On the IND-CCA1 Security of FHE Schemes

Prastudy Fauzi[1], Martha Norberg Hovd[1,2], and Håvard Raddum[1]

[1] Simula UiB, Bergen, Norway
{prastudy,martha,haavardr}@simula.no
[2] Department of Informatics, University of Bergen, Norway

Abstract. Fully homomorphic encryption (FHE) is a powerful tool in cryptography that allows one to perform arbitrary computations on encrypted material without having to decrypt it first. There are numerous FHE schemes, all of which are expanded from somewhat homomorphic encryption (SHE) schemes, and some of which are considered viable in practice. However, while these FHE schemes are semantically (IND-CPA) secure, the question of their IND-CCA1 security is much less studied.
In this paper, we group SHE schemes into broad categories based on their similarities and underlying hardness problems. For each category, we show that the SHE schemes are susceptible to either known adaptive key recovery attacks, a natural extension of known attacks, or our proposed attacks. Finally, we discuss the known techniques to achieve IND-CCA1 secure FHE and SHE schemes.

Keywords: FHE schemes, IND-CCA, cryptanalysis

1 Introduction

Fully homomorphic encryption (FHE) allows meaningful computation to be performed on encrypted material without decrypting it. Slightly more formally: for any circuit C, we require that $\mathrm{Dec}(C(c)) = C(\mathrm{Dec}(c))$ for any ciphertext c. The notion was initially suggested in the late seventies by Rivest et al. [69], and Gentry proposed the first successful FHE scheme in 2009 [46].

Ever since Gentry's breakthrough, there has been a lot of development in FHE, especially with regards to practicality. Although Gentry's initial scheme was later implemented [47], it was little more than a proof of concept and far from being practical. Today the situation is very different, with several FHE libraries and examples of practical applications to real-world problems. For instance, recent field trials in the financial sector [63] include applying homomorphic encryption on a machine learning pipeline, making it possible to outsource work and analyses previously limited to in-house treatment. Furthermore, an international ISO/IEC standard is due by 2024 [56], and NIST is tracking the progress of FHE as a privacy-enhancing technology [1].

As FHE moves further into the realm of practicality and real-world application, it is crucial to study the security of these schemes in detail. The security notion most FHE schemes have aimed for, and several provably achieved based

on hardness assumptions, is *indistinguishability under chosen plaintext attacks* (IND-CPA). The security notion captures whether an adversary with access to the public key is able to distinguish between encryptions of messages, i.e., whether a ciphertext leaks information about its content.

However, IND-CPA offers no security guarantees if an adversary can obtain *decryptions* of ciphertexts. Such a scenario is hardly far-fetched, e.g., if a malicious cloud does not return the ciphertext arising from an honest computation, but rather an adulterine ciphertext, the cloud may observe if the client rejects the ciphertext as invalid by requesting a recomputation. Zhang et al. show how such an attack may be used to construct a probabilistic decryption oracle [81]. Chillotti et al. [32] demonstrate that the very set-up of FHE schemes makes them susceptible to reaction and safe error attacks. Similarly, a padding oracle attack [62,77] enables one to decrypt arbitrary ciphertexts by making use of a server that only tells whether or not the padding of an encrypted message is valid. Furthermore, it is difficult to rule out any adversarial access to decrypted material in the settings with rich data flows where FHE is being, or suggested to be, used; for example, in scenarios with multiple collaborating parties exchanging data or in combination with machine learning [26,67]. Finally, Cheon et al. [28] point out that some applications demand that decrypted values are shared with each involved party, not just those in possession the secret key, and give multi-party computation and differential privacy as examples of such applications. In short, IND-CPA is not sufficient to guarantee security in some of the scenarios where FHE is being used.

Beyond IND-CPA, there are additional levels of indistinguishability: IND-CCA1, where the adversary gets a decryption oracle for a limited amount of time, and IND-CCA2, where the adversary gets a decryption oracle that can be used at any time. It is well known that if an encryption scheme has *any* homomorphic property, it cannot achieve IND-CCA2 security. Finding a fully homomorphic encryption scheme that achieves IND-CCA1 security is still an open problem.[3] Some generic constructions of FHE schemes achieving IND-CCA1 security have been suggested, but none of them have been instantiated. Furthermore, although these constructions provably achieve IND-CCA1 security, they do suffer from various shortcomings, as will be discussed in Section 6.

All acknowledged FHE schemes are constructed in the same way. The starting point is a somewhat or leveled homomorphic encryption (SHE or LHE) scheme, with noise-based encryption, able to perform a limited number of homomorphic operations on ciphertexts before correct decryption is not guaranteed. The 'base scheme' is then expanded into an FHE scheme by homomorphically evaluating the decryption circuit on a ciphertext, which reduces the amount of noise in the ciphertext. This procedure is called *bootstrapping* and requires publishing some encrypted secret key material.

It is worth stressing that bootstrapping *excludes* IND-CCA1 security. After all, if an adversary may decrypt any ciphertext she chooses, she can simply choose to decrypt the published secret key material, which then allows her to

[3] In contrast, group homomorphic schemes can be IND-CCA1 secure, e.g., CS-lite [39].

decrypt other ciphertexts. Furthermore, several LHE and SHE schemes rely on publishing encrypted versions of various secret keys for other purposes than bootstrapping, such as key switching (e.g., BGV [17]) or relinearisation (e.g., BV11a and BV11b [19,20]), which makes homomorphic evaluations composable. Note that publishing such encryptions also prevents IND-CCA1 security. Hence, it seems that the best we can hope for is an IND-CCA1 secure SHE or LHE scheme that does not rely on key switching, relinearisation, or any other operation which requires publishing encryptions of secret keys. We stress that there is no result stating that homomorphic encryption schemes *cannot* achieve IND-CCA1 security. However, the presently known methods of constructing these schemes prevent such a level of security from being obtained, or introduce some shortcoming.

Almost no published SHE or LHE schemes aim at achieving IND-CCA1 security, though, but settle for IND-CPA. Furthermore, the schemes that have attempted to achieve IND-CCA1 security have, as we shall see, since been proven insecure. Several SHE and LHE schemes have been shown to be susceptible to adaptive key recovery attacks: an adversary is able to recover the secret key if she has access to a decryption oracle. However, since security against adversaries able to decrypt has not been strived for, vulnerable schemes have been developed and improved further in different ways, but not necessarily with any effort to make them IND-CCA1 secure, or resilient against adaptive key recovery attacks. Dahab, Galbraith and Morais [40] make this point as well: "There are adaptive key recovery attacks on many schemes and these schemes were adapted and optimised later; thus such constructions should be assessed in order to verify whether the attacks are still feasible".

It is worth emphasizing that these adaptive key recovery attacks are not just a theoretical threat: as previously mentioned, it is possible to construct a (probabilistic) decryption oracle using reaction attacks [81], and there are use cases where decrypted material being made available to adversaries is either probable or inevitable. The fact that it is possible to mount attacks that recover the secret key also on LHE and SHE schemes, i.e., schemes that do *not* employ bootstrapping, makes assessing the security of these schemes especially pressing. These schemes are a natural alternative to bootstrapped FHE schemes if the users wish to avoid the risk of key recovery through the published bootstrapping key, and it is therefore important to assess whether there is still a risk of the secret key being recovered also for LHE and SHE schemes.

Our contributions. In this paper, we provide precisely the assessment requested by Dahab et al.: we review all acknowledged SHE and LHE schemes with respect to adaptive key recovery attacks and show that *all* schemes are susceptible to such an attack. We determine if the schemes are vulnerable to a previously published attack and provide novel attacks for those not susceptible to already known attacks. Seeing as schemes strongly inspired by earlier schemes are likely to be susceptible to the same attack, we also provide a table of scheme genealogy: an overview of which schemes have been directly based on which.

Attack	Affected schemes	Extends to
Chenal and Tang ((R)LWE) [26]	Bra12 [15], BGV [17], BV11a [19], BV11b [20], GSW [48]	CKKS [29], FV [43], BCIV [13], AN [5], CLPX [25], CIL [24], BV14 [21], BL [9], CCS [23], CM [33], CGGI [31], Jou [52], BP [18], AH [4], PS [66], CS17 [37]
Loftus et al.* (Ideal Lattice) [60]	Gen [46], GH* [47], SV*[72]	SS* [74], SV14* [73]
Zhang et al. (AGCD) [80] and Chenal and Tang (AGCD) [26]	CMNT [35], vDGHV [76]	CNT [36], CS15 [30] CLT [34],CCKLLTY [27], KLYC [54]
Dahab et al. (NTRU) [40]	BLLN [14], LATV* [61]	RC* [71]
Fauzi et al. (other) [44]	LGM [59]	
Section 5.2 (AGCD)	Per [68], BBL [8]	
Section 5.3 (other)	DHPSSWZ [41], AFFHP [3]	

Table 1. IND-CCA1 attacks and affected schemes. The second column lists schemes mentioned by the attack paper, while the third column lists schemes we found to be affected as well. An asterisk (*) denotes schemes that have been shown to not be IND-CPA secure. Note that the paper by Loftus et al. presents both an attack and a scheme, where the asterisk denotes that the scheme is not IND-CPA secure, and the attack breaks the schemes listed in the table. The two final rows are novel attacks.

Additionally, we provide an overview of which schemes are no longer deemed IND-CPA secure. A summary of our findings with regards to security is listed in Table 1. We also discuss the prospect of noiseless schemes and the existing generic constructions for IND-CCA1 secure homomorphic encryption schemes. However, we do not discuss implemented variants of schemes, e.g., if an implementation improves the bootstrapping procedure of a presented scheme, as it does not affect the achieved security of the basis scheme.

We group schemes into five broad categories primarily based on the underlying hardness problem: (R)LWE, ideal lattices, AGCD, NTRU, and 'other'. Note that the schemes in the category 'ideal lattices' are based on several hardness assumptions, which is why we prefer referring to this category by the structure the schemes are based on. Furthermore, all the schemes based on AGCD are defined over the integers, and we also refer to these schemes as 'integer-based schemes'. The motivation for this categorisation is that it essentially corresponds to groups of schemes susceptible to the same attack. For example, a large majority of the schemes based on AGCD is broken by an attack by Chenal and Tang [26]. The schemes in the fifth 'other' category are schemes that are not based on either of the already mentioned hardness assumptions. For each category, we sketch the existing attacks, and show that these attacks can be extended to several other SHE schemes in the same category by relatively easy modifications. As

mentioned previously, we also provide novel attacks for the schemes not affected by any of the already known attacks or their extensions.

Related work. The security notions mentioned so far all assume the decryption procedure returns an *exact* message: if m has been encrypted, an unaltered ciphertext will decrypt to exactly m. However, this is not the case for all homomorphic encryption schemes, most notably CKKS [29], where decryption returns an approximation of the message. This discrepancy was noted by Li and Micciancio [57], who also proposed a novel security notion for approximate encryption schemes, termed IND-CPA^D, which reduces to IND-CPA for exact schemes, and showed that CKKS is not IND-CPA^D secure. They also presented a key recovery attack that does *not* rely on a decryption oracle, demonstrating the practical implication of the discovered insecurity. Cheon et al. has published a report addressing the insecurity, discussed which applications the attack is applicable to, and introduced extensions to libraries to prevent the key recovery attack [28].

Loftus et al. [60] discuss attacks on homomorphic encryption schemes in the verification setting, IND-CVA, where the adversary has access to a verfication oracle, rather than a decryption oracle, which determines if the decryption algorithm will output $\perp$ when given a specific ciphertext c. Loftus et al. demonstrate that an IND-CCA1 secure scheme may be IND-CVA2 insecure, if the adversary has access to the verification oracle after she has received the challenge ciphertext [60]. The vast majority of homomorphic encryption schemes consider all elements in the ciphertext domain as valid inputs to the decryption procedure, though. In these settings, the verification oracle is of no help to the adversary, as all ciphertexts are valid.

However, Chillotti et al. [32] and Zhang et al. [81] argue that the notion of a verification oracle should be extended from whether a ciphertext *decrypts*, to whether a ciphertext decrypts to something *meaningful*. They present the following scenario: a malicious cloud returns a 'dishonest' ciphertext to a client who has asked for a computation. If this ciphertext does not decrypt to something the client deems meaningful or valid, the client will ask the cloud for a recomputation, thus leaking that the decryption was not meaningful, and similarly leaking that a decryption was meaningful if no action is taken. Either result leaks information about the decrypted result. Chillotti et al. and Zhang et al. show that the very structure of homomorphic encryption make schemes susceptible to these fault injection, safe-error and reaction attacks, and that the attacks are a viable threat to real-world applications.

Finally, keyed-homomorphic encryption is a primitive, suggested by Lai et al. [55], closely related to homomorphic encryption. Here, any homomorphic evaluation of a ciphertext requires access to a particular evaluation key, which does not enable decryption. Lai et al. suggest an instantiation of such a scheme and prove that encrypted messages are indistinguishable even if an adversary has access to a decryption oracle.

2 Preliminaries

2.1 Notation

We use two distinct modular reductions: $p = r \bmod q$ denotes centrally reducing p modulo q to $r \in (-q/2, q/2]$, whilst $p = r \mod q$ denotes the modular reduction to $r \in [0, q-1]$. In both cases, we may also write $p \equiv r \bmod q$ or $p \equiv r \mod q$ if we wish to stress that p is equivalent to r modulo q: $p = r + nq$. This convention generalises to vectors and polynomials.

Vectors are denoted by bold lower case letters, whilst matrices are denoted by bold upper case letters. Normal letters are used to denote both integers and polynomials. We have aimed to strike a balance between having a consistent notation throughout the paper and also not diverging too much from the original articles. For example, parameters are denoted by Greek letters for schemes defined over the integers but not for the other schemes.

Several schemes use the gadget vector or matrix with respect to a base b and a parameter l: the gadget vector $\mathbf{g}$ is defined as the column vector $(1, b, \ldots, b^{l-1})^T$, and the gadget matrix is defined as $\mathbf{G} = \mathbf{I}_n \otimes \mathbf{g} \in \mathbb{Z}^{nl \times n}$ (i.e., $\mathbf{G}$ is a matrix with $\mathbf{g}$ on the 'diagonal'). For an integer $a < b^l$, we define the function $g^{-1}(a) : \mathbb{Z} \to \mathbb{Z}^l$ so that $g^{-1}(a)$ is the row vector with the (signed) base b decomposition of a, i.e., $g^{-1}(-a) = -g^{-1}(a)$ for a positive integer a, and $g^{-1}(a) \cdot \mathbf{g} = a$. This notion extends naturally to vectors, so for an n-length vector $\mathbf{a}$, we define the function $G^{-1}(\mathbf{a}) = [g^{-1}(a_1), g^{-1}(a_2), \ldots, g^{-1}(a_n)] \in \mathbb{Z}^{nl}$. The function extends similarly to matrices: for a matrix $\mathbf{A} \in \mathbb{Z}^{n \times n}$, $G^{-1}(\mathbf{A}) \in \mathbb{Z}^{n \times nl}$ simply applies G^{-1} to each row of $\mathbf{A}$. Moreover, $G^{-1}(\mathbf{A}) \cdot \mathbf{G} = \mathbf{A}$ [68].

We always denote schemes by the initials of the authors or the first three letters of the surname if there is a single author. This is mostly in line with the naming convention of the field, but we stress that we do this also for schemes with other proposed names, e.g., TFHE (CGGI [31]) and YASHE (BLLN [14]).

2.2 Notions

There are different flavours of homomorphic encryption, and we briefly present them here. As previously mentioned, all suggested schemes use the same approach of adding noise to a message to encrypt it, which builds up as the ciphertext is evaluated. If the noise grows too much, decryption is not guaranteed to be correct. For a more thorough discussion, see Appendix A.

- *Somewhat homomorphic encryption* (SHE) refers to schemes able to perform a limited number of homomorphic additions and/or multiplications before an evaluated ciphertext is not guaranteed to decrypt correctly. Although the number of operations may be estimated, it cannot be set explicitly.
- *Leveled homomorphic encryption* (LHE) schemes are similar to SHE schemes in that they allow for a limited amount of operations to be performed on a ciphertext. Here, though, the amount *can* be set explicitly, and is included as a parameter, the 'levels' L, in the key generation.

– *Fully homomorphic encryption* (FHE) allows for an unlimited number of homomorphic operations to be performed on a ciphertext. The only known way of achieving an FHE scheme is to bootstrap an SHE or LHE scheme.

It is required that all these types of schemes achieve *compactness*:[4] there exists a polynomial p such that the size of any evaluated ciphertext is less than $p(\lambda)$, where λ is the security parameter of the scheme in question. In other words, the growth of an evaluated ciphertext should be independent of both the size of the circuit and, in the case of LHE, the level parameter L [6].

We refer to all these three types of schemes collectively as *HE schemes to emphasise that the schemes are homomorphic both with respect to addition and multiplication. We also refer to them as 'homomorphic encryption schemes', which must *not* be confused with group homomorphic schemes.

We recall the notion of *indistinguishability under (non-adaptive) chosen ciphertext attack* (IND-CCA1) for an encryption scheme $\mathcal{E} = (\text{KeyGen}, \text{Enc}, \text{Dec})$: any PPT adversary $\mathbb{A}$ has at most a $1/2 + \epsilon$ chance of winning the following game against a challenger $\mathcal{C}$, where ϵ is negligible in the security parameter λ:

– $\mathcal{C}$ draws a key pair $(pk, sk) \leftarrow \text{KeyGen}(params)$, and sends pk to $\mathbb{A}$.
– $\mathbb{A}$ makes polynomially many ciphertext queries to her decryption oracle $\mathcal{O}_{\text{Dec}}$, which returns $\text{Dec}(c)$ for any ciphertext c that $\mathbb{A}$ has sent it.
– $\mathbb{A}$ sends two plaintexts of equal length (m_0, m_1) to $\mathcal{C}$.
– $\mathcal{C}$ returns $c \leftarrow \text{Enc}(pk, m_b)$ to $\mathbb{A}$, for a randomly chosen bit $b \in \{0, 1\}$.
– $\mathbb{A}$ outputs the bit b^*, and wins if $b^* = b$.

Here, $params$ denotes the security parameter λ of the scheme, and possibly other parameters output by a $\text{SetUp}(\lambda)$ algorithm. The notion of IND-CPA security is defined in a similar way, but here, $\mathbb{A}$ does not have access to a decryption oracle.

Finally, we note that all the attacks discussed in this paper are *adaptive key recovery attacks*, where an adversary with access to a decryption oracle is able to recover the secret key. These attacks are strictly stronger than a regular IND-CCA1 attack, as it allows an adversary to decrypt any ciphertext of her choosing, not just distinguishing between the encryptions of two chosen messages.

3 Schemes

Most schemes are grouped according to which known attack they are susceptible to, which largely corresponds to which problem they are based on, and we let the titles of subsections reflect this. There are a handful of schemes that do not fit into either of these groups, i.e. not broken by a known attack nor based on the same problem as previously broken schemes, which are presented separately in Section 3.5. Finally, we briefly discuss noise-free *HE. Unless explicitly stated, none of the schemes or constructions discussed are fully homomorphic.

All the attacks presented in Section 4 and Section 5 only use decryption results of ciphertexts to reconstruct the secret key. We therefore focus on key

[4] Not all definitions insist SHE schemes achieve compactness [6].

Parent	Child(ren)
BGV [17]	CKKS [29]
Bra12 [15]	FV [43]
FV [43]	BCIV [13], AN [5], CLPX [25], CIL [24], AH [4]
GSW [48]	BV14 [21], BL [9], CM [33], CCS [23], CGGI [31], PS [66], BP [18], Jou [52]
Gen [46]	**SS** [74]
SV [72]	**GH** [47], **LMSV** [60], **SV14** [73]
vDGHV [76]	*CLT* [34], *KLYC* [54], *CNT* [36], *CCKLLTY* [27], *CMNT* [35]

Table 2. The genealogy of various homomorphic schemes. 'Children' are schemes directly based on the 'parent' scheme. Schemes in bold are based on ideal lattices, schemes in italics are defined over the integers, and the rest are schemes based on (R)LWE. Schemes which are not based directly on a parent scheme ('orphans') are not listed.

generation, encryption and decryption when presenting schemes. In particular, we do not discuss the addition or multiplication algorithms of the schemes.

As previously mentioned, several schemes are based on earlier work, sometimes more or less copying a 'base' scheme. We refer to such a base scheme as a 'parent' and a scheme that has been based on it (and greatly resembles it) as its 'child'; see Table 2 for an overview of this genealogy. Seeing as there are so many schemes with great similarities, we do not present *all* suggested *HE schemes, but rather the parent schemes, from which others are easily derived, and present a generalisation of all proposed schemes in the (R)LWE case. We also present 'orphan' schemes: those that are not directly based on earlier work.

Finally, we note that, for simplicity, some encryption schemes are presented as symmetric key, rather than as public key. For all the schemes where this is the case, there is a standard transformation to make the scheme public key (see, e.g., [76]): the secret key remains the same, and the public key is a set of different encryptions of the additive identity element in the message space, e.g., 0. A message m is then, roughly speaking, encrypted by adding a small, random subset of the public key to m or an encoding of m, whilst decryption is the same as in the symmetric scheme. For details on the transformation of a specific scheme, the reader is referred to the original article of the scheme in question.

3.1 (R)LWE

There are several (R)LWE-based *HE schemes, most of which are children of BGV [17], Bra12 [15], or GSW [48]. These schemes have a common structure:

- The private key is a vector $\boldsymbol{s} \in R^n$ for some polynomial ring R (in the case of RLWE) or for $R = \mathbb{Z}_q$ (in the case of LWE). For RLWE, $n = 1$. The private key is drawn from either a bounded Gaussian distribution or a uniform distribution over polynomials with binary or ternary coefficients.
- The public key generation first computes an (R)LWE sample $\boldsymbol{a'} = \boldsymbol{A}\boldsymbol{s} + \boldsymbol{e}$, where $\boldsymbol{A} \in R^{N \times n}$ is a randomly sampled matrix and $\boldsymbol{e} \in R^N$ is sampled from a noise distribution χ. Then the public key $\boldsymbol{PK} \in R^{N \times (n+1)}$ is constructed using $\boldsymbol{A}$ and $\boldsymbol{a'}$ such that $\boldsymbol{PK} \cdot (-\boldsymbol{s} \parallel 1) = \boldsymbol{e}$.

– Encryption of a message $m \in \mathbb{M}$ first encodes it as $\boldsymbol{m'} \in R^{n+1}$ (e.g., $\boldsymbol{m'} = (0, \ldots, 0, m)$), samples some randomness $\boldsymbol{r} \in R^N$ and outputs $c = \boldsymbol{r} \cdot \boldsymbol{PK} + \boldsymbol{m'} \in R^{n+1}$. In some variants, $\boldsymbol{m'}$ and $\boldsymbol{r}$ are matrices instead of vectors.
– Decryption parses the ciphertext as $c = (\boldsymbol{a}, b)$ where $\boldsymbol{a} \in R^n$ and $b \in R$, then computes $m = \rho(\langle \boldsymbol{a}, \boldsymbol{s} \rangle - b)$ (for LWE), or $m = \rho(\boldsymbol{a} \cdot \boldsymbol{s} - b)$ (for RLWE) where $\rho : R \to \mathbb{M}$ is a rounding function into the plaintext space.

3.2 Ideal lattices

The most important schemes in this category are Gentry's original scheme Gen [46], and its most notable children GH [47] and SV [72]. Gentry presented his original scheme in rather general terms, whereas later schemes are specialisations of it. As a consequence, the 'general' scheme Gen is based on BDDP and SSSP,[5] but the 'specialisation schemes' GH, SV and their children are based on a different, but related problem, namely the short principal ideal problem (SPIP). This computational assumption has since been proven insecure [10,11,38], meaning in particular that SV and its children (see Table 2) are not IND-CPA secure.

For this reason, we choose not to present any particular scheme in this subsection, nor any attack later in the article. Instead, we mention that both GH and SV were proven to be susceptible to an adaptive key recovery attack, which also extends to Gen [60,40]. The LMSV scheme was suggested in the same paper as the attack [60], which was considered the only IND-CCA1 secure SHE scheme for many years. The starting point for the LMSV construction was the SV scheme, though, and it is therefore now known to not be IND-CPA secure. We will discuss the LMSV scheme and the construction it relied on in Section 6.1.

3.3 Approximate Greatest Common Divisor (AGCD)

vDGHV. The original homomorphic scheme defined over the integers was presented by van Dijk et al. [76], and we refer to it as vDGHV.

The bit-length η of the secret key is chosen according to the security parameter λ, as is the noise distribution χ. The symmetric encryption scheme is:

KeyGen: Choose an odd integer p from the interval $[2^{\eta-1}, 2^{\eta})$. Output $sk = p$.
Enc$(p, m \in \{0, 1\})$: Draw $q, r \leftarrow \chi$ such that $2r < p/2$, output $c = pq + 2r + m$.
Dec(p, c): Output $(c \bmod p) \mod 2$.

The IND-CPA security of vDGHV is based on the difficulty of the AGCD problem: given a collection of ciphertexts, it should not be possible for an adversary to deduce p, the approximate greatest common divisor of the ciphertexts.

BBL. Another scheme based on the AGCD problem is BBL [8], which differs significantly from the vDGHV scheme. The BBL paper presents two schemes: a basic and a batched construction. Since the decryption of the batched construction is a generalisation of the basic one, we present the basic scheme here.

[5] Bounded Distance Decoding Problem, and Sparse Subset Sum Problem

The scheme has public parameters $(\gamma, \rho, \eta, \tau)$, all dependent on the security parameter λ. The parameter γ is the bit-length of a component of the public key, and alongside ρ and the secret key p, it defines the noise distribution $\chi_{\gamma,\rho}$ used during key generation. For the parameters γ, ρ, p, the noise distribution is defined as follows: draw integers $q \leftarrow \mathbb{Z} \cap [0, 2^\gamma)$, $r \leftarrow \mathbb{Z} \cap (-2^\rho, 2^\rho)$, and output $pq + r$. Additionally, τ defines the number of components in the public key, and finally η defines the bit length of the secret key p. Recall that $\mathbf{g}^T$ is the γ-length gadget row vector, and $g^{-1} : \mathbb{Z} \to \{0,1\}^\gamma$ produces a column vector.

KeyGen: Sample an η-bit integer p, and sample an integer $x_0 \leftarrow \chi_{\gamma,\rho}$ such that the bit length is γ. Then sample τ integers $x_i \leftarrow \chi_{\gamma,\rho}$ such that $x_i \leq x_0$ for $1 \leq i \leq \tau$; we write $\mathbf{x} = [x_1, \ldots, x_\tau]$. Output $pk = (x_0, \mathbf{x}), sk = p$.
Enc(pk, m): Draw a matrix $\mathbf{S} \leftarrow \{0,1\}^{\tau \times \gamma}$, and output $\mathbf{c} = m\mathbf{g}^T + \mathbf{xS} \bmod x_0$.
Dec$(p, \mathbf{c})$: Compute $\mu = \mathbf{c}g^{-1}(p/2) \bmod p$. If $|\mu| \geq p/4$, return 1, else return 0.

Per. The schemes defined over the integers presented so far only encrypt single bits, although BBL may be batched. To remedy this, Pereira proposed a scheme that operates natively on vectors and matrices [68]. Furthermore, the size of the message space is a separate parameter that can be made as large as required.

The parameters of the scheme are similar to those of vDGHV and BBL: η is the bit-length of the secret prime p, and γ is the bit-length of the integer x_0 used for modular reductions of ciphertexts. There is also the noise distribution χ and noise parameter ρ. In addition, the scheme relies on the dimension n of vectors and matrices being encrypted, and the bound B on message size: for any integer m being encrypted, we have $|m| \leq B$, which naturally extends to vectors and matrices. Finally, we also have the parameter b, which is the base for the gadget matrix/vector, and the gadget decomposition function G^{-1}. We note that the scheme is designed so that the matrix and vector encryption and decryption use the same secret key material, and that the scheme only allows for matrix-matrix and vector-matrix multiplications.

KeyGen$(\lambda, B, n, \eta, \rho, \rho_0, \gamma)$: Draw an η-bit prime p, then sample x_0 from $\chi_{\rho_0,p}$ such that the bit-length of x_0 is γ and $x_0 = qp + r$ for $|r| \leq 2^{\rho_0}$. Sample $\mathbf{K}$ uniformly from $\mathbb{Z}_{x_0}^{n \times n}$ until $\mathbf{K}^{-1} \mod x_0$ exists. Finally, define $\alpha = \lfloor \frac{2^{\eta-1}}{2B+1} \rceil$, output $sk = (p, \mathbf{K})$, and update the set of public parameters $\{B, n, \eta, \rho, \rho_0, \gamma\}$ to include $\{\alpha, x_0\}$.
EncMat$(sk, \mathbf{M})$: Construct the matrix $\mathbf{X} = p\mathbf{Q} + \mathbf{R}$ by sampling each matrix element from $\chi_{<x_0}$, which only outputs elements smaller than x_0. Compute $\mathbf{C} = (\mathbf{X} + \mathbf{GKM})\mathbf{K}^{-1} \bmod x_0$, and output $\mathbf{C}$.
DecMat$(sk, \mathbf{C})$: Compute $\mathbf{C}' = G^{-1}(\alpha\mathbf{K}^{-1})\mathbf{CK} \bmod x_0$, then $\mathbf{C}^* = \mathbf{C}' \bmod p$, and finally output $\lfloor \mathbf{C}^*/\alpha \rceil$.
EncVec$(sk, \mathbf{m})$: Construct an n-length vector $\mathbf{x} = p\mathbf{q} + \mathbf{r}$, again by sampling every vector element from $\chi_{<x_0}$. Compute and output $\mathbf{c} = (\mathbf{x} + \alpha\mathbf{m})\mathbf{K}^{-1}$.
DecVec$(sk, \mathbf{c})$: Compute $\mathbf{c}' = \mathbf{cK} \bmod x_0$, then $\mathbf{c}^* = \mathbf{c}' \bmod p$. Return $\lfloor \mathbf{c}^*/\alpha \rceil$.

The indistinguishability of ciphertexts is based on AGCD, as the hardness of AGCD ensures that the distribution from which $\mathbf{X}$ and $\mathbf{x}$ are drawn is indistinguishable from the uniform distribution over $\mathbb{Z}_{x_0}$ (Lemma 3 of [68]). Using this indistinguishability, security is proven by a hybrid argument.

3.4 NTRU

There are three schemes that closely resemble NTRU: LATV [61], RC [71], and BLLN [14]. Despite great similarities, they are based on different problems: BLLN is based on RLWE, whilst LATV and RC are based on the Decisional Small Polynomial Ratio (DSPR) problem (defined in [61]), which may be regarded as the standard NTRU problem restricted to a specific set of parameters.

Albrecht et al. [2] found an attack on LATV which recovered the secret key using only the public material, and the same attack was later shown to apply to the RC scheme [51]. The attack takes advantage of precisely the specific parameters of the DSPR problem, implying that neither LATV nor RC are IND-CPA secure, and we therefore do not discuss these schemes further.

Despite the similarities between the schemes, BLLN is not affected by the Albrecht et al. attack, and we present the scheme below. BLLN is based on the version of NTRU proven by Stehlé and Steinfeld to be as secure as RLWE [75]. The size of the parameters chosen for BLLN ensures that the proof of security of Stehlé and Steinfeld extends to BLLN as well. This is not the case for LATV or RC, which is why these schemes reduce to DSPR.

BLLN. The BLLN scheme is defined over the ring $R = \mathbb{Z}[x]/\Phi_d(x)$, where $\Phi_d(x)$ is the d^{th} cyclotomic polynomial. Despite this general definition, we present the case where the ciphertext ring is $R = \mathbb{Z}[x]/(x^n + 1)$ for n a power of two, as this is the parameter the authors mention specifically.

The plaintext space is defined as $R_p = R/pR$ and the ciphertext space as $R_q = R/qR$, for integers $p \ll q$, where we also require that $\gcd(p, q) = 1$. In addition, BLLN uses a bounded noise distribution χ_k defined over R_q for key generation, and a separate noise distribution, χ_e, for encryption. Both these noise distributions are typically some truncated discrete Gaussian, with bounds B_k and B_e, respectively.

ParamsGen(λ): Given λ, fix n to determine the ring $\mathbb{Z}[x]/(x^n + 1)$. The security parameter also determines the moduli q and p, as well as the noise distributions χ_k and χ_e.

KeyGen(n, q, p, χ_k, χ_e): Draw $f', g \leftarrow \chi_k$, set $f = pf' + 1 \bmod q$, compute $f^{-1} \in R_q$ (redrawing if f^{-1} does not exist), and output $(pk, sk) = (h = gf^{-1}, f)$.

Enc($pk, m \in [-p/2, p/2)$): Draw $r, e \leftarrow \chi_e$, and compute $c = \lfloor q/p \rfloor m + r + he \bmod q$ as an element of R.

Dec(sk, c): Compute and output $m = \lceil \frac{p}{q}(fc \bmod q) \rfloor \bmod p \in R$.

3.5 Miscellaneous schemes

AFFHP. Albrecht et al. [3] proposed AFFHP, a scheme whose security relies on the hardness of computing Gröbner bases for ideals of multivariate polynomial rings and the ideal membership (IM) problem. The scheme works over the polynomial ring $\mathbb{F}_q[x_0, \ldots, x_{n-1}]$ for a prime q.

The IM problem states that for a general ideal $I \subset \mathbb{F}_q[x_0, \ldots, x_{n-1}]$ it is hard to determine if a random polynomial from $\mathbb{F}_q[x_0, \ldots, x_{n-1}]$ lies in I or not. AFFHP is designed with respect to general Gröbner bases for ideals in $\mathbb{F}_q[x_0, \ldots, x_{n-1}]$. However, homomorphic multiplication only works when a particular parameter d is equal to 1, resulting in the reduced Gröbner basis always having the form $\mathsf{G} = \{x_0 - a_0, x_1 - a_1, \ldots, x_{n-1} - a_{n-1}\}$ for $a_i \in \mathbb{F}_q$. Since we are only concerned with schemes that are *HE, we only consider this case. Note that for any $f \in \mathbb{F}_q[x_0, \ldots, x_{n-1}]$, we will always have $f \mod \mathsf{G} \in \mathbb{F}_q$ for Gröbner bases of the mentioned form.

The secret key of the scheme is G and the message space is $\{0, 1\}$. A distribution of "small" polynomials e is used for noise. Encryption and decryption of the symmetric key variant of AFFHP then works as follows:

Enc(m): Draw f of bounded degree at random from $\mathbb{F}_q[x_0, \ldots, x_{n-1}]$. Compute $f_0 = f - (f \mod \mathsf{G})$, and select $e \leftarrow \chi$ at random. Return $C = f_0 + 2e + m$.
Dec(C): Compute and return $(C \mod \mathsf{G}) \mod 2$.

DHPSSWZ. The SHE scheme DHPSSWZ was proposed by Doröz et al. [41], where the security is based on the Finite Field Isomorphism (FFI) problem. Informally, the FFI problem states that for a prime q, elements chosen from a particular distribution in one representation of the field $\mathbb{F}_{q^n}$ are distributed uniformly at random when mapped to a different representation of $\mathbb{F}_{q^n}$.

Let f and F be two monic irreducible polynomials of degree n over $\mathbb{F}_q$. The message space of the scheme is elements $m(x) \in \mathbb{X} = \mathbb{F}_q[x]/(f(x))$, where the coefficients of $m(x)$ are taken from $\{0, 1\}$. To stunt the growth of the noise when performing operations on the ciphertexts, f has to be sparse and have small coefficients, typically from $\{-1, 0, 1\}$. The ciphertext space is $\mathbb{Y} = \mathbb{F}_q[y]/(F(y))$. The private key consists of f, as well as explicit isomorphisms between $\mathbb{X}$ and $\mathbb{Y}$. More specifically, $\phi(y)$ is the particular element of $\mathbb{Y}$, written as a polynomial in y of degree at most $n - 1$, which is the root of f denoted by x in $\mathbb{X}$. Likewise, $\psi(x)$ is the specific element of $\mathbb{X}$ isomorphic to the root of F labelled y in $\mathbb{Y}$. Both $\phi(y)$ and $\psi(x)$ are therefore understood as fixed, known polynomials in the description of the encryption and decryption. The encryption and decryption of the symmetric key variant of the scheme are done as follows:

Enc($m(x)$): Draw $r(x)$ from a distribution giving small polynomials. Output $C(y) = 2r(\phi(y)) + m(\phi(y)) \underline{\mod} F(y)$.
Dec($C(y)$): Replace y by $\psi(x)$ in the polynomial C and output $(C(\psi(x)) \underline{\mod} f(x)) \mod 2$.

LGM. As we will see in Section 4.1, schemes based on LWE (e.g., GSW) can leak one bit of the secret key from a small number of decryption queries. Essentially, they have public keys of the form $(\boldsymbol{A}, \boldsymbol{A}\boldsymbol{s}+\boldsymbol{e})$ with secret key $\boldsymbol{s}$, and key recovery attacks either compute $\boldsymbol{s}$, or compute the noise $\boldsymbol{e}$ and use Gaussian elimination to derive $\boldsymbol{s}$. Li, Galbraith and Ma (LGM, [58]) proposed a technique to circumvent such attacks. Firstly, they start with a dual version of GSW, where the public key is of the form $(\boldsymbol{A}, \boldsymbol{A}\boldsymbol{s})$, and the secret key $\boldsymbol{s}$ has small norm; security then depends on the hardness of the inhomogeneous short integer solution (ISIS) problem. Secondly, instead of having one secret key vector $\boldsymbol{s}$, they have t of them; decryption works by using a different random linear combination of the secret keys $\boldsymbol{s}' = \sum_{i=1}^{t} \lambda_i \boldsymbol{s}_i$ for each decryption query. Hence a decryption query leaks, at best, one bit of $\boldsymbol{s}'$: an unknown linear combination of secret key vectors which with high probability will not be reused in other decryption queries. Li et al. argued that this would prevent adaptive key recovery attacks, however, such an attack was later found by Fauzi et al. [45].

The attack does not extend to any other scheme, as the LGM is the only scheme which employs the strategy of using 'ephemeral' secret keys to thwart adaptive key recovery attacks. We present the LGM scheme and provide the intuition of the attack in Section 4.4 for completeness, and refer to Fauzi et al. [45] for further details. Note that, here, κ is the security parameter, whilst λ is a sampled variable.

Setup$(1^\kappa, 1^L)$: Let $n = n(\kappa, L)$ and $m = m(\kappa, L)$, choose a modulus q and bounded noise distribution $\chi = \chi(\kappa, L)$ on $\mathbb{Z}$ such that at least 2^κ security against known attacks is achieved. Choose the number of secret keys $t = O(\log n)$. Let $l = \lfloor \log q \rfloor + 1$ and $N = (t+m)l$. Output $params = (n, q, \chi, m, t, l, N)$.

KeyGen$(params)$: Uniformly sample $\mathbf{B} \in \mathbb{Z}_q^{n \times m}$. For $i \in [1, t]$, sample $\boldsymbol{e}_i$ from χ^m, set $\mathbf{u}_i = \mathbf{B}\boldsymbol{e}_i$ and set $\mathbf{s}_i = (\mathbf{r}_i \parallel -\boldsymbol{e}_i^T)^T$, where $\mathbf{r}_i$ is the i-th row of the $t \times t$ identity matrix. Return the public key $\mathbf{A} = [\mathbf{u}_1 \| \ldots \| \mathbf{u}_t \| \mathbf{B}] \in \mathbb{Z}_q^{n \times (t+m)}$ and the secret key $\mathbf{s} = (\mathbf{s}_1, \ldots, \mathbf{s}_t)$.

Enc$(\mathbf{A}, \mu \in \mathbb{Z}_2)$: Let $\mathbf{G}$ be the $(t+m) \times N$ gadget matrix. Sample $\mathbf{R} \leftarrow \mathbb{Z}_q^{n \times N}$ and $\mathbf{X} \leftarrow \chi^{(t+m) \times N}$. Output $\mathbf{C} = \mu \cdot \mathbf{G} + \mathbf{A}^T\mathbf{R} + \mathbf{X} \in \mathbb{Z}_q^{(t+m) \times N}$.

Dec$(\mathbf{s}, \mathbf{C})$: Sample $(\lambda_1, \ldots, \lambda_t) \in \mathbb{Z}_q^t \setminus \{0\}^t$ until the generated $\mathbf{s}' = \sum_{i=1}^{t} \lambda_i \mathbf{s}_i$ has sufficiently small norm. Let $i \in [1, t], j, I = (i-1)l + j$ be integers such that $\lambda_i \neq 0$, $2^{j-1} \in (q/4, q/2]$ and $I \in [1, tl]$. Compute $u = \langle \mathbf{C}_I, \mathbf{s}' \rangle \bmod q$, where $\mathbf{C}_I$ is the Ith column of the ciphertext matrix $\mathbf{C}$. Finally, output $|\lfloor u/2^{j-1} \rceil| \in \{0, 1\}$.

3.6 Noise-free attempts

All known *HE schemes have a non-constant ciphertext expansion. Gjøsteen and Strand [50] go further and conjecture that the security of a *HE scheme either depends on a massive ciphertext expansion or a weak or strange algebraic structure, limiting the applicability of the scheme. To support their conjecture,

they show that no noise-free and secure schemes can exist in vector spaces or fields; however, there are no final conclusions in the case of rings.

Nuida [65] proposed a generic construction of noise-free FHE based on surjective group homomorphisms $\pi : \mathbb{C} \to \mathbb{M}$. The generic construction is secure if elements in the kernel of π are indistinguishable from elements in the ciphertext space $\mathbb{C}$, which is a variant of the subgroup hiding assumption. Nuida provided a candidate construction using combinatorial group theory; however, it is non-compact (i.e., the ciphertext size can be unbounded), and does not have an explicit proof of IND-CPA security. Since other existing candidate constructions have similar issues, we will disregard noise-free *HE constructions in our work.

4 Existing attacks

4.1 (R)LWE attacks

Chenal and Tang [26] present attacks on several schemes based on (R)LWE, for example BGV [17] and GSW [48], see Table 1 for a full overview. The attacks recover the secret key bit by bit, and coefficient by coefficient (wherever this applies), by using the result of the decryption queries to perform a binary search on secret values.

Recall from Section 3.1 that schemes based on LWE have ciphertexts of the form $c = (\boldsymbol{a}, b)$ where $\boldsymbol{a} \in R^n$ and $b \in R$, for the ring $R = \mathbb{Z}_q$. The private key is a vector $\boldsymbol{s} \in R^n$ and decryption is computed as $m = \rho(\langle \boldsymbol{a}, \boldsymbol{s} \rangle - b)$, where ρ is some rounding function that maps elements from R into the plaintext space. In fact, any scheme with this type of decryption function is vulnerable to the following adaptive key recovery attack.

In the LWE case, the adversary asks for decryptions of $c = (\boldsymbol{e}_i, b)$, where $\boldsymbol{e}_i \in R^n$ is the unit vector with 1 at position i and 0 everywhere else, which leaks information on $\boldsymbol{s}_i$. Doing a binary search with different values for b allows the adversary to find a b_0 such that

$$\rho(\langle (0, \ldots, 1, \ldots, 0), \boldsymbol{s} \rangle - b_0) = m_0 \neq m_1 = \rho(\langle (0, \ldots, 1, \ldots, 0), \boldsymbol{s} \rangle - b_0 + \epsilon),$$

for an arbitrarily "small" value of ϵ. Note that ϵ is an element in a general ring R, but for the homomorphic properties to work with correct decryption, the elements of R need to have a notion of size and distance. Knowing exactly where the border b_0 is, where $\rho(\boldsymbol{s}_i - b_0)$ is rounded to either m_0 or m_1, allows the adversary to determine $\boldsymbol{s}_i$ to any degree of accuracy. Repeating for all positions $1 \leq i \leq n$ gives an adaptive key recovery attack on these schemes.

In the RLWE case, the general attack is equally simple. RLWE uses ciphertexts of the form $c = (a(x), b(x))$ where $a(x), b(x) \in R[x]$ for some polynomial ring $R[x]$ modulo an ideal I. For a secret key $s(x) \in R[x]$ the decryption function in this case takes the form $\mathrm{Dec}(c) = \rho(a(x) \cdot s(x) - b(x))$, where $a(x) \cdot s(x)$ denotes polynomial multiplication in $R[x] \mod I$ and ρ is some rounding function into the plaintext space. Simply querying the decryption oracle with ciphertexts of the form $c = (1, b(x))$ for various choices of $b(x)$ allows one to do a binary search on the coefficients of $s(x)$ in the same way as in the LWE case.

4.2 AGCD attacks

There are two attacks on homomorphic encryption schemes defined over the integers. The attack by Zhang et al. [80] recovers the secret key of the vDGHV scheme, an attack the authors themselves point out is directly transferable to CMNT [35]. Chenal and Tang later gave a more efficient key recovery attack against vDGHV [26]. The two attacks differ in strategy, and we therefore present both approaches. We remind the reader that a ciphertext in vDGHV is of the form $c = m + 2r + qp$, where the odd integer $p \in [2^{\eta-1}, 2^{\eta} - 1]$ is the secret key. Decryption first reduces the ciphertext modulo the secret key p, then modulo 2.

Zhang et al. construct several encryptions of 0, which they then alter so that all the different ciphertexts have a fixed noise. Given a ciphertext c encrypting 0, this construction is achieved by performing a binary search for an added noise term ρ such that $\mathrm{Dec}(c + \rho) = 0$, but $\mathrm{Dec}(c + \rho + 1) = 1$, and define the new ciphertext $c_i = c + \rho + 1$. Given the constructed ciphertexts $c_0, c_1, \ldots, c_k$, they compute $\gcd(c_0 - c_1, c_1 - c_2, \ldots, c_{k-1} - c_k)$, which will be p with overwhelming probability. The attack requires $O(\lambda^2)$ operations, where λ is the security parameter of the scheme.

Whereas Zhang et al. attacked the scheme by essentially constructing ciphertexts such that the underlying problem was easy, Chenal and Tang proposed a more direct search for the secret key. Initially, an adversary has both a lower and upper bound for p, namely $l_p = 2^{\eta-1} + 1$ and $u_p = 2^{\eta} - 1$, respectively. She then takes advantage of the following: if she queries the decryption oracle with an even number $c \in (l_p, u_p)$, it will return 0 if $c < p$, and 1 if $c > p$, as $\mathrm{Dec}(c) = (c \mod p) \mod 2$, and $c < 2p$. Therefore, if she finds the even integer c such that $\mathrm{Dec}(c) = 0$ and $\mathrm{Dec}(c + 2) = 1$, she knows that $p = c + 1$. The attack requires roughly η oracle queries, which is more efficient than the Zhang et al. attack.

4.3 NTRU attack

An adaptive key recovery attack against the NTRU-based scheme BLLN was presented by Dahab, Galbraith and Morais [40]; we give the main idea of the attack and its complexity, and refer to the original article for further details. The attack is not applied to other schemes later, and we state it for completeness.

The attack itself is fairly simple. It takes advantage of the fact that the decryption of BLLN involves multiplying the ciphertext with the secret key and then performing two modular reductions: first modulo q, then p. The idea is to query the decryption oracle with rounded fractions divisible by p. The fractions are also designed so that as long as the denominator is large compared to the secret key f, the fraction is not reduced modulo q, and the decryption algorithm will therefore output 0. However, as the denominator of the queried fraction becomes just smaller than f, the fraction becomes large enough to be reduced modulo q when multiplied with the secret key. This will cause the second modular reduction to result in something nonzero, making it easy to detect when the denominator became smaller than f, and hence calculate the secret key.

For BLLN, the complexity of the attack depends on the parameters of the scheme. If $p > 2$ and the coefficients of f', g are all in $\{-1, 0, 1\}$, it takes a single query to recover the secret key. In the more general case where $p \geq 2$, and no restrictions are placed on the polynomials, the complexity is $O(n \log(B_k))$, where n is the order of the polynomial defining the ciphertext space, and B_k is a bound on the largest coefficient of f.

4.4 LGM attack

Recall that the LGM scheme has a secret key vector consisting of t secret keys $(\mathbf{s}_1, \ldots, \mathbf{s}_t)$, where each key is of the form $\mathbf{s}_i = (\boldsymbol{e}_i \| r_i)$ for some noise vector $\boldsymbol{e}_i$, and that an ephemeral secret key $\mathbf{s}' = \sum_{i=0}^{t} \lambda_i \mathbf{s}_i$ is generated for each decryption procedure.

Although Li et al. discuss several possible distributions to sample λ_i from, their main focus is λ_i drawn uniformly from $\{0, 1\}$. We therefore present the attack for this scenario and refer to Fauzi et al. [45] for details on the attack for more general λ distributions. The attack recovers a vector $\boldsymbol{e}_i$ one component $e_{k,i}$ at the time, and each vector and component is recovered independently of each other. In particular, it is possible to calculate a single vector $\boldsymbol{e}_i$ to recover $\mathbf{s}_i$ and use it for decryption, and we therefore only discuss how to recover $\boldsymbol{e}_1$.

Note that we can construct a ciphertext matrix $\mathbf{C}$ such that the inner product during decryption always results in $\alpha + \sum_{i=0}^{t} \lambda_i e_{i,1}$, by ensuring that for each possible index I, the column $\mathbf{C}_I$ has α in position i, and 1 in position $t + 1$. By repeated queries of this 'diagonal' matrix, one obtains an estimate of $1/2 \sum_{i=0}^{t} e_{i,1}$ by observing how often the decryption oracle returns 0 and 1 for a specific value of α, as the decryption results in 0 if $\alpha + \sum_{i=0}^{t} \lambda_i e_{i,1} < 2^{j-2}$, and 1 otherwise. Essentially, the attack searches for the value of α where the resulting ciphertext decrypts as often to 0 as 1, as this indicates that the average value $\alpha + \sum_{i=0}^{t} e_{i,1}$ is very close to 2^{j-2}, and the adversary can now estimate $\sum_{i=0}^{t} e_{i,1}$

After this estimation of $1/2 \sum_{i=0}^{t} e_{i,1}$ is obtained, the decryption oracle is repeatedly queried with ciphertext matrices where every column is identical, and has a value a in position i, and a 1 in position $t + 1$. Decryption of such a matrix will always result in $\lambda_i a + \lambda e_{i,1} + \sum_{k \neq i} \lambda_k e_{k,1}$, which will provide an estimate of $e_{i,1} + 1/2 \sum_{k \neq i} \lambda_k e_{k,1}$, again by observing how often the decryption oracle returns 0 and 1 for different values of a. Combining these two estimations provides an estimation of $e_{i,1}$, and repeating this for the other indices of $\boldsymbol{e}_1$ allows the adversary to recover the secret key $\boldsymbol{s}_1 = (\mathbf{r}_1 \parallel -\boldsymbol{e}_1^T)^T$.

5 Attacking other schemes

We now discuss how to apply known attacks to other schemes, as susceptibility of an adaptive key recovery attack is inherited from a parent scheme to a child. Because the schemes over ideal lattices and those based on NTRU are, with a few exceptions, not IND-CPA secure, we limit this discussion to schemes based on (R)LWE and AGCD. We also present novel attacks against schemes that are not affected by the attacks presented in Section 4.

5.1 Applying the (R)LWE attack on other schemes

Many schemes based on (R)LWE are built on or slightly adapted from earlier schemes, as is clear from Table 2 in Section 3. For example, the CKKS scheme [29] is the result of a general framework being applied to the BGV scheme. The framework allows for recovery of the message by computing the MSB of a ciphertext, modulo some modulus. Because the structure of the BGV scheme is intact, the key recovery attack on BGV described in [26] is directly applicable to CKKS.

The Chenal and Tang (R)LWE attack also works on all children of GSW mentioned in Table 2. For most of these, the structure and decryption functions are almost identical, hence the attack trivially extends. A special case is CGGI [31], where the plaintext space is a subset of a torus,[6] but the decryption function is otherwise very similar to GSW and CKKS: it takes the inner product of a ciphertext with the private key, subtracts an element and rounds the result to the nearest element in the plaintext space. Hence the attack in Section 4.1 can still be adapted to CGGI.

Similarly, the attack also works on FV [43] and its children, found in Table 2. First, observe that FV itself is essentially an RLWE variant of the LWE-based Bra12 [15], where in particular, the decryption function is the same formula, but in the ring setting. Moreover, all the children follow FV's structure, including almost identical decryption functions; the differences in specific message spaces and algorithms for homomorphic evaluations do not affect the success rate of the Chenal and Tang attack.

The CS17 [37] scheme is a bit different from other (R)LWE schemes in that it relies on the learning with rounding (LWR) problem, which is closely related to (R)LWE. We have verified that the attack procedure from Section 4.1 still applies to the CS17 scheme; the decryption oracle can be made to return the secret key plus a scalable term, rounded into the plaintext space. A binary search on the scalable term allows the adversary to determine each component of the secret key to any degree of accuracy.

5.2 Attacks on AGCD-based schemes

Applying the known attack on other schemes. The adaptive key recovery attack by Chenal and Tang [26] on vDGHV described in Section 4.2 is applicable to all the children of vDGHV listed in Table 2, as well as CS15 [30]. The attack is either directly transferable or requires a trivial generalisation to account for vectors of messages and/or a larger message space than $\{0, 1\}$.

BBL. The BBL scheme presented in Section 3.3 is not immediately affected by any of the attacks presented in Section 4.2 We therefore present a novel adaptive key recovery attack against the scheme.

[6] The torus is defined as the set of real numbers modulo 1, which can be identified with the half-open interval $[0, 1)$ on the number line.

Recall that the decryption algorithm $\mathrm{Dec}(p, \mathbf{c})$ first computes $\mu = \mathbf{c}g^{-1}(p/2) \bmod p$, then outputs 1 if $|\mu| \geq p/4$, and 0 otherwise.

We note that the bit-length η of p is part of the public parameters, so the adversary has both a lower and an upper bound on p: $2^{\eta-1}$ and $2^{\eta} - 1$, respectively. She therefore also has the lower bound $2^{\eta-2}$ for $p/2$, and we emphasise that $2^{\eta-2}$ is not reduced $\bmod p$, for any value of p.

The attack proceeds as follows: the adversary will recover the i^{th} element of $g^{-1}(p/2)$ by querying the decryption oracle with the ciphertext with $2^{\eta-2}$ in position i, and 0 elsewhere. If the i^{th} component of $g^{-1}(p/2)$ is 1, the decryption oracle returns 1, and 0 otherwise. The vector $g^{-1}(p/2)$ is then completely recovered after $\eta - 1$ queries, after which the adversary may simply multiply with the corresponding gadget vector $\mathbf{g}$ to recover $p/2$.

Per. As for the BBL scheme, the Per scheme is not broken by any of the attacks presented in Section 4.2, and we therefore provide a novel adaptive key recovery attack against it. Although the Per scheme is defined for ciphertexts in both matrix and vector form, we only use the matrix version of the scheme to recover the secret key $sk = (p, \mathbf{K})$. We explain how to recover the secret integer p first, then how to recover the secret matrix $\mathbf{K}$. Recall that the decryption works as follows: for an input $\mathbf{C}$, compute $\mathbf{C}' = G^{-1}(\alpha\mathbf{K}^{-1})\mathbf{C}\mathbf{K} \bmod x_0$, then $\mathbf{C}^* = \mathbf{C}' \bmod p$, and finally output $\lfloor \mathbf{C}^*/\alpha \rceil$.

We denote the greatest common divisor of α and x_0 as d, and note that $1 \leq d \leq \alpha$. Since $\alpha \leq \frac{p}{2B+1} < \frac{p}{2}$, we also have that $d \bmod p = d$. Irrespective of the value of d, we let α^{-1} denote the integer such that $\alpha^{-1}\alpha \equiv d \bmod x_0$.

Step 1*: Recovering p.* Our first observation is that, using $G^{-1}(\alpha\mathbf{K}^{-1})\mathbf{G} = \alpha\mathbf{K}^{-1}$ over $\mathbb{Z}_{x_0}$ (see [68, Sec. 3.3]), the decryption of $\mathbf{C} = \alpha^{-1}\mathbf{G}$ proceeds as follows: $\mathbf{C}' = G^{-1}(\alpha\mathbf{K}^{-1})\alpha^{-1}\mathbf{G}\mathbf{K} = \alpha^{-1}\alpha\mathbf{K}^{-1}\mathbf{K} = d \cdot \mathbf{I}_n$, where $\mathbf{I}_n$ is the $n \times n$ identity matrix. Then $\mathbf{C}^* = (d \cdot \mathbf{I}_n) \bmod p = d \cdot \mathbf{I}_n$ and $\mathbf{M} = \lfloor d/\alpha \rceil \mathbf{I}_n$.

We use the ciphertext $\alpha^{-1}\mathbf{G}$ as a starting point to perform a binary search for a factor t such that $td < p/2$, but $(t+1)d \geq p/2$. If $td \leq p/2$, the decryption of $t\alpha^{-1}\mathbf{G}$ will return approximately $td/\alpha\mathbf{I}_n$, which has positive entries when $td < p/2$. If $td \geq p/2$ then the decryption of $t\alpha^{-1}\mathbf{G}$ will return a matrix with negative entries, or 0 if the entries have too small absolute value. It is therefore easy to detect the exact t which produces $td < p/2 \leq (t+1)d$.

For $d = 1$, $p = 2t + 1$ is now recovered. For $d \geq 2$, note that the inequality $td < p/2 \leq (t+1)d$ implies $3td < 3p/2 \leq 3(t+1)d$, and that we can narrow the interval by querying to see if $(3t+1)d < 3p/2$ or $(3t+2)d < 3p/2$. During decryption of $(3t+i)\alpha^{-1}\mathbf{G}$ $(i = 1, 2)$ we know that one whole multiple of p will be subtracted from $(3t+i)d$ when reducing modulo p. If such a subtraction happens, it is because $(3t+i)d \geq 3p/2$, and the decryption result will be negative. If there is no such subtraction, the decryption result will be positive, and we know that $(3t+i)d < 3p/2$. Hence we can narrow down the search interval for p by a factor 3, e.g., if it was determined that $(3t+1)d > 3p/2$ then $2td < p < 2(3t+1)d/3$. We continue multiplying the inequalities by 3 and narrowing down the interval

for p by a factor of 3 in each iteration. After $\log_3 d$ iterations, i.e., $O(\eta + \log_3(d))$ decryption queries, the interval contains a single digit and p is recovered.

The attack hinges on the assumption that the ciphertext is not reduced modulo x_0 during decryption. In order to avoid this modular reduction, we need $3^{\log_3(d)} td < x_0/2$. We have $3^{\log_3(d)} td = td^2 < td\alpha \leq p/2\alpha < 2^{\eta-1}2^{\eta-2} = 2^{2\eta-3}$, where we have used the fact that $\alpha = \lfloor 2^{\eta-1}/(2B+1) \rfloor$ and $B \geq 1$. Furthermore, we have that $x_0 > 2^{\gamma-1}$, and finally that it is required that $\gamma \geq 2\eta$ [68], so we conclude that the ciphertext queries will never be reduced modulo x_0, and our attack recovers p successfully.

Step 2: Recovering $\mathbf{K}$. Successfully recovering p already displays a serious weakness of the scheme, but for a full key recovery attack, we also need to recover the secret matrix $\mathbf{K}$. We show how to find $\mathbf{K}$ for $b = 2$ here, and sketch an attack procedure for a general b.

We take advantage of the decryption procedure including $G^{-1}(\alpha \mathbf{K}^{-1})$, and rather attempt to reconstruct this matrix instead of $\mathbf{K}$ directly. Note that the elements of $G^{-1}(\alpha \mathbf{K}^{-1})$ are all in the interval $[0, b-1]$.

Let $\mathbf{E}_{i,j}$ denote the matrix with 1 in position (i, j), and 0 elsewhere. Querying $\mathbf{E}_{i,j}$ to the decryption oracle will produce

$$\mathbf{C}' = \begin{bmatrix} b_{1,i}k_{j,1} & b_{1,i}k_{j,2} & \cdots & b_{1,i}k_{j,n} \\ b_{2,i}k_{j,1} & b_{2,i}k_{j,2} & \cdots & b_{2,i}k_{j,n} \\ \vdots & \vdots & \cdots & \vdots \\ b_{n,i}k_{j,1} & b_{n,i}k_{j,2} & \cdots & b_{n,i}k_{j,n} \end{bmatrix},$$

where $b_{u,v}$ is the (u, v)-th element of $G^{-1}(\alpha \mathbf{K}^{-1})$, and $k_{u,v}$ is the (u, v)-th element of $\mathbf{K}$. We note that $1 \leq u \leq nl$, and $1 \leq v \leq n$, where $l = \lceil \log_b(2^\gamma) \rceil$. We query the decryption oracle for all possible matrices $\mathbf{E}_{i,j}$.

If $b = 2$, $G^{-1}(\alpha \mathbf{K}^{-1})$ is a binary matrix. We then use the decryption results to determine if a particular element $b_{i,j}$ is non-zero by checking whether the vector of the form $\lfloor b_{i,j} \left[k_{i,1} \cdots k_{i,n} \right] / \alpha \rceil$, obtained from i-th row of the decryption query on $\mathbf{E}_{j,i}$, is non-zero. If it is, we set $b_{i,j} = 1$, else we set it to 0. We can thus obtain $G^{-1}(\alpha \mathbf{K}^{-1})$, and hence also $\alpha \mathbf{K}^{-1}$. It is then easy to reconstruct $\mathbf{K}$.

We now sketch how an attack for a general b can work, but leave out a rigorous procedure. First we note the sizes of various parameters as recommended by [68]: $x_0 \approx 2^{160}, p \approx 2^{80}, 2^{60} \leq \alpha \leq 2^{78}, b \approx 2^7$, and $r \approx 2^{40}$ where $x_0 = qp + r$.

The secret elements $k_{i,j}$ are random in $\mathbb{Z}_{x_0}$ and are multiplied with $b_{i',j'}$ where $b_{i',j'} < b$. Since r and b are relatively small compared to the other parameters, we find that first reducing $b_{i',j'}k_{i,j}$ modulo x_0 and then reducing the result modulo p is approximately the same as just reducing modulo p directly:

$$|(b_{i',j'}k_{i,j} \bmod p) - ((b_{i',j'}k_{i,j} \bmod x_0) \bmod p)| \leq br.$$

With $b \approx 2^7$ and $r \approx 2^{40}$, this difference will be mostly negligible when dividing by $\alpha \geq 2^{60}$ before output. The result will be the same with high probability, or just one off if not the same. Hence we simplify further analysis by just reducing modulo p.

The adversary gets to see the values $\lfloor (b_{i',j'}k_{i,j} \bmod p)/\alpha \rceil$ for all choices of i, j, i', j'. Looking at $\mathbf{C}'$, we see that all elements in one column share the same $k_{i,j}$. Assume for concreteness that $\alpha = 2^{70}$, and that $\lfloor (b_{1,1}k_{1,1} \bmod p)/\alpha \rceil$ is the smallest positive value among the values $\lfloor (b_{i,1}k_{1,1} \bmod p)/\alpha \rceil$ that the attacker sees, which will be in the range $[-511, 512]$.

We then know that $\lfloor (ab_{1,1}k_{1,1} \bmod p)/\alpha \rceil \approx a\lfloor (b_{1,1}k_{1,1} \bmod p)/\alpha \rceil$ for some small values of a, since they will not lead to new reductions modulo p. We can estimate when $b_{i,1} = ab_{1,1}$ by looking at the ratio

$$\frac{\lfloor \frac{(b_{i,1}k_{1,1} \bmod p)}{\alpha} \rceil}{\lfloor \frac{(b_{1,1}k_{1,1} \bmod p)}{\alpha} \rceil} \approx \frac{b_{i,1}}{b_{1,1}} = a.$$

Repeating with different $k_{j,1}$ produces n different estimates of this ratio, which should give approximately the same value for all cases where $\lfloor (b_{1,1}k_{j,1} \bmod p)/\alpha \rceil$ is smaller than $512/a$. We then learn the linear relation $b_{1,1} = ab_{1,i}$. Repeating for different $b_{i,j}$-elements allows the adversary to create a linear system between them, which can either be solved or whose solution space is small enough to be exhaustively searched as b is only of size 2^7.

5.3 Attacks on AFFHP and DHPSSWZ

We now present new attacks on AFFHP and DHPSSWZ using a variant of binary search. Consider the value of $(ka \mod q) \mod p$ for an unknown a and chosen multiple k, where q is prime and $q \gg p$. We show how to recover a in this scenario. In the following we identify $\mathbb{F}_q$ with $[0, q-1]$, and to ease notation we let $D(ka) \in \mathbb{F}_p$ denote the oracle that returns $(ka \mod q) \mod p$.

First, query $D(a)$. As $a < q$, this will simply give us the value of $a \mod p$. As long as $ka < q$ we know that $D(ka) = (k \mod p)(a \mod p)$, and we can check how long this property holds by asking for $D(ka)$ for $k \in \mathbb{N}$. For some k we reach the point where $D(ka) = ka \mod p$ but $D((k+1)a) = ((k+1)a \mod p) - (q \mod p)$. Note that $q \mod p \neq 0$ as q is prime. We are therefore able to determine the exact value $k = k_0$ such that $k_0 a < q \leq (k_0+1)a$. Since a is uniformly distributed over $\mathbb{F}_q$, we expect $O(1)$ queries to determine k_0.

We now have the following inequalities

$$k_0 a < q \leq (k_0+1)a \Leftrightarrow \frac{q}{k_0+1} \leq a < \frac{q}{k_0}$$

Multiplying through by 2 we get $2k_0 a < 2q \leq (2k_0+2)a$. Ask for $D((2k_0+1)a)$. Knowing the value of $a \mod p$, we can determine if the reduction of $(2k_0+1)a$ modulo q subtracted q or $2q$ before reducing modulo p. This determines whether $2k_0 a \leq 2q \leq (2k_0+1)a$ or $(2k_0+1)a \leq 2q \leq (2k_0+2)a$. In other words, we can find k_1 such that $k_1 a_0 < 2q \leq (k_1+1)a_0$. This gives the following inequalities

$$k_1 a < 2q \leq (k_1+1)a \Leftrightarrow \frac{2q}{k_1+1} \leq a < \frac{2q}{k_1}.$$

We continue like this, multiplying through with 2 and asking the oracle $D((2k_1+1)a)$ to determine the value k_2 such that $k_2 a < 4q \leq (k_2+1)a$, etc. After t iterations we have the following inequalities

$$k_t a < 2^t q \leq (k_t+1)a \Leftrightarrow \frac{2^t q}{k_t+1} \leq a < \frac{2^t q}{k_t}.$$

We now show that the interval where a can be found shrinks exponentially fast with increasing t, and only $O(\log q)$ queries are needed to determine a exactly.

First note that $k_0 \geq 1$, and by induction $k_t \geq 2^t$ for all $t \geq 0$. Define $d(t)$ to be the size of the interval where a can be found after t iterations. We then get

$$\frac{2^t q}{k_t} - \frac{2^t q}{k_t+1} = d(t) \quad \Longrightarrow 2^t q = d(t)k_t(k_t+1) > d(t)2^{2t} \quad \Longrightarrow \frac{q}{2^t} > d(t)$$

Hence, after $O(\log q)$ queries the attacker is able to determine a.

CCA key recovery attack on AFFHP Recall that the secret key of AFFHP is a Gröbner basis $\mathsf{G} = \{x_0 - a_0, \ldots, x_{n-1} - a_{n-1}\} \subset \mathbb{F}_q[x_0, \ldots, x_{n-1}]$, where the a_i are unknown coefficients from $\mathbb{F}_q$. For a given ciphertext $C \in \mathbb{F}_q[x_0, \ldots, x_{n-1}]$, the decryption function returns $(C \mod \mathsf{G}) \mod 2$. Setting $C = x_i$ will then yield $x_i \mod \mathsf{G} = a_i$, so the decryption oracle will output $a_i \mod 2$. More generally, the decryption of $C = kx_i$ for some $k \in \mathbb{F}_q$ will return $ka_i \mod 2$. The attacker can therefore recover all the unknown a_i by using the method described above. The complexity for this attack is $O(n \log q)$ decryption queries.

CCA key recovery attack on DHPSSWZ Recall that the decryption function of DHPSSWZ for a ciphertext $C(y)$ is given as $\mathrm{Dec}(C(y)) = (C(\psi(x)) \underline{\mod} f(x)) \mod 2$, where $\psi(x)$ and $f(x)$ are part of the secret key. We first show that the attacker can determine $\psi(x)$ by asking for polynomially many decryptions of some chosen ciphertexts. All ciphertexts will take the form $C(y) = ky$, for selected choices of $k \in \mathbb{F}_q$. Note that if the attacker only asks for decryptions of linear polynomials, i.e., $C(\psi(x)) = k\psi(x)$, no reduction modulo $f(x)$ will occur. Hence we can disregard $f(x)$ in the decryption function, and the decryption oracle will simply output $k\psi(x) \mod 2$ for all ciphertexts the attacker asks for.

Let $\psi(x) = a_{n-1}x^{n-1} + \ldots + a_1 x + a_0$ with $a_i \in \mathbb{F}_q$. The decryption oracle will output $(ka_{n-1} \mod 2)x^{n-1} + \ldots + (ka_0 \mod 2)$ for the chosen ciphertexts $C(y) = ky$. By focusing on one coefficient a_i at a time we can now use the method described at the start of this subsection to recover all $a_0, \ldots, a_{n-1}$ and hence the exact $\psi(x)$, by asking for $O(n \log q)$ decryptions.

Finally, given $\psi(x)$ and the public polynomial $F(y)$, the secret polynomial $f(x)$ can easily be recovered as follows. We know that $y \in \mathbb{Y}$ and $\psi(x) \in \mathbb{X}$ are two different representations of the same element of $\mathbb{F}_q$. This element is defined by being a root of the polynomial F. Hence we have $G(x) := F(\psi(x)) = 0$, where $\deg(G(x)) \leq n(n-1)$ and x is the element of $\mathbb{X}$ defined by being a root

Generic Construction	Instantiation	Notes
*HE + PA-1 [7,60]	GH-variant of SV + lattice knowledge assumption [60]	SV now insecure; PA-1 uses non-falsifiable assumption
Multi-key IBHE [12,22]	Multi-key *HE + IBE [16]	Only compact w.r.t. circuit complexity
	SubExp LWE + random oracle [33]	Only compact w.r.t. circuit complexity
	SubExp iO + SubExp DDH [22,78]	SubExp iO is a very strong assumption
FHE + zk-SNARK [64,22]	FHE without bootstrapping + knowledge assumptions [22]	FHE without bootstrapping currently only known using SubExp iO

Table 3. Generic constructions of IND-CCA1 *HE. The first construction has an insecure instantiation, while the other constructions only have a generic instantiation. Hence, none of these generic strategies provide a concrete instantiation.

of f. Therefore $f(x)|G(x)$, and we can find $f(x)$ by factoring $G(x)$ and seeing which of the irreducible factors has the particular property of having degree n and only small coefficients. Factoring a univariate polynomial of degree d over $\mathbb{F}_q$ has complexity $O(d^2 \log q)$ (or even smaller, see [53]), so the complexity of recovering $f(x)$ in this step is at most $O(n^4 \log q)$.

6 Generic constructions of IND-CCA1 secure *HE

We present here the various more generic approaches for constructing an IND-CCA1 secure *HE scheme, see Table 3 for a summary. The constructions apply existing generic constructions from group-homomorphic cryptosystems to the *HE setting and provide novel (generic) instantiations. The main problem with these constructions is that the resulting schemes are typically not compact and are rather impractical. Therefore, none of the generic constructions has, to the best of our knowledge, been implemented.

6.1 LMSV

The SHE scheme presented by Loftus et al. [60] was for a long time the only *HE scheme thought to be IND-CCA1 secure, but it has since been broken.

The strategy was to construct an SHE scheme that achieves both IND-CPA and plaintext awareness (PA-1), which is known to result in an IND-CCA1 secure scheme [7]. Informally, plaintext awareness states that if an adversary is able to construct a valid ciphertext, she already knows the plaintext it encrypts. Then, intuitively, the decryption oracle is of no use to the adversary in an IND-CCA1 game, as she must already know the plaintext of any ciphertext she queries. In

particular, the adversary is unable to query the decryption oracle with adulterine ciphertexts designed specifically to reveal information about the secret key, seeing as she must already know the secret key to be able to construct such valid ciphertexts. Hence, IND-CPA results in IND-CCA1 security. We state the formal definition of the PA-1 notion in Appendix B, and refer to [7] for further details on both PA-1 and the construction IND-CPA + PA-1 $\rightarrow$ IND-CCA1.

The LMSV scheme added a ciphertext check to the decryption procedure of an IND-CPA secure *HE scheme. The check aims to ensure that only honestly generated ciphertexts are decrypted, as this ensures that the scheme also achieves PA-1 security. The ciphertext check is based on a novel lattice knowledge assumption. Informally, this lattice knowledge assumption states that if an adversary is able to produce a vector c suitably close to a lattice point p when given only the basis of the lattice, then there is an extractor able to output p, given c and the random coins of the adversary. In other words, if an adversary is able to construct a vector sufficiently close to a lattice point, she must already know that lattice point. In the security proof, ciphertexts are likened to the vector c output by an adversary against the lattice knowledge assumption.

The starting scheme of LMSV, SV [72], based its IND-CPA security on SPIP, which was later broken [10,11,38], meaning that LMSV is not IND-CPA and hence not IND-CCA1 secure. Note, however, that the lattice knowledge assumption is unaffected by the attacks on SPIP. Therefore, it is conceivable to rely on this assumption to create a different IND-CCA1 secure scheme, but such a scheme would have to avoid relying on SPIP for IND-CPA security. It should also be noted that the lattice knowledge assumption is highly nonstandard and not well studied.

Despite these drawbacks, the approach of 'adding' PA-1 security to an SHE scheme that is IND-CPA secure is in and of itself both sound and interesting. However, it appears to be challenging to achieve both of these notions for SHE schemes, particularly PA-1, seeing as the LMSV scheme is the only one to suggest such a construction.

6.2 Constructions from Multi-key Identity-Based Encryption

Canetti et al. [22] give a construction for IND-CCA1 secure *HE schemes based on multi-key identity-based homomorphic encryption (IBHE) schemes. The construction is a simple transformation as follows:

KeyGen: Same as for the multi-key IBHE scheme. The secret key is the master secret key msk, and the public key is the master public key mpk.
Enc(mpk, m): Sample a random identity id, compute $c = \text{Enc}_{\text{IBHE}}(mpk, id, m)$, and output (c, id).
Dec$(msk, (\boldsymbol{c}, id))$: Parse $\boldsymbol{c} = (c, id)$, compute $sk_{id} = \text{Ext}(id, msk)$, and output $m = \text{Dec}_{\text{IBHE}}(sk_{id}, id, c)$.
Eval: Uses the IBHE evaluation function.

The IND-CCA1 security of the encryption scheme rests on the selective security for random identities of the multi-key IBHE scheme, which, informally,

ensures that an adversary has a negligible advantage of distinguishing between encryptions of two messages of her choosing under a random identity, even if she has access to an oracle which provides her with the decryption key of any identity of her choosing. For a formal definition of the security notion, the interested reader is referred to Canetti et al. [22].

It is important to note that for an evaluated ciphertext to be decrypted, it will need to contain the identities of *all* the ciphertexts used in the evaluation. This means that the length of an evaluated ciphertext depends on the number of input ciphertexts, so the resulting scheme is not compact. We stress that this entails that the construction results in a scheme that, technically speaking, does not satisfy the definition of *HE schemes.

The authors also give a generic instantiation that achieves multi-key IBHE by combining a multi-key FHE scheme and an identity-based encryption scheme, and also suggest concrete schemes as building blocks. They also provide a generic instantiation that achieves multi-key IBHE in the random oracle model using sub-exponentially secure LWE.

In a concurrent work, Yasuda et al. [79] use the same generic construction based on multi-key IBHE to achieve a non-compact IND-CCA1 secure SHE scheme, with a similar proof for IND-CCA1 security as the one described above. A concrete instantiation for this construction has not been suggested.

6.3 (Probabilistic) iO-based

As discussed previously, Canetti et al. [22] showed that it is possible to construct a (non-compact) *HE scheme from a multi-key IBHE scheme. In the same article, the authors also suggested constructing a multi-key IBHE from a sub-exponentially secure iO (from which a probabilistic iO is constructed) and sub-exponentially secure lossy encryption (which can be based on DDH).

We note that Wang et al. found a technical weakness regarding how identities were handled in evaluations, which enabled an attack on the PiO-based construction. However, Wang et al. also provided a patched version of the construction and proved that their construction achieves IND-CCA1 security [78].

The constructions differ in certain aspects, such as the set-up and generation of keys for identities. Still, there are also important similarities, such as both constructions using a trapdoor encryption scheme during the encryption procedure of the IB *HE scheme. The most important common factor is that the evaluation hinges on PiO: the circuit given as input is parsed as an algebraic circuit with separate addition and multiplication gates, which are then evaluated using obfuscations of different probabilistic programs. For further details on either construction, we refer to the respective article. Neither construction currently has a concrete instantiation.

6.4 zk-SNARK construction

Canetti et al. [22] also gave a generic construction of IND-CCA1 secure FHE from IND-CPA secure FHE using the Naor-Yung paradigm [64]. Essentially, the

public key consists of two different public keys pk_1, pk_2 of the same FHE scheme along with a common reference string crs for the zk-SNARK. An encryption of a message m is then of the form $(\text{Enc}(pk_1, m), \text{Enc}(pk_2, m), \pi)$, where π is a proof of knowledge that the first two elements encrypt the same message m. Intuitively, this construction ensures IND-CCA1 security because the only way an adversary would be able to construct a ciphertext of a message the decryption oracle accepts is to actually encrypt the message, as she should not be able to construct a valid proof π otherwise. Thus, the decryption oracle cannot reveal to the adversary anything which she does not already know.

Note here that the Canetti et al. construction requires a zk-SNARK, instead of the usual NIZK used in the general Naor-Yung paradigm, to ensure π stays compact and hence preserves the compactness of the resulting IND-CCA1 secure FHE. However, due to the black-box separation of SNARKs and falsifiable assumptions [49], IND-CCA1 security would then require a non-falsifiable assumption. Also, note that the construction requires an IND-CPA secure FHE scheme without bootstrapping or key publication. It is not known whether or not a scheme with such properties is possible under standard assumptions.

7 Discussion

We have given an overview of the state of IND-CCA1 security for *HE schemes, both for concrete schemes and generic constructions. We have shown that several schemes are susceptible to the same adaptive key recovery attacks, mainly because many schemes are optimisations of earlier work, so attacks carry over more or less trivially. We also presented new adaptive key recovery attacks against schemes that had not been studied w.r.t. IND-CCA1 security before now.

It is worth discussing why chosen ciphertext attacks are so devastating against *HE schemes. We repeat a point made by Chillotti et al. [32], namely that if the proof of IND-CPA relies on a search-to-decision reduction of a problem, an adversary with access to a decryption oracle may simply follow the steps of this very reduction to recover the secret key. Most *HE schemes do rely on such a reduction, particularly all schemes based on LWE and RLWE.

Moreover, the requirement of both addition and multiplication being homomorphic might make achieving IND-CCA1 harder, as there are few mathematical structures to define encryption schemes over that allow for a more or less natural homomorphic evaluation of ciphertexts. It could be the case that these structures themselves complicate achieving IND-CCA1 security. The result by Gjøsteen and Strand showing that secure noiseless schemes cannot exist in vector spaces or fields might suggest that this is the case [50]. Furthermore, there are group homomorphic schemes that achieve IND-CCA1 security, e.g., CS-lite [39], and although there is no proof that it cannot be homomorphic in both multiplication and addition, CS-lite is strongly believed to be strictly group homomorphic. It could be the case that requiring both operations to be homomorphic forces the message or secret key to be so 'accessible' in the ciphertext that some information is always leaked once an adversary has access to a decryption oracle.

An important point in this discussion is that all the adaptive key recovery attacks we presented, both novel and prior work, take advantage of the fact that decryption is a relatively 'easy' procedure, where the secret key is typically just multiplied with the ciphertext or it is applied in a modular reduction. This is a stark contrast to, e.g., block ciphers where the secret key is carefully scrambled, and changing one bit of the ciphertext will result in an avalanche effect so that several bits of the decrypted ciphertext are changed. A naive remedy would be to create a scheme with a more 'convoluted' decryption procedure. However, bootstrapping a scheme hinges both on circular security *and* on the decryption circuit being as easy as possible: if this is not the case, the homomorphic evaluation of the decryption circuit will not reduce the noise in the ciphertext sufficiently to allow for further evaluations of it. In other words, the scheme would not be *fully* homomorphic. The easy decryption procedure is therefore inherent in all bootstrappable schemes. Furthermore, all suggested SHE and LHE schemes have been designed to be bootstrapped so that they may be expanded to an FHE scheme, meaning they all have a shallow decryption circuit. The overview we have provided strongly suggests that this approach is not compatible with providing security against adaptive key recovery attacks or IND-CCA1 security.

We repeat the point from Zhang et al. [81], namely that as the use cases of *HE schemes increase the probability of leakage of decrypted material, IND-CCA1 is an essential requirement for a secure homomorphic encryption scheme. We believe that a good starting point for creating an *HE scheme which achieves IND-CCA1 security would be the generic construction using zk-SNARKs by Canetti et al. [22]. As mentioned in Section 6.4, the construction relies on non-falsifiable assumptions. However, as the authors noted, one may use weaker primitives such as designated-verifier zk-SNARKs, and it is an interesting open problem to determine the minimum flavour of zero-knowledge that is needed to get IND-CCA1 security using the Canetti et al. construction. Alternatively, developing pure SHE or LHE schemes that are *not* designed to be bootstrappable might be a fruitful strategy, as this could allow for a decryption procedure 'convoluted' enough to result in IND-CCA1 security or protection against adaptive key recovery attacks.

References

1. Privacy enhancing technologies. `https://csrc.nist.gov/Projects/pec`, accessed: 23. September 2021
2. Albrecht, M.R., Bai, S., Ducas, L.: A subfield lattice attack on overstretched NTRU assumptions - cryptanalysis of some FHE and graded encoding schemes. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016, Part I. LNCS, vol. 9814, pp. 153–178. Springer, Heidelberg (Aug 2016)
3. Albrecht, M.R., Farshim, P., Faugère, J.C., Perret, L.: Polly cracker, revisited. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 179–196. Springer, Heidelberg (Dec 2011)
4. Arita, S., Handa, S.: Subring homomorphic encryption. In: Kim, H., Kim, D.C. (eds.) ICISC 17. LNCS, vol. 10779, pp. 112–136. Springer, Heidelberg (Nov / Dec 2018)

5. Arita, S., Nakasato, S.: Fully homomorphic encryption for point numbers. In: International Conference on Information Security and Cryptology. pp. 253–270. Springer (2016)
6. Armknecht, F., Boyd, C., Carr, C., Gjøsteen, K., Jäschke, A., Reuter, C.A., Strand, M.: A guide to fully homomorphic encryption. Cryptology ePrint Archive, Report 2015/1192 (2015), http://eprint.iacr.org/2015/1192
7. Bellare, M., Palacio, A.: Towards plaintext-aware public-key encryption without random oracles. In: Lee, P.J. (ed.) ASIACRYPT 2004. LNCS, vol. 3329, pp. 48–62. Springer, Heidelberg (Dec 2004)
8. Benarroch, D., Brakerski, Z., Lepoint, T.: FHE over the integers: Decomposed and batched in the post-quantum regime. In: Fehr, S. (ed.) PKC 2017, Part II. LNCS, vol. 10175, pp. 271–301. Springer, Heidelberg (Mar 2017)
9. Berkoff, A., Liu, F.H.: Leakage resilient fully homomorphic encryption. In: Lindell, Y. (ed.) TCC 2014. LNCS, vol. 8349, pp. 515–539. Springer, Heidelberg (Feb 2014)
10. Biasse, J.F., Fieker, C.: Subexponential class group and unit group computation in large degree number fields. LMS Journal of Computation and Mathematics 17(A), 385–403 (2014)
11. Biasse, J.F., Song, F.: Efficient quantum algorithms for computing class groups and solving the principal ideal problem in arbitrary degree number fields, pp. 893–902 (2016), https://epubs.siam.org/doi/abs/10.1137/1.9781611974331.ch64
12. Boneh, D., Canetti, R., Halevi, S., Katz, J.: Chosen-ciphertext security from identity-based encryption. SIAM Journal on Computing 36(5), 1301–1328 (2007)
13. Bootland, C., Castryck, W., Iliashenko, I., Vercauteren, F.: Efficiently processing complex-valued data in homomorphic encryption. Journal of Mathematical Cryptology 14(1), 55–65 (2020)
14. Bos, J.W., Lauter, K., Loftus, J., Naehrig, M.: Improved security for a ring-based fully homomorphic encryption scheme. In: Stam, M. (ed.) 14th IMA International Conference on Cryptography and Coding. LNCS, vol. 8308, pp. 45–64. Springer, Heidelberg (Dec 2013)
15. Brakerski, Z.: Fully homomorphic encryption without modulus switching from classical GapSVP. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 868–886. Springer, Heidelberg (Aug 2012)
16. Brakerski, Z., Cash, D., Tsabary, R., Wee, H.: Targeted homomorphic attribute based encryption. Cryptology ePrint Archive, Report 2016/691 (2016), http://eprint.iacr.org/2016/691
17. Brakerski, Z., Gentry, C., Vaikuntanathan, V.: (Leveled) fully homomorphic encryption without bootstrapping. In: Goldwasser, S. (ed.) ITCS 2012. pp. 309–325. ACM (Jan 2012)
18. Brakerski, Z., Perlman, R.: Lattice-based fully dynamic multi-key FHE with short ciphertexts. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016, Part I. LNCS, vol. 9814, pp. 190–213. Springer, Heidelberg (Aug 2016)
19. Brakerski, Z., Vaikuntanathan, V.: Efficient fully homomorphic encryption from (standard) LWE. In: Ostrovsky, R. (ed.) 52nd FOCS. pp. 97–106. IEEE Computer Society Press (Oct 2011)
20. Brakerski, Z., Vaikuntanathan, V.: Fully homomorphic encryption from ring-LWE and security for key dependent messages. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 505–524. Springer, Heidelberg (Aug 2011)
21. Brakerski, Z., Vaikuntanathan, V.: Lattice-based FHE as secure as PKE. In: Naor, M. (ed.) ITCS 2014. pp. 1–12. ACM (Jan 2014)

22. Canetti, R., Raghuraman, S., Richelson, S., Vaikuntanathan, V.: Chosen-ciphertext secure fully homomorphic encryption. In: Fehr, S. (ed.) PKC 2017, Part II. LNCS, vol. 10175, pp. 213–240. Springer, Heidelberg (Mar 2017)
23. Chen, H., Chillotti, I., Song, Y.: Multi-key homomorphic encryption from TFHE. In: Galbraith, S.D., Moriai, S. (eds.) ASIACRYPT 2019, Part II. LNCS, vol. 11922, pp. 446–472. Springer, Heidelberg (Dec 2019)
24. Chen, H., Iliashenko, I., Laine, K.: When heaan meets fv: a new somewhat homomorphic encryption with reduced memory overhead. IACR Cryptol. ePrint Arch. 2020, 121 (2020)
25. Chen, H., Laine, K., Player, R., Xia, Y.: High-precision arithmetic in homomorphic encryption. In: Smart, N.P. (ed.) CT-RSA 2018. LNCS, vol. 10808, pp. 116–136. Springer, Heidelberg (Apr 2018)
26. Chenal, M., Tang, Q.: On key recovery attacks against existing somewhat homomorphic encryption schemes. In: Aranha, D.F., Menezes, A. (eds.) LATINCRYPT 2014. LNCS, vol. 8895, pp. 239–258. Springer, Heidelberg (Sep 2015)
27. Cheon, J.H., Coron, J.S., Kim, J., Lee, M.S., Lepoint, T., Tibouchi, M., Yun, A.: Batch fully homomorphic encryption over the integers. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 315–335. Springer, Heidelberg (May 2013)
28. Cheon, J.H., Hong, S., Kim, D.: Remark on the security of ckks scheme in practice. Cryptology ePrint Archive, Report 2020/1581 (2020), `https://eprint.iacr.org/2020/1581`
29. Cheon, J.H., Kim, A., Kim, M., Song, Y.S.: Homomorphic encryption for arithmetic of approximate numbers. In: Takagi, T., Peyrin, T. (eds.) ASIACRYPT 2017, Part I. LNCS, vol. 10624, pp. 409–437. Springer, Heidelberg (Dec 2017)
30. Cheon, J.H., Stehlé, D.: Fully homomophic encryption over the integers revisited. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015, Part I. LNCS, vol. 9056, pp. 513–536. Springer, Heidelberg (Apr 2015)
31. Chillotti, I., Gama, N., Georgieva, M., Izabachène, M.: TFHE: Fast fully homomorphic encryption over the torus. Journal of Cryptology 33(1), 34–91 (Jan 2020)
32. Chillotti, I., Gama, N., Goubin, L.: Attacking FHE-based applications by software fault injections. Cryptology ePrint Archive, Report 2016/1164 (2016), `http://eprint.iacr.org/2016/1164`
33. Clear, M., McGoldrick, C.: Multi-identity and multi-key leveled FHE from learning with errors. In: Gennaro, R., Robshaw, M.J.B. (eds.) CRYPTO 2015, Part II. LNCS, vol. 9216, pp. 630–656. Springer, Heidelberg (Aug 2015)
34. Coron, J.S., Lepoint, T., Tibouchi, M.: Scale-invariant fully homomorphic encryption over the integers. In: Krawczyk, H. (ed.) PKC 2014. LNCS, vol. 8383, pp. 311–328. Springer, Heidelberg (Mar 2014)
35. Coron, J.S., Mandal, A., Naccache, D., Tibouchi, M.: Fully homomorphic encryption over the integers with shorter public keys. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 487–504. Springer, Heidelberg (Aug 2011)
36. Coron, J.S., Naccache, D., Tibouchi, M.: Public key compression and modulus switching for fully homomorphic encryption over the integers. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 446–464. Springer, Heidelberg (Apr 2012)
37. Costache, A., Smart, N.P.: Homomorphic encryption without Gaussian noise. Cryptology ePrint Archive, Report 2017/163 (2017), `http://eprint.iacr.org/2017/163`

38. Cramer, R., Ducas, L., Peikert, C., Regev, O.: Recovering short generators of principal ideals in cyclotomic rings. In: Fischlin, M., Coron, J.S. (eds.) EUROCRYPT 2016, Part II. LNCS, vol. 9666, pp. 559–585. Springer, Heidelberg (May 2016)
39. Cramer, R., Shoup, V.: A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In: Krawczyk, H. (ed.) CRYPTO'98. LNCS, vol. 1462, pp. 13–25. Springer, Heidelberg (Aug 1998)
40. Dahab, R., Galbraith, S., Morais, E.: Adaptive key recovery attacks on NTRU-based somewhat homomorphic encryption schemes. In: Lehmann, A., Wolf, S. (eds.) ICITS 15. LNCS, vol. 9063, pp. 283–296. Springer, Heidelberg (May 2015)
41. Doröz, Y., Hoffstein, J., Pipher, J., Silverman, J.H., Sunar, B., Whyte, W., Zhang, Z.: Fully homomorphic encryption from the finite field isomorphism problem. In: Abdalla, M., Dahab, R. (eds.) PKC 2018, Part I. LNCS, vol. 10769, pp. 125–155. Springer, Heidelberg (Mar 2018)
42. ElGamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. In: Blakley, G.R., Chaum, D. (eds.) CRYPTO'84. LNCS, vol. 196, pp. 10–18. Springer, Heidelberg (Aug 1984)
43. Fan, J., Vercauteren, F.: Somewhat practical fully homomorphic encryption. Cryptology ePrint Archive, Report 2012/144 (2012), http://eprint.iacr.org/2012/144
44. Fauzi, P., Hovd, M.N., Raddum, H.: A practical adaptive key recovery attack on the lgm (gsw-like) cryptosystem. Cryptology ePrint Archive, Report 2021/658 (2021), https://eprint.iacr.org/2021/658
45. Fauzi, P., Hovd, M.N., Raddum, H.: A practical adaptive key recovery attack on the lgm (gsw-like) cryptosystem. In: Cheon, J.H., Tillich, J.P. (eds.) PQCRYPTO 2021. pp. 483–498. Springer, Cham (July 2021)
46. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Mitzenmacher, M. (ed.) 41st ACM STOC. pp. 169–178. ACM Press (May / Jun 2009)
47. Gentry, C., Halevi, S.: Implementing Gentry's fully-homomorphic encryption scheme. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 129–148. Springer, Heidelberg (May 2011)
48. Gentry, C., Sahai, A., Waters, B.: Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part I. LNCS, vol. 8042, pp. 75–92. Springer, Heidelberg (Aug 2013)
49. Gentry, C., Wichs, D.: Separating succinct non-interactive arguments from all falsifiable assumptions. In: Fortnow, L., Vadhan, S.P. (eds.) 43rd ACM STOC. pp. 99–108. ACM Press (Jun 2011)
50. Gjøsteen, K., Strand, M.: Fully homomorphic encryption must be fat or ugly? Cryptology ePrint Archive, Report 2016/105 (2016), http://eprint.iacr.org/2016/105
51. Hovd, M.N.: A successful subfield lattice attack on a fully homomorphic encryption scheme. In: Proceedings of the 11th Norwegian Information Security Conference (2018)
52. Joux, A.: Fully homomorphic encryption modulo Fermat numbers. Cryptology ePrint Archive, Report 2019/187 (2019), https://eprint.iacr.org/2019/187
53. Kedlaya, K.S., Umans, C.: Fast polynomial factorization and modular composition. SIAM Journal on Computing 40(6), 1767–1802 (2011)
54. Kim, J., Lee, M.S., Yun, A., Cheon, J.H.: CRT-based fully homomorphic encryption over the integers. Cryptology ePrint Archive, Report 2013/057 (2013), http://eprint.iacr.org/2013/057

55. Lai, J., Deng, R.H., Ma, C., Sakurai, K., Weng, J.: CCA-secure keyed-fully homomorphic encryption. In: Cheng, C.M., Chung, K.M., Persiano, G., Yang, B.Y. (eds.) PKC 2016, Part I. LNCS, vol. 9614, pp. 70–98. Springer, Heidelberg (Mar 2016)
56. Laine, K.: Updates on iso/iec standardization. Email sent to the mailing list standards@homomorphicencryption.org 15. September 2021
57. Li, B., Micciancio, D.: On the security of homomorphic encryption on approximate numbers. In: Canteaut, A., Standaert, F.X. (eds.) EUROCRYPT 2021. LNCS, vol. 12696, pp. 648–677. Springer, Cham (2021)
58. Li, Z., Galbraith, S.D., Ma, C.: Preventing adaptive key recovery attacks on the gentry-sahai-waters leveled homomorphic encryption scheme. Cryptology ePrint Archive, Report 2016/1146 (2016), http://eprint.iacr.org/2016/1146
59. Li, Z., Galbraith, S.D., Ma, C.: Preventing adaptive key recovery attacks on the GSW levelled homomorphic encryption scheme. In: Chen, L., Han, J. (eds.) ProvSec 2016. LNCS, vol. 10005, pp. 373–383. Springer, Heidelberg (Nov 2016)
60. Loftus, J., May, A., Smart, N.P., Vercauteren, F.: On CCA-secure somewhat homomorphic encryption. In: Miri, A., Vaudenay, S. (eds.) SAC 2011. LNCS, vol. 7118, pp. 55–72. Springer, Heidelberg (Aug 2012)
61. López-Alt, A., Tromer, E., Vaikuntanathan, V.: On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In: Karloff, H.J., Pitassi, T. (eds.) 44th ACM STOC. pp. 1219–1234. ACM Press (May 2012)
62. Manger, J.: A chosen ciphertext attack on RSA optimal asymmetric encryption padding (OAEP) as standardized in PKCS #1 v2.0. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 230–238. Springer, Heidelberg (Aug 2001)
63. Masters, O., Hunt, H., Steffinlongo, E., Crawford, J., Bergamaschi, F., Rosa, M.E.D., Quini, C.C., Alves, C.T., de Souza, F., Ferreira, D.G.: Towards a homomorphic machine learning big data pipeline for the financial services sector. Cryptology ePrint Archive, Report 2019/1113 (2019), https://eprint.iacr.org/2019/1113
64. Naor, M., Yung, M.: Public-key cryptosystems provably secure against chosen ciphertext attacks. In: 22nd ACM STOC. pp. 427–437. ACM Press (May 1990)
65. Nuida, K.: Candidate constructions of fully homomorphic encryption on finite simple groups without ciphertext noise. Cryptology ePrint Archive, Report 2014/097 (2014), http://eprint.iacr.org/2014/097
66. Peikert, C., Shiehian, S.: Multi-key FHE from LWE, revisited. In: Hirt, M., Smith, A.D. (eds.) TCC 2016-B, Part II. LNCS, vol. 9986, pp. 217–238. Springer, Heidelberg (Oct / Nov 2016)
67. Peng, Z.: Danger of using fully homomorphic encryption: A look at microsoft SEAL. CoRR abs/1906.07127 (2019), http://arxiv.org/abs/1906.07127
68. Pereira, H.V.L.: Efficient agcd-based homomorphic encryption for matrix and vector arithmetic. In: International Conference on Applied Cryptography and Network Security. pp. 110–129. Springer (2020)
69. Rivest, R.L., Adleman, L., Dertouzos, M.L., et al.: On data banks and privacy homomorphisms. Foundations of secure computation 4(11), 169–180 (1978)
70. Rivest, R.L., Shamir, A., Adleman, L.M.: A method for obtaining digital signatures and public-key cryptosystems. Communications of the Association for Computing Machinery 21(2), 120–126 (1978)
71. Rohloff, K., Cousins, D.B.: A scalable implementation of fully homomorphic encryption built on NTRU. In: Böhme, R., Brenner, M., Moore, T., Smith, M. (eds.) FC 2014 Workshops. LNCS, vol. 8438, pp. 221–234. Springer, Heidelberg (Mar 2014)

72. Smart, N.P., Vercauteren, F.: Fully homomorphic encryption with relatively small key and ciphertext sizes. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 420–443. Springer, Heidelberg (May 2010)
73. Smart, N.P., Vercauteren, F.: Fully homomorphic simd operations. Designs, codes and cryptography 71(1), 57–81 (2014)
74. Stehlé, D., Steinfeld, R.: Faster fully homomorphic encryption. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 377–394. Springer, Heidelberg (Dec 2010)
75. Stehlé, D., Steinfeld, R.: Making NTRU as secure as worst-case problems over ideal lattices. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 27–47. Springer, Heidelberg (May 2011)
76. van Dijk, M., Gentry, C., Halevi, S., Vaikuntanathan, V.: Fully homomorphic encryption over the integers. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 24–43. Springer, Heidelberg (May / Jun 2010)
77. Vaudenay, S.: Security flaws induced by CBC padding - applications to SSL, IPSEC, WTLS... In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 534–546. Springer, Heidelberg (Apr / May 2002)
78. Wang, B., Wang, X., Xue, R.: CCA1 secure FHE from pio, revisited. Cybersecurity 1(1), 11 (2018), https://doi.org/10.1186/s42400-018-0013-8
79. Yasuda, S., Kitagawa, F., Tanaka, K.: Constructions for the IND-CCA1 secure fully homomorphic encryption. In: Mathematical Modelling for Next-Generation Cryptography: CREST Crypto-Math Project, pp. 331–347 (2017)
80. Zhang, Z., Plantard, T., Susilo, W.: On the cca-1 security of somewhat homomorphic encryption over the integers. pp. 353–368 (04 2012)
81. Zhang, Z., Plantard, T., Susilo, W.: Reaction attack on outsourced computing with fully homomorphic encryption schemes. In: Kim, H. (ed.) ICISC 11. LNCS, vol. 7259, pp. 419–436. Springer, Heidelberg (Nov / Dec 2012)

A Background

We give a more thorough explanation of homomorphic encryption, for more details, see the Guide to FHE by Armknecht et al. [6].

Group Homomorphic Encryption (GHE) refers to schemes that are homomorphic in a single group operation, i.e., $\mathrm{Dec}(\mathrm{Enc}(m_1) \cdot \mathrm{Enc}(m_2)) = m_1 \circ m_2$, where $\cdot$ is a binary operation in the ciphertext space, and $\circ$ is a binary operation in the plaintext space. If the operation is addition, the scheme is called *additively homomorphic encryption* (AHE), and similarly *multiplicatively homomorphic encryption* (MHE) if the operation is multiplication.

Examples of GHE schemes are RSA [70] and ElGamal [42]. It is worth noting that for these schemes, there is no limit to how much a ciphertext may be evaluated with respect to the homomorphic operation. The product of two (decryptable) RSA ciphertexts will always result in a decryptable ciphertext.

This is not the case when the scheme is homomorphic with respect to both addition and multiplication, as these are typically limited in what circuits they are able to evaluate homomorphically. This is because all suggested *HE schemes employ noise to encrypt ciphertexts, which grows during evaluation. Typically, the growth is substantially bigger during multiplication than for addition. Unlike in the GHE case, there is a risk of decryption error if ciphertexts are processed

too much, as the plaintext may 'drown' in too much noise. At this point, there is no guarantee that the message output by the decryption is indeed the correct message, i.e., there is a non-negligible probability that $\mathrm{Dec}(C(\mathrm{Enc}(m))) \neq C(m)$ for some circuit C and message m.

The key difference between the various types of homomorphic encryption is how much control they have over the noise growth, and the limits on the circuits they are able to evaluate correctly. We define an encryption scheme $\mathcal{E} =$ (KeyGen, Enc, Dec, Eval) to evaluate a circuit C correctly if

$$\Pr[\mathrm{Dec}(sk, \mathrm{Eval}(evk, C, c_1, \ldots, c_n)) = C(m_1, \ldots, m_n)] = 1 - \mathrm{negl}(\lambda),$$

where $(sk, pk, evk) \leftarrow \mathrm{KeyGen}(\lambda)$, and $c_i \leftarrow \mathrm{Enc}(pk, m_i)$. We allow for a negligible probability of decryption error.

Somewhat Homomorphic Encryption (SHE) refers to schemes that are homomorphic in both addition and multiplication, but may only perform a limited number of these operations before the noise becomes unmanageable. There is little to no means of reducing the noise in these schemes, only stunting the growth. Furthermore, the noise growth is not predictable enough to estimate precisely when the noise will become unmanageable. The number of operations performed on ciphertexts is therefore limited, but the limit may not be explicitly set. More formally, an SHE scheme correctly evaluates some circuits, though there is no formal requirement as to what schemes it is able to evaluate correctly.

Leveled Homomorphic Encryption (LHE) schemes are similar to SHE schemes in that these schemes also only allow for a limited amount of additions and multiplications. However, unlike SHEs, for leveled schemes the number of operations that can be performed before decryption could fail may be explicitly set. The desired level is a separate parameter, L, in the key generation, and corresponds to the maximum depth of an arithmetic circuit C the scheme is able to evaluate correctly.

Fully Homomorphic Encryption (FHE) schemes are able to evaluate *any* arithmetic circuit correctly.

So far, the only known way to achieve a scheme which is fully homomorphic is to apply bootstrapping to an SHE or LHE scheme. Bootstrapping entails evaluating the decryption circuit homomorphically. This removes all the 'old noise' built up in the ciphertext during evaluation, but introduces more noise in the process of the homomorphic evaluation of the decryption circuit. However, as long as the noise introduced is small enough to still allow for just a single additional homomorphic operation, it is possible to perform any number of operations, and therefore evaluate any circuit correctly.

It is required that LHE and FHE schemes also achieve compactness, which we define as follows:

Definition 1 (Compactness). *A scheme is* compact *if there is a polynomial p such that for all ciphertexts c_i, and all circuits C the scheme is able to evaluate correctly, it is the case that* $|\mathrm{Eval}(C, c_0, c_1, \ldots, c_n)| < p(\lambda)$.

In other words: the size of any evaluated ciphertext is independent on the size of the circuit. In the case of LHE, we also require that the length of the evaluation output is independent of the level parameter L [6].

Whilst compactness is a requirement for both LHE and FHE schemes, this is not necessarily the case for SHE schemes. However, it is preferable for these schemes to be compact as well, as it implies a somewhat stunted and controlled noise growth.

B Plaintext Awareness

This section is based on Section 5 of [60].

Let the polynomial time adversary $\mathbb{A}$ be the *ciphertext creator*, which takes a public key as input, and may query ciphertexts to an oracle. Then, an algorithm $\mathbb{A}^*$ is called a *successful extractor* for $\mathbb{A}$ if it can provide responses to $\mathbb{A}$ which are computationally indistinguishable from those provided by the decryption oracle. A scheme is said to be PA-1 if there exists a successful extractor for any ciphertext creator that makes a polynomial number of queries. The extractor $\mathbb{A}^*$ gets the same public key as $\mathbb{A}$, and also has access to the random coins used by $\mathbb{A}$. We define it formally as follows:

Definition 2 (PA-1). *Let $\mathcal{E}$ be a public key encryption scheme, and $\mathbb{A}$ be an algorithm with access to an oracle $\mathcal{O}$ taking input pk and returning a string. Let $\mathcal{D}$ be an algorithm that takes as input a string and returns a single bit, and let $\mathbb{A}^*$ be an algorithm which takes as input a string and some state information and returns either a string or the symbol $\perp$, plus a new state. We call $\mathbb{A}$ a* ciphertext creator*, $\mathbb{A}^*$ a* PA-1 extractor*, and $\mathcal{D}$ a* distinguisher*. For security parameter λ we define the (distinguishing and extracting) experiments below, and define the PA-1 advantage as:*

$$\mathrm{Adv}^{PA-1}_{\mathcal{E},\mathbb{A},\mathcal{D},\mathbb{A}^*}(\lambda) = |\Pr(\mathrm{Exp}^{PA-1-d}_{\mathcal{E},\mathbb{A},\mathcal{D}}(\lambda) = 1) - \Pr(\mathrm{Exp}^{PA-1-x}_{\mathcal{E},\mathbb{A},\mathcal{D},\mathbb{A}^*}(\lambda) = 1)|.$$

We say $\mathbb{A}^*$ is a successful PA-1 extractor for $\mathbb{A}$ if for every polynomial time distinguisher the above advantage is negligible.

$\mathrm{Exp}^{PA-1-d}_{\mathcal{E},\mathbb{A},\mathcal{D}}(\lambda)$
$(pk, sk) \leftarrow \mathrm{KeyGen}(\lambda)$
$x \leftarrow \mathbb{A}^{\mathrm{Decrypt}(\cdot,sk)}(pk)$
$d \leftarrow \mathcal{D}(x)$
Return d

$\mathrm{Exp}^{PA-1-x}_{\mathcal{E},\mathbb{A},\mathcal{D},\mathbb{A}^*}(\lambda)$
$(pk, sk) \leftarrow \mathrm{KeyGen}(\lambda)$
Choose coins coins[$\mathbb{A}$] and coins[$\mathbb{A}^*$]
$\mathrm{St} \leftarrow (pk, \mathrm{coins}[\mathbb{A}])$
$x \leftarrow \mathbb{A}^{\mathcal{O}}$, replying to oracle queries
$\mathcal{O}(c)$:
  $(m, \mathrm{St}) \leftarrow \mathbb{A}^*(c, \mathrm{St}; \mathrm{coins}[\mathbb{A}^*])$
  Return m to $\mathbb{A}$
$d \leftarrow \mathcal{D}(x)$
Return d