Studies on the Security of Selected Advanced Asymmetric Cryptographic Primitives

Martha Norberg Hovd

Thesis for the degree of Philosophiae Doctor (PhD)
University of Bergen, Norway
2022

UNIVERSITY OF BERGEN

Studies on the Security of Selected Advanced Asymmetric Cryptographic Primitives

Martha Norberg Hovd

Thesis for the degree of Philosophiae Doctor (PhD)
at the University of Bergen

Date of defense: 11.03.2022

Year: 2022

Title: Studies on the Security of Selected Advanced Asymmetric Cryptographic Primitives

Name: Martha Norberg Hovd

Print: Skipnes Kommunikasjon / University of Bergen

Scientific environment

I have been employed by the University of Bergen for the entire duration of my doctoral studies, and been enrolled as a PhD student at the Department of Informatics. I have been supervised by Håvard Raddum, Martijn Stam, and Øyvind Ytrehus at the research lab Simula UiB, which has also been my workplace. I have also been enrolled in the Research School of Computer and Information Security (COINS).

Acknowledgements

I cannot adequately express the acknowledgement that is due to anyone mentioned herein due to the shortcoming of language, and I cannot mention everyone to be acknowledged due to restraints of space. Nevertheless, I shall try.

I thank my supervisors Håvard, Martijn, and Øyvind for their guidance, challenges, encouragements, advice, discussions, and knowledge they have so generously shared with me.

I thank Prastudy, for his insights, enthusiasm, and laser sharp attention to details in our collaborations. I thank Mohsen, for his initial suggestion of working on the IND-CCA1 security of fully homomorphic encryption schemes, and for being my co-supervisor as I started my PhD.

I thank all the people I have worked with at Simula UiB, for making the whole endeavour so much more enjoyable than I had ever thought possible.

I thank my friends, both at and outside of work, for keeping me grounded, seeing me as a multifaceted human being, and for reminding (and sometimes lovingly forcing) me to make room and take time for the enjoyable aspects of life which exist outside of research; including, but not limited to, conversations (brief, lengthy, and in between), laughter, tea, excellent jokes, chocolate truffles, wine, terrible jokes, books, gin, karaoke, films, and dancing. I thank Marta in particular, my colleague in spirit, among everything else.

Finally, I thank my family, for their unconditional love and support.

Abstract

The main goal of asymmetric cryptography is to provide confidential communication, which allows two parties to communicate securely even in the presence of adversaries. Ever since its invention in the seventies, asymmetric cryptography has been improved and developed further, and a formal security framework has been established around it. This framework includes different security goals, attack models, and security notions. As progress was made in the field, more advanced asymmetric cryptographic primitives were proposed, with other properties in addition to confidentiality. These new primitives also have their own definitions and notions of security.

This thesis consists of two parts, where the first relates to the security of fully homomorphic encryption and related primitives. The second part presents a novel cryptographic primitive, and defines what security goals the primitive should achieve.

The first part of the thesis consists of Article I, II, and III, which all pertain to the security of homomorphic encryption schemes in one respect or another. Article I demonstrates that a particular fully homomorphic encryption scheme is insecure in the sense that an adversary with access only to the public material can recover the secret key. It is also shown that this insecurity mainly stems from the operations necessary to make the scheme fully homomorphic. Article II presents an adaptive key recovery attack on a leveled homomorphic encryption scheme. The scheme in question claimed to withstand precisely such attacks, and was the only scheme of its kind to do so at the time. This part of the thesis culminates with Article III, which is an overview article on the IND-CCA1 security of all acknowledged homomorphic encryption schemes.

The second part of the thesis consists of Article IV, which presents Vetted Encryption (VE), a novel asymmetric cryptographic primitive. The primitive is designed to allow a recipient to vet who may send them messages, by setting up a public filter with a public verification key, and providing each vetted sender with their own encryption key. There are three different variants of VE, based on whether the sender is identifiable to the filter and/or the recipient. Security definitions, general constructions and comparisons to already existing cryptographic primitives are provided for all three variants.

List of publications

1. Martha Norberg Hovd: A Successful Subfield Lattice Attack on a Fully Homomorphic Encryption Scheme. In Stig Frode Mjølsnes and Ragnar Soleng, editors, *Proceedings of the 11th Norwegian Information Security Conference*, September 2018.

2. Prastudy Fauzi, Martha Norberg Hovd and Håvard Raddum. A Practical Adaptive Key Recovery Attack on the LGM (GSW-like) Cryptosystem. In Jung Hee Cheon and Jean-Pierre Tillich, editors, *Post-Quantum Cryptography - 12th International Conference, PQCrypto 2021*, pages 483-498, Springer, Cham, July 2021.

3. Prastudy Fauzi, Martha Norberg Hovd and Håvard Raddum. On the IND-CCA1 Security of FHE Schemes. Cryptology ePrint Archive, Report 2021/1624, 2021. `https://eprint.iacr.org/2021/1624`.

4. Martha Norberg Hovd and Martijn Stam. Vetted Encryption. In Karthikeyan Bhargavan, Elisabeth Oswald and Manoj Prabhakaran, editors, *INDOCRYPT 2020*, volume 12578 of *LNCS*, pages 488-507. Springer, Heidelberg, December 2020.

Contents

Chapter 1

Introduction

1.1 The Ancient Art of Selectively Sharing Secret Messages

Cryptology is, like most terms, derived from Greek: ‘kryptós’ meaning hidden, and ‘logia’ meaning study. Though this small lesson in etymology might lead the mind of the reader to archaeologists studying the long-hidden crypts in ancient Egypt, studying cryptology rarely requires a digging kit. What is hidden is not tombs or treasures covered by the dust of time, but rather *text* hidden by other text.

The art of creating such hidden texts is known as cryptography, where the new postfix is derived from the Greek word ‘graphein’: ‘to write’. Cryptography is used to rewrite a message, or plaintext, as a ciphertext. A particular method for creating a ciphertext given a plaintext is known as a cipher, a cryptosystem, or an encryption scheme, where the two latter are more modern terms.

The earliest known example of cryptography is actually in an ancient Egyptian crypt, where non-standard hieroglyphs were written on the wall of a tomb roughly 4000 years ago. Although this is the earliest example, it is far from the only one, as we know of several other ancient cultures that also used cryptography for various reasons ranging from planning and executing military strategies, preventing corporate espionage, concealing tax records, avoiding persecution, to communicating with secret lovers. Then, as now, there are many types of secrets people have wanted to keep from prying eyes.

There are two basic principles that underlie all ciphers: transposition and substitution. In transposition ciphers, the letters the plaintext consists of are simply shuffled around for the ciphertext construction, meaning the two texts consist of the same letters, but in a different order. The most primitive cryptographic technologies, a scytale, implements a transposition cipher. A scytale is a cylinder of a certain dimension used by the ancient Greeks for military communication. The sender would wrap a piece of parchment around the cylinder, then write the message along the length of the cylinder. When this was unwound from the scytale, the letters were shifted along the parchment, and in order to read the intended message the receiver would wrap the parchment around a cylinder of the same dimension as the sender's scytale.

In substitution ciphers, on the other hand, a letter in the plaintext is mapped to another letter in the alphabet when constructing the ciphertext. The simplest of these ciphers is the shift cipher, where the mapping of a letter is simply shifting it a set number of positions in the alphabet. For the classic Caesar cipher, the number of shifts is three, meaning an '**a**' in the plaintext is written as a '*d*' in the ciphertext, '**b**' is '*e*' and so on. In a more general substitution cipher, there is no dependency between the mappings of letters, other than the restriction that two distinct plaintext letters cannot be substituted by the same letter. For the English alphabet, there are therefore a total of 26! possible ways to set up a general substitution cipher.

However, where there are those that hide, there are typically also those who seek. Cryptography, the art of hiding, is only one part of cryptology; the natural counterpart, that of seeking to uncover the text which has been hidden, is known as cryptanalysis.

The most naive form of cryptanalysis when presented with a ciphertext is to simply try all the possible ways a plaintext might be revealed. For example, for the ciphertext *n mfyj fsi n qtaj. bmd n it ymnx ujwmfux dtz fxp. n pstb sty, gzy n kjjq ny mfuujsnsl, fsi n fr ytwyzwji* encrypted by the shift cipher, the cryptanalyst could simply try shifting the letters back one, two, and so on times before being able to read **i hate and i love. why i do this perhaps you ask. i know not, but i feel it happening, and i am tortured** after the fifth shift. For the shift cipher, this brute force method was feasible for anyone, but for the more general substitution cipher, with more than 4×10^{26} possible ways of mapping a letter of the plaintext alphabet to a letter in the ciphertext alphabet, this task was much too daunting for anyone. For this cipher, and indeed most others, a cryptanalyst must have a different strategy than using brute force, a strategy which exploits a weakness in the cipher itself.

The weakness of the substitution cipher is that although a letter of the plaintext is mapped to a different letter for the ciphertext, it is always mapped to the same letter, thus preserving the letter frequency: if there are 10 **a**s in the plaintext, and **a** is mapped to *t*, there will be 10 *t*s in the ciphertext. This property makes the cipher susceptible to frequency analysis, which exploits the fact that the various letters of a language appear at different frequencies in texts. In English, for example, ‘e’ is the letter which occurs the most often, with ‘j’ being on the other side of the spectrum. A cryptanalyst being presented with a ciphertext may thus count how many times a letter occurs in the ciphertext before her, attempt to replace the most frequent ones with those letters that occur most frequently in English, and thus attempt to recover the plaintext. Frequency analysis effectively makes certain ‘letter mappings’ more probable than others and allows the attacker to start the attack with one of the most likely mappings.

Frequency analysis was discovered in the Middle East around the ninth century by Muslim scholars, who also used it in cryptanalysis. The consequences for this particular breakthrough were numerous and include Mary, Queen of Scots being executed, and Edgar Allan Poe rising to fame [31].

And so the history of cryptology goes: a cipher is broken by cryptanalysts, making it insecure, forcing cryptographers to develop new ciphers. Of course, the history is not quite so linear, as new ciphers were, and are still, being developed constantly, and insecure ciphers are used despite them being susceptible to attacks.

Throughout millennia of progress and development the main principles of transposition and substitution remained the same. Even the famous Enigma machine implements, in essence, a substitution cipher, and the two principles are cornerstones in the modern AES cryptosystem, though not the only ones.

An aspect which *did* evolve throughout history is the concept of a cipher, and how it was regarded. Of course, it is difficult to pinpoint exactly how something has been perceived historically, for example what the ancient Greeks thoughts on cryptography were as they wrapped a piece of parchment around a scytale. But what we do know is that Auguste Kerckhoff was the first to formulate the notion of a secret key in his articles on *La Cryptographie Militaire* in 1883 [23, 31], which does indicate a shift in perception. Kerckhoff stated six principles, the most important of which is that the cipher should remain indecipherable even if everything about it is known except for the key. This is a separation between the encryption procedure, the algorithm, and the secret key that determines the encryption. The secret key of the scytale is its diameter, the secret key of the shift cipher is the number of shifts, the secret key of the substitution cipher is the letter mapping being used, and the secret key of the Enigma machine is the starting positions of the rotors.

The advantage of separating the key from the algorithm is that it is far easier to change the key than the algorithm, so that even if the machinery or encryption algorithm falls into enemy hands, it is still safe to use for encryption so long as the key is kept secret. It is the knowledge of the secret key that separates an intended recipient and a cryptanalyst, not the knowledge of the cipher being used. The fact that the cryptanalyst, or adversary, does not know the secret key is what forces her to use cryptanalysis to recover the message or the secret key.

For all the cryptography discussed so far, the setup is the same: two parties agree on a cipher and share a secret key. This secret key is used both to encrypt messages and decrypt ciphertexts: the setting is symmetric, both in the sense that the same secret key is shared, and also that the process of encrypting and decrypting are mirror images of each other. This is the very basic setting of cryptography, but other and broader settings were found in the middle of the 20th century.

1.2 The Gradual Shift from Art to Science

A central issue with symmetric cryptography is that the two parties have to agree on a secret key. This is all well and good if it is possible to agree in person in advance what the key is going to be, or they have a system for creating such keys. There are myriads of examples of these agreements, ranging from minuscule books with secret keys printed in them to memorising long texts such as poems to use as a key. But what if setting up such an agreement in person is impossible? How do you agree on a secret key that *only* you and the other person know, when you cannot meet?

This problem is known as key distribution, and was solved by Diffie, Hellman and Merkle in the seventies, and published in an article aptly titled ‘New Directions in Cryptography’ in 1976 [11]. The solution is a key exchange protocol known as Diffie-Hellman, which allows two users to agree on a common secret key known only to them, even though their entire communication is over a public channel. Just as with regular cryptography, an adversary has to rely on cryptanalysis if she is to recover the secret key, as she only possesses the public material.

With a viable solution to the key distribution problem, cryptography became substantially easier to use, seeing as there was no longer a need for either meeting someone in person to exchange secret keys, or trust in third parties or other lines of communication. The seventies therefore saw a surge of cryptography used in practice, outside of the military and larger corporations. Another important factor to this increase was the standardisation of cryptography, as the Data Encryption Standard (DES) was published in 1975 and became a standard in 1977. The standard made it substantially easier for cryptography to be used across fields, and also inspired cryptanalysts to band together in scrutiny. The DES was prodded and poked for years in search of weaknesses, and many were found, due to both technological and theoretical advances. Due to these advances it was eventually deemed to no longer provide sufficient security, as the key size was too small to prevent brute-force attacks. The key is a 64-bit string, of which eight are used for parity checks, so the system has 2^{56} possible keys, and by the mid-nineties technology was sufficiently fast for an adversary to simply try decrypting with all the keys until she found the correct one in a not unreasonable amount of time. The DES was therefore replaced by the Advanced Encryption Standard (AES) in 2001, once again repeating the history of cryptographers creating new ciphers after cryptanalysts have demonstrated that a cipher no longer is secure. The AES is still the standard today, and cryptanalysts are still studying it and looking for possible attacks. What has been will be again, what has been done will be done again; there is nothing new under the sun.

Except, sometimes, new things *do* appear under the sun. So far, we have been in the symmetric cryptographic setting, where the sender and recipient share the same key, and decryption is essentially the reversal of encryption. This was the default setting of cryptology ever since its beginning, and it was taken as the only setting. In 1975, however, the horizon broadened in a spectacular fashion, as Diffie introduced the concept of *asymmetric* cryptography [11, 31].

As already mentioned, the two communicating parties share the same secret key in the symmetric setting. This key is used for both encryption and decryption, and it is therefore essential that it is kept away from prying eyes. Even though this was the default scenario for millennia, there is no natural law of cryptology stating that it must be thus. In the asymmetric setting there are two different keys: one separate key for encryption and another, different, key for decryption. The key used for encryption is made public and is at the complete disposal of senders and adversaries alike, but the key used for decryption is kept secret. For this reason, asymmetric cryptography is also known as public key cryptography.

Although Diffie introduced the concept of asymmetric cryptography in 1975, and published it with Hellman in ‘New Directions in Cryptography’ [11] the year after, it was only the idea that this was theoretically possible, no encryption scheme was suggested. The first asymmetric cryptosystem, RSA, was published by Rivest, Shamir, and Adleman in 1977, something truly new under the sun [27]. As a side note, their article also introduced the most famous couple in cryptographic history: Alice and Bob, the two characters who want to send confidential messages to each other. The couple’s eavesdropper and assumed ever-present adversary was also in the article, but she was not named Eve until almost 10 years later [1, 6].

However, it is worth mentioning that there are corners the sun does not reach, such as the corners of the Government Communication Headquarters (GCHQ) in Cheltenham, UK. Away from public eyes, researchers of this British facility discovered both a public key cryptosystem resembling RSA and the solution to the key distribution problem in 1973 and 1974, respectively. The fact that these discoveries had taken place were a well-kept secret until the reveal to the public in 1997, in a lecture by Clifford Cocks, the researcher who had invented the RSA-like cryptosystem four years before Rivest, Shamir, and Adleman [31].

As is clear from this brief history lesson, there was an enormous development in cryptography in the 20th century, with several new cryptosystems being developed, and even a new form of cryptography. This was not the only aspect where progress was large though. As cryptography became more rigorous, so did the concept of how to judge the security of a scheme.

But what does it even mean for a cryptosystem to be secure? Not even Kerckhoff [23], with his six principles, formulated this explicitly, only that the cryptosystem should remain 'indecipherable' even if everything except the secret key is known about the system. What precisely 'indecipherable' entails, however, is not discussed explicitly. This must be inferred from what a cryptosystem was meant to achieve: encrypt a message into a ciphertext so that it is nonsensical to eavesdroppers, but the ciphertext is decryptable for those in possession of the secret key, allowing them to read the original message. A cryptosystem is deemed secure as long as it achieves this. In other words: a system is secure as long as an adversary is unable to recover the message when she has access to the ciphertext which encrypts it. In the vast majority of cases, recovering the message indirectly requires recovering the secret key. A system which does not achieve the security it strives for is informally referred to as a 'broken' system.

With this, albeit ad hoc, definition of security, security of schemes may also be measured by how long it takes an adversary to find the message. In this sense, a substitution cipher is more secure than a shift cipher, because it takes an adversary more time to recover a message encrypted by substitution rather than just shifting the letters. This difference is related to how many possible secret keys there are for the adversary to try, as the brute force attack of simply trying every single key is always an option for any adversary. There are 26 possibilities for the shift cipher, but 26! possibilities for the substitution cipher. Although a large number of possible keys is necessary for the security of a cryptosystem, it is not a sufficient condition, as demonstrated by the substitution cipher being insecure. Furthermore, as technology improves and becomes faster, cryptosystems previously considered secure may become vulnerable to attacks, as was the case for DES. Having 2^{56} possible keys *was* sufficient to protect against brute-force attacks at the time

DES was proposed, but this is no longer the case.

After the Second World War, the somewhat ad hoc understandings were replaced with mathematical rigour, as cryptology became less of an art and more of a science. The main pioneer in this era was Claude Shannon, who published the seminal paper 'Communication Theory of Secrecy Systems' in 1949 [30]. The paper presents a more mathematical approach to cryptosystems, regarding them as sets of transformations. In Shannon's presentation, the secret key determines a unique and reversible transformation between a message space and a ciphertext space. Encrypting a message is simply applying the transformation determined by the key, and decryption is applying the inverse of the transformation to the ciphertext. The cryptosystem is then defined as the family of these possible transformations, and the security of the system is derived from the fact that the transformation which will decrypt a ciphertext exists alongside a collection of other transformations.

A critical point for Shannon in his security theory is that each key, or transformation, is chosen with a particular a priori probability, and similarly each possible message in the message space is associated with an a priori probability of being encrypted. Furthermore, it is assumed that the adversary knows all these probabilities, that is, the only difference between the adversary and decrypter is that the latter knows the transformation to be performed. Once the adversary observes a ciphertext, she calculates the a posteriori probabilities of the various keys and messages: given the observed ciphertext, what is the probability that a particular message has been encrypted, or that a particular key has been used for encryption. This calculation is a generalisation of cryptanalysis.

Shannon defines perfect secrecy with respect to these a posteriori probabilities, which is the highest security a cryptosystem can achieve. Perfect secrecy is defined by requiring that the a posteriori probability that the message being sent is m, given the ciphertext c, is identical to the a priori probability that the message is m. A necessary and sufficient condition for perfect secrecy is that the conditional probability of the ciphertext c being an encryption of the message m is the same as the ciphertext being an encryption of *any* message. For any message m, and any ciphertext c, there must be at least one key defining a transformation which maps m to c. In particular, this implies that there are at least as many possible keys as possible messages.

The most famous example of a cipher with perfect secrecy is the one-time pad. The key is simply a string of random and independent numbers, as long as the message to be encrypted, and encryption is shifting the plaintext letter in position i by the number in position i in the key. Because it achieves perfect secrecy, the one-time pad is an unconditionally secure encryption scheme, meaning that it can resist *any* cryptanalytic attack, no matter how much computational power and time the adversary has, so long as she only has access to the ciphertext and the key is only used once (hence the name one-time pad). This property is because every single key is chosen with uniform probability, meaning that, mathematically speaking, the ciphertext *c tlls* is just as likely to encrypt **i hate** as **i love**. It is therefore impossible for an adversary with a ciphertext to determine what the plaintext is, as every conceivable plaintext may be transformed into the ciphertext in question under a particular key. The only way for an adversary to be able to recover the plaintext from the ciphertext is if she already knows the plaintext.

Most cryptosystems are not unconditionally secure, as some keys may be more likely to be chosen than others, or the ciphertexts they produce are more likely to be encryptions of some messages than others. For example, it is impossible that the ciphertext *cat ul lus* encrypts **odi et amo** in a shift or substitution cipher, simply due to the way these cryptosystems are defined. The encryption procedure for both ciphers is restricted, so a message is not mapped to the entire ciphertext space under the possible keyed transformations. This property means that a ciphertext contains sufficient information for an adversary to find the underlying plaintext and the key which was used to encrypt it. The only protection the encryption procedure offers is that it will take time to calculate the plaintext or key.

Cryptosystems whose security is due to the computational cost of cryptanalysis, but which would succumb to an adversary with unlimited time are computationally secure, and the absolute vast majority of systems fall into this category. The security of computationally secure schemes is not proven in the same way as unconditionally secure schemes, in fact, it is not known if they can be proven at all. The security of such a scheme is first and foremost demonstrated by the inability of adversaries to break the scheme, like the history of cryptology shows. However, there is a more formal way of demonstrating security as well: by proving that for an adversary to be able to recover either the secret key or the plaintext, she has to solve a particular mathematical problem. If this problem is hard to solve, then so is recovering the secret key or plaintext. Computationally secure schemes may therefore be regarded as conditionally secure, as they are secure under the condition that a particular hardness assumption holds.

1.3 Security, Security Proofs, and Security Notions

It is valuable to have some sort of notion of just *how* secure a computationally secure system actually is, how resilient they are against various attacks. The resilience is obviously going to depend on the problems they are based on, and the computational cost of solving them. However, a scheme may be broken by a brute-force attack unless the scheme is unconditionally secure. It is therefore this common attack that is the benchmark of the 'quantifying security' process of computationally secure schemes.

If an encryption scheme has an n-bit secret key, there are 2^n possible keys for an adversary to try in a brute-force attack. As n increases, so does the cost of a brute-force attack, and n is therefore referred to as the security parameter of the scheme. The cost of other attacks also increases as n does, though the rate of the increase will depend on the attack.

It is common to require that the cost of the attack should increase rapidly as the security parameter increases. In the best case the increase is exponential, and one of the worst is a polynomial growth, i.e., by a small power of n. Polynomial running time is often taken to mean efficient when it comes to algorithms. An algorithm which runs in polynomial time may still be wildly inefficient in practice, but in general there is sufficient overlap between polynomial run time and efficiency. For a scheme to be deemed secure, it is therefore common to require that there should not exist an attack whose cost scales polynomially against it. In other words: a scheme is considered secure if there does not exist an efficient attack that can break it.

In addition to looking at the cost of an attack, we may also examine how likely it is that an attack will succeed. Just as for the cost, the probability of the attack being successful will depend on the security parameter. Ideally, the probability of an adversary with a polynomial amount of time on her hands succeeding in her attack will get substantially smaller the more the security parameter n is increased. This concept is more formally captured in the notion of negligibility: a function f is negligible if, for every polynomial p, $\lim_{n\to\infty} p(n)f(n) = 0$ [29]. When the probability of a successful attack is expressed as a function f of the security parameter, this function should be negligible.

Negligibility is also central for defining computational indistinguishability, which is an important concept in security. Let $\{X_n\}_{n\in\mathbb{N}}$ and $\{Y_n\}_{n\in\mathbb{N}}$ be two probability distribution ensembles, and let D be a distinguishing algorithm running in polynomial time, which takes an element sampled from either probability distribution as input, and guesses which distribution the element was sampled from. Without loss of generality, assume that D outputs 0 to guess that the element is drawn from an X distribution, and 1 to guess a Y distribution. If

$$|\Pr[s \leftarrow X_n\ :\ D(s) = 1] - \Pr[s \leftarrow Y_n\ :\ D(s) = 1]| \leq \epsilon(n)$$

for a negligible function ϵ for any distinguisher D running in polynomial, the two probability distribution ensembles are said to be *computationally indistinguishable* [15]. Here, $\Pr[\mathit{Code} : \mathit{Event}]$ denotes probabilities where *Code* induces a probability distribution over which *Event* is defined. The expression $|\Pr[s \leftarrow X_n : D(s) = 1] - \Pr[s \leftarrow Y_n : D(s) = 1]|$ is the advantage of the algorithm D: the probability that it will guess wrong, minus the probability that it will be correct. In other words: two probability distributions are computationally indistinguishable if no efficient algorithm may tell them apart with any reasonable success.

An example of computational indistinguishability is the decisional Diffie-Hellman (DDH) assumption [8]. It should come as no surprise that the assumption is closely related the Diffie-Hellman key exchange protocol mentioned in the previous section. The DDH assumption is that the following two probability distributions are computationally indistinguishable. Let G be a multiplicative group of order p with a generator g, where G depends on the security parameter n. The first distribution is of tuples (g^x, g^y, g^z) for x, y, z sampled uniformly from $\mathbb{Z}_p$, whereas the second distribution is tuples (g^x, g^y, g^{xy}) for x, y sampled uniformly from $\mathbb{Z}_p$. The advantage of a distinguisher given (p, g, g^x, g^y, g^z) trying to guess if $z = xy$ is assumed to be negligible in the security parameter [7].

The actual Diffie-Hellman key exchange works as follows: Alice and Bob publicly agree on a multiplicative group of order p with a generator g. Alice chooses a random integer x, computes g^x and sends it to Bob. Similarly, Bob chooses a random integer y, computes g^y and sends it to Alice. Once they have exchanged group elements, Alice computes $(g^y)^x = g^{xy}$ and Bob computes $(g^x)^y = g^{xy}$, and they use g^{xy} as their shared secret key [27]. The adversary Eve only knows the group Alice and Bob have decided on, as well as the elements g^x and g^y they have sent to each other.

At first glance, it might seem that Eve will break the scheme if she is able to calculate the secret key g^{xy} given only this public information. However, this phrasing is not quite strict enough: if Eve is able to calculate 80% of the secret key, she is technically not breaking the scheme according to this naive definition, but she may still pose a threat to Alice and Bob. A more precise definition is that Eve will break the scheme if she is able to deduce *any* information about the secret key. A minimum criterion for this deduction is that given an element of the group, Eve is able to determine if this element is the secret key. Another way to phrase this is that if Eve is given the public information of the key exchange protocol and (g^x, g^y, g^z), she is able to determine if $z = xy$ [7].

So, we now have a problem and a scheme that seem very related, the question is then how to show that an adversary attacking the scheme essentially has to solve the problem in order to break the scheme? This may be proven by a reduction from solving the actual problem to the problem of 'breaking the scheme'. Such a reduction is performed by constructing an adversary $\mathbb{A}$ trying to solve the actual problem, and give $\mathbb{A}$ access to an oracle. The oracle models an adversary with success probability 1 against the scheme:

she is always able to break it. If $\mathbb{A}$ is able to solve her problem with non-negligible probability in polynomial time as long as she gets help from the oracle, it demonstrates that an adversary $\mathbb{B}$ with a non-negligible probability of breaking the scheme may be 'transformed' into an adversary with a non-negligible probability of solving the problem.

For the DDH problem and Diffie-Hellman key exchange, the reduction is fairly straightforward. An adversary against the DDH problem is given (g^x, g^y, g^z) in addition to some information on the group g is a generator of, and essentially has to decide whether $z = xy$. An oracle against the Diffie-Hellman key exchange, though, will output precisely this answer when given the input (p, g, g^x, g^y, g^z). The adversary therefore feeds the oracle the information it requires, and simply copies the oracle's answer. With the help of the oracle, the adversary will guess correctly every single time.

This reduction demonstrates that if it is possible for Eve to recognize the secret key g^{xy} if she sees it after being given the public information, g^x and g^y, then she would also be able to solve the decisional Diffie-Hellman problem. However, the DDH problem is assumed to be hard to solve for certain families of groups. The natural conclusion is therefore that if the assumption that the DDH problem is hard holds for the chosen group, then Eve cannot even recognize the secret key g^{xy} of Alice and Bob after they have performed the Diffie-Hellman key exchange if she sees it, much less actually compute any part of the secret key.

However, it is important to emphasize that this security proof rests on the assumption that Eve *only* eavesdrops. If Eve is assumed to be an active adversary, the situation is very different: Eve may simply intercept Alice's and Bob's transmission of g^x and g^y respectively, and send them both her own element g^z. By doing this, Eve has agreed on the secret key g^{xz} with Alice, and another secret key g^{yz} with Bob, all whilst Alice and Bob believe they have agreed on a secret key with each other. This sets Eve up as a (wo)man in the middle, and she may therefore lean back and read any messages Alice and Bob send to each other, as long as she can intercept them. Note that Eve managed to do this without having to solve the DDH problem, or any other computational problem for that matter.

Although the example used so far is not an encryption scheme, the idea of reduction is the same, as is the fact that we can have different attack models based on how active the adversary is. Before we get into more details on security for public key encryption schemes, though, we give a formal definition of what it actually is.

Definition 1 (Public Key Encryption Scheme [5]) *A public key encryption scheme is given by a triple of algorithms* $\mathrm{PKE} = (\mathsf{Pke.Kg}, \mathsf{Pke.Enc}, \mathsf{Pke.Dec})$ *where*

- $\mathsf{Pke.Kg}$*, the key generating algorithm, is a probabilistic algorithm that takes a security parameter* $n \in \mathbb{N}$ *and returns a pair* (pk, sk) *of matching public and secret keys.*
- $\mathsf{Pke.Enc}$*, the encryption algorithm, is a probabilistic algorithm that takes a public key* pk *and a message* m *in the message space* $\mathcal{M}$ *and returns a ciphertext* c*.*
- $\mathsf{Pke.Dec}$*, the decryption algorithm, is a deterministic algorithm that takes a secret key* sk *and a ciphertext* c *and returns either a message* m *in the message space or a special symbol* $\perp$ *to indicate that the ciphertext is invalid.*

All these algorithms should be computable in polynomial time. We also require that the scheme is correct: for all pairs (sk, pk) *output by* $\mathsf{Pke.Kg}$*, all messagges* $m \in \mathcal{M}$ *and all ciphertext* c *output by* $\mathsf{Pke.Enc}(pk, m)$*, we have that* $\mathsf{Pke.Dec}(sk, c) = m$*.*

How do we judge the security of an asymmetric encryption scheme? Although perfect secrecy and unconditional security is known not to be achievable for asymmetric schemes, the security captured in the notion is still relevant for these schemes. Recall that perfect secrecy states that a ciphertext does not give the adversary *any* information about the corresponding plaintext. For computationally secure schemes, this goal may be eased so that a polynomial time adversary should only be able to deduce a negligible amount of information about a plaintext given a ciphertext which encrypts it.

This is a security goal known as *semantic security* (SS) and was introduced by Goldwasser and Micali in 1984 [17], as they pointed out that the contemporary security requirements for asymmetric encryption schemes did not rule out that an adversary could compute some partial information about, or in fact the entirety of, the message m given an encryption c of it. Goldwasser and Micali, in the tradition of Shannon, originally formulated the security goal with respect to the a priori probabilities of messages and the a posteriori probabilities of observed ciphertext encrypting certain messages, and required that these probabilities should be the same for a polynomial time adversary. Goldreich [16] later defined the security goal in terms of whether an adversary with a ciphertext was better at computing information about the corresponding message than someone without the ciphertext. We give a somewhat simplified version of Golreich's presentation of semantic security here.

The notion of security is captured by comparing the probability of two algorithms winning a game. One is an adversary $\mathbb{A}$ attacking the encryption scheme PKE, the other is a simulator $\mathbb{A}'$, and they both win the game if they are able to compute some information $f(m)$ about a message. For the adversary, the procedure is as follows: first the encryption scheme is set up and a random message m is sampled from the message space $\mathcal{M}$, which is then encrypted $\mathsf{Pke.Enc}(pk, m) \to c$. The public key pk, the length of the message m, and the challenge ciphertext c are sent to the adversary $\mathbb{A}$, who tries to extract some partial information $f(m)$ about the message. For the simulator $\mathbb{A}'$, the procedure is the same, with the important difference that whilst $\mathbb{A}'$ gets the public key pk and length of m she does *not* get the ciphertext c. In other words: the simulator is tasked with calculating the same partial information as $\mathbb{A}$, but without the actual ciphertext. After a polynomial time, both the adversary and simulator output ν, and they win the game if $\nu = f(m)$. It stands to reason that if the success rate of the adversary and of the simulation are only negligibly different, no matter the choice of distribution over the message space $\mathcal{M}$ or function f, the adversary cannot deduce any partial information from the ciphertext [16].

Just as in the case of the Diffie-Hellman key exchange, there is an assumption here that the adversary is passive and only listens in. This attack is known as a Chosen Plaintext Attack (CPA) and is unavoidable in the asymmetric setting, seeing as the adversary has access to the public information and key. She may therefore encrypt whichever messages she likes. There are two other types of attacks, however, where the adversary has the power to *decrypt* certain ciphertexts of her own choosing. The two attacks are non-adaptive and adaptive Chosen Ciphertext Attacks (CCA1 and CCA2), and gives the adversary either temporary or full access to a decryption oracle $\mathcal{O}$. The adversary may send ciphertext queries to the oracle, and the oracle will return the decryption results of these ciphertexts. A non-adaptive attack captures the notion of an adversary with temporary access to decryption services, and is sometimes referred to as a lunch-time attack because it 'mimics' an adversary breaking in to a secret facility during the lunch break to study and getting to temporarily test the decryption in general, but not related to her problem specifically. An adaptive attack corresponds to the scenario where an adversary, for example, has stolen a decryption machine. Obviously, CCA2 is a stronger attack than CCA1, which itself is stronger than CPA, as the adversary has more power [5].

The description of semantic security implicitly assumed a chosen plaintext attack. For the CCA1 and CCA2 attacks the simulator runs as for the original CPA case, but the adversary is different. For the CCA1 attack, the adversary may query the decryption oracle polynomially many times after she is given the public key, but *before* she is given the challenge ciphertext. Once the actual challenge is given to her, she loses access to the decryption oracle. In the CCA2 attack, the decryption oracle is always available to the adversary, though she is prohibited from asking the oracle to decrypt the challenge ciphertext c, as this will result in a trivial win.

Although the goal of semantic security is a rather natural one, it is a bit cumbersome in practice, seeing as it is defined for *all* functions f expressing partial information of a message. Semantic security has, however, been proven to be equivalent to the security goal *indistinguishability* (IND) [32, 17]. This means that a scheme is SS-CPA/CCA1/CCA2 secure if and only if it is also IND-CPA/CCA1/CCA2 secure. Indistinguishability is a bit less of a natural security goal, but much more practical to work with. This goal is also captured in a game where the adversary is given the public information about the encryption scheme she is attacking. If she is mounting a chosen ciphertext attack she also has access to a decryption oracle she may query polynomially many times. She then chooses two distinct messages m_0, m_1 of equal length, and she sends them to a challenger. This oracle encrypts one of the messages and sends the encryption c back to the adversary. At this stage in the game she only has access to a decryption oracle if it is a CCA2 attack, and if so, she may not query the oracle with her challenge ciphertext c. Her task is to guess whether the challenger encrypted m_0 or m_1, and she wins the game if she guesses correctly. Since she has a 50% chance of guessing correctly by a coin flip, her advantage should only be negligibly better than 1/2.

Another security goal in addition to semantic security and indistinguishability is *non-malleability* (NM). Whereas the two former goals relate to how much information is leaked by the ciphertext, non-malleability relates to whether the adversary can meaningfully manipulate the ciphertext. Slightly more formally: for a scheme to be non-malleable, an adversary should only have negligible success when she is given a ciphertext c encrypting an unknown message m and tasked with producing a ciphertext c' and a nontrivial relation R of her own choosing such that $R(m, \mathsf{Pke.Dec}(c'))$ is true [5].

Combining any of the security goals mentioned with a type of attack presented gives rise to a security notion: a scheme which achieves the goal semantic security under a non-adaptive chosen ciphertext attack is an IND-CCA1 secure encryption scheme. As previously mentioned, semantic security has been proven to be equivalent to indistinguishability. Furthermore, Bellare et al. [5] have proven that the security notion NM-CCA2 is equivalent to IND-CCA2, and that if a scheme meets NM-CPA or NM-CCA1, then that scheme also meets the security notion IND-CPA or IND-CCA1, respectively.

The described advances have moved the discussion of security somewhat away from the trial and error based approach of the earlier history of cryptology. Instead, security is proven under certain assumptions, and it is these assumptions which in turn are attacked, not necessarily the schemes themselves. If an assumption proves to not hold, for example the DDH assumption, then the schemes whose proof of security is built on this assumption are possibly insecure. By basing security on established hardness assumptions and intractable computations, the foundation is much sturdier and the security clearer compared to before the 1940s, when security essentially boiled down to 'it seems difficult to break the scheme, therefore it is'.

1.4 Introduction to Advanced Asymmetric Cryptographic Primitives and Their Security

It was not just in the field of security analysis and proofs that strides were made after the Second World War, though, also cryptography saw much progress. The invention of both key exchange and asymmetric cryptography has already been mentioned, but these were far from the only schemes being developed. One new type of cryptography that may also be traced back to the seventies is what is known today as fully homomorphic encryption (FHE), which was first suggested very briefly after the RSA cryptosystem was proposed in 1978.

More specifically, Rivest, Adleman and Dertouzos presented the notion of 'privacy homomorphisms': "special encryption functions which permit encrypted data to be operated on without preliminary decryption of the operands"[26]. It is highly likely that the RSA cryptosystem inspired this notion, as RSA has the particular property of preserving multiplication of ciphertexts: $\mathrm{Dec}(\mathrm{Enc}(m_0) \cdot \mathrm{Enc}(m_1)) = m_0 m_1$, where $\cdot$ denotes the multiplication of ciphertexts and juxtaposition denotes the multiplication of plaintexts. Following this, encryption schemes which support computation on encrypted data became known as *homomorphic encryption* schemes. If an encryption scheme could support both addition and multiplication, it would be *fully homomorphic*, and one could perform any computation on ciphertexts without needing to decrypt them first.

Any function, and therefore any computation, may be expressed as a circuit with addition and multiplication gates. A more precise phrasing of fully homomorphic encryption is therefore that such a scheme would be able to evaluate any circuit C such that given encryptions $c_0, c_1, \ldots, c_t$ of messages $m_0, m_1, \ldots, m_t$, the evaluation of C given the ciphertexts as input will output a valid encryption of $C(m_0, m_1, \ldots, m_t)$. The name 'homomorphic' reflects this: the transformation by the function to the encrypted messages should be equal regardless of whether it is performed before or after decryption [3, 14].

Rivest et al. provided examples of privacy homomorphisms, but they were all later proven to be insecure, as an adversary would be able to recover the secret key [9]. Although several schemes able to homomorphically evaluate circuits with only addition or multiplication gates were suggested over the years, a concrete fully homomorphic encryption scheme was not suggested until Gentry did so in 2009 [14]. We give a formal definition of fully homomorphic encryption here.

Definition 2 (Fully Homomorphic Encryption [3, 14]) *A fully homomorphic encryption (FHE) scheme is given by a tuple* ($\mathsf{Fhe.Kg}$, $\mathsf{Fhe.Enc}$, $\mathsf{Fhe.Dec}$, $\mathsf{Fhe.Eval}$) *of probabilistic algorithms running in polynomial time, where*

- $\mathsf{Fhe.Kg}$*, the key generation algorithm, is an algorithm that takes a security parameter n and outputs a triple (pk, sk, evk), where pk, sk are as for public key encryption schemes, and evk is a key used for evaluation.*
- $\mathsf{Fhe.Enc}$*, the encryption algorithm, is an algorithm that takes a public key pk and a message m in the message space $\mathcal{M}$ as input and outputs a ciphertext c.*
- $\mathsf{Fhe.Eval}$*, the evaluation algorithm, is an algorithm that takes as input an evaluation key evk, a circuit C and a tuple of inputs that can be a mix of ciphertexts and previous evaluation results. It produces an evaluation output.*
- $\mathsf{Fhe.Dec}$*, the decryption algorithm, is an algorithm that takes a secret key sk and either a ciphertext c or evaluation result as input, and outputs either a message m or the invalid symbol $\perp$.*

As for public key encryption schemes, we require that an FHE scheme decrypts correctly: for all messages $m \in \mathcal{M}$, $\Pr[\mathsf{Fhe.Dec}(sk, \mathsf{Fhe.Enc}(pk, m)) = m] = 1$*. Furthermore, for all circuits C and for all ciphertexts c_i where $m_i \leftarrow \mathsf{Fhe.Dec}(sk, c_i)$ we require that the circuit C is evaluated correctly,*

$$\Pr[\mathsf{Fhe.Dec}(sk, \mathsf{Fhe.Eval}(evk, C, c_0, \ldots, c_t)) = C(m_0, \cdots, m_t)] = 1 - \epsilon(n),$$

where ϵ is a negligible function and n is the security parameter. Finally, an FHE scheme should be compact: for any circuit C and any ciphertext c_i, the size of the output $\mathsf{Fhe.Eval}(evk, C, c_0, \ldots, c_t)$ *is polynomial in the security parameter n, and independent of the size of the circuit.*

Note that this definition requires that the scheme correctly evaluates *any* circuit, this is what makes it *fully* homomorphic. If a scheme is only able to correctly evaluate a restricted set of circuits, the scheme is a *somewhat* homomorphic encryption (SHE) scheme, and a *leveled* homomorphic encryption scheme if it is able to correctly evaluate any circuit C with up to L consecutive gates for some chosen $L \in \mathbb{Z}$ [3].

It is also important to note that the requirement of correct evalution in the definition is for any *circuit*, singular. There is no inherent guarantee that the output of the evaluation algorithm is a valid input to the evaluation algorithm, hence the distinction of ciphertext and evaluation output in the definition. In other words: *staged* evaluations are not required to be correct. Any fully homomorphic encryption scheme which correctly evaluates any evaluation of i stages is said to be i-hop correct. This notion also extends to somewhat and leveled homomorphic encryption schemes [3].

The starting point for Gentry's FHE construction is an SHE scheme able to homomorphically evaluate its own decryption circuit. If a circuit is too deep for the SHE scheme to evaluate homomorphically, the circuit is essentially broken up into 'blocks' so that as much as possible of the circuit is evaluated, then the decryption circuit is evaluated, then the next 'block' of the original circuit. Homomorphically evaluating the decryption circuit essentially 'refreshes' the ciphertext, allowing it to be evaluated further. There is no limit to how many times the ciphertext may be refreshed, which means *any* circuit may be correctly evaluated homomorphically. The procedure of evaluating the decryption circuit homomorphically is known as 'bootstrapping', and is often a bottleneck in the homomorphic evaluation. Note also that bootstrapping requires that an encryption of the secret key is published, as it must be given as an input to the decryption circuit. Every acknowledged FHE scheme to date has used Gentry's blueprint of bootstrapping an SHE scheme in order to achieve the fully homomorphic property [20].

Despite these homomorphic properties, a fully homomorphic encryption scheme is still first and foremost an encryption scheme, and so the security notions for asymmetric cryptography have been carried over to FHE schemes. Obviously, FHE schemes are specifically designed to be malleable, and they will therefore never be secure with respect to any of the non-malleability security notions. Due to the equivalence with IND-CCA2, this notion is also impossible to satisfy for schemes with any homomorphic property. Several FHE schemes are proven to achieve IND-CPA, but the situation is more complicated for the IND-CCA1 case, as is discussed at length in Article III [12] of this thesis.

So far, the topic has been that of 'classical' cryptology, for lack of a better term, where the basic scenario is two people who want to send messages to each other no one else should be able to read. Phrased differently: only authorized people should be able to access the communication. This property is confidentiality, and it is the backbone of all cryptographic schemes. However, there are many other properties one may wish for in communication, and after the invention of asymmetric cryptography, the field has been expanded to not just regard confidentiality.

The earliest example is the issue of authentication: to ensure that the person who *claims* to have authored something is indeed the *real* author. This is tackled with digital signatures, which may be regarded as a companion of asymmetric encryption, and was introduced by Diffie and Hellman in the same article describing asymmetric cryptography [11]. In digital signatures, the sender signs the message using her own private key, and the resulting signature may be publicly verified to be hers by the public key, also called the verification key. The order of the private and public acts are switched around compared to asymmetric encryption. Because of this close relationship between asymmetric cryptography and digital signatures, Rivest, Shamir and Adleman were also the first to suggest a concrete digital signature scheme, which was based on the RSA cryptosystem.

The security of an encryption scheme is based on how easy it is for an adversary to recover the key or the message, that is, how easy it is to breach the confidentiality of the scheme. Similarly, since the main purpose of digital signature schemes is to verify the authenticity of a signature, the security of signature schemes is based on how easy it is for an adversary to forge a signature, and by doing so undermine the authenticity of the signature scheme.

Just as for encryption schemes, a security notion for digital signature schemes is created by demanding that a scheme obtains a security goal under a certain type of attack. Unforgeability is the standard security goal of a signature scheme, with different flavours of unforgeability, for example if the adversary is able to forge the signature of *any* message, or merely a subset of messages. The different attacks range from the adversary only knowing the public verification key, knowing the signatures of a subset of messages, or being able to see the signature of any message of her choosing. The various security notions for digital signature schemes were first formalised by Goldwasser, Micali and Rivest in 1984 [19].

Confidentiality and authentication are possible to obtain separately, but is it possible to create a scheme with both properties? The naive way is to simply encrypt the message, and then sign the resulting ciphertext. It is possible to prove that this encrypt-then-sign procedure preserves the security properties of both schemes as well as the public verification of the digital signature. However, there is a more efficient alternative to this naive approach: signcryption, which provides a single output which serves both as an encryption of and a signature on a message. Signcryption comes in a variety of flavours, with different security concerns for the different scenarios, for example if the scheme is to be used by two or multiple users [10].

The different varieties of signcryption therefore have different security requirements, and so further properties may be beneficial, in addition to confidentiality and authentication. One important such property is anonymity, meaning that a signcryption ciphertext should not reveal any information about the originator of it to anyone but the intended recipient. The reason anonymity arises as a security goal in signcryption is because in the multi users scenario identities and unique keys must be assigned to users, and it is conceivable that a signcryption ciphertext could reveal information about the key used, and therefore the identity of the signer [4]. For asymmetric encryption, every sender uses the same public key, and so this information leakage is not a concern, and a signature is designed so that anyone can publicly verify that it was created by a particular secret key. Hence anonymity is not a concern in either asymmetric encryption or digital signature schemes, and thus the security of signcryption is greater than the sum of its parts.

A common theme in the security goals discussed so far is that ciphertexts or signatures should be 'honest' somehow, for example that they should be non-malleable or impossible to forge, respectively. But how is this possible? How is it possible to prove that something is honestly generated, especially when this relates to whether something has been generated using a secret key, for example a signature? In the presence of an adversary with access to everything *but* the secret key, one way for the honest user to prove that their signature is genuine is to prove that they know the secret key, but without actually revealing said key.

Proving possession of knowledge without revealing the knowledge may be accomplished by using a zero knowledge proof protocol, first suggested by Goldwasser, Micali and Rackoff in 1985 [18]. In such a protocol, the prover Peggy wants to convince the verifier Victor that she possesses some knowledge, but without actually sharing this knowledge with Victor. In order to convince Victor, Peggy produces a proof that proves *that* she knows what she claims, but nothing more. As is to be expected, there are several security goals related to zero knowledge proof protocols, derived from the attacks one would want to prevent. For example the goal of zero knowledge: that Victor should not be able to extract any actual knowledge from the proof, and soundness: that Eve should not be able to cook up a convincing proof that she possesses knowledge she in fact does not.

Zero knowledge proof protocols are a powerful tool to provide security in the face of strong and dishonest adversaries, and the protocols have found several use cases as the field has been developed. Examples of such cases include fully homomorphic encryption, cryptocurrencies, electronic voting, and asymmetric encryption schemes both with and without additional properties.

As with asymmetric encryption, the security of advanced cryptographic primitives often relies on security reductions by demonstrating that if an adversary is able to undermine some particular notion of security of the scheme, she is also able to solve some problem which is believed to be hard. This may be phrased as "if the scheme does not achieve the security notion, then the hardness assumption does not hold". In this case, the cryptanalysis of the scheme in question reduces to analyzing the hard problem it reduces to, because it follows that if the assumption does not hold, then the reduction does not provide security. Although such a scenario is not enough to render a scheme insecure in general, as there might be other reductions that *do* provide security, it is often the case that schemes based on failed hardness assumptions are deemed insecure. Seeing as several schemes may reduce their security to the hardness of the same problem, trying to solve a particular hard problem is a more ‘general’ cryptanalysis than merely trying to break one particular scheme.

However, recall that a particular security notion is a combination of an attack and a security goal, and that just because a scheme achieves a particular goal in one attack model, does not mean that it obtains it in another model, nor another goal in the same attack model unless the two security notions are proven to be equivalent. It can be relevant to study the security of a particular scheme for other notions than what is captured by a particular reduction. In these cases, the specific scheme is analysed and attacked with respect to the particular security notion. This is also the cryptanalytical procedure for schemes that do not base their security on reductions, or base their security on heuristics. Fully homomorphic encryption schemes are a good example of this, as several schemes which are IND-CPA secure are demonstrably not IND-CCA1 secure, and no scheme with any homomorphic property can achieve non-malleability in any attack model, nor IND-CCA2 security. It follows that IND-CCA1 is the highest possible security notion for FHE schemes. There are presently no concrete FHE/LHE/SHE schemes that achieve IND-CCA1 security, and so analysing schemes with respect to this security may determine if the notion is at all possible for these schemes. Cryptanalysis may also provide important insights into how successful attacks may be thwarted for future schemes.

Finally, it is important to stress that the cryptanalysis described so far is just one rather theoretical aspect of cryptanalysis in general. There are many other aspects of security to account for before exclaiming that a scheme is secure, as the way in which a particular scheme is actually implemented and used may render even the strongest scheme insecure. A famous example of this is the Soviets not using truly random keys, or even using a key more than once, when encrypting messages with the unconditionally secure one-time pad during the Second World War. This misuse of the scheme enabled messages to be recovered from the ciphertexts [31]. All of which is to say that a reduction to a hard problem is no guarantee that a cryptographic scheme is secure in a real world setting.

And so the historical cycle of cryptanalytical and cryptographic progress is set to continue, the progress of one field forcing the other to advance and improve. What has been will be again, what has been done will be done again, but as we have seen, this cycle does cause new innovations and improvements to appear under the sun.

1.5 Contribution

We here briefly present the main findings of the four papers of the thesis, and how they fit into the context of the four previous sections. There are two thematic parts, where Article I, II, and III focus on the security of FHE/LHE/SHE schemes, and Article IV defines a new asymmetric cryptographic primitive.

Article I [21] shows that the fully homomorphic encryption scheme RC [28] is susceptible to a subfield lattice attack by Albrecht et al. [2], which recovers the secret key given only the public information of the encryption scheme. The RC scheme based its proof of security on the Decisional Small Polynomial Ratio (DSPR) problem, which is closely related to the well-established NTRU problem, but sufficiently different for the attack by Albrecht et al. to extend to RC, but not to schemes based on the NTRU problem. Article I is, in short, an example of a scheme shown to be insecure because the scheme reduces its security to a problem which is not sufficiently hard to provide security.

Furthermore, it is demonstrated that it is additional operations introduced to the scheme to accommodate for the homomorphic properties which forces the security of the scheme to reduce to the DSPR problem. Any attempt to avoid the subfield lattice attack will therefore come at the expense of sacrificing the fully homomorphic property of the scheme.

Article II [13] provides an adaptive key recovery attack on the leveled homomorphic encryption scheme LGM [24]. The LGM scheme claimed to achieve security against such an attack, but was unable to provide a proof of this security. At the time of publication, the LGM scheme was the only LHE or SHE scheme which was believed to be secure against adaptive key recovery attacks. Article II therefore falls into the other category of cryptanalysis described in Section 1.4, as in this case it was the scheme itself, not an underlying hardness assumption which was analysed and attacked.

The attack was implemented and ran on two realistic set of parameters for the scheme: 98.9% of the secret key was recovered after approximately 12 hours for one set, and the entire secret key was recovered after 48 hours for the other set of parameters. The implementation of the attack is open and available on GitHub [25].

Article III [12] provides an overview on all acknowledged FHE, LHE, and SHE schemes with regards to IND-CCA1 security, and show that none of them are secure with respect to this security notion. As alluded to in Section 1.4, IND-CCA1 security is where things get a bit complicated for homomorphic encryption schemes: there is no result stating that FHE/LHE/SHE schemes cannot achieve IND-CCA1 security, yet several of them are susceptible to key recovery attacks. Furthermore, any scheme which relies on bootstrapping cannot be IND-CCA1 secure, since bootstrapping requires that an encryption of the secret key is published. Therefore, an adversary with temporary access to a decryption oracle may simply decrypt this ciphertext to recover the secret key and decrypt *any* ciphertext without the oracle's help. As mentioned in Section 1.4, all acknowledged FHE schemes are based on Gentry's blueprint, and therefore rely on boostrapping, hence none of them are IND-CCA1 secure.

However, bootstrapping does not affect SHE and LHE schemes, and a handful of these schemes have attempted to achieve IND-CCA1 security, or like LGM, security against adaptive key recovery attacks. However, none of them have succeeded, and in fact most SHE and LHE schemes do not aim for this level of security, and several have shown to be susceptible to rather trivial attacks. Furthermore, the vulnerable schemes have been developed further, without attempts to improve the security of them with respect to IND-CCA1 security. The motivation of the paper is therefore to provide an assessment of the security of these constructions, as well as schemes that are not based on earlier schemes.

Article III extends already published key recovery attacks to new schemes, and also provides novel adaptive key recovery attack. We note that none of the attacked schemes aimed at achieving IND-CCA1 security, and so Article III may therefore be regarded as several examples of the final type of cryptanalysis mentioned in Section 1.4: analysing schemes with respect to other security notions than they are designed to achieve. Finally, we also provide an overview of possible general constructions for IND-CCA1 secure FHE/LHE/SHE schemes.

Article IV [22] presents a framework for Vetted Encryption (VE), a new cryptographic primitive with three different variants. The common scenario for all three variants is the following: a recipient vets which senders can send them messages by setting up a filter which publicly verifies that a particular sender is in fact vetted. This filter receives a single public key, and every vetted sender receives one personal encryption key.

The three variants are Anonymous, Identifiable and Opaque Vetted Encryption (AVE, IVE and OVE), where the difference is whether the identity of the sender is revealed to the filter and/or the recipient. In the AVE setting, the sender remains anonymous for both the filter and recipient, whereas the sender is identifiable to both entities in IVE. Both of these settings have great similarities to different variants of signcryption, and these similarities are also discussed in the article. Finally, in the OVE setting, the sender is anonymous for the filter, but is identified to the recipient. This setting of VE is more closely related to group signatures, and we also discuss these similarities.

We define security goals and notions for all three variants, and also provide general constructions which achieve the defined security notions. All the constructions rely on well-established cryptographic primitives such as public key encryption schemes, signature schemes, and zero knowledge proof protocols. As with signcryption, the security goals of either version of vetted encryption are larger than the sum of their parts.

Bibliography

[1] Alice and Bob: A History of The World's Most Famous Cryptographic Couple. `http://cryptocouple.com`. Accessed: 10.12.2021.

[2] Martin R. Albrecht, Shi Bai, and Léo Ducas. A subfield lattice attack on overstretched NTRU assumptions - cryptanalysis of some FHE and graded encoding schemes. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part I*, volume 9814 of *LNCS*, pages 153–178. Springer, Heidelberg, August 2016.

[3] Frederik Armknecht, Colin Boyd, Christopher Carr, Kristian Gjøsteen, Angela Jäschke, Christian A. Reuter, and Martin Strand. A guide to fully homomorphic encryption. Cryptology ePrint Archive, Report 2015/1192, 2015. `https://eprint.iacr.org/2015/1192`.

[4] Paulo S.L.M. Barreto, Benoît Libert, Noel McCullagh, and Jean-Jacques Quisquater. Signcryption Schemes Based on Bilinear Maps. In Alexander W. Dent and Yuliang Zheng, editors, *Practical Signcryption*, pages 71–97. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[5] Mihir Bellare, Anand Desai, David Pointcheval, and Phillip Rogaway. Relations among notions of security for public-key encryption schemes. In Hugo Krawczyk, editor, *CRYPTO'98*, volume 1462 of *LNCS*, pages 26–45. Springer, Heidelberg, August 1998.

[6] Charles H. Bennett, Gilles Brassard, and Jean-Marc Robert. How to reduce your enemy's information (extended abstract). In Hugh C. Williams, editor, *CRYPTO'85*, volume 218 of *LNCS*, pages 468–476. Springer, Heidelberg, August 1986.

[7] Dan Boneh. The Decision Diffie-Hellman problem. In Joe P. Buhler, editor, *Algorithmic Number Theory*, pages 48–63. Springer Berlin Heidelberg, 1998.

[8] Stefan A. Brands. An efficient off-line electronic cash system based on the representation problem. CWI Technical report, CS-R9323, 1993.

[9] Ernest F. Brickell and Yacov Yacobi. On privacy homomorphisms (extended abstract). In David Chaum and Wyn L. Price, editors, *EUROCRYPT'87*, volume 304 of *LNCS*, pages 117–125. Springer, Heidelberg, April 1988.

[10] Alexander W. Dent and Yuliang Zheng. *Practical Signcryption.* Springer, Berlin, Heidelberg, 2010.

[11] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[12] Prastudy Fauzi, Martha Norberg Hovd, and Håvard Raddum. On the IND-CCA1 Security of FHE Schemes. Cryptology ePrint Archive, Report 2021/1624, 2021. `https://eprint.iacr.org/2021/1624`.

[13] Prastudy Fauzi, Martha Norberg Hovd, and Håvard Raddum. A Practical Adaptive Key Recovery Attack on the LGM (GSW-like) Cryptosystem. In Jung Hee Cheon and Jean-Pierre Tillich, editors, *Post-Quantum Cryptography - 12th International Conference, PQCrypto 2021*, pages 483–498, Springer, Cham, July 2021.

[14] Craig Gentry. *A fully homomorphic encryption scheme.* PhD thesis, Stanford University, 2009.

[15] Oded Goldreich. *Foundations of Cryptography: Basic Tools*, volume 1. Cambridge University Press, Cambridge, UK, 2001.

[16] Oded Goldreich. *Foundations of Cryptography: Basic Applications*, volume 2. Cambridge University Press, Cambridge, UK, 2004.

[17] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.

[18] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186–208, 1989.

[19] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):281–308, April 1988.

[20] Shai Halevi. Homomorphic Encryption. In Yehuda Lindell, editor, *Tutorials on the Foundations of Cryptography: Dedicated to Oded Goldreich*, pages 219–276. Springer, Cham, 2017.

[21] Martha Norberg Hovd. A Successful Subfield Lattice Attack on a Fully Homomorphic Encryption Scheme. In Stig Frode Mjølsnes and Ragnar Soleng, editors, *Proceedings of the 11th Norwegian Information Security Conference*, Septemer2018.

[22] Martha Norberg Hovd and Martijn Stam. Vetted encryption. In Karthikeyan Bhargavan, Elisabeth Oswald, and Manoj Prabhakaran, editors, *INDOCRYPT 2020*, volume 12578 of *LNCS*, pages 488–507. Springer, Heidelberg, December 2020.

[23] Auguste Kerckhoff. La cryptographie militaire. *Journal des Sciences Militaires*, pages 5–38, January 1883.

[24] Zengpeng Li, Steven D. Galbraith, and Chunguang Ma. Preventing adaptive key recovery attacks on the gentry-sahai-waters leveled homomorphic encryption scheme. Cryptology ePrint Archive, Report 2016/1146, 2016. `https://eprint.iacr.org/2016/1146`.

[25] Håvard Raddum and Prastudy Fauzi. LGM-attack. Available at `https://github.com/Simula-UiB/LGM-attack`, 2021.

[26] Ronald L. Rivest, Leonard M. Adleman, Michael L. Dertouzos, et al. On data banks and privacy homomorphisms. *Foundations of secure computation*, 4(11):169–180, 1978.

[27] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the Association for Computing Machinery*, 21(2):120–126, 1978.

[28] Kurt Rohloff and David Bruce Cousins. A scalable implementation of fully homomorphic encryption built on NTRU. In Rainer Böhme, Michael Brenner, Tyler Moore, and Matthew Smith, editors, *FC 2014 Workshops*, volume 8438 of *LNCS*, pages 221–234. Springer, Heidelberg, March 2014.

[29] Mike Rosulek. The Joy of Cryptography. Online textbook. Available at `https://joyofcryptography.com`. Accessed: 10.12.2021.

[30] Claude E. Shannon. Communication theory of secrecy systems. *Bell Systems Technical Journal*, 28(4):656–715, 1949.

[31] Simon Singh. *Koder*. Aschehoug, Oslo, 2000.

[32] Yodai Watanabe, Junji Shikata, and Hideki Imai. Equivalence between semantic security and indistinguishability against chosen ciphertext attacks. In Yvo Desmedt, editor, *PKC 2003*, volume 2567 of *LNCS*, pages 71–84. Springer, Heidelberg, January 2003.

Chapter 2

Articles

Article I

2.1 A Successful Subfield Lattice Attack on a Fully Homomorphic Encryption Scheme

Martha Norberg Hovd

In Stig Frode Mjølsnes and Ragnar Soleng, editors, *Proceedings of the 11th Norwegian Information Security Conference*, September 2018. Presented by the candidate at *NISK* in September 2018.

Version reproduced here in Cryptology ePrint Archive, Report 2021/1626, 2021. https://eprint.iacr.org/2021/1626.

The article is based on the Master's thesis of the candidate, which was supervised by Kristian Gjøsteen at NTNU.

A Successful Subfield Lattice Attack on a Fully Homomorphic Encryption Scheme*

Martha Norberg Hovd[1,2]

[1] Simula UiB, Norway
[2] University of Bergen, Norway

Abstract

We present the application of a known subfield lattice attack on a fully homomorphic encryption scheme based on NTRU. We show that the scheme is vulnerable to the attack due to a particular parameter having to satisfy a derived lower bound. We also show that, due to the structure of the scheme, the attack is successful in all practical instantiations of the scheme.

1 Introduction

Fully homomorphic encryption (FHE) schemes are encryption schemes with the following property: for any function f defined over the message space, $\mathrm{Dec}(\mathrm{Eval}(f, c)) = f(\mathrm{Dec}(c))$, where $c = \mathrm{Enc}(m)$ for a message m, and Eval is an evaluation algorithm. The first such scheme was presented by Gentry in 2009 [4], and several schemes have been presented since. They mostly follow the same structure and have the same starting point: an encryption scheme where both multiplication and addition of **freshly generated** ciphertexts are homomorphic: $\mathrm{Dec}(\mathrm{Enc}(m_1) + \mathrm{Enc}(m_2)) = m_1 + m_2$, and $\mathrm{Dec}(\mathrm{Enc}(m_1)\mathrm{Enc}(m_2)) = m_1 m_2$ for two (possibly distinct) messages m_1, m_2.

All these starting schemes add bounded randomness to the plaintext to obscure it, and decryption is guaranteed to be correct so long as the randomness stays within the bounds prescribed during set-up, meaning that an encryption of m will actually decrypt to m. This bounded randomness is also referred to as 'noise'. The problem is that as operations are performed on a ciphertext, the noise may grow until it no longer respects the required bounds. At this point, the noise is said to have become unmanageable, as we no longer have any guarantee of correct decryption. These schemes which allow for a limited amount of homomorphic operations to be performed are merely somewhat homomorphic.

In order to have a fully homomorphic scheme the noise in the ciphertexts must be reduced, which is usually achieved through a combination of operations. These operations may stunt the growth of noise, or reduce it slightly, but it is not enough to provide an FHE scheme. To create an FHE scheme, bootstrapping is applied: a homomorphic evaluation of the decryption algorithm. Bootstrapping reduces the noise sufficiently to allow for homomorphic evaluation of any function, but it is a very time-consuming procedure. It is therefore preferable to construct an FHE scheme by relying on other strategies and using bootstrapping only as a last resort, as a scheme heavily dependent on bootstrapping is very impractical.

In some cases, the somewhat homomorphic 'starting scheme' is based on a previous scheme, but with different parameter settings, which may result in a less secure scheme. We show one such example in this article, namely that the NTRU-based FHE scheme RC by Rohloff and Cousins [8] is vulnerable to an attack by Albrecht et al. [1]. The RC scheme has different

*This paper appeared at the NISK 2018 conference.

parameter settings compared to the standard NTRU scheme to accommodate for the noise-reducing operations needed to perform homomorphic operations. In particular, this means that the attack by Albrecht et al. does not break the original NTRU encryption scheme.

2 Preliminaries

2.1 Notation

All vectors are row vectors and will be denoted with bold lower case letters: $\mathbf{v}, \mathbf{w}$, whilst matrices will be denoted using bold upper case letters: $\mathbf{A}, \mathbf{B}$. Elements of either a vector, a matrix or a polynomial ring will be denoted with a lower case letter in italics: a, b. Vectors will be written as $\mathbf{a} = [a_1, a_2, \dots, a_n]$, whereas sets will be denoted by $\{0, 1, \dots\}$.

Multiplication of integers, or an integer and a vector or polynomial is denoted by simple juxtaposition: $ab, a\mathbf{v}, af(x)$. Multiplication of a vector and a matrix will be denoted by a single dot: $\mathbf{v} \cdot \mathbf{A}$, and finally, the multiplication of two polynomials will be denoted by an asterisk: $f * g$. Furthermore, this polynomial multiplication always takes place in some polynomial ring, and the main motivation of the multiplicative notation is to serve as a reminder of this during computations. It should be clear from the context whether or not a given element is a polynomial, and any polynomial f will therefore, with very few exceptions, not be written $f(x)$.

Let $\mathbf{v}, \mathbf{w}$ be arrays of the same length k with elements from a polynomial ring R. We then define the inner product of them as $\langle \mathbf{v}, \mathbf{w} \rangle = \sum_{i=1}^{k} v_i * w_i \in R$. In addition, we have the following notation: for any two polynomials $a = \sum_{i=0}^{n-1} a_i x^i$, $b = \sum_{i=0}^{n-1} b_i x^i$, let $[a, b]$ denote the coefficient vector $[a_0, \dots, a_{n-1}, b_0, \dots, b_{n-1}]$.

The modular reduction $p = r \underline{\bmod} q$ reduces p modulo q to $r \in (-q/2, q/2]$. We also write $p \equiv r \underline{\bmod} q$ if we wish to stress that p is equivalent to r modulo q: $p = r + kq$, for $k \in \mathbb{Z}$. The notation generalizes to vectors and polynomials.

The Euclidean norm of an integer vector $\mathbf{v}$ is denoted by $\|\mathbf{v}\| = \sqrt{v_1^2 + v_2^2 + \cdots + v_n^2}$, whilst $\|\cdot\|_\infty$ denotes the infinity norm: $\|\mathbf{v}\|_\infty = \max_i\{|v_i|\}$. Supposing f is an integer polynomial, $\|f\|, \|f\|_\infty$ refers to calculating either norm of the coefficient vector of f.

For a probability distribution χ, $x \leftarrow \chi$ refers to drawing x according to χ. Furthermore, any logarithm log will be to the base 2.

Finally, throughout this paper, the following lemma will prove quite useful.

Lemma 2.1. *The following bound holds for any two elements $a, b \in \mathbb{Z}[x]/(x^n + 1)$:*

$$\|a * b\|_\infty \leq n\|a\|_\infty \|b\|_\infty.$$

Proof. Seeing as $a_i \leq \|a\|_\infty$, $b_i \leq \|b\|_\infty$ $\forall i \in \{0, 1, \dots, n-1\}$, it follows that $|a_i b_j| \leq \|a\|_\infty \|b\|_\infty$. Since the polynomial is reduced with respect to $x^n + 1$, every product $a_i b_j x^{i+j}$ with $i + j \geq n$ is reduced to $-a_i b_j x^{i+j-n}$ in the resulting polynomial ring element. Therefore, every coefficient of $a * b$ is a sum of n terms $a_i b_j$, and so it holds that $\|a * b\|_\infty \leq n\|a\|_\infty \|b\|_\infty$. $\square$

2.2 Lattices

Definition 2.2. *Let $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_\eta\}$ be a set of linearly independent vectors, with $\mathbf{v}_i \in \mathbb{R}^m$ $\forall i \in \{1, \ldots, \eta\}$. The lattice $\mathcal{L}$ generated by $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_\eta$ is the set of linear combinations of these vectors with coefficients in $\mathbb{Z}$:*

$$\mathcal{L} = \{a_1\mathbf{v}_1 + a_2\mathbf{v}_2 + \cdots + a_\eta\mathbf{v}_\eta : a_1, a_2, \ldots, a_\eta \in \mathbb{Z}\}.$$

A basis for the lattice $\mathcal{L}$ is any set of independent vectors that generates $\mathcal{L}$, and any two such sets will have the same dimension. Suppose $m = \eta$, we may then represent a basis by a square matrix (where the basis vectors form the rows of the matrix) and so we may calculate the determinant of it. There are of course many possible bases of a lattice $\mathcal{L}$, but as Proposition 6.14 of Hoffstein et al. [6] shows, any two bases of a lattice are related by an integer matrix with determinant ± 1. It follows from this result that for any two basis matrices $\mathbf{B}, \mathbf{B}'$ we have: $|\det(\mathbf{B})| = |\det(\mathbf{B}')|$. In other words, the determinant of basis matrices is a lattice invariant, defined as the determinant of the lattice.

Definition 2.3. *Let $\mathcal{L}$ be a lattice of dimension η with basis $B = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_\eta\}$, where $\mathbf{v}_i \in \mathbb{R}^\eta \quad \forall i \in \{1, 2, \ldots, \eta\}$. The determinant of $\mathcal{L}$ is defined as*

$$\det(\mathcal{L}) = |\det(\mathbf{B})|.$$

Any vector $\mathbf{v} \in \mathcal{L}$ has a (Euclidean) length, which we use to formulate the shortest vector problem of a lattice $\mathcal{L}$ [6].

The shortest vector problem (SVP): Find a shortest nonzero vector in a lattice $\mathcal{L}$, i.e. find a nonzero vector $\mathbf{v} \in \mathcal{L}$ that minimizes $\|\mathbf{v}\|$.

It may be shown that solving SVP is NP-hard under the randomized reduction hypothesis [6]. Due to this proven hardness, SVP is used in cryptographic settings, so that breaking an encryption scheme requires solving SVP for a certain instance. However, solving SVP precisely is not always necessary; in some cases, it suffices to compute merely an approximation of the vectors in question; that is, solving the following problem [6]:

Approximate-SVP: Let $\psi(\eta)$ be a function of the lattice dimension η of a lattice $\mathcal{L}$, with $\|\mathbf{v}_0\|$ the length of the shortest vectors in $\mathcal{L}$. Find a nonzero vector $\mathbf{v} \in \mathcal{L}$ such that $\|\mathbf{v}\| \leq \psi(\eta)\|\mathbf{v}_0\|$.

Of course, the length of the shortest vector $\mathbf{v}_0 \in \mathcal{L}$ is not always given, but an upper bound on $\|\mathbf{v}_0\|$ is always given by the following theorem:

Theorem 2.4 (Hermite's Theorem (Theorem 6.25 [6]))**.** *Every lattice $\mathcal{L}$ of dimension η has at least one nonzero vector $\mathbf{v} \in \mathcal{L}$ satisfying $\|\mathbf{v}\| \leq \sqrt{\eta}\det(\mathcal{L})^{1/\eta}$.*

Another result by Hermite is that for a given dimension η the Hermite's constant γ_η is the smallest value such that every lattice $\mathcal{L}$ of dimension η contains a nonzero vector $\mathbf{v} \in \mathcal{L}$ satisfying $\|\mathbf{v}\| \leq \sqrt{\gamma_\eta}\det(\mathcal{L})^{1/\eta}$. It follows that $\gamma_\eta \leq \eta$ [6]. Hermite's constant is generally not known. However, we may use the inequality to rephrase the approximate-SVP into the Hermite Shortest Vector Problem [3]:

HSVP: Given a lattice $\mathcal{L}$ and an approximation factor $\alpha > 0$, find a nonzero vector $\mathbf{v} \in \mathcal{L}$ such that $\|\mathbf{v}\| \leq \alpha\det(\mathcal{L})^{1/\eta}$.

The approximation factor α may be expressed as δ^η, where δ is known as the Hermite root factor.

Of course, a solution to any of these problems is seldom apparent given a basis B for a lattice, and the most efficient way of solving any of the presented problems is to find a basis which contains the solution of either stated problem. This is known as basis reduction, and the main algorithms are LLL [7] and its generalisation, BKZ [9], both of which are HSVP-algorithms [3].

LLL works by swapping two vectors in the basis and performing a reduction, whereas BKZ works similarly, only with more than two vectors. The number of vectors BKZ works with is known as the block size, denoted by β. The larger β is, the more precise the result of BKZ will be. Although the algorithms are not fully understood, it is known that BKZ outperforms LLL. BKZ also performs much better, both with respect to time and the resulting approximation factors, than any theoretical bound predicts [3].

2.3 An Introduction to NTRU and its Security

The original NTRU encryption scheme is defined over the polynomial ring $\mathbb{Z}[x]/(x^N - 1)$ for an integer N. The integer $q > 1$ is an additional parameter of the scheme, as most operations are performed modulo q [5]. We present the idea of a key recovery attack on NTRU here because it is the basis of a security argument for the RC scheme.

We present enough details of the NTRU-based scheme RC here to discuss a possible key recovery attack on it, and defer a full presentation to Section 4. The scheme follows the general structure of NTRU quite closely, the main difference is that the RC scheme is defined over the polynomial ring $R = \mathbb{Z}[x]/(x^n + 1)$, for $n = 2^k$. The secret key of the scheme is a polynomial $f \leftarrow \chi$, for a distribution χ over R, and f must be invertible modulo q. The public key is defined as $h = f^{-1} * g \underline{\text{mod}}\ q$, for the polynomial $g \leftarrow \chi$.

One way to try to find the secret key f given only the public information q, n and h is to reformulate the problem into one based on lattices. This is done by constructing a $2n \times 2n$ basis matrix for a lattice $\mathcal{L}_{\text{NTRU}}$. For an NTRU public key polynomial $h(x) = h_0 + \cdots + h_{n-1}x^{n-1}$, the basis matrix of the lattice $\mathcal{L}_{\text{NTRU}}$ is:

$$\mathbf{B}_{\text{NTRU}} = \begin{bmatrix} 1 & 0 & \dots & 0 & h_0 & h_1 & \dots & h_{n-1} \\ 0 & 1 & \dots & 0 & -h_{n-1} & h_0 & \dots & h_{n-2} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 & -h_1 & -h_2 & \dots & h_0 \\ 0 & 0 & \dots & 0 & q & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 & 0 & q & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & q \end{bmatrix}.$$

Recall that $h = g * f^{-1}$ and $f * f^{-1} = 1 + qf'$, so we must have $f * h = g + qu$ for some polynomial $u = g * f'$.

Proposition 2.5. *For the polynomials f, g and u as described above, we have: $[f, -u] \cdot \mathbf{B}_{\text{NTRU}} = [f, g]$.*

Proof. The n first coefficients of the resulting vector of $[f, -u] \cdot \mathbf{B}_{\text{NTRU}}$ are obviously f. Coefficient $n+1+k$, for $k \in \{0, 1, \dots n-1\}$ is expressed as:

$$\sum_{\substack{i,j=0 \\ i+j=k}}^{n-1} f_i h_j - \sum_{\substack{i,j=0 \\ i+j=k+n}}^{n-1} f_i h_j - qu_k = g_k + qu_k - qu_k = g_k,$$

where the fact that x^n is equivalent to -1 in $R = \mathbb{Z}[x]/(x^n+1)$ has been applied. Hence, $[f, -u] \cdot \mathbf{B}_{\text{NTRU}} = [f, g]$, which means that $[f, g]$ belongs to $\mathcal{L}_{\text{NTRU}}$, since the vector may be expressed as a linear combination of the basis vectors of $\mathcal{L}_{\text{NTRU}}$ using only integers. □

Supposing $[f, g]$ is among the shortest vectors in the lattice $\mathcal{L}_{\text{NTRU}}$, it follows that if an adversary is able to solve SVP in $\mathcal{L}_{\text{NTRU}}$, she is able to compute f based solely on public information, and thus break the scheme. Furthermore, any pair of polynomials $[\bar{f}, \bar{g}]$ with sufficiently small coefficients satisfying the relation $\bar{f} * h \equiv \bar{g} \underline{\text{mod}}\ q$ will also suffice, as will probably any solution to approximate-SVP for an approximation factor smaller than $\sqrt{n}$ [6]. Thus, recovering the secret key f of the encryption scheme reduces to solving approximate-SVP for the lattice $\mathcal{L}_{\text{NTRU}}$. We stress again that the vector $[f, g]$ being among the shortest vectors in the lattice $\mathcal{L}_{\text{NTRU}}$ is a condition for this strategy to work.

3 Subfield Lattice Attack

There may be more efficient attacks than applying LLL or BKZ on the lattice basis, depending on the properties of the scheme. We present one such attack here, by Albrecht et al. [1].

3.1 Algebraic Background

Let $\mathbb{K} = \mathbb{Q}[\omega]$ be a field, for a root of unity ω of order $2n$, for n a power of 2, and let $\mathbb{L}$ be a subfield of $\mathbb{K}$ such that $\mathbb{L} = \mathbb{Q}[\omega']$, for ω' a root of unity of order $2n'$, where $n' \leq n$ is also a power of 2, and define $\rho = n/n'$. These fields will have rings of integers $\mathbb{Z}[\omega]$ and $\mathbb{Z}[\omega']$, respectively. These rings of integers may be shown to be isomorphic to the polynomial rings $R = \mathbb{Z}[x]/(x^n+1)$ and $R' = \mathbb{Z}[x]/(x^{n'}+1)$ [1].

We know from Galois theory that there is a Galois group G' of automorphisms $\{\varphi_i\}$ on $\mathbb{K}$ that fixes $\mathbb{L}$ pointwise [1]. Using these automorphisms, we may define the norm function $\text{N}_{\mathbb{K}/\mathbb{L}} : \mathbb{K} \to \mathbb{L}$, as $\text{N}_{\mathbb{K}/\mathbb{L}}(a) = \prod_{\varphi_i \in G'} \varphi_i(a)$.

3.2 The Attack

Given an instance of an NTRU-based encryption scheme, with $sk = f$ and $pk = h = f^{-1} * g$, we define $f' = \text{N}_{\mathbb{K}/\mathbb{L}}(f), g' = \text{N}_{\mathbb{K}/\mathbb{L}}(g), h' = \text{N}_{\mathbb{K}/\mathbb{L}}(h)$ and a new lattice $\mathcal{L}'_{\text{NTRU}}$ defined by h' and q as described in Section 2.3. The approach of the attack is to find a short vector $[x', y'] \in \mathcal{L}'_{\text{NTRU}}$ by performing LLL on the basis B'_{NTRU} and lift this vector up to $[x, y]$ in the original lattice, using the canonical inclusion map. If the vector $[x', y']$ has certain properties, the vector $[x, y]$ will be short in $\mathcal{L}_{\text{NTRU}}$, and might therefore function as a secret key.

The actual attack rests on the following heuristic:

Heuristic 3.1. ***[Heuristic 1** [1]**]*** *For any n and any $f, g \in R$ with reasonable isotropic distribution of variance σ^2 and any constant $c > 0$, there exists a constant C such that $\|f'\| \leq (\sigma n^C)^\rho$ and $\|g'\| \leq (\sigma n^C)^\rho$, except with probability $\mathcal{O}(n^{-c})$.*

Moreover, Theorem 1 of Albrecht et al. [1] assures us of the existence of a lattice reduction algorithm with block-size β which is able to find a vector $[x', y'] \in R'$ such that $\|[x', y']\| \leq \beta^{\Theta(n'/\beta)}\|\mathbf{v}_0\|$ when applied to the basis of the lattice $\mathcal{L}'_{\text{NTRU}}$, where $\|\mathbf{v}_0\|$ denotes the length of the shortest vectors in the lattice. When combined with the observation that $\|\mathbf{v}_0\| \leq \|[f', g']\|$ and Heuristic 3.1, we conclude that there exists a lattice reduction algorithm which with high probability is able to find a vector $[x', y'] \in R'$ such that

$$\|[x', y']\| \leq \beta^{\Theta(n/\beta\rho)}\|[f', g']\| \leq \beta^{\Theta(n/\beta\rho)}(n\sigma)^{\Theta(\rho)}.$$

Furthermore, we also have the following theorem:

Theorem 3.2. ***[Theorem 2** [1]**]*** *Let $f', g' \in R'$ be such that $\langle f' \rangle$ and $\langle g' \rangle$ are coprime ideals*[1] *and $h' * f' = g' \ \underline{\text{mod}}\ q$ for some $h' \in R'$. If $[x', y'] \in \mathcal{L}'_{\text{NTRU}}$ has length satisfying*

$$\|[x', y']\| < \frac{q}{\|[f', g']\|}, \tag{1}$$

then $[x', y'] = v[f', g']$ for some $v \in R'$.

Based on the result derived from Heuristic 3.1 and Theorem 1 of Albrecht et al. [1], we conclude that for bound (1) to hold, and therefore for the attack to succeed, it suffices that

$$\beta^{\Theta(n/\beta\rho)}(n\sigma)^{\Theta(\rho)} \leq q. \tag{2}$$

Once the vector $[x', y']$ is found, we lift $x', y' \in R'$ to R using the canonical inclusion map $L : \mathbb{L} \to \mathbb{K}$:

$$\begin{aligned} x &= L(x') = L(v) * L(f'), \\ y &= L(y') * h/L(h') \ \underline{\text{mod}}\ q = L(v) * L(g') * h/L(h') \ \underline{\text{mod}}\ q, \end{aligned}$$

Here, v is as in Theorem 3.2. For simplicity, we set $\tilde{f} = L(f')/f$, $\tilde{g} = L(g')/g$ and $\tilde{h} = L(h')/h$; we then have

$$\begin{aligned} x &= L(v) * \tilde{f} * f \ \underline{\text{mod}}\ q \\ y &= L(v) * L(g')/\tilde{h} = L(v) * g * \tilde{g}/\tilde{h} = L(v) * \tilde{f} * g \ \underline{\text{mod}}\ q \\ \Rightarrow [x, y] &= u * [f, g] \in \mathcal{L}_{\text{NTRU}} \quad \text{with } u = L(v) * \tilde{f} \in R. \end{aligned}$$

In other words: the subfield attack finds a (small) multiplicative of $[f, g]$ under certain reasonable assumptions.

4 A Fully Homomorphic Encryption Scheme based on NTRU

4.1 The Somewhat Homomorphic Encryption Scheme

We now state the RC encryption scheme by Rohloff and Cousins [8]. The scheme is defined over the polynomial ring $R = \mathbb{Z}[x]/(x^n + 1)$, for n a power of 2. The scheme has the integer parameters q, p, chosen such that $q \gg p \geq 2$ and $\gcd(p, q) = 1$. Given these integers, the rings $R_p = \mathbb{Z}_p[x]/(x^n + 1)$ and $R_q = \mathbb{Z}_q[x]/(x^n + 1)$ are defined as the message and ciphertext space, respectively. In addition, the probability distribution χ over R_q is defined, which will typically be some discrete Gaussian distribution. The scheme consists of the following operations:

[1]Albrecht et al. note that the probability of $\langle f' \rangle$ and $\langle g' \rangle$ being coprime is roughly 3/4, and also that coprimality does not seem strictly necessary for the attack to be successful in practice [1].

KeyGen: Draw $f \leftarrow \chi$ such that $f \equiv 1 \underline{\text{mod}}\ p$ and $\exists f^{-1} \underline{\text{mod}}\ q$. Draw $g \leftarrow \chi$ as well, and output $pk = h = g * f^{-1} \underline{\text{mod}}\ q$ and $sk = f$.

Enc($pk = h, m \in R_p$): Draw $e, r \leftarrow \chi$ such that $e \equiv m \underline{\text{mod}}\ p$.
Output $c = pr * h + e \underline{\text{mod}}\ q$, $d = 1$.

Dec($sk = f, c \in R_q$, d): Compute $\bar{b} = f^d * c \underline{\text{mod}}\ q$ and lift this to the integer polynomial $b \in R$ with coefficients in $(-q/2, q/2]$. Output $m = b \underline{\text{mod}}\ p$.

EvalAdd(c_0, c_1, d_0, d_1): Output: $c = c_0 + c_1 \underline{\text{mod}}\ q$, $d = \max(d_0, d_1)$.

EvalMult(c_0, c_1, d_0, d_1): Output: $c = c_0 * c_1 \underline{\text{mod}}\ q$, $d = d_0 + d_1$.

The two latter operations are the homomorphic operations, and it is also these that necessitate the notion of the degree d of a ciphertext, which denotes the power of f^{-1} in the ciphertext. Note that $f^k * b = m \underline{\text{mod}}\ p$ for any power $k \geq 0$, whilst this is not necessarily the case for f^{-1}, as there is no guarantee that $f^{-1} = 1 \underline{\text{mod}}\ p$. Therefore, the decryption procedure will decrypt any ciphertext of degree at most the given d, assuming $f^d * c = f^k * b \underline{\text{mod}}\ q$, which is why d is set as $\max(d_0, d_1)$ in EvalAdd.

The polynomials f, g, r and e must be chosen so that they ensure correct decryption, so χ should have parameters ensuring that these polynomials are 'short enough'. What precisely this entails will be discussed at some length throughout this section. Essentially: we derive bounds the coefficients of these polynomials should satisfy to ensure correct decryption, even after the noise reducing operations have been performed on a ciphertext. The resulting bounds will be used to derive a final bound on q.

We start with the lower bound to be met on the coefficients of the polynomials f, g, r and e which ensures correct decryption of a freshly generated ciphertext.

Proposition 4.1. *If every coefficient of the polynomials f, g, r and e is strictly less than $\sqrt{\frac{q}{4pn}}$, any freshly generated ciphertext will be decrypted correctly.*

Proof. The decryption of $c = pr * h + e \underline{\text{mod}}\ q$ proceeds as follows, when viewed as an operation in R, as opposed to R_q:

$$\begin{aligned}\bar{b} &= f * c = f * (pr * h + e) = pf * r * g * f^{-1} + f * e \\ &= pq * r * g * f' + pr * g + f * e,\end{aligned}$$

where $f * f^{-1} = qf' + 1$. Consider the polynomial $pr * g + f * e$ in R. To ensure correct decryption, every coefficient of this polynomial should have absolute value less than $q/2$, or else the result is $b = pr * g + f * e - q\sum_{i=0}^{n-1} a_i x^i$ where some $a_i \neq 0$ and hence, $b \underline{\text{mod}}\ p$ need not equal m. Therefore, if the inequlity $\|pr * g + f * e\|_\infty < q/2$ is satisfied, any freshly generated ciphertext is decrypted correctly. Using the triangle inequality and Lemma 2.1, we may compute:

$$\begin{aligned}\|pr * g + f * e\|_\infty &\leq \|pr * g\|_\infty + \|f * e\|_\infty \\ &\leq pn\|r\|_\infty\|g\|_\infty + n\|f\|_\infty\|e\|_\infty \\ &\leq pn\|r\|_\infty\|g\|_\infty + pn\|f\|_\infty\|e\|_\infty \leq 2pnB^2, \qquad (3)\end{aligned}$$

for B a bound on the largest coefficient of r, g, f and e. If we assume (3) is less than $q/2$, then any fresh ciphertext will decrypt correctly. This assumption is true if the polynomials r, g, f and e are sampled from a distribution χ such that any coefficient is strictly less than $\sqrt{\frac{q}{4pn}}$. $\square$

4.2 Noise Reductions

The RC scheme uses key switching, ring reduction, modulus switching and bootstrapping as strategies to reduce the noise of a ciphertext, and thus turn the somewhat homomorphic scheme into a fully homomorphic encryption scheme. However, only key switching and modulus switching are being performed after every multiplication; we therefore only focus on these two operations in the following.

Note that this, strictly speaking, only makes the scheme presented here leveled homomorphic, as we need bootstrapping to make it truly fully homomorphic. Nevertheless, we refer to the scheme presented here as a fully homomorphic scheme, mainly to separate it from the 'starting scheme' presented in Section 4.1, and refer the interested reader to Rohloff and Cousins for details on the bootstrapping procedure [8].

4.2.1 Key Switching

Key switching converts a ciphertext of degree at most d encrypted under f_1 into a ciphertext of degree 1 encrypted under the secret key f_2. This procedure requires a hint, namely $a_{1\to 2} = \bar{a} * f_1^d * f_2^{-1} \underline{\bmod}\, q$, for $\chi \to \bar{a} \equiv 1 \underline{\bmod}\, p$. Given the hint, the actual key switching is the following procedure:

KeySwitch$(c_1, a_{1\to 2})$: Output: $c_2 = a_{1\to 2} * c_1 \underline{\bmod}\, q$.

Proposition 4.2. *Suppose c_1 is an encryption of m under f_1 of degree d which decrypts correctly:* $\mathrm{Dec}(f_1, c_1, d) = m$. *If every coefficient of f_1, f_2, g, r, e and $\bar{a}$ is strictly less than $(\frac{q}{2^{d+1}p^d n^{2d}})^{\frac{1}{2d+1}}$, then* $\mathrm{Dec}(f_2, c_2, 1) = m$, *with $a_{1\to 2}$ and c_2 generated according to the above KeySwitch procedure.*

Proof. Decryption of c_2 results in:

$$\begin{aligned}\bar{b}_2 &= f_2 * c_2 = f_2 * a_{1\to 2} * c_1 = f_2 * \bar{a} * f_1^d * f_2^{-1} * c_1 \\ &\equiv \bar{a} * f_1^d * c_1 \equiv \bar{a} * \bar{b}_1 \underline{\bmod}\, q\end{aligned}$$

If the inequality $\|\bar{a} * \bar{b}_1\|_\infty < q/2$ holds, decryption is guaranteed to be correct, i.e., $b_2 = \bar{a} * b_1 = \bar{a} * m = m \underline{\bmod}\, p$.

Seeing as c_1 is a ciphertext of degree d, it must be the result of $d-1$ multiplications so, without loss of generality, let $\bar{b}_1 = f_1^d * (pr * g * f_1^{-1} + e)^d \underline{\bmod}\, q$. If $\|\bar{a} * \bar{b}_1\|_\infty < q/2$ holds, it is the case that:

$$\begin{aligned}\|\bar{a} * f_1^d * (pr * g * f_1^{-1} + e)^d\|_\infty &= \|\bar{a} * f_1^d * \sum_{i=0}^{d} \binom{d}{i} p^i r^i * g^i * f_1^{-i} * e^{d-i}\|_\infty \\ &= \|\bar{a} * \sum_{i=0}^{d} \binom{d}{i} p^i r^i * g^i * f_1^{d-i} * e^{d-i}\|_\infty\end{aligned}$$

By Lemma 2.1 :

$$\begin{aligned}&\le n^{2d}\|\bar{a}\|_\infty \sum_{i=0}^{d} \binom{d}{i} p^i \|r\|_\infty^i \|g\|_\infty^i \|f\|_\infty^{d-i} \|e\|_\infty^{d-i} \\ &\le p^d n^{2d} B^{2d+1} \sum_{i=0}^{d} \binom{d}{i} = 2^d p^d n^{2d} B^{2d+1} < q/2.\end{aligned}$$

Here, B is a bound on the largest coefficient in $\bar{a}, r, g, f$ and e, and the condition of B being strictly less than $(\frac{q}{2^{d+1}p^d n^{2d}})^{\frac{1}{2d+1}}$ to ensure correct decryption after switching keys immediately follows. $\square$

It follows that key switching should be performed after every multiplication to minimize this bound. In the case $d = 2$ we have:

$$B^5 < \frac{q}{8p^2n^4}. \tag{4}$$

4.2.2 Modulus Switching

Modulus switching converts a ciphertext from modulus q to a smaller modulus, $\bar{q} = q/q'$ for some factor q' of q, by essentially dividing the ciphertext by q'. This operation will reduce the underlying noise of the ciphertext by a factor of approximately q'. The operation works by adding Δ, a small multiple of p equivalent to $-c$ modulo q', to the ciphertext c, so $c + \Delta$ is divisible by q'. This should only cause a slight increase in the noise of the ciphertext, and thus ensure that the underlying message is preserved. Seeing as $q'|q$, it follows that $\gcd(q', p) = 1 \Rightarrow \exists v$ s.t. $v = (q')^{-1} \underline{\text{mod}}\ p$. The procedure $\text{ModSwitch}(c, q, q')$ is performed as follows:

1. Compute a short $\varrho \in R$ such that $\varrho = c \ \underline{\text{mod}}\ q'$.
2. Compute a short $\Delta \in R$ such that $\Delta = (q'v - 1)\varrho \ \underline{\text{mod}}\ (pq')$.
3. Let $\varrho' = c + \Delta \ \underline{\text{mod}}\ q$. Note that q' divides ϱ' by construction.
4. Output $c' = (\varrho'/q') \in R_{\bar{q}}$.

Note that the final step indirectly multiplies ϱ with v, which is easily compensated for by either multiplying with q' in the final step of the decryption procedure or ensuring that $q' \equiv 1 \ \underline{\text{mod}}\ p$.

Proposition 4.3. *Suppose c is an encryption of degree 1 of the message m under the secret key f. Let $c' = \text{ModSwitch}(c, q, q')$. If every coefficient of f, g, r and e is less than or equal to B, which satisfies $\frac{1}{q'}(2pnB^2 + nB\frac{pq'}{2}) < \frac{q}{2q'}$, then $v\text{Dec}(f, c, 1) = \text{Dec}(f, c', 1)$.*

Proof. Let $\bar{q} = q/q'$. As $\varrho = c \ \underline{\text{mod}}\ q'$ and $v = (q')^{-1} \ \underline{\text{mod}}\ p$, we may write

$$\varrho = c - q'l \qquad \text{for } l \in R, \qquad q'v = 1 + pk \qquad \text{for } k \in \mathbb{Z}.$$

Following the procedure, we have[2]:

$$\begin{aligned}
(q'v - 1)\varrho &= pk(c - q'l) = pkc - pq'kl. \\
\Rightarrow \Delta &= pkc - pq's \text{ for } s \in R, \quad \text{as} \quad R \ni \Delta = (q'v-1)\varrho \equiv pkc \ \underline{\text{mod}}\, pq'. \\
\varrho' &= c + \Delta \ \underline{\text{mod}}\, q = c + pkc - pq's = (1 + pk)c - pq's \\
&\equiv q'vc - pq's \ \underline{\text{mod}}\, q. \\
c' &= \varrho'/q' \equiv vc - ps \ \underline{\text{mod}}\, \bar{q}.
\end{aligned}$$

If the inequality $\|vc - ps\|_\infty < \bar{q}/2$ is satisfied, decryption is correct:

$$f * c' = vf * c - pf * s = v(pr * g(qf' + 1) + f * e) - pf * s \in R$$

[2]Throughout this proof, pk denotes p multiplied with k, **not** the public key.

$$\equiv vpg * r + vf * e - pf * s \ \underline{\mathrm{mod}}\ \bar{q}$$

If the inequality $\|vf*e+vpg*r-pf*s\|_\infty < \bar{q}/2$, is respected, we will have: $(f*c' \ \underline{\mathrm{mod}}\bar{q}) \ \underline{\mathrm{mod}} p = vm$, and decryption of c' will be correct. Thus, the following expression should be satisfied for correct decryption:

$$\|f * c'\|_\infty = \|f * (c + \Delta)/q'\|_\infty \leq \frac{1}{q'}(\|f * c)\|_\infty + \|f * \Delta\|_\infty)$$

We use $c = p * r * g * f^{-1} + e$ as well as Lemma 2.1 and derive:

$$\begin{aligned} \frac{1}{q'}(\|pg * r + f * e\|_\infty + \|f * \Delta\|_\infty) &\leq \frac{1}{q'}(2pnB^2 + nB\|\Delta\|_\infty) \\ &\leq \frac{1}{q'}(2pnB^2 + nB\frac{pq'}{2}) < q/2q'. \qquad (5) \end{aligned}$$

□

4.3 ComposedEvalMult and the Growth of q

The operation ComposedEvalMult is simply the sequential execution of EvalMult, KeySwitch and ModSwitch.

Proposition 4.4. *Suppose c_0, c_1 are encryptions of messages m_0, m_1, respectively, under the public key $h = g * f_1^{-1}$, both of degree 1. Correctness of ComposedEvalMult means*

$$\mathrm{Dec}(f_2, \mathrm{ComposedEvalMult}(c_0, c_1), 1) = \mathrm{Dec}(f_1, c_0, 1) * \mathrm{Dec}(f_1, c_1, 1),$$

where f_2 is the new secret key after KeySwitch has been performed. The condition for correct decryption is that the polynomials $f_1, f_2, g, r_0, r_1, e_0, e_1$ and $\bar{a}$ are drawn from a distribution χ so that their largest coefficient is smaller than B, and that B satisfies

$$\frac{1}{q'}(4p^2n^4B^5 + nB\frac{pq'}{2}) < \frac{q}{2q'}.$$

Proof. Based on the proofs of propositions 4.2 and 4.3, it follows that

$$f_2 * \mathrm{ComposedEvalMult}(c_0, c_1) \equiv \bar{b} \ \underline{\mathrm{mod}}\ \bar{q},$$

where $\bar{b} = m_0 * m_1 \ \underline{\mathrm{mod}}\ p$. What needs to be calculated is the bound the drawn polynomials should satisfy so the noise added during multiplication and switching keys is sufficiently lowered by switching the modulus. The ciphertext ComposedEvalMult(c_0, c_1) outputs is of the form $c = \frac{1}{q'}(a_{1\to 2} * c_0 * c_1 + \Delta)$ for a factor q' of q. We have the following:

$$\begin{aligned} f_2 * c &= f_2 * \frac{1}{q'}(a_{1\to 2} * c_0 * c_1 + \Delta) \\ &= \frac{1}{q'} f_2 * (\bar{a} * f_2^{-1} * f_1^2 * (pr_0 * g * f_1^{-1} + e_0)(pr_1 * g * f_1^{-1} + e_1) + \Delta) \\ = \ldots &\equiv \frac{1}{q'}(p^2\bar{a} * r_0 * r_1 * g^2 + p\bar{a} * r_0 * g * f_1 * e_1 \\ &+ p\bar{a} * r_1 * g * f_1 * e_0 + \bar{a} * f_1^2 * e_0 * e_1 + f_2 * \Delta) = b' \equiv \bar{b} \ \underline{\mathrm{mod}}\ \bar{q}. \end{aligned}$$

If the inequality $\|b'\|_\infty < \bar{q}/2$ holds, the equality $b' = \bar{b}$ also holds. To achieve a bound on the coefficients of the polynomials, we use Lemma 2.1 and set

$$\|\bar{a}\|_\infty = \|r_0\|_\infty = \|r_1\|_\infty = \|g\|_\infty = \|f_1\|_\infty = \|f_2\|_\infty = \|e_0\|_\infty = \|e_1\|_\infty = B,$$

and we compute:

$$\begin{aligned} \|b'\|_\infty &\leq \frac{1}{q'}(p^2n^4B^5 + 2pn^4B^5 + n^4B^5 + nB\|\Delta\|_\infty) \\ &\leq \frac{1}{q'}(4p^2n^4B^5 + nB\frac{pq'}{2}). \end{aligned} \tag{6}$$

If $\frac{1}{q'}(4p^2n^4B^5 + nB\frac{pq'}{2})$ is less than $\frac{q}{2q'}$, ComposedEvalMult outputs a ciphertext guaranteed to be decrypted correctly. $\square$

Given this final bound on all the coefficients of the noise-inducing polynomials, we may use it to derive the final bound on q. This bound will depend on other parameters of the scheme and the probability distribution χ, and if the bound is satisfied decryption will be correct.

Suppose any of the polynomials affecting the noise level are drawn from a discrete Gaussian distribution with parameter r, and set w as an assurance measure so that it is highly improbable for any polynomial drawn from this distribution to have an Euclidean length greater than rw. It follows that we may set a bound on the infinity norm of any such distributed polynomial as $\frac{rw}{\sqrt{n}}$. Using this bound and expression (6), we set the condition that

$$\frac{1}{q'}(4p^2n^4(\frac{rw}{\sqrt{n}})^5 + n\frac{rw}{\sqrt{n}}\frac{pq'}{2}) = \frac{1}{q'}(4p^2n^{1.5}r^5w^5 + \frac{1}{2}pq'\sqrt{n}rw) < q/2q'$$

should be satisfied for decryption to be correct after a call to ComposedEvalMult.

Assuming that $4p^2n^{1.5}r^5w^5 < q'$ holds, it follows from the condition above that $1 + \frac{1}{2}p\sqrt{n}rw < q/2q'$. Furthermore, $q/q' \geq q_1$ for q_1 the smallest factor of q and thus also the smallest possible ciphertext modulus. In theory, q_1 could be significantly smaller than the other factors of q, as q_1 can be set as the final ciphertext modulus, which would not be subjected to a modulus switching. We would therefore only require q_1 to be large enough to decrypt ciphertexts that have undergone D modulus switchings. A more practical approach however, is to set the following universal bound for any factor of q, as Rohloff and Cousins, and we do:

$$q_i > 4p^2r^5w^5n^{1.5}. \tag{7}$$

We may therefore conclude that if all factors of q satisfies bound (7), the noise is sufficiently reduced to ensure correct decryption of any freshly generated ciphertext and output of ComposedEvalMult, given that the input ciphertexts has at most the same noise level as any freshly generated ciphertexts for the current ciphertext modulus $\bar{q}$. Hence, q should satisfy the following lower bound, as not doing so might result in an incorrect decryption:

$$q > (4p^2r^5w^5n^{1.5})^{D+1}. \tag{8}$$

5 Subfield Lattice Attack on the NTRU-based Fully Homomorphic Encryption Scheme

5.1 Applicability and Success of the Attack

It remains to be shown that the attack of section 3 is applicable to the RC scheme, and that the attack will be successful.

We note first that Albrecht et al. state in particular that Heuristic 3.1 holds for the Gaussian distribution [1], which is the distribution suggested for the RC scheme [8]. The attack is therefore applicable to the RC scheme.

However, as emphasized in the final paragraph of Section 2.3, an attack based on solving the SVP or approximate-SVP for the lattice $\mathcal{L}_{\mathrm{NTRU}}$ rests on the assumption that $[f, g]$ is among the shortest vectors in this lattice. This assumption must hold for the attack to produce a vector which can be used as a secret key. The assumption does in all likelihood hold, as the following proposition shows:

Proposition 5.1. *With overwhelming probability, the vector* $[f, g]$ *is one of the shortest vectors in the lattice* $\mathcal{L}_{\mathrm{NTRU}}$.

Proof. Recall Theorem 2.4: the length of the shortest vector in any lattice $\mathcal{L}$ is at most $\sqrt{\eta}\det(\mathcal{L})^{1/\eta}$. For $\mathcal{L}_{\mathrm{NTRU}}$, we get $\|\mathbf{v}_0\| \leq \sqrt{2n}(q^n)^{1/2n} = \sqrt{2nq}$.

We may calculate a bound on $\|[f, g]\|$, using the upper bound $\|f\|_\infty, \|g\|_\infty < \sqrt{\frac{q}{4pn}}$, derived in the proof of Proposition 4.1:

$$\|[f, g]\| = \sqrt{f_0^2 + \dots f_{n-1}^2 + g_0^2 + \dots + g_{n-1}^2} \leq \sqrt{2n\Big(\sqrt{\frac{q}{4pn}}\Big)^2} = \sqrt{\frac{q}{2p}}.$$

Comparing the two bounds, we have: $\sqrt{\frac{q}{2p}}\Big/\sqrt{2nq} = \sqrt{\frac{1}{4pn}} \ll 1$. Thus, seeing as the bound on $\|[f, g]\|$ is much smaller than the Hermite bound, it is highly probable that $[f, g]$ is one of the shortest vectors in $\mathcal{L}_{\mathrm{NTRU}}$. □

Thus, the attack is applicable to the RC scheme, and it will produce a vector usable as a secret key with overwhelming probability.

Regarding the success of the attack: recall bound (2) of Section 3.2:

$$\beta^{\Theta(n/\beta\rho)}(n\sigma)^{\Theta(\rho)} \leq q,$$

satisfaction of which ensures that the attack succeeds. The bound being satisfied is more likely as q grows larger with respect to n, i.e., the more factors q consists of, allowing for more CompEvalMult operations to be performed, the more likely the bound is to be satisfied.

If D modulus switchings are possible, then q will be of size $(4p^2r^5w^5n^{1.5})^{D+1}$, in accordance with bound (8). Allowing for D modulus switchings is desireable as it allows for at most D multiplications to be performed before needing to bootstrap. The success of the attack therefore hinges on whether the parameters also result in q satisfying bound (2). As the next subsection shows, the attack is successful for an extensive range of parameters, as q does satisfy bound (2) more often than not, and that setting the parameters in such a way that the attack fails results in an impractical encryption scheme.

5.2 Results

Albrecht et al. carried out experiments to test their attack on actual systems [1], which is neccessary due to a lack of understanding of the performance of the basis reduction algorithms LLL and BKZ. The experiments were carried out on NTRU bases over the ring $R = \mathbb{Z}[x]/(x^n + 1)$, for n a power of 2, which means that the experimental results are transferable to the RC scheme. We may therefore use the experimental data given by Albrecht et al. [1] to judge how successful such an attack may be on the RC scheme. We set the following values: $p = 2, r = w = 6$, which are the parameter values Rohloff and Cousins suggest [8].

For example, a successful attack was carried out in 3.5 hours for $n = 2^{11}$ when $\log(q) \geq 165$, which corresponds to $D = 3$, for $q = (4p^2r^5w^5n^{1.5})^{D+1}$ for the RC scheme. To achieve the same success by running BKZ on the full lattice (that is, not exploiting the possibility of using the sub-field strategy), an attacker would have to run BKZ with block size 27 to achieve $\delta = 1.0141$. For this block-size, BKZ is still considered practical, and the subfield lattice attack might therefore not be too big an improvement in this specific instance [2].

The highest dimension the attack was carried out in was $n = 2^{12}$, with success for $\log(q)$ as low as 190, yet again corresponding to $D = 3$, with the same parameter values as before. This attack took 120 hours, whereas a direct attack on the full lattice would require running BKZ with block size 131 to achieve $\delta = 1.0081$, an attack that seems unfeasible at this point, as $\beta = 131$ is much too large a block-size to be practical [2].

It follows from these utilizations of the attack that the RC scheme must be considered insecure if the scheme is also to make meaningful use of the noise reduction strategies presented. Note also that the subfield attacks used LLL to reduce the subfield basis. Therefore it seems reasonable to expect better attacks if BKZ was used on these bases instead, as BKZ consistently outperforms LLL.

6 Conclusions

We have shown that the subfield lattice attack described by Albrecht et al. [1] can be applied to the NTRU-based fully homomorphic encryption scheme RC by Rohloff and Cousins [8]. The attack requires the integer parameter q of the encryption scheme to satisfy a lower bound in order to be successful. At the same time, utilization of necessary operations that reduce the noise in a ciphertext *also* requires q to satisfy a second lower bound, which is typically much larger than the one required for the attack to be applicable. For the scheme to be safe from the attack, the parameters of the scheme make it very impractical, and essentially unusable, as it would result in a scheme overly dependent on bootstrapping. Thus, we conclude that the susceptibility of the described attack is inevitable, for all intents and purposes, if the scheme is to make meaningful use of its noise reducing operations.

References

[1] M. R. Albrecht, S. Bai, and L. Ducas. A subfield lattice attack on overstretched NTRU assumptions - cryptanalysis of some FHE and graded encoding schemes. In *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part I*, pages 153–178, 2016.

[2] Y. Chen and P. Q. Nguyen. BKZ 2.0: Better lattice security estimates. In *Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings*, pages 1–20, 2011.

[3] N. Gama and P. Q. Nguyen. Predicting lattice reduction. In *Advances in Cryptology - EUROCRYPT 2008, 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Istanbul, Turkey, April 13-17, 2008. Proceedings*, pages 31–51, 2008.

[4] C. Gentry. *A fully homomorphic encryption scheme.* PhD thesis, Stanford University, 2009.

[5] J. Hoffstein, J. Pipher, and J. H. Silverman. NTRU: A ring-based public key cryptosystem. In *Algorithmic Number Theory, Third International Symposium, ANTS-III, Portland, Oregon, USA, June 21-25, 1998, Proceedings*, pages 267–288, 1998.

[6] J. Hoffstein, J. Pipher, J. H. Silverman, and J. H. Silverman. *An introduction to mathematical cryptography*. Springer New York, second edition, 2014.

[7] A. K. Lenstra, H. W. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische annalen*, 261:515–534, 1982.

[8] K. Rohloff and D. B. Cousins. A scalable implementation of fully homomorphic encryption built on NTRU. In *Financial Cryptography and Data Security - FC 2014 Workshops, BITCOIN and WAHC 2014, Christ Church, Barbados, March 7, 2014, Revised Selected Papers*, pages 221–234, 2014.

[9] C.-P. Schnorr. A hierarchy of polynomial time lattice basis reduction algorithms. *Theoretical computer science*, 53(2-3):201–224, 1987.

Article II

2.2 A Practical Adaptive Key Recovery Attack on the LGM (GSW-like) Cryptosystem

Prastudy Fauzi, Martha Norberg Hovd and Håvard Raddum

In Jung Hee Cheon and Jean-Pierre Tillich, editors, *Post-Quantum Cryptography - 12th International Conference, PQCrypto 2021*, pages 483-498, Springer, Cham, July 2021. Presented by the candidate at *PQCrypto* in July 2021.

Version reproduced here in Cryptology ePrint Archive, Report 2021/658, 2021. `https://eprint.iacr.org/2021/658`.

The development of the attack, writing and editing was split equally between the authors. The candidate did not contribute to the implementation of the attack.

A Practical Adaptive Key Recovery Attack on the LGM (GSW-like) Cryptosystem[⋆]

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

[1] Simula UiB, Norway
[2] University of Bergen, Norway

Abstract. We present an adaptive key recovery attack on the leveled homomorphic encryption scheme suggested by Li, Galbraith and Ma (Provsec 2016), which itself is a modification of the GSW cryptosystem designed to resist key recovery attacks by using a different linear combination of secret keys for each decryption. We were able to efficiently recover the secret key for a realistic choice of parameters using a statistical attack. In particular, this means that the Li, Galbraith and Ma strategy does not prevent adaptive key recovery attacks.

Keywords: Key recovery, somewhat homomorphic encryption, GSW, statistical attack

1 Introduction

Fully homomorphic encryption (FHE) is a powerful primitive which allows for meaningful computations to be performed on encrypted data, without the need for decryption. An FHE scheme allows for ciphertexts to be evaluated over any circuit without risking an erroneous decryption of the resulting ciphertext, i.e., $\mathrm{Dec}(C(\mathrm{Enc}(m))) \neq C(m)$ for some circuit C. There are other flavours of homomorphic encryption as well: leveled homomorphic encryption (LHE) and somewhat homomorphic encryption (SHE), which both allow for a *limited amount* of operations to be performed on a ciphertext before there is a risk of decryption failure. A key difference between LHE and SHE is that the key generation of LHE schemes takes an extra parameter as input, which specifies the depth of the deepest circuit the scheme is able to homomorphically evaluate.

Many F/L/SHE schemes rely on a quantum secure assumption, such as the hardness of learning with errors (LWE) and ring learning with errors (RLWE), to provide security against key recovery attacks and/or message recovery. In fact, these schemes are typically shown to achieve IND-CPA security, meaning an adversary with access only to the public key and parameters is provably unable to distinguish between the encryptions of any two messages. However, most existing F/L/SHE schemes are known to be susceptible to *adaptive* key recovery attacks [8,10]. In these attacks an adversary has temporary access to

[⋆] This paper appeared at PQCrypto 2021. The final authenticated publication is available online at `https://doi.org/10.1007/978-3-030-81293-5_25`.

a decryption oracle, and is able to recover the secret key based on information leaked from the decryption queries.

For example, schemes based on LWE or RLWE (such as GSW [11]) can leak one bit of the secret key from a small number of decryption queries. These schemes have public keys of the form $(\mathbf{A}, \mathbf{A}\mathbf{s} + \mathbf{e})$ with secret key $\mathbf{s}$, a matrix $\mathbf{A}$, and noise $\mathbf{e}$. Key recovery attacks either compute $\mathbf{s}$ directly, or first compute the noise $\mathbf{e}$ and use this to derive $\mathbf{s}$. Chenal and Tang used this approach to attack several (R)LWE schemes, one of which was GSW [8].

Li, Galbraith and Ma (LGM, [13]) proposed a technique to circumvent such key recovery attacks: instead of decrypting using a single secret key $\mathbf{s}$, they suggested changing secret keys for every decryption, so any information leaked from two different decryption queries would be unrelated, which should make it impossible for an adversary to recover any secret key. They constructed an LHE scheme based on GSW they claimed achieved IND-CCA1 security, though they were unable to provide a formal proof. [3]

Concretely, they start with the dual version of GSW, where the public key is of the form $(\mathbf{A}, \mathbf{A}\mathbf{s})$, but the secret key $\mathbf{s}$ must have small norm; security now depends on the hardness of the inhomogeneous short integer solution (ISIS) problem, which is also assumed to be quantum secure. Instead of having one secret key $\mathbf{s}$, they generate t distinct secret keys: $\mathbf{s}_1, \ldots, \mathbf{s}_t$. During decryption a random linear combination of the secret keys, $\mathbf{s}' = \sum_{i=1}^{t} \lambda_i \mathbf{s}_i$, is used, where the λ_i's are redrawn from a distribution for each decryption. The message space is $\mathbb{Z}_2$, so a decryption query leaks, at best, one bit of $\mathbf{s}'$: an unknown linear combination of secret keys unlikely to ever be reused, since the λ_is that generate it are redrawn for every decryption. The technique successfully thwarts the known adaptive key recovery attacks on GSW and similar schemes, and Li et al. therefore argue that their scheme achieves IND-CCA1 security, though they are unable to prove it.

In this paper we show that the LGM scheme is still susceptible to an adaptive key recovery attack, as we are able to recover a secret key using a statistical attack. We go even further to claim that the general approach of using a random linear combination of secret keys for each decryption query is susceptible to statistical adaptive key recovery attacks. To the best of our knowledge, the LGM scheme is currently the only concrete leveled homomorphic encryption scheme attempting to achieve IND-CCA1 security, which has proven a difficult security notion to achieve for SHE or LHE schemes. [4]

2 Preliminaries

Vectors are denoted by bold, lower case letters, and are assumed to be in column form. Logarithms are always base 2. For a real number x, let $\lfloor x \rceil = \lfloor x + \frac{1}{2} \rfloor$ be the

[3] The original paper was published in ProvSec 2016 [14] however, the ePrint version [13] of the paper contains major changes. In particular, the scheme we mount the adaptive key recovery attack on in this article is found in the ePrint version.

[4] There are suggestions for generic constructions achieving IND-CCA1 security (e.g., [7]), but there are no concrete instantiations of these constructions.

closest integer to x. For any integers x and q, let $x \mod q$ denote the modular reduction centered around zero. For a vector $\mathbf{v}$, let $\|\mathbf{v}\|$ be its Euclidean norm. Unless stated otherwise, we refer to somewhat, leveled, and fully homomorphic encryption schemes as simply *homomorphic encryption schemes.*

The gadget vector $\mathbf{g}$ is defined as the column vector $(1, 2, \ldots, 2^{l-1})^T$, and the gadget matrix is defined as $\mathbf{G} = \mathbf{I}_n \otimes \mathbf{g} \in \mathbb{Z}_q^{n \times nl}$ (i.e., $\mathbf{G}$ is a matrix with $\mathbf{g}$ on the diagonal), for $l = \lfloor \log(q) \rceil + 1$.

Define $\mathbf{G}^{-1} : \mathbb{Z}_q^{n \times n'} \to \{0,1\}^{nl \times n'}$ to be the operation such that for any matrix $\mathbf{M} \in \mathbb{Z}_q^{n \times n'}$, we have that $\mathbf{G} \cdot \mathbf{G}^{-1}(\mathbf{M}) = \mathbf{M}$.

IND-CCA1. An encryption scheme $\mathcal{E} = (\text{Setup}, \text{KeyGen}, \text{Enc}, \text{Dec})$ achieves the security notion *indistinguishability under (non-adaptive) chosen ciphertext attack* (IND-CCA1) if any probabilistic polynomial time adversary $\mathbb{A}$ has at most a $1/2 + \text{negl}$ chance of winning the following game against a challenger $\mathcal{C}$:

- $\mathcal{C}$ derives the parameters using $params \leftarrow \text{Setup}(1^\kappa)$, draws a key pair $(pk, sk) \leftarrow \text{KeyGen}(params)$, and sends pk and the parameters to $\mathbb{A}$.
- $\mathbb{A}$ sends ciphertexts c to her decryption oracle $\mathcal{O}_{\text{Dec}}$, which returns $\text{Dec}(c)$.
- $\mathbb{A}$ sends two messages 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 $b^* = b$.

The notion of IND-CPA security is defined in a similar way, but here $\mathbb{A}$ does not have access to a decryption oracle.

An adaptive key recovery attack is stronger than an IND-CCA1 attack, as recovering the secret key enables an adversary to decrypt *all* ciphertexts, not just distinguish between the encryptions of two chosen messages.

LWE. The *Learning With Errors* (LWE) distribution is defined as follows: for a fixed vector $\mathbf{s}$ drawn uniformly at random from $\mathbb{Z}_q^n$, sample a vector $\mathbf{a}$ uniformly at random from $\mathbb{Z}_q^n$ and an error e from some noise distribution χ, and output $(\mathbf{a}, b = \mathbf{a} \cdot \mathbf{s}^T + e \mod q)$. The search problem of LWE is to find $\mathbf{s}$ given m samples of the LWE distribution, where $\mathbf{s}$ is fixed for all the samples.

ISIS. Given a modulus q, a matrix $\mathbf{B} \in \mathbb{Z}_q^{n \times m}$, and a vector $\mathbf{u}$, the *Inhomogeneous Short Integer Solution* (ISIS) problem is to find a vector $\mathbf{e}$ drawn from a distribution χ with bound B such that $\mathbf{Be} = \mathbf{u} \mod q$, if such a vector exists. It is required that $m > n$ to prevent an adversary simply finding $\mathbf{e}$ using Gaussian elimination [13,5].

2.1 Distributions

For integers $a \leq b$, let $[a, b]$ denote the set of integers x such that $a \leq x \leq b$. A distribution over values $S = [a, b]$ for integers $a \leq b$ is *discrete uniform* if all $n = b - a + 1$ values $x \in S$ can occur with equal probability $1/n$. Such a distribution has mean $\frac{a+b}{2}$ and variance $\frac{n^2-1}{12}$.

A distribution over values $\mathbb{R}$ is *Gaussian* with mean μ and variance σ^2 if it follows the probability density function

$$g(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{(x-\mu)^2}{2\sigma^2}} \quad .$$

A random variable following a Gaussian distribution is also said to be *normally distributed*. The value σ is also known as the *standard deviation*.

We provide, without proof, some known properties of Gaussian distributions.

Lemma 1. *Let $(X_i)_{i=1}^n$ be normally distributed independent random variables with mean μ_i and variance σ_i^2 for $i \in \{1, 2, \ldots, n\}$. Let $(a_i)_{i=1}^n$ be real numbers. Then $X = \sum_{i=1}^n a_i X_i$ is also normally distributed, with mean $\sum_{i=1}^n a_i \mu_i$ and variance $\sum_{i=1}^n a_i^2 \sigma_i^2$.*

Lemma 2. *Let X be a random variable following a Gaussian distribution with mean μ and variance σ^2. Then $Pr[|X - \mu| \geq t\sigma] = \mathbf{erf}(t/\sqrt{2})$, where $\mathbf{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$ is the* error function*. In particular, $Pr[|X - \mu| \geq 5\sigma] \leq 2^{-20}$.*

Theorem 1 (Central limit theorem for sample). *Let $X_1, \ldots, X_n$ be independent random variables from a distribution with mean μ and variance σ^2. Let $X = 1/n \cdot \sum_{i=1}^n X_i$. Then if n approaches infinity, $X - \mu$ converges to a Gaussian distribution with mean 0 and variance σ^2/n.*

Informally, by taking a large enough sample size n, $X - \mu$ has mean ϵ_μ and variance $\sigma^2/n + \epsilon_\sigma$, where $\epsilon_\mu, \epsilon_\sigma$ may both be made arbitrarily small.

Discrete Gaussian Distribution. The *discrete Gaussian distribution* may be viewed as a Gaussian distribution where the values are restricted to a countable set, say $\mathbb{Z}$. To preserve the desirable properties of Gaussian distributions mentioned above, one should not simply sample a Gaussian and round to the closest integer. Instead, we adapt the definition of Gaussian distributions over $S \subseteq \mathbb{Z}$ presented by Micciancio and Walter [16]. For the more general definition over $S \subseteq \mathbb{Z}^n$, we refer to [1,13].

Definition 1. *Let S be a subset of $\mathbb{Z}$. For $c \in \mathbb{R}$ and a parameter $\sigma > 0 \in \mathbb{R}$, define $\rho_{\sigma,c}(x) = e^{-\pi\frac{(x-c)^2}{\sigma^2}}$ and $\rho_{\sigma,c}(S) = \sum_{x \in S} \rho_{\sigma,c}(x)$. The discrete Gaussian distribution over S with center c and standard deviation $\sigma$is defined as*

$$\forall x \in S : D_{S,\sigma,c}(x) = \frac{\rho_{\sigma,c}(x)}{\rho_{\sigma,c}(S)}.$$

We also state Theorem 7 (one dimensional leftover hash lemma) of [13], as it is central for the derivation of parameters for the LGM scheme. Li et al. present it as a special case of Theorem 2 of Agrawal et al. [2].

Theorem 2. *Let $\sigma, \epsilon \in \mathbb{R}$ be such that $\epsilon > 0$ and $\sigma > C$ for some absolute constant C (see [2]). Let $t, \sigma' \in \mathbb{R}$ be such that $t \geq 10\log(8t^{1.5}\sigma)$ and $\sigma' \geq 4t\log(1/\epsilon)$. Then the statistical difference between the following two distributions is bounded by 2ϵ.*

- *Choose a length t vector $\mathbf{x} \in \mathbb{Z}^t$ with entries chosen from the discrete Gaussian distribution on $\mathbb{Z}^t$ with parameter σ and a length t vector $\mathbf{z} \in \mathbb{Z}^t$ with entries chosen from a discrete Gaussian distribution on $\mathbb{Z}^t$ with parameter σ' and compute the output $\mathbf{x}^T\mathbf{z}$.*
- *Choose and output an element from the discrete Gaussian distribution on $\mathbb{Z}$ with parameter $\sigma\sigma'$.*

3 The LGM Scheme

The leveled homomorphic encryption scheme LGM [13] is also known as DMGSW since it uses a multi-key and dual version of GSW. We present it using mostly the original notation, but denote the secret keys as $\mathbf{s}_i = (\mathbf{r_i} \parallel -\mathbf{e}_i^T)^T$, as opposed to $\mathbf{e}_i = (\mathbf{I}_i \parallel -\mathbf{t}_i^T)^T$. Note also that we omit the details of homomorphic addition and multiplication, as they are not relevant for our attack.

Setup$(1^\kappa, 1^L)$: Let $n = n(\kappa, L)$ and $m = m(\kappa, L)$ be parameters $n < m$ that depend on the security parameter κ and number of levels L. Choose a modulus q and bounded noise distribution $\chi = \chi(\kappa, L)$ on $\mathbb{Z}$ with bound B such that it achieves at least 2^κ security against known attacks. 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 $\mathbf{e}_i$ from χ^m, set $\mathbf{u}_i = \mathbf{B}\mathbf{e}_i$ and set $\mathbf{s}_i = (\mathbf{r}_i \parallel -\mathbf{e}_i^T)^T$, 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 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 $u = \langle \mathbf{C}_I, \mathbf{s}' \rangle \mod q$, where $\mathbf{C}_I$ is the Ith column of the ciphertext matrix $\mathbf{C}$. Finally, output $|\lfloor u/2^{j-1} \rceil| \in \{0, 1\}$.

Correct decryption of honestly generated ciphertexts follows from the following observations: first, note that $\mathbf{A}\mathbf{s}_i = 0$ for all i by construction, which ensures that $\mathbf{A}\mathbf{s}' \equiv 0 \mod q$. Then, due to $\mathbf{s}'$ being small and the choice of I, $u = \mu\mathbf{G}^T\mathbf{s}' + \mathbf{X}^T\mathbf{s}' = \mu 2^{j-1} + E$ for some small E. It is clear that the rounded division with 2^{j-1} will result in the message μ.

Li et al. mainly focus on the case where the λ_is are drawn uniformly at random from $\{0, 1\}$, but they also discuss other possible distributions to sample from, such as a larger uniform distribution or a discrete Gaussian distribution. We consider the security of the scheme in all these cases.

Deducing a message from a ciphertext boils down to solving the LWE-like instance $\mathbf{B}^T\mathbf{R} + \mathbf{X}$, whilst security against (non-adaptive) key recovery attacks is based on the ISIS problem.

The intuition behind LGM's claimed IND-CCA1 security is that since a new secret key is being used to decrypt every time the oracle is called, an adversary will be unable to deduce anything meaningful about either the summands of the key or the key itself, as she gets at most one bit of information from each decryption query, since the message space is $\mathbb{Z}_2$. Li et al. argued that any information leaked from one decryption query cannot be combined with information from another query, since the secret keys are different every time.

3.1 Parameter derivation

The authors do not suggest a concrete parameter setting for the LGM scheme; we therefore derive a realistic choice for parameters based on the information and bounds provided in [13], which we also state here.

- The parameters m, n, q and the bound B must all be chosen so that the instantiated cases of LWE and ISIS problems are hard to solve.
- The inequality $tB + mB^2 < q/8$ must be satisfied in order to prevent an erroneous decryption of a fresh (i.e., unevaluated) ciphertext.
- Li et al. suggest setting $B = 6\sigma$.
- If the distribution of the values of $\langle \mathbf{C}_I, \mathbf{s}' \rangle \mod q = \langle \mathbf{C}_I, \sum_{i=1}^{t} \lambda_i \mathbf{s}_i \rangle \mod q$ resembles a uniform distribution over $\mathbb{Z}_q$, it must be indistinguishable from a uniform distribution over $\mathbb{Z}_q$. By the leftover hash lemma, we must have $t \geq \log(q) + 3\kappa$ where κ is the security parameter of the scheme for the two distributions to be indistinguishable.
- If the distribution of the values of $\langle \mathbf{C}_I, \mathbf{s}' \rangle \mod q = \langle \mathbf{C}_I, \sum_{i=1}^{t} \lambda_i \mathbf{s}_i \rangle \mod q$ resembles a discrete Gaussian distribution over $\mathbb{Z}_q$, t and σ must satisfy the bounds of Theorem 2, i.e., $t \geq 10 \log(8t^{1.5}\sigma)$ and $\sigma \geq C$. This ensures that statistical difference of the distribution of values of $\langle \mathbf{C}_I, \mathbf{s}' \rangle \mod q$ and a discrete Gaussian with parameter $\sigma\sigma'$ is bounded by 2ϵ.

We stress that the distribution mentioned in the two final points arise naturally during decryption, and that the properties of the distribution depends on $\mathbf{C}_I$, so both points must be taken into account. We start from the final point to derive $\sigma \leq C$, for $C \geq 18K\eta_\epsilon(\mathbb{Z})$, where $K > 1$ is some universal constant and $\eta_\epsilon(\mathbb{Z}) \leq \sqrt{\ln(\epsilon/44 + 2/\epsilon)/\pi}$ is the smoothing parameter of the integers [2,18]. Setting $\epsilon = 0.005$ to provide a statistical difference of 0.01 according to Theorem 2 and assuming $K \approx 1$, we derive $\sigma \geq 25$, so we choose $\sigma = 25$. Using the other bounds, we get $t = 400$, $m = 525$, $B = 150$ and $q = 94,980,001$, which ensures a 120-bit security against currently known attacks on LWE and ISIS [3,5]. [5]

[5] Seeing as we do not use n in our attack, we do not set it explicitly. We do note, though, that it affects the hardness of the LWE instance, and is implicitly set by the requirement $m > n$. We assume $m \approx n$.

4 The Key Recovery Attack

First note that any non-zero linear combination of the secret vectors can be used to decrypt. Hence, to perform a successful key recovery attack we will only need to recover any single $\mathbf{s}_i = (\mathbf{r}_i \parallel -\mathbf{e}_i^T)^T$. We present our attack to recover the entire secret key $\{\mathbf{s}_1, \dots, \mathbf{s}_t\}$, by recovering the coefficients at a particular index across the t secrets $\mathbf{s}_i$, index by index. However, for the concrete experiments, we recover just one secret key.

We first assume the suggested variant of LGM where the values λ_i are chosen uniformly at random from $\mathbb{Z}_2$, and present the basic attack for this case. Afterwards we show that the attack generalises to the cases $\lambda_i \in [0, b-1]$ and $\lambda_i \in [-b, b]$, where b is some (very) small constant. This constraint on λ_i is necessary to ensure that $\|\mathbf{s}' = \sum \lambda_i \mathbf{s}_i\|$ is small, as is required by the scheme. We also discuss the security of the scheme for the case where the λ_is are sampled from a discrete Gaussian distribution.

$\boldsymbol{\lambda_i \in \{0, 1\}}$. Recall that decryption works by first choosing $\lambda_1, \dots, \lambda_t$ uniformly at random from $\{0, 1\}$ and then generating a one-time decryption key $\mathbf{s}'$ as

$$\mathbf{s}' = \lambda_1 \mathbf{s}_1 + \ldots + \lambda_t \mathbf{s}_t = (\lambda_1, \dots, \lambda_t, \sum_{i=1}^{t} \lambda_i e_{i,1}, \dots, \sum_{i=1}^{t} \lambda_i e_{i,m}).$$

Next, a column $\mathbf{C}_I$ of the ciphertext matrix $\mathbf{C}$ is chosen, where I corresponds to some index k that satisfies $\lambda_k = 1$. Then $u = \langle \mathbf{C}_I, \mathbf{s}' \rangle \mod q$ is computed, and the decryption oracle returns $|\lfloor u/2^{j-1} \rceil|$, for the unique (and known) value j for which $q/4 < 2^{j-1} \leq q/2$.

In the following we focus on recovering the first component of each $\mathbf{e}_i$, namely the $e_{1,1}, e_{2,1}, \dots, e_{t,1}$ that are linearly combined in position $t+1$ of $\mathbf{s}'$. The same attack can be carried out to recover all the other m positions with an easy adaptation. We construct our chosen ciphertexts from column vectors $c_{a,i}$ with some integer a in position i for $1 \leq i \leq t$, a 1 in position $t+1$ and 0 elsewhere:

$$c_{a,i} = (\underbrace{0, \dots, a, \dots, 0}_{\text{length } t,\, a \text{ in pos. } i}, \underbrace{1, 0, \dots, 0}_{\text{length } m})^T.$$

Let D_α be the ciphertext matrix where for all i the column corresponding to λ_i is $c_{\alpha,i}$, and let $R_{a,i}$ be the ciphertext matrix where every column is $c_{a,i}$:

$$D_\alpha = \begin{bmatrix} \alpha & 0 & \cdots & 0 \\ 0 & \alpha & \cdots & 0 \\ 0 & 0 & \cdots & \alpha \\ 1 & 1 & \cdots & 1 \\ & & \mathbf{0}_{(m-1)\times t} & \end{bmatrix}, \quad R_{a,i} = \begin{bmatrix} & & \mathbf{0}_{(i-1)\times t} & \\ a & a & \cdots & a \\ & & \mathbf{0}_{(t-i)\times t} & \\ 1 & 1 & \cdots & 1 \\ & & \mathbf{0}_{(m-1)\times t} & \end{bmatrix}.$$

Asking for the decryption of D_α will result in the following expression for $u = u(D_\alpha)$, no matter which index k corresponds to the chosen column $\mathbf{C}_I$:

$$u(D_\alpha) = \langle c_{\alpha,i}, \mathbf{s}' \rangle = \alpha + \sum_{i=1}^{t} \lambda_i e_{i,1}.$$

This is because it is a requirement that $\lambda_k = 1$ for the chosen index I. The output of the decryption query will depend on the size of α, as well as the value of $\sum_{i=1}^{t} \lambda_i e_{i,1}$. In the attack we will only use values of α that are close to 2^{j-2}. In particular, we will always have $0 < \alpha + \sum_{i=1}^{t} \lambda_i e_{i,1} < q$, and thus will never have to consider any reductions modulo q. If $\alpha + \sum_{i=1}^{t} \lambda_i e_{i,1} < 2^{j-2}$ the decryption oracle will return 0, and if $\alpha + \sum_{i=1}^{t} \lambda_i e_{i,1} \geq 2^{j-2}$ it will return 1. Define the value $E = \sum_{i=1}^{t} \lambda_i e_{i,1}$. Asking for the decryption of D_α many times will make E a stochastic variable that takes its value according to a discrete Gaussian distribution over the interval $[E_{\min}, E_{\max}]$, where $E_{\min}$ and $E_{\max}$ are the minimum and maximum values E can take, respectively. Denoting the expected value of E by $\mathbb{E}(E)$, we get $\mathbb{E}(E) = 1/2 \sum_{i=1}^{t} e_{i,1}$. Approximately half of the time E will take a value that is smaller than $\mathbb{E}(E)$ and approximately half of the time the value of E will be greater than $\mathbb{E}(E)$.

Similarly, asking for the decryption of $R_{a,i}$ will give the following expression for $u = u(R_{a,i})$:

$$u(R_{a,i}) = \langle c_{a,i}, \mathbf{s}' \rangle = \lambda_i a + \lambda_i e_{i,1} + \sum_{k \neq i} \lambda_k e_{k,1}.$$

In this case we do not know if λ_i is 0 or 1. If $\lambda_i = 0$, the result is $u(R_{a,i}) = \sum_{k \neq i} \lambda_k e_{k,1} \ll 2^{j-2}$, and so the decryption will output 0. If $\lambda_i = 1$ the output of the decryption query will depend on the size of a and the value of the sum $\sum_{k \neq i} \lambda_k e_{k,1}$. Define E_i to be $E_i = \sum_{k \neq i} \lambda_k e_{k,1}$. In the same way as above, asking for decryptions of $R_{a,i}$ multiple times will make E_i be normally distributed over some interval, with an expected value $\mathbb{E}(E_i) = 1/2 \sum_{k \neq i} e_{k,1}$.

The main idea of the attack is to ask for many decryptions of D_α and $R_{a,i}$, and count how often the decryption oracle returns 1. Asking for sufficiently many decryptions makes the randomness of the unknown and varying λ_i's even out to their expected values. Counting how often the decryption oracle returns 1 for various values of α and a allows us to extract information about the size of $e_{i,1}$, and accurately estimate its value.

Attack procedure. For a more detailed explanation of how the attack works, we start with the following definition.

Definition 2. *Let $h(\alpha)$ be the number of times the decryption oracle returns* 1 *when asked for a number of decryptions of D_α, and let $h_i(a)$ be the number of times the decryption oracle returns* 1 *when asked for a number of decryptions of $R_{a,i}$.*

The number of times we ask for the decryption of the same ciphertext is denoted by T, and its exact value will be determined later. By doing a binary search, find the integer α_0 such that $h(\alpha_0) < T/2 \leq h(\alpha_0 + 1)$. Next, make an interpolated value α_{est} that we estimate would give exactly $T/2$ decryptions returning 1 if we were allowed to ask for decryptions of D_α for $\alpha \in \mathbb{R}$:

$$\alpha_{est} = \frac{h(\alpha_0 + 1) - T/2}{h(\alpha_0 + 1) - h(\alpha_0)}\alpha_0 + \frac{T/2 - h(\alpha_0)}{h(\alpha_0 + 1) - h(\alpha_0)}(\alpha_0 + 1).$$

Note that the value α_{est} is a real value, and it is our best estimate for the equation $\alpha_{est} + 1/2\sum_{i=1}^{t} e_{i,1} = 2^{j-2}$ to hold. To be precise, we get the equation

$$\alpha_{est} + 1/2\sum_{i=1}^{t} e_{i,1} = 2^{j-2} + \epsilon, \tag{1}$$

where $|\epsilon|$ becomes small when the sample size T grows large.

Next, we repeat the process and ask for decryptions of $R_{a,i}$ for $i = 1, \ldots, t$. Note that $\lambda_i = 0$ half of the time in these decryptions, which always causes the oracle to return the value 0. So the values $h_i(a)$ will be approximately half of $h(a)$. This is compensated for by finding the value a_0 such that $2h_i(a_0) < T/2 \leq 2h_i(a_0 + 1)$. Knowing that $\lambda_i = 1$ whenever we get a 1-decryption, we do the same interpolation as above and find an estimate $a_{est} \in \mathbb{R}$ such that

$$a_{est} + e_{i,1} + 1/2\sum_{k \neq i} e_{k,1} = 2^{j-2} + \epsilon_i, \tag{2}$$

where $|\epsilon_i|$ is small. Subtracting (2) from (1) and rearranging we get

$$e_{i,1} = 2(\alpha_{est} - a_{est}) + 2(\epsilon_i - \epsilon). \tag{3}$$

Rounding the right-hand side value recovers the correct $e_{i,1}$, provided that T is large enough to make $|\epsilon_i| < 1/8$ and $|\epsilon| < 1/8$.

The attack can be repeated to recover all the $e_{i,x}$ for $x = 2, 3, \ldots, m$ by setting the 1 in $c_{a,i}$ to be in position $t + x$. One can also focus on fully recovering only one of the vectors $\mathbf{e}_i$ by recovering $e_{i,x}$ for some fixed i and $x = 1, 2, \ldots, m$. Note that the recovery of an entry of any $\mathbf{e}_i$-vector is independent of the recovery of any other entry. This is what enables us to recover a single $\mathbf{e}_i$-vector in its entirety, which can be used as a decryption key.

For the attack to work with high probability, we need $T \in O(t \cdot \sigma^2)$. In particular, we have the following:

Lemma 3. *If $T \geq 800 \cdot t \cdot \sigma^2$ then in Eq.* (1) *we have that $Pr[|\epsilon| \geq 1/8] \leq 2^{-20}$ and in Eq.* (2) *we have that $Pr[|\epsilon_i| \geq 1/8] \leq 2^{-20}$.*

Proof. Since λ_i is taken from the uniform distribution over $\{0, 1\}$ it has mean 1/2 and variance 1/4. Moreover, $e_{i,1}$ is taken from a Gaussian distribution with mean 0 and variance σ^2. Then $\lambda_i \cdot e_{i,1}$ is taken from a Gaussian distribution with mean $1/2 \cdot \sum e_{i,1}$ and variance $1/2 \cdot \sigma^2$. Hence $\epsilon = \sum_{i=1}^{t} \lambda_i \cdot e_{i,1} - \mathbb{E}(E)$ is a sum of Gaussians, which by Lemma 1 is also a Gaussian with mean 0 and variance $t/2 \cdot \sigma^2$. However, if we take an average of T samples of such a function, then by

the central limit theorem for sample means we get a Gaussian X with mean 0 and variance $\frac{t}{2T} \cdot \sigma^2$. If $T \geq 800 \cdot t \cdot \sigma^2$ then X has standard deviation $\leq 1/40$, in which case $Pr[|\epsilon| \geq 1/8] \leq 2^{-20}$ by Lemma 2. Similarly, $Pr[|\epsilon_i| \geq 1/8] \leq 2^{-20}$.
□

Remark 1. We cannot get a smaller lower bound for T using Lemma 8.1 of [1] since there is no guarantee that the Gaussian X in the above proof is integral.

The running time will then be $O(Tm) = O(t\sigma^2 m)$. We have that t can only be polynomially large in the security parameter to have efficient encryption. Also, $\|\mathbf{s}'\|$ must be small in order for the underlying ISIS problem to be hard. Therefore, the attack runs in polynomial time.

4.1 Generalisation of the attack

The above attack assumes that the values λ_i were sampled uniformly at random from $\{0, 1\}$. We now investigate whether the attack can be prevented by choosing the λ_i from a larger set. The two generalisations we consider are $\lambda_i \in \{0, 1, \ldots, b-1\}$, or $\lambda_i \in \{-b, \ldots, b\}$. As before, we can take $T \in O(t \cdot \sigma^2)$. We show that the attack can be generalised to work in both cases.

$\boldsymbol{\lambda_i \in \{0, 1, \ldots, b-1\}}$. The attack can be adapted to work when the λ_i are sampled uniformly at random from $\{0, 1, \ldots, b-1\}$. We again focus on recovering the coefficients $e_{i,1}$, the other $e_{i,x}$'s are recovered by repeating the attack with the same adaptation as above. This also means that we may choose to recover a single $\mathbf{e}_i$-vector here as well.

When the decryption oracle is given the ciphertext matrix D_α, it will compute $u(D_\alpha) = \lambda_k \alpha + \sum_{i=1}^t \lambda_i e_{i,1}$, where $\lambda_k \neq 0$. When $\alpha \approx 2^{j-2}/(b-1)$, the decryption oracle will return 0 whenever $\lambda_k < b-1$, since $\lambda_k \alpha + \sum_{i=1}^t \lambda_i e_{i,1} < 2^{j-2}$ in this case. Only when $\lambda_k = b-1$ can we get decryptions that return 1. We know that $\lambda_k \neq 0$ when decrypting D_α, so the probability that $\lambda_k = b-1$ is $1/(b-1)$.

As before, we scale the numbers $h(\alpha)$ with $b-1$ to do a binary search and find the value α_0 such that $(b-1)h(\alpha_0) < T/2 \leq (b-1)h(\alpha_0+1)$. We then use this to estimate the α_{est} for which we would expect $(b-1)h(\alpha_{est}) = T/2$ if we were allowed to ask for decryptions of D_α where $\alpha \in \mathbb{R}$.

When $\lambda_i \in \{0, 1, \ldots, b-1\}$, the expected value of $\sum_{i=1}^t \lambda_i e_{i,1}$ is $\frac{b-1}{2} \sum_{i=1}^t e_{i,1}$. The α_{est} we find therefore gives the equation

$$(b-1)\alpha_{est} + \frac{b-1}{2} \sum_{i=1}^{t} e_{i,1} = 2^{j-2} + \epsilon, \tag{4}$$

where $|\epsilon|$ is small for large T.

In the same fashion we can ask for decryptions of $R_{a,i}$ where $a \approx 2^{j-2}/(b-1)$ and count the number of 1-decryptions we get. When decrypting $R_{a,i}$ we may have $\lambda_i = 0$, so the probability that $\lambda_i = b-1$ (which is necessary for the decryption oracle to return 1) is $1/b$. We therefore scale the values of $h_i(a)$

by b, and find an interpolated value for a_{est} based on the value a_0 for which $bh_i(a_0) < T/2 \leq bh_i(a_0+1)$. This yields the equation

$$(b-1)a_{est} + (b-1)e_{i,1} + \frac{b-1}{2}\sum_{k\neq i}^{t} e_{k,1} = 2^{j-2} + \epsilon_i, \tag{5}$$

where $|\epsilon_i|$ is small. Subtracting (5) from (4) gives

$$e_{i,1} = 2(\alpha_{est} - a_{est} + \frac{\epsilon_i - \epsilon}{b-1}),$$

and rounding this value recovers the correct $e_{i,1}$, provided $|\epsilon_i| < (b-1)/8$ and $|\epsilon| < (b-1)/8$.

The proof of the following lemma is almost identical to Lemma 3 and is thus omitted. In fact, the upper bound for $Pr[|\epsilon_i| \geq (b-1)/8]$ and $Pr[|\epsilon| \geq (b-1)/8]$ will be even smaller than 2^{-20} for $b > 2$.

Lemma 4. *If $T \geq 800 \cdot t \cdot \sigma^2$ then in Eq.* (4) *we have that $Pr[|\epsilon| \geq 1/8] \leq 2^{-20}$ and in Eq.* (5) *we have that $Pr[|\epsilon_i| \geq 1/8] \leq 2^{-20}$.*

$\boldsymbol{\lambda \in \{-b, \ldots, b\}}$**.** We start by asking for T decryptions of D_α, where $\alpha \approx 2^{j-2}/b$, and count how often the decryption oracle returns 1. Recall that the decryption outputs the absolute value of $\lfloor u/2^{j-1}\rceil$, so there are now two cases where the decryption oracle can return 1, namely when $\lambda_i = b$ or $\lambda_i = -b$. There are $2b+1$ numbers in $\{-b, \ldots, b\}$, but 0 cannot be chosen for λ_k when decrypting D_α, so the probability of having λ_k equal to $-b$ or b is $2/2b = 1/b$. We scale the numbers $h(\alpha)$ by a factor b to compensate for this. We then interpolate like before to find the value α_{est} such that we would expect $h(\alpha_{est}) = T/2$ if we were allowed to ask for decryptions of $D_{\alpha_{est}}$ for $\alpha \in \mathbb{R}$. When the set of values that λ_i can take is symmetric around 0, the expected value of $\sum \lambda_i e_{i,1}$ is 0. The equation we get for α_{est} is then simplified to

$$b\alpha_{est} = 2^{j-2} + \epsilon, \tag{6}$$

where $|\epsilon|$ is small. Note that we do not need to distinguish between the cases $\lambda_k = -b$ and $\lambda_k = b$, as this is incorporated in the probability $2/2b$ for having the possibility of 1-decryption. So the α_{est} we find covers both the cases $-b\alpha < -2^{j-2}$ and $b\alpha > 2^{j-2}$, which both result in 1-decryptions.

When decrypting $R_{a,i}$ we can have $\lambda_i = 0$, so the probability of $\lambda_i = -b$ or $\lambda_i = b$ is then $2/(2b+1)$. We ask T times for decryptions of $R_{a,i}$, and as before find the value a_{est} that is the best estimate for $\frac{2b+1}{2}h_i(a_{est}) = T/2$. We then get the equation

$$ba_{est} + be_{i,1} = 2^{j-2} + \epsilon_i, \tag{7}$$

where $|\epsilon_i|$ is small. Subtracting (7) from (6) gives us

$$e_{i,1} = \alpha_{est} - a_{est} + \frac{\epsilon_i - \epsilon}{b}.$$

Rounding this value to the nearest integer recovers the correct $e_{i,1}$, provided T is large enough to make $|\epsilon_i| < b/4$ and $|\epsilon| < b/4$.

The proof of the following lemma is almost identical to Lemma 3 and is thus omitted. In fact, the upper bound for $Pr[|\epsilon_i| \geq b/4]$ and $Pr[|\epsilon| \geq b/4]$ will be even smaller than 2^{-20}.

Lemma 5. *If $T \geq 800 \cdot t \cdot \sigma^2$ then in Eq.* (6) *we have that $Pr[|\epsilon| \geq 1/8] \leq 2^{-20}$ and in Eq.* (7) *we have that $Pr[|\epsilon_i| \geq 1/8] \leq 2^{-20}$.*

4.2 Implementation of the attack

We have implemented the attack and verified that it works as explained. The code for the attack was written in C, and can be found at [17]. The secret $\mathbf{e}_i$-vectors were sampled using the DGS library [4].

For testing the attack we have used a Dell server with 75 CPU cores (AMD Epyc 7451). We ran the attack twice, using the parameter sets deduced in Section 3.1. For the first attack we used $t = 190, m = 525, b = 2$, and $\sigma = 25$, which gives the sample size $T = 95,000,000$ according to Lemma 3. For the second attack we used $t = 400, m = 525, b = 2$, and $\sigma = 25$, for which Lemma 3 gives the sample size $T = 200,000,000$. In both attacks we drew the λ_i's uniformly from $\{0, 1\}$ and aimed to recover all the 525 coefficients of $\mathbf{e}_2$[6].

In the attack on the $t = 190$ case, 519 of the 525 coefficients were recovered correctly. The six coefficients that were wrong all had a difference 1 with the correct value. In the attack on the $t = 400$ case, all 525 coefficients of $\mathbf{e}_2$ were recovered correctly.

The run time for the first attack was approximately 12 hours, and for the second attack approximately 48 hours, both using 75 CPU cores in parallel. However, the code can be optimised in several ways to reduce the run time. In particular, it is possible to abort early and not do all T decryptions when it is clear that $h(\alpha)$ or $h_i(a)$ will be much greater or smaller than $T/2$. We did not implement this optimisation, and computed all T decryptions every time.

Our attack does not necessarily recover a secret key flawlessly, as demonstrated above with the $t = 190$ case, where 6 out of 525 estimated coefficients were either -1 or 1 off from their true value. In these cases, we need a second phase to recover the entire secret key. It is straightforward to check whether such a second phase is necessary, as we may simply check if $\mathbf{A}\tilde{\mathbf{s}_i} = 0$, for an estimated secret key $\tilde{\mathbf{s}_i} = (\mathbf{r}_i, -\tilde{\mathbf{e}_i})^T$, where $\tilde{\mathbf{e}_i}$ is the estimate of $\mathbf{e}_i$ we get by running the attack. If $\mathbf{A}\tilde{\mathbf{s}_i} \neq 0$, there is at least one wrong entry of $\tilde{\mathbf{e}_i}$, implying $\mathbf{e}_i = \tilde{\mathbf{e}_i} + \epsilon$, where ϵ is a non-zero vector sampled from a Gaussian distribution with mean 0 and a very small standard deviation. Recall that the public key is $\mathbf{A} = [\mathbf{u}_1 \parallel \ldots \parallel \mathbf{u}_t \parallel \mathbf{B}]$, where $\mathbf{u}_i = \mathbf{B}\mathbf{e}_i$, so we can calculate $\mathbf{B}\epsilon = \mathbf{B}\tilde{\mathbf{e}_i} - \mathbf{u}_i$. Now, $\mathbf{B}\epsilon$ may be described as n highly unbalanced instantiations of the knapsack problem: only approximately 1% of the coefficients of ϵ are either -1 or 1. These instantiations are much simpler to solve than the standard knapsack

[6] We chose $\mathbf{e}_2$ arbitrarily; the attack works to recover any $\mathbf{e}_i$, $i \in \{1, \ldots, t\}$.

problem: one estimation for the time complexity for the $\mathbf{B}\epsilon$ case is $\tilde{O}(2^{0.03n})$[6], another is $\tilde{O}(2^{0.0473n})$[12], though the first algorithm does not guarantee finding the solution. Even though the memory requirement is higher in either case, the (potential) second phase of the attack does not contribute in any substantial way to the cost of the key recovery attack, and ensures that the secret key is completely recovered.

4.3 $\boldsymbol{\lambda_i}$ drawn from a non-uniform distribution

Whilst the authors of LGM mainly focus on the $\lambda_i \in \{0, 1\}$ case, they also discuss other distributions it would be possible to sample from, e.g., other uniform distributions, or a discrete Gaussian distribution. As shown above, sampling λ_i from other uniform distributions does not prevent our attack, however, a line of argument in the LGM paper suggests that a particular choice of a discrete Gaussian distribution might.[7]

The argument is as follows: if the values of $\langle \mathbf{C}_I, \sum_{i=1}^{t} \mathbf{s}_i \rangle$ resemble samples from a discrete Gaussian distribution, and the standard deviation σ' of the λ-distribution satisfies the condition of Theorem 2, the theorem itself is applicable to the distribution of the values of $\langle \mathbf{C}_I, \sum_{i=1}^{t} \lambda_i \mathbf{s}_i \rangle$. Then, seeing as $\langle \mathbf{C}_I, \sum_{i=1}^{t} \lambda_i \mathbf{s}_i \rangle$ is statistically close to a 'regular' discrete Gaussian distribution with standard deviation $\sigma\sigma'$ by Theorem 2, the result from the decryption oracle cannot leak any information about the secret key.

A rough estimate for the parameters required to achieve 120-bit security in these cases is: $t = 400, \sigma = 25, B = 150, \sigma' = 12,231, m = 940$, and q is a 40-bit number. These parameters prevent any practical use of the system, especially given the fact that the scheme encrypts a single bit at a time.

But that aside, could the scheme with this λ-distribution be regarded as a theoretical construct to demonstrate that IND-CCA1 security is achievable for homomorphic encryption schemes? We argue that the answer is no. Even if the λ-distribution is chosen according to Theorem 2, the theorem only guarantees that the distribution over $\langle \mathbf{C}_I, \sum_{i=1}^{t} \lambda_i \mathbf{s}_i \rangle$ is statistically close to a discrete Gaussian distribution *if* the matrix column $\mathbf{C}_I$ is such that the values of $\langle \mathbf{C}_I, \sum_{i=1}^{t} \mathbf{s}_i \rangle$ appear to be drawn from a discrete Gaussian distribution themselves.

We stress that the choice of $\mathbf{C}_I$ is *entirely* up to the adversary in an IND-CCA1 game, as she can simply submit a ciphertext matrix where every column is $\mathbf{C}_I$. It is therefore feasible for her to submit a $\mathbf{C}_I$ such that the values of $\langle \mathbf{C}_I, \sum_{i=1}^{t} \mathbf{s}_i \rangle$ do not appear to be drawn from a discrete Gaussian distribution, meaning Theorem 2 does not apply. It is therefore not possible to positively conclude that no useful information is leaked by a decryption query, even if it merely results in an adversary obtaining a non-negligible advantage in the IND-CCA1 game, and not a complete recovery of the secret key. Furthermore, the adversary can adapt her choice of $\mathbf{C}_I$, whilst the choice of the distribution from which the λ_is are drawn is fixed when the system is generated. The λ-distribution therefore cannot be constructed to fit both the situation where $\mathbf{C}_I$ is designed

[7] See discussion in Section 7 of [13].

to provide values of $\langle \mathbf{C}_I, \sum_{i=1}^{t} \mathbf{s}_i \rangle$ seemingly drawn from a discrete Gaussian distribution and when it is designed to *not* provide such values.

4.4 Thwarting the attack

We discuss some possible ideas to prevent the key recovery attack described above, and argue that they will not work.

Decryption oracle uses the same $(\lambda_1, \ldots, \lambda_t)$ for the same ciphertext. One can ensure that an attacker that queries the same ciphertext C (say, $C = D_\alpha$ as defined previously) multiple times will have the same set of λ-values chosen in every decryption. It will then be impossible to do the attack we presented, as it relies on having random and independent λ-values chosen for each query. This can be done by setting $(\lambda_1, \ldots, \lambda_t) = \mathsf{PRF}(C)$ for some pseudo-random function PRF that returns a vector of small values.

To circumvent this measure, the attacker can add a few 1's to the large part of the top t rows of D_α or $R_{a,i}$ that are defined to be 0. The number of 0's in this part of D_α or $R_{a,i}$ is $t(t-1)$. So if the attacker intends to ask for T decryptions of the same matrix, she can make the matrices unique by adding up to ρ 1's in all possible ways in the top t rows. The computation in the decryption will still be approximately the same. The largest number ρ of 1's that must be added in this way is the smallest integer that satisfies

$$\sum_{i=0}^{\rho} \binom{t(t-1)}{i} \geq T.$$

For the parameters used in our attack ($t = 190, T = 200.000.000$) this is satisfied already for $\rho = 2$.

When adding two 1's to D_α or $R_{a,i}$, the estimates in Eq. (1) and Eq. (2) will be disturbed by an extra 1 in approximately $(2/t) \cdot (1/2) = 1/t$ of the queries (the chosen column contains an extra 1 with probability $2/t$, and the λ_j-value it meets in the inner product will also be 1 with probability $1/2$). This error can be compensated for in the estimation of α_{est} and a_{est} by adding an extra term to the equations in Eq. (1) and Eq. (2), but both values will be the same and anyway cancel out in Eq. (3). Hence the attacker can overcome such a countermeasure.

Repeat multiple decryptions and return a value only if they are consistent. Alternatively, one can define a new decryption function which runs the original decryption function ℓ times and return bit b only if all ℓ evaluations return b, and abort (i.e., return $\perp$) if they are not all equal. However, note that we now have 3 return values $(0, 1, \perp)$ instead of 2, and can compute the expected value of each return value for every α similar to before.

For instance, consider querying D_α to the new decryption oracle, where α is a value such that the original decryption oracle would return 0 with probability p, and 1 with probability $1 - p$. Then the new decryption oracle aborts with probability $1-(p^\ell+(1-p)^\ell)$, which achieves maximum at $p=1/2$. Hence to detect an optimal α_{est} as in Eq. (1), one asks for T decryptions of $D_{\alpha_{est}}$ and makes sure $\approx (1-2^{1-\ell})\cdot T$ of them abort. Note that the attack strategy fails if ℓ is sufficiently large (e.g., $\ell \in \Theta(\kappa)$), but a large choice of ℓ also severely restricts the LGM scheme's level of homomorphism, since a noisy ciphertext obtained from a homomorphic evaluation would also fail (i.e., return $\perp$ or the wrong bit) with non-negligible probability in the new decryption function.

Ciphertext checks. A plausible strategy to thwart our attack would be to add a ciphertext check during decryption, to ensure that the ciphertext to be decrypted has been honestly generated. Using a ciphertext check, Loftus et al. [15] constructed an SHE scheme that provably achieved IND-CCA1 security, although the underlying hardness assumption was later shown to be insecure; see discussion in [13] and the references therein. If such a ciphertext check is added to the decryption procedure, maliciously generated ciphertexts may simply be rejected by the decryption oracle, which will make it impossible to mount our attack. As illustrated by the previous idea, it is far from clear how to successfully add an efficient ciphertext check to the LGM scheme.

We argue that in general any such ciphertext check that uses the same secret key value both to check ciphertexts are well-formed and to decrypt will naturally give some information about the secret key. One can instead have two secret key values as in the CCA1-secure group homomorphic encryption scheme CS-lite [9], where the first value is used only for checking ciphertexts are well-formed, while the second value ensures indistinguishability even if the first value is revealed. We leave as an open problem how such a method can work with LGM or other homomorphic encryption schemes.

5 Conclusion

We have shown that the LGM scheme is susceptible to an adaptive key recovery attack, disproving the authors' claim that the scheme achieves IND-CCA1 security. The attack is practical for λ_i's drawn uniformly from $\{0,1\}$, and is still practical and efficient for λ_i's drawn uniformly from a larger set of integers. We have also argued that the scheme is not secure even if the λ_i's are drawn from a discrete Gaussian distribution. In short, none of the distributions suggested by Li et al. ensures the IND-CCA1 security of the LGM scheme.

A plausible strategy to thwart our attack would be to add a ciphertext check during decryption, but we do not know if the strategy can be applied to the LGM scheme, and we know of no other strategies that may be applicable to the scheme to achieve IND-CCA1 security. We therefore do not know how to tweak the LGM scheme to be resistant to our proposed statistical attack.

References

1. Agrawal, S., Boneh, D., Boyen, X.: Efficient lattice (H)IBE in the standard model. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 553–572. Springer, Heidelberg (May / Jun 2010)
2. Agrawal, S., Gentry, C., Halevi, S., Sahai, A.: Discrete Gaussian leftover hash lemma over infinite domains. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT 2013, Part I. LNCS, vol. 8269, pp. 97–116. Springer, Heidelberg (Dec 2013)
3. Albrecht, M.R., Player, R., Scott, S.: On the concrete hardness of learning with errors. Cryptology ePrint Archive, Report 2015/046 (2015), `http://eprint.iacr.org/2015/046`
4. Albrecht, M.R., Walter, M.: dgs, Discrete Gaussians over the Integers (2018), available at `https://bitbucket.org/malb/dgs`
5. Bai, S., Galbraith, S.D., Li, L., Sheffield, D.: Improved combinatorial algorithms for the inhomogeneous short integer solution problem. Journal of Cryptology 32(1), 35–83 (Jan 2019)
6. Becker, A., Coron, J.S., Joux, A.: Improved generic algorithms for hard knapsacks. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 364–385. Springer, Heidelberg (May 2011)
7. 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)
8. 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)
9. 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)
10. 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)
11. 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)
12. Howgrave-Graham, N., Joux, A.: New generic algorithms for hard knapsacks. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 235–256. Springer, Heidelberg (May / Jun 2010)
13. 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`
14. 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)
15. 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)
16. Micciancio, D., Walter, M.: Gaussian sampling over the integers: Efficient, generic, constant-time. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017, Part II. LNCS, vol. 10402, pp. 455–485. Springer, Heidelberg (Aug 2017)

17. Raddum, H., Fauzi, P.: LGM-attack (2021), available at `https://github.com/Simula-UiB/LGM-attack`
18. Zheng, Z., Xu, G., Zhao, C.: Discrete Gaussian measures and new bounds of the smoothing parameter for lattices. Cryptology ePrint Archive, Report 2018/786 (2018), `https://eprint.iacr.org/2018/786`

Article III

2.3 On the IND-CCA1 Security of FHE Schemes

Prastudy Fauzi, Martha Norberg Hovd and Håvard Raddum

In Cryptology ePrint Archive, Report 2021/1624, 2021. `https://eprint.iacr.org/2021/1624`.

The majority of the work and writing was done by the candidate. The editing was split equally between the authors.

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: $p = r \underline{\bmod} q$ 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 $p \equiv r \underline{\bmod} q$ 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 $(c \underline{\bmod} p) \mod 2$.

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 $g^{-1} : \mathbb{Z} \rightarrow \{0,1\}^\gamma$ 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 $\mathbf{c} = m\mathbf{g}^T + \mathbf{x}\mathbf{S} \ \underline{\text{mod}}\ x_0$.

Dec$(p, \mathbf{c})$: Compute $\mu = \mathbf{c}g^{-1}(p/2) \ \underline{\text{mod}}\ p$. 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 $\mathbf{K}^{-1} \bmod x_0$ 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 $\mathbf{C} = (\mathbf{X} + \mathbf{GKM})\mathbf{K}^{-1} \ \underline{\text{mod}}\ x_0$, and output $\mathbf{C}$.

DecMat$(sk, \mathbf{C})$: Compute $\mathbf{C}' = G^{-1}(\alpha\mathbf{K}^{-1})\mathbf{CK} \ \underline{\text{mod}}\ x_0$, then $\mathbf{C}^* = \mathbf{C}' \ \underline{\text{mod}}\ p$, 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 $\mathbf{c}' = \mathbf{cK} \ \underline{\text{mod}}\ x_0$, then $\mathbf{c}^* = \mathbf{c}' \ \underline{\text{mod}}\ p$. 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 $f = pf' + 1 \underline{\bmod} q$, 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 $c = \lfloor q/p \rfloor m + r + he \underline{\bmod} q$ as an element of R.

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

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 $C(y) = 2r(\phi(y)) + m(\phi(y)) \underline{\mathrm{mod}}\ F(y)$.

Dec($C(y)$): Replace y by $\psi(x)$ in the polynomial C and output $(C(\psi(x)) \underline{\mathrm{mod}}\ f(x)) \mod 2$.

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 $(\boldsymbol{A}, \boldsymbol{As}+\boldsymbol{e})$ 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 $(\boldsymbol{A}, \boldsymbol{As})$, 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 $\mathbf{A} = [\mathbf{u}_1 \| \dots \| \mathbf{u}_t \| \mathbf{B}] \in \mathbb{Z}_q^{n\times(t+m)}$ and the secret key $\mathbf{s} = (\mathbf{s}_1, \dots, \mathbf{s}_t)$.

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 $(\lambda_1, \dots, \lambda_t) \in \mathbb{Z}_q^t \setminus \{0\}^t$ 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 $\mu = \mathbf{c}g^{-1}(p/2)$ $\underline{\mathrm{mod}}$ 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 $\underline{\mathrm{mod}}$ 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 $\mathbf{C}' = G^{-1}(\alpha\mathbf{K}^{-1})\mathbf{C}\mathbf{K}$ $\underline{\mathrm{mod}}$ x_0, then $\mathbf{C}^* = \mathbf{C}'$ $\underline{\mathrm{mod}}$ 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 $\underline{\mathrm{mod}}$ $p = d$. Irrespective of the value of d, we let α^{-1} denote the integer such that $\alpha^{-1}\alpha \equiv d$ $\underline{\mathrm{mod}}$ 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 $\mathbf{C}^* = (d \cdot \mathbf{I}_n)$ $\underline{\mathrm{mod}}$ $p = d \cdot \mathbf{I}_n$ 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 $\lfloor b_{i,j} \begin{bmatrix} k_{i,1} \cdots k_{i,n} \end{bmatrix} / \alpha \rceil$, 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:

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

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 $\lfloor (b_{i',j'}k_{i,j} \underline{\text{mod}}\ p)/\alpha \rceil$ 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 $\lfloor (b_{1,1}k_{1,1} \underline{\text{mod}}\ p)/\alpha \rceil$ is the smallest positive value among the values $\lfloor (b_{i,1}k_{1,1} \underline{\text{mod}}\ p)/\alpha \rceil$ that the attacker sees, which will be in the range $[-511, 512]$.

We then know that $\lfloor (ab_{1,1}k_{1,1} \underline{\text{mod}}\ p)/\alpha \rceil \approx a\lfloor (b_{1,1}k_{1,1} \underline{\text{mod}}\ p)/\alpha \rceil$ 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

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

Repeating with different $k_{j,1}$ produces n different estimates of this ratio, which should give approximately the same value for all cases where $\lfloor (b_{1,1}k_{j,1} \underline{\text{mod}}\ p)/\alpha \rceil$ 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 $k_2a < 4q \leq (k_2+1)a$, etc. After t iterations we have the following inequalities

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

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

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

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 Dec($C(y)$)= $(C(\psi(x)) \underline{\bmod}\ f(x)) \mod 2$, 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 $\psi(x) = a_{n-1}x^{n-1} + \ldots + a_1x + a_0$ 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 $\to$ 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 $c = \mathrm{Enc}_{\mathrm{IBHE}}(mpk, id, m)$, and output (c, id).

Dec($msk, (\boldsymbol{c}, id)$): Parse $\boldsymbol{c} = (c, id)$, compute $sk_{id} = \mathrm{Ext}(id, msk)$, and output $m = \mathrm{Dec}_{\mathrm{IBHE}}(sk_{id}, id, c)$.

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 $(\mathrm{Enc}(pk_1, m), \mathrm{Enc}(pk_2, m), \pi)$, 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 $\mathcal{E} = (\mathrm{KeyGen}, \mathrm{Enc}, \mathrm{Dec}, \mathrm{Eval})$ 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[$\mathbb{A}^*$]
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 $d$
```

Article IV

2.4 Vetted Encryption

Martha Norberg Hovd and Martijn Stam

The development of the systems, writing and editing was split equally between the authors.

A shorter version of the paper appeared in Karthikeyan Bhargavan, Elisabeth Oswald and Manoj Prabhakaran, editors, *INDOCRYPT 2020*, volume 12578 of *LNCS*, pages 488-507. Springer, Heidelberg, December 2020. Presented by the candidate at *Indocrypt* in December 2020.

Version reproduced here in Cryptology ePrint Archive, Report 2020/1348, 2020. `https://eprint.iacr.org/2020/1348`.

Vetted Encryption

Martha Norberg Hovd[1,2] and Martijn Stam[1]

[1] Simula UiB
Merkantilen (3rd floor)
Thormøhlensgate 53D
N-5006 Bergen, Norway.
`martha,martijn@simula.no`
[2] University of Bergen
Høyteknologisenteret i Bergen
Thormøhlensgate 55
N-5008 Bergen, Norway.

Abstract. We introduce Vetted Encryption (VE), a novel cryptographic primitive, which addresses the following scenario: a receiver controls, or vets, who can send them encrypted messages. We model this as a filter publicly checking ciphertext validity, where the overhead does not grow with the number of senders. The filter receives one public key for verification, and every user receives one personal encryption key.
We present three versions: Anonymous, Identifiable, and Opaque VE (AVE, IVE and OVE), and concentrate on formal definitions, security notions and examples of instantiations based on preexisting primitives of the latter two. For IVE, the sender is identifiable both to the filter and the receiver, and we make the comparison with identity-based signcryption. For OVE, a sender is anonymous to the filter, but is identified to the receiver. OVE is comparable to group signatures with message recovery, with the important additional property of confidentiality of messages.

Keywords: Encryption · Group Signatures · Signcryption

1 Introduction

Spam and phishing messages are a bane of modern communication methods, especially email. These days, most email still happens in the clear without end-to-end cryptographic protection. Yet, there are standards, such as S/MIME and OpenPGP, that aim to secure email using a combination of public key and symmetric key confidentiality and authentication primitives. Intuitively, the primitive that best models secure email is signcryption [46, 28]. Although signcryption allows receivers to verify *locally* whether an email was from its purported sender or not, this ability does not immediately lead to an efficient mechanism to filter spam centrally.

A different, though not completely unrelated, scenario arrises with electronic voting systems and eligibility verifiability. This notion informally states that it should be possible to publicly verify that only those with the right to vote have done so. For obvious reasons, voters should still be anonymous, and so whitelisting is not a viable option to prevent ballot stuffing by the bulletin board, for instance.

In this work we propose an alternative primitive called *vetted encryption*, which is closely related to both signcryption and group signatures. Vetted encryption lets a user, the *recipient*, to restrict who can send them encrypted messages by enabling an *outside filter* to detect which users are and are not vetted. The key features of vetted encryption are that a recipient only needs to vet each sender once (with out-of-band communication), yet does *not* need to tell the filter which users they have vetted.

Vetted encryption comes in different flavours, depending on whether senders should be identified and authenticated or, in contrast, should remain anonymous. This choice of authentication versus anonymity can be made with respect to the outside filter and the intended receiver independently of each other, leading to a total of four possible configurations. One configurarion, where the filter would learn the identity of a ciphertext, yet the receiver could not, runs counter to our perspective that the filter is working on behalf of the recipient. Thus, only three settings remain:

1. *Anonymous vetted encryption (AVE)* where the sender remains anonymous to both the filter and the recipient; this scenario can be relevant for a voting system using a bulletin board, on which only eligible users should be able to post, anonymously. For example, the system Belenios [24] applies signatures and credentials to attain eligibility verifiability, which is not too dissimilar from AVE.
2. *Identifiable vetted encryption (IVE)* where the sender is identified for both the filter and the recipient; this scenario is typical for email spam, where the filter gets to see the email-address (or other identifying information) of the sender.

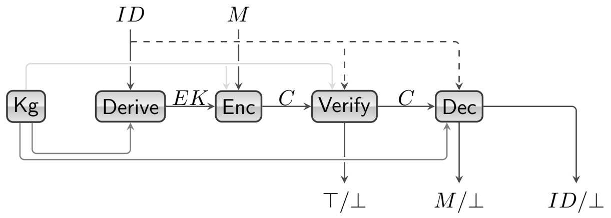

Fig. 1. The algorithms and options involved in vetted encryption for the three options: anonymous includes neither dashed nor solid blue lines; identifiable adds the dashed blue lines only; opaque adds the solid blue lines only and distinguishes between the two red "secret" master keys for derivation resp. decryption.

3. *Opaque vetted encryption (OVE)* where the sender is anonymous to the filter, yet can be identified by the recipient. This primitive is relevant for identifiable communication over an anonymous channel, for example between a trusted anonymous source and a journalist, where the source is anonymous to the newspaper, but identifiable to the reporter. OVE may also be used in an auction setting, where the seller vets who gets to bid. During bidding, the auctioneer may filter the bids, only forwarding bids from vetted participants. However, only the seller knows the identity of the active bidders in the auction.

Our contribution. Fig. 1 provides an overview of the algorithms that constitute a vetted encryption scheme, where we endeavoured to surface all three variants in the picture. The private key material from the key generation (the red lines emanating at the bottom) feeds into two distinct functionalities: firstly to vet users by issuing them an encryption key, and secondly to decrypt ciphertexts. Thus one consideration to make is the possible orthogonality of the corresponding private keys. For both AVE and IVE (Def. 2) we opted for the simplest scenario where both keys are identical, whereas for OVE (Def. 3) we opted for the more challenging scenario where the keys are separate. This choice affects the security definitions and the design space for suitable constructions.

AVE is the simplest variant of vetted encryption and we present it in Appendix B, where we also discuss similarities with signcryption. It turns out all senders can be given the same signing key to encrypt then sign a message. Although this mechanism ensures full anonymity, a malicious sender could make everybody a vetted sender by simply forwarding her key. We therefore focus on IVE and OVE here.

IVE is most closely related to a simplified form of identity-based signcryption with public verifiability. As far as we are aware, the related signcryption flavour is virtually unstudied. We give a full comparison of IVE and signcryption in Section 3.3. Our use case for IVE allows us in Section 3 to navigate carefully through the possible definitional choices, esp. quite how much power is available to an adversary trying to break confidentiality, resp. integrity. Our choice allows us to use the novel primitive of "outsider-unique" identity-based signatures, which we show can be constructed by a combination of derandomization and unique signatures (see Appendix A). The unique property is fundamental to provide non-malleability and hence confidentiality against chosen ciphertext attacks for our encrypt-then-IBsign construction (Fig. 6).

OVE bears similarities with group signatures with message recovery, where additionally the message should remain confidential. However, as we will argue in Section 4, our use case allows us to relax security slightly, which in turn enables a slight simplification of the well-known sign-encrypt-proof paradigm for group signatures [13] which we dub *verifiably encrypted certificates* (Fig. 13). We also give a thorough comparison with group signatures in Section 4.3.

1.1 Related Work

Comparison with signcryption. Both AVE and IVE are most closely related to signcryption in its various guises. For the original signcryption concept [46], two users Anna and Bob might want to communicate together in a manner that is simultaneously confidential and authenticated. In a public key setting, if Anna knows Bob's public encryption key and Bob knows Anna's public verification key, then Anna can combine digital signing and public encryption of a message using the signcryption primitive.

Signcryption security is best studied in the multi-user setting, but let us consider just the two user scenario [4]. Confidentiality can be captured by left-or-right indistinguishability under adaptive chosen ciphertext attacks, where an important modelling choice has to be made with respect to the adversary's control over keys. If the adversary is an outsider, only public key information is available. If the adversary knows private keys (e.g. Anna's signing key when attacking confidentiality or Bob's decryption key when attacking authenticity), we speak of insider security. Insider-secure confidentiality is essential to achieve forward secrecy, that is if Anna's signing key gets compromised, past messages should still remain confidential. Insider-secure authenticity is needed for non-repudiation, where receiver Bob can convince a third party a message really originated from Anna (and wasn't cooked up by Bob himself). Indeed, for most realistic use cases, insider security is required [8].

Clearly insider security is harder to achieve than outsider security. The natural way for Anna and Bob to attempt signcryption would be to combine a public key encryption scheme with a digital signature scheme using generic composition. There are essentially two ways of doing so sequentially: either first encrypt and then sign the ciphertext (encrypt-then-sign) or first sign and then encrypt both message and signature (sign-then-encrypt). The third, parallel alternative of encrypting the message and signing the message is more problematic from a generic composition perspective.

Signcryption with public verifiability [32] could be used as an alternative solution to anonymous vetted encryption, but the precise flavour of signcryption needed is not immediate (see Appendix B.3 for details). Signcryption appears to be unsuitable for OVE (Section 4.3): although it is possible to achieve for instance IB-signcryption with anonymity [20, 22] [10, Section 5.4], crucially these schemes cannot support public verifiability, ruling out the ability to outsource their verification to a filter. Signcryption with public verifiability and explicit whitelisting could be used as a less efficient alternative to IVE, in addition to identity based signcryption with transferable public verifiability, see Section 3.3 for further discussions.

Comparison with group signatures. The setting for group signatures is the following: a group, with a single manager, consist of various members, all with their own secret signing key to sign messages. It may be publicly verified that a signature belongs to a member of the group without revealing *which* member has signed the message. Only the group manager, who possesses a secret opening key, may identify the signer, given a signature on a message.

The classic security notions of group signature schemes encompass both anonymity and traceability [13]. Informally, this means that a signature does not reveal the identity of a signer to anyone who does not possess the opening key, and that one cannot forge someone's signature unless one has their secret key.

With regards to AVE and IVE, the comparison is somewhat natural in the big picture: in both cases, the sender has to verify membership of a group (of vetted senders). The similarities end here, though, as the notion of revealing the identity of the sender runs counter to AVE, and hiding the identity from the filter does not line up with the intention of IVE. However, the setting overlaps to a great extent with desireable features of OVE, with the important exception of confidentiality of messages.

A straightforward, but naive, fix to this would be to simply encrypt the message, then sign the ciphertext using the group signature scheme. Although this seems to add confidentiality to the scheme, it also introduces the following weakness: any group member Eve may intercept a ciphertext, (group)signature pair from Alice, sign the ciphertext using her own key, and thus pass Alice's confidential message off as her own. In particular, if Eve has access to a decryption oracle, she may ask to have the ciphertext decrypted, and by that read the message Alice sent. Thus, we provide a construction of OVE motivated by group signatures, rather than using them as a primitive.

Comparison with matchmaking encryption. Matchmaking encryption (ME) allows, in a sense, for a sender and receiver to vet each other: both may specify policies the other party must satisfy in order for the sent message to be revealed to the recipient. The sender may specify what properties the receiver must have in order to read the message, and the receiver may specify the requirements a sender must meet in order to send the receiver a message. Furthermore, the only information leaked is whether or not a policy match occured, that is: whether or not the recipient received the decrypted message [7].

The set up relies on a trusted authority to generate both encryption and decryption keys for the sender and receiver, respectively, both associated with attributes. In addition, there is a decryption key associated with the policy a sender should satisfy, which is also generated by the trusted authority. Finally, a sender can specify a policy which a reciever must satisfy to be able to decrypt the sent message.

There is an identity based version of ME, which bears some reseblance to OVE. In this version, the more general attributes of the sender and receiver are replaced with a simple identity, so that the sender specifies the identity of the desired recipient, and the identity of the sender is an explicit input of the decryption procedure.

The latter point is an important difference to the OVE primitive, where the identity of the sender is an output of decryption, rather than an input. In other words: we do not assume that the reciever knows who has sent a message before it has been decrypted. Another important difference is that we do not allow the sender to demand any certain attributes of the receiver, though the identity of the receiver is indirectly determined by the sender during encryption, as this involves an encryption key unique to the recipient. We also note that in OVE, the keys are derived and distributed by the recipient, not a third party.

Finally: in ME, determining whether or not a match will occur, that is, if the recipient and sender have vetted each other, is not publicly verifiable. This requires the decryption key of the recipent and the key related to the policy of the sender. This is in contrast with OVE, where determining whether a message has been sent from a vetted sender is possible using only a public key.

Comparison with access control encryption. Access control encryption (ACE) allows for different reading and writing rights to be assigned to different senders and receivers, for example the right to read messages classified as 'Secret', and ensuring a sender with clearance 'Top Secret' cannot send messages classified as 'Public'. This is achieved by introducing a sanitizer into the network, who manipulates every message before it is published on the network, we note in particular that a message is not sent directly to its intended recipient. Now, if a recipient tries to decrypt a message he does not have the right to read, the ciphertext will decrypted into a random string [26].

Although both ACE and OVE in some sense deal with the notion of vetting senders, there are several differences. First of all: an honest filter in OVE does not change the ciphertext in any way, it simply checks whether a sender has been vetted. In particular, if a received ciphertext decrypts to gibberish, it is because the sender intended it so. Furthermore: the sender is always identifiable to the receiver, assuming a message was received. Finally, a sent message is forwarded to the intended recipient, as opposed to published on a network.

We also note that the power dynamic in the two primitives differ. In ACE, the sanitizer enforces a security protocol, typically on behalf of a third party, and determines which subset of a public set of messages anyone is able to read. In OVE, however, all power lies with the recipient, by generating and distributing all the keys. The filter is merely doing the recipient's bidding, as it were, by only allowing messages from vetted senders to reach the recipient.

2 Preliminaries

When defining security, we will use concrete advantages throughout. Moreover, these advantages are defined in terms of an adversary interacting with a game or experiment. While these experiments depend on the schemes at hand, there will be no additional quantifications (e.g. over high entropy sources, simulators, or extractors). We use $\Pr[Code : Event]$ to denote probabilities where the *Code* is used to induce a probability distribution over which *Event* is defined (not to be confused with conditional probabilities). We write $\mathbb{A}^{\mathcal{O}}$ for an adversary $\mathbb{A}$ having access to oracle(s) $\mathcal{O}$ in security games and reductions.

We use a number of standard primitives and their associated security notions. For completeness, and for the avoidance of any ambiguities in our notation, these are recapitulated below.

- A *public key encryption* scheme PKE consists of a triple of algorithms (Pke.Kg, Pke.Enc, Pke.Dec). The default security notion we consider is single-user multi-query left-or-right indistinguishability under chosen ciphertext attacks (IND-CCA).
- A *signature scheme* SIG consists of a triple of algorithms (Sig.Kg, Sig.Sign, Sig.Verify). The default security notion we consider is single-user strong existential unforgeability under chosen message attacks (EUF-CMA). We often require the SIG to be *unique* (USS), which means that given the verification key, for every message there is only a single signature that verifies.
- A *signcryption* scheme SCR consist of six algorithms (Scr.Kgr, Scr.Kgs, Scr.Signcrypt, Scr.Verify, Scr.Unsigncrypt), where Scr.Kgr generates the receiver's keys and Scr.Kgs the sender's keys.
- An *identity-based signature scheme* IBS consists of the four algorithms (Ibs.Kg, Ibs.Derive, Ibs.Sign, Ibs.Verify), where Ibs.Kg derives a master signing key MSK and a verification key VK. The derivation algorithm

Ibs.Derive takes the master signing key and an identity ID as input, and outputs a user signing key USK. The signing takes a message M, an identity ID and a user signing key USK as input, and outputs a signature σ. Finally, verification takes the verification key VK, a message M, an identity ID and a signature σ as input, and outputs $\top$ or $\bot$. As with signature schemes, we consider EUF-CMA as the default security notion for IBS schemes.

QA-NIZKs. Non-Interactive Zero-Knowledge (NIZK) proofs are defined for families of languages with associated binary relations R, such that for pairs $(\phi, \omega) \in R$ a prover may convince a verifier that the statement ϕ is part of the language, without revealing anything else (such as the witness ω). For the proof to be non-interactive, we require that the only necessary communication between the prover and verifier is the sending of the proof π. This non-interaction requirement disregards the communication required for the set-up of the scheme, which involves the prover and verifier sharing a common reference string (CRS). For Quasi-Adaptive NIZKs (QA-NIZKs) [35], we allow this CRS to depend on the parameters defining the language and its witness relation R. In the following, we let the relation R be given as input to the set-up algorithm and various adversaries, this is to be understood as the parameters defining said relation. Moreover, we let $\mathcal{R}$ be a distribution over the family of languages for which the NIZK is suited.

Definition 1 (Quasi-Adaptive Non-Interactive Zero-Knowledge (QA-NIZK) proofs). *An efficient prover publicly verifiable* Quasi-Adaptive Non-Interactive Zero-Knowledge *(QA-NIZK) for $\mathcal{R}$ is a quadruple of probablilistic algorithms* (Nizk.Setup, Nizk.Prove, Nizk.Verify, Nizk.Sim) *such that*

- Nizk.Setup *produces a CRS σ and a simulation trapdoor τ for the relation R:* $(\sigma, \tau) \leftarrow\$ \text{Nizk.Setup}(R)$.
- Nizk.Prove *takes as input a CRS σ and a tuple $(\phi, \omega) \in R$ and returns a proof π:*
  $\pi \leftarrow\$ \text{Nizk.Prove}(\sigma, \phi, \omega)$
- Nizk.Verify *either rejects ($\bot$) or accepts ($\top$) a proof π for a statement ϕ when given these, as well as a CRS σ:*
  $\top/\bot \leftarrow \text{Nizk.Verify}(\sigma, \phi, \pi)$.
- Nizk.Sim *takes as input a simulation trapdoor τ, and a statement ϕ and returns a proof π:* $\pi \leftarrow\$ \text{Nizk.Sim}(\tau, \phi)$.

Completeness. The notion of *perfect completeness* states that, for any true statement ϕ, an honest prover should be able to convince an honest verifier. More formally, we require that for all $R \in \mathcal{R}$ and $(\phi, \omega) \in R$:

$$\Pr[(\sigma, \tau) \leftarrow\$ \text{Nizk.Setup}(R); \pi \leftarrow\$ \text{Nizk.Prove}(\sigma, \phi, \omega) \ : \ \text{Nizk.Verify}(\sigma, \phi, \omega) \rightarrow \top] = 1 \ .$$

Soundness. For a QA-NIZK to achieve *computational soundness*, we require that it is computationally infeasible for an adversary $\mathbb{A}$ given the relation R and the CRS σ, to output a pair (ϕ, π) that satisfy the following conditions: 1) ϕ does not lie in the language defined by R, that is: there does not exist a witness $\bar{\omega}$ such that $(\phi, \bar{\omega}) \in R$, and 2) $\text{Nizk.Verify}(\sigma, \phi, \omega) \rightarrow \top$. Formally, we define the advantage:

$$\text{Adv}^{\text{sound}}_{\text{QANIZK}}(\mathbb{A}) = \Pr\left[\begin{array}{c} R \leftarrow\$ \mathcal{R} \\ (\sigma, \tau) \leftarrow\$ \text{Nizk.Setup}(R) \\ (\phi, \pi) \leftarrow\$ \mathbb{A}(R, \sigma) \end{array} : \ \phi \notin L_R \land \text{Nizk.Verify}(\sigma, \phi, \omega) \rightarrow \top \right] .$$

Zero-knowledge. Informally, a QA-NIZK is *zero-knowledge* if nothing other than the truth of the statement may be inferred by the proof. We formally define the distinguishing advantage using a real and a sim experiment (Fig. 2), and define the advantage of the adversary $\mathbb{A}$ as

$$\text{Adv}^{\text{zk}}_{\text{QANIZK}}(\mathbb{A}) = \Pr\left[\text{Exp}^{\text{zk-real}}_{\text{QANIZK}}(\mathbb{A}) \ : \ \hat{b} = 0\right] - \Pr\left[\text{Exp}^{\text{zk-sim}}_{\text{QANIZK}}(\mathbb{A}) \ : \ \hat{b} = 0\right] .$$

We speak of *perfect* zero-knowledge if $\text{Adv}^{\text{zk}}_{\text{QANIZK}}(\mathbb{A}) = 0$ for all adversaries. Perfect zero-knowledge can alternatively be characterized with a single query and a universal quantifier for the choice of language and statement to prove. Many known NIZKs achieve perfect zero-knowledge, facilitating their composability.

$\mathsf{Exp}^{\text{zk-real/sim}}_{\text{QANIZK}}(\mathbb{A})$	prove-real(ϕ, ω)	prove-sim(ϕ, ω)
$R \leftarrow\$ \mathcal{R}$	**require** $(\phi, \omega) \in R$	**require** $(\phi, \omega) \in R$
$(\sigma, \tau) \leftarrow\$ \mathsf{Nizk.Setup}(R)$	$\pi \leftarrow\$ \mathsf{Nizk.Prove}(\sigma, \phi, \omega)$	$\pi \leftarrow\$ \mathsf{Nizk.Sim}(\tau, \phi)$
$\hat{b} \leftarrow \mathbb{A}^{O}(R, \sigma)$	**return** π	**return** π

Fig. 2. The real and simulated zero-knowledge experiments.

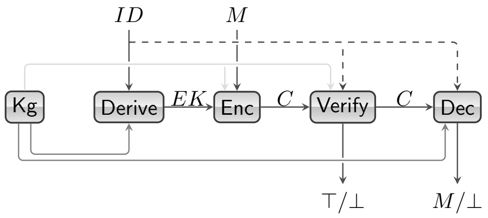

Fig. 3. The algorithms and their inputs/outputs for identifiable vetted encryption.

Unbounded simulation-soundness. A QA-NIZK achieves *unbounded simulation-soundness* if an adversary $\mathbb{A}$ is unable to simulate proofs of any false statement, even after having seen such proofs of arbitrary statements. We define the advantage of the adversary $\mathbb{A}$ as

$$\mathsf{Adv}^{\text{uss}}_{\text{QANIZK}}(\mathbb{A}) = \Pr\left[\begin{array}{c} R \leftarrow\$ \mathcal{R} \\ (\sigma, \tau) \leftarrow\$ \mathsf{Nizk.Setup} \\ (\phi, \pi) \leftarrow \mathbb{A}^{\mathsf{Nizk.Sim}(\sigma,\tau,\cdot)}(R, \sigma) \end{array} : \begin{array}{c} (\phi, \pi) \notin Q \wedge \phi \notin L_R \\ \wedge \mathsf{Nizk.Verify}(\sigma, \phi, \pi) \longrightarrow \top \end{array}\right],$$

where Q is the set of query–response pairs (ϕ, π) to the simulator.

After the introduction of QANIZK protocols [35], a large number of protocols for a large variety of languages (or distributions thereof) has appeared in the literature. They are particularly efficient for linear subspaces, which facilitates pairing based constructions (see [3] and the references contained therein).

3 Identifiable Vetted Encryption (IVE)

3.1 Syntax and Security of IVE

The algorithms. For identifiable vetted encryption, both the filter and the recipient may learn the identity of the sender, which we assume have received the identity via out-of-band communication. An IVE scheme consists of five algorithms, see Def. 2. The identity *ID* is not only an explicit input to the derivation algorithm, but also to both the verification and decryption algorithm, modelling the out-of-band communication. However, encryption does not take *ID* as an input, instead relying on a user's encryption key *EK* implicitly encoding said identity.

We allow encryption to fail, modelled by $\perp$ as output. As we will see, for honestly generated encryption keys, we insist encryption never fails, but for adversarially generated encryption keys, allowing for explicit encryption failure turns out to be useful. One could alternatively introduce a separate algorithm to verify the validity of a private encryption key for a given public encryption/verification key; our approach looks simpler.

Definition 2 (Identifiable Vetted Encryption (IVE)). *An* identifiable vetted encryption *scheme* IVE *consists of a 5-tuple of algorithms* (Ive.Kg, Ive.Derive, Ive.Enc, Ive.Verify, Ive.Dec)*, which behave as follows:*

– Ive.Kg generates *a key pair* (PK, SK)*, where* PK *is the public encryption (and verification) key and* SK *is the private derivation and decryption key. We allow* Ive.Kg *to depend on paramaters* $param$ *and write* $(PK, SK) \leftarrow\$ \mathsf{Ive.Kg}(param)$*. Henceforth, we will assume that* PK *can be uniquely and efficiently computed given* SK.
– Ive.Derive derives *an encryption key* EK *based on the private derivation key* SK *and a user's identity* ID*. Thus,* $EK \leftarrow\$ \mathsf{Ive.Derive}_{SK}(ID)$.

– Ive.Enc encrypts *a message M given the public encryption key PK and using the private encryption key EK, creating a ciphertext C or producing a failed encryption symbol $\perp$. So, $C \leftarrow_\$ \mathsf{Ive.Enc}_{PK,EK}(M)$ where possibly $C = \perp$.*
– Ive.Verify verifies *the validity of a ciphertext C given the public verification key PK and a user's identity ID. With a slight abuse of notation, $\top/\perp \leftarrow \mathsf{Ive.Verify}^{ID}_{PK}(C)$.*
– Ive.Dec decrypts *a ciphertext C using the private key SK, given the user's identity ID. The result can either be a message M or the invalid-ciphertext symbol $\perp$. In short, $M/\perp \leftarrow \mathsf{Ive.Dec}^{ID}_{SK}(C)$.*

The first three algorithms are probabilistic, the final two deterministic.

Correctness and consistency. For correctness, we require that all honestly generated ciphertexts are received as intended, that is, for all parameters *param*, identities ID and messages M, we have that

$$\Pr\left[\begin{array}{ccc} (PK, SK) \leftarrow_\$ \mathsf{Ive.Kg}(param) & & C \neq \perp \\ EK \leftarrow_\$ \mathsf{Ive.Derive}_{SK}(ID) & : & \wedge\, \mathsf{Ive.Verify}^{ID}_{PK}(C) = \top \\ C \leftarrow_\$ \mathsf{Ive.Enc}_{PK,EK}(M) & & \wedge\, \mathsf{Ive.Dec}^{ID}_{SK}(C) = M \end{array}\right] = 1\ .$$

Conceptually, a ciphertext may be rejected at two different stages: the filter using Ive.Verify might reject or decryption using Ive.Dec might fail. Thus, we can consider two possible sets of 'valid' ciphertexts: those accepted by verification, and those accepted by decryption. Ideally, these sets coincide, but a priori this cannot be guaranteed. We call a scheme *consistent* if any ciphertext accepted by decryption will also be accepted by the filter verification, whereas we say the scheme is *strict* if any ciphertext that passes the filter, will decrypt to a message.

Formally, we define both strictness and consistency in terms of rejected 'invalid' ciphertexts, thus flipping the order of Ive.Verify and Ive.Dec in the implications below (compared to the intuitive notion described above). That is for all possible keys (PK, SK) output by Ive.Kg and all ciphertexts C, we have

– *Consistency:* $\mathsf{Ive.Verify}^{ID}_{PK}(C) = \perp \Rightarrow \mathsf{Ive.Dec}^{ID}_{SK}(C) = \perp$;
– *Strictness:* $\mathsf{Ive.Dec}^{ID}_{SK}(C) = \perp \Rightarrow \mathsf{Ive.Verify}^{ID}_{PK}(C) = \perp$.

Fortunately, it is relatively easy to guarantee consistency; the trivial transformation that runs verification as part of decryption takes care of this. Henceforth we will concentrate on consistent schemes.

On the other hand, strictness is harder to guarantee a priori. Thus we will allow ciphertexts to pass the filter that are subsequently deemed invalid by decryption. Note that, for *honestly generated* ciphertexts, correctness ensures that decryption will actually succeed, so this scenario can only occur for 'adulterine' ciphertexts.

Security. The security of IVE comprises of two components: *integrity* to ensure the filter cannot be fooled, and *confidentiality* of the messages to outsiders. With reference to the games defined in Fig. 4 and Fig. 5, the advantages are defined as follows:

– *Integrity*:

$$\mathsf{Adv}^{\mathrm{int}}_{\mathrm{IVE}}(\mathbb{A}) = \Pr\left[\mathsf{Exp}^{\mathrm{int}}_{\mathrm{IVE}}(\mathbb{A})\ :\ \begin{array}{c} \hat{ID} \notin \mathcal{E} \wedge (\hat{ID}, \hat{C}) \notin C \\ \wedge\, \mathsf{Ive.Verify}^{\hat{ID}}_{PK}(\hat{C}) = \top \end{array}\right]\ .$$

– *Confidentiality*:

$$\mathsf{Adv}^{\mathrm{conf}}_{\mathrm{IVE}}(\mathbb{A}) = \Pr\left[\mathsf{Exp}^{\mathrm{conf}\text{-}0}_{\mathrm{IVE}}(\mathbb{A})\ :\ \hat{b} = 0\right] - \Pr\left[\mathsf{Exp}^{\mathrm{conf}\text{-}1}_{\mathrm{IVE}}(\mathbb{A})\ :\ \hat{b} = 0\right]\ .$$

Integrity. A server running the verification algorithm to filter out invalid ciphertexts should not be easily fooled by an adversary: unless one is in possession of an encryption key (i.e. has been vetted), it should not be possible to construct a valid ciphertext. Even a *vetted* sender should not be able to construct a ciphertext which is considered valid under a different identity. We formally capture integrity in a game (Fig. 4) where we use the output of the verification algorithm as an indication of validity. For consistent schemes this choice is the strongest, as a forgery with respect to decryption will always be a forgery with respect to verification.

```
$\mathsf{Exp}^{\mathrm{int}}_{\mathrm{IVE}}(\mathbb{A})$
    $(PK, SK) \leftarrow\$ \mathsf{Ive.Kg}$
    $h \leftarrow 0; C \leftarrow \emptyset; \mathcal{E} \leftarrow \emptyset$
    $(\hat{ID}, \hat{C}) \leftarrow \mathbb{A}^{O}(PK)$
    **winif** $\hat{ID} \notin \mathcal{E} \wedge (\hat{ID}, \hat{C}) \notin C$
        $\wedge\ \mathsf{Ive.Verify}^{\hat{ID}}_{PK}(\hat{C}) = \top$

$\mathsf{derive}(ID)$
    $EK[h] \leftarrow \mathsf{Ive.Derive}_{SK}(ID)$
    $h \leftarrow h + 1$
    **return** $h$

$\mathsf{corrupt}(H)$
    $\mathcal{E} \leftarrow \mathcal{E} \cup H.ID$
    **return** $EK[H]$

$\mathsf{encrypt}(H, M)$
    $C \leftarrow\$ \mathsf{Ive.Enc}_{PK, EK[H]}(M)$
    $C \leftarrow C \cup \{(H.ID, C)\}$
    **return** $C$

$\mathsf{decrypt}(ID, C)$
    $M \leftarrow \mathsf{Ive.Dec}^{ID}_{SK}(C)$
    **return** $M$
```

Fig. 4. The integrity game for IVE.

The adversary is given the verification key as well as encryptions of messages of her own choosing under honest encryption keys. We use *handles* to grant an adversary control over the encryption keys that are used: an adversary can trigger the creation of an arbitrary number of keys for chosen identities and then indicate which key (by order of creation) should be used for a particular encryption query.

Additionally, an adversary can adaptively ask for encryption keys from a corruption oracle. Obviously, a corrupted encryption key trivially allows for the construction of further valid ciphertexts for the underlying identity, so we exclude corrupted identities from the win condition. Similarly, ciphertexts resulting from an encryption query do not count as a win under the original query's identity.

Finally, an adversary has access to a decryption oracle. This oracle is superfluous for uncorrupted encryption keys, but an adversary could potentially use it to her advantage by querying it with ciphertext created under a corrupted identity. These ciphertexts will, of course, not help her win the integrity game directly, as the corresponding identity is corrupted. Yet, the oracle response might leak information about SK, which could help the adversary construct a valid ciphertext for an *uncorrupted* identity, hence giving an advantage in winning the integrity game. Constructing a non-strict pathological IVE scheme exploiting this loophole is easy: simply allow ciphertexts outside the support of the encryption algorithm to gradually leak the secret key based on their validity under decryption. We stress that in our instantiation of IVE we do *not* face this issue.

Confidentiality. We adopt the CCA security notion for public key encryption to the setting of identifiable vetted encryption (Fig. 5). An adversary can, repeatedly, ask its challenge oracle for the encryption of one of two messages under an adversarially chosen encryption key. We give the adversary an oracle to derive and immediately learn encryption keys; moreover these known honest keys may be fed to the challenge encryption oracle.

We want to avoid the decryption oracle being used by an adversary to win trivially, namely by simply querying a challenge ciphertext under the corresponding identity. But what is this corresponding identity? The encryption *algorithm* only takes as input an encryption key EK that may or may not allow easy extraction of an identity ID. One solution would be to only allow the adversary to ask for challenge encryptions on honestly derived encryption keys (so the game can keep track of the identity when EK is derived). Instead, we opted for a stronger version where the adversary provides the challenge encryption *oracle* with both an encryption key EK *and* a purported identity ID. If verification shows that the freshly generated challenge ciphertext does *not* correspond to ID, which can only happen for dishonestly generated pairs (EK, ID), then the encryption oracle rejects the query by outputting $\perp_G$.

Intuitively, the decryption oracle is mainly relevant for identities that the adversary has previously queried to its derivation oracle: after all, if the decryption oracle would return anything but $\perp$ for a fresh ciphertext under a fresh identity, this would constitute a break of the integrity game.

3.2 Encrypt-then-IBS

An obvious first attempt to create an identifiable vetted encryption scheme is to combine the confidentiality provided by a public key encryption scheme with the authenticity of that of an identity based signature scheme. There are three basic methods for the generic composition: sign-then-encrypt, encrypt-then-sign, and encrypt-and-sign. For the first option, the signature ends up being encrypted, which destroys public verifiability as required for the filter to do its work. The parallel encrypt-and-sign is well-known to be problematic,

```
$\mathrm{Exp}^{\text{ind-cca-}b^*}_{\mathrm{IVE}}(\mathbb{A})$
    $(PK, SK) \leftarrow\!\!\$\ \mathsf{Ive.Kg}$
    $\mathcal{C} \leftarrow \emptyset$
    $\hat{b} \leftarrow \mathbb{A}^{O}(PK)$

derive($ID$)
    $EK \leftarrow \mathsf{Ive.Derive}_{SK}(ID)$
    return $EK$

encrypt($ID, EK, M_0, M_1$)
    $C^* \leftarrow\!\!\$\ \mathsf{Ive.Enc}_{PK,EK}(M_{b^*})$
    if $\mathsf{Ive.Verify}^{ID}_{PK}(C^*) = \perp$ then
        return $\perp_G$
    $\mathcal{C} \leftarrow \mathcal{C} \cup \{(ID, C^*)\}$
    return $C^*$

decrypt($ID, C$)
    require $(ID, C) \notin \mathcal{C}$
    $M \leftarrow \mathsf{Ive.Dec}^{ID}_{SK}(C)$
    return $M$
```

Fig. 5. The confidentiality game for IVE.

```
$\mathsf{Ive.Kg}()$
    $(PK, DK) \leftarrow \mathsf{Pke.Kg}$
    $(MVK, MSK) \leftarrow \mathsf{Uibss.Kg}$
    return $((PK, MVK), (DK, MSK))$

$\mathsf{Ive.Enc}_{(PK,MVK),USK,ID}(M)$
    $C \leftarrow \mathsf{Pke.Enc}_{PK}(M \| ID)$
    $\sigma \leftarrow \mathsf{Uibss.Sign}_{USK}(C)$
    if $\mathsf{Uibss.Verify}^{ID}_{MVK}(C, \sigma) = \perp$ then
        return $\perp$
    return $(C, \sigma)$

$\mathsf{Ive.Verify}^{ID}_{PK,MVK}(C, \sigma)$
    return $\mathsf{Uibss.Verify}^{ID}_{MVK}(C, \sigma)$

$\mathsf{Ive.Derive}_{(DK,MSK)}(ID)$
    $USK \leftarrow \mathsf{Uibss.Derive}_{MSK}(ID)$
    return $USK$

$\mathsf{Ive.Dec}^{ID}_{DK,MSK}(C, \sigma)$
    if $\mathsf{Uibss.Verify}^{ID}_{MVK}(C, \sigma) = \perp$ then
        return $\perp$
    if $\mathsf{Pke.Dec}_{DK}(C) = \perp$ then
        return $\perp$
    if $\mathsf{Pke.Dec}_{DK}(C) \rightarrow M \| \bar{ID} \wedge \bar{ID} \neq ID$ then
        return $\perp$
    return $M$
```

Fig. 6. Encrypt-then-IBSign (EtIBS): A straightforward composition of public key encryption and an identity-based signature scheme.

as the unencrypted signature directly on the message inevitably leaks information on the message, even when the signatures are confidential [27] (as the signature allows for an easy check whether a given plaintext was encrypted or not). Thus only encrypt-then-sign remains as option, and we specify the construction in Fig. 6.

We show the scheme achieves integrity and confidentiality in Lemmas 1 and 2, respectively. Integrity of the construction follows from the unforgeability of the underlying signature scheme. However, for IVE to inherit the confidentiality of the encryption scheme, we use an identity-based signature scheme with *outsider unique* signatures.

Without unique signatures, an adversary who has received a challenge ciphertext (C, σ) could simply create a new tuple (C, σ') with a secondary valid signature σ'. This tuple will be accepted by a decryption oracle, and hence the adversary will learn the encrypted message, breaking confidentiality. To the best of our knowledge, unique identity-based signatures have not been studied before. It turns out that for our purposes, a computational version of uniqueness suffices (the details are in Appendix A).

Correctness and consistency. Both correctness and consistency follow easily by inspection. The signature verification as part of decryption is needed for consistency, cf. the transformation mentioned previously.

Integrity. Integrity of the Encrypt-then-IBS construction boils down to the unforgeability of the underlying identity-based signature scheme. As the decryption key of the underlying encryption scheme is unrelated to the issuing key of the signature scheme (so an adversary cannot hope to learn any useful information about the issuing key by querying the decryption oracle with ciphertexts of corrupted identities), the reduction is fairly straightforward.

```
$\mathsf{Exp}^{\mathrm{int}}_{\mathrm{IVE}}(\mathbb{A})$
    $(PK, DK) \leftarrow\!\!\$\ \mathsf{Pke.Kg}$
    $(VK, MSK) \leftarrow\!\!\$\ \mathsf{Uibss.Kg}$
    $h \leftarrow 0; \mathcal{C} \leftarrow \emptyset; \mathcal{E} \leftarrow \emptyset$
    $(\hat{ID}, \hat{C}) \leftarrow \mathbb{A}^{O}(PK)$
    winif $\hat{ID} \notin \mathcal{E} \land (\hat{ID}, \hat{C}) \notin \mathcal{C}$
        $\land\ \mathsf{Ive.Verify}^{\hat{ID}}_{PK}(\hat{C}) = \top$

derive($ID$)
    $ID_h \leftarrow ID$
    $h \leftarrow h + 1$
    return $h$

corrupt($H$)
    $EK_H \leftarrow \mathsf{Uibss.Derive}_{MSK}(ID_H)$
    $\mathcal{E} \leftarrow \mathcal{E} \cup ID_H$
    return $EK_H$

encrypt($H, M$)
    $C \leftarrow\!\!\$\ \mathsf{Pke.Enc}_{PK}(M \| ID_H)$
    $\sigma \leftarrow \mathsf{Uibss.Sign}_{EK_H}(C)$
    $\mathcal{C} \leftarrow \mathcal{C} \cup \{(C, \sigma), ID_H\}$
    return $(C, \sigma)$

decrypt($((C, \sigma), ID)$)
    if $\mathsf{Uibss.Verify}^{ID}_{VK}(C, \sigma) = \perp$
        return $\perp$
    $M \| ID \leftarrow \mathsf{Pke.Dec}_{DK}(C)$
    if decryption or parsing fails
        return $\perp$
    return $M$
```

Fig. 7. The game for the proof of integrity for the Encrypt-then-IBS construction

Lemma 1 (Integrity of Encrypt-then-IBS). *For all adversaries $\mathbb{A}_{\mathrm{int}}$ there exists a similarly efficient adversary $\mathbb{B}_{\text{euf-cma}}$ such that*

$$\mathsf{Adv}^{\mathrm{int}}_{\mathrm{IVE}}(\mathbb{A}_{\mathrm{int}}) \leq \mathsf{Adv}^{\text{euf-cma}}_{\mathrm{UIBSS}}(\mathbb{B}_{\text{euf-cma}}) \,.$$

Proof. The integrity game defined in Fig. 4 applied to our construction is shown in Fig. 7. Based on this game, we may construct a reduction to a forging game of the underlying identity based signature scheme. An adversary $\mathbb{B}_{\text{euf-cma}}$ is given the verification key VK of the signature scheme. She constructs an encryption scheme and generates the keys $(PK, DK) \leftarrow\!\!\$\ \mathsf{Pke.Kg}$, and sends (PK, VK) to $\mathbb{A}_{\mathrm{int}}$. Whenever $\mathbb{A}_{\mathrm{int}}$ makes a derivation query on an identity ID, $\mathbb{B}_{\text{euf-cma}}$ simply does the administrative work herself, by ascribing the identity with a handle, and returning this. Any encryption queries on a message M under a handle H is managed by $\mathbb{B}_{\text{euf-cma}}$ first producing $C \leftarrow\!\!\$\ \mathsf{Pke.Enc}_{PK}(M \| ID_H)$, and then querying her own signature oracle on (C, ID_H), receiving the signature σ. She then sends (C, σ) to $\mathbb{A}_{\mathrm{int}}$. Any corruption queries on H is answered by $\mathbb{B}_{\text{euf-cma}}$ querying her own corruption oracle on ID_H, and forewarding the given signing key. Finally, all decryption queries from $\mathbb{A}_{\mathrm{int}}$ are handled solely by $\mathbb{B}_{\text{euf-cma}}$, as she can perform all the checks and decryptions herself. When $\mathbb{A}_{\mathrm{int}}$ outputs $((\hat{C}, \hat{\sigma}), \hat{ID})$, $\mathbb{B}_{\text{euf-cma}}$ simply copies this as her own answer. It is clear that $\mathbb{B}_{\text{euf-cma}}$ will win in precisely the same cases as $\mathbb{A}_{\mathrm{int}}$, and so the claim follows.

□

Confidentiality. The confidentiality of the Encrypt-then-IBS hinges on both the confidentiality of the encryption scheme and the computational hardness of finding a signature collision in the IBS scheme. As the IVE adversary does not have access to the master private key of the underlying IBS scheme, it suffices that signatures are unique with respect to individual signing keys (that can be obtained through the derive oracle). That allows us to rule out mauling of a challenge ciphertext (C, σ) through the signature component, leaving the adversary with the only option of breaking the confidentiality of the encryption scheme.

Lemma 2 (Confidentiality of Encrypt-then-IBS). *For all adversaries $\mathbb{A}_{\mathrm{conf}}$ there exist similarly efficient adversaries $\mathbb{B}_{\mathrm{cca}}$ and $\mathbb{B}_{\mathrm{ou}}$ such that*

$$\mathsf{Adv}^{\mathrm{conf}}_{\mathrm{IVE}}(\mathbb{A}_{\mathrm{conf}}) \leq \mathsf{Adv}^{\mathrm{conf}}_{\mathrm{PKE}}(\mathbb{B}_{\mathrm{conf}}) + \mathsf{Adv}^{\mathrm{ou}}_{\mathrm{UIBSS}}(\mathbb{B}_{\mathrm{ou}}) \,.$$

Proof. We introduce a series of games for the adversary $\mathbb{A}_{\mathrm{conf}}$ to play, gradually changing the original game into a distinguishing game against the underlying encryption scheme.

Game $\mathbf{G}_0^{b^*}$: This is the original game applied to our construction, presented in Fig. 8.The advantage of $\mathbb{A}_{\mathrm{conf}}$ may be expressed as $\mathsf{Adv}^{\mathrm{conf}}_{\mathrm{IVE}}(\mathbb{A}_{\mathrm{conf}}) = \Pr\left[\mathbf{G}_0^0 : \mathbb{A}_{\mathrm{conf}} \to 1\right] - \Pr\left[\mathbf{G}_0^1 : \mathbb{A}_{\mathrm{conf}} \to 1\right]$.

Game $\mathbf{G}_1^{b^*}$: Here, we change the decryption procedure, so that instead of demanding that a query $((C, \sigma), ID) \notin \mathcal{C}$, we require only that C has not been part of a challenge recieved from the encryption oracle. An adversary able to distinguish between these two games would also be able to find two distinct and verifying signatures

```
Game $\mathbf{G_0}^{b^*}$
$(PK, DK) \leftarrow\!\$\ \text{Pke.Kg}$
$(VK, SK) \leftarrow \text{Uibss.Kg}$
$C \leftarrow \emptyset$
$\hat{b} \leftarrow \mathbb{A}^{O}((PK, VK)$
```

```
encrypt($(M_0, ID, USK), (M_1, ID, USK)$)
$C_{b^*} \leftarrow\!\$\ \text{Pke.Enc}_{PK}(M_{b^*} \| ID)$
$\sigma_{b^*} \leftarrow\!\$\ \text{Uibss.Sign}_{USK}(C_{b^*})$
$C^* \leftarrow ((C_{b^*}, \sigma_{b^*}), ID)$
$C \leftarrow C \cup \{C^*\}$ return $C^*$

derive($ID$)
$USK \leftarrow \text{Uibss.Derive}_{MSK}(ID)$
return $USK$
```

```
decrypt($C, \pi$)
require $(C, \sigma) \notin C$
if $\text{Uibss.Verify}_{VK}^{ID}(C, \sigma) = \perp$
    return $\perp$
$M \| ID \leftarrow \text{Pke.Dec}_{DK}(C)$
if decryption or parsing fails
    return $\perp$
return $M$
```

Fig. 8. Game $\mathbf{G}_1^{b^*}$ for the confidentiality proof of the Encrypt-then-IBS construction.

on the same message. It follows that the difference between $\mathbf{G}_0^{b^*}$ and $\mathbf{G}_1^{b^*}$ may be bounded by the advantage an adversary has of breaking the outsider unicity of the underlying signature scheme.

Given this, we may construct a reduction from $\mathbf{G}_1^{b^*}$ to a standard indistinguishability game of the underlying public key encryption scheme in the following way: an adversary $\mathbb{B}_{\text{cca}}$ given the public key PK of an encryption scheme generates the keys (VK, MSK) for an unique identity based signature scheme, and sends (PK, VK) to the adversary $\mathbb{A}_{\text{conf}}$. Any derivation queries may be answered by $\mathbb{B}_{\text{cca}}$ alone, seeing as she possesses MSK. Whenever $\mathbb{A}_{\text{conf}}$ sends a challenge query (M_0, M_1, ID, USK), $\mathbb{B}_{\text{cca}}$ sends $(M_0 \| ID, M_1 \| ID)$ to her encryption oracle, and when she gets the challenge ciphertext back, she signs it using the user secret key USK before sending the tuple to $\mathbb{A}_{\text{conf}}$. Any decryption query is handled by $\mathbb{B}_{\text{cca}}$ first verifying the signature σ, and sending C to her own decyrption oracle if the signature verifies, and passing on the response from the oracle to $\mathbb{A}_{\text{conf}}$. Once $\mathbb{A}_{\text{conf}}$ guesses $\hat{b}$, $\mathbb{B}_{\text{cca}}$ copies it, and so it follows that $\text{Adv}_{\text{IVE}}^{\text{conf}}(\mathbb{A}_{\text{conf}}) \leq \text{Adv}_{\text{PKE}}^{\text{conf}}(\mathbb{B}_{\text{conf}}) + \text{Adv}_{\text{UIBSS}}^{\text{ou}}(\mathbb{B}_{\text{ou}})$.

□

3.3 Discussion of IVE

IVE resembles identity-based signcryption in many ways, as both primitives offer confidentiality of messages and integrity of communication between two individuals identifiable to each other. In both cases, this concerns insider security: reading the message requires nothing less than the secret key/decryption key of the recipient, and forging the signature of a sender requires the user key/private key of that particular sender. There is also a notion of verification in identity based signcryption, which guarantees that a decrypted message was in fact written by the sender [21].

In addition, it is common for identity based signcryption to satisfy the security notion of ciphertext unlinkability: it is not possible to link a sender to a specific ciphertext, even if the ciphertext decrypts to a message signed by the sender in question. Another security notion relevant for identity based signcryption is insider ciphertext anonymity, which informally means that deducing either the sender or recipient of a given ciphertext requires the private key of the recipient [21].

It is obvious that the two latter security notions do not combine with a central feature of IVE, namely public verification that a sender has in fact been vetted, seeing as the verification algorithm takes the sender identity as input. To filter out messages sent from unvetted individuals is an essential part of IVE, and this does require a public verification algorithm.

There are identity based signcryption schemes that offer such public verification. However, several of the schemes require the receiver to collaborate by supplying the verification algorithm with additional information. For example, in the signcryption scheme proposed by Libert and Quisquater, the receiver has to supply the verifier with an ephemeral key [38]. Again, this runs counter to the idea of IVE, namely that the filter is able to do the filtering without assistance from the recipient. Querying the recipient to check whether a message is sent from a vetted sender renders the filter pointless.

Finally, there does exist identity based signcryption schemes which offer *transferable public verification.* In these schemes, it is possible for a third party to verify that a ciphertext has indeed been signed by the alleged sender, without help from the receiver. To the best of our knowledge, there are only two such schemes, and both of them adopt an encrypt-and-sign approach [42, 44], where the former does not have a proof of security.

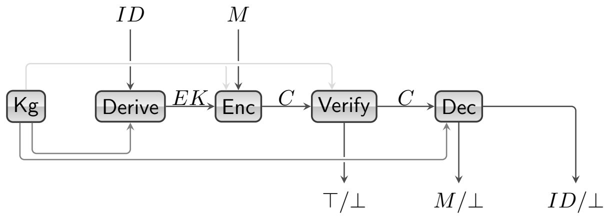

Fig. 9. The algorithms and their inputs/outputs for opaque vetted encryption.

The scheme which is provably secure is based on bilinear pairings, requires very large public parameters, and produces ciphertexts of a large size. We believe that our more general approach might result in a concrete scheme with more favourable sizes, both with regards to parameters and ciphertext size.

4 Opaque Vetted Encryption (OVE)

4.1 Syntax and Security of OVE

The algorithms. For opaque vetted encryption, we are in the most challenging, 'asymmetric' scenario where the filter does *not* learn the identity of the sender, yet the recipient does. In the definition below, we model this change by letting the identity be *output* as part of decryption, in addition to the message of course. Having two outputs also affects how invalid ciphertexts are dealt with: for our syntax and security we deal with the general case where either component can lead to rejection, independently of each other. Thus we allow a large number of error messages, unlike the AVE or IVE case, where only a single error message was modeled.

As we will see, OVE is quite similar to a group signature with message recovery, which seems to be an overlooked primitive. In line with the literature on group signatures, in Definition 3 we split the private key in two: an issuing key IK to derive identity-specific encryption keys and a master decryption key SK to decrypt ciphertexts. Throughout we will also borrow group signature terminology, for instance by referring to the derivation of an encryption key as 'issuing' (of course, in the group signature setting, said key would be a signing key instead), or use 'opening' to extract the identity from a ciphertext as part of decryption.

Definition 3 (Opaque Vetted Encryption (OVE)). *An* opaque vetted encryption *scheme* OVE *is a 5-tuple of algorithms* (Ove.Kg, Ove.Derive, Ove.Enc, Ove.Verify, Ove.Dec) *that satisfy*

- Ove.Kg generates *a key triple* (PK, SK, IK)*, where* PK *is the public encryption (and verification) key,* SK *is the private decryption key, and* IK *is the issuing key. We allow* Ove.Kg *to depend on parameters* $param$ *and write* $(PK, SK, IK) \leftarrow_\$ \mathsf{Ove.Kg}(param)$*. Henceforth, we will assume that* PK *can be uniquely and efficiently computed given either* SK *or* IK*.*
- Ove.Derive issues *an encryption key* EK *based on the issuing key* IK *and a user's identity* ID*. We write* $EK \leftarrow_\$ \mathsf{Ove.Derive}_{IK}(ID)$*.*
- Ove.Enc encrypts *a message* M *given the public encryption key* PK *and private encryption key* EK*, producing a ciphertext* C *or a failed encryption symbol* $\bot$*. So,* $C \leftarrow_\$ \mathsf{Ove.Enc}_{PK,EK}(M)$ *with maybe* $C = \bot$*.*
- Ove.Verify verifies *the validity of a ciphertext* C *given the public verification key* PK*. With a slight abuse of notation,* $\top/\bot \leftarrow \mathsf{Ove.Verify}_{PK}(C)$*.*
- Ove.Dec decrypts *a ciphertext* C *using the private key* SK*, resulting in a message–identity pair* (M, ID)*. Both the message* M *and the identity* ID *may, independently of each other, result in a rejection,* $\bot$*. Again, with a slight abuse of notation,* $(M/\bot, ID/\bot) \leftarrow \mathsf{Ove.Dec}_{SK}(C)$*.*

The first three algorithms are probabilistic, the final two deterministic.

Correctness and consistency. As is the case for AVE and IVE, *correctness* captures that honest usage of the scheme ensures that messages are received as intended, and assigned to the actual sender. For all parameters $param$, identities ID and messages M we have

```
$\mathsf{Exp}^{\mathrm{trac}}_{\mathrm{OVE}}(\mathbb{A})$
$(PK, SK, IK) \leftarrow_{\$} \mathsf{Ove.Kg}$
$h \leftarrow 0; \mathcal{C} \leftarrow \emptyset$
$\mathcal{CU} \leftarrow \{\bot\}$
$\hat{C} \leftarrow \mathbb{A}^{\mathcal{O}}(PK)$
$(M, ID) \leftarrow \mathsf{Ove.Dec}_{SK}(\hat{C})$
winif $\hat{C} \notin \mathcal{C} \wedge ID \notin \mathcal{CU}$
    $\wedge\ \mathsf{Ove.Verify}_{PK}(\hat{C}) = \top$

derive($ID$)
$EK[h] \leftarrow \mathsf{Ove.Derive}_{IK}(ID)$
$h \leftarrow h + 1$
return $h$

corrupt($H$)
$\mathcal{CU} \leftarrow \mathcal{CU} \cup \{H.ID\}$
return $EK[H]$

encrypt($H, M$)
$C \leftarrow_{\$} \mathsf{Ove.Enc}_{PK,EK[H]}(M)$
$\mathcal{C} \leftarrow \mathcal{C} \cup \{C\}$
return $C$

decrypt($C$)
$(M, ID) \leftarrow \mathsf{Ove.Dec}_{SK}(C)$
return $(M, ID)$
```

Fig. 10. The traceability game for OVE.

```
$\mathsf{Exp}^{\mathrm{int}}_{\mathrm{OVE}}(\mathbb{A})$
$(PK, SK, IK) \leftarrow_{\$} \mathsf{Ove.Kg}$
$\hat{C} \leftarrow \mathbb{A}^{\mathcal{O}}(PK, SK, IK)$
$(M, ID) \leftarrow \mathsf{Ove.Dec}_{SK}(\hat{C})$
winif $\mathsf{Ove.Verify}_{PK}(\hat{C}) = \top \wedge (M = \bot \vee ID = \bot)$
```

Fig. 11. The integrity game for OVE.

$$\Pr\left[\begin{array}{c} (PK, SK, IK) \leftarrow_{\$} \mathsf{Ove.Kg}(param) \\ EK \leftarrow_{\$} \mathsf{Ove.Derive}_{IK}(ID) \\ C \leftarrow_{\$} \mathsf{Ove.Enc}_{PK,EK}(M) \end{array} : \begin{array}{c} C \neq \bot \\ \wedge\ \mathsf{Ove.Verify}_{PK}(C) = \top \\ \wedge\ \mathsf{Ove.Dec}_{SK}(C) = (M, ID) \end{array}\right] = 1.$$

As with the previously presented schemes, *consistency* means that any ciphertext which decrypts to a valid message and identity, will also pass the filter. Thus we treat any occurrence of $\bot$ in the decryption, as either message or identity, as an invalid ciphertext. Again, we can easily transform a correct scheme into one that is consistent as well: as part of decryption, run the verification, and if verification returns $\bot$, then decryption returns $(\bot, \bot)$.

Security. The security of OVE is an amalgam of the vetted encryption notions we have encountered so far and those for group signatures, primarily the static "BMW" notions [13]. The integrity component we saw earlier now splits into two: on the one hand, we want that ciphertexts that pass the filter (so verify) can be pinned to a user after decryption, yet on the other hand we want to avoid users being falsely suspected of spamming (by an honest recipient). We relabel the first notion integrity and strengthen it slightly, so it becomes essentially a computational equivalent of strictness. The second notion is traceability, known from group signatures. We also require confidentiality of the messages and anonymity of the senders, but it turns out we can fold these two concepts into a single notion, dubbed privacy. Formally, we define the following advantages, with specifications and explanations of the corresponding experiments described below:

– *Traceability*: $\mathsf{Adv}^{\mathrm{trac}}_{\mathrm{OVE}}(\mathbb{A}) = \Pr\left[\mathsf{Exp}^{\mathrm{trac}}_{\mathrm{OVE}}(\mathbb{A}) : \mathbb{A} \text{ wins}\right]$.
– *Integrity*: $\mathsf{Adv}^{\mathrm{int}}_{\mathrm{OVE}}(\mathbb{A}) = \Pr\left[\mathsf{Exp}^{\mathrm{int}}_{\mathrm{OVE}}(\mathbb{A}) : \mathbb{A} \text{ wins}\right]$.
– *Privacy*: $\mathsf{Adv}^{\mathrm{priv}}_{\mathrm{OVE}}(\mathbb{A}) = \Pr\left[\mathsf{Exp}^{\mathrm{priv}\text{-}0}_{\mathrm{OVE}}(\mathbb{A}) : \hat{b} = 0\right] - \Pr\left[\mathsf{Exp}^{\mathrm{priv}\text{-}1}_{\mathrm{OVE}}(\mathbb{A}) : \hat{b} = 0\right]$.

Traceability. This notion (Fig. 10) ensures that a colluding group of vetted users cannot successfully create a ciphertext that opens to the identity of another user (outside the collusion). As we do not incorporate a PKI in our model (cf. the dynamic "BSZ" notions for group signatures [16]), we need to exclude the issuing key IK from the adversary's grasp. Furthermore, in contrast to BMW's traceability, we also do not provide the decryption key DK to the adversary. Our weakening is motivated by the intended use case: the main purpose of the scheme is to trace messages which pass the filter back to an identity and the recipient has no motive to try and create ciphertexts that it will then subsequently open and trace incorrectly. Of course, in order to provide forward security, one could also consider *strong traceability*, where an adversary does have access to the decryption key SK.

```
$\mathsf{Exp}^{\text{priv-}b^*}_{\text{OVE}}(\mathbb{A})$
$(PK, SK, IK) \leftarrow\!\$\ \mathsf{Ove.Kg}$
$\mathcal{C} \leftarrow \emptyset$
$\hat{b} \leftarrow \mathbb{A}^{\mathcal{O}}(PK, IK)$
```

```
decrypt($C$)
require $C \notin \mathcal{C}$
$(M, ID) \leftarrow \mathsf{Ove.Dec}_{SK}(C)$
return $(M, ID)$
```

```
encrypt($EK_0, EK_1, M_0, M_1$)
$C_0 \leftarrow\!\$\ \mathsf{Ove.Enc}_{PK,EK_0}(M_0)$
$C_1 \leftarrow\!\$\ \mathsf{Ove.Enc}_{PK,EK_1}(M_1)$
if $C_0 \neq \perp \wedge C_1 \neq \perp$ then
    $C^* \leftarrow C_{b^*}$
    $\mathcal{C} \leftarrow \mathcal{C} \cup \{C^*\}$
else
    $C^* \leftarrow \perp$
return $C^*$
```

```
encrypt$_x$($EK_0, EK_1, M_0, M_1$)
$C_0 \leftarrow\!\$\ \mathsf{Ove.Enc}_{PK,EK_0}(M_0)$
$C_1 \leftarrow\!\$\ \mathsf{Ove.Enc}_{PK,EK_1}(M_1)$
$C_x \leftarrow\!\$\ \mathsf{Ove.Enc}_{PK,EK_1}(M_0)$
if $C_0 \neq \perp \wedge C_1 \neq \perp$ then
    if $C_x = \perp$ then set bad
    $C^* \leftarrow C_x$
    $\mathcal{C} \leftarrow \mathcal{C} \cup \{C^*\}$
else
    $C^* \leftarrow \perp$
return $C^*$
```

Fig. 12. The privacy game for OVE (first three columns); the final column is used in the proof of Lemma 3.

Finally, we initialize $\mathcal{CU}$ to contain $\perp$ as we consider the case where the ciphertext opens to an invalid identity, so $\mathsf{Ove.Dec}_{SK}(C) = (M, \perp)$, only as a breach of integrity, not of traceability. Again, this fits the intended use case: an adversary being able to pass the filter without being identified afterwards can effectively "spam" the receiver, who then does not know which sender to have a word with. As the protection against spamming is the raison d'être of our scheme, we will put much stronger guarantees in place to prevent it (as part of integrity).

Integrity. In stark contrast to traceability, *integrity* ensures that even an adversary in possession of all the keys of the scheme cannot create a message which verifies, so $\mathsf{Ove.Verify}_{PK}(C) = \top$, yet does not open to a valid message–identity pair, i.e. leads to $\mathsf{Ove.Dec}_{SK}(C) = (\perp, ID)$, $\mathsf{Ove.Dec}_{SK}(C) = (M, \perp)$ or $\mathsf{Ove.Dec}_{SK}(C) = (\perp, \perp)$. Thus any ciphertext that passes the verification, is opened without a failure message.

We reiterate that we treat $\mathsf{Ove.Dec}_{SK}(C) = (M, \perp)$ as a breach of integrity rather than traceability. One interpretation is that C decrypted successfully to an anonymous message. Yet allowing for anonymous messages would clearly defeat the purpose of opaque vetted encryption, namely that any ciphertext which verifies can be attributed to a vetted sender.

Privacy, confidentiality, and anonymity. Any party not in possession of the decryption key should be unable to determine who is the sender of a ciphertext, and also what the ciphertext decrypts to. Note that we allow an adversary access to the issuing key IK. This is seemingly a contradiction to the discussed honest use case, where the recipient both issues keys and decrypts messages, which was after all the reasoning for denying the adversary the opening key in the traceability case. However, there is a possible separation of authorities, and even though we regard the recpient as the "owner" of the scheme, they may choose to delegate the authority of issuing keys to another authority. We require that even this party should not be able to infer the sender or the content when given a ciphertext.

We formalize this notion as *privacy* (Fig. 12), which we model with a challenge encryption oracle that an adversary can query on two pairs of encryption keys and messages: (EK_0, M_0) and (EK_1, M_1). The oracle either returns an encryption of the left, 0-subscripted or the right, 1-subscripted key–message pair; the adversary should figure out which one. To avoid trivial wins based on faulty encryption keys, we encrypt both pairs, and reject the query if one of the encryptions fail. Privacy should hold even against adversaries knowing the issuing key IK. Our notion of *privacy* encompasses both *anonymity* and *confidentiality* of encryption schemes. We define anonymity as the privacy game with the restriction that for all challenge queries $M_0 = M_1$, and confidentiality as the privacy game where we insist $EK_0 = EK_1$ for all challenge queries.

The resulting anonymity game resembles anonymity known from group signatures. One notable difference is the additional mechanism we put in place by encrypting under both encryption keys and only output the ciphertext if both encryptions are successful. We are not aware of a similar mechanism to define anonymity of group signatures, i.e. where you would sign under both user signing keys and only release the group signature if both are successful: BMW only deal with honestly generated keys and BSZ have a join protocol that alleviates the need for an additional check.

For confidentiality, arguably one could consider a stronger game where one directly encrypts the relevant challenge message under the adversarially chosen key. Yet, this strenghtening is not entirely without gain of generality, as one could concoct a pathological counterexample where for some fake encryption key some messages are more likely to result in an encryption error than others. Henceforth, we will ignore this subtlety.

By definition, privacy obviously implies anonymity and confidentiality (with a small caveat for the latter, as explained above). The converse is true as well, namely that *jointly* anonymity and confidentiality imply privacy. However, in general this is not true, as can be shown by a simple, pathological counterexample.

Consider a scheme that is secure, now modify the scheme so that key derivation prepends keys with a 0-bit. Encryption with a key starting with a 0-bit removes this bit and behaves as before. This fully describes the honest behaviour of the scheme and we proceed to describe behaviour that could only be triggered by an adversary: namely, our modified scheme's encryption with a key starting with a 1-bit outputs the message iff that message equals the key, and rejects otherwise. Essentially, all 1-keys are fake, but it is possible to make each key accept on a single message (and each message can only be used for a single fake key). For the confidentiality and anonymity games, these fake keys cannot be exploited as the reject-filtering mechanism causes the oracle to reject; for the privacy game however it's easy to win exploiting these fake keys.

For schemes that behave nicely however, we show in Lemma 3 that the privacy game is implied by combination of anonymity and confidentiality. Here 'nicely' refers to the property that an encryption key is either always successful on the full message space, or it always rejects.

Lemma 3 (OVE-Anonymity + OVE-Confidentiality implies OVE-Privacy). *Let* OVE *sport encryption keys* EK *with the property that for all messages* M *in the message space,* $\mathsf{Ove.Enc}_{PK,EK}(M) = \perp$*, or every message encrypts to a ciphertext with probability 1. Then for any privacy adversary* $\mathbb{A}_{\mathrm{priv}}$ *against an* OVE *scheme, there exist anonymity and confidentiality adversaries* $\mathbb{B}_{\mathrm{conf}}$ *and* $\mathbb{B}_{\mathrm{anon}}$ *of comparable efficiency such that*

$$\mathsf{Adv}^{\mathrm{priv}}_{\mathrm{OVE}}(\mathbb{A}_{\mathrm{priv}}) \le \mathsf{Adv}^{\mathrm{anon}}_{\mathrm{OVE}}(\mathbb{B}_{\mathrm{anon}}) + \mathsf{Adv}^{\mathrm{conf}}_{\mathrm{OVE}}(\mathbb{B}_{\mathrm{conf}}) \,.$$

Proof. First, we define the games we will use throughout the proof. In all cases, the challenge oracle receives $((EK_0, M_0), (EK_1, M_1))$, but different inputs are selected for encryption as the challenge ciphertext:

- $\mathbf{G_0}$: the challenge oracle chooses (EK_0, M_0);
- $\mathbf{G_1}$: the challenge oracle chooses (EK_1, M_1);
- $\mathbf{G_x}$: the challenge oracle chooses (EK_1, M_0).

Furthermore all three games, including $\mathbf{G_x}$ use the first two cases to decide whether to reject a query (output $\perp$) or not. In the case of $\mathbf{G_x}$, if the encryption itself fails but the check is passed, we set a flag bad. The code for the encryption oracle of $\mathbf{G_x}$ is provided in Fig. 12.

We may express the advantage of $\mathbb{A}_{\mathrm{priv}}$ as:

$$\begin{aligned}\mathsf{Adv}^{\mathrm{priv}}_{\mathrm{OVE}}(\mathbb{A}_{\mathrm{priv}}) &= \Pr\left[\mathbf{G_0} \,:\, \mathbb{A}_{\mathrm{priv}} \to 0\right] - \Pr\left[\mathbf{G_1} \,:\, \mathbb{A}_{\mathrm{priv}} \to 0\right] \\ &= \Pr\left[\mathbf{G_0} \,:\, \mathbb{A}_{\mathrm{priv}} \to 0\right] - \Pr\left[\mathbf{G_x} \,:\, \mathbb{A}_{\mathrm{priv}} \to 0\right] \\ &\quad + \Pr\left[\mathbf{G_x} \,:\, \mathbb{A}_{\mathrm{priv}} \to 0\right] - \Pr\left[\mathbf{G_1} \,:\, \mathbb{A}_{\mathrm{priv}} \to 0\right] .\end{aligned}$$

We claim existence of $\mathbb{B}_{\mathrm{anon}}$ and $\mathbb{B}_{\mathrm{conf}}$ such that

$$\Pr\left[\mathbf{G_0} \,:\, \mathbb{A}_{\mathrm{priv}} \to 0\right] - \Pr\left[\mathbf{G_x} \,:\, \mathbb{A}_{\mathrm{priv}} \to 0\right] \le \mathsf{Adv}^{\mathrm{anon}}_{\mathrm{OVE}}(\mathbb{B}_{\mathrm{anon}})$$

as well as

$$\Pr\left[\mathbf{G_x} \,:\, \mathbb{A}_{\mathrm{priv}} \to 0\right] - \Pr\left[\mathbf{G_1} \,:\, \mathbb{A}_{\mathrm{priv}} \to 0\right] \le \mathsf{Adv}^{\mathrm{conf}}_{\mathrm{OVE}}(\mathbb{B}_{\mathrm{conf}}) \,.$$

We prove the first claim: given a privacy adversary $\mathbb{A}_{\mathrm{priv}}$, we may construct an anonymity adversary in the following way: $\mathbb{B}_{\mathrm{anon}}$ gets input (PK, IK), which she passes along to $\mathbb{A}_{\mathrm{priv}}$. When $\mathbb{A}_{\mathrm{priv}}$ sends her challenge request $((EK_0, M_0), (EK_1, M_1))$, $\mathbb{B}_{\mathrm{anon}}$ first encrypts (EK_1, M_1) herself. If this results in $\perp$, she sends a rejection to $\mathbb{A}_{\mathrm{priv}}$, simulating the response from a privacy encryption oracle. If $\mathsf{Ove.Enc}_{PK,EK_1}(M_1) \neq \perp$, then $\mathbb{B}_{\mathrm{anon}}$ sends the requests $((EK_0, M_0), (EK_1, M_0))$ to her challenge oracle. By the assumption that an encryption key will either encrypt all messages or none, this cannot result in the bad event $\mathsf{Ove.Enc}_{PK,EK_1}(M_0) = \perp$. Thus, if the encryption oracle returns $\perp$, this is caused by (EK_0, M_0), and the rejection is therefore in line with a privacy encryption oracle. Once $\mathbb{B}_{\mathrm{anon}}$ receives the challenge ciphertext, she passes it to $\mathbb{A}_{\mathrm{priv}}$. Any decryption query made by $\mathbb{A}_{\text{priv}}$ is answered by $\mathbb{B}_{\text{anon}}$'s decryption oracle. When $\mathbb{A}_{\text{priv}}$ outputs a bit b, $\mathbb{B}_{\text{anon}}$ answers the same, and will thus have the same advantage in her game as $\mathbb{A}_{\text{priv}}$ has in hers. The claim follows.

The second claim is proven anologously: given a privacy adversary $\mathbb{A}_{\text{priv}}$, we may construct a confidentiality adversary as follows: $\mathbb{B}_{\text{conf}}$ gets input (PK, IK), which she passes along to $\mathbb{A}_{\text{priv}}$. When $\mathbb{A}_{\text{priv}}$ queries a challenge by sending $((EK_0, M_0), (EK_1, M_1))$, $\mathbb{B}_{\text{conf}}$ encrypts (EK_0, M_0) herself, and rejects the query if the encryption results in $\perp$. This simulates the rejection from a privacy encryption oracle. If she does not reject, $\mathbb{B}_{\text{conf}}$ sends the requests $((EK_1, M_0), (EK_1, M_1)$ to her challenge oracle, and sends the challenge ciphertext she receives to $\mathbb{A}_{\text{priv}}$. Again, if the encryption oracle rejects, this is caused by (EK_1, M_1), and is in line with the behaviour of a privacy encryption oracle. Given the assumption of valid or invalid encryption keys, the bad event $\mathsf{Ove.Enc}_{PK,EK_1}(M_0) = \perp$ does not happen. Any decryption query made by $\mathbb{A}_{\text{priv}}$ is answered by $\mathbb{B}_{\text{conf}}$'s decryption oracle. When $\mathbb{A}_{\text{priv}}$ outputs a bit b, $\mathbb{B}_{\text{conf}}$ answers the same, and will thus have the same advantage in her game as $\mathbb{A}_{\text{priv}}$ has in hers. The claim follows.

Based on these steps, we have:

$$\mathsf{Adv}^{\text{priv}}_{\text{OVE}}(\mathbb{A}_{\text{priv}}) \leq \mathsf{Adv}^{\text{anon}}_{\text{OVE}}(\mathbb{B}_{\text{anon}}) + \mathsf{Adv}^{\text{conf}}_{\text{OVE}}(\mathbb{B}_{\text{conf}}) \,.$$

□

4.2 Generic Construction: Verifiably Encrypted Certificates

Our construction is inspired by the sign-encrypt-proof construction for group signature schemes [13]. This provenance is natural, given the close relationship between OVE and group signatures (albeit with message recovery). The most important difference, aside from having to keep the message confidential, is our weakening of traceability, by not availing the adversary with the decryption key. We reflect on the difference between our scheme and known group signature schemes in Section 4.3.

Our scheme uses an IND-CCA secure PKE, an EUF-CMA secure SIG and a simulation-sound QANIZK; the construction is fleshed out in Fig. 13. The key generation algorithm generates the key pairs (PK, DK), (VK, SK) for the PKE and SIG respectively, as well as the crs σ and trapdoor τ for the QANIZK scheme. The public key for the OVE is the triple (PK, VK, σ), the derivation key is SK, and finally the decryption key is DK. We stress that the trapdoor τ is discarded after derivation: it is used only in the security reductions, not in the actual scheme itself, and accidentally including it in the private derivation or decryption key would actually invalidate integrity!

For a given user with identity ID, the derivation issues a certificate $CERT_{ID}$ by signing ID using the signature scheme. The certificate may then be regarded as the encryption key of the user with identity ID.

To encrypt a message M, on input the public key of the OVE as well as the identity ID and certificate $CERT_{ID}$ of the encryptor, first the validity of the certificate is checked to guard against dishonest certificates. If the certificate passes, the concatenated string $M \| CERT_{ID} \| ID$ is encrypted to C using the underlying encryption scheme. Next, a QANIZK proof π is generated for the statement that the ciphertext is created honestly, specifically that it contains a valid $ID, CERT_{ID}$ pair. The OVE encryption algorithm finally outputs (C, π).

Formally, for the QANIZK proof, the language $L_{(PK,VK)}$ is determined by the public key (PK, VK) and consists of valid ciphertexts, i.e.,

$$L_{(PK,VK)} = \{C : \exists_{M,r,CERT_{ID},ID}\ C = \mathsf{Pke.Enc}_{PK}(M \| CERT_{ID} \| ID; r) \wedge \mathsf{Sig.Verify}_{VK}(ID, CERT_{ID}) = \top\}.$$

Thus the message M and the randomness r used to encrypt are additional witnesses used to create the QANIZK proof π; the full witness is the tuple $(r, M, CERT_{ID}, ID)$.

For the filter to verify a pair (C, π), it simply runs the verification algorithm of the QANIZK scheme, with the public key of the OVE scheme as well as (C, π) as input.

Finally, in order to decrypt an OVE ciphertext (C, π), the receiver first verifies the proof π using the verification algorithm of the QANIZK. If the QANIZK verification fails, the receiver rejects. Otherwise, it decrypts C and attempts to parse the output as $M \| CERT_{ID} \| ID \leftarrow \mathsf{Pke.Dec}_{DK}(C)$. If either decryption or parsing fails, the receiver rejects. If both succeed, it returns (M, ID). There is no need to explicitly run the verification algorithm of the signature scheme on the certificate as its validity is already implicitly checked by the QANIZK verification. Note that we output the rejection symbol $(\perp, \perp)$ in all cases (failure of the verification, decryption, or parsing), and in particular that we do not distinguish between a failure to decrypt the message M or the identity ID, as the syntax (Fig. 9) allows for.

```
Ove.Kg()
    $(PK, DK) \leftarrow\!\!\$\ \mathsf{Pke.Kg}$
    $(VK, SK) \leftarrow\!\!\$\ \mathsf{Sig.Kg}$
    $(\sigma, \tau) \leftarrow\!\!\$\ \mathsf{Nizk.Setup}$
    return $((PK, VK, \sigma), DK, SK)$

Ove.Derive$_{SK}(ID)$
    $CERT_{ID} \leftarrow\!\!\$\ \mathsf{Sig.Sign}_{SK}(ID)$
    return $CERT_{ID}$

Ove.Verify$_{PK,VK}(C, \pi)$
    return $\mathsf{Nizk.Verify}_{PK,VK,\sigma}(C, \pi)$

Ove.Enc$^{ID}_{PK,VK,\sigma,ID,CERT_{ID}}(M)$
    if $\mathsf{Sig.Verify}_{VK}(ID, CERT_{ID}) = \perp$, return $\perp$
    $C \leftarrow\!\!\$\ \mathsf{Pke.Enc}_{PK}(M \| CERT_{ID} \| ID; r)$
    $\pi \leftarrow \mathsf{Nizk.Prove}_{PK,VK,\sigma}(r, M, CERT_{ID}, ID)$
    return $(C, \pi)$

Ove.Dec$_{DK}(C, \pi)$
    if $\mathsf{Nizk.Verify}_{PK,VK,\sigma}(C, \pi) = \perp$
        return $(\perp, \perp)$
    $M \| CERT_{ID} \| ID \leftarrow \mathsf{Pke.Dec}_{DK}(C)$
    if decryption or parsing fails
        return $(\perp, \perp)$
    return $(M, ID)$
```

Fig. 13. Our "Verifiably Encrypted Certificate" construction for OVE.

```
Game $\mathbf{G_0}$
    $(PK, DK) \leftarrow\!\!\$\ \mathsf{Pke.Kg}$
    $(VK, SK) \leftarrow \mathsf{Sig.Kg}$
    $(\sigma, \tau) \leftarrow\!\!\$\ \mathsf{Nizk.Setup}$
    $h \leftarrow 0; \mathcal{C} \leftarrow \emptyset$
    $\mathcal{CU} \leftarrow \emptyset$
    $(\hat{C}, \hat{\pi}) \leftarrow \mathbb{A}^{O}(PK, VK, \sigma)$
    $(M, ID) \leftarrow \mathsf{Ove.Dec}_{PK}(\hat{C})$
    winif $(\hat{C}, \hat{\pi}) \notin \mathcal{C} \wedge ID \notin \mathcal{CU} \wedge$
        $\mathsf{Ove.Verify}_{PK,VK,\sigma}(\hat{C}, \hat{\pi}) = \top$

derive$(ID)$
    $ID_h = ID$
    $CERT_{ID_h} \leftarrow \mathsf{Sig.Sign}_{SK}(ID_h)$
    $h \leftarrow h + 1$
    return $h$

corrupt$(H)$
    $\mathcal{CU} \leftarrow \mathcal{CU} \cup \{ID_H\}$
    return $CERT_{ID_H}$

encrypt$(H, M)$
    $C \leftarrow\!\!\$\ \mathsf{Pke.Enc}_{PK}(M \| CERT_{ID_H} \| ID_H; r)$
    $\pi \leftarrow \mathsf{Nizk.Prove}_{PK,VK,\sigma}(r, M, CERT_{ID_H}, ID_H)$
    $\mathcal{C} \leftarrow \mathcal{C} \cup \{(C, \pi)\}$
    return $(C, \pi)$

decrypt$(C, \pi)$
    if $\mathsf{Nizk.Verify}_{PK,VK,\sigma}(C, \pi) = \perp$
        return $(\perp, \perp)$
    $M \| CERT_{ID} \| ID \leftarrow \mathsf{Pke.Dec}_{DK}(C)$
    if decryption or parsing fails
        return $(\perp, \perp)$
    return $(M, ID)$
```

Fig. 14. The initial traceability game $\mathbf{G_0}$ for our Verifiably Encrypted Certificate construction for OVE.

Correctness and consistency. Correctness follows from the correctness of the underlying PKE and SIG, as well as the completeness of the QANIZK. Consistency is guaranteed by checking the proof in decryption, as this ensures that any ciphertext which decrypts also passes the filter.

Traceability. Intuitively, the traceability of the scheme boils down to the unforgeability of the signature scheme used as a building block. The other properties of the PKE and QANIZK ensure that the encryption oracle is harmless, i.e. that the returned components (C, π) do not leak any information about the valid and potentially honest certificate used.

Lemma 4 (Traceability of OVE). *For all adversaries $\mathbb{A}_{\text{trac}}$, there exist similarly efficient adversaries $\mathbb{B}_{\text{sound}}$, $\mathbb{B}_{\text{zk}}$, $\mathbb{B}_{\text{cca}}$ and $\mathbb{B}_{\text{euf-cma}}$ such that*

$$\begin{aligned}\mathsf{Adv}^{\text{trac}}_{\mathrm{OVE}}(\mathbb{A}_{\text{trac}}) \leq& \mathsf{Adv}^{\text{sound}}_{\mathrm{QANIZK}}(\mathbb{B}_{\text{sound}}) + \mathsf{Adv}^{\text{zk}}_{\mathrm{QANIZK}}(\mathbb{B}_{\text{zk}}) \\ &+ \mathsf{Adv}^{\text{cca}}_{\mathrm{PKE}}(\mathbb{B}_{\text{cca}}) + \mathsf{Adv}^{\text{euf-cma}}_{\mathrm{SIG}}(\mathbb{B}_{\text{euf-cma}}).\end{aligned}$$

Proof. We introduce a series of games which the adversary $\mathbb{A}_{\text{trac}}$ plays, rendering the encryption oracle less and less potent. We bound the advantage between the games using various reductions $\mathbb{B}_{\dots}$, to finally conclude with a reduction linking the advantage in the final game to the EUF-CMA-advantage against the signature scheme.

Game $\mathbf{G_0}$: This is the original traceability game as presented in Fig. 10, see Fig. 14 for the adaption to our OVE scheme. We note that

$$\begin{aligned}\mathsf{Adv}^{\mathrm{trac}}_{\mathrm{OVE}}(\mathbb{A}_{\mathrm{trac}}) &= \Pr[\mathbb{A}_{\mathrm{trac}} \text{ wins } G_0] \\ &= \Pr\left[\mathbb{A}_{\mathrm{trac}} \text{ wins } G_0 \wedge C \in L_{(PK,VK)}\right] + \Pr\left[\mathbb{A}_{\mathrm{trac}} \text{ wins } G_0 \wedge C \notin L_{(PK,VK)}\right] ,\end{aligned}$$

where the final probability can be bounded by the advantage of a soundness adversary $\mathbb{B}_{\mathrm{sound}}$ attacking the underlying QANIZK scheme. Henceforth we assume that $\mathbb{A}_{\mathrm{trac}}$ only wins with a valid ciphertext, $C \in L_{(PK,VK)}$.

Game $\mathbf{G_1}$: This is the same as $\mathbf{G_0}$, except for the generation of π during the encryption query. Instead of generating it using Nizk.Prove, the challenger now uses a simulator. The difference in the perception of $\mathbf{G_0}$ and $\mathbf{G_1}$ for the adversary may be bounded by the advantage of a zero-knowledge adversary $\mathbb{B}_{\mathrm{zk}}$ attacking the underlying QANIZK scheme: $\Pr\left[\mathbb{A}_{\mathrm{trac}} \text{ wins } G_0 \wedge C \in L_{(PK,VK)}\right] - \Pr[\mathbb{A}_{\mathrm{trac}} \text{ wins } G_1] \leq \mathsf{Adv}^{\mathrm{zk}}_{\mathrm{QANIZK}}(\mathbb{B}_{\mathrm{zk}})$.

Game $\mathbf{G_2}$: For this game, we change the decryption oracle so that after the Nizk.Verify check is performed, it checks to see whether there is a π' such that $(C, \pi') \in \mathcal{C}$. If so, the oracle also knows which query (H, M) this was a result of, and so outputs (M, ID_H) (without further processing of C). If C is not part of a previous output of the encryption oracle, then decryption proceeds as normal. This modification does not change the adversary's view, so $\Pr[\mathbb{A}_{\mathrm{trac}} \text{ wins } G_2] = \Pr[\mathbb{A}_{\mathrm{trac}} \text{ wins } G_2]$.

Game $\mathbf{G_3}$: This game differs from the previous games in the encryption oracle. Instead of encrypting the plaintext $M \| CERT_{ID_H} \| ID_H$, it encrypts a plaintext of the same length drawn at random from the message space. The different views of the adversary in $\mathbf{G_2}$ and $\mathbf{G_3}$ is then bound by $\mathsf{Adv}^{\mathrm{ror-cca}}_{\mathrm{PKE}}(\mathbb{B}_{\mathrm{ror-cca}})$, where ror-cca denotes the real-or-random security notion for public key encryption schemes. Real-or-random security is well-known to be implied by left-or-right indistinguishability [12], namely $\mathsf{Adv}^{\mathrm{ror-cca}}_{\mathrm{PKE}}(\mathbb{B}_{\mathrm{ror-cca}}) \leq \mathsf{Adv}^{\mathrm{cca}}_{\mathrm{PKE}}(\mathbb{B}_{\mathrm{cca}})$. It follows that $\Pr[\mathbb{A}_{\mathrm{trac}} \text{ wins } G_2] - \Pr[\mathbb{A}_{\mathrm{trac}} \text{ wins } G_3] \leq \mathsf{Adv}^{\mathrm{cca}}_{\mathrm{PKE}}(\mathbb{B}_{\mathrm{cca}})$.

We may now create a reduction from EUF-CMA to traceability by constructing an adversary $\mathbb{B}_{\text{euf-cma}}$ playing $\mathbf{G_3}$ with $\mathbb{A}_{\mathrm{trac}}$, and using the output to solve her own challenge. $\mathbb{B}_{\text{euf-cma}}$ is given the verification key VK of a signature scheme, and she generates $(PK, DK) \leftarrow_{\$} \mathsf{Pke.Kg}$ and $(\sigma, \tau) \leftarrow_{\$} \mathsf{Nizk.Setup}$ herself, and finally sends (PK, VK, σ) to $\mathbb{A}_{\mathrm{trac}}$. Whenever $\mathbb{A}_{\mathrm{trac}}$ queries the derivation oracle on an identity, $\mathbb{B}_{\text{euf-cma}}$ queries her signing oracle, and forwards the signature to $\mathbb{A}_{\mathrm{trac}}$. Any other query she makes, $\mathbb{B}_{\text{euf-cma}}$ can answer using the decryption key DK and QANIZK trapdoor τ. When $\mathbb{A}_{\mathrm{trac}}$ outputs $(\hat{C}, \hat{\pi})$ as her answer, $\mathbb{B}_{\text{euf-cma}}$ decrypts $\hat{C}$, parses $M \| CERT \| ID \leftarrow \mathsf{Pke.Dec}_{DK}(\hat{C})$, and passes $(CERT, ID)$ as her forgery. Whenever $\mathbb{A}_{\mathrm{trac}}$ wins, so does $\mathbb{B}_{\text{euf-cma}}$.

From all this, it follows that

$$\mathsf{Adv}^{\mathrm{trace}}_{\mathrm{OVE}}(\mathbb{A}_{\mathrm{trace}}) \leq \mathsf{Adv}^{\mathrm{sound}}_{\mathrm{QANIZK}}(\mathbb{B}_{\mathrm{sound}}) + \mathsf{Adv}^{\mathrm{zk}}_{\mathrm{QANIZK}}(\mathbb{B}_{\mathrm{zk}}) + \mathsf{Adv}^{\mathrm{cca}}_{\mathrm{PKE}}(\mathbb{B}_{\mathrm{cca}}) + \mathsf{Adv}^{\text{euf-cma}}_{\mathrm{SIG}}(\mathbb{B}_{\text{euf-cma}}).$$

□

Integrity. The integrity of the OVE scheme follows from the zero-knowledge property of the QANIZK scheme, as well as the correctness of the PKE scheme. Informally, there are only two ways the adversary can win the game: either C has a witness, or it does not. If it does not, the adversary has been able to generate a verifiable proof for an invalid statement, which breaches the soundness of the QANIZK scheme. If C has a witness, it is generated by encrypting a plaintext, and such a ciphertext will decrypt correctly by correctness of PKE, so winning this way is not possible.

Lemma 5 (Integrity of OVE). *For all adversaries $\mathbb{A}_{\mathrm{int}}$, there exist an equally efficient adversary $\mathbb{B}_{\mathrm{sound}}$ such that*

$$\mathsf{Adv}^{\mathrm{int}}_{\mathrm{OVE}}(\mathbb{A}_{\mathrm{int}}) \leq \mathsf{Adv}^{\mathrm{sound}}_{\mathrm{QANIZK}}(\mathbb{B}_{\mathrm{sound}}) .$$

Proof. We present the integrity game for the OVE scheme in Fig. 15. The advantage of $\mathbb{A}_{\mathrm{int}}$ is

$$\begin{aligned}\Pr\left[\mathsf{Exp}^{\mathrm{int}}_{\mathrm{OVE}}(\mathbb{A}_{\mathrm{int}}) = 1\right] &= \Pr\left[\mathsf{Exp}^{\mathrm{int}}_{\mathrm{OVE}}(\mathbb{A}_{\mathrm{int}}) = 1 \wedge \hat{C} \in L_{(PK,VK)}\right] \\ &\quad + \Pr\left[\mathsf{Exp}^{\mathrm{int}}_{\mathrm{OVE}}(\mathbb{A}_{\mathrm{int}}) = 1 \wedge \hat{C} \notin L_{(PK,VK)}\right] ,\end{aligned}$$

where the latter probability may be bounded by the advantage of a soundness adversary against the QANIZK scheme, as the definition of the two adversaries match.

With regards to the former probability, $\hat{C} \in L_{(PK,VK)}$ implies that, for some $M, CERT_{ID}$, and ID, $\hat{C} = \mathsf{Pke.Enc}_{PK}(M \| CERT_{ID} \| ID; r)$. Correctness of the encryption scheme ensure that decryption will uniquely recover $M, CERT_{ID}$, and ID, and Ove.Dec will not reject. Thus the corresponding probability is zero. □

```
$\mathsf{Exp}^{\mathrm{int}}_{\mathrm{OVE}}(\mathbb{A})$
$(PK, DK) \leftarrow\$ \mathsf{Pke.Kg}$
$(VK, SK) \leftarrow \mathsf{Sig.Kg}$
$(\sigma, \tau) \leftarrow\$ \mathsf{Nizk.Setup}$
$h \leftarrow 0; C \leftarrow \emptyset$
$\mathcal{CU} \leftarrow \emptyset$
$(\hat{C}, \hat{\pi}) \leftarrow \mathbb{A}^{O}((PK, VK, \sigma), SK, DK)$
$\mathsf{Ove.Verify}_{PK}(\hat{C}) = \top \wedge \mathsf{Ove.Dec}_{DK}(\hat{C}, \hat{\pi}) = (\perp, \perp)$
```

Fig. 15. The integrity game for our Verifiably Encrypted Certificate construction for OVE.

```
Game $\mathbf{G_1}^{b^*}$
$(PK, DK) \leftarrow\$ \mathsf{Pke.Kg}$
$(VK, SK) \leftarrow \mathsf{Sig.Kg}$
$(\sigma, \tau) \leftarrow\$ \mathsf{Nizk.Setup}$
$C \leftarrow \emptyset$
$\hat{b} \leftarrow \mathbb{A}^{O}((PK, VK, \sigma), SK)$

encrypt$((M_0, CERT_{ID_0}, ID_0), (M_1, CERT_{ID_1}, ID_1))$
$C_{b^*} \leftarrow\$ \mathsf{Pke.Enc}_{PK}(M_{b^*} \| CERT_{ID_{b^*}} \| ID_{b^*}; r)$
$\pi_{b^*} \leftarrow \mathsf{Nizk.Prove}_{PK,VK,\sigma}(r, M_{b^*}, CERT_{ID_{b^*}}, ID_{b^*})$
$C^* \leftarrow (C_{b^*}, \pi_{b^*})$
$C \leftarrow C \cup \{C^*\}$return $C^*$

decrypt$(C, \pi)$
require $(C, \pi) \notin C$
if $\mathsf{Nizk.Verify}_{PK,VK,\sigma}(C, \pi) = \perp$
    return $(\perp, \perp)$
$M \| CERT_{ID} \| ID \leftarrow \mathsf{Pke.Dec}_{DK}(C)$
if decryption or parsing fails
    return $(\perp, \perp)$
return $(M, ID)$
```

Fig. 16. Game $\mathbf{G}_1^{b^*}$ for the privacy proof of our Verifiably Encrypted Certificate construction for OVE.

Privacy. The notion of privacy for the OVE rests on the security of the underlying encryption scheme and QANIZK protocol. In essence, the CCA notion of the PKE ensures that the C component does not leak any information about the message or the identity, whilst the zk notion of the QANIZK protocol guards against the proof π revealing anything useful to an adversary. Finally, the simulation soundness of the QANIZK helps guarantee that the adversary cannot forge a proof π', and thus take advantage of a decryption oracle.

Lemma 6 (Privacy of OVE). *For all adversaries $\mathbb{A}_{\mathrm{priv}}$, there exist similarly efficient adversaries $\mathbb{B}_{\mathrm{uss}}$, $\mathbb{B}_{\mathrm{zk}}$ and $\mathbb{B}_{\mathrm{cca}}$ such that*

$$\mathsf{Adv}^{\mathrm{priv}}_{\mathrm{OVE}}(\mathbb{A}_{\mathrm{priv}}) \leq 2\mathsf{Adv}^{\mathrm{zk}}_{\mathrm{QANIZK}}(\mathbb{B}_{\mathrm{zk}}) + 2\mathsf{Adv}^{\mathrm{uss}}_{\mathrm{QANIZK}}(\mathbb{B}_{\mathrm{uss}}) + 3\mathsf{Adv}^{\mathrm{cca}}_{\mathrm{PKE}}(\mathbb{B}_{\mathrm{cca}}) .$$

Proof. Just as in the traceability game, we introduce a series of games for the adversary $\mathbb{A}_{\mathrm{priv}}$ to play, which gradually changes the original game into a reduction to the CCA game against the underlying encryption scheme.

Game $\mathbf{G}_0^{b^*}$: This is the original game, presented in Fig. 12, applied to our construction. The advantage of $\mathbb{A}_{\mathrm{priv}}$ may be expressed as $\mathsf{Adv}^{\mathrm{priv}}_{\mathrm{OVE}}(\mathbb{A}_{\mathrm{priv}}) = \Pr\left[\mathbf{G}_0^0 : \mathbb{A}_{\mathrm{priv}} \rightarrow 1\right] - \Pr\left[\mathbf{G}_0^1 : \mathbb{A}_{\mathrm{priv}} \rightarrow 1\right]$.

Game $\mathbf{G}_1^{b^*}$: In this game, we assume that the adversary will only forward valid encryption queries, i.e., all queried certificates validates as signatures for identities. We therefore do not need any checks of the validity of signatures in the game and can simplify accordingly, see Fig. 16. The restriction is without loss of generality, as an adversary can check the validity of the certificates. Thus, for $b^* \in \{0, 1\}$, we have $\Pr\left[\mathbf{G}_0^{b^*} : \mathbb{A}_{\mathrm{priv}} \rightarrow 1\right] = \Pr\left[\mathbf{G}_1^{b^*} : \mathbb{A}_{\mathrm{priv}} \rightarrow 1\right]$.

Game $\mathbf{G}_2^{b^*}$: Here, we change the generation of π during the encryption query, so that $\pi \leftarrow \mathsf{Nizk.Sim}_{\tau}(C)$. For both possible values of b^*, the difference in the adversary's view between $\mathbf{G}_1^{b^*}$ and $\mathbf{G}_2^{b^*}$ may be bounded by the advantage of an adversary $\mathbb{B}_{\mathrm{zk}}$ attacking the zero-knowledge property of the underlying QANIZK scheme, i.e., $\Pr\left[\mathbf{G}_1^{b^*} : \mathbb{A}_{\mathrm{priv}} \rightarrow 1\right] - \Pr\left[\mathbf{G}_2^{b^*} : \mathbb{A}_{\mathrm{priv}} \rightarrow 1\right] \leq \mathsf{Adv}^{\mathrm{zk}}_{\mathrm{QANIZK}}(\mathbb{B}_{\mathrm{zk}})$.

Game $\mathbf{G}_3^{b^*}$: In the final game, we replace the decryption procedure, so that any decryption query of the format (C, π) where C has been part of a challenge output, yet π was not, is rejected. In other words: we do not allow the privacy adversary to query challenge ciphertexts with new, valid proofs (obviously invalid proofs would be rejected regardless). The games $\mathbf{G}_2^{b^*}$ and $\mathbf{G}_3^{b^*}$ are therefore identical-until-bad, and we will analyse the probability of the bad event in the final step of the proof.

Given an adversary distinguishing between $\mathbf{G}_3^0$ and $\mathbf{G}_3^1$, we may construct a reduction to the CCA-security of the PKE as follows. An adversary $\mathbb{B}^2_{\mathrm{cca}}$ who is given the public key PK of an encryption scheme PKE sets up a signature scheme with keys $(VK, SK) \leftarrow_{\$} \mathsf{Sig.Kg}$ and a QANIZK with $(\sigma, \tau) \leftarrow_{\$} \mathsf{Nizk.Setup}$, and sends $((PK, VK, \sigma), SK)$ to $\mathbb{A}_{\mathrm{priv}}$. Encryption queries for $((M_0, CERT_{ID_0}, ID_0), (M_1, CERT_{ID_1}, ID_1))$ are answered by $\mathbb{B}_{\mathrm{cca}}$ querying her decryption oracle with $(M_0 \| CERT_{ID_0} \| ID_0, M_1 \| CERT_{ID_1} \| ID_1)$ then simulating a proof π on the received challenge ciphertext C, and sending (C, π) to $\mathbb{A}_{\mathrm{priv}}$. For any decryption query of (C', π') by $\mathbb{A}_{\mathrm{priv}}$, $\mathbb{B}^2_{\mathrm{cca}}$ rejects the query if π' does not verify, or $C = C'$. Otherwise, she sends C' to her decryption oracle: if it returns $\perp$, then $\mathbb{B}^2_{\mathrm{cca}}$ returns $(\perp, \perp)$; if not, $\mathbb{B}^2_{\mathrm{cca}}$ parses the received plaintext as $M \| CERT_{ID} \| ID$ and returns (M, ID). When $\mathbb{A}_{\mathrm{priv}}$ outputs $\hat{b}$, $\mathbb{B}^2_{\mathrm{cca}}$ copies it, and thus it follows that $\Pr\left[\mathbf{G}^1_3 : \mathbb{A}_{\mathrm{priv}} \to 1\right] - \Pr\left[\mathbf{G}^0_3 : \mathbb{A}_{\mathrm{priv}} \to 1\right] \leq \mathsf{Adv}^{\mathrm{cca}}_{\mathrm{PKE}}(\mathbb{B}^2_{\mathrm{cca}})$.

Finally, we bound the probability of the bad event in game $\mathbf{G}^{b^*}_3$, where the adversary queries the decryption oracle with a tuple consisting of a challenge ciphertext C and a new, valid proof π'. We introduce a new game, $\mathbf{G}^{b^*}_{\mathbf{x}}$ where any encryption query is answered as follows: draw a plaintext at random from the plaintext space, of the same length as a plaintext from an honest query. The plaintext is then encrypted to C, and a proof π for it is simulated, and (C, π) is sent to $\mathbb{A}_{\mathrm{priv}}$. For both values of b^*, we then have $\Pr\left[\mathbf{G}^{b^*}_3 : \mathrm{Bad}\right] - \Pr\left[\mathbf{G}^{b^*}_{\mathbf{x}} : \mathrm{Bad}\right] \leq \mathsf{Adv}^{\mathrm{ror-cca}}_{\mathrm{PKE}}(\mathbb{B}_{\mathrm{ror-cca}})$, where ror-cca denotes the real-or-random security notion for public key encryption schemes. It is well-known that real-or-random security is implied by left-or-right indistinguishability [12]: $\mathsf{Adv}^{\mathrm{ror-cca}}_{\mathrm{PKE}}(\mathbb{B}_{\mathrm{ror-cca}}) \leq \mathsf{Adv}^{\mathrm{cca}}_{\mathrm{PKE}}(\mathbb{B}_{\mathrm{cca}})$. Furthermore, $\Pr\left[\mathbf{G}^{b^*}_{\mathbf{x}} : \mathrm{Bad}\right] \leq \mathsf{Adv}^{\mathrm{uss}}_{\mathrm{QANIZK}}(\mathbb{B}_{\mathrm{uss}})$, and so the following inequality $\Pr\left[\mathbf{G}^{b^*}_3 : \mathrm{Bad}\right] \leq \mathsf{Adv}^{\mathrm{cca}}_{\mathrm{PKE}}(\mathbb{B}^{b^*}_{\mathrm{cca}}) + \mathsf{Adv}^{\mathrm{uss}}_{\mathrm{QANIZK}}(\mathbb{B}_{\mathrm{uss}})$ holds for both values of b^*.

A final detail is combining the three different CCA adversaries from game $\mathbf{G}_3$, $\mathbb{B}^0_{\mathrm{cca}}$, $\mathbb{B}^1_{\mathrm{cca}}$ and $\mathbb{B}^2_{\mathrm{cca}}$ by constructing a 'master' adversary $\mathbb{B}_{\mathrm{cca}}$. This adversary plays the CCA game by uniformly at random picking which sub-reduction to run. We therefore have: $\mathsf{Adv}^{\mathrm{cca}}_{\mathrm{PKE}}(\mathbb{B}_{\mathrm{cca}}) = \frac{1}{3}\mathsf{Adv}^{\mathrm{cca}}_{\mathrm{PKE}}(\mathbb{B}^0_{\mathrm{cca}}) + \frac{1}{3}\mathsf{Adv}^{\mathrm{cca}}_{\mathrm{PKE}}(\mathbb{B}^1_{\mathrm{cca}}) + \frac{1}{3}\mathsf{Adv}^{\mathrm{cca}}_{\mathrm{PKE}}(\mathbb{B}^2_{\mathrm{cca}})$. We finally conclude that:

$$\mathsf{Adv}^{\mathrm{priv}}_{\mathrm{OVE}}(\mathbb{A}_{\mathrm{priv}}) \leq 2\mathsf{Adv}^{\mathrm{zk}}_{\mathrm{QANIZK}}(\mathbb{B}_{\mathrm{zk}}) + 2\mathsf{Adv}^{\mathrm{uss}}_{\mathrm{QANIZK}}(\mathbb{B}_{\mathrm{uss}}) + 3\mathsf{Adv}^{\mathrm{cca}}_{\mathrm{PKE}}(\mathbb{B}_{\mathrm{cca}}).$$

□

We note that our bound is not as tight as the corresponding one for anonymity in BMW. The difference is primarily due to the proof strategy: instead of game hops, Bellare et al. directly provided the code of two CCA adversaries that integrated a bad event and a hop between two games $\mathbf{G}^0$ and $\mathbf{G}^1$, coupled with a refined analysis of the relevant advantages. The integrated approach allowed for some terms in the derivation to cancel, leading to the slighlty tighter bound. We opted for simplicity instead, also as we deal with multi-query games as opposed to the single-query games in the BMW construction. Thus we can potentially avoid a tightness loss as a result of a hybrid argument by plugging in appopriate multi-query secure primitives.

4.3 Discussion of OVE

To the best of our knowledge, OVE schemes offer a combination of functionality and security hitherto unstudied. However, as mentioned before, there are great similarities with group signatures, with the crucial distinction that group signatures do not offer message recovery, nor confidentiality of messages. Our construction was directly inspired by the BMW construction for group signatures [13], with some notable differences. In the following, we explore these differences and also address how ideas from other group signature schemes might apply to OVE. Finally, we briefly compare signcryption to OVE.

A significant difference between our construction and BMW's sign-encrypt-proof is the use of signatures. In the BMW group signature scheme, the user signing key consists of a personal key pair for the signature scheme in addition to a certificate binding the personal verification key to the identifying index. When signing a message, the sender first signs the message using their personal signing key, and then encrypts *this* signature, along with the certificate and personal verification key. This may be regarded as a signature tree of depth two, as the certificate is a signature on the verification key. This indirection enables full traceability, so that even an adversary with access to the group master opening key is unable to forge a signature of an uncompromised group member.

We flattened the construction by removing the personal signature key-pair. The gain in efficiency results in our weaker notion of traceability: an adversary in possession of the secret key of our scheme can readily decrypt a ciphertext to learn the identity and certificate of an honest user, and subsequently send any message in the name of this user. As discussed previously, this weaker notion suits the intended use of the OVE scheme, where the recipient who holds the secret key has no motivation of sending spam to themselves. Our perspective is that the recipient, holding the decryption key "owns" the system yet might wish to delegate the vetting: thus we introduce separate keys and insist privacy holds against the issuer, but traceability need not hold against the decryptor. A further weakening would completely identify issuer and decryptor as Kiayias and Yung considered for group signatures [36]. If the stronger version of traceability is deemed desirable for OVE, a closer fit with BMW should work.

One difference between OVE and the BMW framework is how the identity of the sender, resp. group member, is treated. For OVE, the identity itself, as input to the key derivation, is retrieved during decryption. For BMW, the identity is linked to an index instead, and it is this index which is part of the various algorithms. In order to get the actual identity of the group member, an additional look up table is required, necessitating further coordination between the issuing of keys and the opening of signatures. With some abuse of naming, we will nevertheless refer to this index i as (part of) the identity in what follows. A side-effect of BMW's use of indices is that they do not model a separate key derivation algorithm, instead generating all user keys as part of the initial key generation. One implication is that, syntactically, users can no longer be added to the group after set-up: this would require regenerating new keys for everyone. Obviously for the construction, it is straightforward to isolate an issuing algorithm, and adding users on the fly is not an issue.

Separate key derivation, or issuing, algorithms are known from dynamic group signature schemes [16, 19], where a useful distinction can be made between partially dynamic schemes where users can join but may never leave, and fully dynamic where a user's credentials may be revoked. A noticeable difference between the dynamic group signatures and OVE is that the former binds signatures to a PKI, providing non-repudiation and requiring the opener to output a proof to demonstrate publicly that the purported identity of signer of the message is correct. These differences render adaptation of the known group signature schemes less immediate as simplifications can likely be made—with the appropriate care. For instance, Groth [33] suggests increasing the depth of the signature tree to three by incorporating an additional one-time secure signature scheme. The advantage of his approach is much more efficient instantiations of the underlying primitives, including the NIZK, resulting in constant size group signatures. Similar ideas might be useful for optimizing OVE.

A more challenging inspiration for OVE arises from a brand new paradigm to construct compact and efficient group signatures based on structure preserving signatures (SPS) and signatures of knowledge (SoK) [2, 37, 29]. Here the signing algorithm does not involve an encryption scheme. Instead, the SPS is used to find a new representative of the user key, which is then signed along with the message using a SoK. Adaption to the OVE setting likely requires some additional tweaking, for example letting the SoK sign an encryption of the desired message, rather than the message itself.

So far we have only looked at the Hotel California situation where users are added dynamically, but they can never leave. The most challenging scenario for OVE is one where senders may become unvetted, such that their ciphertexts no longer pass the filter. This corresponds to fully dynamic group signatures [19], which can be achieved based on an accountable ring signature scheme (the signing of the message is simply applying the signing algorithm of said ring signature scheme). Adding unvetting would be a useful feature to OVE, but ideally without incurring the overhead of ring signatures: black listing at the filter is probably easier to achieve than the white listing at the senders (implicit when using ring signatures).

Finally, we note that generic transforms from either group signatures or signcryption to OVE are less obvious. For signcryption schemes, as we observed before, the combination of hiding the sender while still allowing for public verification appear mutually exclusive. On the other hand, a simple encrypt-then-groupsign transform fails privacy, as user Eve can simply intercept user Anna's ciphertext and supplant the group signature with one of her own, and ask for it to be decrypted. Where for IVE, unicity of signatures prevented such an attack, here no such protection is possible. Also a group signature's implicit encryption capacity [1, 31] appears hard to unlock generically to serve OVE.

5 Conclusion

We introduced vetted encryption, which allows a recipient to specify who is allowed to send them messages and outsource the filtering to any third party. We concentrated on only a single receiver in two distinct scenarios: the filter would or would not learn the identity of the sender. Either way, the sender would remain identifiable to the recipient. OVE has the potential to facilitate confidential communication with whistleblowers, sources for journalists and other scenarios for anonymous communication where an organization wants to filter the anonymous traffic, yet the individual needs to be identified to the recipient in a way that is convincing to the recipient while allowing repudiation by the sender.

When considering multiple receivers, a possible extension would be to allow a single filter in such a way that the intended recipient remains anonymous to the filter as well. Such an extension could be relevant for all three types of vetted encryption, though it is possibly more natural in the AVE and OVE setting. We have, after all, already lifted anonymity from the filter altogether in the IVE setting, which at the very least opens up the possibility to use the recipient's identity.

For identifiable vetted encryption we made the link with signcryption; one could further try to extend this link by considering an alternative multi-recipient scenario where a single sender wants to transmit the same message to multiple recipients simultaneously. This is quite common in email applications and one expects some performance benefits due to amortization (though the security definitions might become more complex, cf. multi-user signcryption).

Finally, for our constructions we concentrated on proofs of concepts. For both IVE and OVE we leave open the challenge of designing the most efficient scheme, either by suitably instantiating our generic construction or by taking further inspiration from, respectively, signcryption and group signatures, and beyond. Another possible feature for either primitive would be to revoke the right to send.

References

1. Michel Abdalla and Bogdan Warinschi. On the minimal assumptions of group signature schemes. In Javier López, Sihan Qing, and Eiji Okamoto, editors, *ICICS 04*, volume 3269 of *LNCS*, pages 1–13. Springer, Heidelberg, October 2004.
2. Masayuki Abe, Georg Fuchsbauer, Jens Groth, Kristiyan Haralambiev, and Miyako Ohkubo. Structure-preserving signatures and commitments to group elements. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 209–236. Springer, Heidelberg, August 2010.
3. Masayuki Abe, Charanjit S. Jutla, Miyako Ohkubo, Jiaxin Pan, Arnab Roy, and Yuyu Wang. Shorter QA-NIZK and SPS with tighter security. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part III*, volume 11923 of *LNCS*, pages 669–699. Springer, Heidelberg, December 2019.
4. Jee Hea An, Yevgeniy Dodis, and Tal Rabin. On the security of joint signature and encryption. In Lars R. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 83–107. Springer, Heidelberg, April / May 2002.
5. Jee Hea An and Tal Rabin. Security for signcryption: The two-user model. In Dent and Zheng [28], pages 21–42.
6. Elena Andreeva, Andrey Bogdanov, Atul Luykx, Bart Mennink, Nicky Mouha, and Kan Yasuda. How to securely release unverified plaintext in authenticated encryption. In Palash Sarkar and Tetsu Iwata, editors, *ASIACRYPT 2014, Part I*, volume 8873 of *LNCS*, pages 105–125. Springer, Heidelberg, December 2014.
7. Giuseppe Ateniese, Danilo Francati, David Nuñez, and Daniele Venturi. Match me if you can: Matchmaking encryption and its applications. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part II*, volume 11693 of *LNCS*, pages 701–731. Springer, Heidelberg, August 2019.
8. Christian Badertscher, Fabio Banfi, and Ueli Maurer. A constructive perspective on signcryption security. In Dario Catalano and Roberto De Prisco, editors, *SCN 18*, volume 11035 of *LNCS*, pages 102–120. Springer, Heidelberg, September 2018.
9. Feng Bao and Robert H. Deng. A signcryption scheme with signature directly verifiable by public key. In Hideki Imai and Yuliang Zheng, editors, *PKC'98*, volume 1431 of *LNCS*, pages 55–59. Springer, Heidelberg, February 1998.
10. Paulo S. L. M. Barreto, Benoît Libert, Noel McCullagh, and Jean-Jacques Quisquater. Signcryption schemes based on bilinear maps. In Dent and Zheng [28], pages 71–97.
11. Mihir Bellare, Alexandra Boldyreva, Anand Desai, and David Pointcheval. Key-privacy in public-key encryption. In Colin Boyd, editor, *ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 566–582. Springer, Heidelberg, December 2001.
12. Mihir Bellare, Anand Desai, Eric Jokipii, and Phillip Rogaway. A concrete security treatment of symmetric encryption. In *38th FOCS*, pages 394–403. IEEE Computer Society Press, October 1997.
13. Mihir Bellare, Daniele Micciancio, and Bogdan Warinschi. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In Eli Biham, editor, *EUROCRYPT 2003*, volume 2656 of *LNCS*, pages 614–629. Springer, Heidelberg, May 2003.
14. Mihir Bellare, Chanathip Namprempre, and Gregory Neven. Security proofs for identity-based identification and signature schemes. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 268–286. Springer, Heidelberg, May 2004.
15. Mihir Bellare and Phillip Rogaway. The exact security of digital signatures: How to sign with RSA and Rabin. In Ueli M. Maurer, editor, *EUROCRYPT'96*, volume 1070 of *LNCS*, pages 399–416. Springer, Heidelberg, May 1996.
16. Mihir Bellare, Haixia Shi, and Chong Zhang. Foundations of group signatures: The case of dynamic groups. In Alfred Menezes, editor, *CT-RSA 2005*, volume 3376 of *LNCS*, pages 136–153. Springer, Heidelberg, February 2005.
17. Tor E. Bjørstad. Hybrid signcryption. In Dent and Zheng [28], pages 121–147.
18. Tor E. Bjørstad and Alexander W. Dent. Building better signcryption schemes with tag-KEMs. In Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, and Tal Malkin, editors, *PKC 2006*, volume 3958 of *LNCS*, pages 491–507. Springer, Heidelberg, April 2006.
19. Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Essam Ghadafi, and Jens Groth. Foundations of fully dynamic group signatures. In Mark Manulis, Ahmad-Reza Sadeghi, and Steve Schneider, editors, *ACNS 16*, volume 9696 of *LNCS*, pages 117–136. Springer, Heidelberg, June 2016.
20. Xavier Boyen. Multipurpose identity-based signcryption (a swiss army knife for identity-based cryptography). In Dan Boneh, editor, *CRYPTO 2003*, volume 2729 of *LNCS*, pages 383–399. Springer, Heidelberg, August 2003.
21. Xavier Boyen. Identity-based signcryption. In Dent and Zheng [28], pages 195–216.
22. Liqun Chen and John Malone-Lee. Improved identity-based signcryption. In Serge Vaudenay, editor, *PKC 2005*, volume 3386 of *LNCS*, pages 362–379. Springer, Heidelberg, January 2005.
23. Jean-Sébastien Coron. On the exact security of full domain hash. In Mihir Bellare, editor, *CRYPTO 2000*, volume 1880 of *LNCS*, pages 229–235. Springer, Heidelberg, August 2000.
24. Véronique Cortier, Pierrick Gaudry, and Stéphane Glondu. Belenios: A simple private and verifiable electronic voting system. In *Foundations of Security, Protocols, and Equational Reasoning*, volume 11565 of *Lecture Notes in Computer Science*, pages 214–238. Springer, 2019.
25. Ronald Cramer and Victor Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM Journal on Computing*, 33(1):167–226, 2003.
26. Ivan Damgård, Helene Haagh, and Claudio Orlandi. Access control encryption: Enforcing information flow with cryptography. In Martin Hirt and Adam D. Smith, editors, *TCC 2016-B, Part II*, volume 9986 of *LNCS*, pages 547–576. Springer, Heidelberg, October / November 2016.
27. Alexander W. Dent, Marc Fischlin, Mark Manulis, Martijn Stam, and Dominique Schröder. Confidential signatures and deterministic signcryption. In Phong Q. Nguyen and David Pointcheval, editors, *PKC 2010*, volume 6056 of *LNCS*, pages 462–479. Springer, Heidelberg, May 2010.
28. Alexander W. Dent and Yuliang Zheng, editors. *Practical Signcryption.* ISC. Springer, Heidelberg, 2010.
29. David Derler and Daniel Slamanig. Highly-efficient fully-anonymous dynamic group signatures. In Jong Kim, Gail-Joon Ahn, Seungjoo Kim, Yongdae Kim, Javier López, and Taesoo Kim, editors, *ASIACCS 18*, pages 551–565. ACM Press, April 2018.
30. Yevgeniy Dodis and Aleksandr Yampolskiy. A verifiable random function with short proofs and keys. In Serge Vaudenay, editor, *PKC 2005*, volume 3386 of *LNCS*, pages 416–431. Springer, Heidelberg, January 2005.

31. Keita Emura, Goichiro Hanaoka, and Yusuke Sakai. Group signature implies PKE with non-interactive opening and threshold PKE. In Isao Echizen, Noboru Kunihiro, and Ryôichi Sasaki, editors, *IWSEC 10*, volume 6434 of *LNCS*, pages 181–198. Springer, Heidelberg, November 2010.
32. Chandana Gamage, Jussipekka Leiwo, and Yuliang Zheng. Encrypted message authentication by firewalls. In Hideki Imai and Yuliang Zheng, editors, *PKC'99*, volume 1560 of *LNCS*, pages 69–81. Springer, Heidelberg, March 1999.
33. Jens Groth. Simulation-sound NIZK proofs for a practical language and constant size group signatures. In Xuejia Lai and Kefei Chen, editors, *ASIACRYPT 2006*, volume 4284 of *LNCS*, pages 444–459. Springer, Heidelberg, December 2006.
34. Ik Rae Jeong, Hee Yun Jeong, Hyun Sook Rhee, Dong Hoon Lee, and Jong In Lim. Provably secure encrypt-then-sign composition in hybrid signcryption. In Pil Joong Lee and Chae Hoon Lim, editors, *ICISC 02*, volume 2587 of *LNCS*, pages 16–34. Springer, Heidelberg, November 2003.
35. Charanjit S. Jutla and Arnab Roy. Shorter quasi-adaptive NIZK proofs for linear subspaces. In Kazue Sako and Palash Sarkar, editors, *ASIACRYPT 2013, Part I*, volume 8269 of *LNCS*, pages 1–20. Springer, Heidelberg, December 2013.
36. Aggelos Kiayias and Moti Yung. Group signatures: Provable security, efficient constructions and anonymity from trapdoor-holders. Cryptology ePrint Archive, Report 2004/076, 2004. http://eprint.iacr.org/2004/076.
37. Benoît Libert, Thomas Peters, and Moti Yung. Short group signatures via structure-preserving signatures: Standard model security from simple assumptions. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 296–316. Springer, Heidelberg, August 2015.
38. Benoît Libert and Jean-Jacques Quisquater. New identity based signcryption schemes from pairings. Cryptology ePrint Archive, Report 2003/023, 2003. http://eprint.iacr.org/2003/023.
39. Benoît Libert and Jean-Jacques Quisquater. Efficient signcryption with key privacy from gap Diffie-Hellman groups. In Feng Bao, Robert Deng, and Jianying Zhou, editors, *PKC 2004*, volume 2947 of *LNCS*, pages 187–200. Springer, Heidelberg, March 2004.
40. Benoît Libert and Jean-Jacques Quisquater. Improved signcryption from q-Diffie-Hellman problems. In Carlo Blundo and Stelvio Cimato, editors, *SCN 04*, volume 3352 of *LNCS*, pages 220–234. Springer, Heidelberg, September 2005.
41. Anna Lysyanskaya. Unique signatures and verifiable random functions from the DH-DDH separation. In Moti Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*, pages 597–612. Springer, Heidelberg, August 2002.
42. Noel McCullagh and Paulo S. L. M. Barreto. Efficient and forward-secure identity-based signcryption. Cryptology ePrint Archive, Report 2004/117, 2004. http://eprint.iacr.org/2004/117.
43. Kenneth G. Paterson and Jacob C. N. Schuldt. Efficient identity-based signatures secure in the standard model. In Lynn Margaret Batten and Reihaneh Safavi-Naini, editors, *ACISP 06*, volume 4058 of *LNCS*, pages 207–222. Springer, Heidelberg, July 2006.
44. S. Sharmila Deva Selvi, S. Sree Vivek, Dhinakaran Vinayagamurthy, and C. Pandu Rangan. ID based signcryption scheme in standard model. In Tsuyoshi Takagi, Guilin Wang, Zhiguang Qin, Shaoquan Jiang, and Yong Yu, editors, *ProvSec 2012*, volume 7496 of *LNCS*, pages 35–52. Springer, Heidelberg, September 2012.
45. Shiuan-Tzuo Shen, Amir Rezapour, and Wen-Guey Tzeng. Unique signature with short output from CDH assumption. In Man Ho Au and Atsuko Miyaji, editors, *ProvSec 2015*, volume 9451 of *LNCS*, pages 475–488. Springer, Heidelberg, November 2015.
46. Yuliang Zheng. Digital signcryption or how to achieve cost(signature & encryption) ≪ cost(signature) + cost(encryption). In Burton S. Kaliski Jr., editor, *CRYPTO'97*, volume 1294 of *LNCS*, pages 165–179. Springer, Heidelberg, August 1997.

A Unique Identity Based Signature Scheme

We construct an *identity based signature scheme with unique signatures* (UIBSS) by using the known certificate based transformation on an unique signature scheme [14]. We generate a master signing and verification key for a USS scheme, as well as set up a PRF. Given an identity ID we use the PRF to derive randomness, which is then fed into the key generation algorithm of a USS scheme. In other words: the key generation of the USS is derandomised, with the given randomness depending on a given identity. The resulting key pair (SK, VK) are components of the user key USK of ID. The key USK also includes a certificate, which is $VK \| ID$ signed under the master signing key. A signature of a message M in the UIBSS scheme is simply $(\sigma, VK, CERT_{ID})$, where $\sigma \leftarrow \mathsf{Uss.Sign}(SK, M)$. Finally, verification requires that both signatures σ and $CERT_{ID}$ verifies on M and $VK \| ID$, respectively. We present the construction of our UIBSS in Fig. 17, see 2 for the syntax of IBS schemes.

We adopt the security notion of existential unforgeability of identity based signature schemes to our scheme [43]. Informally, the notion states that given access to a signing oracle and a corruption oracle, an adversary should not be able to find a tuple (M, ID, σ) which passes the verification algorithm, where she has not asked to corrupt ID, and not asked for a signature on (M, ID). Since the general certificate construction has been proven to produce identity-based signature schemes that satisfy this notion of security, it follows that our scheme is secure with respect to existential unforgeability [14].

For unique signature schemes $\mathsf{Uss.Verify}_{VK}(M, \sigma) = \mathsf{Uss.Verify}_{VK}(M, \sigma')$ implies $\sigma = \sigma'$ [45]. However, we will relax this requirement, and rather require that it is computationally hard for an adversary to win the following game: given the verification key, and access to a derivation oracle, find a tuple $(M, ID, \varsigma, \varsigma')$ such that $\mathsf{Uibss.Verify}_{VK,ID}(M, \varsigma) = \top = \mathsf{Uibss.Verify}_{VK,ID}(M, \varsigma')$, yet $\varsigma \neq \varsigma'$. We define this security notion as *outsider unicity*, with the game formally defined in Fig. 18. As always, the advantage of the adversary is her probability of winning the game.

```
Uibss.Kg()
    $(MSK, MVK) \leftarrow$ Uss.Kg()
    $k \leftarrow$ Prf.Kg()
    return $((MSK, k), MVK)$

Uibss.Derive$_{(MSK,k)}(ID)$
    $R \leftarrow$ PRF$(k, ID)$
    $(SK, VK) \leftarrow$ Uss.Kg$(; R)$
    $CERT_{ID} \leftarrow$ Uss.Sign$(MSK, VK \| ID)$
    return $USK \leftarrow (SK, VK, CERT_{ID})$

Uibss.Sign$_{USK,ID}(M)$
    $\sigma \leftarrow$ Uss.Sign$(SK, M)$
    return $\varsigma \leftarrow (\sigma, VK, CERT_{ID})$

Uibss.Verify$_{MVK,ID}(M, \varsigma)$
    if Uss.Verify$_{MVK}(VK \| ID, CERT_{ID}) = \perp$
        return $\perp$
    if Uss.Verify$_{VK}(M, \sigma) = \perp$
        return $\perp$
    else
        return $\top$
```

Fig. 17. The construction of an identity based signature scheme with unique signatures using a unique signature scheme (USS) and a psuedo random function (PRF). Note that we denote the signing and verification key generated by Uss.Kg during Uibss.Kg as (MSK, MVK) solely to distinguish these keys from the signing and verification keys that constitute the USK of a particular ID.

```
Exp$^{\mathrm{ou}}_{\mathrm{UIBSS}}(\mathbb{A})$
    $(MSK, VK) \leftarrow$ Uibss.Kg
    $(\hat{ID}, \hat{M}, \hat{\sigma}, \hat{\sigma}') \leftarrow \mathbb{A}^{O}(PK)$
    winif Uibss.Verify$_{VK}(\hat{ID}, \hat{M}, \hat{\sigma}) = \top \wedge$
        Uibss.Verify$_{VK}(\hat{ID}, \hat{M}, \hat{\sigma}') = \top \wedge \hat{\sigma} \neq \hat{\sigma}'$

derive$_{MSK}(ID)$
    return $USK \leftarrow$ Uibss.Derive$_{MSK}(ID)$
```

Fig. 18. The outsider unicity game for unique identity based signature schemes.

Our certificate based UIBSS scheme achieves outsider unicity due to the unicity property of the underlying unique signature scheme, as well as it's notion of unforgeability. Informally, the unicity of signatures forces an adversary to find a forgery on $VK \| ID$.

Lemma 7 (Outsider unicity of UIBSS construction). *For all adversaries $\mathbb{A}_{\mathrm{ou}}$, there exists an adversary $\mathbb{B}_{\text{euf-cma}}$ such that*

$$\mathsf{Adv}^{\mathrm{ou}}_{\mathrm{UIBSS}}(\mathbb{A}_{\mathrm{ou}}) \leq \mathsf{Adv}^{\text{euf-cma}}_{\mathrm{USS}}(\mathbb{B}_{\text{euf-cma}}).$$

Proof. The adversary $\mathbb{B}_{\text{euf-cma}}$ is given a verification key MVK, which she passes on to $\mathbb{A}_{\mathrm{ou}}$, and creates a key k from Prf.Kg. Whenever $\mathbb{A}_{\mathrm{ou}}$ sends a derivation query for an identity ID, $\mathbb{B}_{\text{euf-cma}}$ generates $R \leftarrow \mathsf{PRF}(k, ID)$, which she uses to derive $(SK, VK) \leftarrow \mathsf{Uss.Derive}(; R)$. She then queries her signing oracle with the message $VK \| ID$, and uses the received signature as $CERT_{ID}$. She then sends $(SK, VK, CERT_{ID})$ to $\mathbb{A}_{\mathrm{ou}}$. Eventually, $\mathbb{A}_{\mathrm{ou}}$ will output a tuple $(M, ID, \varsigma, \varsigma')$, where $\varsigma = (\sigma, VK, CERT_{ID})$. Assuming $\varsigma \neq \varsigma'$, at least one of the three components must differ. Due to the unicity of signatures in USS, we cannot have that $\varsigma = (\sigma, VK, CERT_{ID})$, $\varsigma' = (\sigma', VK, CERT_{ID})$. Similarly, we cannot have $\varsigma = (\sigma, VK, CERT_{ID})$, $\varsigma' = (\sigma, VK, CERT'_{ID})$, as this would mean $VK \| ID$ has two distinct signatures. It must therefore be the case that there are two different verification keys VK and VK', and that *at most one of them* has been issued by $\mathbb{B}_{\text{euf-cma}}$, meaning she has queried her signing oracle at most one of $VK \| ID$, $VK' \| ID$. Assuming she queried $VK \| ID$, she outputs $(VK' \| ID, CERT'_{ID})$ as the answer to her challenge. It is clear that $\mathbb{B}_{\text{euf-cma}}$ wins with the same probability as $\mathbb{A}_{\mathrm{ou}}$. □

B Anonymous Vetted Encryption (AVE)

B.1 Syntax and Security of AVE

The algorithms. For anonymous vetted encryption, neither the filter nor the recipient should be able to identify who encrypted a message. An AVE scheme consists of five algorithms, as listed in Definition 4 below. We remark on two slightly less obvious definitional choices.

Firstly, the input of the identity ID to Ave.Derive is not really needed, and any decent anonymous system would simply ignore this input. However, for full generality and ease of comparison with IVE and OVE

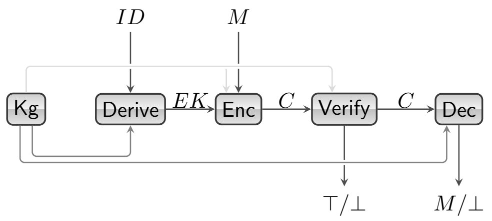

Fig. 19. The algorithms and their inputs/outputs for anonymous vetted encryption.

that do require ID as input in order to derive an encryption key, we allow Ave.Derive to depend on a user's identity ID.

Secondly, we allow encryption to fail, as captured by the $\bot$ output. As we will see, for honestly generated encryption keys, we insist encryption never fails, but for adversarially generated encryption keys, it turns out useful to allow for explicit encryption failure. Of course, one could alternatively introduce a separate algorithm to verify the validity of a private encryption key for a given public encryption/verification key, but our approach appears simpler.

Definition 4 (Anonymous Vetted Encryption (AVE)). *An* anonymous vetted encryption *scheme* AVE *consists of a 5-tuple of algorithms* (Ave.Kg, Ave.Derive, Ave.Enc, Ave.Verify, Ave.Dec) *that satisfy*

- Ave.Kg generates *a key pair* (PK, SK)*, where* PK *is the public encryption (and verification) key and* SK *is the private derivation and decryption key. We allow* Ave.Kg *to depend on parameters* $param$ *and write* $(PK, SK) \leftarrow_\$ \mathsf{Ave.Kg}(param)$*. Henceforth, we will assume that* PK *can be uniquely and efficiently computed given* SK*.*
- Ave.Derive derives *an encryption key* EK *based on the private derivation key* SK *and a user's identity* ID*. We write* $EK \leftarrow_\$ \mathsf{Ave.Derive}_{SK}(ID)$*.*
- Ave.Enc encrypts *a message* M *given the public encryption key* PK *and using the private encryption key* EK*, creating a ciphertext* C *or producing a failed encryption symbol* $\bot$*. In other words,* $C \leftarrow_\$ \mathsf{Ave.Enc}_{PK,EK}(M)$ *where possibly* $C = \bot$*.*
- Ave.Verify verifies *the validity of a ciphertext* C *given the public verification key* PK*, leading to either accept '*$\top$*' or reject '*$\bot$*'. With a slight abuse of notation,* $\top/\bot \leftarrow \mathsf{Ave.Verify}_{PK}(C)$*.*
- Ave.Dec decrypts *a ciphertext* C *using the private key* SK*. The result can either be a message* M *or the invalid-ciphertext symbol* $\bot$*. In short,* $M/\bot \leftarrow \mathsf{Ave.Dec}_{SK}(C)$*.*

The first three algorithms are probabilistic, whereas we assume that the final two algorithms are deterministic.

Correctness and consistency. *Correctness* captures that honest usage results in messages being received as intended. That is, for all parameters $param$, identities ID and messages M, we have that

$$\Pr\left[\begin{array}{l}(PK, SK) \leftarrow_\$ \mathsf{Ave.Kg}(param)\\ \;EK \leftarrow_\$ \mathsf{Ave.Derive}_{SK}(ID)\\ \;\;C \leftarrow_\$ \mathsf{Ave.Enc}_{PK,EK}(M)\end{array} : C \neq \bot \wedge \mathsf{Ave.Verify}_{PK}(C) = \top \wedge \mathsf{Ave.Dec}_{SK}(C) = M\right] = 1$$

As with IVE and OVE, *consistency* ensures that any ciphertext which decrypts to a valid message also passes the filter. We guarantee consistency of an AVE scheme by running verification as part of decryption, which is the same transformation we applied to IVE. We note here as well that correctness ensures that all honestly generated ciphertexts will decrypt to a valid message

Security. The security of AVE comprises three components: integrity to ensure the filter cannot be fooled, confidentiality of the message to outsiders, and finally sender anonymity even from the recipient. With reference to the games defined in Figures 20, 21, and 22, the relevant advantages are defined as follows:

- *Integrity*

$$\mathsf{Adv}^{\mathsf{int}}_{\mathrm{AVE}}(\mathbb{A}) = \Pr\left[\mathsf{Exp}^{\mathsf{int}}_{\mathrm{AVE}}(\mathbb{A}) \,:\, \hat{C} \notin \mathcal{C} \wedge \mathsf{Ave.Verify}_{PK}(\hat{C}) = \top\right].$$

```
$\mathsf{Exp}^{\mathrm{int}}_{\mathrm{AVE}}(\mathbb{A})$
$(PK, SK) \leftarrow\$ \mathsf{Ave.Kg}$
$h \leftarrow 0; \mathcal{C} \leftarrow \emptyset$
$\hat{C} \leftarrow \mathbb{A}^{\mathcal{O}}(PK)$
winif $\hat{C} \notin \mathcal{C} \wedge \mathsf{Ave.Verify}_{PK}(\hat{C}) = \top$

derive($ID$)
$EK[h] \leftarrow \mathsf{Ave.Derive}_{SK}(ID)$
$h \leftarrow h + 1$
return $h$

encrypt($H, M$)
$C \leftarrow\$ \mathsf{Ave.Enc}_{PK, EK[H]}(M)$
$\mathcal{C} \leftarrow \mathcal{C} \cup \{C\}$
return $C$

decrypt($C$)
$M \leftarrow \mathsf{Ave.Dec}_{SK}(C)$
return $M$
```

Fig. 20. The integrity game for AVE.

```
$\mathsf{Exp}^{\mathrm{ind\text{-}cca\text{-}}b^*}_{\mathrm{AVE}}(\mathbb{A})$
$(PK, SK) \leftarrow\$ \mathsf{Ave.Kg}$
$\mathcal{C} \leftarrow \emptyset$
$\hat{b} \leftarrow \mathbb{A}^{\mathcal{O}}(PK)$

derive($ID$)
$EK \leftarrow \mathsf{Ave.Derive}_{SK}(ID)$
return $EK$

encrypt($EK, M_0, M_1$)
$C^* \leftarrow\$ \mathsf{Ave.Enc}_{PK, EK}(M_{b^*})$
$\mathcal{C} \leftarrow \mathcal{C} \cup \{C^*\}$
return $C^*$

decrypt($C$)
require $C \notin \mathcal{C}$
$M \leftarrow \mathsf{Ave.Dec}_{SK}(C)$
return $M$
```

Fig. 21. The confidentiality game for AVE.

– *Confidentiality*

$$\mathsf{Adv}^{\mathrm{conf}}_{\mathrm{AVE}}(\mathbb{A}) = \Pr\left[\mathsf{Exp}^{\mathrm{conf\text{-}0}}_{\mathrm{AVE}}(\mathbb{A}) \,:\, \hat{b} = 0\right] - \Pr\left[\mathsf{Exp}^{\mathrm{conf\text{-}1}}_{\mathrm{AVE}}(\mathbb{A}) \,:\, \hat{b} = 0\right].$$

– *Anonymity*

$$\mathsf{Adv}^{\mathrm{anon}}_{\mathrm{AVE}}(\mathbb{A}) = \Pr\left[\mathsf{Exp}^{\mathrm{anon\text{-}0}}_{\mathrm{AVE}}(\mathbb{A}) \,:\, \hat{b} = 0\right] - \Pr\left[\mathsf{Exp}^{\mathrm{anon\text{-}1}}_{\mathrm{AVE}}(\mathbb{A}) \,:\, \hat{b} = 0\right].$$

Integrity. Integrity may informally be stated as: unless one has been vetted and is in possession of an encryption key, it should not be possible to furnish a valid ciphertext. We capture this integrity security notion in a game (Fig. 20), where the goal of the adversary is to create a valid ciphertext. As with IVE, we use the output of the verification algorithm as the indicator of validity. We note that for consistent AVE schemes, this choice of integrity is the strongest, as a forgery w.r.t. decryption will always be a forgery w.r.t. verification.

The adversary is given the verification key and additionally can ask for encryptions of messages of its choosing. We use the same *handle* mechanism as previously, so the adversary has some control over the encryption keys that are used: an adversary can trigger the game into the creation of an arbitrary number of keys (given an identity) and then indicate which key (by order of creation) to use for a particular encryption query. Obviously, the adversary does *not* receive any encryption keys themselves.

For full generality, we also grant access to a decryption oracle. One could also consider a weaker flavour of integrity without this oracle access. For consistent schemes, the decryption oracle is essentially pointless: if querying it on a fresh ciphertext C were to result in some message (so not $\perp$), then $\mathsf{Ave.Verify}_{PK}(C) = \top$ would already constitute a valid forgery.

Note that if decryption would somehow leak information—say when there are multiple possible decryption failures [25, Remark 14] or unverified plaintext is released early [6]—the decryption oracle would increase an adversary's power. As became evident in the treatment of IVE and OVE, the introduction of identities for the decryption algorithm also renders the corresponding decryption oracle more powerful and relevant.

Confidentiality. In Fig. 21, we adapt the well-trodden IND–CCA notion for public key encryption to the setting of anonymmous vetted encryption. An adversary can (repeatedly) ask its challenge oracle for the encryption of one of two messages. However, our new syntax requires an encryption key in addition to the verification key. We allow the adversary to specify the encryption key to use, which may or may not be honestly generated. In contrast to the integrity game, the key derivation oracle here does provide the adversary with encryption keys for its chosen identities.

Weaker definitions are possible by insisting an adversary can only query the challenge encryption oracle on honest encryption keys (as provided by the derivation oracle), or even hiding said keys using a similar handle-based mechanism as for the integrity game. We believe the stronger notion with adversarially chosen keys is easier to deal with and, as we will see in B.2, still relatively easy to achieve based on standard public key primitives. For IVE and especially OVE, the stronger notion is the more natural one as well.

Anonymity. The idea of anonymity is that a recipient has no clue from which of the vetted people a ciphertext originated, as anonymity towards the recipient implies anonymity towards the filter. Here anonymity extends beyond not being able to extract or link a specific identity ID to a ciphertext: we also want to ensure that ciphertexts created using the same encryption key EK remain unlinkable.

We model our notion of anonymity using a distinguishing game, where the adversary knows the private key SK and gets to choose the encryption keys EK to use. If it cannot tell apart which encryption key EK was used (by the challenge encryption oracle), we deem the scheme anonymous. There is one caveat though: the game is likely winnable by deriving one true encryption key EK_0 (using knowledge of SK) and creating one fake EK_1. Assuming integrity, the challenge ciphertext will verify iff $b^* = 0$. To avoid these trivial wins, we only output a challenge ciphertext if it is valid irrespective of the challenge bit. Our game implements this mechanic by creating possible ciphertexts for both challenge bits and, rather than check based on verification, we put the onus on the encryption itself.

Anonymity is reminiscent of key privacy for public key encryption schemes [11] or its "ciphertext anonymity" adaptation to signcryption (which we will discuss later in B.3).

B.2 Generic Composition: Encrypt-then-Sign

As with IVE, an obvious first attempt to create an anonymous vetted encryption scheme is to combine the confidentiality provided by a public key encryption scheme with the authenticity of a signature scheme. Based on the reasoning as for IVE, we opt for the encrypt-then-sign approach.

The general construction is described in Fig. 23. First, the receiving party generates two key pairs: one for a PKE scheme and one for a signature scheme. It hands out the same signing key to whomever it wants to vet, so any vetted party can use the public encryption key and the received signing key to first encrypt, then sign. Verification by the filter consists of a simple signature verification.

The scheme inherits its authenticity from the signature scheme and its confidentiality from the encryption scheme. The latter inheritance only works when the signature scheme is *unique*. The signature is also verified as part of encryption and decryptions, which at first sight might appear superfluous. However, signature verification at decryption time is required for consistency (in line with the generic transform to achieve consistency), whereas the signature verification at encryption time is required to ensure anonymity even against malicious receivers.

Correctness and consistency. Both correctness and consistency follow easily by inspection. The signature verification as part of decryption is needed for consistency, in line with the transformation from before.

Integrity. Integrity of the scheme follows from the unforgeability of the underlying signature scheme. The proof is by a simple black-box reduction $\mathbb{B}_{\text{euf-cma}}$ where the PKE-ciphertexts in the int-game become messages in the EUF-CMA game. The overhead of $\mathbb{B}_{\text{euf-cma}}$ is running Pke.Kg once, plus one public-key encryption per encryption query posed by $\mathbb{A}_{\text{int}}$. As EtS is consistent, without loss of generality we assume $\mathbb{A}_{\text{int}}$ does not make any decryption queries.

```
$\mathsf{Exp}^{\text{anon-}b^*}_{\text{AVE}}(\mathbb{A})$
$(PK, SK) \leftarrow\!\$\ \mathsf{Ave.Kg}$
$C \leftarrow \emptyset$
$\hat{b} \leftarrow \mathbb{A}^{O}(PK, SK)$

$\mathsf{encrypt}(EK_0, EK_1, M)$
$C_0 \leftarrow\!\$\ \mathsf{Ave.Enc}_{PK,EK_0}(M)$
$C_1 \leftarrow\!\$\ \mathsf{Ave.Enc}_{PK,EK_1}(M)$
if $C_0 \neq \perp \wedge C_1 \neq \perp$ then
    $C^* \leftarrow C_{b^*}$
else
    $C^* \leftarrow \perp$
return $C^*$
```

Fig. 22. The anonymity game for AVE.

Lemma 8 (Integrity of EtS). *For all adversaries $\mathbb{A}_{\text{int}}$, there exists an equallly efficient adversary $\mathbb{B}_{\text{euf-cma}}$ such that*

$$\mathsf{Adv}^{\text{int}}_{\mathrm{AVE}}(\mathbb{A}_{\text{int}}) \leq \mathsf{Adv}^{\text{euf-cma}}_{\mathrm{SIG}}(\mathbb{B}_{\text{euf-cma}}) \, .$$

Proof. Upon receiving a verification key VK, $\mathbb{B}_{\text{euf-cma}}$ generates a key pair $(PK, DK) \leftarrow\!\!\$\ \mathsf{Pke.Kg}$ and runs $\mathbb{A}_{\text{int}}$ on input (PK, VK). Whenever $\mathbb{A}_{\text{int}}$ makes an encryption query, $\mathbb{B}_{\text{euf-cma}}$ performs the public-key encryption to obtain a ciphertext on which it uses its own signing oracle to obtain a signature. When $\mathbb{A}_{\text{int}}$ manages to create a forgery, then it has to create a valid PKE-ciphertext–signature pair that has not been returned by its encryption oracle. From $\mathbb{B}_{\text{euf-cma}}$'s perspective, this means a valid message–signature pair that has not been returned by its signature oracle. The claim follows. □

Confidentiality. Confidentiality of the scheme follows from that of the public key encryption scheme, provided the signature scheme has unique signatures. Unicity of the signature scheme appears necessary. After all, an adversary $\mathbb{A}_{\text{conf}}$ knows the signing key and thus if there are multiple valid signatures, it will be able to generate these and knowing a second signature for a challenge ciphertext would lead to a valid ciphertext to the decryption oracle, learning M_{b^*} and thus b^*, breaking confidentiality. In particular, derandomising a probabilistic signature scheme would be insufficient as $\mathbb{A}_{\text{conf}}$ could simply ignore the derandomisation and generate a signature using fresh randomness, exploiting that verification does not—and usually cannot—check whether the randomness used was constructed deterministically as prescribed by the derandomisation.

With unique signatures in place, the proof is by a simple black-box reduction $\mathbb{B}_{\text{ind-cca}}$, whose overhead is running $\mathsf{Sig.Kg}$ once, plus one signature per challenge encryption query and one signature verification per decryption query posed by $\mathbb{A}_{\text{conf}}$.

Lemma 9 (Confidentiality of EtS). *Let* SIG *be a unique signature scheme. Then for all adversaries $\mathbb{A}_{\text{conf}}$, there exists an equallly efficient adversary $\mathbb{B}_{\text{ind-cca}}$ such that*

$$\mathsf{Adv}^{\text{conf}}_{\mathrm{AVE}}(\mathbb{A}_{\text{conf}}) \leq \mathsf{Adv}^{\text{ind-cca}}_{\mathrm{PKE}}(\mathbb{B}_{\text{ind-cca}}) \, .$$

Proof. Upon receiving a public key PK, $\mathbb{B}_{\text{ind-cca}}$ generates a key pair $(VK, SK) \leftarrow\!\!\$\ \mathsf{Sig.Kg}$ and runs $\mathbb{A}_{\text{conf}}$ on input (PK, VK). Whenever $\mathbb{A}_{\text{conf}}$ makes a challenge encryption query, $\mathbb{B}_{\text{ind-cca}}$ uses its own challenge encryption query to get a PKE-ciphertext, which it subsequently signs using SK. Thus by design, the challenge bits in the PKE and AVE games coincide.

The only potential complication is answering decryption queries (C, σ) by $\mathbb{A}_{\text{conf}}$. If (C, σ) was returned by $\mathbb{B}_{\text{ind-cca}}$ to $\mathbb{A}_{\text{conf}}$ as a previous challenge ciphertext, then the query may be ignored. So let us assume that (C, σ) is fresh. Then $\mathbb{B}_{\text{ind-cca}}$ first verifies whether σ is a valid signature on C. If not, return $\perp$, otherwise there are two possibilities: either C itself if fresh, i.e. it has not been returned by the game as a challenge ciphertext to $\mathbb{B}_{\text{ind-cca}}$, or it is not fresh, meaning it *is* a challenge ciphertext. In the former case, $\mathbb{B}_{\text{ind-cca}}$ can forward C to its own decryption oracle, receive a message M as result, and forward M to $\mathbb{A}_{\text{conf}}$. In the latter case, $\mathbb{B}_{\text{ind-cca}}$ cannot realistically forward C to its own decryption oracle (as it would be rejected). Luckily, the latter case cannot actually occur due to the unicity of signatures. □

Anonymity. Intuitively, anonymity follows from all encryption keys being the same, so independent of any identity or any additional randomness. However, in our security model an adversary is allowed to provide

```
$\mathsf{Ave.Kg}()$
$(PK, DK) \leftarrow\!\!\$\ \mathsf{Pke.Kg}$
$(VK, SK) \leftarrow\!\!\$\ \mathsf{Uss.Kg}$
return $((PK, VK), (DK, SK))$

$\mathsf{Ave.Derive}_{(DK,SK)}(ID)$
return $SK$

$\mathsf{Ave.Enc}_{(PK,VK),SK}(M)$
$C \leftarrow\!\!\$\ \mathsf{Pke.Enc}_{PK}(M)$
$\sigma \leftarrow \mathsf{Uss.Sign}_{SK}(C)$
if $\mathsf{Uss.Verify}_{VK}(C, \sigma) = \perp$ then
    return $\perp$
return $(C, \sigma)$

$\mathsf{Ave.Verify}_{PK,VK}(C, \sigma)$
return $\mathsf{Uss.Verify}_{VK}(C, \sigma)$

$\mathsf{Ave.Dec}_{DK,SK}(C, \sigma)$
if $\mathsf{Uss.Verify}_{VK}(C, \sigma) = \perp$ then
    return $\perp$
return $\mathsf{Pke.Dec}_{DK}(C)$
```

Fig. 23. Encrypt-then-Sign (EtS): A straightforward composition of public key encryption and signature scheme.

the encryption keys and thus deviate from honestly generated ones. Luckily, the unique-signatures property coupled with signature verification as part of the encryption routine, ensures an adversary can gain no benefit from such deviations, allowing us to show that anonymity of the scheme holds unconditionally.

Lemma 10 (Anonymity of EtS). *Let* SIG *be a unique signature scheme. Then for all adversaries* $\mathbb{A}$*,*

$$\mathsf{Adv}^{\mathrm{anon}}_{\mathrm{AVE}}(\mathbb{A}) = 0 \ .$$

Proof. The standard anonymity game for AVE allows multiple queries, but by a straightforward hybrid argument we can consider a single query only; as we target advantage 0 anyway, this hybrid will not incur a tightness loss. So consider $\mathbb{A}$'s single query SK_0, SK_1, M. The first part of EtS encryption calculates $C \leftarrow \mathsf{Pke.Enc}_{PK}(M)$. The resulting random variable C is clearly indepedent of the challenge bit. Next, a signature on C is produced, using either the signing key SK_0 or SK_1. If either of the signatures produced is invalid, the verification step as part of the encryption will notice and the game ensures the challenge oracle will output $\perp$ (making distinguishing impossible). Thus assume that both signatures pass the verification step. Then unicity of the signature scheme implies the signatures are in fact the same, thus the output of the challenge encryption oracle is independent of the challenge bit b^*: even an information theoretic adversary $\mathbb{A}$ cannot do better than random guessing. □

Instantiations. There is an abundance of efficient IND-CCA-secure PKE schemes available, based on a wide variety of cryptographic hardness assumptions. Unique signatures are rarer, especially in the standard model [41, 30]. In the random oracle model, an obvious candidate would be RSA-FDH [15, 23].

Remark 1. In practice a sender could of course "precompute" the signature verification by checking whether the signing key received as part of of the EK-derivation routine is valid for the public verification key. Such a precomputation is not entirely without loss of generality as it requires a signing-key checking algorithm that cannot be fooled (namely that once a signing key is accepted, acceptance of the resulting signatures is guaranteed for *all* messages).

B.3 Alternative Approaches

Signcryption. In many ways, AVE is reminiscent of signcryption, thus a natural question is whether one can turn a signcryption scheme into an AVE scheme. In order to answer this question, we need to zoom in on the right kind of signcryption scheme: which functionality does it need to support and which security does it need to provide?

From a functional perspective, the main restriction is the need for public verifiability [9, 32] as for AVE the filter needs to be able to verify ciphertexts without access to private key material. Thus, we consider a signcryption scheme to consist of six algorithms (Scr.Kgr, Scr.Kgs, Scr.Signcrypt, Scr.Verify, Scr.Unsigncrypt), where Scr.Kgr generates the receiver's keys and Scr.Kgs the sender's keys. We can transform such a signcryption scheme into an AVE scheme by simply letting Ave.Kg run both $(PK, DK) \leftarrow_{\$} \mathsf{Scr.Kgr}$ and $(VK, SK) \leftarrow_{\$} \mathsf{Scr.Kgs}$, and setting (PK, VK) to the public key of the AVE, keeping (DK, SK) private. The sender private key SK will serve as the encryption key and is therefore returned by Ave.Derive (irrespective of the identity). For the final three algorithms, there is a clean correspondence:

- $\mathsf{Ave.Enc}_{(PK,VK),SK}(M) = \mathsf{Scr.Signcrypt}_{(PK,VK),SK}(M)$;
- $\mathsf{Ave.Verify}_{PK,VK}(C) = \mathsf{Scr.Verify}_{VK}(C)$;
- $\mathsf{Ave.Dec}_{DK,SK}(C) = \mathsf{Scr.Unsigncrypt}_{DK}(C)$.

For the resulting AVE scheme, *correctness* is directly inherited from that of the signcryption scheme and *consistency* is satisfied provided the signcryption satisfies a similar notion (between verification and unsigncryption). For the security notions, our AVE setting only has two users, which implies the two-user model for signcryption suffices [5, Section 2.2]. In that case, *integrity* is a consequence of strong outsider secure unforgeability under chosen message attacks [5, Section 2.2.1.2]. Here outside security suffices as, in the AVE integrity game, an adversary does not have access to the derived encryption keys (which would correspond to a sender's private signcryption key). In contrast, for confidentiality we do need *insider* secure indistinguishability under chosen ciphertext attacks, as the sender's private signcryption key will be readily available to an adversary in the corresponding AVE confidentiality game (through the derive oracle).

```
$\mathsf{Exp}^{\text{anon-}b^*}_{\mathrm{SCR}}(\mathbb{A})$
$(PK_r, SK_r) \leftarrow\!\$\ \mathsf{Scr.Kgr}$
$C \leftarrow \emptyset$
$\hat{b} \leftarrow \mathbb{A}^{O}(PK_r, SK_r)$
```

```
$\mathrm{signcrypt}(SK_{s0}, SK_{s1}, M)$
$C_0 \leftarrow\!\$\ \mathsf{Scr.Signcrypt}_{PK, SK_{s0}}(M)$
$C_1 \leftarrow\!\$\ \mathsf{Scr.Signcrypt}_{PK, SK_{s1}}(M)$
if $C_0 \neq \perp \wedge C_1 \neq \perp$ then
    $C^* \leftarrow C_{b^*}$
else
    $C^* \leftarrow \perp$
return $C^*$
```

Fig. 24. The anonymity game needed when constructing AVE from signcryption.

Finally, *anonymity* is hardest to place, so let's look at anonymity notions for signcryption. The original ciphertext anonymity [20] captures only indistinguishability for honestly generated keys; moreover it attempts to hide both sender and receiver (the latter is irrelevant for us). Later incarnations of ciphertext anonymity [39, 40], do consider adversarially generated sender-keys. However, the syntax does not explicitly allow signcryption failure (even though signcryption can fail for some constructions). Moreover, the corresponding security game does not seem to care if challenge signcryption fails for only one of the two "left-or-right" adversarially provided sender signcryption keys (cf. [10, Section 5.6.2]), as we do in our anonymity game. It is relatively straighforward to derive the matching anonymity game for signcryption needed for the signcryption-to-AVE transform to work (see Fig. 24).

As an aside, although encrypt-then-sign has been studied in the signcryption literature, we are not aware of the potential of using unique signatures in order to achieve insider IND-CCA security (cf. [5, Theorem 2.2]).

Hybrid encryption. The typical operations associated with public key primitives are typically considerably more expensive than their symmetric counterparts. Hybrid encryption allows one to leverage the speed of symmetric cryptography, while maintaining the functionality and security of public key cryptography. A natural question is how applicable the concepts of hybrid encryption are for AVE.

Obviously, in the EtS transform it is possible to use a hybrid PKE. A natural question is whether our transform could then deal with distinct decryption failures from the KEM, resp. the DEM [25, Remark 14]. As we mentioned, for the integrity game, the decryption oracle might come into play, but for the EtS construction they do not cause any trouble (the reduction knows the PKE decryption key and the signing key isn't use by the AVE decryption). For the other two security properties, the proofs go through as is.

Potentially even more relevant and potent is the idea of hybrid signcryption [34, 17]. Here the signcryption KEM takes as input the receiver's public signcryption key and the sender's private signcryption key, returning a signcryptext as well as an ephemeral key for the (standard) DEM. The problem of the resulting construction is that it does not provide insider security and it is easy to see how the resulting AVE fails to provide proper integrity: after observing a valid pair (C, C') where C is the signcryptext and C' the DEM-ciphertext, simply substitute the second component. An alternative is the use of signcryption tag-KEMs [18], which do provide insider security. The signcryption scheme with public verifiability [32] can be cast this way and thus would be a good candidate for AVE (not entirely surprising given the original design goal).

Graphic design: Communication Division, UIB / Print: Skipnes Kommunikasjon AS

uib.no

ISBN: 9788230855997 (print)
9788230859438 (PDF)