol{m}_1}_{\text{BD}}[k][l], \hat{\boldsymbol{C}}^{\boldsymbol{m}_2}\rangle &= \sum_{i<n}\sum_{j<\ell} \boldsymbol{C}_{i,j}[k][l] \cdot \left(\boldsymbol{P}\overline{\boldsymbol{R}}_{i,j} + 2^j y^i \boldsymbol{M}_2\right) \\ &= \sum\sum \boldsymbol{P}\overline{\boldsymbol{R}}_{i,j} + \boldsymbol{P}\overline{\boldsymbol{R}}_1\boldsymbol{M}_2 + 2^k y^l \boldsymbol{M}_1\boldsymbol{M}_2.\end{aligned}$$

Here we compute an element of the matrix using same approach that we used for LHS-RHS multiplication. We take an element in the matrix at any location (k,l) and apply the bit decomposition of that element $\boldsymbol{C}^{\boldsymbol{m}_1}_{\text{BD}}[k][l]$. Later, we compute component-wise products, which gives us the ciphertext result at location (k,l) in the result matrix. One $\mathsf{RHS} \times \mathsf{RHS}$ multiplication requires $n\ell$ multiplications of $\mathsf{LHS} \times \mathsf{RHS}$ type. Also, multiplication does not require one-sided evaluation as in the One-Sided Homomorphic Multiplication method. Since we can create an RHS operand, we can evaluate an arbitrary circuit, which gives an advantage over One-Sided Homomorphic Multiplication. The noise growth in multiplications is still low, but it accumulates as we compute depth D multiplication using a binary tree multiplication. This leads to a worse noise growth compared to LHS-RHS multiplication. But just as in method 1, we have still eliminated the effect of $\boldsymbol{f}(x)$ on noise.

Lemma 4. *Let n be the polynomial degree, let $q = 2^{n^{\varepsilon}}$ be the modulus, let $\chi = D_{\mathbb{Z}^n,r}$ is the β-bounded Gaussian distribution, and let D be the multiplicative level. Then, the proposed Generic Homomorphic Multiplication algorithm has noise bound $(n\ell+1)^D\|\mathsf{ps}r\| = O((n\log q)^D)$ for fixed s and β.*

3.3.3 Leveled Homomorphic Public Key Scheme Instantiation We construct a leveled homomorphic scheme using the noise management technique described above and the SHFF-PKscheme. Here we list the primitive functions of the Leveled Homomorphic Public Key scheme:

– $\mathsf{LHFF\text{-}PK.Keygen}(1^{\kappa})$:
  - Compute $\mathsf{SHFF\text{-}PK.Keygen}(1^{\kappa})$.
– $\mathsf{LHFF\text{-}PK.Enc}(\mathcal{S}, \boldsymbol{m})$:
  - We form n by ℓ ciphertext matrix $\hat{\boldsymbol{C}}$ by computing its elements $\boldsymbol{C}(y)[i][j] = \mathsf{SHFF} - \mathsf{PK.Enc}(\mathcal{S}, 2^{\mathsf{i}}\boldsymbol{\psi}^{\mathsf{j}}\boldsymbol{m})$ for $i < \ell$ and $j < n$.
  - Output $\hat{\boldsymbol{C}}$ as the ciphertext.
– $\mathsf{LHFF\text{-}PK.Dec}(\boldsymbol{f}, \boldsymbol{\psi}, \hat{\boldsymbol{C}})$:
  - Compute $\mathsf{SHFF\text{-}PK.Dec}(\boldsymbol{f}, \boldsymbol{\psi}, \boldsymbol{C}[0][0])$.
– $\mathsf{LHFF\text{-}PK.Eval}(C, \hat{\boldsymbol{C}}_1, \hat{\boldsymbol{C}}_2, \ldots, \hat{\boldsymbol{C}}_\ell)$:
  - We follow a similar approach to that we used in SHFF-SK. We show that the homomorphic properties are preserved under the binary circuit evaluation with gates $\{+, \times\}$. This proves that any circuit C can be evaluated using two gates with two binary inputs.

Homomorphic Addition (+). Homomorphic addition of two ciphertext matrices $\hat{\boldsymbol{C}}_1$ and $\hat{\boldsymbol{C}}_2$ is evaluated by performing a matrix addition, $\hat{\boldsymbol{C}} = \hat{\boldsymbol{C}}_1 + \hat{\boldsymbol{C}}_2$. Namely, we compute the elements of the ciphertext matrix at each location (k, l) by computing $\boldsymbol{C}(y)[k][l] = \boldsymbol{C}_1(y)[k][l] + \boldsymbol{C}_2(y)[k][l] \pmod{\boldsymbol{F}(y)}$. The summation at each location preserves the ciphertext matrix property, $\boldsymbol{C}[k][l] = (\boldsymbol{P}\overline{\boldsymbol{R}}_1 + 2^k y^l \boldsymbol{M}_1) + (\boldsymbol{P}\overline{\boldsymbol{R}}_2 + 2^k y^l \boldsymbol{M}_2)$, which simplifies to $\boldsymbol{C}[k][l] = \boldsymbol{P}(\overline{\boldsymbol{R}}_1 + \overline{\boldsymbol{R}}_2) + 2^k y^l(\boldsymbol{M}_1 + \boldsymbol{M}_2)$. This shows that the ciphertext property of the matrix holds. Also, the first element $\boldsymbol{C}[0][0]$ is decryptable and gives us the result of the summation.

Homomorphic Multiplication (×). Homomorphic multiplication is evaluated using the multiplication method that is explained in Section 3.3.2. A matrix ciphertext multiplication preserves its format, which allows it to continue the homomorphic process. This may be sees by comparing the format of a fresh ciphertext and a product of ciphertexts. First we recall the format of an element of a fresh ciphertext: $\boldsymbol{C}^{\boldsymbol{m}_1}[k][l] = \boldsymbol{P}\overline{\boldsymbol{R}}_1 + 2^k y^l \boldsymbol{M}_1$. Next we recall the result of multiplication using multiplication method 2:

$$\boldsymbol{C}^{\boldsymbol{m}_1\cdot\boldsymbol{m}_2}[k][l] = \langle \hat{\boldsymbol{C}}^{\boldsymbol{m}_1}_{\mathrm{BD}}[k][l], \hat{\boldsymbol{C}}^{\boldsymbol{m}_2}\rangle = \sum\sum \boldsymbol{P}\overline{\boldsymbol{R}}_{i,j} + \boldsymbol{P}\overline{\boldsymbol{R}}_1\boldsymbol{M}_2 + 2^k y^l \boldsymbol{M}_1\boldsymbol{M}_2.$$

When we compare the ciphertext elements, it is clear that in a multiplication, we preserve the ciphertext matrix format while computing the multiplication, i.e., $2^k y^l \boldsymbol{M}_1 \boldsymbol{M}_2$. Also, in order to decrypt successfully, we need only decrypt the first element $\boldsymbol{C}[0][0]$ of the matrix .

Multiplicative Level *D*. We capture the multiplicative depth of the leveled homomorphic scheme as follows.

Lemma 5. *The encryption scheme*

$$\mathcal{E}_{\mathsf{LH}}\{\mathsf{LHFF-PK.KeyGen}, \mathsf{LHFF-PK.Enc}, \mathsf{LHFF-PK.Dec}, \mathsf{LHFF-PK.Eval}\}$$

described above is leveled homomorphic for circuits having depth $D = O(n^\varepsilon/\log n)$ where $q = 2^{n^\varepsilon}$ with $\varepsilon \in (0,1)$, and χ is a β-bounded Gaussian distribution for random sampling.

Proof. In order to determine an upper bound for depth D, we use the noise bound that is calculated in Section 3.3.2. The noise has a bound $(n \log q + 1)^D \|pr\|$, which is equal to $(n \log q + 1)^D (\mathsf{s}\beta')$. We require that this be smaller than $q/2$, which gives an upper bound for multiplicative level D in the form $(n \log q + 1)^D (\mathsf{s}\beta') < q/2$. Taking the logarithm of both sides gives $D \log(n \log q + 1) + \log(\mathsf{s}\beta') < \log q - 1$. Since $1 \ll n \log q$, using $q = 2^{n^\varepsilon}$ yields

$$D < \frac{n^\varepsilon - 1 - \log(\mathsf{s}\beta')}{\log n + \varepsilon \log n}.$$

In big-$\mathcal{O}$ notation, this gives an upper bound of the form $O(n^\varepsilon/\log n)$.□

3.3.4 Security The construction of the leveled homomorphic encryption is based on the Somewhat Homomorphic Finite Field Encryption scheme. Since there is not any significant change that affects the security, the leveled version of our construction is based on the same security assumptions as SHFF-PK: the hardness of the Decisional FFI and the subset sum problems.

Lemma 6. *Let n be the polynomial degree, let $q = 2^{n^\varepsilon}$ be the modulus, and let $\chi = D_{\mathbb{Z}^n, r}$ be a Gaussian distribution. Then, the proposed leveled homomorphic encryption scheme*

$$\mathcal{E}_{\mathsf{LH}}\{\mathsf{LHFF-PK.KeyGen}, \mathsf{LHFF-PK.Enc}, \mathsf{LHFF-PK.Dec}, \mathsf{LHFF-PK.Eval}\}$$

is secure under the assumptions of hardness of the Decisional Finite Field Isomorphism problem and the subset sum problem.

3.3.5 Bootstrapping In order to demonstrate that $\mathcal{E}$ is fully homomorphic, we show that the depth of the decryption circuit can be homomorphically achieved by our scheme. First we look at the depth of the decryption circuit.

Decryption Circuit Depth. We recall that decryption is given by evaluating $\boldsymbol{c}'(x) = \boldsymbol{C}(\boldsymbol{\psi}(x)) \pmod{\boldsymbol{p}(x), \boldsymbol{f}(x)}$. Denoting the coefficients of $\boldsymbol{C}(y)$ by

$\boldsymbol{\zeta}_i$, this can be expanded as $\boldsymbol{c}'(x) = \boldsymbol{\zeta}_0 + \boldsymbol{\zeta}_1\boldsymbol{\psi}(x) + \boldsymbol{\zeta}_2\boldsymbol{\psi}(x)^2 + \dots \boldsymbol{\zeta}_{n-1}\boldsymbol{\psi}(x)^{n-1}$ $(\bmod\ \boldsymbol{f}(x), \boldsymbol{p}(x))$. Modular reduction by $\boldsymbol{f}(x)$ can be avoided by pre-computing $\boldsymbol{\psi}'^{(i)}(x) = \boldsymbol{\psi}(x)^i \pmod{\boldsymbol{f}(x)}$. This turns decryption into summation of polynomials are multiplied by scalars, $\boldsymbol{c}'(x) = \sum_{i<n} \boldsymbol{\zeta}_i \boldsymbol{\psi}'^{(i)}(x)$. Let $\boldsymbol{c}'_j$ be the coefficients of the result $\boldsymbol{c}'(x)$. Then each coefficient is evaluated by computing $\boldsymbol{c}'_j = \sum_{i<n} \boldsymbol{\zeta}_i \boldsymbol{\psi}'^{(i)}_j$ where $\boldsymbol{\psi}'^{(i)}_j$ denotes the j^{th} coefficient of $\boldsymbol{\psi}'^{(i)}$.

In [6, Lemma 4.5] the authors prove that evaluating the sum of n elements with $\log q$ bits results in circuit depth $O(\log n + \log\log q)$. They also show that they can do modular reduction mod q with circuit depth $O(\log n + \log\log q)$. Since $\boldsymbol{p}(x)$ is small, say $\boldsymbol{p}(x) = 2$, we can perform modular reduction mod $\boldsymbol{p}$ by taking the first bit, which does not require any circuit. Therefore, the bootstrapping operation has an upper bound $O(\log n + \log\log q)$.

Theorem 3. *Let χ is a β-bounded distribution for β = poly(n), and let $q = 2^{n^{\varepsilon}}$ for $0 < \varepsilon < 1$. Then there exists a fully homomorphic encryption scheme based on the the leveled homomorphic encryption scheme $\mathcal{E}$ =* LHFF-PK *with the assumptions that scheme is secure under the Decisional Finite Field Isomorphism Problem and that it is weakly circular secure.*

Proof. The decryption circuit requires $O(\log n + \log\log q)$ depth, and our scheme can compute $O(n^{\varepsilon}/\log n)$ depth circuits (Lemma 5). Therefore, the following inequality is sufficient in order to be bootstrappable:

$$\Upsilon(\log n + \log\log q) < n^{\varepsilon}/\log n$$

where $\Upsilon > 0$ is used to capture the constants in the circuit. Since $0 < \varepsilon < 1$, in worst case scenario we obtain $2\Upsilon < \log q/\log^2 n$. □

4 Conclusion

In this work we proposed a new conjectured hard problem: the finite field isomorphism problem. Informally, the FFI problem asks one to construct an explicit isomorphism between two representations of a finite field, given only access to long (large norm) representations of field elements and the assurance of the existence of a representation where each of these elements has a short (low norm) expression. We formalized the FFI problem and study the effectiveness of various approaches, including lattice attacks and non-lattice algebraic techniques, for recovering the secret isomorphism. Relying on the assumed hardness of the decisional-FFI problem, we first presented a secret-key somewhat homomorphic encryption scheme. This was extended, using a subset-sum problem technique, to a public-key scheme. We briefly analyze the noise performance of both schemes and introduced a bit-decomposition-based noise managements scheme that allows us to reduce the noise growth to single exponential. This yielded a bootstrapable, and thus a fully homomorphic encryption scheme.

References

1. Ajtai, M.: The shortest vector problem in l_2 is NP-hard for randomized reductions (extended abstract). In: Thirtieth Annual ACM Symposium on the Theory of Computing (STOC 1998). pp. 10–19 (1998)
2. Alkim, E., Ducas, L., Pöppelmann, T., Schwabe, P.: Post-quantum key exchange - A new hope. In: 25th USENIX Security Symposium, USENIX Security 16, Austin, TX, USA, August 10-12, 2016. pp. 327–343 (2016), https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/alkim
3. Bos, J.W., Lauter, K., Loftus, J., Naehrig, M.: Cryptography and Coding: 14th IMA International Conference, 2013. Proceedings, chap. Improved Security for a Ring-Based Fully Homomorphic Encryption Scheme, pp. 45–64. Springer Berlin Heidelberg, Berlin, Heidelberg (2013), http://dx.doi.org/10.1007/978-3-642-45239-0_4
4. Brakerski, Z.: Fully Homomorphic Encryption without Modulus Switching from Classical GapSVP, pp. 868–886. Springer Berlin Heidelberg, Berlin, Heidelberg (2012), http://dx.doi.org/10.1007/978-3-642-32009-5_50
5. Brakerski, Z., Gentry, C., Vaikuntanathan, V.: Fully homomorphic encryption without bootstrapping. Electronic Colloquium on Computational Complexity (ECCC) 18, 111 (2011)
6. Brakerski, Z., Vaikuntanathan, V.: Efficient fully homomorphic encryption from (standard) LWE. In: FOCS. pp. 97–106 (2011)
7. Chen, Y., Nguyen, P.Q.: BKZ 2.0: Better lattice security estimates. In: ASIACRYPT. pp. 1–20 (2011)
8. Cheon, J.H., Jeong, J., Lee, C.: An algorithm for ntru problems and cryptanalysis of the ggh multilinear map without a low level encoding of zero. Cryptology ePrint Archive, Report 2016/139 (2016), https://eprint.iacr.org/2016/139
9. Coron, J.S., Lepoint, T., Tibouchi, M.: Scale-Invariant Fully Homomorphic Encryption over the Integers, pp. 311–328. Springer Berlin Heidelberg, Berlin, Heidelberg (2014), http://dx.doi.org/10.1007/978-3-642-54631-0_18
10. Coron, J.S., Mandal, A., Naccache, D., Tibouchi, M.: Fully homomorphic encryption over the integers with shorter public keys. In: CRYPTO. pp. 487–504 (2011)
11. Coron, J.S., Naccache, D., Tibouchi, M.: Public key compression and modulus switching for fully homomorphic encryption over the integers. In: EUROCRYPT. pp. 446–464 (2012)
12. van Dijk, M., Gentry, C., Halevi, S., Vaikuntanathan, V.: Fully homomorphic encryption over the integers. In: EUROCRYPT. pp. 24–43 (2010)
13. Doröz, Y., Hu, Y., Sunar, B.: Homomorphic aes evaluation using the modified ltv scheme. Designs, Codes and Cryptography pp. 1–26 (2015), http://dx.doi.org/10.1007/s10623-015-0095-1
14. Doröz, Y., Sunar, B.: Flattening NTRU for evaluation key free homomorphic encryption. Cryptology ePrint Archive, Report 2016/315 (2016), http://eprint.iacr.org/2016/315
15. Ducas, L., Durmus, A., Lepoint, T., Lyubashevsky, V.: Lattice signatures and bimodal gaussians. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, LNCS, vol. 8042, pp. 40–56. Springer (2013)
16. Gama, N., Nguyen, P.Q.: Predicting lattice reduction. In: Proceedings of the theory and applications of cryptographic techniques 27th annual international conference on Advances in cryptology. pp. 31–51. EUROCRYPT'08, Springer-Verlag, Berlin, Heidelberg (2008), http://dl.acm.org/citation.cfm?id=1788414.1788417

17. Gentry, C.: A fully homomorphic encryption scheme. Ph.D. thesis, Stanford University (2009), crypto.stanford.edu/craig
18. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing. pp. 169–178. STOC '09, ACM, New York, NY, USA (2009)
19. Gentry, C., Halevi, S.: Implementing Gentry's fully-homomorphic encryption scheme. In: EUROCRYPT. pp. 129–148 (2011)
20. Gentry, C., Halevi, S., Smart, N.P.: Homomorphic Evaluation of the AES Circuit, pp. 850–867. Springer Berlin Heidelberg, Berlin, Heidelberg (2012), http://dx.doi.org/10.1007/978-3-642-32009-5_49
21. Gentry, C., Sahai, A., Waters, B.: Advances in Cryptology – CRYPTO 2013: 33rd Annual Cryptology Conference, 2013. Proceedings, Part I, chap. Homomorphic Encryption from Learning with Errors: Conceptually-Simpler, Asymptotically-Faster, Attribute-Based, pp. 75–92. Springer Berlin Heidelberg, Berlin, Heidelberg (2013)
22. Gentry, C., Szydlo, M.: Cryptanalysis of the revised NTRU signature scheme. In: Advances in Cryptology - EUROCRYPT 2002, International Conference on the Theory and Applications of Cryptographic Techniques, Amsterdam, The Netherlands, April 28 - May 2, 2002, Proceedings. pp. 299–320 (2002), https://doi.org/10.1007/3-540-46035-7_20
23. Halevi, S., Shoup, V.: HElib, homomorphic encryption library (2012)
24. Hoffstein, J., Pipher, J., Schanck, J.M., Silverman, J.H., Whyte, W., Zhang, Z.: Choosing parameters for ntruencrypt. In: Topics in Cryptology - CT-RSA 2017 - The Cryptographers' Track at the RSA Conference, 2017, Proceedings. pp. 3–18 (2017), http://dx.doi.org/10.1007/978-3-319-52153-4_1
25. Hoffstein, J., Pipher, J., Silverman, J.H.: NTRU: A ring-based public key cryptosystem. In: ANTS. pp. 267–288 (1998)
26. Ireland, K., Rosen, M.: A Classical Introduction to Modern Number Theory. Springer-Verlag (1990)
27. Kirchner, P., Fouque, P.: Revisiting lattice attacks on overstretched NTRU parameters. In: Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part I. pp. 3–26 (2017), https://doi.org/10.1007/978-3-319-56620-7_1
28. López-Alt, A., Tromer, E., Vaikuntanathan, V.: On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In: Proceedings of the Forty-fourth Annual ACM Symposium on Theory of Computing. pp. 1219–1234. STOC '12, ACM (2012), http://doi.acm.org/10.1145/2213977.2214086
29. Martin Albrecht, Shi Bai, L.D.: A subfield lattice attack on overstretched ntru assumptions: Cryptanalysis of some fhe and graded encoding schemes. Cryptology ePrint Archive, Report 2016/127 (2016), http://eprint.iacr.org/
30. May, A., Silverman, J.H.: Dimension reduction methods for convolution modular lattices. In: Cryptography and Lattices, International Conference, CaLC 2001, Providence, RI, USA, March 29-30, 2001, Revised Papers. pp. 110–125 (2001), https://doi.org/10.1007/3-540-44670-2_10
31. Smart, N.P., Vercauteren, F.: Fully homomorphic simd operations. Designs, Codes and Cryptography 71(1), 57–81 (2014), http://dx.doi.org/10.1007/s10623-012-9720-4
32. Stehlé, D., Steinfeld, R.: Making NTRU as secure as worst-case problems over ideal lattices. Advances in Cryptology – EUROCRYPT '11 pp. 27–4 (2011)
33. The PARI Group, Bordeaux: PARI/GP version 2.7.0 (2014), available from http://pari.math.u-bordeaux.fr/

A Constructing the Inverse Isomorphism

The map defined by $x \mapsto \boldsymbol{\phi}(y)$ is a field isomorphism. It follows that there is an inverse isomorphism, and that inverse isomorphism is determined by the image of y. So we write the inverse isomorphism as

$$y \longmapsto \boldsymbol{\psi}(x) = \sum_{i=0}^{n-1} c_i x^i, \tag{6}$$

and our goal is to determine the c_i coefficients. We know that the composition $y \longmapsto \boldsymbol{\psi}(x) \longmapsto \boldsymbol{\psi}\big(\boldsymbol{\phi}(y)\big)$ gives an automorphism of $\mathbb{F}_q[y]/(\boldsymbol{F}(y))$, so

$$\boldsymbol{\psi}\big(\boldsymbol{\phi}(y)\big) \equiv y \pmod{\boldsymbol{F}(y)}. \tag{7}$$

Hence it suffices to determine the (unique) polynomial $\boldsymbol{\psi}(x)$ of degree less than n satisfying (7). Using the expression (6) for $\boldsymbol{\psi}(x)$, we want to find c_i so that $\sum_{i=0}^{n-1} c_i \boldsymbol{\phi}(y)^i \equiv y \pmod{\boldsymbol{F}(y)}$. We write each power $\boldsymbol{\phi}(y)^i$ modulo $\boldsymbol{F}(y)$ as a polyomial of degree less than n. In other words, we use the known values of $\boldsymbol{\phi}(y)$ and $\boldsymbol{F}(y)$ to write $\boldsymbol{\phi}(y)^i = \sum_{j=0}^{n-1} a_{ij} y^j \pmod{\boldsymbol{F}(y)}$ for $0 \le i < n$. Substituting this into $\boldsymbol{\psi}\big(\boldsymbol{\phi}(y)\big)$ yields $\boldsymbol{\psi}\big(\boldsymbol{\phi}(y)\big) = \sum_{i=0}^{n-1} c_i \boldsymbol{\phi}(y)^i \equiv \sum_{i=0}^{n-1} c_i \sum_{j=0}^{n-1} a_{ij} y^j \pmod{\boldsymbol{F}(y)} \equiv \sum_{j=0}^{n-1} \left(\sum_{i=0}^{n-1} a_{ij} c_i\right) y^j \pmod{\boldsymbol{F}(y)}$. Hence $\boldsymbol{\psi}$ will satisfy (7) if we choose $c_0, \ldots, c_{n-1}$ to satisfy $\sum_{i=0}^{n-1} a_{ij} c_i = 1$ if $j = 1$, and $\sum_{i=0}^{n-1} a_{ij} c_i = 0$ if $j = 0$. This is a system of n equations for the n variables $c_0, \ldots, c_{n-1}$ over the finite field $\mathbb{F}_q$, hence is easy to solve, which gives the desired polynomial $\boldsymbol{\psi}(y)$.

B Noise Analysis

To estimate the noise, we need to find the effect of modular reduction reduction operation (with $\boldsymbol{f}(x)$) on the norm. One way is to use Barrett's Reduction algorithm. In Barrett's algorithm, a precomputed factor $\boldsymbol{M}(x) = x^{2n}/\boldsymbol{f}(x)$ plays a key role in estimating the quotient of the division with the modulus. Therefore, determining $\boldsymbol{M}(x)$ will give us the main contributing factor to the noise level. Our goal is to bound the norm of the factor $\boldsymbol{M}(x)$ as tightly as possible. We start by rearranging $\boldsymbol{M}(x) = \lfloor x^{2n}/\boldsymbol{f}(x) \rfloor = \left\lfloor \frac{x^n}{1+\frac{\boldsymbol{f}'(x)}{x^n}} \right\rfloor$ Note that $\deg(\boldsymbol{f}'(x)) < n$ and the floor operator simply truncates the polynomial beyond the constant term. This allows us to write the Taylor Series expansion (polynomial equivalent for $1/(1+x)$) as follows $\boldsymbol{M}(x) = \left\lfloor x^n + \sum_{i=1}^{i=\ell} (-1)^i \frac{\boldsymbol{f}'(x)^i}{x^{(i-1)n}} \right\rfloor$ Set $d = \deg(\boldsymbol{f}'(x))$. Then, each element in the series contributes up to a polynomial degree in the summation. It is important to notice that since $n > d$ each term in the expansion of $\boldsymbol{M}(x)$ the degree is bounded by d (except of course the x^n term. Therefore $\deg(\boldsymbol{M}(x) - x^n) \le d$. In the series expansion a power $\boldsymbol{f}'(x)^i$ contributes to the series as long as $(i-1)n \le id$. For larger i values the new additive term is simply truncated away, i.e. has no effect on $M(x)$. Therefore in the summation we only need to consider up to a degree ℓ which is determined as follows $\ell = \lfloor n/(n-d) \rfloor$. In the special case of $d < n/2$ we have $\ell = 1$ and $\boldsymbol{M}(x) = 1 - f'(x)$ and $\beta_M = \beta_f$. In the general case, to bound the norm of $\boldsymbol{M}(x)$, we have to find the largest possible value for each term in the expansion. Assume that we sample $f'(x)$ from a β-bounded distribution. We first assume $\beta = 1$ and later generalize the worst and average case bounds to cover arbitrary β values.

B.1 Worst Case Analysis

For clarity we first consider the first few terms in the expansion and then generalize the contribution to an arbitrary term:

f'(x): Since this is a fresh polynomial, the coefficients are sampled from a β-bounded distribution. For $\beta = 1$ in the worst case all coefficients are set to 1, i.e. $\boldsymbol{f}'(x) = x^d + x^{d-1} + x^{d-2} + \cdots + x^1 + 1$.

$\mathbf{f'(x)^2/x^n}$: Assume we compute the square of $\boldsymbol{f}'(x)$ using as schoolbook multiplication. It is easy to see that starting from the middle degree d, the coefficients of the result decrease as we go to lower and higher degrees. In other words, the coefficients of $\boldsymbol{f}'(x)^2$ are symmetric around the middle degree. Since $\beta = 1$, we can write the polynomial as $x^{2d} + 2x^{2d-1} + \cdots + (d+1)x^d + \cdots + 2x + 1$. The division by x^n eliminates the first n terms. This results in following polynomial $x^{2d-n} + 2x^{2d-n-1} + 3x^{2d-n-2} + \cdots + (2d-n+1)x^0$. Since $d < n$ then $2d - n < d$ and thus the largest coefficient is the constant coefficient with value $(2d-n+1)$.

$\mathbf{f'(x)^i/x^{(i-1)n}}$: We are now ready to generalize the approach to find the largest coefficient for a degree i. When computing $\boldsymbol{f}'(x)^i = \boldsymbol{f}'(x)^{i-1} \cdot \boldsymbol{f}'(x)$ since it is divided by $x^{(i-1)n}$, we only use the last $id - (i-1)n + 1$ coefficients of $\boldsymbol{f}'(x)^{i-1}$. We multiply $\boldsymbol{f}'(x)^{i-1}$ with each coefficient of $\boldsymbol{f}'(x)$ and only take the last $id - (i-1)n + 1$ coefficients. If $\beta_{i-1} = \max(\boldsymbol{f}'(x)^{i-1})$, then we add $id - (i-1)n + 1$ of $B_{i-1} \cdot \beta$ which makes the upper bound $(id - (i-1)n + 1) \cdot \beta_{i-1} \cdot \beta$. If we apply this recursively to compute for previous values of i, we achieve an upper bound $(id - (i-1)n + 1)^{i-1}$ for $\beta = 1$.

B.2 Worst Case for Arbitrary β

$\mathbf{f'(x)^i/x^{(i-1)n}}$. We use the general formula as explained in the section above. For the current i we have $(id - (i-1)n + 1) \cdot \beta_{i-1} \cdot \beta$ as the upper bound. For any β, recursively we have β^i so the upper bound will be $(id - (i-1)n + 1)^{i-1} \cdot \beta^i$. The overall bound on $\boldsymbol{M}(x)$ is therefore $B_M = ||\boldsymbol{M}(x)|| \leq \sum_{i=1,..,\ell}(id - (i-1)n + 1)^{i-1}\beta^i$ where $B_0 = \beta$ and as established before $\ell = \lfloor n/(n-d) \rfloor$. Our goal is to bound the norm $||\boldsymbol{a}(x)\boldsymbol{b}(x) \bmod f(x)||$ using Barrett Reduction. We assume both $||\boldsymbol{a}(x)||, ||\boldsymbol{b}(x)|| \leq \beta$ and $\deg(f(x)) = n$. We compute the worst case noise bound using the following steps:

- Step 1. Compute $\boldsymbol{M}(x) = \lfloor x^{2n}/\boldsymbol{f}(x) \rfloor$ ($\boldsymbol{M}(x)$ is the quotient of the division). Also assume $||\boldsymbol{M}(x)|| = \beta_M$.

- Step 2. Compute regular product $\boldsymbol{c}(x) = \boldsymbol{a}(x)\boldsymbol{b}(x)$. $||\boldsymbol{c}(x)|| = n\beta^2$.
- Step 3. Estimate quotient of $\boldsymbol{c}(x)/\boldsymbol{f}(x)$ (dropping (x) for brevity) $\boldsymbol{q}_1 = \lfloor \boldsymbol{c}/x^n \rfloor$. Since we take half of $\boldsymbol{c}$, worst case noise still remains: $||\boldsymbol{q}_1|| = n\beta^2$.
  $q_2 = \boldsymbol{M}\boldsymbol{q}_1$. This yields $||\boldsymbol{q}_2|| = (d+1) \cdot \beta_M \cdot n\beta^2 = n(d+1)\beta_M\beta^2$
  $q_3 = \lfloor \boldsymbol{q}_2/x^n \rfloor$. Worst case noise remains the same as $\boldsymbol{q}_2$: $||\boldsymbol{q}_3|| = n(d+1)\beta_M\beta^2$
- Step 4. Fix the result using the lower half of $\boldsymbol{c}(x)$
  $\boldsymbol{r}_1 = c \bmod x^n$, thus $||\boldsymbol{r}_1|| = n\beta^2$,
  $\boldsymbol{r}_2 = \boldsymbol{q}_3 f \bmod x^n$ $||\boldsymbol{r}_2|| = n \cdot (d+1)^2 \beta_M \beta^2 \cdot \beta_f$, where we choose $||\boldsymbol{f}(x)|| = \beta_f$.
  $\boldsymbol{r} = \boldsymbol{r}_1 - \boldsymbol{r}_2 = \boldsymbol{a}(x)\boldsymbol{b}(x) \bmod \boldsymbol{f}(x)$. This gives us an overall bound of $||\boldsymbol{a}(x)\boldsymbol{b}(x) \bmod \boldsymbol{f}(x)|| \leq n\beta^2 + n(d+1)^2\beta^2\beta_M\beta_f$

For $d < n/2$ and $\beta_f = 1$, we have $\beta_M = 1$ and the worst case norm simplifies to $||\boldsymbol{a}(x)\boldsymbol{b}(x) \bmod \boldsymbol{f}(x)|| \leq n[(d+1)^2+1]\beta^2$. In the average case the noise norm can be approximated by $||\boldsymbol{a}(x)\boldsymbol{b}(x) \bmod \boldsymbol{f}(x)||_{avg} \approx n^{1/2}\beta^2 + n^{1/2}(d+1)\beta^2\beta_M\beta_f$.

C Sample parameters and their security estimates

In Table 1 we present some parameters for the somewhat homomorphic encryption scheme. The proposed parameter set does not take into account our noise management technique. We compute the levels (circuit depth) by doing straightforward multiplications. In all 5 examples, we choose $\beta = 2$ and $d = n/2$ (recall that d is the degree of $\boldsymbol{f}'(x)$ where $\boldsymbol{f}(x) = x^n + \boldsymbol{f}'(x)$). For each level we give a noise estimate and also give a maximum selectable q size.

Level	n	$\log noise$	$\log max(q)$	Ciphertext Size	root of Ratio	BKZ 2.0 cost
1	256	13	15	0.4 KB	1.0060	$> 2^{145}$
2	2048	50	83	12.5 KB	1.0065	$> 2^{135}$
3	4096	127	161	63.5 KB	1.0066	$> 2^{136}$
4	8192	293	317	293 KB	1.0066	$> 2^{137}$
5	32768	698	1250	2.7 MB	1.0066	$> 2^{139}$

Table 1: Sample parameters for somewhat homomorphic encryption

Target Root Hermite Factor	1.01	1.009	1.008	1.007	1.006
Approximate Block Size	85	106	133	168	216

Table 2: Requried Blocksize for target root Hermite factor [7]

block size b	100	110	120	130	140	150	160	170	180	190	200	210	220	230	240	250
LogNodes(b)	39	44	49	54	60	66	72	78	84	96	99	105	111	120	127	134

Table 3: Upper bounds on $\log_2$ number of nodes enumerated in one call to enumeration subroutine of BKZ 2.0 [7].

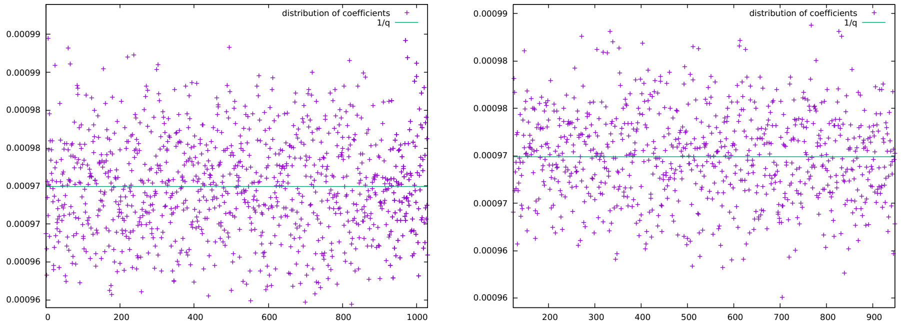
To estimate the cost of BKZ 2.0, we follow the cryptanalysis in [2, 24]. We use Table 2 and 3 to estimate the block size and the number of nodes for a given root Hermite factor. Then we use the following formula ([24], which is an interpolation of data reported in [7] to get the cost of BKZ 2.0).

$$\text{BKZCost}(dim, b, rounds) = \text{LogNodes}(b) + \log_2(dimension \cdot rounds) + 7.$$

We remark that in [2] the authors proposed to use (quantum) sieving, rather than enumeration with extreme pruning, to estimate the cost of BKZ 2.0. In this analysis, we stick to the original estimation model, to show a proof-of-concept that practical parameters can be derived for our scheme. We leave the parameter derivation under the more conservative model in [2] to future work.

D Testing Results for Observation 2

We test the soundness of Observation 2 as follows. We setup toy size isomorphisms with $n \in \{20, 30, 40, 80\}$ and $q \in \{1031, 2053, 2^{20} + 7\}$. For each test we generate a long transcript of elements in $\mathbb{X}$ and $\mathbb{Y}$; We examine the distribution of the coefficients in $\mathbb{Y}$ and compare it with uniform distribution; We show that the Renyi divergence between our distribution and a uniform distribution scales properly with $\log_2(q/n)$. Two example distribution of the coefficients are shown in Figure 1. We compute the Renyi divergence with $\alpha = 2$. Our results shows that our distribution is less than 2^{-14} away from a uniform distribution for out toy example with $n = 20$ and $q = 1031$.



(a) n=20, q = 1031

(b) n=40, q = 1031

Fig. 1: Testing results for Observation 2

q	$n = 20$	$n = 30$	$n = 40$	$n = 80$
1031	$2^{-14.3}$	$2^{-14.8}$	$2^{-15.3}$	$2^{-16.2}$
2053	$2^{-13.3}$	$2^{-13.9}$	$2^{-14.3}$	$2^{-15.3}$
2^{20}+7	$2^{-4.3}$	$2^{-4.8}$	$2^{-5.3}$	$2^{-6.2}$

Table 4: Renyi Divergence

We summarize the testing result in Table 4. As one can see the exponent of the divergence is linear in $\log_2(q/n)$. We estimate that for moderate $n \approx q$ the divergence is around 2^{-11}.