Department of Computer Science and Applied Mathematics
Faculty of Mathematics and Computer Science
Weizmann Institute of Science

# Cryptographic Methods for the Clouds

Thesis Submitted for the Degree of

“Doctor of Philosophy”

by

**Zvika Brakerski**

Under the Supervision of Professor Shafi Goldwasser

Submitted to the Scientific Council of
the Weizmann Institute of Science
Rehovot, Israel
July 2011

## Abstract

The recent transition to “cloud computing” is changing the way people store and use digital data. Data is more often stored remotely (on a cloud server) and accessed by very weak devices over the internet (e.g. smartphones). In addition to storage, the cloud is expected to manipulate and process the data per the user’s requests, since local processing on the end user’s device is infeasible. This allows for great flexibility in the use of data, but also means new challenges for data privacy.

To answer these challenges, we need to add new tools to our cryptographic toolbox. We want to store our data in the cloud in an encrypted form, for privacy purposes, but still want the cloud to process the data for us. We need a fast local decryption solution that can be executed securely in the presence of side channel attacks. We need to store cryptographic keys in a secure manner to utilize cloud services securely, which necessitates the ability to encrypt the cryptographic keys themselves.

In this thesis, we present three results that push the state of the art for public-key encryption in the face of cloud challenges.

1. We present a fully homomorphic encryption scheme that is based on worst-case short-vector problems in arbitrary lattices. Fully homomorphic encryption allows to process encrypted data in an arbitrary way without decrypting it, and hence is tremendously useful for outsourcing computation.

   Our construction is simpler than previous candidates and does not require the infamous “squashing” phase. Thus it could be a step towards a practical fully homomorphic scheme.

2. We present a public key encryption scheme that is secure against continual leakage of information on the secret key. That is, we show how to periodically update the secret key (without any change to the public key) so that even if information constantly leaks to an adversary, our encryptions stay secure.

3. We construct a circular secure public-key encryption scheme based on factoring-related assumptions. A circular secure scheme is one that can securely encrypt its own secret key (or have a number of secret keys encrypting each other).

   We present a unified approach for proving a number of interesting properties: circular security, bounded leakage resilience and auxiliary-input security, in a generic way that applies to a number of cryptographic assumptions.

**Acknowledgements**

I was very fortunate to witness a great leap in cryptographic research during the time of my studies. Even more so, to be given the opportunity to interact with the great people who made this leap happen — I really feel that I learned from the best.

First and foremost, my advisor, Shafi Goldwasser, who exposed me to new ideas and challenges from day one. Thank you so much for countless discussions and priceless words of advice, and for helping me stay on track despite my tendency to stray. It was with great awe that I first approached Shafi, asking her to advise me, and quite honestly, I still feel the same way.

I am extremely thankful to the incredible cryptography group at the Weizmann Institute: Shafi, Oded Goldreich, Moni Naor, Omer Reingold and Adi Shamir, together with an incredible group of post-docs and students. I especially thank Moni and Oded for serving on my Ph.D. supervising committee and to Adi for chairing the Ph.D. approval committee.

This thesis is based on works co-authored by a great group of people: Shafi, Vinod Vaikuntanathan, Jonathan Katz and Yael Tauman Kalai. I am so very grateful to each and every one of you. I also thank my collaborators in works not included in this thesis: Mihir Bellare, Oded Goldreich, Moni Naor, Boaz Patt-Shamir, Thomas Ristenpart, Guy Rothblum, Gil Segev, Hovav Shacham, Arkady Yerukhimovich and Scott Yilek.

Special thanks to Yael, an "academic big sister" if there ever was one, and my two-times internship supervisor. Working with her shifted gears in my research, and her views on life and research taught me a lot. In addition, to my co-authors and friends, Vinod and Gil. Working with you guys has been a pleasure, let alone drinking beer and hanging out. Thanks for making my work fun. I also thank my Master's advisor, Boaz Patt-Shamir, with whom I always enjoy talking.

So many other people influenced me throughout my studies, no list that I can make will be exhaustive. Let me therefore express my gratitude to all of you collectively: My hosts on various visits, office mates, teachers, speakers and partners in various conversations.

I want to thank the computer science and artificial intelligence lab (CSAIL) at MIT for repeatedly hosting me during my studies, I learned a lot from this great group of researchers and students. I wish to thank the very impressive Microsoft Research – New-England lab for the two internships I spent with them. Thanks for the great working environment and for making me feel at home.

Lastly, to Elette, who made all the difference.

# Contents

# Chapter 1

# Introduction

“*We stand today on the brink of a revolution in cryptography.*”

– “New Directions in Cryptography”, W. Diffie and M. E. Hellman, 1976.

This quotation opens the seminal paper of Diffie and Hellman [DH76], which marks the birth of public-key cryptography. The projected revolution was driven by the increasing availability of computers and computer networks, which necessitated solutions for secure storage and transmission of digital data.

Today, we are fully realizing this revolution, as the size and weight of computers decreased to the point that a computer with internet connectivity can be found virtually in any pocket. We want our data to be accessible from anywhere, so we store it “in the cloud”, on remote servers; To increase battery life and reduce the size of our portable devices, we build them with the weakest possible processors, and outsource the computation itself to the cloud. We now hold in our hands an extremely weak device, which nonetheless gives us access to limitless storage and computation resources.

The opportunities and flexibility are great, but so are the risks. To ensure our privacy, we would like to perform all cloud operations in an encrypted way: Encrypt our data before we send it to the cloud (so the cloud itself doesn’t know the content), and then have the cloud compute on this encrypted data. If we could do that, then all our portable device needs is sufficient computational power to encrypt and decrypt the data sent to and received from the cloud, and a way to securely store cryptographic keys. To implement this plan, we need the following tools:

1. An encryption scheme that can perform any desirable operation on encrypted data (clearly, an arbitrary encryption scheme will not allow processing encrypted data without decrypting).

2. A way to decrypt securely on local devices in spite of hacking attacks.

3. A way to store cryptographic keys in an encrypted form (surprisingly, this does not follow trivially from secure encryption).

Indeed, in the last few years, there has been a flux of research effort on these matters, achieving a number of breakthroughs.

1. **Homomorphic Encryption.** A fully homomorphic encryption scheme is one where encrypted data can be processed without decrypting. Thus we can get the cloud to perform a computation for us, while revealing nothing of the input (or output).

   The question of whether fully homomorphic encryption exists was put forth by Rivest Adleman and Dertouzos [RAD78] back in 1978, and remained open for 30 years (with some partial progress), until the first candidate scheme was presented by Gentry late in 2008 [Gen09b, Gen09a] in one of the major breakthroughs in cryptography of recent years.

   Gentry's solution showed us that fully homomorphic encryption can, in principle, be constructed. However, his solution involves new and relatively untested cryptographic assumptions: quantum hardness of short-vector problems in ideal lattices, and the hardness of the sparse subset-sum problem. In addition, it is quite complex and far from usable in practice. A small number of follow-up works [SV10, vDGHV10, BV11a] used slightly different assumptions to achieve similar results.

   The question of finding simpler, more efficient constructions, and most importantly, ones that are based on more traditional and well studied hardness assumptions, remained open.

2. **Key Leakage Resilience.** The basic notion of *semantic security* for public key encryption, presented by Goldwasser and Micali [GM82, GM84], guarantees that an attacker that only knows the public key, cannot find two messages that he can distinguish between their encryptions. This is equivalent to not being able to learn anything on the message given a ciphertext — exactly what we require. However, it gives no guarantee for the case where an attacker gets a hold of some information about the secret key as well (e.g. by hacking the device where the key is stored). Indeed, for many schemes, even releasing a small amount of information about the secret key makes the scheme insecure. This motivated a number of attacks such as "side channel attacks" [Koc96, KJJ99] or the "cold-boot attack" [HSH$^+$08], that take advantage of the physical implementation of the scheme to gather information about the secret key and breach security.

   An important step in this direction was made, late in 2008, by Akavia, Goldwasser and Vaikuntanathan [AGV09]. They showed a specific scheme that (provably) remains secure even if the attacker can obtain a bounded amount of (arbitrary) information about the secret key. This notion of security was termed the *bounded memory leakage model.* Naturally, this opened the door to many questions. Is it possible to achieve such leakage resilience for other schemes?How much of the secret key can be leaked while keeping the scheme secure? Naor and Segev [NS09] gave partial answers by extending the set of assumptions for which bounded leakage resilience could be achieved.

   More importantly, it seemed that the bounded memory leakage model does not capture the full extent of side channel attacks: A hacker might keep extracting information from the device until it acquires the entire secret key, in which case security cannot hold. To protect against such *continual leakage*, one would need to keep updating the secret key, to prevent the attacker from obtaining all of it. Whether this can be done without changing the public key (which at the time of the attack has already been published) remained an intriguing open problem.

3. **Circular Security.** Using a variety of cloud services naturally requires managing a large number of cryptographic keys. It should come as no surprise, therefore, that sometimes the secret keys themselves are encrypted. This turns out to open the door for a new security risk. Consider a case where we use two cloud services, and we would like to store the password for cloud A in cloud B and vice versa. It turns out that such *key cycles* are not guaranteed to be secure by standard security notions (this was already noticed by Goldwasser and Micali when they defined semantic security). Black, Rogaway and Shrimpton [BRS02] formally defined this notion of security, coining the term *key-dependent message* (KDM) security (though the name *circular security* is also widely used).

   KDM-security also comes up in other (seemingly unrelated) areas. For example, in axiomatic security, and interestingly also in the context of fully homomorphic encryption. The security of all known candidate fully homomorphic encryption schemes relies on the circular security of some derived scheme.

   The first scheme that was provably circular secure (without random oracles) was presented in 2008, more than 25 years after the problem was raised, by Boneh, Halevi, Hamburg and Ostrovsky [BHHO08], followed by a work of Applebaum, Cash, Peikert and Sahai [ACPS09]. Their schemes allowed to encrypt any linear function of the secret key. Interestingly, these works utilized similar properties to those used for showing (bounded) memory leakage resilience.

   Several intriguing open questions remained: Does the similarity in technique imply a connection between circular security and leakage resilience? Can circular security be achieved under additional cryptographic assumptions? Specifically assumptions that are related to the problem of factoring large numbers. In addition, can circular security be achieved with respect to bigger classes of functions?

   We remark that interestingly enough, we do not know of any attack that utilizes encryptions of the secret key (putting aside contrived examples). Thus, for all we know, it is possible that all schemes (under some restriction) are circular secure!

In this thesis, we present a number of novel results, using a variety of new techniques, that push forward the state of the art in the field, thus advancing the new cryptographic transition.

1. We present the first fully homomorphic encryption scheme that is based on worst-case short-vector problems in arbitrary lattices (via the learning with errors assumption). Our construction is simpler than previous candidates and does not require the hardness of sparse subset sum or the infamous "squashing" phase. Thus it could be a step towards a practical fully homomorphic scheme. See Section 1.1 for more details.

2. We present the first public key encryption scheme that is secure against continual leakage. To this end, we present new linear-algebraic techniques for leakage resilience. See Section 1.2.

3. We present the first construction of a circular secure and bounded memory leakage resilient public-key encryption scheme based on factoring related assumptions (specifically: quadratic residuosity and decisional composite residuosity). This is done using a new and generic proof technique that applies to both properties and can be instantiated from a number of hardness assumptions. See Section 1.3.

## 1.1 Fully Homomorphic Encryption from LWE

We present a fully homomorphic encryption scheme based on the learning with errors ($\mathsf{LWE}$) assumption. Using known reductions, the security of our scheme can be shown to rely on the hardness of short vector problems in worst case (arbitrary) lattices. We further use our scheme as a building block for a near-optimal $\mathsf{LWE}$-based private information retrieval (PIR) protocol.

Recall that fully homomorphic encryption, which allows evaluating arbitrarily complex programs on encrypted data, was contemplated in 1978 but a first plausible candidate was only presented by Gentry 30 years later (although, there has been partial progress in the meanwhile [GM82, Pai99, BGN05, IP07]).

Gentry's solution showed us that fully homomorphic encryption can in principle be constructed (where many believed it might be straight out impossible!). His solution, however, involved new and relatively untested cryptographic assumptions: the quantum hardness of short-vector problems in *ideal lattices* (roughly, ideal lattices correspond to a geometric embedding of an ideal in a number field) in conjunction with the hardness of the sparse subset sum problem. This made his construction difficult to evaluate (as one might be leery of untested assumptions at start). Our work aims to put fully homomorphic encryption on a firm theoretical footing, by basing it on standard, well-studied cryptographic assumptions.

Gentry's construction had two main building blocks: A so-called "somewhat" homomorphic encryption scheme, which can only evaluate low-complexity functions, was constructed based on the hardness of (quantumly) finding short-vectors in ideal lattices; and an additional "squashing" phase that simplified the decryption circuit of the aforementioned somewhat homomorphic scheme.[1] This was done, roughly, by giving the encryptor an obfuscated version of the secret key, and having him do a part of the decryption himself. To argue that the obfuscated secret key can be released without breaking the security of the scheme, Gentry needed to make an additional strong hardness assumption, namely the hardness of the sparse subset-sum problem.

Subsequent works [SV10, vDGHV10, BV11a] followed the same outline, implementing the somewhat homomorphic scheme under slightly different assumptions, and then squashing.

Our construction breaks from this paradigm. We present a somewhat homomorphic scheme whose decryption complexity is low "right out of the box", without squashing. The security of our scheme relies on the hardness of worst-case short vector-problems in arbitrary lattices: Either on the classical hardness of the $\mathsf{gapSVP}_{\gamma,\zeta}$ problem (using the reduction in [Pei09]) or on the quantum hardness of the $\mathsf{SIVP}$ or $\mathsf{gapSVP}$ problems (using the reduction in [Reg05]), all with sub-exponential approximation factor. This is achieved by relying on the learning with errors ($\mathsf{LWE}$) assumption and applying known reductions.

Our construction is based on two new techniques:

1. *Re-linearization* that is used to obtain an $\mathsf{LWE}$-based somewhat homomorphic encryption (without hardness assumptions on ideals).

[1]Gentry's bootstrapping theorem showed that somewhat homomorphic scheme that can evaluate its own decryption circuit can be made "leveled" fully homomorphic (the parameters of a leveled scheme depend on the depth of the evaluated function). A third component to the construction, which is less relevant to this discussion, is a circular security assumption which is needed to make the scheme depth independent. Such an assumption is required in order to achieve the stronger variant in all known constructions, including ours.

2. *Dimension-modulus reduction*, that turns our somewhat homomorphic scheme into a fully homomorphic one, without squashing and without additional assumptions.

We proceed to construct (single-server) private information retrieval (PIR) protocols, where a client wishes to retrieve a single entry from a very large (e.g. exponential size) database maintained by a powerful server, without the server learning anything about the entry being indexed. This problem was introduced in an information theoretic multi-server setting by Chor, Goldreich, Kushilevitz and Sudan [CGKS95, CKGS98] and in the cryptographic single server setting by Kushilevitz and Ostrovsky [KO97]. A first efficient construction (with polylogarithmic complexity) was presented by Cachin, Micali and Stadler [CMS99].

The main quality measure for such a protocol is its communication complexity, namely the number of bits being exchanged between the parties until the client is satisfied. We show how to use our homomorphic scheme, in conjunction with other techniques, to obtain a scheme whose communication complexity, for database size $N$ and security parameter $k$, is $\log N + k \cdot \text{polylog}(k)$ in the public-key model (where each client is associated with a publicly known key that is generated ahead of time independently of the database or the query). This is near optimal since $\log N$ is an obvious information theoretic lower bound. The previous best known [CMS99, Lip05, GR05] was $O(\log^{3-o(1)} N)$ for $k = O(\log N)$ (with a delicate interplay between the security parameter and the database size).

## 1.2 Continual Memory Leakage Resilience

We construct the first public-key encryption scheme that is secure against continual memory leakage (namely, against an attacker that continuously steals secret information from our device). We show how to periodically update the secret key, leaving the public key unaffected, such that the scheme remains secure even if a $(1 - o(1))$ fraction of the secret key is leaked between any two consecutive updates. Prior to our work, it was not known whether this is achievable even in weaker adversarial models.

We present a model for continual memory leakage (CML) which generalizes the previous "only computation leaks information" model of Micali and Reyzin [MR04] and the "bounded memory leakage" model of Akavia, Goldwasser and Vaikuntanathan [AGV09]. The secret key in our model is updated over time and the total amount of leakage, over the lifetime of the system, is unbounded. The leakage between key updates can be an arbitrary function of the secret key, subject only to a restriction on the total length of the leakage. We believe that this more accurately models new attacks such as side channel and cold boot attacks.

Our continual memory leakage resilient scheme is based on the $d$-linear assumption in bilinear groups (for any $d \geq 1$). The allowed leakage between consecutive updates, i.e. the amount of information the attacker can obtain in that period without breaking the scheme, is up to a $(1/d - o(1))$ fraction of the secret key length. Hence, if we rely on the SXDH (1-linear) assumption, we obtain a scheme that can allow almost the entire key to leak before updating.

A principle component in our proof is a linear algebraic theorem that states (roughly) that "random linear subspaces are resilient to continual memory leakage". This theorem has already found applications in follow-up works. Our secret keys are essentially (obfuscated) vectors in a linear subspace, and an update operation is a random rotation of the secret key. We show that to an adversary, this rotation is computationally indistinguishable from sampling random vectors from the subspace which, in turn, statistically hides the identity of the subspace (using the aforementioned theorem).

## 1.3 Circular Security and Leakage Resilience from Subgroup Indistinguishability

We present a unified approach for achieving public-key encryption with key-dependent message security (circular security), as well as high resilience to secret key leakage and high resilience to the presence of auxiliary input information.[2]. We instantiate this approach under the quadratic residuosity (QR) assumption and more generally, under a class of assumptions that includes QR as well as Paillier's decisional composite residuosity (DCR) assumption.

In particular, under what we call the *subgroup indistinguishability assumption*, of which QR and DCR are special cases, we can construct a scheme that has:

- *Key-dependent message (circular) security.* Achieves security even when encrypting affine functions of its own secret key (in fact, w.r.t. affine "key-cycles" of predefined length). This is leveraged to security w.r.t. broader classes of functions of the key using the techniques of [BGK11, BHHI10, App11].
- *Leakage resilience.* Remains secure even if any adversarial low-entropy (efficiently computable) function of the secret key is given to the adversary. A proper selection of parameters allows for a "leakage rate" of $(1 - o(1))$ of the length of the secret key.
- *Auxiliary-input security.* Remains secure even if any sufficiently *hard to invert* (efficiently computable) function of the secret key is given to the adversary.

Our scheme is the first to achieve key-dependent security and auxiliary-input security based on the DCR and QR assumptions. Previous schemes that achieved these properties relied either on the DDH [BHHO08, DGK$^+$10] or LWE [ACPS09, DGK$^+$10] assumptions. The proposed scheme is also the first to achieve leakage resiliency for leakage rate $(1 - o(1))$ of the secret key length, under the QR assumption. This was again known before under the LWE (extension of [AGV09]) or DDH [NS09] assumptions.

To achieve the above, we notice that LWE and DDH share the common feature that, very roughly, a random linear combination of elements (over some group and possibly with added noise) is computationally indistinguishable from uniform. This feature, seemed to play a central role in proving all above properties, cannot be obtained from QR.

We show that in fact this feature is not necessary. We present a proof technique that we call "adaptive vector game", and show that it can be used to prove all above properties. We show how to achieve a secure adaptive vector game under QR,[3] and then how to use it to construct a scheme that has the desired properties.

[2]Auxiliary-input security is a generalization of leakage resilience to the case where the bound on the amount of leakage is computational rather than information theoretic. We do not elaborate further in this high level introduction.

[3]We remark that adaptive vector games under LWE and DDH are fairly easy to achieve using the common feature we describe above.

Our results generalize to a class of assumptions that share common algebraic structure with QR. This specifically includes Paillier's DCR assumption. We formally define this class of assumptions and present the generalized version of our QR construction.

## 1.4 Sources and Organization of This Thesis

The material for the three main parts of the thesis is taken from the following papers, respectively.

1. Z. Brakerski and V. Vaikuntanathan. Efficient Fully Homomorphic Encryption from (Standard) LWE. In 52nd Annual IEEE Symposium on Foundations of Computer Science — FOCS 2011.

2. Z. Brakerski, Y. T. Kalai, J. Katz and V. Vaikuntanathan. Overcoming the Hole in the Bucket: Public-Key Cryptography Resilient to Continual Memory Leakage. In 51st Annual IEEE Symposium on Foundations of Computer Science — FOCS 2010.

3. Z. Brakerski and S. Goldwasser. Circular and Leakage Resilient Public-Key Encryption Under Subgroup Indistinguishability (or: Quadratic Residuosity Strikes Back). In Advances in Cryptology, 30th International Cryptology Conference — CRYPTO 2010.

Chapters 2, 3, 4 are dedicated to these contributions, respectively. In Chapter 5 we conclude and discuss open problems in the field. The appendix overviews additional works that were not included in this thesis (see below).

## 1.5 Works Not Included in This Thesis

In the appendix, we briefly overview additional projects that were not included in the thesis:

1. Z. Brakerski and Y. T. Kalai. A Parallel Repetition Theorem for Leakage Resilience. Manuscript, 2011.

2. Z. Brakerski and V. Vaikuntanathan. Fully Homomorphic Encryption from Ring-LWE and Security for Key Dependent Messages. In Advances in Cryptology, 31th International Cryptology Conference — CRYPTO 2011.

3. Z. Brakerski and G. Segev. Better Security for Deterministic Public-Key Encryption: The Auxiliary-Input Setting. In Advances in Cryptology, 31th International Cryptology Conference — CRYPTO 2011.

4. Z. Brakerski, J. Katz, G. Segev and A. Yerukhimovich. Limits on the Power of Zero-Knowledge Proofs in Cryptographic Constructions. In Theory of Cryptography, 8th Theory of Cryptography Conference — TCC 2011.

5. Z. Brakerski, S. Goldwasser and Y. T. Kalai. Black-Box Circular-Secure Encryption Beyond Affine Functions. In Theory of Cryptography, 8th Theory of Cryptography Conference — TCC 2011.

6. Z. Brakerski and Y. T. Kalai. A Framework for Efficient Signatures, Ring Signatures and Identity Based Encryption in the Standard Model. Manuscript, 2010.
7. M. Bellare, Z. Brakerski, M. Naor, T. Ristenpart, G. Segev, H. Shacham and S. Yilek. Hedged Public-Key Encryption: How to Protect Against Bad Randomness. In Advances in Cryptology, 15th International Conference on the Theory and Application of Cryptology and Information Security — ASIACRYPT 2009.
8. Z. Brakerski and B. Patt-Shamir. Distributed Discovery of Large Near-Cliques. In Distributed Computing, 23rd International Symposium — DISC 2009 (brief announcement in PODC 2009).
9. Z. Brakerski and O. Goldreich. From Absolute Distinguishability to Positive Distinguishability. In Studies in Complexity and Cryptography, 2011.
10. Z. Brakerski, S. Goldwasser, G. N. Rothblum, V. Vaikuntanathan. Weak Verifiable Random Functions. In Theory of Cryptography, 6th Theory of Cryptography Conference — TCC 2009.

# Chapter 2

# Fully Homomorphic Encryption from LWE

Fully-homomorphic encryption is one of the holy grails of modern cryptography. In a nutshell, a fully homomorphic encryption scheme is an encryption scheme that allows evaluation of arbitrarily complex programs on encrypted data. The problem was suggested by Rivest, Adleman and Dertouzos [RAD78] back in 1978, yet the first plausible candidate came thirty years later with Gentry's breakthrough work [Gen09b, Gen10] (although, there has been partial progress in the meanwhile [GM82, Pai99, BGN05, IP07]).

Gentry's work showed for the first time that fully homomorphic encryption can be based on cryptographic assumptions. However, his solution involved new and relatively untested such assumptions. Our work aims to base fully homomorphic encryption on standard, well-studied cryptographic assumptions.

The main building block in Gentry's construction (a so-called "somewhat" homomorphic encryption scheme) was based on the (worst-case, quantum) hardness of problems on *ideal lattices.* [1] Although lattices have become standard fare in cryptography and lattice problems have been fairly well-studied, ideal lattices are a special breed that we know relatively little about. Ideals are a natural mathematical object to use to build fully homomorphic encryption in that they natively support both addition and multiplication (whereas lattices are closed under addition only). Indeed, all subsequent constructions of fully homomorphic encryption [SV10, vDGHV10, BV11a] relied on ideals in various rings in an explicit way. Our first contribution is the construction of a "somewhat" homomorphic encryption scheme whose security relies solely on the (worst-case, classical) hardness of standard problems on *arbitrary* (not necessarily ideal) *lattices*.

Secondly, in order to achieve *full* homomorphism, Gentry had to go through a so-called "squashing step" which forced him to make an additional strong hardness assumption – namely, the hardness of the (average-case) sparse subset-sum problem. As if by a strange law of nature, all the subsequent solutions encountered the same difficulty as Gentry did in going from a "somewhat" to a fully homomorphic encryption, and they all countered this difficulty by relying on the same sparse subset-sum assumption. This additional assumption was considered to be the main caveat

[1] Roughly speaking, ideal lattices correspond to a geometric embedding of an ideal in a number field. See [LPR10] for a precise definition.

of Gentry's solution and removing it has been, perhaps, the main open problem in the design of fully homomorphic encryption schemes. Our second contribution is to remove the necessity of this additional assumption.

Thus, in a nutshell, we construct a fully homomorphic encryption scheme whose security is based solely on the classical hardness of solving standard lattice problems in the worst-case.[2] Specifically, out scheme is based on the learning with errors (LWE) assumption that is known to be at least as hard as solving hard problems in general lattices. Thus our solution does not rely on lattices directly and is fairly natural to understand and implement.

To achieve our goals, we deviate from two paradigms that ruled the design of (a handful of) candidate fully homomorphic encryption schemes [Gen09b, SV10, vDGHV10, BV11a]:

1. We introduce the *re-linearization* technique, and show how to use it to obtain a *somewhat* homomorphic encryption that does not require hardness assumptions on *ideals*.

2. We present a *dimension-modulus reduction* technique, that turns our somewhat homomorphic scheme into a fully homomorphic one, without the need for the artificial *squashing* step and the sparse subset-sum assumption.

We provide a detailed overview of these new techniques in Section 2.1.1, 2.1.2 below.

Interestingly, the ciphertexts of the resulting fully homomorphic scheme are very short. This is a desirable property which we use, in conjunction with other techniques, to achieve very efficient private information retrieval protocols (see Section 2.1.3 below for overview).

**Chapter Organization.** We start by providing an overview of our contributions and techniques, as well as surveying related and follow-up works, in Section 2.1.

Some preliminaries and notation are described in Section 2.2. We formally define somewhat and fully homomorphic encryption and describe Gentry's "bootstrapping theorem" in Section 2.3. The main technical section of this chapter is Section 2.4, where our scheme is presented and fully analyzed. Lastly, our private information retrieval protocol is presented in Section 2.5.

## 2.1 Overview of Our Contributions and Techniques

In this section, we overview our contributions and techniques, as well as survey related prior and follow-up work. We start, in Sections 2.1.1, 2.1.2, by explaining our re-linearization and dimension-modulus reduction techniques (respectively) and their use for our scheme. We proceed in Section 2.1.3 to overview our private information retrieval protocol. Additional related work is discussed in Section 2.1.4.

### 2.1.1 Somewhat Homomorphic Encryption without Ideals

The starting point of Gentry's construction is a "somewhat" homomorphic encryption scheme. For a class of circuits $\mathcal{C}$, a $\mathcal{C}$-homomorphic scheme is one that allows evaluation of any circuit in the

[2]Strictly speaking, under this assumption, our scheme can evaluate polynomial-size circuits with a-priori bounded (but arbitrary) depth. A fully homomorphic encryption scheme independent of the circuit depth can be obtained by making an additional "circular security" assumption. See Section 2.3.

class $\mathcal{C}$. The simple, yet striking, observation in Gentry's work is that if a (slightly augmented) decryption circuit for a $\mathcal{C}$-homomorphic scheme resides in $\mathcal{C}$, then the scheme can be converted (or "bootstrapped") into a fully homomorphic encryption scheme.

It turns out that encryption schemes that can evaluate a non-trivial number of addition and multiplication operations[3] are already quite hard to come by (even without requiring that they are bootstrappable).[4] Gentry's solution to this was based on the algebraic notion of *ideals* in rings. In a very high level, the message is considered to be a ring element, and the ciphertext is the message masked with some "noise". The novelty of this idea is that the noise itself belonged to an ideal $I$. Thus, the ciphertext is of the form $m + xI$ (for some $x$ in the ring). Observe right off the bat that the scheme is born additively homomorphic; in fact, that will be the case with all the schemes we consider in this paper. The ideal $I$ has two main properties: first, a random element in the ideal is assumed to "mask" the message; and second, it is possible to generate a secret trapdoor that "annihilates" the ideal, i.e., implementing the transformation $m + xI \to m$. The first property guarantees security, while the second enables multiplying ciphertexts. Letting $c_1$ and $c_2$ be encryptions of $m_1$ and $m_2$ respectively,

$$c_1c_2 = (m_1 + xI)(m_2 + yI) = m_1m_2 + (m_1y + m_2x + xyI)I = m_1m_2 + zI$$

When decrypting, the ideal is annihilated and the product $m_1m_2$ survives. Thus, $c_1c_2$ is indeed an encryption of $m_1m_2$, as required. This nifty solution required, as per the first property, a hardness assumption on ideals in certain rings. Gentry's original work relied on hardness assumptions on *ideal lattices*, while van Dijk, Gentry, Halevi and Vaikuntanathan [vDGHV10] presented a different instantiation that considered ideals over the integers.

Our somewhat homomorphic scheme is based on the hardness of the "learning with errors" (LWE) problem, first presented by Regev [Reg05]. The LWE assumption states that if $\mathbf{s} \in \mathbb{Z}_q^n$ is an $n$ dimensional "secret" vector, any polynomial number of "noisy" random linear combinations of the coefficients of $\mathbf{s}$ are computationally indistinguishable from uniformly random elements in $\mathbb{Z}_q$. Mathematically,

$$\left\{\mathbf{a}_i, \langle \mathbf{a}_i, \mathbf{s}\rangle + e_i\right\}_{i=1}^{\mathrm{poly}(n)} \quad \overset{c}{\approx} \quad \left\{\mathbf{a}_i, u_i\right\}_{i=1}^{\mathrm{poly}(n)} \ ,$$

where $\mathbf{a}_i \in \mathbb{Z}_q^n$ and $u_i \in \mathbb{Z}_q$ are uniformly random, and the "noise" $e_i$ is sampled from a noise distribution that outputs numbers much smaller than $q$ (an example is a discrete Gaussian distribution over $\mathbb{Z}_q$ with small standard deviation).

The LWE assumption does not refer to ideals, and indeed, the LWE problem is at least as hard as finding short vectors in *any lattice*, as follows from the worst-case to average-case reductions of Regev [Reg05] and Peikert [Pei09]. As mentioned earlier, we have a much better understanding of the complexity of lattice problems (thanks to [LLL82, Ajt98, Mic00] and many others), compared to the corresponding problems on ideal lattices. Despite considerable effort, the best known algorithms to solve the $n$-dimensional LWE problem with noise magnitude $q/2^{n^{\epsilon}}$ run in time roughly $2^{n^{1-\epsilon}}$. The LWE assumption also turns out to be particularly amenable to the construction of simple, efficient

[3]All known schemes, including ours, treat evaluated functions as arithmetic circuits. Hence we use the terminology of "addition and multiplication" gates. The conversion to the boolean model (AND, OR, NOT gates) is immediate.

[4]We must mention here that we are interested only in *compact* fully homomorphic encryption schemes, namely ones where the ciphertexts do not grow in size with each homomorphic operation. If we do allow such growth in size, a number of solutions are possible. See, e.g., [SYY99, GHV10a, MGH10].

and highly expressive cryptographic schemes (e.g., [Reg05, GPV08, AGV09, ACPS09, CHKP10, ABB10] and many others). Our construction of a fully homomorphic encryption scheme from LWE is perhaps additional testament to its power and elegance.

Constructing a (symmetric-key) encryption scheme whose security is based on the LWE assumption is rather straightforward. To encrypt a bit $m \in \{0,1\}$ using secret key $\mathbf{s} \in \mathbb{Z}_q^n$, we choose a random vector $\mathbf{a} \in \mathbb{Z}_q^n$ and a "noise" $e$ and output the ciphertext

$$c \quad = \quad (\mathbf{a}, b = \langle \mathbf{a}, \mathbf{s} \rangle + 2e + m) \qquad \in \mathbb{Z}_q^n \times \mathbb{Z}_q \ .$$

The key observation in decryption is that the two "masks" – namely, the secret mask $\langle \mathbf{a}, \mathbf{s} \rangle$ and the "even mask" $2e$ – do not interfere with each other.[5] That is, one can decrypt this ciphertext by annihilating the two masks, one after the other: The decryption algorithm first re-computes the mask $\langle \mathbf{a}, \mathbf{s} \rangle$ and subtracts it from $b$, resulting in $2e + m \pmod q$. Since $e \ll q$, then $2e + m \pmod q = 2e + m$. Removing the even mask is now easy – simply compute $2e + m \pmod 2$.[6] As we will see below, the scheme is naturally additive homomorphic, yet multiplication presents a thorny problem. In fact, a recent work of Gentry, Halevi and Vaikuntanathan [GHV10b] showed that (a slight variant of) this scheme supports *just a single* homomorphic multiplication, but at the expense of a huge blowup to the ciphertext which made further advance impossible.

To better understand the homomorphic properties of this scheme, let us shift our focus away from the encryption algorithm, on to the decryption algorithm. Given a ciphertext $(\mathbf{a}, b)$, consider the symbolic linear function $f_{\mathbf{a},b} : \mathbb{Z}_q^n \to \mathbb{Z}_q$ defined as:

$$f_{\mathbf{a},b}(\mathbf{x}) = b - \langle \mathbf{a}, \mathbf{x} \rangle \pmod q = b - \sum_{i=1}^{n} \mathbf{a}[i] \cdot \mathbf{x}[i] \qquad \in \mathbb{Z}_q \ ,$$

where $\mathbf{x} = (\mathbf{x}[1], \ldots, \mathbf{x}[n])$ denotes the variables, and $(\mathbf{a}, b)$ forms the public coefficients of the linear equation. Clearly, decryption of the ciphertext $(\mathbf{a}, b)$ is nothing but evaluating this function on the secret key $\mathbf{s}$, and then taking the result modulo 2. (A similar representation of the ciphertext was also used in [BV11a].)

Homomorphic addition and multiplication can now be described in terms of this function $f$. Adding two ciphertexts corresponds to the addition of two linear functions, which is again another linear function. In particular, $f_{(\mathbf{a}+\mathbf{a}', b+b')}(\mathbf{x}) = f_{\mathbf{a},b}(\mathbf{x}) + f_{(\mathbf{a}',b')}(\mathbf{x})$ is the linear function corresponding to the "homomorphically added" ciphertext $(\mathbf{a} + \mathbf{a}', b + b')$. Similarly, multiplying two such ciphertexts corresponds to a symbolic multiplication of these linear equations

$$\begin{aligned} f_{(\mathbf{a},b)}(\mathbf{x}) \cdot f_{(\mathbf{a}',b)}(\mathbf{x}) &= (b - \sum \mathbf{a}[i]\mathbf{x}[i]) \cdot (b' - \sum \mathbf{a}'[i]\mathbf{x}[i]) \\ &= h_0 + \sum h_i \cdot \mathbf{x}[i] + \sum h_{i,j} \cdot \mathbf{x}[i]\mathbf{x}[j] \ , \end{aligned}$$

which results in a degree-2 polynomial in the variables $\mathbf{x} = (\mathbf{x}[1], \ldots, \mathbf{x}[n])$, with coefficients $h_{i,j}$ that can be computed from $(\mathbf{a}, b)$ and $(\mathbf{a}', b')$ by opening parenthesis of the expression above.

[5] We remark that using $2e$ instead of $e$ as in the original formulation of LWE does not adversely impact security, so long as $q$ is odd (since in that case 2 is a unit in $\mathbb{Z}_q$).

[6] Such "even mask" was required in the schemes of [Gen09b, vDGHV10] as well, in addition to the aforementioned "ideal mask". Roughly speaking, they needed this to ensure semantic security, as we do.

Decryption, as before, involves evaluating this quadratic expression on the secret key $\mathbf{s}$ (and then reducing modulo 2). We now run into a serious problem – the decryption algorithm has to know all the coefficients of this quadratic polynomial, which means that the size of the ciphertext just went up from $(n+1)$ elements to (roughly) $n^2/2$.

**A New Technique: Re-Linearization.** This is where our re-linearization technique comes into play. Re-linearization is a way to reduce the size of the ciphertext back down to $(n+1)$ elements. The main idea is the following: imagine that we publish "encryptions" of all the linear and quadratic terms in the secret key $\mathbf{s}$, namely all the numbers $\mathbf{s}[i]$ as well as $\mathbf{s}[i]\mathbf{s}[j]$, under a new secret key $\mathbf{t}$. Thus, these ciphertexts (for the quadratic terms) look like $(\mathbf{a}_{i,j}, b_{i,j})$ where

$$b_{i,j} = \langle \mathbf{a}_{i,j}, \mathbf{t} \rangle + 2e_{i,j} + \mathbf{s}[i] \cdot \mathbf{s}[j] \approx \langle \mathbf{a}_{i,j}, \mathbf{t} \rangle + \mathbf{s}[i] \cdot \mathbf{s}[j] \ .$$

Actually, calling these "encryptions" is inaccurate: $\mathbf{s}[i] \cdot \mathbf{s}[j] \in \mathbb{Z}_q$ is not a single bit and therefore the "ciphertext" cannot be decrypted. However, we feel that thinking of these as encryptions may benefit the reader's intuition. The sum $h_0 + \sum h_i \cdot \mathbf{s}[i] + \sum h_{i,j} \cdot \mathbf{s}[i]\mathbf{s}[j]$ can now be written (approximately) as

$$h_0 + \sum h_i(b_i - \langle \mathbf{a}_i, \mathbf{t} \rangle) + \sum_{i,j} h_{i,j} \cdot (b_{i,j} - \langle \mathbf{a}_{i,j}, \mathbf{t} \rangle) \ ,$$

which is a linear function in $\mathbf{t}$! The bottom-line is that multiplying the two linear functions $f_{(\mathbf{a},b)}$ and $f_{(\mathbf{a}',b')}$ and then re-linearizing the resulting expression results in a linear function with $(n+1)$ coefficients. Evaluating this new linear function on the new secret key $\mathbf{t}$ results in the product of the two original messages (upon reducing modulo 2). The resulting ciphertext is simply the $(n+1)$ coefficients of this linear function. This ciphertext will decrypt to $m \cdot m'$ using the secret key $\mathbf{t}$.

In this semi-formal description, we ignored an important detail which has to do with the coefficients' $h_{i,j}$ being potentially large. Thus, even though $(b_{i,j} - \langle \mathbf{a}_{i,j}, \mathbf{t} \rangle) \approx \mathbf{s}[i]\mathbf{s}[j]$, it may be the case that $h_{i,j} \cdot (b_{i,j} - \langle \mathbf{a}_{i,j}, \mathbf{t} \rangle) \not\approx h_{i,j} \cdot \mathbf{s}[i]\mathbf{s}[j]$. This is handled by considering the binary representation of $h_{i,j}$, namely $h_{i,j} = \sum_{\tau=0}^{\lfloor \log q \rfloor} 2^\tau \cdot h_{i,j,\tau}$. If, for each value of $\tau$, we had a pair $(\mathbf{a}_{i,j,\tau}, b_{i,j,\tau})$ such that

$$b_{i,j,\tau} = \langle \mathbf{a}_{i,j,\tau}, \mathbf{t} \rangle + 2e_{i,j,\tau} + 2^\tau \mathbf{s}[i] \cdot \mathbf{s}[j] \approx \langle \mathbf{a}_{i,j,\tau}, \mathbf{t} \rangle + 2^\tau \mathbf{s}[i] \cdot \mathbf{s}[j] \ ,$$

then indeed

$$h_{i,j} \cdot \mathbf{s}[i]\mathbf{s}[j] = \sum_{\tau=0}^{\lfloor \log q \rfloor} h_{i,j,\tau} 2^\tau \mathbf{s}[i]\mathbf{s}[j] \approx \sum_{\tau=0}^{\lfloor \log q \rfloor} h_{i,j,\tau}(b_{i,j,\tau} - \langle \mathbf{a}_{i,j,\tau}, \mathbf{t} \rangle) \ ,$$

since $h_{i,j,\tau} \in \{0,1\}$. This increases the number of pairs we need to post by a (polynomial) factor of $(\lfloor \log q \rfloor + 1)$.

This process allows us to do one multiplication without increasing the size of the ciphertext, and obtain an encryption of the product under a new secret key. *But why stop at two keys* $\mathbf{s}$ *and* $\mathbf{t}$? Posting a "chain" of $L$ secret keys (together with encryptions of quadratic terms of one secret key using the next secret key) allows us to perform up to $L$ levels of multiplications without blowing up the ciphertext size. It is possible to achieve multiplicative depth $L = \epsilon \log n$ (which corresponds to evaluating a degree $D = n^\epsilon$ polynomial) for an arbitrary constant $\epsilon < 1$ under reasonable assumptions, but beyond that, the growth of the error in the ciphertext kicks in, and destroys the ciphertext. Handling this requires us to use the machinery of bootstrapping, which we explain in the next section.

In conclusion, the above technique allows us to remove the need for "ideal assumptions" and obtain *somewhat* homomorphic encryption from $\mathsf{LWE}$. This scheme, which we call $\mathsf{SH}$, will be a building block towards our full construction. The scheme $\mathsf{SH}$ is formally presented in Section 2.4.1.

### 2.1.2 Fully Homomorphic Encryption Without Squashing

As explained above, the "bootstrapping" method for achieving full homomorphism requires a $\mathcal{C}$-homomorphic scheme whose decryption circuit resides in $\mathcal{C}$. All prior somewhat homomorphic schemes fell short in this category and failed to achieve this requirement in a natural way. Thus Gentry, followed by all other previous schemes, resorted to "squashing": a method for reducing the decryption complexity at the expense of making an additional and fairly strong assumption, namely the sparse subset sum assumption.

We show how to "upgrade" our somewhat homomorphic scheme (explained in Section 2.1.1) into a scheme that enjoys the same amount of homomorphism but has a much smaller decryption circuit. All of this, without making any additional assumption (beyond $\mathsf{LWE}$).

Our starting point is the somewhat homomorphic scheme from Section 2.1.1. Recall that a ciphertext in that scheme is of the form $(\mathbf{a}, b = \langle \mathbf{a}, \mathbf{s} \rangle + 2e + m) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$, and decryption is done by computing $(b - \langle \mathbf{a}, \mathbf{s} \rangle \bmod q) \pmod 2$. One can verify that this computation, presented as a polynomial in the bits of $\mathbf{s}$, has degree at least $\max(n, \log q)$, which is more than the maximal degree $D$ that our scheme can homomorphically evaluate. The bottom line is that decryption complexity is governed by $(n, \log q)$ which are too big for our homomorphism capabilities.

**A New Technique: Dimension-Modulus Reduction.** Dimension-modulus reduction enables us to take a ciphertext with parameters $(n, \log q)$ as above, and convert it into a ciphertext of the same message, but with parameters $(k, \log p)$ which are much smaller than $(n, \log q)$. To give a hint as to the magnitude of improvement, we typically set $k$ to be of size the security parameter and $p = \mathrm{poly}(k)$. We can then set $n = k^c$ for essentially *any constant* $c$, and $q = 2^{n^\epsilon}$. We will thus be able to homomorphically evaluate functions of degree roughly $D = n^\epsilon = k^{c \cdot \epsilon}$ and we can choose $c$ to be large enough so that this is sufficient to evaluate the $(k, \log p)$ decryption circuit.

To understand dimension-modulus reduction technically, we go back to re-linearization. We showed above that, posting proper public parameters, one can convert a ciphertext $(\mathbf{a}, b = \langle \mathbf{a}, \mathbf{s} \rangle + 2e + m)$, that corresponds to a secret key $\mathbf{s}$, into a ciphertext $(\mathbf{a}', b' = \langle \mathbf{a}', \mathbf{t} \rangle + 2e' + m)$ that corresponds to a secret key $\mathbf{t}$.[7] The crucial observation is that $\mathbf{s}$ and $\mathbf{t}$ need not have the same dimension $n$. Specifically, if we chose $\mathbf{t}$ to be of dimension $k$, the procedure still works. This brings us down from $(n, \log q)$ to $(k, \log q)$, which is a big step but still not sufficient.

Having the above observation in mind, we wonder if we can take $\mathbf{t}$ to have not only low dimension but also small modulus $p$, thus completing the transition from $(n, \log q)$ to $(k, \log p)$. This is indeed possible using some additional ideas, where the underlying intuition is that $\mathbb{Z}_p$ can "approximate" $\mathbb{Z}_q$ by simple scaling, up to a small error.

[7] In the previous section, we applied re-linearization to a quadratic function of $\mathbf{s}$, while here we apply it to the ciphertext $(\mathbf{a}, b)$ that corresponds to a linear function of $\mathbf{s}$. This only makes things easier.

The public parameters for the transition from $\mathbf{s}$ to $\mathbf{t}$ will be $(\mathbf{a}_{i,\tau}, b_{i,\tau}) \in \mathbb{Z}_p^k \times \mathbb{Z}_p$, where

$$b_{i,\tau} = \langle \mathbf{a}_{i,\tau}, \mathbf{t} \rangle + e + \left\lfloor \frac{p}{q} \cdot 2^\tau \cdot \mathbf{s}[i] \right\rceil .[8]$$

Namely, we scale $2^\tau \cdot \mathbf{s}[i] \in \mathbb{Z}_q$ into an element in $\mathbb{Z}_p$ by multiplying by $p/q$ and rounding. The rounding incurs additional noise of magnitude at most $1/2$. It follows that

$$2^\tau \cdot \mathbf{s}[i] \approx \frac{q}{p} \cdot (b_{i,\tau} - \langle \mathbf{a}_{i,\tau}, \mathbf{t} \rangle) ,$$

which enables converting a linear equation in $\mathbf{s}$ into a linear equation in $\mathbf{t}$. The result of dimension-modulus reduction, therefore, is a ciphertext $(\hat{\mathbf{a}}, \hat{b}) \in \mathbb{Z}_p^k \times \mathbb{Z}_p$ such that $\hat{b} - \langle \hat{\mathbf{a}}, \mathbf{t} \rangle = m + 2\hat{e}$. For security, we need to assume the hardness of $\mathsf{LWE}$ with parameters $k, p$. We can show that in the parameter range we use, this assumption is as hard as the one used for the somewhat homomorphic scheme.[9]

In conclusion, dimension-modulus reduction allows us to achieve a bootstrappable scheme, based on the $\mathsf{LWE}$ assumption alone. We refer the reader to Section 2.4 for the formal presentation and full analysis of our entire solution. Specifically, dimension-modulus reduction is implemented in the scheme $\mathsf{BTS}$ described in Section 2.4.2.

As a nice byproduct of this technique, the ciphertexts of the resulting fully homomorphic scheme become very short. They now consist of $(k+1)\log p = O(k \log k)$ bits. This is a desirable property which enables efficient private information retrieval protocols (see Section 2.1.3 below).

### 2.1.3 Near-Optimal Private Information Retrieval

In (single-server) private information retrieval (PIR) protocols, a very large *database* is maintained by a *sender* (the sender is also sometimes called the server, or the database). A *receiver* wishes to obtain a specific entry in the database, without revealing any information about this entry to the server. Typically, we consider databases of size exponential in the security parameter and hence we wish that the receiver's running time and communication complexity are polylogarithmic in the size of the database $N$. The first polylogarithmic candidate protocol was presented by Cachin, Micali and Stadler [CMS99] and additional polylograithmic protocols were introduced by Lipmaa [Lip05] and by Gentry and Ramzan [GR05]. Of which, the latter achieves the best communication complexity of $O(\log^{3-o(1)}(N))$.[10] The latter two protocols achieve constant amortized communication

[8] A subtle technical point refers to the use of noise term $e$, instead of $2e$ as we did for re-linearization. The reason is roughly that $\frac{q}{p} \cdot 2$ is non-integer. Therfore we "divide by 2" before performing the dimension-reduction and "multiply back" by 2 after.

[9] For the informed reader we mention that while $k, p$ are smaller than $n, q$ and therefore seem to imply lesser security, we are able to use much higher relative noise in our $k, p$ scheme since it needs not support homomorphism. Hence the two assumptions are of roughly the same hardness.

[10] It is hard to compare the performance of different PIR protocols due to the multitude of parameters. To make things easier to grasp, we compare the protocols on equal grounds: We assume that the database size and the adversary's running time are exponential in the security parameter and assume the maximal possible hardness of the underlying assumption against known attacks. We also assume that each query retrieves a single bit. We will explicitly mention special properties of individual protocols that are not captured by this comparison.

complexity when retrieving large consecutive blocks of data. See a survey in [OS07] for more details on these protocols.

Fully homomorphic, or even somewhat homomorphic, encryption is known to imply polylogarithmic PIR protocols (to be precise, one needs sub-exponentially secure such schemes). Most trivially, the receiver can encrypt the index it wants to query, and the database will use that to homomorphically evaluate the database access function, thus retrieving an encryption of the answer and sending it to the receiver. The total communication complexity of this protocol is the sum of lengths of the public key, encryption of the index and output ciphertext. However, the public key is sent only once, it is independent of the database and the query, and it can be used for many queries. Therefore it is customary to analyze such schemes in the *public key model* where sending the public key does not count towards the communication complexity. Gentry [Gen09a] proposes to use his somewhat homomorphic scheme towards this end, which requires $O(\log^3 N)$ bit communication.[11] We show how, using our somewhat homomorphic scheme, in addition to new ideas, we can bring down communication complexity to a near optimal $\log N \cdot \mathrm{polyloglog}\, N$ (one cannot do better than $\log N$). To obtain the best parameters, one needs to assume $2^{\widetilde{\Omega}(k)}$-hardness of polynomial-factor approximation for short vector problems in arbitrary dimension $k$ lattices, which is supported by current knowledge. Details follow.

A major obstacle in the naïve use of somewhat homomorphic encryption for PIR is that homomorphism is obtained with respect to the boolean representation of the evaluated function. Therefore, the receiver needs to encrypt the index to the database in a bit-by-bit manner. The query is then composed of $\log N$ ciphertexts, which necessitate at least $\log^2 N$ bits of communication. As a first improvement, we notice that the index needs not be encrypted under the somewhat homomorphic scheme. Rather, we can encrypt using any *symmetric* encryption scheme. The database will receive an encrypted symmetric key (under the homomorphic scheme), which will enable it to convert symmetric ciphertexts into homomorphic ciphertexts without additional communication. The encrypted secret key can be sent as a part of the public key as it is independent of the query. This, of course, requires that our somewhat homomorphic scheme can homomorphically evaluate the decryption circuit of the symmetric scheme. Fully homomorphic schemes will certainly be adequate for this purpose, but known somewhat homomorphic schemes are also sufficient (depending on the symmetric scheme to be used). Using the most communication efficient symmetric scheme, we bring down the query complexity to $O(\log N)$. As for the sender's response, our dimension-modulus reduction technique guarantees very short ciphertexts (essentially as short as non-homomorphic $\mathsf{LWE}$ based schemes). This translates into $\log N \cdot \mathrm{polyloglog}\, N$ bits per ciphertext, and the communication complexity of our protocol follows. We remark that in terms of retrieving large blocks of consecutive data, one can slightly reduce the overhead to $O(\log N)$ bits of communication for every bit of retrieved data. We leave it as an open problem to bring the amortized communication down to a constant. See Section 2.5 for the full details.

Prior to this work, it was not at all known how to achieve even polylogarithmic PIR under the $\mathsf{LWE}$ assumption. We stress that even if the size of the public key does count towards the communication complexity, our protocol still has polylogarithmic communication.

[11] Gentry does not provide a detailed analysis of this scheme, the above is based on our analysis of its performance.

### 2.1.4 Other Related Work

Aside from Gentry's scheme (and a variant thereof by Smart and Vercauteren [SV10] and an optimization by Stehle and Steinfeld [SS10]), there are two other fully homomorphic encryption schemes [vDGHV10, BV11a]. The innovation in both these schemes is the construction of a new *somewhat homomorphic* encryption scheme. Both these works then invoke Gentry's squashing and bootstrapping transformations to convert it to a fully homomorphic scheme, and thus the security of both these schemes relies on the sparse subset-sum assumption (plus other assumptions). The first of these schemes is due to van Dijk, Gentry, Halevi and Vaikuntanathan [vDGHV10]. Their scheme works over the integers and relies on a new assumption which, roughly speaking, states that finding the greatest common divisor of many "noisy" multiples of a number is computationally hard. They cannot, however, reduce their assumption to worst-case hardness. The second is a recent work with Vaikuntanathan [BV11a], where somewhat homomorphic encryption scheme is constructed based on the ring LWE problem [LPR10] whose security can be reduced to the worst-case hardness of problems on *ideal lattices*.

The efficiency of implementing Gentry's scheme also gained much attention. Smart and Vercauteren [SV10], as well as Gentry and Halevi [GH11b] conduct a study on reducing the complexity of implementing the scheme.

In a recent independent work, Gentry and Halevi [GH11a] showed how the sparse subset sum assumption can be replaced by either the (decisional) Diffie-Hellman assumption or an ideal lattice assumption, by representing the decryption circuit as an arithmetic circuit with only one level of (high fan-in) multiplications.

In a follow-up work with Gentry and Vaikuntanathan [BGV11], we extend the techniques in this chapter. Specifically we show how to reduce the modulus in such a way that the growth of the error is better bounded. This implies a scheme that is based on weaker LWE variants (i.e. harder problems) and with greater efficiency.

## 2.2 Preliminaries

**Notations.** Let $\mathcal{D}$ denote a distribution over some finite set $S$. Then, $x \stackrel{\$}{\leftarrow} \mathcal{D}$ denotes that $x$ is chosen from the distribution $\mathcal{D}$. When we say $x \stackrel{\$}{\leftarrow} S$, we mean that $x$ is chosen from the uniform distribution over $S$. Unless explicitly mentioned, all logarithms are to base 2.

In this work, we utilize "noise" distributions over integers. The only property of these distributions we use is their magnitude. Hence, we define a $B$-bounded distribution to be a distribution over the integers where the magnitude of a sample is bounded with high probability. A definition follows.

**Definition 2.2.1** ($B$-bounded distributions)**.** *A distribution ensemble $\{\chi_n\}_{n\in\mathbb{N}}$, supported over the integers, is called $B$-bounded if*

$$\Pr_{e \stackrel{\$}{\leftarrow} \chi_n}\left[|e| > B\right] \le 2^{-\widetilde{\Omega}(n)} .$$

We denote scalars in plain (e.g. $x$) and vectors in bold lowercase (e.g. $\mathbf{v}$), and matrices in bold uppercase (e.g. $\mathbf{A}$). The $\ell_i$ norm of a vector is denoted by $\|\mathbf{v}\|_i$. Inner product is denoted by $\langle \mathbf{v}, \mathbf{u} \rangle$, recall that $\langle \mathbf{v}, \mathbf{u} \rangle = \mathbf{v}^T \cdot \mathbf{u}$. Let $\mathbf{v}$ be an $n$ dimensional vector. For all $i = 1, \ldots, n$, the $i^{\text{th}}$ element in $\mathbf{v}$ is denoted $\mathbf{v}[i]$. We use the convention that $\mathbf{v}[0] \triangleq 1$.

We use the following variant of the leftover hash lemma [ILL89].

**Lemma 2.2.1** (matrix-vector leftover hash lemma)**.** *Let $\kappa \in \mathbb{N}$, $n \in \mathbb{N}$, $q \in \mathbb{N}$, and $m \geq n \log q + 2\kappa$. Let $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{m \times n}$ be a uniformly random matrix, let $\mathbf{r} \xleftarrow{\$} \{0,1\}^m$ and let $\mathbf{y} \xleftarrow{\$} \mathbb{Z}_q^n$. Then,*

$$\mathtt{dist}\left((\mathbf{A}, \mathbf{A}^T\mathbf{r}), (\mathbf{A}, \mathbf{y})\right) \leq 2^{-\kappa} ,$$

*where $\mathtt{dist}(A, B)$ denotes the statistical distance between the distributions $A$ and $B$.*

## 2.2.1 Learning With Errors (LWE)

The LWE problem was introduced by Regev [Reg05] as a generalization of "learning parity with noise". For positive integers $n$ and $q \geq 2$, a vector $\mathbf{s} \in \mathbb{Z}_q^n$, and a probability distribution $\chi$ on $\mathbb{Z}_q$, let $A_{\mathbf{s},\chi}$ be the distribution obtained by choosing a vector $\mathbf{a} \xleftarrow{\$} \mathbb{Z}_q^n$ uniformly at random and a noise term $e \xleftarrow{\$} \chi$, and outputting $(\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle + e) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$. A formal definition follows.

**Definition 2.2.2** (LWE)**.** *For an integer $q = q(n)$ and an error distribution $\chi = \chi(n)$ over $\mathbb{Z}_q$, the learning with errors problem $\mathsf{LWE}_{n,m,q,\chi}$ is defined as follows: Given $m$ independent samples from $A_{\mathbf{s},\chi}$ (for some $\mathbf{s} \in \mathbb{Z}_q^n$), output $\mathbf{s}$ with noticeable probability.*

*The (average-case) decision variant of the LWE problem, denoted $\mathsf{DLWE}_{n,m,q,\chi}$, is to distinguish (with non-negligible advantage) $m$ samples chosen according to $A_{\mathbf{s},\chi}$ (for uniformly random $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$), from $m$ samples chosen according to the uniform distribution over $\mathbb{Z}_q^n \times \mathbb{Z}_q$. We denote by $\mathsf{DLWE}_{n,q,\chi}$ the variant where the adversary gets oracle access to $A_{\mathbf{s},\chi}$, and is not a-priori bounded in the number of samples.*

For cryptographic applications, we are primarily interested in the average case decision problem DLWE, where $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$. There are known quantum [Reg05] and classical [Pei09] reductions between $\mathsf{DLWE}_{n,m,q,\chi}$ and approximating short vector problems in lattices. Specifically, these reductions take $\chi$ to be (discretized versions of) the Gaussian distribution, which is $B$-bounded for an appropriate $B$. Since the exact distribution $\chi$ does not matter for our results, we state a corollary of the results of [Reg05, Pei09] in terms of the bound on the distribution.

**Corollary 2.2.2** ([Reg05, Pei09])**.** *Let $q = q(n) \in \mathbb{N}$ be a product of co-prime numbers $q = \prod q_i$ such that for all $i$, $q_i = \mathrm{poly}(n)$, and let $B \geq n$. Then there exists an efficiently sampleable $B$-bounded distribution $\chi$ such that if there is an efficient algorithm that solves the (average-case) $\mathsf{DLWE}_{n,q,\chi}$ problem. Then:*

- *There is a quantum algorithm solving $\mathsf{SIVP}_{\tilde{O}(n\sqrt{n} \cdot q/B)}$ and $\mathsf{gapSVP}_{\tilde{O}(n\sqrt{n} \cdot q/B)}$ on any $n$ dimensional lattice, and runs in time $\mathrm{poly}(n)$.*
- *There is a classical algorithm solving the $\zeta$-to-$\gamma$ decisional shortest vector problem $\mathsf{gapSVP}_{\zeta,\gamma}$, where $\gamma = \tilde{O}(n\sqrt{n} \cdot q/B)$, and $\zeta = \tilde{O}(q\sqrt{n})$, on any $n$-dimensional lattice, and runs in time $\mathrm{poly}(n)$.*

We refer the reader to [Reg05, Pei09] for the formal definition of these lattice problems, as they have no direct connection to this work. We only note here that the best known algorithms for these problems run in time nearly exponential in the dimension $n$ [AKS01, MV10]. More generally, the best algorithms that approximate these problems to within a factor of $2^k$ run in time $2^{\tilde{O}(n/k)}$.

### 2.2.2 Symmetric Encryption

A symmetric encryption scheme $\mathsf{SYM} = (\mathsf{SYM.Keygen}, \mathsf{SYM.Enc}, \mathsf{SYM.Dec})$, over message space $\mathcal{M} = \{\mathcal{M}_\kappa\}_{\kappa \in \mathbb{N}}$, is a triple of PPT algorithms as follows. (We always denote the security parameter by $\kappa$.)

- **Key generation.** The algorithm $sk \leftarrow \mathsf{SYM.Keygen}(1^\kappa)$ takes a unary representation of the security parameter and outputs symmetric encryption/decryption key $sk$.
- **Encryption.** The algorithm $c \leftarrow \mathsf{SYM.Enc}_{sk}(\mu)$ takes the symmetric key $sk$ and a message $\mu \in \mathcal{M}_\kappa$ and outputs a ciphertext $c$.
- **Decryption.** The algorithm $\mu^* \leftarrow \mathsf{SYM.Dec}_{sk}(c)$ takes the symmetric key $sk$ and a ciphertext $c$ and outputs a message $\mu^* \in \mathcal{M}_\kappa$.

Correctness and security against chosen plaintext attacks (IND-CPA security) are defined as follows.

**Definition 2.2.3.** *A symmetric scheme* $\mathsf{SYM}$ *is correct if for all* $\mu$ *and all* $sk \leftarrow \mathsf{SYM.Keygen}(1^\kappa)$*,*

$$\Pr[\mathsf{SYM.Dec}_{sk}(\mathsf{SYM.Enc}_{sk}(\mu)) \neq \mu] = \mathrm{negl}(\kappa) \ ,$$

*where the probability is over the coins of* $\mathsf{SYM.Keygen}$*,* $\mathsf{SYM.Enc}$ *(we can assume w.l.o.g that* $\mathsf{SYM.Dec}$ *is deterministic).*

**Definition 2.2.4.** *A symmetric scheme* $\mathsf{SYM}$ *is* $(t, \epsilon)$*-IND-CPA secure if for any adversary* $\mathcal{A}$ *that runs in time* $t$ *it holds that*

$$\left| \Pr[\mathcal{A}^{\mathsf{SYM.Enc}_{sk}(\cdot)}(1^\kappa) = 1] - \Pr[\mathcal{A}^{\mathsf{SYM.Enc}_{sk}(0)}(1^\kappa) = 1] \right| \leq \epsilon \ ,$$

*where the probability is over* $sk \leftarrow \mathsf{SYM.Keygen}(1^\kappa)$*, the coins of* $\mathsf{SYM.Enc}$ *and the coins of the adversary* $\mathcal{A}$*.*

Namely, no adversary can distinguish between an oracle that encrypts messages of its choice and an oracle that only returns encryptions of 0 (where 0 is some arbitrary element in the message space).

## 2.3 Homomorphic Encryption: Definitions and Tools

In this section we discuss the definition of homomorphic encryption and its properties as well as some related subjects. We start by defining homomorphic and fully homomorphic encryption in Section 2.3.1. Then, in Section 2.3.2 we discuss Gentry's bootstrapping theorem.

We note that there is a number of ways to define homomorphic encryption and to describe the bootstrapping theorem. We chose the definitions that best fit our constructions and we urge even the knowledgeable reader to go over them so as to avoid confusion in interpreting our results.

### 2.3.1 Homomorphic Encryption – Definitions

We define homomorphic encryption and its desired properties. Throughout this section (and this work) we use $\kappa$ to indicate the security parameter. In addition, all schemes in this paper encrypt bit-by-bit and therefore our definitions only refer to this case. The generalization to an arbitrary message space is immediate.

A homomorphic (public-key) encryption scheme $\mathsf{HE} = (\mathsf{HE.Keygen}, \mathsf{HE.Enc}, \mathsf{HE.Dec}, \mathsf{HE.Eval})$ is a quadruple of PPT algorithms as follows.

- **Key generation.** The algorithm $(pk, evk, sk) \leftarrow \mathsf{HE.Keygen}(1^\kappa)$ takes a unary representation of the security parameter and outputs a public encryption key $pk$, a public evaluation key $evk$ and a secret decryption key $sk$.

- **Encryption.** The algorithm $c \leftarrow \mathsf{HE.Enc}_{pk}(\mu)$ takes the public key $pk$ and a single bit message $\mu \in \{0,1\}$ and outputs a ciphertext $c$.

- **Decryption.** The algorithm $\mu^* \leftarrow \mathsf{HE.Dec}_{sk}(c)$ takes the secret key $sk$ and a ciphertext $c$ and outputs a message $\mu^* \in \{0,1\}$.

- **Homomorphic evaluation.** The algorithm $c_f \leftarrow \mathsf{HE.Eval}_{evk}(f, c_1, \ldots, c_\ell)$ takes the evaluation key $evk$, a function $f : \{0,1\}^\ell \to \{0,1\}$ and a set of $\ell$ ciphertexts $c_1, \ldots, c_\ell$, and outputs a ciphertext $c_f$.

  The representation of the function $f$ can vary between schemes. In this work, $f$ will be represented by an arithmetic circuit over GF(2).

We note that while one can treat the evaluation key as a part of the public key, as has been done in the literature so far, we feel that there is an expository value to treating it as a separate entity. We thus distinguish between the public elements that are used for encryption and those that are used only for homomorphic evaluation.

The only security notion we consider in this chapter is semantic security, namely security w.r.t. passive adversaries. We use its widely known formulation as IND-CPA security, defined as follows.

**Definition 2.3.1** (CPA security)**.** *A scheme* $\mathsf{HE}$ *is IND-CPA secure if for any polynomial time adversary* $\mathcal{A}$ *it holds that*

$$\mathrm{Adv}_{\mathrm{CPA}}[\mathcal{A}] \triangleq |\Pr[\mathcal{A}(pk, evk, \mathsf{HE.Enc}_{pk}(0)) = 1] - \Pr[\mathcal{A}(pk, evk, \mathsf{HE.Enc}_{pk}(0)) = 1]| = \mathrm{negl}(\kappa) \ ,$$

*where* $(pk, evk, sk) \leftarrow \mathsf{HE.Keygen}(1^\kappa)$.

In fact, based on the best known about lattices, the schemes we present in this paper will be secure against even stronger adversaries. In order for our reductions to make sense for such adversaries as well, we also consider a parameterized version of CPA security. There, we allow the adversary to run in time $t$ (which is typically super-polynomial) and succeed with probability $\epsilon$ (which is typically sub-polynomial).

**Definition 2.3.2** ($(t,\epsilon)$-CPA security)**.** *A scheme* $\mathsf{HE}$ *is* $(t,\epsilon)$*-IND-CPA secure if for any adversary* $\mathcal{A}$ *that runs in time* $t$ *for* $t = t(\kappa)$ *it holds that*

$$\mathrm{Adv}_{\mathrm{CPA}}[\mathcal{A}] \triangleq |\Pr[\mathcal{A}(pk, evk, \mathsf{HE.Enc}_{pk}(0)) = 1] - \Pr[\mathcal{A}(pk, evk, \mathsf{HE.Enc}_{pk}(0)) = 1]| \leq \epsilon = \epsilon(\kappa) \ ,$$

*where* $(pk, evk, sk) \leftarrow \mathsf{HE.Keygen}(1^\kappa)$.

We move on to define the homomorphism property. Note that we do not define the "correctness" of the scheme separately, but rather (some form of) correctness will follow from our homomorphism properties.

We start by defining $\mathcal{C}$-homomorphism, which is homomorphism with respect to a specified class $\mathcal{C}$ of functions. This notion is also known as "somewhat homomorphism".

**Definition 2.3.3** ($\mathcal{C}$-homomorphism)**.** *Let $\mathcal{C} = \{\mathcal{C}_\kappa\}_{\kappa \in \mathbb{N}}$ be a class of functions (together with their respective representations). A scheme* HE *is $\mathcal{C}$-homomorphic (or, homomorphic for the class $\mathcal{C}$) if for any sequence of functions $f_\kappa \in \mathcal{C}_\kappa$ and respective inputs $\mu_1, \ldots, \mu_\ell \in \{0,1\}$ (where $\ell = \ell(\kappa)$), it holds that*

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

*where* $(pk, evk, sk) \leftarrow \mathsf{HE.Keygen}(1^\kappa)$ *and* $c_i \leftarrow \mathsf{HE.Enc}_{pk}(\mu_i)$.

We point out two important properties that the above definition *does not* require. First of all, we do not require that the ciphertexts $c_i$ are decryptable themselves, only that they become decryptable after homomorphic evaluation.[12] Secondly, we do not require that the output of HE.Eval can undergo additional homomorphic evaluation (this is termed "1-hop homomorphism" in [GHV10a]).

Before we define full homomorphism, let us define the notion of *compactness* (note that a $\mathcal{C}$-homomorphic scheme is not necessarily compact).

**Definition 2.3.4** (compactness)**.** *A homomorphic scheme* HE *is compact if there exists a polynomial $s = s(\kappa)$ such that the output length of* $\mathsf{HE.Eval}(\cdots)$ *is at most $s$ bits long (regardless of $f$ or the number of inputs).*

We give the minimal definition of fully homomorphic encryption, which suffices for most applications.

**Definition 2.3.5** (fully homomorphic encryption)**.** *A scheme* HE *is fully homomorphic if it is both compact and homomorphic for the class of all arithmetic circuits over $GF(2)$.*

As in the definition of $\mathcal{C}$ homomorphism, one can require that the outputs of HE.Eval can again be used as inputs for homomorphic evaluation ("multi-hop homomorphism"). Indeed, all known schemes have this additional property. However, due to the complexity of the formal definition in this case, we refrain from describing it.

An important relaxation of fully homomorphic encryption is the following.

**Definition 2.3.6** (leveled fully homomorphic encryption)**.** *A leveled fully homomorphic encryption scheme is a homomorphic scheme where the* HE.Keygen *gets an additional input $1^L$ (now $(pk, evk, sk) \leftarrow \mathsf{HE.Keygen}(1^\kappa, 1^L)$) and the resulting scheme is homomorphic for all depth-$L$ binary arithmetic circuits. The bound $s(\kappa)$ on the ciphertext length must remain independent of $L$.*

In most cases, the only parameter of the scheme that becomes dependent on $L$ is the bit-length of the evaluation key $evk$.

[12] This might seem strange at first, but such notion of somewhat homomorphism is all that is required to bootstrap into full homomorphism. This definition makes our schemes easier to describe and we note that one can always perform a "blank" homomorphic operation and then decrypt, so functionality is not hurt.

### 2.3.2 Gentry's Bootstrapping Technique

In this section we formally define the notion of a bootstrappable encryption scheme and present Gentry's bootstrapping theorem [Gen09b, Gen09a] which implies that a bootstrappable scheme can be converted into a fully homomorphic one.

**Definition 2.3.7** (bootstrappable encryption scheme)**.** *Let* $\mathsf{HE}$ *be* $\mathcal{C}$*-homomorphic, and let* $f_{\mathsf{add}}$ *and* $f_{\mathsf{mult}}$ *be the augmented decryption functions of the scheme defined as*

$$f_{\mathsf{add}}^{c_1,c_2}(s) = \mathsf{HE.Dec}_s(c_1) \ \textit{XOR} \ \mathsf{HE.Dec}_s(c_2) \qquad \textit{and} \qquad f_{\mathsf{mult}}^{c_1,c_2}(s) = \mathsf{HE.Dec}_s(c_1) \ \textit{AND} \ \mathsf{HE.Dec}_s(c_2) \ .$$

*Then* $\mathsf{HE}$ *is bootstrappable if*

$$\left\{f_{\mathsf{add}}^{c_1,c_2}, f_{\mathsf{mult}}^{c_1,c_2}\right\}_{c_1,c_2} \subseteq \mathcal{C} \ .$$

*Namely, the scheme can homomorphically evaluate* $f_{\mathsf{add}}$ *and* $f_{\mathsf{mult}}$.

We describe two variants of Gentry's bootstrapping theorem. The first implies leveled fully homomorphic encryption but requires no additional assumption; where the second makes an additional *(weak) circular security* assumption and achieves the stronger (non-leveled) variant of Definition 2.3.5.

The first variant follows.

**Theorem 2.3.1** ([Gen09b, Gen09a])**.** *Let* $\mathsf{HE}$ *be a bootstrappable scheme, then there exists a leveled fully homomorphic encryption scheme as per Definition 2.3.6.*

Specifically, the leveled homomorphic scheme is such that only the length of the evaluation key depends on the level $L$. All other parameters of the scheme are distributed identically regardless of the value of $L$.

For the second variant, we need to define circular security.

**Definition 2.3.8** (weak circular security)**.** *A public key encryption scheme* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *is weakly circular secure if it is IND-CPA secure even for an adversary with auxiliary information containing encryptions of all secret key bits:* $\{\mathsf{Enc}_{pk}(sk[i])\}_i$.

Namely, no polynomial time adversary can distinguish an encryption of 0 from an encryption of 1 even given the additional information.

We can now state the second theorem.

**Theorem 2.3.2** ([Gen09b, Gen09a])**.** *Let* $\mathsf{HE}$ *be a bootstrappable scheme that is also weakly circular secure. Then there is a fully homomorphic encryption scheme as per Definition 2.3.5.*

Finally, we want to make a statement regarding the ciphertext length of a bootstrapped scheme. The following is implicit in [Gen09b, Gen09a].

**Lemma 2.3.3.** *If a scheme* $\mathsf{FH}$ *is obtained from applying either Theorem 2.3.1 or Theorem 2.3.2 to a bootstrappable scheme* $\mathsf{HE}$*, then both* $\mathsf{FH.Enc}$ *and* $\mathsf{FH.Eval}$ *produce ciphertexts of the same length as* $\mathsf{HE.Eval}$ *(regardless of the length of the ciphertext produced by* $\mathsf{HE.Enc}$*).*

## 2.4 The New Fully Homomorphic Encryption Scheme

In this section, we present our fully homomorphic encryption scheme and analyze its security and performance. We present our scheme in a gradual manner. First, in Section 2.4.1 we present an LWE-based somewhat homomorphic scheme, SH, that will serve as building block for our construction (the scheme SH by itself is not sufficient to achieve full homomorphism). The main technique used here is re-linearization. Our bootstrappable scheme, BTS, which utilizes dimension-modulus reduction, is presented in Section 2.4.2. In Section 2.4.3 we prove the security of the schemes based on LWE and discuss the worst case hardness implied by known reductions. In Section 2.4.4 we analyze the homomorphic properties of SH and BTS. This analysis enables us to prove (in Section 2.4.5) that the bootstrapping theorem is applicable to BTS, and thus obtain a fully homomorphic scheme based on LWE. We then discuss the parameters and efficiency of our scheme.

### 2.4.1 The Scheme SH: A Somewhat Homomorphic Encryption Scheme

We present a somewhat homomorphic public-key encryption scheme, based on our re-linearization technique, whose message space is GF(2).[13] Let $\kappa \in \mathbb{N}$ be the security parameter. The scheme is parameterized by a dimension $n \in \mathbb{N}$, a positive integer $m \in \mathbb{N}$, an odd modulus $q \in \mathbb{N}$ (note that $q$ needs not be prime) and a noise distribution $\chi$ over $\mathbb{Z}$, all of which are inherited from the LWE assumption we use. An additional parameter of the scheme is a number $L \in \mathbb{N}$ which is an upper bound on the maximal multiplicative depth that the scheme can homomorphically evaluate.

During the exposition of the scheme, we invite the reader to keep the following range of parameters in mind: the dimension $n$ is polynomial in the security parameter $\kappa$, $m \geq n \log q + 2\kappa$ is a polynomial in $n$, the modulus is an odd number $q \in [2^{n^\epsilon}, 2 \cdot 2^{n^\epsilon})$ which is sub-exponential in $n$ ($\epsilon \in (0,1)$ is some constant), $\chi$ is some noise distribution that produces small samples (say, of magnitude at most $n$) in $\mathbb{Z}$, and the depth bound is $L \approx \epsilon \log n$.

- Key generation $\mathsf{SH.Keygen}(1^\kappa)$: For key generation, sample $L+1$ vectors $\mathbf{s}_0, \ldots, \mathbf{s}_L \xleftarrow{\$} \mathbb{Z}_q^n$, and compute, for all $\ell \in [L]$, $0 \leq i \leq j \leq n$, and $\tau \in \{0, \ldots, \lfloor \log q \rfloor\}$, the value

  $$\psi_{\ell,i,j,\tau} := \Big(\mathbf{a}_{\ell,i,j,\tau}\ ,\ b_{\ell,i,j,\tau} := \langle \mathbf{a}_{\ell,i,j,\tau}, \mathbf{s}_\ell \rangle + 2 \cdot e_{\ell,i,j,\tau} + 2^\tau \cdot \mathbf{s}_{\ell-1}[i] \cdot \mathbf{s}_{\ell-1}[j]\Big) \in \mathbb{Z}_q^n \times \mathbb{Z}_q\ , \quad (2.1)$$

  where $\mathbf{a}_{\ell,i,j,\tau} \xleftarrow{\$} \mathbb{Z}_q^n$, $e_{\ell,i,j,\tau} \xleftarrow{\$} \chi$ (recall that, according to our notational convention, $\mathbf{s}_{\ell-1}[0] \triangleq 1$). We define $\Psi \triangleq \{\psi_{\ell,i,j,\tau}\}_{\ell,i,j,\tau}$ to be the set of all these values.[14] At this point, it may not yet be clear what is the purpose of the $2^\tau$ factors; indeed, this will be explained later when we explain homomorphic multiplication.

  The key-generation algorithm proceeds to choose a uniformly random matrix $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{m \times n}$ and a vector $\mathbf{e} \xleftarrow{\$} \chi^m$, and compute $\mathbf{b} := \mathbf{A}\mathbf{s}_0 + 2\mathbf{e}$.

[13] It is quite straightforward to generalize the scheme to work over a message space GF($t$), where $t$ is relatively prime to $q$. Since we mostly care about the binary case, we choose not to present this generalization.

[14] A knowledgeable reader may notice that the above is similar to encryptions of $2^\tau \cdot \mathbf{s}_{\ell-1}[i] \cdot \mathbf{s}_{\ell-1}[j] \pmod q$ via an LWE-based scheme, except this "ciphertext" is not decryptable since the "message" is not a single bit value.

It then outputs the secret key $sk = \mathbf{s}_L$, the evaluation key $evk = \Psi$, and the public key $pk = (\mathbf{A}, \mathbf{b})$ (the public key $pk$ is essentially identical to the public key in Regev's scheme).

- Encryption $\mathsf{SH.Enc}_{pk}(\mu)$: Recall that $pk = (\mathbf{A}, \mathbf{b})$. To encrypt a message $\mu \in \mathrm{GF}(2)$, sample a vector $\mathbf{r} \xleftarrow{\$} \{0,1\}^m$ and set (just like in Regev's scheme)

$$\mathbf{v} := \mathbf{A}^T\mathbf{r} \qquad \text{and} \qquad w := \mathbf{b}^T\mathbf{r} + \mu \ .$$

  the encryption algorithm outputs $c := ((\mathbf{v}, w), 0)$. The output ciphertext $c$ contains the pair $(\mathbf{v}, w)$, in addition to a "level tag" which is used during homomorphic evaluation and indicates the "multiplicative depth" in which the ciphertext has been generated. For a freshly encrypted ciphertext, therefore, the level tag is zero.

- Homomorphic evaluation $\mathsf{SH.Eval}_{evk}(f, c_1, \ldots, c_t)$ where $f : \{0,1\}^t \to \{0,1\}$: We require that $f$ is represented by a binary arithmetic circuit with '+' gates of arbitrary fan-in and '$\times$' gates with fan-in 2. We further require that the circuit is *layered*, namely that it is composed of homogenous layers of either all '+' gates or all '$\times$' gates (it is easy to see that any arithmetic circuit can be converted to this form). Lastly, we require that the multiplicative depth of the circuit (the total number of '$\times$' layers) is exactly $L$. (Jumping ahead, additional restrictions on the circuit structure will be imposed when we turn to prove homomorphism, see Section 2.4.4.)

  We homomorphically evaluate the circuit $f$ gate by gate. Namely, we will show how to perform homomorphic addition (of arbitrarily many ciphertexts) and homomorphic multiplication (of two ciphertexts). Combining the two, we will be able to evaluate any such function $f$.

**Ciphertext structure during evaluation.** During the homomorphic evaluation, we will generate ciphertexts of the form $c = ((\mathbf{v}, w), \ell)$, where the tag $\ell$ indicates the multiplicative level at which the ciphertext has been generated (as we saw, fresh ciphertexts are tagged with 0). The requirement that $f$ is layered will make sure that throughout the homomorphic evaluation, all of the inputs to a gate have the same tag. In addition, we will keep the invariant that the output of each gate evaluation: $c = ((\mathbf{v}, w), \ell)$, is such that

$$w - \langle \mathbf{v}, \mathbf{s}_\ell \rangle = \mu + 2 \cdot e \pmod q \ , \tag{2.2}$$

where $\mu$ is the correct plaintext output of the gate, and $e$ is a noise term that depends on the gate's input ciphertexts.

Note that all ciphertexts will have tag $\ell \le L$ due to the bound on the multiplicative depth, and that the output of the homomorphic evaluation of the entire circuit is expected to have tag $\ell = L$.

**Homomorphic evaluation of gates:**

– *Addition gates.* Homomorphic evaluation of a '+' gate on inputs $c_1, \ldots, c_t$, where $c_i = ((\mathbf{v}_i, w_i), \ell)$, is performed by outputting

$$c_{\mathsf{add}} = ((\mathbf{v}_{\mathsf{add}}, w_{\mathsf{add}}), \ell) := \left( \left( \sum_i \mathbf{v}_i, \sum_i w_i \right), \ell \right) \ .$$

Informally, one can see that

$$w_{\mathsf{add}} - \langle \mathbf{v}_{\mathsf{add}}, \mathbf{s}_\ell \rangle = \sum_i (w_i - \langle \mathbf{v}_i, \mathbf{s}_\ell \rangle) = \sum_i (\mu_i + 2e_i) = \sum_i \mu_i + 2\sum_i e_i \ ,$$

where $\mu_i$ is the plaintext corresponding to $\mu_i$. The output of the homomorphic evaluation, thus, corresponds to the sum of the inputs, with the noise term being the sum of input noises.

– *Multiplication gates.* We show how to multiply ciphertexts $c, c'$ where $c = ((\mathbf{v}, w), \ell)$ and $c' = ((\mathbf{v}', w'), \ell)$ (recall that multiplication gates have fan-in 2), to obtain an output ciphertext $c_{\mathsf{mult}} = ((\mathbf{v}_{\mathsf{mult}}, w_{\mathsf{mult}}), \ell + 1)$. Note that the level tag increases by 1.

We first consider an $n$-variate *symbolic* polynomial over the unknown vector $\mathbf{x}$:

$$\phi(\mathbf{x}) = \phi_{(w,\mathbf{v}),(w',\mathbf{v}')}(\mathbf{x}) \triangleq (w - \langle \mathbf{v}, \mathbf{x} \rangle) \cdot (w' - \langle \mathbf{v}', \mathbf{x} \rangle) \ . \tag{2.3}$$

We symbolically open the parenthesis of this quadratic polynomial, and express it as

$$\phi(\mathbf{x}) = \sum_{0 \le i \le j \le n} h_{i,j} \cdot \mathbf{x}[i] \cdot \mathbf{x}[j] \ ,$$

where $h_{i,j} \in \mathbb{Z}_q$ are known (we can compute them from $(\mathbf{v}, w), (\mathbf{v}', w')$ by opening parenthesis in Eq. (2.3)).[15]

For technical reasons (related to keeping the error growth under control), we want to express $\phi(\cdot)$ as a polynomial with small coefficients. We consider the binary representation of $h_{i,j}$, letting $h_{i,j,\tau}$ be the $\tau^{\text{th}}$ bit in this representation. In other words

$$h_{i,j} = \sum_{\tau=0}^{\lfloor \log q \rfloor} h_{i,j,\tau} \cdot 2^\tau \ ,$$

for $h_{i,j,\tau} \in \{0, 1\}$.

We can express $\phi$ therefore as

$$\phi(\mathbf{x}) = \sum_{\substack{0 \le i \le j \le n \\ \tau \in \{0,\ldots,\lfloor \log q \rfloor\}}} h_{i,j,\tau} \cdot \left(2^\tau \cdot \mathbf{x}[i] \cdot \mathbf{x}[j]\right) \ ,$$

which can be interpreted as a polynomial with small coefficients in variables $(2^\tau \cdot \mathbf{x}[i] \cdot \mathbf{x}[j])$.

We recall that the evaluation key $evk = \Psi$ contains elements of the form $\psi_{\ell,i,j,\tau} = (\mathbf{a}_{\ell,i,j,\tau}, b_{\ell,i,j,\tau})$ such that

$$2^\tau \mathbf{s}_\ell[i]\mathbf{s}_\ell[j] \approx b_{\ell+1,i,j,\tau} - \langle \mathbf{a}_{\ell+1,i,j,\tau}, \mathbf{s}_{\ell+1} \rangle \ .$$

The homomorphic multiplication algorithm will thus set

$$\mathbf{v}_{\mathsf{mult}} := \sum_{\substack{0 \le i \le j \le n \\ \tau \in \{0,\ldots,\lfloor \log q \rfloor\}}} h_{i,j,\tau} \cdot \mathbf{a}_{\ell+1,i,j,\tau} \ ,$$

[15] We once again remind the reader that because of the notational trick of setting $\mathbf{x}[0] \triangleq 1$, this expression captures the constant term in the product, as well as all the linear terms, thus homogenizing the polynomial $\phi(\mathbf{x})$.

and

$$w_{\mathsf{mult}} = \sum_{\substack{0 \le i \le j \le n \\ \tau \in \{0,\ldots,\lfloor \log q \rfloor\}}} h_{i,j,\tau} \cdot b_{\ell+1,i,j,\tau} \ .$$

The final output ciphertext will be

$$c_{\mathsf{mult}} := ((\mathbf{v}_{\mathsf{mult}}, w_{\mathsf{mult}}), \ell + 1) \ .$$

Note that the level tag is increased by one as expected. Let us now verify that our invariant as per Eq. (2.2) still holds for the new ciphertext:

$$\begin{aligned} w_{\mathsf{mult}} - \langle \mathbf{v}_{\mathsf{mult}}, \mathbf{s}_{\ell+1} \rangle &= \sum_{\substack{0 \le i \le j \le n \\ \tau \in \{0,\ldots,\lfloor \log q \rfloor\}}} h_{i,j,\tau} \cdot (b_{\ell+1,i,j,\tau} - \langle \mathbf{a}_{\ell+1,i,j,\tau}, \mathbf{s}_{\ell+1} \rangle) \\ &= \sum_{\substack{0 \le i \le j \le n \\ \tau \in \{0,\ldots,\lfloor \log q \rfloor\}}} h_{i,j,\tau} \cdot 2^{\tau} \cdot \mathbf{s}_\ell[i] \cdot \mathbf{s}_\ell[j] + 2 \cdot h_{i,j,\tau} \cdot e_{\ell+1,i,j,\tau} \\ &= \phi(\mathbf{s}_\ell) + \sum_{\substack{0 \le i \le j \le n \\ \tau \in \{0,\ldots,\lfloor \log q \rfloor\}}} 2 \cdot h_{i,j,\tau} \cdot e_{\ell+1,i,j,\tau} \\ &= (w - \langle \mathbf{v}, \mathbf{s}_\ell \rangle) \cdot (w' - \langle \mathbf{v}', \mathbf{s}_\ell \rangle) + \sum_{\substack{0 \le i \le j \le n \\ \tau \in \{0,\ldots,\lfloor \log q \rfloor\}}} 2 \cdot h_{i,j,\tau} \cdot e_{\ell+1,i,j,\tau} \\ &= (\mu + 2e)(\mu' + 2e') + \sum_{\substack{0 \le i \le j \le n \\ \tau \in \{0,\ldots,\lfloor \log q \rfloor\}}} 2 \cdot h_{i,j,\tau} \cdot e_{\ell+1,i,j,\tau} \\ &= \mu\mu' + 2 \left( \mu e' + \mu' e + 2ee' + \sum_{\substack{0 \le i \le j \le n \\ \tau \in \{0,\ldots,\lfloor \log q \rfloor\}}} h_{i,j,\tau} \cdot e_{\ell+1,i,j,\tau} \right) . \end{aligned} \tag{2.4}$$

Indeed, we get the plaintext output $\mu\mu'$, in addition to a noise term that is inherited from the input ciphertexts and from the evaluation key.

- Decryption $\mathsf{SH.Dec}_{\mathbf{s}_L}(c)$: Recall that we are only required to decrypt ciphertexts that are output by $\mathsf{SH.Eval}(\cdots)$ and those will always have level tag $L$. To decrypt a ciphertext $c = ((\mathbf{v}, w), L)$, compute

$$(w - \langle \mathbf{v}, \mathbf{s}_L \rangle \pmod{q}) \pmod{2} \ . \tag{2.5}$$

### 2.4.2 The Scheme BTS: A Bootstrappable Scheme

We now utilize the dimension-modulus reduction technique to present the scheme BTS, which uses SH as a building block. BTS inherits its homomorphic properties from SH, but has much shorter ciphertexts and lower decryption complexity. These properties will enable us to apply the bootstrapping theorem to obtain full homomorphism.

Our bootstrappable scheme is parameterized by $(n, m, q, \chi, L)$, which are the parameters for $\mathsf{SH}$, and additional parameters $(k, p, \hat{\chi})$. We refer to $n, q \in \mathbb{N}$ as the "long" dimension and modulus respectively, while $k, p$ are the "short" dimension and modulus. $\chi, \hat{\chi}$ are the long and short noise distributions, respectively, over $\mathbb{Z}$. The parameter $m \in \mathbb{N}$ is used towards public key generation. The parameter $L$ is an upper bound on the multiplicative depth of the evaluated function.

While we discuss parameter values below, we encourage the reader to consider the following (non-optimal, but easier to understand) settings as a running example: $k = \kappa$, $n = k^4$, $q \approx 2^{\sqrt{n}}$, $L = 1/3 \log n = 4/3 \log k$, $p = (n^2 \log q) \cdot \mathrm{poly}(k) = \mathrm{poly}(k)$, $m = O(n \log q)$. The distributions $\chi, \hat{\chi}$ can be thought of as being $n$- and $k$-bounded, respectively.

- Key generation $\mathsf{BTS.Keygen}(1^\kappa)$: Run $\mathsf{SH.Keygen}(1^\kappa)$ to obtain the secret key $\mathbf{s}_L$, evaluation key $\Psi$ and public key $(\mathbf{A}, \mathbf{b})$ of $\mathsf{SH}$.

  Recall that $\mathbf{s}_L \in \mathbb{Z}_q^n$, $(\mathbf{A}, \mathbf{b}) \in \mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m$, and $\Psi \in (\mathbb{Z}_q^n \times \mathbb{Z}_q)^{(n+1)^2 \cdot (\lfloor \log q \rfloor + 1) \cdot L}$.

  Proceed by sampling the "short" secret key $\hat{\mathbf{s}} \xleftarrow{\$} \mathbb{Z}_p^k$ and compute the following additional parameters for the evaluation key: For all $i \in [n]$, $\tau \in \{0, \ldots, \lfloor \log q \rfloor\}$, sample $\hat{\mathbf{a}}_{i,\tau} \xleftarrow{\$} \mathbb{Z}_p^k$, $\hat{e}_{i,\tau} \xleftarrow{\$} \hat{\chi}$, and compute

  $$\hat{b}_{i,\tau} := \langle \hat{\mathbf{a}}_{i,\tau}, \hat{\mathbf{s}} \rangle + \hat{e}_{i,\tau} + \left\lfloor \frac{p}{q} \cdot \left(2^\tau \cdot \mathbf{s}_L[i]\right) \right\rceil \pmod{p} .$$

  Set $\hat{\psi}_{i,\tau} := \left(\hat{\mathbf{a}}_{i,\tau}, \hat{b}_{i,\tau}\right) \in \mathbb{Z}_p^k \times \mathbb{Z}_p$, and

  $$\hat{\Psi} := \{\hat{\psi}_{i,\tau}\}_{i \in [n], \tau \in \{0, \ldots, \lfloor \log q \rfloor\}} .$$

  This is very similar to the generation of $\Psi$ in the scheme $\mathsf{SH}$, but now $\hat{\psi}_{i,\tau}$ "encodes" scaled linear terms, rather than quadratic terms.

  Finally, output the secret key $sk = \hat{\mathbf{s}}$, evaluation key $evk = (\Psi, \hat{\Psi})$, and public key $pk = (\mathbf{A}, \mathbf{b})$. Note that the public key is identical to that of $\mathsf{SH}$.

- Encryption $\mathsf{BTS.Enc}_{pk}(\mu)$: Use the same encryption algorithm as $\mathsf{SH}$. To encrypt a bit $\mu \in \{0, 1\}$, compute $c \leftarrow \mathsf{SH.Enc}_{(\mathbf{A}, \mathbf{b})}(\mu)$ and output $c$ as the ciphertext.

- Homomorphic evaluation $\mathsf{BTS.Eval}_{evk}(f, c_1, \ldots, c_t)$, where $f : \{0, 1\}^t \to \{0, 1\}$: Recall that $evk = (\Psi, \hat{\Psi})$. To perform homomorphic evaluation, we will use the homomorphic evaluation function of $\mathsf{SH}$. We thus require that $f$ is represented by a binary arithmetic circuit which is a legal input for $\mathsf{SH.Eval}$.

  The first step in the homomorphic evaluation is computing

  $$c_f \leftarrow \mathsf{SH.Eval}_\Psi(f, c_1, \ldots, c_t) .$$

  This results in a ciphertext of the form $c_f = ((\mathbf{v}, w), L) \in \mathbb{Z}_q^n \times \mathbb{Z}_q \times \{L\}$.

Next, we reduce the dimension and modulus of $c_f$ to $k, p$ as follows. Consider the following function from $\mathbb{Z}^n$ into the rationals modulo $p$

$$\phi(\mathbf{x}) \triangleq \phi_{\mathbf{v},w}(\mathbf{x}) \triangleq \frac{p}{q} \cdot \left( \frac{q+1}{2} \cdot (w - \langle \mathbf{v}, \mathbf{x} \rangle) \right) \pmod{p} .$$

Rearranging, one can find $h_0, \ldots, h_n \in \mathbb{Z}_q$ such that

$$\phi(\mathbf{x}) = \sum_{i=0}^{n} h_i \cdot (\frac{p}{q} \cdot \mathbf{x}[i]) \pmod{p} .$$

Let $h_{i,\tau}$ be the $\tau^{\text{th}}$ bit of $h_i$, for all $\tau \in \{0, \ldots, \lfloor \log q \rfloor\}$. Then

$$\phi(\mathbf{x}) = \sum_{i=0}^{n} \sum_{\tau=0}^{\lfloor \log q \rfloor} h_{i,\tau} \cdot (\frac{p}{q} \cdot 2^\tau \cdot \mathbf{x}[i]) .$$

Using the parameters in $\hat{\Psi}$, we create a new ciphertext $\hat{c} = (\hat{\mathbf{v}}, \hat{w}) \in \mathbb{Z}_p^k \times \mathbb{Z}_p$ by setting

$$\begin{aligned} \hat{\mathbf{v}} &:= 2 \cdot \sum_{i=0}^{n} \sum_{\tau=0}^{\lfloor \log q \rfloor} h_{i,\tau} \cdot \hat{\mathbf{a}}_{i,\tau} \pmod{p} \in \mathbb{Z}_p^k \\ \hat{w} &:= 2 \cdot \sum_{i=0}^{n} \sum_{\tau=0}^{\lfloor \log q \rfloor} h_{i,\tau} \cdot \hat{b}_{i,\tau} \pmod{p} \in \mathbb{Z}_p . \end{aligned}$$

The output of BTS.Eval is the new ciphertext $\hat{c} \in \mathbb{Z}_p^k \times \mathbb{Z}_p$. Note that the bit-length of $\hat{c}$ is $(k+1)\log p$.

Recall the invariant we enforce on the structure of ciphertexts of SH (see Eq. 2.2). We show that a similar invariant holds for $\hat{c}$: Namely, that if $c_f$ is such that $w - \langle \mathbf{v}, \mathbf{s}_L \rangle = \mu + 2e \pmod{q}$, then

$$\hat{w} - \langle \hat{\mathbf{v}}, \hat{\mathbf{s}} \rangle = \mu + 2\hat{e} \pmod{p} ,$$

where $\hat{e}$ is proportional to $\frac{p}{q}e$ (an appropriately scaled version of $e$), plus some additional noise.

To see the above (in a somewhat indirect way, which will prove to be useful later), recall that $(p+1)/2$ is the inverse of 2 modulo $p$, and notice that

$$\begin{aligned} \frac{p+1}{2} (\hat{w} - \langle \hat{\mathbf{v}}, \hat{\mathbf{s}} \rangle) &= \sum_{i=0}^{n} \sum_{\tau=0}^{\lfloor \log q \rfloor} h_{i,\tau} \cdot \left( \hat{b}_{i,\tau} - \langle \hat{\mathbf{a}}_{i,\tau}, \hat{\mathbf{s}} \rangle \right) \pmod{p} \\ &= \sum_{i=0}^{n} \sum_{\tau=0}^{\lfloor \log q \rfloor} h_{i,\tau} \left( \hat{e}_{i,\tau} + \left\lfloor \frac{p}{q} \cdot \left(2^\tau \cdot \mathbf{s}_L[i]\right) \right\rceil \right) \pmod{p} \end{aligned}$$

$$= \phi(\mathbf{s}_L) + \underbrace{\sum_{i=0}^{n} \sum_{\tau=0}^{\lfloor \log q \rfloor} h_{i,\tau} \left(\hat{e}_{i,\tau} + \hat{\omega}_{i,\tau}\right)}_{\triangleq\ \delta_1} \pmod{p} \ , \tag{2.6}$$

where we define

$$\hat{\omega}_{i,\tau} \triangleq \left\lfloor \frac{p}{q} \cdot \left(2^\tau \cdot \mathbf{s}_L[i]\right) \right\rceil - \frac{p}{q} \cdot \left(2^\tau \cdot \mathbf{s}_L[i]\right) \ ,$$

and notice that $|\hat{\omega}_{i,\tau}| \leq 1/2$. Since $h_{i,\tau} \in \{0,1\}$ and since $\hat{e}_{i,\tau}$ is small, it follows that $\delta_1$ (defined in Eq. (2.6)) is "small" as well.

Now, letting $w = \langle \mathbf{v}, \mathbf{s}_L \rangle + 2e + \mu \pmod{q}$, we wish to examine $\phi(\mathbf{s}_L) \triangleq \phi_{(\mathbf{v},w)}(\mathbf{s}_L)$ more closely, as follows.

$$\begin{aligned}
\phi(\mathbf{s}_L) &\triangleq \frac{p}{q} \cdot \left(\frac{q+1}{2} \cdot (w - \langle \mathbf{v}, \mathbf{s}_L \rangle)\right) \pmod{p} \\
&= \frac{p}{q} \cdot \left(\frac{q+1}{2} \cdot (2e + \mu + Mq)\right) \pmod{p} \qquad \text{(where } M \in \mathbb{Z}\text{)} \\
&= \frac{p}{q} \cdot \left(\frac{q+1}{2}\mu + e + M'q\right) \pmod{p} \qquad \text{(where } M' = M + e \in \mathbb{Z}\text{)} \\
&= \frac{p}{q} \cdot \frac{q+1}{2}\mu + \frac{p}{q} \cdot e \pmod{p} \\
&= \frac{p+1}{2} \cdot \mu + \underbrace{(\frac{p}{q} - 1) \cdot \frac{\mu}{2} + \frac{p}{q} \cdot e}_{\triangleq \delta_2} \pmod{p} \\
&= \frac{p+1}{2} \cdot \mu + \delta_2 \ .
\end{aligned} \tag{2.7}$$

Notice that if $p \leq q$ (as is the case in our setting), $|\delta_2| \leq \frac{p}{q}|e| + \frac{1}{2}$.

Putting together Eq. (2.6) and (2.7), we see that

$$\frac{p+1}{2}\left(\hat{w} - \langle \hat{\mathbf{v}}, \hat{\mathbf{s}} \rangle\right) = \frac{p+1}{2} \cdot \mu + (\delta_1 + \delta_2) \pmod{p} \ . \tag{2.8}$$

Multiplying by 2, we have

$$\hat{w} - \langle \hat{\mathbf{v}}, \hat{\mathbf{s}} \rangle = \mu + 2(\delta_1 + \delta_2) \pmod{p} \ . \tag{2.9}$$

Now defining $\hat{e} \triangleq \delta_1 + \delta_2$, the invariant follows.

It is important to notice that, while not immediate from its definition, $\hat{e} = \delta_1 + \delta_2$ is an integer. To see this, note that it can be represented as a difference between integers:

$$\delta_1 + \delta_2 = \frac{p+1}{2}(\hat{w} - \langle \hat{\mathbf{v}}, \hat{\mathbf{s}} \rangle) - \frac{p+1}{2} \cdot \mu \ .$$

- Decryption $\mathsf{BTS.Dec}_{\hat{\mathbf{s}}}(\hat{c})$: To decrypt $\hat{c} = (\hat{\mathbf{v}}, \hat{w}) \in \mathbb{Z}_p^k \times \mathbb{Z}_p$ (recall, again, that we only need to decrypt ciphertexts that are output by $\mathsf{BTS.Eval}$), compute

$$\mu^* := (\hat{w} - \langle \hat{\mathbf{v}}, \hat{\mathbf{s}} \rangle \pmod p) \pmod 2 \ .$$

  If indeed $\hat{w} - \langle \hat{\mathbf{v}}, \hat{\mathbf{s}} \rangle = \mu + 2\hat{e} \pmod p$, and if $\hat{e}$ is small enough, then we get $\mu^* = \mu$.

### 2.4.3 Security Analysis

In this section, we analyze the security of SH and BTS based on LWE. Then, using known connections, we base security on worst case hardness of lattice problems.

The following theorem asserts the security of SH and BTS based on two DLWE problems: One with modulus $q$, dimension $n$ and noise $\chi$, and one with modulus $p$, dimension $k$ and noise $\hat{\chi}$. The security of SH follows based only on the former while the security of BTS requires both.

**Theorem 2.4.1** (security)**.** *Let $n = n(\kappa), k = k(\kappa), q = q(\kappa), p = p(\kappa)$ and $L = L(\kappa)$ be functions of the security parameter. Let $\chi, \hat{\chi}$ be some distributions over the integers, and define $m \triangleq (n+1)\log q + 2\kappa$. Then the following hold:*

1. *The scheme SH is CPA secure under the $\mathsf{DLWE}_{n,q,\chi}$ assumption. In particular, if the $\mathsf{DLWE}_{n,q,\chi}$ problem is $(t, \epsilon)$-hard, then the scheme is $(t - \mathrm{poly}(\kappa), 2L \cdot (2^{-\kappa} + \epsilon))$-semantically secure.*

2. *The scheme BTS is CPA secure under the $\mathsf{DLWE}_{n,q,\chi}$ and the $\mathsf{DLWE}_{k,p,\hat{\chi}}$ assumptions. In particular, if both the $\mathsf{DLWE}_{n,q,\chi}$ and the $\mathsf{DLWE}_{k,p,\hat{\chi}}$ problems are $(t, \epsilon)$-hard, then the scheme is $(t - \mathrm{poly}(\kappa), 2(L+1) \cdot (2^{-\kappa} + \epsilon))$-semantically secure.*

Essentially, the view of a CPA adversary for our scheme is very similar to Regev's scheme, with the exception that our adversary also gets to see the evaluation key. However, the evaluation key contains a sequence of LWE instances which, based on our assumption, are indistinguishable from uniform. Therefore our reduction, for the scheme SH, will perform a sequence of $L$ hybrids to replace the $\Psi$ component of the evaluation key with a set of completely uniform elements. For the scheme BTS, an additional hybrid will imply the same for $\hat{\Psi}$. Once this is done, we will use the known proof techniques from Regev's scheme and get the security of our scheme. A formal proof follows.

*Proof.* We first prove CPA security for BTS. CPA security for SH will then follow as a special case of our argument.

Let $\mathcal{A}$ be an IND-CPA adversary for BTS that runs in time $t$. We consider a series of hybrids where $\mathrm{Adv}_H[\mathcal{A}]$ denotes the success probability of $\mathcal{A}$ in hybrid $H$.

- **Hybrid $\hat{H}_{L+1}$:** This is the identical to the IND-CPA game, where the adversary gets properly distributed keys $pk, evk$, generated by $\mathsf{BTS.Keygen}$, and an encryption of either 0 or 1 computed using $\mathsf{BTS.Enc}$. By definition,

$$\mathrm{Adv}_{\hat{H}_{L+1}}[\mathcal{A}] \triangleq \left| \Pr[\mathcal{A}(pk, \mathsf{SH.Enc}_{pk}(\mu_0) = 1] - \Pr[\mathcal{A}(pk, \mathsf{SH.Enc}_{pk}(\mu_1) = 1] \right| = \mathrm{Adv}_{\mathrm{CPA},\mathsf{BTS}}[\mathcal{A}] \ .$$

- **Hybrid $H_{L+1}$:** This hybrid is identical to $\hat{H}_{L+1}$ in everything except the generation of $\hat{\Psi}$. In this hybrid, $\hat{\Psi}$ is not generated as prescribed, but is rather sampled uniformly. Namely, for all $i, \tau$ we set $\hat{\psi}_{i,\tau} \xleftarrow{\$} \mathbb{Z}_p^k \times \mathbb{Z}_p$.

  It follows that there exists an adversary $\widehat{\mathcal{B}}$ that solves the $\mathsf{DLWE}_{k,p,\hat{\chi}}$ problem in time $t + \mathrm{poly}(\kappa)$ and advantage

  $$\mathsf{DLWE}_{k,p,\hat{\chi}}\mathrm{Adv}[\widehat{\mathcal{B}}] \geq 1/2 \cdot \left| \mathrm{Adv}_{\hat{H}_{\ell+1}}[\mathcal{A}] - \mathrm{Adv}_{H_{\ell+1}}[\mathcal{A}] \right| \ .$$

  The adversary $\widehat{\mathcal{B}}$ will sample all vectors $\mathbf{s}_0, \ldots, \mathbf{s}_L$ by himself and generate $pk, \Psi$. Then, he will use the $\mathsf{LWE}$ oracle to obtain either $A_{\hat{\mathbf{s}},\hat{\chi}}$ samples, which will result in properly generated $\hat{\Psi}$, or uniform samples which will result in a uniform $\hat{\Psi}$. $\mathcal{B}$ will then sample a uniform $b \xleftarrow{\$} \{0, 1\}$ and return 1 if and only if $\mathcal{A}(pk, (\Psi, \hat{\Psi}), \mathsf{BTS.Enc}_{pk}(\mu_b)) = b$. Using simple algebra, the result follows.

- **Hybrid $H_\ell$, for $\ell \in [L]$:** Hybrid $H_\ell$ is identical to $H_{\ell+1}$, except for a change in some of the elements in $\Psi$. Specifically, we change the components $\psi_{\ell,i,j,\tau}$, for all $i, j, \tau$: Instead of computing $\psi_{\ell,i,j,\tau}$ as prescribed (i.e., $(\mathbf{a}_{\ell,i,j,\tau}, \langle \mathbf{a}_{\ell,i,j,\tau}, \mathbf{s}_\ell \rangle + 2e_{\ell,i,j,\tau} + 2^\tau \cdot \mathbf{s}_{\ell-1}[i] \cdot \mathbf{s}_{\ell-1}[j])$), we set $\psi_{\ell,i,j,\tau} \xleftarrow{\$} \mathbb{Z}_q^n \times \mathbb{Z}_q$ (namely, a uniform sample).

  It follows that there exists an adversary $\mathcal{B}_\ell$ that solves the $\mathsf{DLWE}_{n,q,\chi}$ problem in time $t + \mathrm{poly}(\kappa)$ with advantage

  $$\mathsf{DLWE}_{n,q,\chi}\mathrm{Adv}[\mathcal{B}_\ell] = 1/2 \cdot \left| \mathrm{Adv}_{H_\ell}[\mathcal{A}] - \mathrm{Adv}_{H_{\ell+1}}[\mathcal{A}] \right| \ .$$

  The argument is very similar to the previous hybrid: We note that at this point, $\hat{\Psi}$ and $\{\psi_{\lambda,i,j,\tau}\}_{\lambda > \ell, i,j,\tau}$ are completely uniform and can be generated without any knowledge of $\mathbf{s}_{\ell+1}, \ldots, \mathbf{s}_L, \hat{\mathbf{s}}$. The adversary $\mathcal{B}_\ell$ will sample all vectors $\mathbf{s}_0, \ldots, \mathbf{s}_{\ell-1}$ himself, and turn to the $\mathsf{LWE}$ oracle for samples in order to generate $\psi_{\ell,i,j,\tau}$. This will result in $\Psi$ being identical to $H_{\ell+1}$ if the oracle returns $A_{\mathbf{s},\chi}$ samples, or $\Psi$ being identical to $H_\ell$ if the oracle returns uniform elements. Once again, sampling a random $b$ and checking whether $\mathcal{A}$'s response is identical to $b$ completes the argument.

  Note that in the hybrid $H_1$, the evaluation key $evk = (\Psi, \hat{\Psi})$ is completely uniform, and hence the view of the adversary is like in Regev's scheme.

- **Hybrid $H_0$:** Hybrid $H_0$ is identical to $H_1$ except that the vector $\mathbf{b}$ in the public key is chosen uniformly at random from $\mathbb{Z}_q^m$, rather than being computed as $\mathbf{A} \cdot \mathbf{s}_0 + 2\mathbf{e}$. Under the $\mathsf{DLWE}_{n,q,\chi}$ assumption, hybrids $H_0$ and $H_1$ are indistinguishable. Namely, there exists an adversary $\mathcal{B}_0$ that runs in time $t + \mathrm{poly}(\kappa)$ and whose advantage is

  $$\mathsf{DLWE}_{n,q,\chi}\mathrm{Adv}[\mathcal{B}_0] = 1/2 \cdot \left| \mathrm{Adv}_{H_1}[\mathcal{A}] - \mathrm{Adv}_{H_0}[\mathcal{A}] \right| \ .$$

  The adversary $\mathcal{B}_0$ gets $m$ samples from the $\mathsf{LWE}$ oracle and uses them to generate $(\mathbf{A}, \mathbf{b})$. If the samples come from $A_{\mathbf{s},\chi}$, then $\mathbf{b}$ is distributed like in $H_1$, but if they are uniform then $\mathbf{b}$ is distributed like in $H_0$. The same testing of $\mathcal{A}$'s success as before implies the argument.

- **Hybrid $H_{\mathsf{rand}}$:** Hybrid $H_{\mathsf{rand}}$ is identical to $H_0$ except that the ciphertext is chosen uniformly at random from $\mathbb{Z}_q^n \times \mathbb{Z}_q$, rather than being computed as $(\mathbf{A}^T \cdot \mathbf{r}, \mathbf{b}^T \cdot \mathbf{r} + \mu)$.

  We now claim that
  $$|\mathrm{Adv}_{H_0}[\mathcal{A}] - \mathrm{Adv}_{H_{\mathsf{rand}}}[\mathcal{A}]| \le 2^{-\kappa} \ .$$
  This is due to the Leftover hash lemma (Lemma 2.2.1), since $m > (n+1)\log q + 2\kappa$.

Note that in $H_{\mathsf{rand}}$, all the elements of both the public key and the ciphertext are uniformly random and independent of the message. Thus,
$$\mathrm{Adv}_{H_{\mathsf{rand}}}[\mathcal{A}] = 0 \ .$$

Putting these together, we get that
$$\mathrm{Adv}_{\mathrm{CPA},\mathsf{BTS}}[\mathcal{A}] \le 2^{-\kappa} + 2 \cdot \left( \mathsf{DLWE}_{k,p,\hat{\chi}}\mathrm{Adv}[\widehat{\mathcal{B}}] + \sum_{\ell=0}^{L} \mathsf{DLWE}_{n,q,\chi}\mathrm{Adv}[\mathcal{B}_\ell] \right) \ ,$$
and the security of $\mathsf{BTS}$ follows.

To finish the proof, we notice that the view of $\mathcal{A}$ in hybrid $H_{L+1}$ is equivalent to the view of an adversary in the IND-CPA game for $\mathsf{SH}$ (since $\hat{\Psi}$ is replaced by a uniform sequence). Therefore for every adversary $\mathcal{A}$,
$$\mathrm{Adv}_{\mathrm{CPA},\mathsf{SH}}[\mathcal{A}] \le 2^{-\kappa} + 2 \cdot \sum_{\ell=0}^{L} \mathsf{DLWE}_{n,q,\chi}\mathrm{Adv}[\mathcal{B}_\ell] \ ,$$
and the security of $\mathsf{SH}$ follows. □

**Specific Parameters and Worst-Case Hardness.** The parameters we require for homomorphism (see Theorem 2.4.2 below) are as follows. We require that $q = 2^{n^\epsilon}$ for some $\epsilon \in (0,1)$, $\chi$ is $n$-bounded, $p = 16nk\log(2q)$ and $\hat{\chi}$ is $k$-bounded. In order to achieve the best lattice reduction, we will choose $q$ as a product of polynomially bounded co-prime numbers. Applying known results (see Corollary 2.2.2), $\mathsf{DLWE}_{n,q,\chi}$ translates into approximating short-vector problems in worst case $n$-dimensional lattices to within a factor of $\widetilde{O}\left(\sqrt{n} \cdot 2^{n^\epsilon}\right)$, while $\mathsf{DLWE}_{k,p,\hat{\chi}}$ translates to approximating $k$-dimensional lattice problems to within $\widetilde{O}\left(n^{1+\epsilon} \cdot k^{1.5}\right)$ factor.[16] These problems are essentially incomparable as the hardness of the problem increases as the dimension increases on one hand, but decreases as the approximation factor increases on the other. The best known algorithms solve the first problem in time (roughly) $2^{\widetilde{O}(n^{1-\epsilon})}$, and the second in time $2^{\widetilde{O}(k)}$.

The relation between $n$ and $k$ is determined based on the required homomorphic properties. In this work, we only prove there there exists a constant $C$ such that setting $n = k^{C/\epsilon}$ implies fully homomorphic encryption. Given the value of $C$, setting $\epsilon \approx 1 - \frac{1}{C+1}$ will make the two problems equally hard (at least based on the current state of the art).

[16]We do not mention the specific lattice problem or the specific type of reduction (quantum vs. classical) since, as one can observe from Corollary 2.2.2, the approximation factor we get is essentially the same for all problems, and the state of the art is roughly the same as well.

### 2.4.4 Homomorphic Properties of SH And BTS

In this section, we analyze the homomorphic properties of SH and BTS. Both schemes have essentially the same homomorphic properties, but BTS has the additional advantage of having low decryption complexity (as analyzed in Section 2.4.5). Thus, BTS would be our main focus, and the properties of SH will follow as a by-product of our analysis.

We start by formally defining the class of functions for which we prove homomorphism, and proceed by stating the homomorphic properties and proving them.

**The Function Class $\mathsf{Arith}[L,T]$.** We consider arithmetic circuits over GF(2) with bounded fan-in and bounded depth, and with an additional final "collation": a high fan-in addition gate at the last level. We require that the circuit is structured in a canonical "layered" manner as defined below.

**Definition 2.4.1.** *Let $L = L(\kappa), T = T(\kappa)$ be functions of the security parameter. The class $\mathsf{Arith}[L,T]$ is the class of arithmetic circuits over GF$(2)$, with $\{+,\times\}$ gates, with the following structure. Each circuit contains exactly $2L+1$ layers of gates (numbered $1,\dots,2L+1$ starting from the input level), gates of layer $i+1$ are fed only by gates of layer $i$. The odd layers contain only '$+$' gates and the even layers contain only '$\times$' gates. The gates at layers $1,\dots,2L$ have fan-in 2, while the final addition gate in layer $2L+1$ is allowed to have fan-in $T$.*

We note that $\mathsf{Arith}[L,T]$ conforms with the requirements on the evaluated function imposed by SH.Eval and BTS.Eval. Indeed, the multiplicative depth of any circuit in $\mathsf{Arith}[L,T]$ is exactly $L$, and hence, homomorphic evaluation is well defined on any such function.

To motivate the choice of this function class, we first note that any arithmetic circuit of fan-in 2 and depth $D$ can be trivially converted into a circuit in $\mathsf{Arith}[D,1]$.[17] This will be useful for the purpose of bootstrapping. Jumping ahead, the collation gate will be useful for constructing a private information retrieval protocol, where we will need to evaluate polynomials with a very large number of monomials and fairly low degree. The collation gate will thus be used for the final aggregation of monomials.

Our goal is now to prove that with the appropriate choice of parameters, SH and BTS are $\mathsf{Arith}[L,T]$-homomorphic.

**Theorem 2.4.2.** *Let $n = n(\kappa) \geq 5$ be any polynomial, $q \geq 2^{n^\epsilon} \geq 3$ for some $\epsilon \in (0,1)$ be odd, $\chi$ be any $n$-bounded distribution, and $m = (n+1)\log q + 2\kappa$. Let $k = \kappa$, $p = 16nk\log(2q)$ (odd) and $\hat{\chi}$ be any $k$-bounded distribution. Then SH and BTS are both $\mathsf{Arith}[L = \Omega(\epsilon \log n), T = \sqrt{q}]$-homomorphic.*

Not surprisingly, the homomorphism class depends only on $n$ and not on $k$. This is because, recalling the definition of BTS.Eval, the homomorphism property is inherited from SH.Eval. We note that it is possible to further generalize the class of circuits that we can homomorphically evaluate

[17] One way to do this is to separate each level of the original circuit into two levels – an addition level and a multiplication level – and finally, adding a dummy fan-in-1 addition gate at the top. This gives us a $2D+1$ depth circuit with alternating addition and multiplication levels, or, in other words, the transformed circuit belongs to $\mathsf{Arith}[D,1]$.

(for example, circuits with high multiplicative depth but low multiplicative degree), however since this is not required for our results, and since the proof will use the exact same tools, we choose not to further complicate the theorem statement and proof.

To prove the theorem, we introduce a sequence of lemmas as follows. Recall that the encryption algorithms of both schemes are identical, and that $\mathsf{BTS.Eval}$ first calls $\mathsf{SH.Eval}$ as a subroutine. We first analyze the growth of the noise in the execution of $\mathsf{SH.Eval}$ in Lemma 2.4.3 (which will imply the theorem for $\mathsf{SH}$), and then, in Lemma 2.4.4, we complete the noise calculation of $\mathsf{BTS.Eval}$, which will complete the proof of the theorem.

To track the growth of the noise, we define, for any ciphertext $c = ((\mathbf{v}, w), \ell)$ a noise measure $\eta(c) \in \mathbb{Z}$ as follows. We let $e \in \mathbb{Z}$ be the smallest integer (in absolute value) such that

$$\mu + 2e = w - \langle \mathbf{v}, \mathbf{s}_d \rangle \pmod{q} \ ,$$

and define $\eta(c) \triangleq \mu + 2e$ (note that $\eta(c)$ is defined over the integers, and not modulo $q$). We note that so long as $|\eta(c)| < q/2$, the ciphertext is decryptable. We can now bound the error in the execution by bounding $\eta(c_f)$ of the output ciphertext.

**Lemma 2.4.3.** *Let $n = n(\kappa) \geq 5$, $q = q(\kappa) \geq 3$, $\chi$ be $B$-bounded and $L = L(\kappa)$ and let $f \in \mathsf{Arith}[L, T]$, $f : \{0,1\}^t \to \{0,1\}$ (for some $t = t(\kappa)$). Then for any input $\mu_1, \ldots, \mu_t \in \{0,1\}$, if we let $(pk, evk, sk) \leftarrow \mathsf{SH.Keygen}(1^\kappa)$, $c_i \leftarrow \mathsf{BTS.Enc}_{pk}(\mu_i) = \mathsf{SH.Enc}_{pk}(\mu_i)$ and we further let $c_f = ((\mathbf{v}, w), L) \leftarrow \mathsf{SH.Eval}_{evk}(f, c_1, \ldots, c_t)$ be the encryption of $f(\mu_1, \ldots, \mu_t)$, it holds that with all but negligible probability*

$$|\eta(c_f)| \leq T \cdot (16nB \log q)^{2^L} \ .$$

*Proof.* We assume that all samples of $\chi$ (there are only polynomially many of them) are indeed of magnitude at most $B$. This happens with all but exponentially small probability. The remainder of the analysis is completely deterministic.

We track the growth of noise as the homomorphic evaluation proceeds.

- **Fresh ciphertexts.** Our starting point is level-0 ciphertexts $((\mathbf{v}, w), 0)$, that are generated by the encryption algorithm. By definition of the encryption algorithm we have that

  $$w - \langle \mathbf{v}, \mathbf{s}_0 \rangle = \mathbf{r}^T \cdot \mathbf{b} + \mu - \mathbf{r}^T \cdot \mathbf{A} \cdot \mathbf{s}_0 = \mu + \mathbf{r}^T \cdot (\mathbf{b} - \mathbf{A}\mathbf{s}_0) = \mu + 2\mathbf{r}^T \cdot \mathbf{e} \pmod{q} \ .$$

  Since $\left|\mu + 2\mathbf{r}^T \cdot \mathbf{e}\right| \leq 1 + 2nB$, it follows that

  $$|\eta(c)| \leq 2nB + 1 \ . \tag{2.10}$$

- **Homomorphic addition gates.** When evaluating '+' on ciphertexts $c_1, \ldots, c_t$ to obtain $c_{\mathsf{add}}$, we just sum their $(\mathbf{v}, w)$ values. Therefore

  $$|\eta(c_{\mathsf{add}})| \leq \sum_i |\eta(c_i)| \ .$$

- **Homomorphic multiplication gates.** When evaluating '$\times$' on $c = ((\mathbf{v}, w), \ell)$, $c' = ((\mathbf{v}', w'), \ell)$ to obtain $c_{\mathsf{mult}} = ((\mathbf{v}_{\mathsf{mult}}, w_{\mathsf{mult}}), \ell + 1)$, we get, by Eq. (2.4), that

$$w_{\mathsf{mult}} - \langle \mathbf{v}_{\mathsf{mult}}, \mathbf{s}_{\ell+1} \rangle = \eta(c) \cdot \eta(c') + 2 \sum_{\substack{0 \le i \le j \le n \\ \tau \in \{0, \ldots, \lfloor \log q \rfloor\}}} h_{i,j,\tau} \cdot e_{\ell+1,i,j,\tau} \pmod{q} .$$

It follows that

$$|\eta(c_{\mathsf{mult}})| \le |\eta(c)| \cdot \left|\eta(c')\right| + 2 \cdot \frac{(n+1)(n+2)}{2} \cdot (\log q + 1) \cdot B .$$

If we define

$$E \triangleq \max \left\{ |\eta(c)|, \left|\eta(c')\right|, (n+2)\sqrt{B \log(2q)} \right\} ,$$

then $|\eta(c_{\mathsf{mult}})| \le 2E^2$.

Putting all of the above together, let

$$E_0 = \max \left\{ 2nB + 1, (n+2)\sqrt{B \log(2q)} \right\} \le 2nB \log q$$

be an upper bound on $|\eta(c)|$ of fresh ciphertexts.

Then it holds that a bound $E_{2\ell}$ on $|\eta(c)|$ of the outputs of layer $2\ell$ (recall that the even layers contain multiplication gates) is obtained by

$$E_{2\ell} \le 2(2E_{2(\ell-1)})^2 .$$

Therefore, recursively,

$$E_{2L} \le (8E_0)^{2^L} \le (16nB \log q)^{2^L} .$$

And after the final collation gate it holds that

$$|\eta(c_f)| \le T \cdot (16nB \log q)^{2^L} . \qquad \square$$

We now analyze the noise in the ciphertext $\hat{c} = (\hat{\mathbf{v}}, \hat{w})$, produced by $\mathsf{BTS.Eval}(f, \cdots)$: Recall that by Eq. (2.9), $\hat{w} - \langle \hat{\mathbf{v}}, \hat{\mathbf{s}} \rangle = \mu + 2\hat{e} \pmod{p}$, where $\hat{e} = \delta_1 + \delta_2$ (defined in Eq. (2.6) and (2.7)). Therefore, in order for $\hat{c}$ to decrypt correctly, we want to show that $|\delta_1 + \delta_2| < p/4$. As a step in that direction, the following lemma bounds $|\delta_1 + \delta_2|$ in terms of $\eta(c_f)$.

**Lemma 2.4.4.** *Let $n = n(\kappa) \ge 5$, $q = q(\kappa) \ge 3$, $\chi$ be $B$-bounded and $L = L(\kappa)$. Let $p = p(\kappa)$, $k = k(\kappa)$ and $\hat{\chi}$ be $\hat{B}$-bounded. Consider a homomorphic evaluation $\hat{c} \leftarrow \mathsf{BTS.Eval}_{evk}(f, c_1, \ldots, c_t)$, and the terms $\delta_1, \delta_2$ defined in Eq. (2.6) and (2.7), respectively. Let $c_f \in \mathbb{Z}_q^n \times \mathbb{Z}_q \times \{L\}$ be the intermediate value returned by the call to* $\mathsf{SH.Eval}$. *Then with all but negligible probability*

$$|\delta_1 + \delta_2| \le \frac{p}{2q} |\eta(c_f)| + 2n\hat{B} \log(2q) .$$

*Proof.* We assume that all samples from $\hat{\chi}$ are indeed of magnitude at most $\hat{B}$. This happens with all but exponentially small probability.

By definition (recall that $\delta_1$, $\delta_2$ have been defined over the rationals), we have that

$$|\delta_1| = \left|\sum_{i=0}^{n} \sum_{\tau=0}^{\lfloor \log q \rfloor} h_{i,\tau} \left(\hat{e}_{i,\tau} + \hat{\omega}_{i,\tau}\right)\right| \le (n+1)\log(2q)(\hat{B}+1/2) \ ,$$

and

$$\begin{aligned} |\delta_2| &= \left|(\frac{p}{q}-1)\cdot\frac{\mu}{2}+\frac{p}{q}\cdot e\right| \\ &= \left|\frac{p}{q}\cdot\frac{\mu+2e}{2}-\frac{\mu}{2}\right| \\ &\le \frac{p}{2q}\left|\eta(c_f)\right|+1/2 \ . \end{aligned}$$

Adding the terms together, the result follows. $\square$

We can now finally prove Theorem 2.4.2.

*Proof of Theorem 2.4.2.* Let us consider the homomorphism claim about $\mathsf{BTS}$ (the argument for $\mathsf{SH}$ will follow as by-product): A sufficient condition for ciphertext $\hat{c} = (\hat{\mathbf{v}}, \hat{w})$ to decrypt correctly is that $\hat{e} < p/4$. By Lemma 2.4.4, it is sufficient to prove that

$$\frac{p}{2q}\left|\eta(c_f)\right| + 2n\hat{B}\log(2q) < p/4 \ .$$

Since it holds that $2n\hat{B}\log(2q) \le p/8$, it is sufficient to prove that

$$\left|\eta(c_f)\right| < q/4 \ .$$

We note that if we prove this, then it also follows that $c_f$ is decryptable, and hence the claim about the homomorphism of $\mathsf{SH}$ holds as well.

Plugging the bound on $|\eta(c_f)|$ from Lemma 2.4.3 into the above, we get

$$T\cdot(16nB\log q)^{2^L} < q/4 \ ,$$

and plugging in all the parameters and $T = \sqrt{q}$, we need

$$(16n^{2+\epsilon})^{2^L} < 2^{n^{\epsilon}/2}/4$$

which clearly holds for some $L = \Omega(\epsilon \log n)$. $\square$

### 2.4.5 Bootstrapping and Full Homomorphism

We now show how to apply Gentry's bootstrapping theorem (Theorems 2.3.1, 2.3.2) to achieve full homomorphism. In order to do this, we first need to bound the complexity of an augmented decryption circuit. Since our decryption is essentially a computation of inner product, we bound the complexity of this operation.

**Lemma 2.4.5.** *Let $(\hat{\mathbf{v}}, \hat{w}) \in \mathbb{Z}_p^k \times \mathbb{Z}_p$. There exists an arithmetic circuit with fan-in 2 gates and $O(\log k + \log\log p)$ depth, that on input $\hat{\mathbf{s}} \in \mathbb{Z}_p^k$ (in binary representation) computes*

$$(\hat{w} - \langle \hat{\mathbf{v}}, \hat{\mathbf{s}} \rangle \pmod{p}) \pmod{2} .$$

*Proof.* We let $\hat{\mathbf{s}}[i](j)$ denote the $j^{\text{th}}$ bit of the binary representation of $\hat{\mathbf{s}}[i] \in \mathbb{Z}_p$. We notice that

$$\begin{aligned} \hat{w} - \langle \hat{\mathbf{v}}, \hat{\mathbf{s}} \rangle &= \hat{w} - \sum_{i=1}^{k} \hat{\mathbf{s}}[i]\hat{\mathbf{v}}[i] \pmod{p} \\ &= \hat{w} - \sum_{i=1}^{k} \sum_{j=0}^{\lfloor \log p \rfloor} \hat{\mathbf{s}}[i](j) \cdot (2^j \cdot \hat{\mathbf{v}}[i]) \pmod{p} . \end{aligned}$$

Therefore computing $\hat{w} - \langle \hat{\mathbf{v}}, \hat{\mathbf{s}} \rangle \pmod{p}$ is equivalent to summing up $k(1 + \lfloor \log p \rfloor) + 1$ numbers in $\mathbb{Z}_p$, and then taking the result modulo $p$. The summation (over the integers) can be done in depth $O(\log k + \log\log p)$. In order to take modulo $p$, one needs to subtract, in parallel, all possible multiples of $p$ (there are at most $O(k \log p)$ options) and check if the result is in $\mathbb{Z}_p$. This requires depth $O(\log k + \log\log p)$ again. Then a selection tree of depth $O(\log k + \log\log p)$ is used to choose the correct result. Once this is done, outputting the least significant bit implements the final modulo 2 operation.

The total depth is thus $O(\log k + \log\log p)$ as required. □

We can now apply the bootstrapping theorem to obtain a fully homomorphic scheme.

**Lemma 2.4.6.** *There exists $C \in \mathbb{N}$ such that setting $n = k^{C/\epsilon}$ and the rest of the parameters as in Theorem 2.4.2,* BTS *is bootstrappable as per Definition 2.3.7.*

*Proof.* Lemma 2.4.5 guarantees that the decryption circuit is in $\mathsf{Arith}[O(\log k), 1]$ (note that $\log\log p = o(\log k)$). The augmented decryption circuits only adds 1 to the depth and therefore they are also in $\mathsf{Arith}[O(\log k), 1]$.

Theorem 2.4.2, on the other hand, guarantees homomorphism for any $\mathsf{Arith}[\Omega(\epsilon \log n), \sqrt{q}]$ function. Taking a large enough $C$, it will hold that $\mathsf{Arith}[O(\log k), 1] \subseteq \mathsf{Arith}[\Omega(\epsilon \log n), \sqrt{q}]$ and the lemma follows. □

Finally, we conclude that there exists an LWE based fully homomorphic encryption based on Theorem 2.4.1 and Lemma 2.4.6.

**Corollary 2.4.7.** *There exists a leveled fully homomorphic encryption based on the $\mathsf{DLWE}_{n,q,\chi}$ and $\mathsf{DLWE}_{k,p,\hat{\chi}}$ assumptions.*

*Furthermore, if* BTS *is weakly circular secure (see Definition 2.3.8), then there exists a fully homomorphic encryption based on the same assumptions.*

**Efficiency of the Scheme.** Interestingly, our scheme is comparable to *non-homomorphic* LWE based schemes (e.g. Regev's) in terms of encryption, decryption and ciphertext sizes. Namely, so long as one doesn't use the homomorphic properties of the scheme, she does not need to "pay" for it. To see why this is the case, we observe that our scheme's secret key has length $k \log p = O(\kappa \log \kappa)$ and the ciphertext length is $(k+1)\log p = O(\kappa \log \kappa)$. The decryption algorithm is essentially the same as Regev's. As far as encryption is concerned, it may seem more costly: The public key (as described above) contains $(n+1)((n+1)\log q + 2\kappa)\log q$ bits, and encryption requires performing operations over $\mathbb{Z}_q$. However, we note that one can think of sampling a public key $(\hat{\mathbf{A}}, \hat{\mathbf{b}})$ where $\hat{\mathbf{A}} \xleftarrow{\$} \mathbb{Z}_p^{m \times k}$, $\hat{\mathbf{b}} = \hat{\mathbf{A}}\hat{\mathbf{s}} + 2\hat{\mathbf{e}} \in \mathbb{Z}_p^m$ (where $m = ((k+1)\log p + 2\kappa))$. This will enable generating short ciphertexts that will be "bootstrapped up" during the homomorphic evaluation. If such short public key is used, then encryption also becomes comparable to Regev's scheme.

Homomorphic evaluation is where the high price is paid. the evaluation key has size $O(Ln^2 \log^2 q + n \log q \log p) = \widetilde{O}(n^{2+2\epsilon})$. Considering the fact that $n = \kappa^{C/\epsilon}$, this accumulates to a fairly long evaluation key, especially considering that in a leveled scheme, this size increases linearly with the depth of the circuit to be evaluated. The bright side, as we mention above, is that *evk* only needs to be known to the homomorphic evaluator and is not needed for encryption or decryption.

**Circuit Privacy.** A property that is sometimes desired in the context of fully homomorphic encryption is *circuit privacy.* A scheme is circuit private if the output of homomorphic evaluation reveals no information on the evaluated function (other than its output), even to a party that knows *sk*. Circuit privacy for our scheme can be achieved by adding more noise to the ciphertext $c_f$, right before applying dimension-modulus reduction. Similar techniques were used in previous schemes and thus we feel that a more elaborate discussion is unnecessary here.

## 2.5 LWE-Based Private Information Retrieval

In this section, we present a single-server private information retrieval (PIR) protocol with nearly optimal communication complexity. First, we present the definitions of PIR in Section 2.5.1. Then, in Section 2.5.2, we show a generic construction of PIR from somewhat homomorphic encryption. Finally, in Section 2.5.3, we instantiate the generic construction using our own scheme from Section 2.4 and analyze its parameters.

### 2.5.1 Definitions of Single Server PIR

We define single server private information retrieval in the public-key setting. In this setting, there is a public key (or public parameters) associated with the receiver (who holds the respective secret key). This public key is independent of the query and of the database, it can be generated and sent (or posted) before the interaction begins, and it may be used many times. Thus, the size of the public key is not counted towards communication complexity of the protocol. We formalize this by an efficient setup procedure that runs before the protocol starts and generate this public key.

Letting $\kappa$ be the security parameter and let $N \in \mathbb{N}$ be the database size, a PIR protocol PIR in the public-key setting is defined by a tuple of polynomial-time computable algorithms (PIR.Setup, PIR.Query, PIR.Response, PIR.Decode) as follows:

0. **Setup.** The protocol begins in an off-line setup phase that does not depend on the index to be queried nor on the contents of the database.

   The receiver runs the setup algorithm

   $$(params, setupstate) \leftarrow \mathsf{PIR.Setup}(1^\kappa) \ .$$

   It thus obtains a public set of parameters *params* (the aforementioned "public key") that is sent to the sender, and a secret state *setupstate* that is kept private.

   Once the setup phase is complete, the receiver and sender can run the remainder of the protocol an unbounded number of times.

1. **Query.** When the receiver wishes to receive the $i^{\text{th}}$ element in the database, $\mathtt{DB}[i]$, it runs

   $$(query, qstate) \leftarrow \mathsf{PIR.Query}(1^\kappa, setupstate, i) \ .$$

   The query message *query* is to be sent to the sender, and *qstate* is a query-specific secret information that is kept private.

2. **Response.** The sender has access to a database $\mathtt{DB} \in \{0,1\}^N$. Upon receiving the query message *query* from the receiver, it runs the response algorithm

   $$resp \leftarrow \mathsf{PIR.Response}(1^\kappa, \mathtt{DB}, params, query) \ .$$

   The response *resp* is then sent back to the receiver.

3. **Decode.** Upon receiving *resp*, the receiver decodes the response by running

   $$x \leftarrow \mathsf{PIR.Decode}(1^\kappa, setupstate, qstate, resp) \ .$$

   The output $x \in \{0,1\}$ is the output of the protocol.

We remark that one could consider a definition with multi-round query-response interaction per each database query. However, the simpler (and more round-efficient) definition above is sufficient for our protocols.

The communication complexity of the protocol is defined to be $|query| + |resp|$. Namely, the number of bits being exchanged to transfer a single database element (excluding the setup phase).

Correctness and privacy are defined as follows.

- **Correctness.** For all $\kappa \in \mathbb{N}$, $\mathtt{DB} \in \{0,1\}^*$ where $N \triangleq |\mathtt{DB}|$, and $i \in [N]$, it holds that

  $$\Pr[\mathsf{PIR.Decode}(1^\kappa, setupstate, qstate, resp) \neq \mathtt{DB}[i]] = \mathrm{negl}(\kappa) \ ,$$

  where $(params, setupstate) \leftarrow \mathsf{PIR.Setup}(1^\kappa)$, $(query, qstate) \leftarrow \mathsf{PIR.Query}(1^\kappa, setupstate, i)$ and $resp \leftarrow \mathsf{PIR.Response}(1^\kappa, \mathtt{DB}, params, query)$.

- $(t, \epsilon)$**-Privacy.** For all $\kappa \in \mathbb{N}$, $N \in \mathbb{N}$ and for any adversary $\mathcal{A}$ running in time $t = t_{\kappa,N}$ it holds that

$$\max_{\substack{\mathbf{i}=(i_1,\dots,i_t),\\ \mathbf{j}=(j_1,\dots,j_t)\in[N]^t}} \left|\Pr[\mathcal{A}(params, \mathbf{i}, query_{\mathbf{i}}) = 1] - \Pr[\mathcal{A}(params, \mathbf{j}, query_{\mathbf{j}}) = 1]\right| \le \epsilon \ \big( = \epsilon_{\kappa,N}\big) \ ,$$

where $(params, setupstate) \leftarrow \mathsf{PIR.Setup}(1^\kappa)$, $(query_{i_\ell}, qstate_{i_\ell}) \leftarrow \mathsf{PIR.Query}(1^\kappa, setupstate, i_\ell)$ and $(query_{j_\ell}, qstate_{j_\ell}) \leftarrow \mathsf{PIR.Query}(1^\kappa, setupstate, j_\ell)$, for all $\ell \in [t]$.

We note that the definition of privacy above differs from the one usually found in literature. The standard definition refers to vectors $\mathbf{i}, \mathbf{j}$ of dimension 1. That is, only allow the adversary to see one query to the database. The reason is that a hybrid argument can show that (allowing proper degradation in parameters) the dimension-1 definition guarantees privacy for our definition as well. However, in the public-key setting, where the same public key is used for all queries, this hybrid argument no longer works. Thus, we must require that the adversary is allowed to view many query strings.[18] In fact, one could consider even stronger attacks in the public-key setting, which are outside the scope of this work.

The definition of privacy deserves some further discussion since we did not define the ranges of parameters for $(t, \epsilon)$ for which the protocol is considered "private". Indeed there are several meaningful ways to define what it means for a protocol to be private. Let us discuss two options and provide corresponding definitions.

i. The first approach is to argue that the resources of the adversary are similar to those of an honest server (we can think of an adversary as a "server gone bad"). Thus, in this approach the adversary can run in polynomial time in $N, \kappa$ and must still not succeed with non-negligible probability in $N, \kappa$. We say that a protocol is $(i)$-private if it is $(p(\kappa, N), 1/p(\kappa, N))$-private for any polynomial $p(\cdot, \cdot)$.

ii. The second approach argues that the security parameter is the "real" measure for privacy. Thus the protocol needs to be exponentially secure in the security parameter. Thus a protocol is $(ii)$-private if it is $(2^{\Omega(\kappa)}, 2^{-\Omega(\kappa)})$-private.

### 2.5.2 PIR via Somewhat Homomorphic and Symmetric Encryption

In this section, we describe a generic PIR protocol that uses a somewhat homomorphic encryption and an arbitrary symmetric encryption schemes as building blocks. This protocol has the useful property that the somewhat homomorphic scheme is not used to encrypt the index to the database. Rather, we use the symmetric scheme to encrypt the index, and have the server homomorphically decrypt it during query evaluation. Thus, the receiver's query can be rather short.

Our PIR protocol relies on two building blocks – a semantically secure symmetric encryption scheme $\mathsf{SYM} = (\mathsf{SYM.Keygen}, \mathsf{SYM.Enc}, \mathsf{SYM.Dec})$ over the message space $[N]$, and a somewhat homomorphic encryption scheme $\mathsf{HE} = (\mathsf{HE.Keygen}, \mathsf{HE.Enc}, \mathsf{HE.Dec}, \mathsf{HE.Eval})$.

[18] We feel that our definition captures the essence of an attack on a PIR protocol better than the standard single-query definition, even in the non-public-key setting: As we mention above, converting between the definitions incurs a linear blowup in the adversary's advantage. Namely, a $(t, \epsilon)$-private protocol according to the old definition is only $(t, t\epsilon)$-private according to ours (recall that $t$ can be super-polynomial).

Jumping ahead, the level of somewhat homomorphism required for the protocol depends on the symmetric scheme being used (in particular, the decryption complexity of the symmetric scheme). Specifically, relying on a leveled fully homomorphic scheme (such as the one obtained from $\mathsf{LWE}$ in Section 2.4) will allow us to use an arbitrary symmetric scheme. A clever selection of the symmetric scheme, however, could imply that weaker somewhat homomorphic schemes (such as $\mathsf{BTS}$) will suffice.

Our protocol $\mathsf{PIR} = (\mathsf{PIR.Setup}, \mathsf{PIR.Query}, \mathsf{PIR.Response}, \mathsf{PIR.Decode})$ runs as follows.

- $\mathsf{PIR.Setup}(1^\kappa)$: In the setup procedure, we generate a symmetric key $symsk \leftarrow \mathsf{SYM.Keygen}(1^\kappa)$ and keys for the somewhat homomorphic scheme $(hpk, hevk, hsk) \leftarrow \mathsf{HE.Keygen}(1^\kappa)$.

  The symmetric key is then encrypted using the homomorphic public key to create a ciphertext

  $$c_{symsk} \leftarrow \mathsf{HE.Enc}_{hpk}(symsk) \ .$$

  We note that if $\mathsf{HE}$ is a bit encryption scheme, then $symsk$ is encrypted bit by bit.

  The setup procedure then outputs the public parameters

  $$params := (hevk, c_{symsk}) \ ,$$

  and the secret state

  $$setupstate := (hpk, hsk, symsk) \ .$$

- $\mathsf{PIR.Query}(1^\kappa, setupstate, i)$: To generate a query string, we just encrypt $i$ using the symmetric scheme. Recall that $setupstate = (hpk, hsk, symsk)$, then

  $$query \leftarrow \mathsf{SYM.Enc}_{symsk}(i) \ .$$

  In our scheme, no additional information needs to be saved per query: $qstate := \phi$.

- $\mathsf{PIR.Response}(1^\kappa, \mathtt{DB}, params, query)$: Upon receiving a query, a response is computed as follows. Recall that $params = (hevk, c_{symsk})$ and consider the function $h$ defined as follows:

  $$h(x) \triangleq \mathtt{DB}[\mathsf{SYM.Dec}_x(query)] \ .$$

  Namely, the function $h$ uses its input as a symmetric key to decrypt the query, and then uses the plaintext to index the database and retrieve the appropriate value. Note that $h(symsk) = \mathtt{DB}[i]$, where $i$ is the index embedded in $query$.

  While $\mathsf{PIR.Response}$ does not know $symsk$, it does know $c_{symsk}$ and thus can homomorphically evaluate $h(symsk)$ and set

  $$resp \leftarrow \mathsf{HE.Eval}_{hevk}(h, c_{symsk}) \ .$$

  Note that $resp$ should correspond to a decryptable ciphertext of $\mathtt{DB}[i]$.

- $\mathsf{PIR.Decode}(1^\kappa, setupstate, qstate, resp)$: We recall that $setupstate = (hpk, hsk, symsk)$ and that $qstate$ is null. To decode the response to the query, we decrypt the ciphertext associated with $resp$, outputting

  $$b \leftarrow \mathsf{HE.Dec}_{hsk}(resp) \ .$$

Correctness and privacy are easily reduced to those of the underlying primitives in the following lemmas.

**Lemma 2.5.1** (correctness). *If our symmetric scheme* $\mathsf{SYM}$*, and our somewhat homomorphic scheme* $\mathsf{HE}$ *are correct, and if the somewhat homomorphic scheme can evaluate the function* $h$ *defined above, then our PIR protocol is correct.*

*Proof.* Since $\mathsf{HE}$ is correct with regards to homomorphic evaluation, then with all but negligible probability $b = h(symsk)$. Since $\mathsf{SYM}$ is correct, it follows that $h(symsk) = \mathtt{DB}[i]$ with all but negligible probability. □

**Lemma 2.5.2** (privacy). *If our somewhat homomorphic scheme is* $(t \cdot \mathrm{poly}(\kappa), \epsilon_1)$*-CPA secure and our symmetric scheme is* $(t+\mathrm{poly}(\kappa), \epsilon_2)$*-CPA secure, then our PIR protocol is* $(t, 2(\epsilon_1+\epsilon_2))$*-private.*

*Proof.* We prove this by a series of hybrids (or experiments). Let $\mathcal{A}$ be an adversary that runs in time $t$ against the privacy of our protocol and has advantage $\epsilon$. We consider the behavior of $\mathcal{A}$ in a number of hybrids $H_0, H_1, H_2$ defined below. We let $\mathrm{Adv}_{H_i}[\mathcal{A}]$ denote the advantage of $\mathcal{A}$ in hybrid $H_i$.

- **Hybrid** $H_0$**.** This is identical to the original privacy game of the protocol. By definition
$$\mathrm{Adv}_{H_0}[\mathcal{A}] = \epsilon \ .$$

- **Hybrid** $H_1$**.** We now change the game so that instead of computing $c_{symsk} \leftarrow \mathsf{HE.Enc}_{hpk}(symsk)$ in $\mathsf{PIR.Setup}$, we will set $c_{symsk} \leftarrow \mathsf{HE.Enc}_{hpk}(0)$.

  There exists an adversary $\mathcal{B}$ for the CPA-security of the somewhat homomorphic scheme that runs in time $t \cdot \mathrm{poly}(\kappa)$ and whose advantage is
$$\mathrm{CPAAdv}[\mathcal{B}] = (1/2) \cdot |\mathrm{Adv}_{H_0}[\mathcal{A}] - \mathrm{Adv}_{H_1}[\mathcal{A}]| \ .$$
  It follows that
$$|\mathrm{Adv}_{H_0}[\mathcal{A}] - \mathrm{Adv}_{H_1}[\mathcal{A}]| \le 2\epsilon_1 \ .$$

- **Hybrid** $H_2$**.** We now change the game so that instead of setting $query_\ell \leftarrow \mathsf{SYM.Enc}_{symsk}(i_\ell)$ in $\mathsf{PIR.Query}$, we will set $query_\ell \leftarrow \mathsf{SYM.Enc}_{symsk}(0)$ for all $\ell \in [t]$.

  There exists an adversary $\mathcal{C}$ for the CPA-security of the symmetric scheme that runs in time $t + \mathrm{poly}(\kappa)$ and whose advantage is
$$\mathrm{CPAAdv}[\mathcal{C}] = (1/2) \cdot |\mathrm{Adv}_{H_1}[\mathcal{A}] - \mathrm{Adv}_{H_2}[\mathcal{A}]| \ .$$
  It follows that
$$|\mathrm{Adv}_{H_1}[\mathcal{A}] - \mathrm{Adv}_{H_2}[\mathcal{A}]| \le 2\epsilon_2 \ .$$
  However, in $H_2$, the view of the adversary is independent of the queried indices. Therefore
$$\mathrm{Adv}_{H_2}[\mathcal{A}] = 0 \ .$$

It follows that $\epsilon \leq 2(\epsilon_1 + \epsilon_2)$ as required. □

Lastly, let us analyze the communication complexity of our protocol. It follows by definition that the query size is the length of an encryption of $\{0,1\}^{\lceil \log N \rceil}$ bits using our symmetric scheme, and the response is the encryption of a single bit using our somewhat homomorphic scheme.

### 2.5.3 Instantiating the Components: The PIR Protocol

We show how to implement the primitives required in Section 2.5.2 in two different ways.

**An Explicit $\mathsf{LWE}$-Based Solution.** The first idea is to use an optimized, symmetric-key $\mathsf{LWE}$-based encryption as the symmetric encryption scheme in the PIR protocol, together with our scheme $\mathsf{BTS}$ as the homomorphic scheme. Specifically, using the same parameters $k, p$ as in our bootstrappable scheme, we get a symmetric scheme whose decryption is almost identical to that of our bootstrappable scheme. We overview our approach below and then provide the formal argument.

We apply the optimization of [PVW08, ACPS09] to Regev's scheme to get a scheme that encrypts the $\log N$ bits of the index using a ciphertext of only $O(\log N) + O(k \log k)$ bits. Roughly speaking, this optimization is based on two observations: First, rather than encrypting a single bit using an element of $\mathbb{Z}_p$, we can "pack in" $O(\log p)$ bits, if we set the error in the $\mathsf{LWE}$ instances to be correspondingly smaller (but still a $1/\mathrm{poly}(k)$ fraction of $p$). Secondly, observe that in a symmetric ciphertext $(\mathbf{v}, w) \in \mathbb{Z}_p^k \times \mathbb{Z}_p$, most of the length is dedicated to the vector $\mathbf{v}$. The observation of [PVW08, ACPS09] is that $\mathbf{v}$ can be re-used to encrypt multiple messages using different secret keys $\mathbf{s}_1, \ldots, \mathbf{s}_\ell$. Using these optimizations, the resulting PIR protocol has query length of $O(k \log k + \log N)$ bits and response length $O(k \log k)$, for $k = \mathrm{poly}(\kappa)$. The following corollary summarizes the properties of this scheme.

**Corollary 2.5.3** ([PVW08, ACPS09])**.** *Let $p, k, \hat{\chi}$ be as in Theorem 2.4.2. Then there exists a $\mathsf{DLWE}_{k,p,\hat{\chi}}$-secure symmetric encryption scheme whose ciphertext length is $O(k \log k + \ell)$ for $\ell$-bit messages, and whose decryption circuit has the same depth as that of* $\mathsf{BTS.Dec}$.

Recall the analysis of $\mathsf{BTS}$ from Section 2.4. We can prove that the function $h$ can be evaluated homomorphically.

**Lemma 2.5.4.** *Let* $\mathsf{SYM}$ *be the scheme from Corollary 2.5.3, then $h(x) \triangleq \mathtt{DB}[\mathsf{SYM.Dec}_x(query)]$ is such that $h \in \mathsf{Arith}[O(\log k) + \log\log N, N]$.*

*Proof.* We implement $h$ as follows. First, we decrypt the value of *query* to obtain an index $i$. Then, we compute the function $\sum_{j \in [N]} \mathtt{DB}[j] \cdot \mathbb{1}_{i=j}$. The decryption circuit is implemented in depth $O(\log k)$ as in Lemma 2.4.5. The function $\mathbb{1}_{i=j}$ is implemented using a comparison tree of depth $\log\log N$. Finally, a collation gate of fan-in $N$ is used to compute the final sum. The result follows. □

This means that we can choose $n$ to be large enough such that $h$ can be evaluated by $\mathsf{BTS}$.

**Theorem 2.5.5.** *There exists a PIR protocol with communication complexity $O(k \log k + \log N)$ based on the $\mathsf{DLWE}_{n,q,\chi}$ and $\mathsf{DLWE}_{k,p,\hat{\chi}}$ assumptions, for $n = \mathrm{poly}(k)$ and the remainder of the parameters as in Theorem 2.4.2.*

*Proof.* We choose $n$ such that $L = \Omega(\epsilon \log n) > O(\log k) + \log \log N$ and such that $\sqrt{q} = 2^{n^\epsilon/2} \geq N$. This will result in $n = \mathrm{poly}(k, \log N)$. The result follows from Theorem 2.4.2 and Theorem 2.4.3 (recall that the communication complexity does not depend on $n$). □

For the best currently known attacks on LWE (see [MR09, LP11, RS10]), this protocol is $(2^{\Omega(k/\mathrm{polylog} k)}, 2^{-\Omega(k/\mathrm{polylog} k)})$-private. Thus, going back to our definitions in Section 2.5.1, and setting $k = \kappa \cdot \mathrm{polylog}(\kappa)$, we get a $(ii)$-private PIR protocol with a total communication complexity of $O(\log N) + O(\kappa \cdot \mathrm{polylog}(\kappa))$; and a $(i)$-private protocol with communication complexity $\log N \cdot \mathrm{polyloglog}(N)$ by setting $\kappa = \log N \cdot \mathrm{polyloglog}(N) = \omega(\log N)$.

**An Almost Optimal Solution Using Pseudorandom Functions.** A second instantiation aims to bring the $(ii)$-private communication complexity down to $\log N + \kappa \cdot \mathrm{polylog}(\kappa)$. This can be done by instantiating our generic construction with an ciphertext-length-optimal symmetric encryption scheme, i.e. one with ciphertexts length $\log N$. Such a scheme follows immediately given any pseudo-random function (PRF).

If we want to base security solely on $\mathsf{LWE}$, we can use the $\mathsf{LWE}$-based PRF that is obtained by applying the GGM transformation [GGM86] to an $\mathsf{LWE}$ based pseudorandom generator. Note that using such instantiation, we cannot argue that $h \in \mathsf{Arith}[L, T]$ for reasonable $L, T$ (since the complexity of evaluating the PRF might be high). However, we can use our leveled fully homomorphic scheme to evaluate a function of any depth, and in particular the aforementioned PRF.

**The Complexity of Transmitting the Public Parameters.** Finally, we note that the parameters produced in the setup phase of our protocol are of length $\mathrm{poly}(\kappa)$. Thus our protocol can be trivially modified to work in a setting without setup, with communication complexity $\log N + \mathrm{poly}(\kappa)$ (under the $(ii)$-private notion) and $\mathrm{polylog}(N)$ (under the $(i)$-private notion).

# Chapter 3

# Continual Memory Leakage

In most of cryptography, key secrecy is paramount, and compromise of even a portion of the secret key often yields a break of the entire cryptosystem. In practice, unfortunately, it is difficult to maintain perfect secrecy of cryptographic keys. For one, it has been shown that various *side-channel attacks* [Koc96, KJJ99], exploiting physical characteristics of the execution of a cryptographic algorithm (e.g., timing measurements, power consumption, or electromagnetic radiation), can be used to obtain information about secret keys. More recently, so-called "cold boot" attacks [HSH$^+$08] have been used to recover some (noisy) fraction of the secret keys used in poor cryptographic implementations.

Responding to this challenge, cryptographers over the past several years have begun to investigate the construction of cryptographic schemes that are provably resilient to various forms of key leakage. There are, of course, many different ways in which this desired notion of *leakage resilience* can be modeled. Early work (e.g., [Riv97, CDH$^+$00]) focused on an adversary who could learn (a bounded number of) *specific bits* of the secret key. Considering known attacks, the early models had a number of undesirable properties:

1. The attacker is allowed to obtain information about the secret key, but not about intermediate values that are used for the computation (e.g. secret randomness). A side channel attack performed at the time of signing, for example, quite plausibly exposes information about intermediate values.

2. The attacker's view is further restricted to include only bits of the secret key (or some other very specific function). Thus an assumption is made on the nature of the measuring devices used by the attacker. As technology advances, this is undesirable.

3. The time dimension of the attack is ignored. There is an absolute upper bound on the amount of information gained throughout the attack. Naturally, in real life we cannot guarantee that the attacker will stop gathering information at some point in time.

Newer works, therefore, tried to address these issues and bring about protection against broader classes of attacks. Ishai, et al. [ISW03, IPSW06] addressed the first and third drawbacks. They consider a circuit that implements the cryptographic functionality at hand, and allow the attacker to obtain the values of some arbitrary wires in that circuit. The circuit can further be updated after each computation, so as to allow for future probing. They showed how to compile *any* cryptographic computation into a circuit that is secure in their model.

Micali and Reyzin [MR04] presented a model addressing the second drawback under the simplifying assumption that "only computation leaks information". They consider an adversary that can learn *arbitrary information* (and not just the values of specific bits) about portions of memory that are accessed during a particular computational step, but is assumed to learn nothing about any portions of memory that are untouched. Various works (see related work below) showed how to implement various primitives in this model, some of which also included an update operation to address continual (time unbounded) attacks.

Akavia, Goldwasser, and Vikuntanathan [AGV09] addressed the first and second drawbacks, allowing the leakage to depend on *all* parts of the memory arbitrarily (regardless of whether they are computed on at some specific time period), and allowing to output an arbitrary function of the entire memory, provided that the *total-length* of the leaked information is bounded. Namely, they impose a *global* bound on the amount of leakage in the entire lifetime of the scheme, regardless of the time dimension. Variants of this so called *bounded memory leakage model* require that the secret key retains sufficient min-entropy given the leakage [NS09] (see also [DOPS04]), or require that the secret key remains computationally hard to compute given the leakage [DKL09]. Subsequent work in the bounded memory leakage model includes [KV09, ADW09, DGK$^+$10, ADN$^+$10].

**Continual Memory Leakage.** In this work, we introduce a new model that adds the time dimension to the aforementioned bounded memory leakage model (hence "continual memory leakage"). We consider implementations in which the secret state is updated over time, and the total leakage over the lifetime of the system is *unbounded*, while continuing to allow the leakage between updates to be an *arbitrary* function of the secret state, subject only to a restriction on the rate of the leakage.

Our model, therefore, is unrestricted in both *time* (since leakage may occur forever) and *space* (since leakage may always depend on all portions of the secret memory). We remark, however, that we do make the implicit assumption in our model that data can be completely erased from memory (as otherwise the adversary would be able, over time, to leak the entire initial secret key).

While this thesis is focused on public-key encryption, our model applies to virtually any cryptographic primitive. We will therefore illustrate our model in the context of public-key encryption, while keeping in mind its applicability to other tasks (digital signatures will be used as running example). Initially, a public and secret key-pair is generated and the secret key is stored on the decrypting device. The lifetime of the system is divided into discrete time periods, where during each time period the decrypting device may use the secret key to decrypt messages at will (more generally: performing any cryptographic task, like generating digital signatures). At the end of each time period, a "refresh" or "update" operation is performed on the secret key, and the old secret key is erased. We stress that, as in forward-secure cryptosystems, the public key remains fixed throughout the lifetime of the system and is all that is needed for encryption (or signature verification).

We consider an attacker that can leak information about the secret state of the decrypting device, as we now make precise. We view the decrypting device as containing two types of memory:

1. *Public memory* that stores, e.g., the public key and any public randomness used.

2. *Secret memory* that stores the secret key and any secret randomness used to perform the desired cryptographic task (e.g. signing).

We assume the attacker can see the contents of the public memory in its entirety, at all times. Leakage from the secret memory is bounded *per time period*, but unbounded overall.

Formally, at each time period, the adversary can specify a leakage function $f$, and is given $f(sk, r)$, where $sk$ is the secret key at that time period and $r$ is the secret random tape. We remark that the case of decryption is simpler since this is a deterministic task (compared to signing, for example). The only restriction placed on $f$ is that the output length of all such leakage functions during any specific time period should be bounded to some pre-specified fraction of the number of bits of the secret state. Our model also incorporates leakage that may occur during the key-update process itself.

**Our Contributions.** We present the first public-key encryption scheme that is secure in the continual memory leakage model. To this effect, we formalize the model in a general context and also specifically for public-key encryption. We present a new linear algebraic technique which we use to finally present a continual memory leakage resilient encryption scheme and prove its security.

Our scheme can be based on any $d$-linear assumption in bilinear groups: Given a group $\mathbb{G}$, the $d$-linear assumption is that rank $d$ matrices of group elements are computationally indistinguishable from completely random matrices. For $d = 1$, this is equivalent to the decisional Diffie-Hellman assumption. For a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, the $d$-linear assumption is that the former holds for both $\mathbb{G}_1$ and $\mathbb{G}_2$. There are groups on which the assumption is believed to hold even for $d = 1$, and as $d$ increases, the assumption becomes weaker (i.e. more secure). For $d \geq 2$, the assumption is believed to hold in some groups where $\mathbb{G}_1 = \mathbb{G}_2$.

Our scheme is secure even if a $(1/d - o(1))$ fraction of the secret state is leaked per time period, which translates to an optimal $(1 - o(1))$ fraction when using 1-linear (an assumption also known as SXDH or "symmetric external Diffie-Hellman"). In addition, for all $d$, our scheme can be shown resilient to $O(\log k)$ bits of leakage during the key-update process itself (and during the key generation process), where $k$ is the security parameter. (If we are willing to rely on sub-exponential hardness assumptions, we can tolerate $O(k^\epsilon)$ bits of leakage from each key update.) We prove this property (resilience to leakage during updates) in a semi-generic way:[1] We show that any scheme that is secure without leakage from the updates, but allows the adversary to "rewind" and go back to an older key, is also secure with logarithmic leakage from the update.

It is easy to see that any public-key encryption scheme resilient to continual leakage automatically implies a private-key encryption scheme with similar leakage resilience. Thus, our scheme can be used to enable two parties who share a secret key to interact over an insecure channel in the presence of side-channel attacks. The parties simply *individually and independently* update their secret keys. Thus, at any point of time, the two communicating parties might have completely different secret keys, and still they will be able to communicate meaningfully. We stress that, as opposed to previous solutions to this problem, here the key update requires no interaction whatsoever.

Since the focus of this thesis is on public-key encryption, we do not include additional results that appeared in the original paper [BKKV10]. Specifically, there, we showed a continual memory leakage resilient identity-based encryption scheme and digital signature schemes.

[1] This proof technique is implicit in our original proof in [BKKV10], as was pointed out to us by Daniel Wichs.

**Other Related Work.** Dziembowski and Pietrzak [DP08, Pie09] constructed stream ciphers under the "only computation leaks information" assumption. Faust, et al. [FKPR10] construct a signature scheme that is secure in the same leakage model.

Additional research has aimed at achieving leakage resilience by relying on a small amount of perfectly *leakage-proof hardware*. In this model, Juma and Vahlis [JV10] and Goldwasser and Rothblum [GR10] show how to obtain continual leakage resilience for general cryptographic algorithms, based also on the assumption that "only computation leaks information" (for computation not done on the secure hardware). Faust, et al. [FRR$^+$10] show how to use leakage-proof hardware to tolerate length-bounded leakage computed by a function in $\mathbf{AC}^0$

In work done concurrently with our own, Dodis, et al. [DHLW10] construct efficient signature schemes, identification schemes, and authenticated key-agreement protocols that are secure in the model of continual memory leakage. Unlike this work, they do not address the issue of leakage from the key-update process and, furthermore, they do not construct public-key encryption schemes.

**Follow-Up Work.** In the short period of time since our work has first been presented, a large number of follow-ups has already appeared.

Lewko, Lewko and Waters [LLW11] showed a scheme that can tolerate a linear fraction of the secret state being leaked even during key updates (in our work, we could only prove for a logarithmic number of bits). Again, linear algebraic techniques extending ours were used.

Kalai, Kanukurthi and Sahai [KKS11] extend our model to consider continual *tampering*, where the adversary can, in addition to leakage, change the contents of the secret memory. They present results based on the $d$-linear assumption by extending our linear algebraic techniques. In addition, they show how to modify our scheme to tolerate leakage of $(1 - o(1))$ fraction of the secret key, regardless of $d$.

In the context of identity-based encryption, Lewko, Rouselakis and Waters [LRW11] showed, using our linear algebraic techniques among others, how to construct a hierarchal identity-based encryption scheme that is resilient to continual memory leakage from all secret keys. For digital signatures, Malkin, Teranishi, Vahlis and Yung [MTVY11] and Boyle, Segev and Wichs [BSW11] addressed the issue of leakage during the signing process, presenting improved results in this context.

Akavia, Goldwasser and Hazay [AGH10] considered a different, distributed model, where the key is split between two continually-leaky entities who communicate over a public channel to perform cryptographic tasks. They present an identity based encryption scheme in that model, that can tolerate $(1-o(1))$ leakage even during key updates. Very recently, a distributed storage scheme where the updates require no interaction was presented by Dodis, Lewko, Waters and Wichs [DLWW11]. Their result also simplifies and improves upon the public-key scheme of [LLW11] mentioned above.

**Chapter Organization.** Some notational conventions and definitions appear in Section 3.1, followed by a formal definition of the $d$-linear assumption in bilinear groups in Section 3.2. We present our model for continual memory leakage in Section 3.3. Section 3.4 contains our linear algebraic tool: "random subspaces are continual memory leakage resilient", that will be used to prove the security of our scheme, but is of independent interest. Finally, in Section 3.5 we present our scheme and a proof overview, while the full proof appears in Section 3.6.

## 3.1 Definitions and Notation

We use the following notational conventions. Bold uppercase denotes matrices ($\mathbf{X} \in \mathbb{Z}_q^{n\times k}$) and bold lowercase denotes vectors ($\mathbf{x} \in \mathbb{Z}_q^n$). All vectors are column vectors, row vectors are denoted by $\mathbf{x}^T$. For a scalar (usually a group element) $g$ and a matrix $\mathbf{X} \in \mathbb{Z}^{k\times n}$ (or a vector, as a special case), we let $g^{\mathbf{X}}$ denote a $k \times n$ matrix such that $(g^{\mathbf{X}})_{i,j} = g^{(\mathbf{X})_{i,j}}$.

**Linear Algebra.** We use the term *span* of a matrix to indicate its row span, i.e. for $\mathbf{A} \in \mathbb{Z}_q^{n\times k}$, we denote $\mathrm{Span}(\mathbf{A}) = \{\mathbf{z}^T \cdot \mathbf{A} : \mathbf{z} \in \mathbb{Z}_q^n\}$. The *kernel* of a matrix is the linear space that is orthogonal to its span, this corresponds to defining $\ker(\mathbf{A}) = \{\mathbf{x} \in \mathbb{Z}_q^k : \mathbf{A} \cdot \mathbf{x} = \mathbf{0}\}$.

The set of $(n \times k)$-matrices having rank-$d$ is denoted by $\mathrm{Rk}_d(\mathbb{Z}_p^{n\times k})$. We will also use the following simple fact (see e.g. [BK10]), we provide a proof for completeness.

**Lemma 3.1.1.** *Consider a finite field $\mathbb{F}$ of order $p$. For any $n, m \in \mathbb{N}$ such that $m \geq n$, the distance between the distributions $U(\mathbb{F}^{n\times m})$ and $U(\mathrm{Rk}_n(\mathbb{F}^{n\times m}))$ is at most $1/(p^{m-n} \cdot (p-1))$.*

*Proof.* Consider sampling the matrix row by row. The probability that row $i$ is a linear combination of previous rows is at most $p^{-m} \cdot p^{i-1}$. Applying the union bound gives the result. □

**Groups and Bilinear Maps.** If $g \in \mathbb{G}$ is an element in a group $\mathbb{G}$ of order $p$ and $\mathbf{X} \in \mathbb{Z}_p^{m\times n}$ is a matrix, then $g^{\mathbf{X}} \in \mathbb{G}^{m\times n}$ denotes the matrix where $(g^{\mathbf{X}})_{i,j} = g^{(\mathbf{X})_{i,j}}$. Observe that given a matrix $g^{\mathbf{X}} \in \mathbb{G}^{m\times n}$ and a matrix $\mathbf{A} \in \mathbb{Z}_p{}^{\ell\times m}$ one can efficiently compute $g^{\mathbf{AX}}$ by working "in the exponent".

Consider multiplicative groups $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$, all of prime order $p$, and let $g_1, g_2$ be generators for $\mathbb{G}_1, \mathbb{G}_2$ respectively. A bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ has the following properties. *Bilinearity:* for all $x \in \mathbb{G}_1, y \in \mathbb{G}_2$, $a, b \in \mathbb{Z}$ it holds that $e(x^a, y^b) = e(x, y)^{ab}$; *Non-degeneracy:* $e(g_1, g_2) \neq 1$.

Note further that if $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is a bilinear map, then given matrices $g_1^{\mathbf{X}} \in \mathbb{G}^{m\times n}$ and $g_2^{\mathbf{Y}} \in \mathbb{G}^{n\times \ell}$ one can efficiently compute the matrix $e(g_1, g_2)^{\mathbf{XY}} \in \mathbb{G}_T^{m\times \ell}$.

**Probability.** Let $X$ be a probability distribution over a domain $S$, we write $x \overset{\$}{\leftarrow} X$ to indicate that $x$ is sampled from the distribution $X$. We use $x \overset{\$}{\leftarrow} S$ to indicate the sampling uniformly over $S$. For any function $f$ with domain $S$ we let $f(X)$ denote the random variable (or corresponding distribution) obtained by sampling $x \overset{\$}{\leftarrow} X$ and outputting $f(x)$.

We write $\mathrm{negl}(k)$ to denote an arbitrary *negligible* function, i.e. one that vanishes faster than the inverse of any polynomial.

The *statistical distance* between two distributions $X, Y$ (or random variables with those distributions) is denoted $\mathtt{dist}(X, Y)$. Two ensembles $X = \{X_k\}_k$, $Y = \{Y_k\}_k$ are $\epsilon = \epsilon(k)$*-close* if the statistical distance between them is at most $\epsilon(k)$. They are called *statistically indistinguishable* if $\epsilon(k) = \mathrm{negl}(k)$. An ensemble $X = \{X_k\}_k$ over domains $S = \{S_k\}_k$ is $\epsilon = \epsilon(k)$*-uniform* in $S$ if it is $\epsilon$-close to the uniform ensemble over $S$ (we sometimes omit $S$ when it is clear from the context). $X = \{X_k\}_k$, $Y = \{Y_k\}_k$ are *computationally indistinguishable* if every $\mathrm{poly}(k)$-time adversary $\mathcal{A}$ has negligible distinguishing advantage:

$$|\Pr[\mathcal{A}(X_k) = 1] - \Pr[\mathcal{A}(Y_k) = 1]| = \mathrm{negl}(k) \ .$$

## 3.2 The Decision $d$-Linear Assumption in Bilinear Groups

Our construction here relies on the (matrix form of the) decisional linear assumption in bilinear groups [BBS04, NS09].

**Definition 3.2.1.** *Given a multiplicative cyclic group $\mathbb{G}$ (or rather, an ensemble of groups) of order $p$, generated by a generator $g$, the $d$-linear assumption, for $d \in \mathbb{N}$, states that the ensembles $\{g^{\mathbf{X}}\}$ and $\{g^{\mathbf{Y}}\}$ are computationally indistinguishable, where $\mathbf{X}$ is a random rank-$d$ matrix over $\mathbb{Z}_p$, and $\mathbf{Y}$ is a random rank-$(d+t)$ matrix ($t \geq 0$) over $\mathbb{Z}_p$ of the same dimensions.*

*Given (ensembles of) groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$, all cyclic of order $p$, and a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, the bilinear $d$-linear assumption states that the $d$-linear assumption holds for both $\mathbb{G}_1$ and $\mathbb{G}_2$.*

We note that $d$-linear is (potentially) weaker than $(d+1)$-linear (indeed there are black box separations between $d$- and $(d+1)$-linear for all $d$). The 1-linear assumption is equivalent to the decision Diffie-Hellman assumption. Therefore, the bilinear 1-linear assumption cannot hold if $\mathbb{G}_1 = \mathbb{G}_2$ (or if they have an efficient isomorphism between them). The bilinear 1-linear assumption is also known as symmetric external Diffie-Hellman (or SXDH). There exist candidate groups for which this assumption is believed to hold. For $d \geq 2$, there are candidate bilinear maps with $\mathbb{G}_1 = \mathbb{G}_2$ where the assumption is believed to hold.

## 3.3 A Model for Continual Memory Leakage

In this section, we formalize our model of *continual memory leakage* (i.e., the CML model). We choose to start by describing the model at a high level in the context of digital signature schemes, and only then discuss public-key encryption. This is because the signing operation involves (possibly secret) randomness and thus exposes interesting aspects of the model. Deriving a definition for public-key encryption will then be straightforward, as we formally do in Section 3.3.1. (A formal definition for signature schemes is outside the scope of this thesis.)

In the CML model, the lifetime of the system is divided into discrete time intervals, and the secret key is updated at the end of every such interval. Within a given interval, the adversary may submit a *leakage function* $f$ of his choice, and is given $f(sk)$. In addition, as usual, the adversary may make signing queries to obtain valid signatures on messages of its choice. At the time of each signing query, though, the adversary is also allowed to submit a *leakage function* $f$; in return, the adversary is given $f(sk, r)$ where $sk$ is the secret key (at the current time interval) and $r$ is the secret randomness used to answer the signing query. (We assume $r$ is securely erased once signature generation is complete.) The signing process may also use some "public randomness", which is given to the adversary "for free" and not counted toward the leakage bound. We restrict the leakage (i.e., the output length of $f$) per signature query to be some fraction of the total length of the secret state (i.e., $|sk| + |r|$). We also restrict the total leakage per time interval to be some fraction of the length of the secret key.

At the end of an interval, the secret key is updated and the old secret key is erased. We allow the adversary also to probe the memory during this update process. The leakage during the key-update procedure is counted toward both the previous interval and the next interval; this is because the secret keys for both time intervals are completely determined by the inputs to the key-update procedure.

When public-key encryption is considered, the definition is very similar, except, naturally, for leakage during signing. Leakage during decryption does not need special treatment since it is deterministic.

In the formulation of the model just described, the leakage is restricted to be strictly smaller than the length of the secret key (following [AGV09]). Other restrictions could also be used; for example, we could require only that the secret key has sufficient min-entropy given the leakage (as in [NS09]), or that the secret key should be hard to compute given the leakage (see [DKL09]). These, however, are outside the scope of this thesis.

### 3.3.1 Definitions for Public-Key Encryption

A public-key encryption scheme $\mathsf{CML}$ in the CML model is a quadruple of probabilistic polynomial time (in the security parameter $k$) algorithms ($\mathsf{CML.Keygen}, \mathsf{CML.Enc}, \mathsf{CML.Dec}, \mathsf{CML.Update}$):

- **Key generation.** $\mathsf{CML.Keygen}$ takes as input a security parameter $1^k$ and outputs a secret key $sk$ and a public key $pk$. We denote this by $(sk, pk) {\leftarrow} \mathsf{CML.Keygen}(1^k)$.
- **Encryption.** $\mathsf{CML.Enc}$ takes a public key $pk$ and a message $m$, and outputs a ciphertext $c$. We write $c {\leftarrow} \mathsf{CML.Enc}_{pk}(m)$.
- **Decryption.** $\mathsf{CML.Dec}$ takes keys $sk, pk$ and a ciphertext $c$ and outputs a message $m^*$. We write $m^* {:=} \mathsf{CML.Dec}_{sk,pk}(c)$.
- **Key update.** $\mathsf{CML.Update}$ Takes as input keys $sk, pk$ and outputs a "refreshed" secret key $sk'$. We denote this by $sk' {\leftarrow} \mathsf{CML.Update}_{pk}(sk)$.

We remark that we must not allow the update operation to blow up the key. Specifically, in our construction, $|sk| = |\mathsf{CML.Update}_{pk}(sk)|$ (i.e. the size of the secret key remains unchanged by update operations).

Correctness is defined as follows.

**Definition 3.3.1** (correctness)**.** *The scheme* $\mathsf{CML}$ *is correct if for all* $m$ *and all polynomially bounded* $t \in \mathbb{N}$*, setting* $(sk_0, pk) {\leftarrow} \mathsf{CML.Keygen}(1^k)$*,* $sk_i {\leftarrow} \mathsf{CML.Update}_{pk}(sk_{i-1})$ *for* $i \in [t]$*, and* $c {\leftarrow} \mathsf{CML.Enc}_{pk}(m)$*, we have* $m = \mathsf{CML.Dec}_{sk_t,pk}(c)$ *with all but negligible probability (where the probability is over all randomness in the experiment).*

We next define semantic security (i.e., security against chosen-plaintext attacks) in the CML model. Our definition has three leakage parameters $\rho_G, \rho_U, \rho_M$, where $\rho_G$ bounds the leakage rate from the key-generation process, $\rho_U$ bounds the leakage rate from the update process, and $\rho_M$ is a "global" leakage bound that is enforced between key updates. Taking $\rho_G = \rho_U = 0$ corresponds to allowing leakage only during "normal" operation of the system, and not from the key-generation or key-update processes.

**Definition 3.3.2** (security under continual memory leakage)**.** *An encryption scheme* $\mathsf{CML}$ *is* $(\rho_G, \rho_U, \rho_M)$*-leakage resilient if the advantage of any* PPT *adversary* $\mathcal{A}$ *in the following game is negligible:*

1. Initialize. *$\mathcal{A}$ specifies a circuit $f$ with $|f(r,\tau)| \le \rho_G \cdot |r|$ for all $r, \tau$. The challenger chooses "secret randomness" $r$ and "public randomness" $\tau$, generates $(sk_0, pk) \leftarrow \mathsf{CML.Keygen}(1^k; r, \tau)$, sends $(pk, \tau, f(r,\tau))$ to the adversary, and sets $i{:=}0$ and $L_0{:=}\,|f(r,\tau)|$.*

2. Leakage and updates. *$\mathcal{A}$ makes the following queries:*

   - *Update queries $(\mathsf{q\text{-}update}, f)$, where $f$ is a circuit with $|f(sk, r, \tau)| \le \rho_U \cdot (|sk| + |r|)$ for all $sk, r, \tau$. The challenger chooses "secret randomness" $r$ and "public randomness" $\tau$, and computes $sk_{i+1}{:=}\mathsf{CML.Update}_{pk}(sk_i; r, \tau)$. If $L_i + |f(sk_i, r, \tau)| \le \rho_M \cdot |sk_i|$ then the challenger returns $(\tau, f(sk_i, r, \tau))$ to the adversary, sets $i{:=}i+1$, and sets $L_{i+1}{:=}|f(sk_i, r, \tau)|$. Otherwise, the challenger aborts.*
   - *Leakage queries $(\mathsf{q\text{-}mem}, f)$, where $f$ is a circuit. If $L_i + |f(sk_i)| \le \rho_M \cdot |sk_i|$ then the challenger returns $f(sk_i)$ to the adversary and sets $L_i{:=}L_i + |f(sk_i)|$. Otherwise, the challenger aborts.*

3. Challenge. *$\mathcal{A}$ outputs two messages $m_0, m_1$. A random bit $b \xleftarrow{\$} \{0,1\}$ is chosen and $c \leftarrow \mathsf{CML.Enc}_{pk}(m_b)$ is given to $\mathcal{A}$.*

4. Finish. *The adversary outputs $b' \in \{0,1\}$.*

*The advantage of the adversary is $|\Pr[b' = b] - \frac{1}{2}|$.*

We do not explicitly consider leakage during decryption because we assume that decryption is deterministic (and so any such leakage is captured by leakage queries as above).

## 3.4 Random Subspaces are Leakage Resilient

We introduce a linear-algebraic tool that is crucial to our leakage-resilient constructions. Roughly, we show that random subspaces are resilient to continual leakage in an information-theoretic sense. We believe this tool is interesting in its own right, and it indeed found further applications in follow-up works.

For concreteness we work in $\mathbb{Z}_p^m$ ($p$ prime), though our arguments generalize to arbitrary vector spaces over finite fields. Fix an arbitrary leakage function $f$ on the ambient space $\mathbb{Z}_p^m$. Let $\mathbf{X} \in \mathbb{Z}_p^{m \times n}$ be a random matrix of rank $n \ge 2$; abusing notation, we also let $\mathbf{X}$ denote the linear subspace (of dimension $n$) generated by the columns of this matrix. We show that when the output length of $f$ is sufficiently short, then the random variables $(\mathbf{X}, f(\mathbf{v}))$ (for random $\mathbf{v} \in \mathbf{X}$) and $(\mathbf{X}, f(\mathbf{u}))$ (for random $\mathbf{u} \in \mathbb{Z}_p^m$) are statistically close. Since $f(\mathbf{u})$ is independent of $\mathbf{X}$ (and hence leaks no information about $\mathbf{X}$), we conclude that when the output length of $f$ is sufficiently short, then $f(\mathbf{v})$ for a random $\mathbf{v} \in \mathbf{X}$ leaks almost no information about $\mathbf{X}$.

More generally, we consider the case where the leakage function is applied to $d$ random vectors in $\mathbf{X}$. In other words, the leakage function is applied to a random dimension-$d$ subspace $\mathbf{Y} \subseteq \mathbf{X}$ (in matrix notation $\mathbf{Y} = \mathbf{X} \cdot \mathbf{T}$, where $\mathbf{T} \in \mathbb{Z}_p^{n \times d}$ is a random matrix of rank $d$). In this case, we need to assume that the subspace $\mathbf{X}$ has dimension $n \ge 2d$.

**Theorem 3.4.1.** *Let $p$ be prime, $\epsilon > 0$, and $m, n, d \in \mathbb{N}$ with $2 \leq 2d \leq n \leq m$. Fix an arbitrary function $f : \mathbb{Z}_p^{m\times d} \to W$ with $|W| \leq 4 \cdot (1 - 1/p) \cdot p^{n-(2d-1)} \cdot \epsilon^2$. Then*

$$\texttt{dist}\left( (\mathbf{X}, f(\mathbf{X}\cdot\mathbf{T}))\,,\,(\mathbf{X}, f(\mathbf{U}))\right) \leq \epsilon\,, \tag{3.1}$$

*where $\mathbf{X} \xleftarrow{\$} \mathbb{Z}_p^{m\times n}$, $\mathbf{T} \xleftarrow{\$} \mathrm{Rk}_d(\mathbb{Z}_p^{n\times d})$, and $\mathbf{U} \xleftarrow{\$} \mathbb{Z}_p^{m\times d}$.*

Note that for $d = 1$ and $m \approx n$, the leakage function $f(\mathbf{v})$ can leak almost the entire $\mathbf{v}$. This is the case, since according to our parameters, $|\mathbf{v}| = m \cdot \log p$, and as long as the leakage size is at most $\log|W| \leq (n-1)\log p - 2\log(1/\epsilon)$, the leakage $f(\mathbf{v})$ hides the subspaces $\mathbf{X}$, up to an $\epsilon$ statistical distance. Taking $p$ to be super-polynomial in the security parameter and taking $\epsilon = 1/p$ (as we do when applying the theorem in this paper), we get that as long as the leakage contains at most $(n-3)\log p$ bits, $f(\mathbf{v})$ statistically hides the subspaces $\mathbf{X}$.

*Proof.* The proof is an adaptation of the "simple case" in the proof of [BFO08, Lemma 7.1] for the "crooked leftover hash lemma" of [DS05].[2] The first part of our proof, up to and including Eq. (3.2), is taken directly from there (with the required changes of notation), the remainder of the proof is a sequence of straightforward derivations.

Define, for random variables $X, Y$ supported on a domain $W$,

$$\begin{aligned}\mathrm{Col}(X) &\triangleq \sum_{w\in W} \Pr[X = w]^2 \\ D(X,Y) &\triangleq \sum_{w\in W} \left(\Pr[X = w] - \Pr[Y = w]\right)^2 .\end{aligned}$$

It holds that

$$\texttt{dist}\left((\mathbf{X}, f(\mathbf{X}\cdot\mathbf{T})),\ (\mathbf{X}, f(\mathbf{U}))\right) \leq \frac{1}{2}\sqrt{|W| \cdot \mathbb{E}_{\mathbf{X}}[D(f(\mathbf{X}\cdot\mathbf{T}), f(\mathbf{U}))]}\ . \tag{3.2}$$

Let us focus, therefore, on $\mathbb{E}_{\mathbf{X}}[D(f(\mathbf{X}\cdot\mathbf{T}), f(\mathbf{U}))]$, recalling that for all $\mathbf{T} \in \mathrm{Rk}_d(\mathbb{Z}_p^{n\times d})$ it holds that $\mathbf{X}\cdot\mathbf{T}$ is uniformly distributed in $\mathbb{Z}_p^{m\times d}$ (i.e. identically distributed to $\mathbf{U}$).

$$\begin{aligned}\mathbb{E}_{\mathbf{X}}[D(f(\mathbf{X}\cdot\mathbf{T}), f(\mathbf{U}))] &= \mathbb{E}_{\mathbf{X}} \sum_{w\in W} \big(\Pr_{\mathbf{T}}[f(\mathbf{X}\cdot\mathbf{T}) = w] - \Pr_{\mathbf{U}}[f(\mathbf{U}) = w]\big)^2 \\ &= \mathbb{E}_{\mathbf{X}} \sum_{w\in W} \big(\mathbb{E}_{\mathbf{T}}[\mathbb{1}_{f(\mathbf{X}\cdot\mathbf{T})=w}] - \mathbb{E}_{\mathbf{U}}[\mathbb{1}_{f(\mathbf{U})=w}]\big)^2 \\ &= \mathbb{E}_{\mathbf{X},\mathbf{T}_1,\mathbf{T}_2}\big[\sum_{w\in W} \mathbb{1}_{f(\mathbf{X}\cdot\mathbf{T}_1)=f(\mathbf{X}\cdot\mathbf{T}_2)=w}\big] - 2\cdot\mathbb{E}_{\mathbf{X},\mathbf{T},\mathbf{U}}\big[\sum_{w\in W} \mathbb{1}_{f(\mathbf{X}\cdot\mathbf{T})=f(\mathbf{U})=w}\big] \\ &\qquad + \mathbb{E}_{\mathbf{X},\mathbf{U}_1,\mathbf{U}_2}\big[\sum_{w\in W} \mathbb{1}_{f(\mathbf{U}_1)=f(\mathbf{U}_2)=w}\big] \\ &= \mathbb{E}_{\mathbf{X},\mathbf{T}_1,\mathbf{T}_2}[\mathbb{1}_{f(\mathbf{X}\cdot\mathbf{T}_1)=f(\mathbf{X}\cdot\mathbf{T}_2)}] - 2\cdot\mathbb{E}_{\mathbf{X},\mathbf{T},\mathbf{U}}[\mathbb{1}_{f(\mathbf{X}\cdot\mathbf{T})=f(\mathbf{U})}] \\ &\qquad + \mathbb{E}_{\mathbf{U}_1,\mathbf{U}_2}[\mathbb{1}_{f(\mathbf{U}_1)=f(\mathbf{U}_2)}]\end{aligned}$$

[2] We thank Yevgeniy Dodis, Gil Segev, and Daniel Wichs for pointing out this analysis to us.

$$= \mathbb{E}_{\mathbf{X},\mathbf{T}_1,\mathbf{T}_2}[\mathbb{1}_{f(\mathbf{X}\cdot\mathbf{T}_1)=f(\mathbf{X}\cdot\mathbf{T}_2)}] - \mathrm{Col}(f(\mathbf{U})) \ .$$

We are left with analyzing $\mathbb{E}_{\mathbf{X},\mathbf{T}_1,\mathbf{T}_2}[\mathbb{1}_{f(\mathbf{X}\cdot\mathbf{T}_1)=f(\mathbf{X}\cdot\mathbf{T}_2)}]$. Denote by $F$ the event where the matrix $[\mathbf{T}_1 \| \mathbf{T}_2]$ is of full rank (and by $\bar{F}$ the complementary event). Note that for all $\mathbf{T}_1, \mathbf{T}_2$ for which $F$ happens, it holds that $\mathbf{X}\cdot\mathbf{T}_1$ and $\mathbf{X}\cdot\mathbf{T}_2$ are uniform and independent. Therefore

$$\mathbb{E}_{\mathbf{X},\mathbf{T}_1,\mathbf{T}_2}[\mathbb{1}_{f(\mathbf{X}\cdot\mathbf{T}_1)=f(\mathbf{X}\cdot\mathbf{T}_2)}] \le \Pr_{\mathbf{T}_1,\mathbf{T}_2}[\bar{F}] + \mathbb{E}_{\mathbf{T}_1,\mathbf{T}_2|F}\left[\mathbb{E}_{\mathbf{X}}[\mathbb{1}_{f(\mathbf{X}\cdot\mathbf{T}_1)=f(\mathbf{X}\cdot\mathbf{T}_2)}]\right] = \Pr[\bar{F}] + \mathrm{Col}(f(\mathbf{U})) \ .$$

To bound $\Pr_{\mathbf{T}_1,\mathbf{T}_2}[\bar{F}]$, we use Lemma 3.1.1 to get

$$\Pr_{\mathbf{T}_1,\mathbf{T}_2}[\bar{F}] = \Pr\left[[\mathbf{T}_1\|\mathbf{T}_2] \text{ is not full rank}\right] \le \Pr_{\mathbf{R}\xleftarrow{\$}\mathbb{Z}_p^{n\times 2d}}[\mathbf{R} \text{ is not full rank}] \le \frac{p^{2d-n}}{p-1} \ .$$

Putting it all together, we get that

$$\mathtt{dist}\left((\mathbf{X}, f(\mathbf{X}\cdot\mathbf{T})),\ (\mathbf{X}, f(\mathbf{U}))\right) \le \sqrt{\frac{|W|}{4\cdot(1-1/p)\cdot p^{n-(2d-1)}}} \ ,$$

and the result follows. □

## 3.5 A Continual Memory Leakage Resilient Public-Key Encryption Scheme

In this section, we present our continual memory leakage resilient scheme PK-CML and state its properties (in Section 3.5.1). This is followed by an overview of the security proof (in Section 3.5.2). For a formal proof, see Section 3.6. Before we get to these, however, let us start with a high level discussion overviewing the main ideas behind the structure of our scheme.

We start by recalling a simplified version of the identity-based encryption scheme of [BK10], and then we explain what changes need to be made to achieve continual leakage resilience. We stress that in the context of this thesis, we do not discuss identity based encryption and we completely ignore this aspect of the scheme. To simplify this high level discussion, we assume here that $d = 2$ and $\mathbb{G}_1 = \mathbb{G}_2 = \mathbb{G}$ (as is the case in the [BK10] scheme).

Let $\mathbb{G}$ be a group of prime order $p$, with $g \in \mathbb{G}$ a generator. In the scheme of [BK10], the public key is $g^{\mathbf{A}}$ where $\mathbf{A} \in {\mathbb{Z}_p}^{2\times\ell}$ is chosen at random. To encrypt a "0", the sender chooses a random (column) vector $\mathbf{r} \in {\mathbb{Z}_p}^2$ and computes the ciphertext $g^{\mathbf{r}^T\mathbf{A}}$; to encrypt a "1", the sender chooses a random vector $\mathbf{u} \in {\mathbb{Z}_p}^\ell$ and outputs the ciphertext $g^{\mathbf{u}^T}$. Indistinguishability (based on the decisional 2-linear assumption) follows by looking at the $3\times\ell$ matrix defined by the public key and ciphertext: when a 0 is encrypted, this matrix takes the form $g^{\mathbf{X}}$, for $\mathbf{X} \in {\mathbb{Z}_p}^{3\times\ell}$ of rank 2, but when a 1 is encrypted the matrix takes the form $g^{\mathbf{X}}$, for $\mathbf{X}$ of rank 3 (with overwhelming probability).

The next question, of course, is how to decrypt such ciphertexts. Notice that any non-zero vector $\mathbf{y}$ in the kernel of $\mathbf{A}$ can be used to decrypt: given a ciphertext $g^{\mathbf{v}^T}$, the receiver can compute $g^{\mathbf{v}^T\cdot\mathbf{y}}$ and check if the result is $g^0$. This will always be the case for encryptions of 0, and will only happen with negligible probability for encryptions of 1. Namely, any such $\mathbf{y}$ can be used as a secret key. In [BK10], due to constraints related to the identity-based property, $g^{\mathbf{y}}$ is used as the secret key and decryption is performed using the bilinear map. We will use a similar (but not identical) secret key in our scheme, and the main challenge is then to provide a mechanism for this key to be updated while tolerating leakage.

In order to allow for updates, we use a secret key of the form $g^{\mathbf{Y}}$, where $\mathbf{Y} = [\mathbf{y}_1|\mathbf{y}_2] \in \mathbb{Z}_p^{\ell\times 2}$, and $\mathbf{y}_1, \mathbf{y}_2 \in \mathbb{Z}_p^{\ell}$ are random vectors in the kernel of $\mathbf{A}$. Decryption of a ciphertext $g^{\mathbf{v}^T}$ is done by checking whether $e(g,g)^{\mathbf{v}^T\mathbf{Y}}$ is equal to $e(g,g)^{\mathbf{0}}$.

The key-update operation is done by "rotating" the matrix $\mathbf{Y}$; i.e., the receiver samples a random $2\times 2$ full-rank matrix $\mathbf{R} \in \mathbb{Z}_p^{2\times 2}$ and sets the new secret key to $g^{\mathbf{Y}\cdot\mathbf{R}}$. Intuitively, the 2-linear assumption implies that running the update operation is indistinguishable from sampling a fresh random secret key, which turns out to be a useful property.

### 3.5.1 The Scheme PK-CML

We now formally present our scheme, motivated by the discussion above and generalizing it. Our scheme PK-CML = (PK-CML.Keygen, PK-CML.Update, PK-CML.Enc, PK-CML.Dec) is parameterized by 3 groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ of prime order $p$, with a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. Let $g_1, g_2$ be generators for $\mathbb{G}_1, \mathbb{G}_2$ respectively (this means that $e(g_1, g_2)$ is a generator for $\mathbb{G}_T$). Additional parameters are an integer $d \in \mathbb{N}$ that corresponds to the hardness assumption we use (namely, the security of the scheme will be proven based on the $d$-linear assumption on the bilinear groups), and an integer $\ell = \ell(k) \geq 3d + 1$.[3] We will prove that the scheme is resilient to leakage of $\rho_M = (1/d - 5/\ell)$ fraction of the key between updates. Namely, increasing $d$ decreases the asymptotic leakage resilience and efficiency, but potentially increases security (since $(d+1)$-linear is potentially a weaker assumption than $d$-linear). Increasing $\ell$ decreases efficiency as well, but increases the convergence to the asymptotic leakage rate of $1/d$.

- Key generation PK-CML.Keygen$(1^k)$: Sample $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_p^{d\times\ell}$ and $\mathbf{Y} \xleftarrow{\$} \ker^d(\mathbf{A})$ (so that $\mathbf{Y} \in \mathbb{Z}_p^{\ell\times d}$). Output $pk = g_1^{\mathbf{A}}$ and $sk = g_2^{\mathbf{Y}}$.

- Key update PK-CML.Update$_{pk}(sk)$: Recall that $sk = g_2^{\mathbf{Y}} \in \mathbb{G}_2^{\ell\times d}$. To update, sample $\mathbf{R} \xleftarrow{\$} \mathrm{Rk}_d(\mathbb{Z}_p^{d\times d})$ and output $sk' := g_2^{\mathbf{Y}\cdot\mathbf{R}}$.

- Encryption PK-CML.Enc$_{pk}(b)$: Let $pk = g_1^{\mathbf{A}}$. To encrypt a bit $b \in \{0,1\}$, do the following.
  - If $b = 0$, sample $\mathbf{r} \xleftarrow{\$} \mathbb{Z}_p^d$ and output $c = g_1^{\mathbf{r}^T\cdot\mathbf{A}}$.
  - If $b = 1$, sample $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_p^{\ell}$ and output $c = g_1^{\mathbf{u}^T}$.

- Decryption PK-CML.Dec$_{sk}(c)$: Let $sk = g_2^{\mathbf{Y}}$ and $c = g_1^{\mathbf{v}^T}$. Compute $e(g_1, g_2)^{\mathbf{v}^T\mathbf{Y}}$ and output 0 if and only if the result is $e(g_1, g_2)^{\mathbf{0}}$.

The leakage resilience properties of the scheme are summarized in the following theorem.

[3] Due to some slackness in our proof, we will only guarantee leakage resilience for $\ell > 5d$.

**Theorem 3.5.1.** *Based on the decision $d$-linear assumption in bilinear groups, the scheme* PK-CML *is $(\rho_G, \rho_U, \rho_M)$ continual memory leakage resilient for*

$$(\rho_G, \rho_U, \rho_M) = (0, 0, 1/d - 5/\ell) \ ,$$

*and more generally, for all $c > 0$, the scheme is resilient for*

$$(\rho_G, \rho_U, \rho_M) = \left( \frac{c \log k}{2d\ell \log p}, \frac{c \log k}{(d\ell + d^2) \log p}, 1/d - 5/\ell - \frac{c \log k}{d\ell \log p} \right) \ .$$

The theorem implies that we can trade off the leakage between the updates for leakage during the updates (but only up to a logarithmic number of bits). This tradeoff is formalized in Theorem 3.6.3, presented in Section 3.6.3. In fact, for the scheme PK-CML, one can achieve logarithmic leakage from the update without affecting $\rho_M$ at all. However, this complicates the proof a great deal and has marginal effect on the asymptotic behavior, so we avoid it here. In addition, if one is willing to assume the hardness of the $d$-linear assumption against $T(k)$-time adversaries, then the term $c \log k$ can be replaced with $c \log T(k)$ using essentially the same proof.

Lastly, we note that tolerating leakage of $O(\log k)$ bits *once* is trivial, since this leakage can be guessed. However, tolerating leakage of $O(\log k)$ bits *repeatedly* is significantly harder.

### 3.5.2 Proof Outline of Theorem 3.5.1

We present an outline for the proof of Theorem 3.5.1. For the sake of simplicity, we assume here that $\mathbb{G}_1 = \mathbb{G}_2 = \mathbb{G}$ and $d = 2$. The formal proof can be found in Section 3.6.

Our proof contains two main parts. First, we assume $\rho_G, \rho_U = 0$ and prove only leakage resilience between updates. Then, we will describe a semi-generic technique to achieve resilience against leakage of a logarithmic number of bits during each key update and during the key generation.

Let us start by considering a possible proof for standard CPA security of our scheme (as already sketched in the motivating discussion above). The public key $g^{\mathbf{A}}$ contains a random $2 \times \ell$ matrix (in the exponent), and the challenge ciphertext is either a linear combination of the rows of $\mathbf{A}$, or a random vector of length $\ell$. The joint distribution of the public key and ciphertext (we can consider this to be a matrix $\mathbf{A}' \in \mathbb{Z}_p^{3 \times \ell}$ whose first 2 rows are the matrix $\mathbf{A}$ and its last row is the challenge $\mathbf{v}^T$) is statistically close to either a random rank-2 matrix or a random rank-3 matrix (in the exponent), respectively. An adversary that distinguishes these distributions, therefore, immediately breaks the decisional linear assumption.

In order to make the proof work when the adversary can leak information about the secret key, we need to somehow simulate the leakage from the secret key. This seems self defeating, as having a secret key for the scheme should mean that the CPA adversary becomes useless (as we can use the secret key and decrypt the ciphertext ourselves).[4] We show that this discouraging intuition is not correct. To this end, we rely on the fact that decryption in our scheme is not perfectly correct, but instead has negligible probability of error.

Consider, at this point, the case where no update queries are made, i.e., the same secret key is used throughout the attack. At a very high level, we will show how to generate keys and a challenge ciphertext in such a way that the secret key, while being appropriately distributed respective to the

[4] A similar problem arises in most works on leakage resilience, and is addressed in various manners.

public key, is useless against our challenge ciphertext. Namely, the distribution $(pk, sk)$ is proper and the distribution $(pk, c)$ is proper, but $(pk, sk, c)$ is far from being proper, in the sense that the secret key $sk$ is useless in decrypting the specific ciphertext $c$ (specifically, $c$ will always decrypt to 0 using $sk$).[5] Hence, a CPA adversary that decrypts $c$ correctly can actually be useful in breaking the decisional linear assumption.

We then show that if the amount of leakage is sufficiently bounded, an adversary cannot gain sufficient information about the secret key $sk$ to discriminate the ciphertext $c$ from a random ciphertext, and thus should indeed decrypt $c$ correctly (with non-negligible probability). We explain this in detail below and then explain how to extend this to the continual leakage scenario where key updates occur (and thus multiple secret keys are used and are leaked during the attack).

Let us first explain how to generate keys and a challenge ciphertext in such a way that the keys are properly distributed but are still useless against the challenge ciphertext, so that a successful CPA adversary can be used to break the decisional linear assumption. An instance of the decisional linear assumption is a matrix $g^{\mathbf{C}}$, where $\mathbf{C} \in \mathbb{Z}_p^{3\times 3}$ is either rank-2 or rank-3. We will use a successful CPA adversary $\mathcal{A}$ to construct a PPT algorithm $\mathcal{B}$ that distinguishes between the case that $\mathbf{C}$ is of rank 2 and the case that $\mathbf{C}$ is of rank 3.

The algorithm $\mathcal{B}$, on input $g^{\mathbf{C}}$, does the following. It samples a random matrix $\mathbf{V} \overset{\$}{\leftarrow} \mathbb{Z}_p^{3\times \ell}$ and sets $g^{\mathbf{A}'} = g^{\mathbf{C}\cdot\mathbf{V}}$. It can then produce a matrix $\mathbf{X} \overset{\$}{\leftarrow} \ker^{\ell-3}(\mathbf{V})$. Namely, $\mathbf{X}$ is a collection of $\ell - 3$ random vectors in $\ker(\mathbf{V})$. By symmetry, these are also $\ell - 3$ random vectors in $\ker(\mathbf{A}')$.

The algorithm $\mathcal{B}$ feeds the adversary $\mathcal{A}$ with the first two rows of $g^{\mathbf{A}'}$ as the public key (denoted $g^{\mathbf{A}}$) . It generates a secret key $g^{\mathbf{y}_1}, g^{\mathbf{y}_2}$ by choosing at random $\mathbf{y}_1, \mathbf{y}_2 \overset{\$}{\leftarrow} \{\mathbf{X}\cdot\mathbf{t} : \mathbf{t} \in \mathbb{Z}_p^{\ell-3}\}$, i.e., randomly sampling from the column-span of $\mathbf{X}$. Then, it feeds $\mathcal{A}$ with the challenge ciphertext $g^{\mathbf{v}^T}$, which is the last row of $g^{\mathbf{A}'}$. Notice that our secret key will always decrypt the challenge ciphertext $g^{\mathbf{v}^T}$ to 0, even if $\mathbf{v}^T$ is independent of the rows of $\mathbf{A}$ and hence should be decrypted to 1 (thus, our secret key is "crippled" w.r.t. the ciphertext $g^{\mathbf{v}^T}$). A CPA adversary that succeeds in decrypting $c$ correctly (in spite of getting leakage from a "crippled" secret key) will, therefore, break the decisional linear assumption by determining if $\mathbf{A}'$ (and hence $\mathbf{C}$) is rank-2 or rank-3.

We next must explain why the adversary $\mathcal{A}$, which is given leakage from our "crippled" secret key, should nevertheless decrypt $c$ correctly (with noticeable probability). To this end, we use Theorem 3.4.1 to show that a small enough leakage *statistically* hides the subspace $\mathbf{X}$ that the secret key is sampled from, making it statistically hard to distinguish this distribution of the secret key and the legal distribution where $\mathbf{y}_1, \mathbf{y}_2 \overset{\$}{\leftarrow} \ker(\mathbf{A})$. Thus, the adversary cannot use the leakage to distinguish between our "crippled" secret key and a genuine one. This completes the proof for the case of non-continual leakage.

We now address the fact that the leakage is continual, namely, that there is a sequence of phases in which leakage occurs, each followed by a refresh operation. At this point, however, we still assume that the key-update procedure does not leak (i.e., $\rho_U = 0$).

While our "legal" key-update procedure never changes the column-span of the secret key (recall that the update procedure takes $sk = g^{\mathbf{Y}}$ and outputs $sk' = g^{\mathbf{Y}\cdot\mathbf{R}}$ for an invertible $\mathbf{R}$), we notice that this is computationally indistinguishable from an update procedure that re-samples new vectors $\mathbf{y}_1, \mathbf{y}_2$ from the column-span of $\mathbf{X}$ (i.e., an update procedure that sets $sk' = g^{\mathbf{X}\cdot\mathbf{T}}$, for a properly

[5] It is here that we use the fact that our scheme is not perfectly correct.

sampled matrix $\mathbf{T}$, regardless of the previous $sk$). Indistinguishability holds as the legal distribution on keys can be described by setting $sk' = g^{\mathbf{X}' \cdot \mathbf{T}}$, where $\mathbf{X}'$ is a random rank-2 matrix; thus, distinguishing $g^{\mathbf{X}}$ and $g^{\mathbf{X}'}$ is hard by the decisional linear assumption, and this holds even in the presence of the matrix $\mathbf{A}$. (Jumping ahead, it is this indistinguishability argument that becomes troublesome when leakage from the key-update procedure is allowed.) Note that this "improper" update can be simulated, since we explicitly know $\mathbf{X}$. This enables us to apply the above argument consecutively: at each phase, we will leak from new vectors in the column-span of $\mathbf{X}$. Applying Theorem 3.4.1, the view of the adversary is statistically close to the case where the leakage function is applied to $\mathbf{y}_1, \mathbf{y}_2 \overset{\$}{\leftarrow} \ker(\mathbf{A})$ which, in turn, is indistinguishable from "legally" distributed keys.

**Leakage During Key Generation and Update.** To explain how to treat leakage from the key generation and update processes (i.e., $\rho_G, \rho_U > 0$), we start by adapting our specific proof from above, and then generalize the argument to hold for all schemes with certain properties.

It may appear that any amount of leakage during the update process can completely break the indistinguishability argument of the previous paragraph. Intuitively, the adversary is getting leakage from the input to the update procedure (including the random tape used), and thus may have some information on what the legal output should be. Once the adversary sees some memory leakage from an improper secret key, used for the security reduction, it can compare it to what it knows about the legal secret key, thus catching any attempt for a switch of distributions as above. In order to overcome this barrier, we must simulate the leakage from the update in a way that makes $\mathcal{A}$ behave similarly, in spite of the discrepancy between the distributions.

The key observation is that whether or not $\mathcal{A}$ "behaves similarly" is an efficiently checkable event. We can simulate the remainder of the security game, using the proposed improper secret key and some candidate leakage value (as if everything from this point on is done legitimately) and see if $\mathcal{A}$'s success probability decreases or not. If the number of bits being leaked is small enough, specifically $O(\log k)$, we can efficiently go over all possible values until we find one that works.

There is one small problem: what if *no* leakage value works? In this case, we recall again that the improper distribution on keys used in our reduction is indistinguishable from the original one. Therefore, since in the original distribution there is always a good leakage value — the legal value — this should also be the case with the improper one. This holds since the event of not finding an acceptable leakage value is also efficiently checkable (again, by simulating $\mathcal{A}$). A change in behavior in this respect yields a distinguisher between the real and improper distributions, which is impossible under our hardness assumption. (In fact, this is only true with noticeable probability over the sampling of the secret key, and therefore we may need to try a few keys before we find a good one. This turns out to be a real problem when we try to generalize our method.)

We obtain, therefore, that leakage of $O(\log k)$ bits from the update procedure can indeed be tolerated. This can be generalized in a straightforward manner to imply a tolerable leakage of $O(\log T(k))$ bits, if we assume that the decisional linear assumption is hard for (roughly) $T(k)$-time adversaries.

Leakage from the key-generation step is handled similarly: after generating our initial secret key, we go over all possible values of key-generation leakage, and find one for which $\mathcal{A}$ works well.

The above seems like a solution that is tailored to our specific proof, since it requires the reduction to know a secret key value to simulate $\mathcal{A}$ forward. To generalize this technique, we notice that the *leakage function itself* has access to the secret key. Therefore, given a scheme that is secure with $\rho_G, \rho_U = 0$ but $\rho_M > 0$, we can use the memory leakage to simulate the adversary forward and find the logarithmic leakage value for which $\mathcal{A}$ behaves well. Unfortunately, as we remarked above, we sometimes need to try a few secret keys until we find a good one, which we cannot do generically. To this end, we strengthen the model of the attack and require that the scheme we start with (the one without leakage from the key generation and update) is also resilient against *rewinding adversaries* that are allowed to ask to update the key from some previous value. This property holds for our scheme and presumably for others.[6]

## 3.6 Formal Proof of Security

In this section, we prove Theorem 3.5.1. The proof is a formalization of the outline in Section 3.5.2. As we explain, in order to prove leakage from the key generation and update generically, we require the notion of *rewinding adversaries*. We formally define resilience against such adversaries in Section 3.6.1. In Section 3.6.2 we prove leakage resilience against rewinding adversaries, but without leakage from the update or key generation. Finally, in Section 3.6.3 we show how to trade off leakage between updates for leakage from the update and key generation.

Theorem 3.5.1 follows as a straightforward corollary from Theorem 3.6.1 (in Section 3.6.2) and Theorem 3.6.3 (in Section 3.6.3).

### 3.6.1 Continual Leakage with Rewinding Adversaries.

We define a slightly stronger notion of continual memory leakage resilience. In this new notion, the adversary is allowed to instruct the challenger to update the secret key not from the value currently in memory, but rather from some past value. We therefore call this "resilience against rewinding adversaries". A scheme that is secure in this sense will be useful for obtaining leakage resilience during the update and key generation.

**Definition 3.6.1.** *A scheme* $\mathsf{CML}$ *is* $(\rho_G, \rho_U, \rho_M)$*-leakage resilient against rewinding adversaries if any polynomial time adversary* $\mathcal{A}$ *has negligible advantage in a game identical to that in Definition 3.3.2 except for the update queries:*

- *Rewinding update queries* $(\mathsf{q\text{-}rew\text{-}update}, f, j)$*, where* $f$ *is a circuit with* $|f(sk, r, \tau)| \le \rho_U \cdot (|sk| + |r|)$ *for all* $sk, r, \tau$*, and* $j \le i$ *(if* $j > i$*, the challenger aborts). The challenger sets* $i{:=}j$ *and proceeds as before: The challenger chooses "secret randomness"* $r$ *and "public randomness"* $\tau$*, and computes* $sk_{i+1}{:=}\mathsf{CML.Update}_{pk}(sk_i; r, \tau)$*. If* $L_i + |f(sk_i, r, \tau)| \le \rho_M \cdot |sk_i|$ *then the challenger returns* $(\tau, f(sk_i, r, \tau))$ *to the adversary, sets* $i{:=}i+1$*, and sets* $L_{i+1}{:=}|f(sk_i, r, \tau)|$*. Otherwise, the challenger aborts.*

We clarify that when rewinding to secret key $j$, secret key $sk_{j+1}$ is re-written. One can think of a definition where the adversary is allowed to go back to other "branches" in history (indeed, in a followup work [LRW11] uses such a definition), but for our purposes, the above is sufficient.

Note that a rewinding update query is identical to a standard update query if $j = i$. Therefore Definition 3.6.1 is strictly stronger than Definition 3.3.2.

[6] This property will hold for any scheme where the update re-randomizes the key completely, as is the case in ours.

### 3.6.2 Security Without Leakage from Key-Generation or Update

We state and prove the resilience of our scheme against rewinding adversaries.

**Theorem 3.6.1.** *The scheme* PK-CML *is* $(0, 0, 1/d - 5/\ell)$*-secure against rewinding adversaries under the d-linear assumption.*

The proof is a formalization of the outline from Section 3.5.2. However, since we prove by a series of hybrids, the proof goes in "reverse order" compared to the outline. At a high level, we first show that instead of updating by re-randomizing the initial key, we can sample new keys uniformly from the kernel of the public key. Then we use Theorem 3.4.1 to argue that this is indistinguishable from sampling the keys uniformly in the kernel of the matrix $\mathbf{A}'$ (which is a concatenation of the public key and the challenge ciphertext). At this point we have secret keys that are "powerless" against our ciphertext. The next step is going back to sampling not uniformly in the kernel of $\mathbf{A}'$, but from a small subspace of the kernel, similarly to the original scheme (only now we are in the kernel of $\mathbf{A}'$ and not $\mathbf{A}$). At this point, we can use our adversary to break the $d$-linear assumption as our outline suggests.

*Proof.* Consider a rewinding $(0, 0, 1/d - 5/\ell)$-adversary $\mathcal{A}$ that plays the CML security game with the scheme PK-CML. We assume w.l.o.g that $\mathcal{A}$ makes alternating q-rew-update and q-mem queries (the q-rew-update queries are made with empty leakage function). Let $t$ be a polynomial upper bound on the number of q-mem queries made by $\mathcal{A}$. Given that the secret key of the scheme is $\ell d \log p$ bits long, the bit length of each q-mem query is at most

$$\lambda = (1/d - 5/\ell) \cdot (\ell d \log p) = (\ell - 5d) \log p \le (\ell - 3d - 2) \log p \ . \tag{3.3}$$

Let us introduce some notation that will be used throughout the proof. Our proof works by a series of hybrids. For hybrid $H$ we let $\mathrm{Adv}_H[\mathcal{A}]$ denote the advantage of $\mathcal{A}$ when playing in that hybrid (i.e. $|\Pr[b' = b] - 1/2|$). The public key will be denoted $pk = g_1^{\mathbf{A}}$ and the secret key on which $\mathcal{A}$ makes the $j^{\text{th}}$ leakage query is denoted $g_2^{\mathbf{Y}_j}$. Note that this element may not be identical to $sk_j$ due to rewinding queries. We do not reserve any special notation for $sk_i$ (we will see quite early in the argument that this is not necessary). Finally, we let $g_1^{\mathbf{v}^T}$ denote the challenge ciphertext. Using this notation, the view of the adversary is always of the form

$$\mathsf{view}[\mathcal{A}] \triangleq \left(g_1^{\mathbf{A}}, w_1 = f_1(g_2^{\mathbf{Y}_1}), \ldots, w_t = f_t(g_2^{\mathbf{Y}_t}), g_1^{\mathbf{v}^T}\right) \ , \tag{3.4}$$

where the functions $f_j$ are chosen adaptively, and the bit-length of each $w_j$ is at most $\lambda$. We also denote $\mathsf{view}_H[\mathcal{A}]$ to indicate the view of $\mathcal{A}$ in hybrid $H$.

Some of our hybrids will be statistical, and we would like to consider a "statistical view" where we consider the discrete logarithms of the group elements

$$\mathsf{stat\text{-}view}[\mathcal{A}] \triangleq (\mathbf{A}, w_1 = f_1(\mathbf{Y}_1), \ldots, w_t = f_t(\mathbf{Y}_t), \mathbf{v}^T) \ . \tag{3.5}$$

Clearly, $\mathsf{view}_H[\mathcal{A}]$ is a (deterministic) function of $\mathsf{stat\text{-}view}_H[\mathcal{A}]$.

We can now present our series of hybrids.

- **Hybrid $H_0$.** This hybrid is identical to the CML game. By definition
$$\mathrm{Adv}_{H_0}[\mathcal{A}] = \mathrm{Adv}[\mathcal{A}] \ .$$

- **Hybrid $H_1$.** This hybrid is technical and carries no important insight. Essentially, we rearrange the probability space in a way that is easier to manipulate later. Recall that in the previous hybrid, $\mathbf{Y}_0 \stackrel{\$}{\leftarrow} \ker^d(\mathbf{A})$ and $\mathbf{Y}_j{:=}\mathbf{Y}_{j'} \cdot \mathbf{R}_j$, for $\mathbf{R}_j \stackrel{\$}{\leftarrow} \mathrm{Rk}_d(\mathbb{Z}_p^{d\times d})$ and some $j' < j$ (note that $j'$ will not equal $j-1$ in case of rewinding). In this hybrid, we sample one matrix $\mathbf{Y} \stackrel{\$}{\leftarrow} \mathrm{Rk}_d(\ker^d(\mathbf{A}))$ and for all $j$, we set $\mathbf{Y}_j{:=}\mathbf{Y} \cdot \mathbf{R}_j$, for $\mathbf{R}_j \stackrel{\$}{\leftarrow} \mathbb{Z}_p^{d\times d}$.

  The key observation is that in both $H_0$ and $H_1$, the matrices $\mathbf{Y}_j$ are (up to a $O(1/p)$ statistical distance) random samples from a (dimension $d$) subspace of $\ker(\mathbf{A})$ that is spanned by the columns of $\mathbf{Y}_1$. Specifically, if both $\mathbf{Y}_1$ of the hybrid $H_0$ and all $\mathbf{R}_j$'s of the hybrid $H_1$ are full rank, then the two hybrids are identical. Since this happens with all but $O(t/p)$ probability, we have that
$$|\mathrm{Adv}_{H_0}[\mathcal{A}] - \mathrm{Adv}_{H_1}[\mathcal{A}]| \leq O(t/p) \ .$$

- **Hybrid $H_2$.** In this hybrid, secret keys are re-sampled independently in the kernel of $\mathbf{A}$, rather than as rotations of the same key. Formally, we again change the way we compute $\mathbf{Y}_j$. Now we will set $\mathbf{Y}_j{:=}\mathbf{Z}_j$, where $\mathbf{Z}_j \stackrel{\$}{\leftarrow} \ker^d(\mathbf{A})$. Namely, the new $\mathbf{Y}_j$ values will be chosen not from the same dimension $d$ subspace of the kernel, but rather completely uniformly in the span of the kernel.

  Hybrids $H_1$, $H_2$ are computationally indistinguishable (under $d$-linear) since $\mathcal{A}$'s view depends only on $g_2^{\mathbf{Y}_j}$ (and not on $\mathbf{Y}_j$ itself). The $d$-linear assumption asserts that all spaces of dimension at least $d$, taken to the exponent, are indistinguishable from uniform. The formal argument is provided in Lemma 3.6.2 below. Thus
$$|\mathrm{Adv}_{H_1}[\mathcal{A}] - \mathrm{Adv}_{H_2}[\mathcal{A}]| \leq d\mathrm{LINAdv}[\mathcal{B}] \ ,$$
  where $d\mathrm{LINAdv}[\mathcal{B}]$ is the distinguishing advantage of some efficient $\mathcal{B}$ over $d$-linear distributions.

- **Hybrid $H_3$.** In this step we go from meaningful to meaningless secret keys. Once again, we change the distributions of $\mathbf{Y}_j$. We define
$$\mathbf{A}' \triangleq [\mathbf{A}^T \| \mathbf{v}]^T = \left[\begin{array}{c}\mathbf{A}\\ \hline \mathbf{v}^T\end{array}\right] \ ,$$
  a concatenation of the public key and the challenge ciphertext, and let $\mathbf{Y}_j{:=}\mathbf{Z}'_j$ where $\mathbf{Z}'_j \stackrel{\$}{\leftarrow} \ker^d(\mathbf{A}')$. The secret keys in this hybrid cannot decrypt the ciphertext (they will always output 0).

  Our indistinguishability here is statistical. Therefore, it is sufficient to prove for the case where $\mathbf{v}^T \notin \mathrm{Span}(\mathbf{A})$ (otherwise the distributions are identical). We further notice that the conditional distribution of $\mathbf{v}^T$ is uniform (over the respective domain $\mathbf{v}^T \notin \mathrm{Span}(\mathbf{A})$).

We use Theorem 3.4.1 to show that $H_2$ and $H_3$ are statistically indistinguishable in this case. Intuitively, $\ker(\mathbf{A})$ will correspond to the ambient space $\mathbb{Z}_p^m$ of the theorem, and $\ker(\mathbf{A}')$ is a random subspace of dimension $\ell = m - 1$. The matrices $\mathbf{Z}_j$ will thus correspond to uniformly sampling in the ambient space (hence to the matrix $\mathbf{U}$ in the theorem statement), while the matrices $\mathbf{Z}'_j$ will correspond to $\mathbf{X} \cdot \mathbf{T}$ since they are uniformly sampled from the subspace. A formal description follows.

We want to bound $\mathtt{dist}(\mathsf{stat\text{-}view}_{H_2}[\mathcal{A}], \mathsf{stat\text{-}view}_{H_3}[\mathcal{A}])$. A straightforward hybrid argument implies that for a properly defined distribution $D$,

$$\mathtt{dist}(\mathsf{stat\text{-}view}_{H_2}[\mathcal{A}], \mathsf{stat\text{-}view}_{H_3}[\mathcal{A}]) \leq t \cdot \mathtt{dist}\left((\mathbf{A}, w = f(\mathbf{Z}), \mathbf{v}^T), (\mathbf{A}, w = f(\mathbf{Z}'), \mathbf{v}^T)\right) ,$$

with $w$ being at most $\lambda$ bits long, $\mathbf{Z} \xleftarrow{\$} \ker^d(\mathbf{A})$ (which corresponds to the distributions of $\mathbf{Y}_j$ in $H_2$) and $\mathbf{Z}' \xleftarrow{\$} \ker^d(\mathbf{A}')$ (corresponding to the distribution in $H_3$).

Let $\mathbf{B} \in \mathbb{Z}_p^{\ell \times m}$ be some basis for $\ker(\mathbf{A})$, where $m$ is the dimension of the kernel ($m \geq \ell - d$). Let $\mathbf{X}$ be such that $\mathbf{B} \cdot \mathbf{X}$ is a random basis for $\ker(\mathbf{A}')$. Then by symmetry, $\mathbf{X}$ is uniform in $\mathrm{Rk}_{m-1}(\mathbb{Z}_p^{m \times (m-1)})$.

Using the new notation, we need to bound

$$\mathtt{dist}\left((\mathbf{A}, w = f(\mathbf{B} \cdot \mathbf{U}), \mathbf{X}), (\mathbf{A}, w = f(\mathbf{B} \cdot \mathbf{X} \cdot \mathbf{T}), \mathbf{X})\right) , \tag{3.6}$$

where $\mathbf{X} \xleftarrow{\$} \mathrm{Rk}_{m-1}(\mathbb{Z}_p^{m \times (m-1)})$, $\mathbf{U} \xleftarrow{\$} \mathbb{Z}_p^{m \times d}$, $\mathbf{T} \xleftarrow{\$} \mathbb{Z}_p^{(m-1) \times d}$. Note that $\mathbf{v}^T$ does not appear anymore since it is a (randomized) function of $\mathbf{A}, \mathbf{X}$.

To apply Theorem 3.4.1, we change the distributions of $\mathbf{X}$ and $\mathbf{T}$ to $\mathbf{X} \xleftarrow{\$} \mathbb{Z}_p^{m \times (m-1)}$ and $\mathbf{T} \xleftarrow{\$} \mathrm{Rk}_d(\mathbb{Z}_p^{(m-1) \times d})$. This increases the statistical distance by at most $O(1/p)$ (by Lemma 3.1.1). We can then apply Theorem 3.4.1 with $n = m - 1 \geq 2d$, $\epsilon = 1/p$ and (recalling Eq. (3.3))

$$|W| \leq 2^\lambda \leq 2^{(\ell - 3d - 2)\log p} = p^{n-(2d-1)} \cdot (1/p)^2 \leq 4 \cdot (1 - 1/p) \cdot p^{n-(2d-1)} \cdot \epsilon^2 .$$

It follows that

$$\mathtt{dist}\left((\mathbf{A}, w = f(\mathbf{B} \cdot \mathbf{U}), \mathbf{X}), (\mathbf{A}, w = f(\mathbf{B} \cdot \mathbf{X} \cdot \mathbf{T}), \mathbf{X})\right) \leq O(1/p) ,$$

and thus that

$$|\mathrm{Adv}_{H_2}[\mathcal{A}] - \mathrm{Adv}_{H_3}[\mathcal{A}]| \leq O(t/p) .$$

- **Hybrid** $H_4$**.** We change the secret keys to be sampled from only a subspace of $\ker(\mathbf{A}')$. To this end, we change the distribution of $\mathbf{Y}_j$ for the last time. We sample one matrix $\mathbf{Y}' \xleftarrow{\$} \mathrm{Rk}_d(\ker^d(\mathbf{A}'))$, and let $\mathbf{Y}_j := \mathbf{Y}' \cdot \mathbf{R}_j$, where $\mathbf{R}_j \xleftarrow{\$} \mathbb{Z}_p^{d \times d}$.

  This is very similar to what we did in hybrid $H_2$ and indeed the same Lemma 3.6.2 (now working on $\mathbf{A}'$ instead of $\mathbf{A}$) proves that

$$|\mathrm{Adv}_{H_3}[\mathcal{A}] - \mathrm{Adv}_{H_4}[\mathcal{A}]| \leq d\mathrm{LINAdv}[\mathcal{B}'] ,$$

  for some adversary $\mathcal{B}'$.

- **Hybrid $H_5$.** We are a small technical change from being done. We slightly change the way $\mathbf{A}, \mathbf{v}^T$ are sampled to make things easier for the next hybrid.

  In the case where $b = 0$, we will now sample $\mathbf{C}_0 \overset{\$}{\leftarrow} \mathrm{Rk}_d(\mathbb{Z}_p^{(d+1)\times(d+1)})$ and $\mathbf{V} \overset{\$}{\leftarrow} \mathbb{Z}_p^{(d+1)\times \ell}$, and set $\mathbf{A}' := \mathbf{C}_0 \cdot \mathbf{V}$. This changes the distribution of $\mathbf{A}'$ by statistical distance $O(1/p)$.

  In the case where $b = 1$, we will now sample $\mathbf{C}_1 \overset{\$}{\leftarrow} \mathrm{Rk}_{d+1}(\mathbb{Z}_p^{(d+1)\times(d+1)})$, and as before $\mathbf{V} \overset{\$}{\leftarrow} \mathbb{Z}_p^{(d+1)\times \ell}$, and set $\mathbf{A}' \leftarrow \mathbf{C}_1 \cdot \mathbf{V}$. Again, the resulting matrix is $O(1/p)$-close to the previous case.

  Thus
  $$|\mathrm{Adv}_{H_4}[\mathcal{A}] - \mathrm{Adv}_{H_5}[\mathcal{A}]| \le O(1/p) \ .$$

  Recall that this is how we started our outline, the matrix $\mathbf{A}'$ now becomes a product of a rank $d$ or $d+1$ matrix, times a randomizer $\mathbf{V}$ and $\mathbf{Y}'$ becomes uniform in $\mathrm{Rk}_d(\ker^d(\mathbf{V}))$.

  Finally, we notice that given $g_1^{\mathbf{C}_b}$, we can sample $\mathbf{V}, \mathbf{Y}', \mathbf{R}_j$ and generate the entire view of $\mathcal{A}$ by ourselves. Therefore the advantage of $\mathcal{A}$ in $H_5$ is an advantage of distinguishing $g_1^{\mathbf{C}_0}$ and $g_1^{\mathbf{C}_1}$ and therefore
  $$\mathrm{Adv}_{H_5}[\mathcal{A}] \le d\mathrm{LINAdv}[\mathcal{B}''] \ ,$$
  for some adversary $\mathcal{B}''$.

Combining all the hybrids, the result follows. □

**Lemma 3.6.2.** *Let $\mathbb{G}$ be a d-linear hard group of order $p$ with generator $g$. Let $n, \ell$ be integers such that $\ell \ge n + d$ and let $\mathbf{A} \in \mathbb{Z}_p^{n\times \ell}$ be any matrix. Let $\mathbf{Y} \overset{\$}{\leftarrow} \mathrm{Rk}_d(\ker^d(\mathbf{A}))$. Then for any polynomial $t$, under the d-linear assumption it holds that the distributions*
$$(\mathbf{A}, g^{\mathbf{Y}\cdot\mathbf{R}_1}, \ldots, g^{\mathbf{Y}\cdot\mathbf{R}_t}) \quad \textit{and} \quad (\mathbf{A}, g^{\mathbf{Z}_1}, \ldots, g^{\mathbf{Z}_t})$$
*are computationally indistinguishable, where $\mathbf{R}_i \overset{\$}{\leftarrow} \mathbb{Z}_p^{d\times d}$, $\mathbf{Z}_i \overset{\$}{\leftarrow} \ker^d(\mathbf{A})$.*

*Proof.* Letting $\mathbf{R} = [\mathbf{R}_1 \| \cdots \| \mathbf{R}_t] \in \mathbb{Z}_p^{d\times dt}$ and $\mathbf{Z} = [\mathbf{Z}_1 \| \cdots \| \mathbf{Z}_t] \in \mathbb{Z}_p^{d\times dt}$, the aforementioned distributions can be expressed as
$$(\mathbf{A}, g^{\mathbf{Y}\cdot\mathbf{R}}) \quad \text{and} \quad (\mathbf{A}, g^{\mathbf{Z}}) \ .$$

Let $\mathbf{B} \in \mathbb{Z}_p^{\ell\times m}$ be a basis for $\ker(\mathbf{A})$, where $m$ is the dimension of the kernel of $\mathcal{A}$ (so $m \ge \ell - n \ge d$). Clearly $\mathcal{B}$ is easy to compute given $\mathbf{A}$. Then

- $\mathbf{Y} \cdot \mathbf{R}$ is distributed identically to $\mathbf{B} \cdot \mathbf{C}_1 \cdot \mathbf{R}'$ where $\mathbf{C}_1 \overset{\$}{\leftarrow} \mathrm{Rk}_d(\mathbb{Z}_p^{m\times m})$ and $\mathbf{R}' \overset{\$}{\leftarrow} \mathbb{Z}_p^{m\times dt}$.

- $\mathbf{Z}$ is distributed identically to $\mathbf{B} \cdot \mathbf{C}_1 \cdot \mathbf{R}'$ where $\mathbf{C}_2 \overset{\$}{\leftarrow} \mathrm{Rk}_m(\mathbb{Z}_p^{m\times m})$ and as before $\mathbf{R}' \overset{\$}{\leftarrow} \mathbb{Z}_p^{m\times dt}$.

It follows that a distinguisher between the above distributions implies a distinguisher between $g^{\mathbf{C}_1}$ and $g^{\mathbf{C}_2}$: This is done by computing $\mathbf{B}$, sampling $\mathbf{R}'$ and obtaining either of the above distributions, depending on whether $g^{\mathbf{C}_1}$ or $g^{\mathbf{C}_2}$ was given as input. □

### 3.6.3 Leakage from Key Generation and Update: A Generic Reduction

Finally, we provide our semi-generic proof for trading off $\rho_M$ for $\rho_G, \rho_U$. Note that in the transition we lose the resilience for rewinding adversaries.

**Theorem 3.6.3.** *Let* $\mathsf{CML}$ *be* $(0, 0, \rho_M)$*-leakage resilient against rewinding adversaries and let* $(s_G, s_U, s_M)$ *be upper bounds on the size of the secret memory during initialize, update, and leak operations, respectively.*

*Then for all* $c > 0$*,* $\mathsf{CML}$ *is also* $\left(\frac{c \log k}{s_G}, \frac{c \log k}{s_U}, \rho_M - \frac{c \log k}{s_M}\right)$*-leakage resilient against* non-rewinding *adversaries.*

The proof essentially follows the outline in Section 3.5.2, we use the leakage function to estimate the success probability of the adversary, and make sure that it doesn't drop by too much over the experiment.

*Proof.* Let $\mathsf{CML}$ be as above and let $c > 0$. Consider a $\left(\frac{c \log k}{s_G}, \frac{c \log k}{s_U}, \rho_M - \frac{c \log k}{s_M}\right)$ non-rewinding adversary $\mathcal{A}$ against $\mathsf{CML}$ (assume w.l.o.g that $\mathcal{A}$ never causes abortions). Let $t$ be a polynomial upper bound on the running time of $\mathcal{A}$ and let $\epsilon$ denote its advantage. We show a $(0, 0, \rho_M)$ rewinding adversary $\mathcal{B}$ for $\mathsf{CML}$ with success probability $(\epsilon/4) \cdot (\epsilon/2 - \mathrm{poly}(k) \cdot 2^{-k})$ and running time $\mathrm{poly}(k, t, 1/\epsilon)$.

The adversary $\mathcal{A}$ makes at most $t$ queries during its run (including the key generation leakage). We assume w.l.o.g that the first query is the leakage from the key generation, that query $2i$ (for all $i$) is a $\mathsf{q\text{-}mem}$ query, and query $2i + 1$ is a $\mathsf{q\text{-}update}$ query (recall that $\mathcal{A}$ is not allowed rewinding queries). So we have a sequence that starts with key generation query and continues with alternating memory and update queries. We further assume that the key generation and all update queries return $c \log k$ bits, and that all $\mathsf{q\text{-}mem}$ queries return $\lambda$ bits for some integer $\lambda$.

We define values $A_{pk, sk_i, \mathsf{state}}$, parameterized by a public key, the current secret key and the state of the adversary $\mathcal{A}$, which contains its random tape and all information $\mathcal{A}$ saw so far. We let $A_{pk, sk_i, \mathsf{state}}$ denote the advantage of $\mathcal{A}$ conditioned on $pk, sk_i, \mathsf{state}$. Namely, the advantage of $\mathcal{A}$ if the game continues "fairly" from this point and on. Note that since $\mathcal{A}$ is not rewinding, this value depends only on $sk_i$ and not on past secret keys. We stress that $A_{pk, sk_i, \mathsf{state}}$ is not a random variable and its value is deterministically determined by $pk, sk_i, \mathsf{state}$.

Our adversary $\mathcal{B}$ will need to estimate the value of $A_{pk, sk_i, \mathsf{state}}$. The next claim shows that this can be done efficiently.

**Claim 3.6.3.1.** *Given* $(pk, sk_i, \mathsf{state})$*, and a parameter* $\delta$*, there is a probabilistic algorithm that runs in time* $\mathrm{poly}(k, t, 1/\delta)$ *and computes a value* $\hat{A}^{(\delta)}_{pk, sk_i, \mathsf{state}}$ *such that with all but* $2^{-k}$ *probability on the random coins,*

$$\left| A_{pk, sk_i, \mathsf{state}} - \hat{A}^{(\delta)}_{pk, sk_i, \mathsf{state}} \right| \leq \delta \ .$$

*Proof.* Naturally, the claim is proven by the (additive) Chernoff bound. Let $X$ be a random variable that continues the execution of the game between $\mathcal{A}$ and the challenger, and outputs 1 if $\mathcal{A}$ outputs $b' = b$. Then by definition

$$A_{pk, sk_i, \mathsf{state}} = \left| \mathbb{E}[X] - \frac{1}{2} \right| \ .$$

Furthermore, $X$ is efficiently sampleable. Therefore our algorithm will generate $n = \lceil (k + \ln 2)/(2\delta^2) \rceil$ independent samples of $X$, denoted $x_1, \ldots, x_n$, and will return

$$\hat{A}^{(\delta)}_{pk,sk_i,\mathsf{state}} := \left| \frac{1}{n} \sum_{i \in [n]} x_i - \frac{1}{2} \right| .$$

The additive Chernoff bound implies that

$$\Pr\left[ \left| \frac{1}{n} \sum_{i \in [n]} x_i - \mathbb{E}[X] \right| > \delta \right] \leq 2e^{-2\delta^2 n} \leq 2^{-k} ,$$

and the claim follows. ■

We can finally define the rewinding adversary $\mathcal{B}$.

1. $\mathcal{B}$ samples a random tape for $\mathcal{A}$. Upon receiving the public key $pk$, $\mathcal{B}$ will forward it to $\mathcal{A}$.
2. When $\mathcal{A}$ wants to receive the leakage value from the key generation, $\mathcal{B}$ will make a $(\mathsf{q\text{-}mem}, f)$ query, with the following function $f$:

   On input $pk, sk_0$, the function $f$ will go over all values $\alpha \in \{0,1\}^{c \log k}$. For each such value, it will compute $\hat{A}^{(\epsilon/(4t))}_{pk,sk_0,\mathsf{state}}$, where $\mathsf{state}$ is the state of $\mathcal{A}$, assuming it got $\alpha$ as the response to the leakage query. If there exists an $\alpha$ such that the estimated value is at least $\epsilon/2 - \epsilon/(4t)$, then this $\alpha$ is returned, otherwise, $f$ returns $\perp$.

   The adversary $\mathcal{B}$ gets the output of this query, i.e. the value $f(pk, sk_0)$. If $\perp$ is returned, then $\mathcal{B}$ aborts. Otherwise, it returns the output $\alpha$ to $\mathcal{A}$, as if it was the answer to the key generation leakage query.

   Note that this means that if $\perp$ was not returned, the conditional advantage of $\mathcal{A}$ (with all but exponentially small probability) is at least $\epsilon/2 - \epsilon/(2t)$.
3. On the $(2i)^{\text{th}}$ query of $\mathcal{A}$, which is a $\mathsf{q\text{-}mem}$ query, $\mathcal{B}$ forwards this query to the challenger and forwards the answer back to $\mathcal{A}$.
4. On the $(2i+1)^{\text{th}}$ query of $\mathcal{A}$ $(i \geq 1)$, which is a $\mathsf{q\text{-}update}$ query, $\mathcal{B}$ does the following.
   (a) $\mathcal{B}$ makes a $(\mathsf{q\text{-}rew\text{-}update}, \perp, i-1)$ query to the challenger. Namely, requests that the secret key is updated from $sk_{i-1}$. It then makes the following $(\mathsf{q\text{-}mem}, f)$ query. The leakage function $f$ will again go over all possible leakage values $\alpha \in \{0,1\}^{c \log k}$ and compute $\hat{A}^{(\epsilon/(6t))}_{pk,sk_i,\mathsf{state}}$ as if the value $\alpha$ was the answer to the update query that $\mathcal{A}$ just made.

   If there exists an $\alpha$ for which the estimated value is at least $(\epsilon/2 - i\epsilon/(2t) - 2\epsilon/(6t))$, then $f$ outputs this $\alpha$, otherwise $\perp$ is output.

   (b) If $\perp$ is returned, then $\mathcal{B}$ goes back to step 4a. Otherwise, it forwards $\alpha$ to $\mathcal{A}$ as the answer to the query. Note that in such case it holds (with all but exponentially small probability) that the conditional advantage of $\mathcal{A}$ from this point and on is at least $(\epsilon/2 - (i+1)\epsilon/(2t))$.

(c) If $\perp$ has been returned for more than $k \cdot (6t/\epsilon)$ times, then $\mathcal{B}$ aborts.

5. After all $t$ queries have been carried out, if $\mathcal{B}$ hasn't aborted, it holds that with all but exponentially small probability, the conditional advantage of $\mathcal{A}$ is at least

$$\frac{\epsilon}{2} - \frac{t}{2} \cdot \frac{\epsilon}{2t} = \frac{\epsilon}{4} \ .$$

Now it's time for the challenge phase: $\mathcal{A}$ outputs $m_0, m_1$ and asks for a challenge. When this happens, $\mathcal{B}$ forwards the messages to the challenger, gets the challenge, forwards that to $\mathcal{A}$ and returns the same $b'$ that $\mathcal{A}$ returned.

From the presentation above, it is clear that conditioned on $\mathcal{B}$ not aborting, the advantage of $\mathcal{B}$ is at least $\epsilon/4$. It is also clear that $\mathcal{B}$ makes $\mathsf{q\text{-}mem}$ queries that are $c \log k$ bits longer than $\mathcal{A}$'s queries (in order to obtain the value $\alpha$). All that remains is bounding the abortion probability.

**Claim 3.6.3.2.** *With probability at least* $\left(\epsilon/2 - \mathrm{poly}(k) \cdot 2^{-k}\right)$*,* $\mathcal{B}$ *does not abort.*

*Proof.* We start by assuming that all estimations of $\hat{A}$ indeed succeed up to the prescribed $\delta$. Since we do a polynomial number of estimations, the probability that this doesn't hold is at most $\mathrm{poly}(k) \cdot 2^{-k}$. Let us now go over all the places where $\mathcal{B}$ might abort.

- Step 2. Here we use a Markov type argument. With probability at least $\epsilon/2$ over the random tape of $\mathcal{A}$ and the random tape of the key generation, the success probability of $\mathcal{A}$ conditioned on these values is at least $\epsilon/2$.

  Namely, with probability at least $\epsilon/2$, it holds that there exists $\alpha$ for which $A_{pk,sk_0,\mathsf{state}} \geq \epsilon/2$. This $\alpha$, in particular, will have $\hat{A} \geq \epsilon/2 - \epsilon/(4t)$ and therefore $\mathcal{B}$ will not abort.

  Note that if $\mathcal{B}$ doesn't abort, the conditional success probability of $\mathcal{A}$ is at least $\epsilon/2 - \epsilon/(4t) - \epsilon/(4t) = \epsilon/2 - \epsilon/(2t)$.

- Step 4. We assume inductively that when we get to this step, it holds that the conditional success probability of $\mathcal{A}$ is at least $\epsilon/2 - i\epsilon/(2t)$ (note that this indeed holds for $i = 1$).

  A Markov type argument implies that with probability $\epsilon/(6t)$ over the randomness of the update from $sk_{i-1}$ to $sk_i$, the conditional success probability of $\mathcal{A}$ will be at least $\epsilon/2 - i\epsilon/(2t) - \epsilon/(6t)$. If we indeed "hit" the correct randomness for the update, $f$ will not return $\perp$. It follows that a $\perp$ is returned with probability at most $1 - \epsilon/(6t)$. Repeating for $k \cdot (6t/\epsilon)$ guarantees that the probability for abort is at most $2^{-k}$. Furthermore, in case there is no abort, the conditional success probability of $\mathcal{A}$ is at least $\epsilon/2 - i\epsilon/(2t) - 2\epsilon/(6t) - \epsilon/(6t) = \epsilon/2 - (i+1)\epsilon/(2t)$, and the invariant continues to hold.

By the union bound, the probability of non-aborting is at least $\epsilon/2 - \mathrm{poly}(k) \cdot 2^{-k}$. ■

□

# Chapter 4

# Circular Security and Leakage Resilience from Subgroup Indistinguishability

The “classical” definition of *semantic secure* public-key encryption by Goldwasser and Micali [GM82, GM84], requires that an efficient attacker with access to the public encryption key must not be able to find two messages such that it can distinguish a random encryption of one from a random encryption of the other. Numerous candidate public-key encryption schemes that meet this definition have been presented over the years, both under specific hardness assumptions (like the hardness of factoring) and under general assumptions (such as the existence of injective one-way trapdoor functions).

This notion of security, however (as well as other commonly accepted ones), does not capture certain situations that may occur in the “real world”:

- Functions of the secret decryption key can be encrypted and sent (note that semantic security only guarantees security with respect to messages which an efficient attacker can find).
- Information about the secret key may leak.
- The same secret key may be used in more than one application, or more generally, the attacker can somehow obtain the value of a hard-to-invert function of the secret key.

In recent years, extensive research effort has been invested in providing encryption schemes which are provably secure even in the above settings. Such schemes are said to achieve *key-dependent message (*KDM*) security*, *leakage-resilience*, and *auxiliary-input security*, respectively to the aforementioned real world settings. Prior to our work, we knew of: (1) Candidate schemes which are KDM secure under the decisional Diffie-Hellman (DDH) and under the learning with errors (LWE) assumptions; (2) Candidate schemes that are resilient to key leakage of rate $(1 - o(1))$ (relative to the length of the secret key), under the LWE assumption and under the DDH assumption. In addition, we knew of candidate scheme achieving limited leakage resilience based on a general primitive: universal hash-proof systems, where the leakage rate depended on the hash proof system being used (in particular, under the quadratic residuosity assumption, a leakage rate of only $o(1)$ followed); (3) Candidate schemes that are auxiliary input secure under the DDH assumption and under the LWE assumption.

In this chapter, we present an encryption scheme under what we call the *subgroup indistinguishability assumption*, of which quadratic residuosity (QR) and decisional composite residuosity (DCR) are special cases, that has:

1. *Key-dependent message security* (circular security). Achieves security even when encrypting affine functions of its own secret key (in fact, w.r.t. affine "key-cycles" of predefined length).[1]

2. *Leakage resilience.* Remains secure even if any adversarial low-entropy (efficiently computable) function of the secret key is given to the adversary. A proper selection of parameters allows for a "leakage rate" of $(1 - o(1))$ of the length of the secret key.

3. *Auxiliary-input security.* Remains secure even if any sufficiently *hard to invert* (efficiently computable) function of the secret key is given to the adversary.

In addition, our schemes have the following interesting property: the secret key consists of a randomly chosen binary vector independent of the group at hand.[2] The instantiation of our scheme under QR enjoys the same useful properties for protocol design as the original [GM82, GM84] scheme, including re-randomization of ciphertexts and support of the XOR homomorphic operation over the $\{0, 1\}$ message space, with the added benefit of leakage resilience.

**Chapter Organization.** To best describe our results, we start, in Section 4.1, by surveying in detail the background for the new work. This is followed by an overview in Section 4.2, which provides a high level description of our contributions (Section 4.2.1) and techniques (Section 4.2.2), as well as related and follow-up work (Section 4.2.3).

Preliminaries and definitions are presented in Section 4.3. The definition of subgroup indistinguishability assumptions and instantiations from QR and DCR appear in Section 4.4.

Our QR-based encryption scheme is presented in Section 4.5, followed, in Section 4.6, by introduction of the *adaptive vector game*: a central technical tool to be used for the analysis throughout this chapter. The security of our QR based scheme is analyzed in Section 4.7: We start with CPA security in Section 4.7.1, KDM-security is discussed in Section 4.7.2, leakage-resilience in Section 4.7.3 and auxiliary-input security in Section 4.7.4. Since the QR assumption is a special case of subgroup indistinguishability assumptions, we take the liberty to not always provide full proofs in this part, as those can be derived as a special case of the following Section 4.8.

The latter contains the generalization of all results to any subgroup indistinguishability assumption, together with the full proofs. In addition, this last section contains an interesting generalization of our scheme to non-binary secret key distributions.

[1]Our scheme also meets the requirements for extending key-dependent message security to broader classes of functions beyond affine functions using previous techniques of [BGK11, BHHI10, App11].

[2]One can also use a non-binary vector as secret key, subject to some restrictions.

## 4.1 Background

**Key-dependent messages.** The shortcoming of the standard security definition in the case where the plaintext to be encrypted depends on the secret key was already noticed in [GM82]. It was later observed that this situation is not so unlikely, and may sometimes even be desirable [CL01, ABHS05, LC03]. Black, Rogoway and Shrimpton [BRS02] formally defined KDM-security: the attacker can obtain encryptions of (efficient) functions of its choosing, taken from some specified class of functions $\mathcal{F}$, applied to the secret key. The requirement is that the attacker cannot tell if all of its queries are answered by encryptions of some constant symbol 0, instead of the requested values. This definition is extended to the case of many (say $n$) users that can encrypt each others' secret keys: the attacker queries now contain functions to be applied to *all* secret keys, and an identity of the user whose public key should be used to encrypt. This latter case is referred to as KDM$^{(n)}$-security, while the single-user case is called KDM$^{(1)}$-security.

Boneh, Halevi, Hamburg and Ostrovsky [BHHO08] constructed a public key encryption scheme that is KDM$^{(n)}$ secure w.r.t. all affine functions,[3] under the decisional Diffie-Hellman (DDH) assumption, for any polynomial $n$. This first result was followed by the work of Applebaum, Cash, Peikert and Sahai [ACPS09] who proved that a variation of Regev's scheme [Reg05] is also KDM secure w.r.t. all affine functions, under the learning with errors (LWE) assumption.

More recent works by Brakerski, Goldwasser and Kalai [BGK11] and by Barak, Haitner, Hofheinz and Ishai [BHHI10] presented each general and different techniques to extend KDM-security to richer classes of functions. The work of [BHHI10] was later generalized and greatly simplified by Applebaum [App11], as we explain below. In [BGK11], the notion of *entropy-$\kappa$* KDM-security is introduced. A scheme is entropy-$\kappa$ KDM-secure if it remains KDM-secure even if the secret key is sampled from a high-entropy distribution, rather than a uniform one. They show that an entropy-$\kappa$ KDM-secure scheme implies a scheme that is KDM-secure w.r.t. roughly any pre-defined set of functions of polynomial cardinality. In [BHHI10], the notion of *targeted public-key encryption* is introduced and is shown to imply a KDM-secure scheme w.r.t. all functions computable by circuits of some predefined (polynomial) size. In a follow-up, [App11] presented a general completeness theorem for KDM security, which implies that the results of [BHHI10] (and more) are achievable from any scheme that is KDM-secure w.r.t. affine functions of the secret key. Thus targeted encryption was no longer necessary.

These two approaches achieve incomparable performance. While in the former, the public key and ciphertext lengths depend on the size of the function class (but not on its complexity) and are independent of the number of users $n$, in the latter the public key size does not depend on the function class, but the ciphertext length is linear in the product of $n$ times the complexity of the functions.

**Leakage resilience.** The work on cold boot attacks by Halderman et al. [HSH$^+$08], gave rise to the notion of public-key encryption resilient to (bounded) memory leakage attacks, presented by Akavia, Goldwasser and Vaikuntanathan [AGV09] and further explored by Naor and Segev [NS09]. In their definition, security holds even if the attacker gets some information of its choosing (de-

[3]More precisely "affine in the exponent": the secret key is a vector of group elements $g_1, \ldots, g_\ell$ and the scheme is secure w.r.t. functions of the form $h \cdot \prod g_i^{a_i}$.

pending on the value of the public key) on the scheme's secret key, so long as the total amount of information leaked does not exceed an a-priori information theoretic bound. More formally, the attacker can request and receive $f(sk)$ for a length-restricted function $f$.[4] [AGV09, NS09] presented public-key encryption schemes that are resilient to leakage of even a $(1-o(1))$ fraction of the secret key (we call this the "leakage rate"). In particular, [AGV09] showed how this can be achieved under the LWE assumption, while [NS09] showed that this can be achieved under the DDH (or $d$-linear) assumption. It is further shown in [NS09] that some leakage resilience can be achieved using any universal hash proof system (defined in [CS02]), where the leakage rate depends on the parameters of the hash proof system. This implies secure schemes under the the QR and DCR assumptions as well. However, using the known hash proof systems, the leakage rate achievable under the QR assumption was only $o(1)$ — much less than the desired $(1-o(1))$. Based on the DCR assumption, a leakage rate of $(1-o(1))$ was achievable [NS09, CS02, DJ01].

**Auxiliary input.** Dodis, Kalai and Lovett [DKL09] and Dodis, Goldwasser, Kalai, Peikert and Vaikuntanathan [DGK$^+$10] considered the case where the leakage is not restricted information theoretically, but rather *computationally.* In the public key setting, the attacker is allowed to access any information on the secret key, under the following computational restriction: recovering the secret key $sk$ from said information $f(pk, sk)$, for $f$ of the attackers choosing, needs to be computationally hard to a sufficient extent (see discussion of several formalizations in [DGK$^+$10]). This notion of security was termed *security in the presence of auxiliary input* (or *auxiliary-input security*, for short). Public-key auxiliary-input secure encryption schemes under the DDH and LWE assumptions were recently presented in [DGK$^+$10].

To summarize, prior to this work it was known how to achieve KDM security, resilience to leakage of $(1-o(1))$ fraction of the secret key, and auxiliary-input security based on either the LWE or DDH assumption. While the techniques for achieving all of those seemed related, a formal connection was not known. Further, it seemed that previous methods do not extend to assumptions such as quadratic residuosity.

## 4.2 Overview of Our Contributions and Techniques

We give a high level description of our schemes in Section 4.2.1, and of our unified proof technique in Section 4.2.2. Other related work and follow-up work is then discussed in Section 4.2.3.

### 4.2.1 Our Contributions

Let us define a generalized class of assumptions called *subgroup indistinguishability* (SG) assumptions. A subgroup indistinguishability problem is defined by a group $\mathbb{G}_U$ ("the universe group") which is an internal direct product of two groups $\mathbb{G}_U = \mathbb{G}_M \times \mathbb{G}_L$ (interpreted as "the group of messages" and "the language group") whose orders, denoted by $M, L$ respectively, are relatively prime and where $\mathbb{G}_M$ is a cyclic group. Essentially, the *subgroup indistinguishability assumption* is that a random element of the universe $\mathbb{G}_U$ is computationally indistinguishable from a random

[4]To be more precise, the requirement is that the min-entropy of the secret $sk$ drops by at most a bounded amount, given $f(sk)$.

element in $\mathbb{G}_L$. In other words, the language $\mathbb{G}_L$ is hard on average in the universe $\mathbb{G}_U$. The precise definition is a little more involved, see Section 4.4 for details.

Two special cases of the subgroup indistinguishability assumptions are the quadratic residuosity (QR) assumption on Blum integers and Paillier's decisional composite residuosity (DCR) assumption. This is easily seen for QR as follows. Consider an integer $N = p \cdot q$, where $p, q$ are random primes of equal bit-length, let $\mathbb{Z}_N^* = \{x \in \mathbb{Z}_N : \gcd(x, N) = 1\}$, let $\mathbb{J}_N$ denote the group of Jacobi symbol $(+1)$ elements of $\mathbb{Z}_N^*$, and $\mathbb{QR}_N = \{x^2 : x \in \mathbb{Z}_N^*\}$ denote its subgroup of quadratic residues. The *quadratic residuosity* (QR) assumption is then, that the uniform distributions over $\mathbb{J}_N$ and $\mathbb{QR}_N$ are computationally indistinguishable. Taking $N$ to be a *Blum integer* where $p, q = 3 \pmod 4$ (otherwise the orders of $\mathbb{G}_L, \mathbb{G}_M$ we define next will not be relatively prime) and setting $\mathbb{G}_U = \mathbb{J}_N$, $\mathbb{G}_L = \mathbb{QR}_N$ (which is of odd order), and $\mathbb{G}_M = \{\pm 1\}$ (which is cyclic and has order 2), the QR assumption falls immediately into the criteria of subgroup indistinguishability assumptions.

We are now ready to describe our encryption scheme for a given subgroup problem $(\mathbb{G}_U, \mathbb{G}_M, \mathbb{G}_L)$, where $h$ is a generator for $\mathbb{G}_M$. In general, we view the *plaintext message space* as the elements $h^m \in \mathbb{G}_M$ (sometimes the exponent $m$ itself can be viewed as the message). For the case of QR, the plaintext message space is $\mathbb{G}_M = \{\pm 1\}$.

A word on the choice of parameters is in order. All parameters are measured as a function of the security parameter $k$. As customary, in the QR and DCR cases, think of the security parameter as the size of the modulus $N$ (i.e. $k = \lceil \log N \rceil$). We let $\ell$ denote a parameter whose value is polynomially related to $k$,[5] selected in accordance to the desired properties of the scheme (KDM security, amount of leakage resilience etc.).

**The Encryption Scheme for Subgroup Problem $(\mathbb{G}_U, \mathbb{G}_M, \mathbb{G}_L)$ with Parameter $\ell$:**

- *Key generation.* Set the secret key to a random binary vector $\mathbf{s} = (s_1, \ldots, s_\ell)$ of length $\ell$. Set the public key to be the tuple $(g_1, \ldots, g_\ell, g_0)$ where $g_1, \ldots, g_\ell$ are uniformly chosen elements of $\mathbb{G}_L$ and $g_0 = \prod g_i^{-s_i}$. (For the QR assumption, the public key thus consists of $\ell$ random squares, followed by a product of a random subset of them, selected by the secret key $\mathbf{s}$).

- *Encryption.* On input message $h^m$,[6] sample a uniform integer $r$ from a large enough domain and output the ciphertext $(g_1^r, \ldots, g_\ell^r, h^m \cdot g_0^r)$. (For the QR assumption case, encryption is of single bits $\{\pm 1\}$, and the ciphertext is the tuple of squares in the public key, raised to a random power, where the last one is multiplied by the plaintext message.)

- *Decryption.* On ciphertext $(c_1, \ldots, c_\ell, c_0)$, compute $h^m = c_0 \cdot \prod c_i^{s_i}$. (For the case of QR, $m = c_0 \cdot \prod c_i^{s_i}$.) In general, recoverability of the exponent $m$ depends on whether taking discrete logs in base $h$ of $h^m$ is easy.

We remark that in this high level overview we only consider binary secret keys, as we will for the most part of this chapter. However, our scheme is also meaningful when the secret key vector is sampled from a distribution over different domains. The properties of the scheme in such a case become more dependent on the specific group.

[5]More precisely, $\ell$ is a polynomial function $\ell(k)$.

[6]Recall that $h$ is a generator of $\mathbb{G}_M$, which is a part of the description of $\mathbb{G}_U$.

The basic structure of our construction is strikingly similar to [BHHO08], where the public key also contains $\ell$ independent "random" elements and an additional element that is statistically close to uniform, but in fact is a combination of the previous ones. The difference and challenge is in how to prove security. This challenge is due to the fact that subgroup indistinguishability assumptions seem inherently different from the DDH assumption. In the latter, for cyclic group $\mathbb{G}$ where DDH is assumed, the assumption implies that the distribution $(g_1, g_2, g_1^r, g_2^r)$ is computationally indistinguishable from $(g_1, g_2, g_1', g_2')$ , implying complete re-randomization (a similar property follows for LWE). Such re-randomization does not follow nor is it necessarily true from subgroup indistinguishability. Rather, we will have to use the weaker guarantee that $(g_1, g_2, g_1^r, g_2^r)$ is indistinguishable from $(g_1, g_2, h^{r'} \cdot g_1^r, h^{r''} \cdot g_2^r)$, which only "masks" of the $\mathbb{G}_M$ part of the elements.

Similarly to the scheme of [BHHO08], our scheme is lacking in efficiency. This is most noticeable when instantiating with the QR assumption, where a single bit message is encrypted by $\ell+1$ group elements, each of size roughly the security parameter $k$. The situation is somewhat better when relying on DCR, where each such ciphertext encrypts $\Omega(k)$ bits. Improved efficiency can be achieved by using the same values $g_1, \ldots, g_\ell$ with many vectors $\mathbf{s}$, however this makes KDM security hold only with respect to a less natural function class (this is similar to the more efficient LWE based scheme of [ACPS09]) and significantly reduces leakage resilience. Coming up with more efficient KDM secure or leakage resilient schemes remains an interesting open problem and some steps in that direction has been taken, see Section 4.2.3.

On top of standard semantic security, we prove the following properties for the new scheme.

**Property 1:** KDM-**Security**

First, we prove that the scheme is KDM$^{(1)}$-secure w.r.t. affine functions of the secret key. To show this for QR case, we show that for any affine function specified by $a_0, \ldots, a_\ell$, the encryption of $(-1)^{a_0+\sum_i a_i s_i}$ is indistinguishable from the encryption of $(-1)^0$. For the general case, it is more natural to view KDM$^{(1)}$ with respect to the affine functions "in the exponent": for any $h_0, h_1, \ldots, h_\ell \in \mathbb{G}_M$ where $h_i = h^{a_i}$ (and $h$ is a generator), we show that an encryption of $h_0 \cdot \prod h_i^{s_i} = h^{a_0+\sum_i a_i s_i}$ is indistinguishable from an encryption of $h^0$.

Second, we prove that for any polynomial value of $n$, the above encryption scheme satisfies KDM$^{(n)}$ security, if $\ell$ is larger than, roughly, $n \log L$. We note thus that the public key size and ciphertext size grow with $n$ to achieve provable KDM$^{(n)}$ security. Interestingly, in the works of [BHHO08, ACPS09], $\ell$ did not need to grow with $n$. This seems difficult to achieve without the complete "re-randomization" property discussed above which follows from the DDH and LWE assumptions, but not from ours.

Finally, we can also show that our scheme can be used to obtain KDM security for larger classes of functions than affine function: The scheme is *entropy-$\kappa$* KDM-*secure* (for proper values of $\ell$), as required in [BGK11] and therefore implies a scheme that is secure w.r.t. functions of the form $a_0 + \sum_i a_i f_i(sk)$ for (roughly) any set of polynomially-many efficiently computable functions $\{f_1, \ldots, f_\ell\}$. Our scheme also satisfies the conditions for the completeness theorem of [App11], and therefore implies that for any polynomial bound $p$, there is a scheme that is secure w.r.t. all functions computable by size-$p$ circuits. A more complicated proof can be derived using the methods of [BHHI10].

**Property 2: Improved Key-Leakage Resilience**

We prove that the new scheme is resilient to any leakage of a $(1 - o(1))$ fraction of the bits of the secret key. Stated differently, if one specifies in advance the amount of leakage $\lambda$ (a polynomial in the security parameter) to be tolerated, we can choose $\ell$ to obtain a scheme that is secure against a leakage of $\lambda$ bits. The growth of $\ell$ is additive in $\lambda$ (i.e. $\ell = \ell_0 + \lambda$) which implies leakage rate of $(1 - (\ell_0/\ell))$. An appropriate selection of $\ell$ thus brings the rate to $(1 - o(1))$.

We emphasize that while schemes with the above guarantees were known under LWE [AGV09] or DDH [NS09], and even (implicitly) under DCR [NS09, CS02], this was not the case under QR. Previous results with regards to QR-based leakage resilience [NS09, CS02] could only approach a leakage rate of $1/k = o(1)$ (recall that $k$ is the security parameter, or the bit-length of the modulus $N$), compared to $(1 - o(1))$ in our scheme.

In addition, leakage resilience based on QR or DCR was previously only achievable based on hash proof systems. These, in turn, required that the modulus in the assumption $N = p \cdot q$ is such that $p, q$ are *safe primes*. We do not impose this restriction. Under QR, we only require that $p, q = 3 \pmod 4$ (i.e. $N$ is a *Blum integer*), and under DCR, we only require that $p, q$ have the same bit-length.

**Property 3: Auxiliary Input Security**

We prove that our schemes remain secure when the attacker has access to additional information on the secret key $sk$, in the form of $f_{pk}(sk)$, where $f_{pk}$ is a polynomial time function (which may depend on the public key) that is evaluated on the secret key $sk$. First, we consider the case where $f$ is such that the transition $(f_{pk}(sk), pk) \to sk$ is computationally hard. Namely, that retrieving the secret key $sk$ given the public key $pk$ and the auxiliary information $f_{pk}(sk)$, is sufficiently hard. This notion was termed *weak auxiliary-input* security in [DGK$^+$10]. In turn, [DGK$^+$10] show how to leverage weak auxiliary-input security to achieve security when the requirement on $f$ is weaker: now, only the transition $f_{pk}(sk) \to sk$ needs to be hard. The latter is called *auxiliary-input* security.

We conclude that for all $\delta > 0$, we can select the value of $\ell$ such that the scheme is auxiliary-input secure relative to any function that is hard to invert (in polynomial time) with probability $2^{-\ell^\delta}$. We note that the input to the function is the secret key – a length $\ell$ binary string, and therefore we measure hardness as a function of $\ell$ (and not of the security parameter $k$).

### 4.2.2 A Unified Proof Technique

The circular security, leakage resilience and auxiliary-input security properties of our scheme are proved using a new technical tool introduced in this work: the *adaptive vector game*. This proof technique can also provide an alternative proof for the KDM$^{(1)}$-security, leakage resilience and auxiliary-input security of (known) public-key encryption schemes based on DDH and LWE, thus providing an alternative, more generic proof for some of the results of [BHHO08, ACPS09, NS09, DGK$^+$10].[7]

[7]In this work, the adaptive vector game is defined only for our subgroup indistinguishability assumptions, but it easily extends to other assumptions.

This suggests an alternative explanation to the folklore belief that the three notions are related: that it is the proof technique that is related in fact. Namely, the proof techniques for each property can be generalized to adaptive vector games which, in turn, imply the other properties.

We proceed to overview the proofs of security for the various properties of our scheme. Again, let us consider the groups $\mathbb{G}_U = \mathbb{G}_M \times \mathbb{G}_L$ with $h$ being a generator for $\mathbb{G}_M$, such that the subgroup indistinguishability assumption holds.

To best explain the ideas of the proof, let us consider, as a first step, a simple semantically secure encryption scheme (which is a generalization of the Goldwasser-Micali scheme [GM82]). An encryption of 0 is a random element $g \in \mathbb{G}_L$ and an encryption of 1 is $h \cdot g$ (in the QR case, the encryption of $(+1)$ is a random quadratic residue and the encryption of $(-1)$ is a random quadratic non-residue). The two distributions are clearly indistinguishable (consider the indistinguishable experiment where $g$ is uniform in $\mathbb{G}_U$). In order to decrypt, one needs some "trapdoor information" that would enable to distinguish between elements in $\mathbb{G}_L$ and $\mathbb{G}_U$ (such as the factorization of the modulus $N$ in the QR (and DCR) case).

The first modification of this simple idea was to fix $g$ and put it in the public key, and set the ciphertext for $h^m$ to $h^m \cdot g^r$ for $r$ large enough. For the QR case, this modification still amounts to encrypting $(+1)$ by a random square, and $(-1)$ by a random non-square.

The second modification does away with the need of the secret key owner to distinguish between elements in $\mathbb{G}_L$ and $\mathbb{G}_U$ (e.g. with the need to know the factorization of $N$ in the QR case), by replacing the "trapdoor information" with a secret key that is a uniform binary vector $\mathbf{s} = (s_1, \ldots, s_\ell)$. Holding the secret key will not enable us to solve subgroup indistinguishability, but will enable us to decrypt as in [BHHO08]. We take a set of random elements $g_1, \ldots, g_\ell \in \mathbb{G}_L$ and define $g_0 = \prod g_i^{-s_i}$. If $\ell$ is large enough, then the leftover hash lemma guarantees that $g_0$ is almost uniform. As the ciphertext is $(g_1^r, \ldots, g_\ell^r, h^m \cdot g_0^r)$, one can recover $h^m$ using $\mathbf{s}$. Recovering $m$ itself is also possible if the discrete logarithm problem in $\mathbb{G}_M$ is easy, as is the case in the QR scenario.

The crux of the idea in proving security is the following. First, we note that the distribution of $g_0$ is close to uniform in $\mathbb{G}_L$, even given $g_1, \ldots, g_\ell$ (by the leftover hash lemma). Recall that in a DDH-based proof, we could claim that $((g_1, \ldots, g_\ell, g_0), (g_1^r, \ldots, g_\ell^r, g_0^r))$ is computationally indistinguishable from $((g_1, \ldots, g_\ell, g_0), (g'_1, \ldots, g'_\ell, g'_0))$ (where $g'_i$ are uniform). However, based on subgroup indistinguishability, a different method is required: Consider replacing $g_0$ with $g'_0 = h \cdot g_0$, the distribution $((g_1, \ldots, g_\ell, g_0), (g_1^r, \ldots, g_\ell^r, g_0^r))$ is computationally indistinguishable from $((g_1, \ldots, g_\ell, h \cdot g_0), (g_1^r, \ldots, g_\ell^r, h^r \cdot g_0^r))$ under the subgroup indistinguishability assumption. The crucial observation now is that since the orders of $\mathbb{G}_M$ and $\mathbb{G}_L$ are relatively prime, then in fact $g_0'^r = h^{r'} \cdot g_0^r$, where $r'$ is independent of $r$. Combined with the fact that $\mathbb{G}_M$ is cyclic, we get that $((g_1, \ldots, g_\ell, g_0), (g_1^r, \ldots, g_\ell^r, g_0^r))$ is indistinguishable from $((g_1, \ldots, g_\ell, h \cdot g_0), (g_1^r \cdots g_\ell^r, h' \cdot g_0^r))$, for a random $h' \in \mathbb{G}_M$. Semantic security now follows.

To address the issues of circular security, leakage resilience and auxiliary-input, we generalize the idea presented above, and prove that the distribution $((g_1, \ldots, g_\ell), (h^{a_1} \cdot g_1^r, \ldots, h^{a_\ell} \cdot g_\ell^r))$ is indistinguishable from $((g_1, \ldots, g_\ell), (g_1^r, \ldots, g_\ell^r))$. We provide an adaptive variant of this claim, which we call an *adaptive $\ell$-vector game*, where the values of $a_1, \ldots, a_\ell \in \mathbb{Z}$ are selected by the distinguisher, and can depend on $(g_1, \ldots, g_\ell)$, and show that the above is hard even in such case. The adaptive vector game will be employed in the proofs of all properties of the scheme.

For key-dependent message security, we consider the ciphertext $(g_0^r, g_1^r, \ldots, h \cdot g_i^r, \ldots, g_\ell^r)$. This ciphertext will be decrypted to $h^{s_i}$ and in fact can be shown (using an adaptive vector game) to be computationally indistinguishable from a legal encryption of $h^{s_i}$. Key-dependent message security follows from this fact.

Proving KDM$^{(n)}$-security for our scheme is more complex. To illustrate this, we contrast it with the ideas in the proof of [BHHO08]. They used homomorphism and re-randomization to achieve KDM$^{(n)}$-security: Their scheme is shown to have *homomorphic* properties that enable to "shift" public keys and ciphertexts that are relative to a certain secret key, into ones that are relative to another secret key. In order to apply these "shifts", one only needs to know the relation between the original and final keys (and not the keys themselves). In addition, their scheme is shown to have *re-randomization* properties that enable to take a public key (or ciphertext) and produce an independent public key (or ciphertext) that corresponds to the same secret key (and message, in the ciphertext case). These two properties enable simulating the KDM$^{(n)}$-security game using only one "real" secret key, fabricating the $n$ required keys and ciphertexts using homomorphism and re-randomization. In [ACPS09], similar ideas are employed, but the re-randomization can be viewed as implicit in the assumption (the ability to generate independently looking vectors that are in fact linearly related).

Our scheme can be shown to have such homomorphic properties, but it doesn't enjoy as strong re-randomizability as required to use the above techniques. As an example, consider a public key $pk = (g_0, g_1, \dots, g_\ell)$ corresponding to a secret key $sk = (s_1, \dots, s_\ell)$, i.e. $g_0 = \prod g_i^{-s_i}$. Let $j \in [\ell]$ and consider $\hat{pk} = (\hat{g}_0, \hat{g}_1, \dots, \hat{g}_\ell)$ defined as follows: for all $i \notin \{j, 0\}$, set $\hat{g}_i = g_i$; for $j$, set $\hat{g}_j = g_j^{-1}$; and finally set $\hat{g}_0 = g_j \cdot g_0 = \hat{g}_j^{-(1-s_j)} \cdot \prod_{i \neq j} \hat{g}_i^{-s_i}$. We get that $\hat{pk}$ is a properly distributed public key corresponding to the secret key $\hat{sk} = sk \oplus e_j$ ($sk$ XORed with the $j^{\text{th}}$ unit binary string). Namely, we were able to "shift" a public key to correspond to another (related) secret key, without knowing the original key. However, the joint distribution of $pk, \hat{pk}$ is easily distinguishable from that of two independent public keys. What we lack is the ability to re-randomize $\hat{pk}$ so that it is distributed as a public key for $\hat{sk}$ which is independent of the original $pk$.

Intuitively, this shortcoming requires us to use more "real randomness". Our proof simulates the KDM$^{(n)}$-security game using only one "real" secret key, as in the idea presented above. This secret key is used to fabricate $n$ secret and public keys. However, when we want to apply the leftover hash lemma to claim that the $g_0$ components of all $n$ fabricated public keys are close to uniform, we need the one real secret key to have sufficient entropy. This requires a secret key whose size is linear in $n$. These ideas, combined with the ones used to prove KDM$^{(1)}$ security, give our final proof.

To go beyond affine functions, we use techniques from either [BGK11] or [App11]. The property of entropy-$\kappa$ KDM-security [BGK11] requires that the scheme remains secure even when the secret key is sampled from a high-entropy (but not necessarily uniform) distribution. This is shown to hold using the leftover hash lemma, since $\prod g_i^{s_i}$ is a 2-universal hash function. For the completeness theorem of [App11], KDM security w.r.t. affine functions of the bits of the secret key is a sufficient condition.[8]

Leakage resilience and auxiliary-input security are proven by an almost identical argument:

[8]The more complicated methods of [BHHI10] can also be used: a targeted encryption scheme can be constructed based on the ability to "fabricate" ciphertexts that correspond to affine functions of the secret key without knowing the secret key itself.

consider a case where we replace the ciphertext $(h^m \cdot g_0^r, g_1^r, \ldots, g_\ell^r)$ with a computationally indistinguishable one: $(h^{-\sum \sigma_i s_i} \cdot h^m \cdot g_0^r, h^{\sigma_1} \cdot g_1^r, \ldots, h^{\sigma_\ell} \cdot g_\ell^r)$, where $\sigma_i \in \mathbb{Z}_M$ are uniform. Computational indistinguishability (even for a known secret key) follows from the adaptive vector game mentioned above. For leakage-resilience, the leftover hash lemma implies that so long as there is sufficient entropy in $\mathbf{s}$ after the leakage, $\sum \sigma_i s_i$ will be close to uniform and will "mask" the value of $m$. For auxiliary input we use the *generalized Goldreich-Levin* theorem of [DGK$^+$10] to show that $\sum \sigma_i s_i$ is close to uniform in the presence of a function of $\mathbf{s}$ that is hard to invert, even given the public key. Thus obtaining *weak* auxiliary-input security. In the QR case, the inner product is over $\mathbb{Z}_2$ and therefore we can use the "standard" Goldreich-Levin theorem [GL89], which implies better parameters. We use leveraging (as used in [DGK$^+$10]) to obtain the full result.

From the discussion above it is apparent that the distribution of $\mathbf{s}$ only comes into play when the leftover hash lemma is to be applied (or the generalized Goldreich-Levin theorem for the case of auxiliary input). Thus we can use other (non-binary) secret key distributions, so long as the extraction properties are preserved. We formalize this argument in Section 4.8.7.

### 4.2.3 Other Related Work

Cramer and Shoup [CS02] presented the notion of *hash proof systems*, which are similar to subgroup indistinguishability assumptions. Their implementations from QR and DCR does not require the factorization of $N$ in order to decrypt, as in our scheme. However they use the discrete logarithm of (their analog to) the $g_i$'s as a secret key for the system. Our scheme can be seen as taking another step towards "stripping" the secret key of all structure: in our scheme, it is just a uniform sequence of bits (resulting in a weaker form of a hash proof system that is "universal on average").

Boneh, Goh and Nissim [BGN05] construct a scheme with useful homomorphic properties based on the "bilinear decision subgroup assumption". This assumption is similar in syntax to our subgroup indistinguishability assumptions, however their assumption does not fall under our category. The reason is, essentially, the bilinear map: in our work, we require that a generator for each subgroup is publicly known, whereas in [BGN05], since an efficient bilinear map exists, revealing generators for both subgroups breaks the indistinguishability.

Hemenway and Ostrovsky [HO09] show how to construct lossy trapdoor functions (see [PW08] for definition) from the QR and DCR assumptions (among other assumptions). Similar ideas can be used in a straightforward manner to construct lossy trapdoor functions from subgroup indistinguishability assumptions with special properties.

In a follow-up work, Malkin, Teranishi and Yung [MTY11] showed how to use our techniques, among others, to achieve KDM security for a different class of functions, based on the DCR assumption. The main building block in their construction is an instantiation of our scheme with $\ell = 1$, and with a non-binary secret key distribution. The secret key is represented as an integer over some ring, and KDM security will hold w.r.t. the class of arithmetic circuits over this ring. The possibility of working with non-binary secret key distributions was implicit in our original work [BG10] and we now make it explicit in Section 4.8.7.

In a follow-up work with Segev [BS11], we used tools from this work and from [HO09] to present a deterministic encryption scheme that is secure for message distributions that don't have any entropy, but are computationally unpredictable (a short abstract of this work appears in the appendix).

## 4.3 Preliminaries

We denote scalars in plain lowercase ($x \in \{0,1\}$) and vectors in bold lowercase ($\mathbf{x} \in \{0,1\}^n$). The $i^{\text{th}}$ coordinate of $\mathbf{x}$ is denoted $x_i$. The vector inner product of $\mathbf{x}, \mathbf{y}$ is denoted by $\langle \mathbf{x}, \mathbf{y} \rangle$ and is defined to be $\sum x_i \cdot y_i$. The vector $\mathbf{e_i}$ is the $i^{\text{th}}$ unit vector.

For any $K \in \mathbb{N}$, we denote $[K] = \{1, \ldots, K\}$. We let $\mathbb{Z}_K$ denote the ring $\mathbb{Z}/K\mathbb{Z}$ and let $\mathbb{Z}_K^*$ denote the group of units in $\mathbb{Z}_K$. Euler's totient function is denoted by $\varphi(\cdot)$.

Arithmetic operations are always performed in $\mathbb{Z}$ but can sometimes also be interpreted as being performed over $\mathbb{Z}_K$, for some value $K \in \mathbb{N}$. Specifically, if $h$ is an element of order $K$ in a multiplicative group, then $h^x = h^{(x \bmod K)}$, so operations "in the exponent of $h$" can be interpreted as operations over $\mathbb{Z}_K$. We usually write $(\bmod\ K)$ to indicate that an operation is performed over $\mathbb{Z}_K$, but we sometimes omit this when $K$ is clear from the context.

For vectors $\mathbf{g}, \mathbf{h} \in \mathbb{G}^n$, where $\mathbb{G}$ is a multiplicative commutative group, we denote by $\mathbf{g}^r$ the vector whose $i^{\text{th}}$ coordinate is $g_i^r$. We denote by $\mathbf{h} \cdot \mathbf{g}$ the vector whose $i^{\text{th}}$ coordinate is $h_i \cdot g_i$. Note that this does *not* denote an inner product. For a group element $g \in \mathbb{G}$ and a vector $\mathbf{x} \in \mathbb{Z}$, we let $g^{\mathbf{x}}$ denote the vector whose $i^{\text{th}}$ coordinate is $g^{x_i}$.

Let $X$ be a probability distribution over a domain $S$, we write $x \xleftarrow{\$} X$ to indicate that $x$ is sampled from the distribution $X$. The uniform distribution over a set $S$ is denoted $U(S)$. We use $x \xleftarrow{\$} S$ as abbreviation for $x \xleftarrow{\$} U(S)$. For any function $f$ with domain $S$ we let $f(X)$ denote the random variable (or corresponding distribution) obtained by sampling $x \xleftarrow{\$} X$ and outputting $f(x)$. The *min-entropy* of a (discrete) random variable $X$ is $\mathbf{H}_\infty(X) = \min_{x \in S}\{-\log \Pr[X = x]\}$.

We write $\operatorname{negl}(k)$ to denote an arbitrary *negligible* function, i.e. one that vanishes faster than the inverse of any polynomial.

The *statistical distance* between two distributions $X, Y$ (or random variables with those distributions) over a common domain $S$ is $\max_{A \subseteq S} |\Pr[X \in A] - \Pr[Y \in A]|$. Two ensembles $X = \{X_k\}_k$, $Y = \{Y_k\}_k$ are $\epsilon = \epsilon(k)$-*close* if for all $k$, the distance between $X_k$ and $Y_k$ is at most $\epsilon(k)$ and are *statistically indistinguishable* if $\epsilon(k) = \operatorname{negl}(k)$. An ensemble $X = \{X_k\}_k$ over domain $S = \{S_k\}_k$ is $\epsilon = \epsilon(k)$-*uniform* in $S$ if it is $\epsilon$-close to the uniform ensemble over $S$ (we sometimes omit $S$ when it is clear from the context). $X = \{X_k\}_k$, $Y = \{Y_k\}_k$ are *computationally indistinguishable* if every $\operatorname{poly}(k)$-time adversary $\mathcal{A}$ has negligible *distinguishing advantage*:

$$\operatorname{Dist}_{X,Y}\operatorname{Adv}[\mathcal{A}] = |\Pr[\mathcal{A}(X_k) = 1] - \Pr[\mathcal{A}(Y_k) = 1]| = \operatorname{negl}(k) \ .$$

We often abbreviate and write $\operatorname{DistAdv}[\mathcal{A}]$ when $X, Y$ are clear from the context.

### 4.3.1 Public-Key Encryption and Semantic/CPA Security

A public-key encryption scheme $\mathcal{E} = (G, E, D)$ is defined by its key generation, encryption and decryption algorithms. The key generation algorithm $G$ takes as input the unary vector $1^k$, where $k$ is called the *security parameter* of the scheme. All other parameters of the scheme are a function of $k$. We let $\mathcal{S} = \{\mathcal{S}_k\}$ denote the space of secret keys and $\mathcal{M} = \{\mathcal{M}_k\}$ denote the message space of the encryption scheme. The scheme is correct if for any sequence $\{m_k \in \mathcal{M}_k\}$ it holds that

$$\Pr[D_{sk}(E_{pk}(m_k)) \neq m_k] = \operatorname{negl}(k) \ ,$$

where the probability is over the random coins of $(sk, pk) \leftarrow G(1^k)$ and the random coins of $E, D$.

The standard notion of security (against passive adversaries) for public-key encryption is "semantic security". We bring here an equivalent formulation of security under chosen plaintext attack (CPA), which is close to our formulation of KDM security below. Intuitively, CPA security requires that no attacker can distinguish between an encryption of a message of its choice and an encryption of a fixed message "0". We refer the reader to [Gol04] for a formal definition of encryption schemes and their security.

We define the CPA game played between a challenger and an adversary as follows.

- **Initialize.** The challenger selects $b \xleftarrow{\$} \{0,1\}$ and generates a key pair $(sk, pk) \xleftarrow{\$} G(1^k)$. The challenger then sends $pk$ to the adversary.
- **Query.** The adversary sends a message $m$ to the challenger. The challenger sends the following ciphertext to the adversary.

$$c \leftarrow \begin{cases} E_{pk}(m) & \text{if } b = 0 \\ E_{pk}(0) & \text{if } b = 1. \end{cases}$$

- **Finish.** The adversary outputs a guess $b' \in \{0,1\}$.

The scheme $\mathcal{E}$ is CPA secure if any polynomial time adversary $\mathcal{A}$ has negligible advantage:

$$\mathrm{CPAAdv}[\mathcal{A}] = \left|\Pr[b' = 1|b = 0] - \Pr[b' = 1|b = 1]\right| = \mathrm{negl}(k) \ .$$

### 4.3.2 KDM Security

In the scenario of key-dependent messages, we wish to model the case where functions of the secret key can be encrypted, and require that the resulting ciphertexts are indistinguishable from encryptions of 0. We want our definition to apply also for the case of "key cycles" where a function of one user's secret key is encrypted by another's public key and vice versa. The most inclusive definition, therefore, is parameterized by the number of users $n$ and allows encrypting a function of the entire vector of $n$ secret keys under any of the corresponding public keys (this is sometimes referred to as "clique security"). An additional parameter to be considered is the set of functions of the secret key that we allow to encrypt. We use the definition presented in [BHHO08].

Formally, let $\mathcal{E} = (G, E, D)$ be a public key encryption scheme, $n > 0$ be an integer, $\mathcal{S} = \{\mathcal{S}_k\}$ be the space of secret keys, and let $\mathcal{F} = \{\mathcal{F}_k\}$ be a class of functions such that $\mathcal{F}_k \subseteq \mathcal{S}_k^n \to \mathcal{M}_k$.

We define the KDM$^{(n)}$ game, w.r.t. the function class $\mathcal{F}$, played between a challenger and an adversary as follows.

- **Initialize.** The challenger selects $b \xleftarrow{\$} \{0,1\}$ and generates, for all $i \in [n]$, key pairs $(sk_i, pk_i) \xleftarrow{\$} G(1^k)$. The challenger then sends $\{pk_i\}_{i \in [n]}$ to the adversary.
- **Query.** The adversary makes queries of the form $(i, f) \in [n] \times \mathcal{F}_k$. For each query, the challenger computes $y \leftarrow f(sk_1, \ldots, sk_n)$ and sends the following ciphertext to the adversary.

$$c \leftarrow \begin{cases} E_{pk_i}(y) & \text{if } b = 0 \\ E_{pk_i}(0) & \text{if } b = 1. \end{cases}$$

- **Finish.** The adversary outputs a guess $b' \in \{0,1\}$.

The scheme $\mathcal{E}$ is KDM$^{(n)}$ secure if any polynomial time adversary $\mathcal{A}$ has negligible advantage:

$$\mathrm{KDM}^{(n)}\mathrm{Adv}[\mathcal{A}] = \left|\Pr[b' = 1|b = 0] - \Pr[b' = 1|b = 1]\right| = \mathrm{negl}(k) \ .$$

We sometime denote KDM$^{(n)}_{\mathcal{F}}$ to indicate the function class in discussion.

We note that even the notion KDM$^{(1)}$-security generalizes CPA-security. To see this, take $\mathcal{F}$ to be the class of constant functions.

### 4.3.3 Leakage Resilient Encryption

In the scenario of key-leakage resilience, we wish to model the case where some (adversarially selected restricted amount of) information about the secret key is revealed to the attacker. We require that even in the presence of this additional information, the security of the scheme remains intact. The definition below is essentially adopted from [NS09].

Let $\mathcal{E} = (G, E, D)$ be a public key encryption scheme with key-space $\mathcal{S} = \{\mathcal{S}_k\}$ and message space $\mathcal{M} = \{\mathcal{M}_k\}$. We define the $\lambda$-leakage game, for the non-negative parameter $\lambda = \lambda(k)$, played between a challenger and an adversary as follows.

- **Initialize.** The challenger selects $b \overset{\$}{\leftarrow} \{0,1\}$ and generates a key-pair $(sk, pk) \overset{\$}{\leftarrow} G(1^k)$. The challenger sends $pk$ to the adversary.
- **Leakage.** The adversary sends an efficiently computable function $f : \mathcal{S}_k \to \{0,1\}^\lambda$ to the challenger. The challenger computes $f(sk)$ and returns this value to the adversary.
- **Challenge.** The adversary sends $m_0, m_1 \in \mathcal{M}_k$ to the challenger. The challenger computes $y \leftarrow E_{pk}(m_b)$ and sends $y$ to the adversary.
- **Finish.** The adversary outputs a guess $b' \in \{0,1\}$.

The scheme $\mathcal{E}$ is $\lambda$-leakage secure if for any polynomial time adversary $\mathcal{A}$ it holds that

$$\mathrm{Leak}_\lambda\mathrm{Adv}[\mathcal{A}] = \left|\Pr[b' = 1|b = 0] - \Pr[b' = 1|b = 1]\right| = \mathrm{negl}(k) \ .$$

We note that the definition can be extended to the case of chosen ciphertext attacks (see [NS09]). In this work, however, we only consider the case of chosen plaintext attacks described above.

### 4.3.4 Auxiliary-Input Secure Encryption

The scenario of auxiliary-input security is quite similar to that of key-leakage resilience described in Section 4.3.3. As in the previous case, we model a scenario where the attacker can access additional information about the secret key. In this case, however, the restriction on the amount of information is *computational* rather than information theoretic. The definition is adopted (and slightly adapted) from [DGK$^+$10].

Let $\mathcal{E} = (G, E, D)$ be a public key encryption scheme with secret key space $\mathcal{S} = \{\mathcal{S}_k\}$, public key space $\mathcal{P} = \{\mathcal{P}_k\}$ and message space $\mathcal{M} = \{\mathcal{M}_k\}$. For any family of functions $f = \{f_k : \mathcal{S}_k \times \mathcal{P}_k \to$

$\{0,1\}^*\}$, we define the *inverting advantage* and *weak inverting advantage* of an adversary $\mathcal{A}$ as follows.

$$\begin{aligned}\mathrm{Inv}_f\mathrm{Adv}[\mathcal{A}] &= \Pr_{(sk,pk)\leftarrow G(1^k)}[\mathcal{A}(1^k, f_k(sk,pk)) = sk] \ , \\ \mathrm{Inv}_f^{\mathrm{weak}}\mathrm{Adv}[\mathcal{A}] &= \Pr_{(sk,pk)\leftarrow G(1^k)}[\mathcal{A}(1^k, pk, f_k(sk,pk)) = sk] \ .\end{aligned}$$

If the scheme $\mathcal{E}$ has public parameters, then those are given as additional input to $\mathcal{A}$, in both flavors above. Let $\ell$ denote the length of the binary representation of the secret key. A polynomial time computable function $f$ is $\epsilon = \epsilon(\ell)$-hard to invert (resp. $\epsilon$-weakly hard to invert) if for any polynomial time $\mathcal{A}$ it holds that $\mathrm{Inv}_f\mathrm{Adv}[\mathcal{A}] \leq \epsilon$ (resp. $\mathrm{Inv}_f^{\mathrm{weak}}\mathrm{Adv}[\mathcal{A}] \leq \epsilon$). Note that we measure the hardness of $f$ relative to its input length $\ell$ and not the security parameter $k$. We stress that the notion of "hard to invert functions" is weaker than the standard notion of one-way functions, since we require the recovery of the original secret key (and not of just any pre-image). Thus a function that is not one-way can still be hard to invert.

For any efficiently computable function family $f$, we consider the $f$-auxiliary input game, played between a challenger and an adversary as follows.

- **Initialize.** The challenger selects $b \xleftarrow{\$} \{0,1\}$ and generates a key-pair $(sk, pk) \xleftarrow{\$} G(1^k)$. The challenger sends $pk$ to the adversary.
- **Auxiliary input.** The challenger computes $z \leftarrow f_k(sk, pk)$ and sends $z$ to the adversary.
- **Challenge.** The adversary sends $m_0, m_1 \in \mathcal{M}_k$ to the challenger. The challenger computes $y \leftarrow E_{pk}(m_b)$ and sends $y$ to the adversary.
- **Finish.** The adversary outputs a guess $b' \in \{0,1\}$.

The scheme $\mathcal{E}$ is *$\epsilon$-auxiliary input secure* if for any $\epsilon$-hard to invert $f$ and for any polynomial time $\mathcal{A}$ it holds that

$$\mathrm{Aux}_f\mathrm{Adv}[\mathcal{A}] = \left|\Pr[b' = 1|b = 0] - \Pr[b' = 1|b = 1]\right| = \mathrm{negl}(k) \ .$$

$\mathcal{E}$ is *$\epsilon$-weakly auxiliary-input secure* if the above holds for any $\epsilon$-weakly hard to invert function $f$.

### 4.3.5 Technical Tools

We use the following simple lemma.

**Lemma 4.3.1.** *Let $T, N \in \mathbb{N}$ and let $x \xleftarrow{\$} [T]$, then $x \pmod N$ is $(N/T)$-uniform in $\mathbb{Z}_N$.*

*Proof.* Define $d = (T \bmod N)$, then conditioned on the event $x \in [T-d]$, it holds that $(x \bmod N)$ is uniform in $\mathbb{Z}_N$. Therefore $(x \bmod N)$ is $(d/T) \leq (N/T)$-uniform. □

**Simplified Leftover Hash Lemma and Applications**

We use the following lemma which is an immediate corollary of the leftover hash lemma and explicitly appears in [BHHO08, Lemma 2].

**Lemma 4.3.2.** *Let $H$ be a 2-universal hash family from a set $X$ to a set $Y$. Then the distribution $(h, h(x))$ where $h \xleftarrow{\$} H$, $x \xleftarrow{\$} X$, is $\sqrt{\frac{|Y|}{4|X|}}$-uniform in $H \times Y$.*

We also require the following consequence.

**Lemma 4.3.3.** *Let $H$ be a 2-universal hash family from a set $X$ to a set $Y$. Let $f : X \to Z$ be some function. Then the distribution $(h, h(x), f(x))$ where $h \xleftarrow{\$} H$, $x \xleftarrow{\$} X$, is $\sqrt{\frac{|Y|\cdot|Z|}{4|X|}}$-close to $(h, y, f(x))$, for $y \xleftarrow{\$} Y$.*

*Proof.* For all $z \in Z$, denote $X_z = \{x \in X : f(x) = z\}$ and $p_z = |X_z| / |X|$. Then by Lemma 4.3.2, $(h, h(x), f(x))$ conditioned on $f(x) = z$ is $\sqrt{\frac{|Y|}{4|X_z|}}$-close to $(h, y, f(x))$ conditioned on $f(x) = z$. Averaging over all $z \in Z$, we get that the distance between $(h, h(x), f(x))$ and $(h, y, f(x))$ is at most

$$\sum_{z\in Z}\left(p_z \cdot \sqrt{\frac{|Y|}{4\,|X_z|}}\right) = \sqrt{\frac{|Y|}{4\,|X|}} \cdot \sum_{z\in Z}\sqrt{p_z} \le \sqrt{\frac{|Y|\,|Z|}{4\,|X|}} \ .$$

The result follows. ☐

We often use families of 2-universal hash functions of the form presented below.

**Lemma 4.3.4.** *Let $\mathbb{G}$ be any finite commutative group and let $\ell \in \mathbb{N}$. Then the set of functions $H = \{h_{g_1,\dots,g_\ell} : \{0,1\}^\ell \to \mathbb{G}\}_{g_1,\dots,g_\ell\in\mathbb{G}}$ where $h_{g_1,\dots,g_\ell}(\mathbf{x}) = \prod_{i\in[\ell]} g_i^{x_i}$, is 2-universal.*

Note that the group $\mathbb{G}$ needs not be cyclic.

*Proof.* Consider $\mathbf{x} \neq \mathbf{y}$ and assume w.l.o.g that $x_1 \neq y_1$. Then for $g_1, \dots, g_\ell \xleftarrow{\$} \mathbb{G}$ it holds that $g_1^{x_1-y_1}$ is uniformly distributed in $\mathbb{G}$ and the result follows. ☐

**The Goldreich-Levin Theorem**

The Goldreich-Levin hard core predicate is stated in the following theorem.

**Theorem 4.3.5** ([GL89])**.** *Let $f : \{0,1\}^n \to \{0,1\}^*$ be any (possibly randomized) function and let $\mathcal{A}$ be such that*

$$\left| \Pr_{\mathbf{x},\mathbf{r}\xleftarrow{\$}\{0,1\}^n}[\mathcal{A}(f(\mathbf{x}), \mathbf{r}, \langle\mathbf{r},\mathbf{x}\rangle) = 1] - \Pr_{\substack{\mathbf{x},\mathbf{r}\xleftarrow{\$}\{0,1\}^n,\\ u\xleftarrow{\$}\{0,1\}}}[\mathcal{A}(f(\mathbf{x}), \mathbf{r}, u) = 1] \right| \ge \epsilon \ ,$$

*then there exists $\mathcal{B}^{\mathcal{A}}$ that runs in time* $\mathrm{poly}(n, 1/\epsilon)$ *such that*

$$\Pr[\mathcal{B}^{\mathcal{A}}(f(\mathbf{x})) = \mathbf{x}] \ge \Omega(\epsilon^3/n) \ .$$

### A Generalized Goldreich-Levin Theorem

We use a generalized version of Theorem 4.3.5, essentially adopted from [DGK+10, Theorem 1] and slightly adapted (see explanation below).

**Theorem 4.3.6** (adapted from [DGK+10, Theorem 1])**.** *Let $f : \{0,1\}^n \to \{0,1\}^*$ be any (possibly randomized) function, let $K \in \mathbb{N}$, $K > 1$ and let $\mathcal{A}$ be such that*

$$\Big|\Pr[\mathcal{A}(f(\mathbf{x}), \mathbf{r}, \langle \mathbf{r}, \mathbf{x}\rangle) = 1] - \Pr[\mathcal{A}(f(\mathbf{x}), \mathbf{r}, u) = 1]\Big| \geq \epsilon \ ,$$

*where $\mathbf{x} \xleftarrow{\$} \{0,1\}^n, \mathbf{r} \xleftarrow{\$} \mathbb{Z}_K^n, u \xleftarrow{\$} \mathbb{Z}_K$ and the inner product is over $\mathbb{Z}_K$, then there exists $\mathcal{B}^{\mathcal{A}}$ that runs in time* $\mathrm{poly}(n, 1/\epsilon)$ *such that*

$$\Pr[\mathcal{B}^{\mathcal{A}}(f(\mathbf{x})) = \mathbf{x}] \geq \frac{\epsilon}{8 \cdot K^{1+\log(8n/\epsilon^2)}} \ .$$

We remark that the theorem extends to groups that are isomorphic to $\mathbb{Z}_K$ since the adversary $\mathcal{B}$ only needs “oracle access” to $\mathbb{Z}_K$ allowing it to sample elements uniformly, add elements and compare elements.

Comparing our theorem statement to [DGK+10, Theorem 1] shows a number of differences. We allow $K$ to take any value (so long as operations over $\mathbb{Z}_K$ are efficient), while they restricted their attention to prime $K$. This has a technical effect on the proof, since we need to find a big set of elements $\{\rho_i\} \subseteq \mathbb{Z}_K$ such that for all $i \neq j$, $(\rho_i - \rho_j) \in \mathbb{Z}_K^*$ (namely, is a unit in the ring $\mathbb{Z}_K$). This is easy for prime $K$, but harder for a general $K$, whose factorization is unknown. Therefore, in our proof, we use a trivial set with the aforementioned property, $\{0,1\}$, which implies a degradation in parameters: the power of $K$ in the success probability of the inverter is logarithmic in our statement, but constant (specifically, 2) in [DGK+10, Theorem 1]. We remark that if $\mathbb{Z}_K$ is such that finding a non-unit is computationally hard (one example is $\mathbb{Z}_N$ for an RSA number $N$), then similar parameters to [DGK+10, Theorem 1] can be achieved, using the same techniques.

An additional change is that we only allow $f : \{0,1\}^n \to \{0,1\}^*$, while their theorem applies to $f : H^n \to \{0,1\}^*$, for any set $H \subseteq \mathbb{Z}_K$ of polynomial cardinality. This change is mostly to simplify the theorem statement and proof.

The formal proof follows.

*Proof.* Given an algorithm $\mathcal{A}$ as stated in the theorem, it holds that

$$\Pr_{\mathbf{x}}\left[\left|\Pr_{\mathbf{r}}[\mathcal{A}(f(\mathbf{x}), \mathbf{r}, \langle \mathbf{r}, \mathbf{x}\rangle) = 1] - \Pr_{\mathbf{r},u}[\mathcal{A}(f(\mathbf{x}), \mathbf{r}, u) = 1]\right| > \epsilon/2\right] \geq \epsilon/2 \ .$$

Our reduction $\mathcal{B}^{\mathcal{A}}$ will succeed with sufficiently high probability for $y = f(\mathbf{x})$ for which

$$\Pr_{\mathbf{r}}[\mathcal{A}(f(\mathbf{x}), \mathbf{r}, \langle \mathbf{r}, \mathbf{x}\rangle) = 1] - \Pr_{\mathbf{r},u}[\mathcal{A}(f(\mathbf{x}), \mathbf{r}, u) = 1] > \epsilon/2 \ .$$

Note that we removed the absolute value, which incurs a factor of $1/2$ in the final success probability. From now on, condition on this event.

The reduction $\mathcal{B}^{\mathcal{A}}$, on an input $y = f(\mathbf{x})$, runs as follows. Define $m = 8n/\epsilon^2$ and let $c = 1 + \lfloor \log m \rfloor$. Sample $\mathbf{z}_1, \ldots, \mathbf{z}_c \xleftarrow{\$} \mathbb{Z}_K^n$ and $g_1, \ldots, g_c \xleftarrow{\$} \mathbb{Z}_K$. With probability $1/K^c \geq 1/K^{1+\log(8n/\epsilon^2)}$ it holds that for all $j \in [c]$, $\langle \mathbf{z}_j, \mathbf{x} \rangle = g_j$. From now on, condition on this event as well.

For all $\boldsymbol{\rho} \in \{0,1\}^c \setminus \{\mathbf{0}\}$, define $\mathbf{r}_{\boldsymbol{\rho}} = \sum_{j \in [c]} \rho_j \cdot \mathbf{z}_j$ and $h_{\boldsymbol{\rho}} = \sum_{j \in [c]} \rho_j \cdot g_j$. Then $\{\mathbf{r}_{\boldsymbol{\rho}}\}$ are uniformly distributed, pairwise independent (this is where we use the fact that $\pm 1$ are units in $\mathbb{Z}_K$) and it holds that $\langle \mathbf{r}_{\boldsymbol{\rho}}, \mathbf{x} \rangle = h_{\boldsymbol{\rho}}$.

Next, it samples $\tau_{\boldsymbol{\rho}} \xleftarrow{\$} \mathbb{Z}_K$ for all $\boldsymbol{\rho}$ and computes, for all $i \in [n]$ and $\boldsymbol{\rho} \in \{0,1\}^c \setminus \{0\}$,

$$s_{i,\boldsymbol{\rho}} = \mathcal{A}(y, \mathbf{r}_{\boldsymbol{\rho}} + \tau_{\boldsymbol{\rho}} \cdot \mathbf{e_i}, h_{\boldsymbol{\rho}} + \tau_{\boldsymbol{\rho}}) - \mathcal{A}(y, \mathbf{r}_{\boldsymbol{\rho}} + \tau_{\boldsymbol{\rho}} \cdot \mathbf{e_i}, h_{\boldsymbol{\rho}}) \ ,$$

it then computes $s_i = \mathbb{E}_{\boldsymbol{\rho}}[s_{i,\boldsymbol{\rho}}] = \frac{\sum_{\boldsymbol{\rho}} s_{i,\rho}}{2^c - 1}$ (i.e. we consider the uniform distribution over all values of $\boldsymbol{\rho}$) and if $s_i \geq 0$, it sets $x_i = 1$, otherwise it sets $x_i = 0$.

To analyze, we notice that $s_{i,\boldsymbol{\rho}}$ is distributed like $\mathcal{A}(f(\mathbf{x}), \mathbf{r}, \langle \mathbf{r}, \mathbf{x} \rangle) - \mathcal{A}(f(\mathbf{x}), \mathbf{r}, u)$ if $x_i = 1$ and like $\mathcal{A}(f(\mathbf{x}), \mathbf{r}, u) - \mathcal{A}(f(\mathbf{x}), \mathbf{r}, \langle \mathbf{r}, \mathbf{x} \rangle)$ if $x_i = 0$. Thus, for any fixed value of $\boldsymbol{\rho}$, $\mathbb{E}_{\mathbf{r}_{\boldsymbol{\rho}}}[s_{i,\boldsymbol{\rho}}] \geq \epsilon/2$ if $x_i = 1$ and $\mathbb{E}_{\mathbf{r}_{\boldsymbol{\rho}}}[s_{i,\boldsymbol{\rho}}] \leq -\epsilon/2$ if $x_i = 0$. Since $\{s_{i,\boldsymbol{\rho}}\}_{\boldsymbol{\rho}}$ are pairwise independent, we can apply Chebishev's inequality and get that with probability at least $1 - 1/(m \cdot (\epsilon/2)^2) \geq 1 - 1/(2n)$, it holds that $s_i$ is within $\epsilon/2$ of its expected value, in which case $x_i$ is computed correctly. Applying the union bound over all $i$, we have that $\mathbf{x}$ is computed correctly with probability $1/2$.

Combining all of the terms above, the result follows. □

## 4.4 Subgroup Indistinguishability Assumptions

We present the class of subgroup indistinguishability assumptions in Section 4.4.1 and then discuss instantiations under the QR and DCR assumptions in Section 4.4.2.

### 4.4.1 Definition of a Subgroup Indistinguishability (SG) Problem

Let $\mathbb{G}_U$ be a finite commutative multiplicative group, such that $\mathbb{G}_U$ can be decomposed as an internal direct product: $\mathbb{G}_U = \mathbb{G}_M \times \mathbb{G}_L$.[9] The "message group" $\mathbb{G}_M$ is cyclic of order $M$, the "language group" $\mathbb{G}_L$ is of order $L$ (and is not necessarily cyclic), and $\mathbb{G}_U$ is of order $M \cdot L$ (we use $M, L$ both to index the groups and to denote their orders). We require that $\gcd(M, L) = 1$. Let $h$ be a generator for $\mathbb{G}_M$ such that $h$ is efficiently computable from the description of $\mathbb{G}_U$. We require that there exists an efficient algorithm $OP_{\mathbb{G}_U}$ to perform group operations in $\mathbb{G}_U$, and also that there exist efficient sampling algorithms $S_{\mathbb{G}_M}$, $S_{\mathbb{G}_L}$ that sample a random element from $\mathbb{G}_M$, $\mathbb{G}_L$ respectively. We further require that an upper bound $T \geq M \cdot L$ is known.

We stress that, as always, all groups described above are in fact families of groups, indexed by the security parameter $k$. To be more precise, there exists a polynomial time randomized algorithm that given the security parameter $1^k$, outputs $I_{\mathbb{G}_U} = (OP_{\mathbb{G}_U}, S_{\mathbb{G}_M}, S_{\mathbb{G}_L}, h, T)$. We refer to $I_{\mathbb{G}_U}$ as *an instance* of $\mathbb{G}_U$.

For any adversary $\mathcal{A}$ we denote the *subgroup distinguishing advantage* of $\mathcal{A}$ by

$$\mathrm{SGAdv}[\mathcal{A}] = \left| \Pr_{x \xleftarrow{\$} \mathbb{G}_U} [\mathcal{A}(1^k, x)] - \Pr_{x \xleftarrow{\$} \mathbb{G}_L} [\mathcal{A}(1^k, x)] \right| \ .$$

[9]Internal direct product means that $\mathbb{G}_M, \mathbb{G}_L$ are subgroups of $\mathbb{G}_U$, and every element in $\mathbb{G}_U$ can be written as $h \cdot g$, where $h \in \mathbb{G}_M$ and $g \in \mathbb{G}_L$.

That is, the advantage $\mathcal{A}$ has in distinguishing between $\mathbb{G}_U$ and $\mathbb{G}_L$. The *subgroup indistinguishability* (SG) assumption is that for any polynomial $\mathcal{A}$ it holds that for a properly sampled instance $I_{\mathbb{G}_U}$, we have $\mathrm{SGAdv}[\mathcal{A}] = \mathrm{negl}(k)$ (note that in such case it must be that $1/L = \mathrm{negl}(k)$). In other words, thinking of $\mathbb{G}_L \subseteq \mathbb{G}_U$ as a language, the assumption is that this language is hard on average. We define an additional flavor of the assumption by

$$\mathrm{SG'Adv}[\mathcal{A}] = \left| \Pr_{x \overset{\$}{\leftarrow} \mathbb{G}_L}[\mathcal{A}(1^k, h \cdot x)] - \Pr_{x \overset{\$}{\leftarrow} \mathbb{G}_L}[\mathcal{A}(1^k, x)] \right| .$$

It follows immediately that for any adversary $\mathcal{A}$ there exists an adversary $\mathcal{B}$ such that $\mathrm{SG'Adv}[\mathcal{A}] \leq 2 \cdot \mathrm{SGAdv}[\mathcal{B}]$.

### 4.4.2 Instantiations

We instantiate the SG assumption based on the QR and DCR assumptions.

For both instantiations we consider a modulus $N$ of the form $N = pq$, where $p, q$ are random $k$-bit odd primes (such a modulus is sometimes called an “RSA number”). Note that we will not require that $p, q$ are “safe primes”.

#### Instantiation Under the QR Assumption with Any Blum Integer

Consider a modulus $N$ as described above. We use $\mathbb{J}_N$ to denote the set of elements in $\mathbb{Z}_N^*$ with Jacobi symbol $+1$, we use $\mathbb{QR}_N$ to denote the set of *quadratic residues* (squares) modulo $N$. Overloading notation, $\mathbb{J}_N, \mathbb{QR}_N$ also denote the respective groups with the multiplication operation modulo $N$. The groups $\mathbb{J}_N, \mathbb{QR}_N$ have orders $\frac{\varphi(N)}{2}, \frac{\varphi(N)}{4}$ respectively and we denote $N' = \frac{\varphi(N)}{4}$. We require that $N$ is a *Blum integer*, namely that $p, q = 3 \pmod 4$. In such case it holds that $\gcd(2, N') = 1$ and that $(-1) \in \mathbb{J}_N \setminus \mathbb{QR}_N$.

The *quadratic residuosity* (QR) assumption is that for a properly generated $N$, the distributions $U(\mathbb{J}_N)$ and $U(\mathbb{QR}_N)$ are computationally indistinguishable.[10] This leads to the immediate instantiation of the SG assumption by setting $\mathbb{G}_U = \mathbb{J}_N$, $\mathbb{G}_M = \{\pm 1\}$, $\mathbb{G}_L = \mathbb{QR}_N$, $h = (-1)$, $T = N \geq 2N'$.

#### Instantiation Under the DCR Assumption

The *decisional composite residuosity* (DCR) assumption, introduced by Paillier [Pai99], states that for a properly generated RSA number $N$, it is hard to distinguish between a random element in $\mathbb{Z}_{N^2}^*$ and a random element in the subgroup of $N^{\text{th}}$-residues $\{x^N : x \in \mathbb{Z}_{N^2}^*\}$. The group $\mathbb{Z}_{N^2}^*$ can be written as a product of the group generated by $1 + N$ (which has order $N$) and the group of $N^{\text{th}}$ residues (which has order $\varphi(N)$). This implies that setting $\mathbb{G}_U = \mathbb{Z}_{N^2}^*$, $\mathbb{G}_L = \{x^N : x \in \mathbb{Z}_{N^2}^*\}$ and $\mathbb{G}_M = \{(1+N)^i : i \in [N]\}$ provides an instantiation of the SG assumption, setting $h = (1+N)$ and $T = N^2$. It is left to check that indeed $\gcd(N, \varphi(N)) = 1$. This follows since $p, q$ are odd primes of

[10]The QR assumption usually refers to random RSA numbers, which are not necessarily Blum integers. However, since Blum integers have constant density among RSA numbers, the flavor we use is implied.

equal length: assume w.l.o.g that $p/2 < q < p$, then the largest prime divisor of $\varphi(N) = (p-1)(q-1)$ has size at most $(p-1)/2 < p, q$ and the claim follows.[11]

## 4.5 The New Encryption Scheme (Under Quadratic Residuosity)

In the interest of clarity, we first present the QR-based scheme. The general case is presented in Section 4.8.1. The scheme $\mathcal{E}[\ell]$ is defined below.

- **Parameters.** The scheme is parameterized by $\ell \in \mathbb{N}$ which is polynomial in the security parameter. The exact value of $\ell$ is determined based on the specific properties we require of the scheme.

  The message space of $\mathcal{E}[\ell]$ is $\mathcal{M} = \{0,1\}$, i.e. this is a bit-by-bit encryption scheme.

- **Key generation.** The key generator first samples a Blum integer $N$, which can be used as public parameter and only be sampled once for all users. Furthermore, no entity needs to know the factorization of $N$. We assume that $N$ is an implicit input to all of our algorithms.

  The key generator also samples $\mathbf{s} \xleftarrow{\$} \{0,1\}^\ell$ and sets $sk = \mathbf{s}$. It then samples $\mathbf{g} \xleftarrow{\$} \mathbb{QR}_N^\ell$ and sets $g_0 = (\prod_{i\in[\ell]} g_i^{s_i})^{-1}$. The public key is set to be $pk = (g_0, \mathbf{g})$ (with $N$ as an additional implicit public parameter).

- **Encryption.** On inputs a public key $pk = (g_0, \mathbf{g})$ and a message $m \in \{0,1\}$, the encryption algorithm runs as follows: it samples $r \xleftarrow{\$} [N^2]$,[12] and computes $\mathbf{c} = \mathbf{g}^r$ and $c_0 = (-1)^m \cdot g_0^r$. It outputs a ciphertext $(c_0, \mathbf{c})$.

- **Decryption.** On inputs a secret key $sk = \mathbf{s}$ and a ciphertext $(c_0, \mathbf{c})$, the decryption algorithm computes $(-1)^m = c_0 \cdot \prod_{i\in[\ell]} c_i^{s_i}$ and outputs $m$.

The correctness of the scheme follows immediately by definition, security will be treated in the following sections.

We note that correctness holds regardless of the distribution of the vector $\mathbf{s}$. Indeed, we can use secret keys that are not binary strings. The implications are discussed in Section 4.8.7.

[11] If greater efficiency is desired, we can use a generalized form of the assumption, presented in [DJ01]. Let $d \geq 1$ be a parameter that is polynomial in the security parameter and consider the group $\mathbb{G}_U = \mathbb{Z}^*_{N^{d+1}}$. Then $\mathbb{G}_U$ can be written as a product $\mathbb{G}_U = \mathbb{G}_M \times \mathbb{G}_L$ where $\mathbb{G}_M$ is cyclic and has order $M = N^d$ and generator $h = (N+1)$. The group $\mathbb{G}_L$ is the group $\{x^{N^d} : x \in \mathbb{Z}^*_{N^{d+1}}\}$ which is isomorphic to $\mathbb{Z}^*_N$. Clearly $\gcd(N^d, \varphi(N)) = 1$ and we can use the bound $T = N^{d+1} \geq N^d \cdot \varphi(N)$.

It is proven in [DJ01] that under the DCR assumption, the subgroup indistinguishability problem defined by the above groups is hard for any polynomial $d$. Specifically, taking $d = 1$ gives Paillier's original assumption.

[12] A more natural choice is to sample $r \xleftarrow{\$} [|\mathbb{J}_N|]$, but since $|\mathbb{J}_N| = 2N' = \frac{\varphi(N)}{2}$ is hard to compute, we cannot sample from this distribution directly. However, since $r$ is used as an exponent of a group element, it is sufficient that $(r \bmod 2N')$ is uniform in $\mathbb{Z}_{2N'}$, and this is achieved by sampling $r$ from a much larger domain.

Alternatively, we can use $r \xleftarrow{\$} [(N-1)/2]$, since $U([(N-1)/2])$ and $U([\varphi(N)/2])$ are statistically indistinguishable.

## 4.6 The Adaptive Vector Game (Under Quadratic Residuosity)

We define the *adaptive $\ell$-vector* game played between a challenger and an adversary. We only present the QR-based game in this section, and refer the reader to the general definition in Section 4.8.2.

- **Initialize.** The challenger samples $b \xleftarrow{\$} \{0,1\}$ and also generates a Blum integer $N$ and a vector $\mathbf{g} \xleftarrow{\$} \mathbb{QR}_N^\ell$. It sends $N$ and $\mathbf{g}$ to the adversary.
- **Query.** The adversary adaptively makes queries, where each query is a vector $\mathbf{a} \in \{0,1\}^\ell$. For each query $\mathbf{a}$, the challenger samples $r \xleftarrow{\$} [N^2]$ and returns $(-1)^{\mathbf{a}} \cdot \mathbf{g}^r$ if $b = 0$, or $\mathbf{g}^r$ if $b = 1$.
- **Finish.** The adversary outputs a guess $b' \in \{0,1\}$.

The advantage of an adversary $\mathcal{A}$ in the game is defined to be

$$\mathrm{AV}_\ell\mathrm{Adv}[\mathcal{A}] = \left|\Pr[b' = 1|b = 0] - \Pr[b' = 1|b = 1]\right| \ .$$

Under the QR assumption, no poly($k$)-time adversary (where $k$ is the security parameter) can obtain a non-negligible advantage in the game, as formally stated below.

**Lemma 4.6.1.** *Let $\mathcal{A}$ be an adversary for the adaptive $\ell$-vector game that makes at most $t$ queries, then there exists an adversary $\mathcal{B}$ for* QR *such that*

$$\mathrm{AV}_\ell\mathrm{Adv}[\mathcal{A}] \le 4t\ell \cdot \mathrm{QRAdv}[\mathcal{B}] + 2t\ell/N \ .$$

*Proof.* A standard hybrid argument implies the existence of $\mathcal{A}_1$ which is an adversary for a 1-round game ($t = 1$ in our notation) such that $\mathrm{AV}_\ell\mathrm{Adv}[\mathcal{A}] \le t \cdot \mathrm{AV}_\ell\mathrm{Adv}[\mathcal{A}_1]$.

We consider a series of hybrids (experiments). For each hybrid $H_i$, we let $\Pr[H_i]$ denote the probability that the experiment "succeeds" (an event we define below).

- **Hybrid $H_0$.** In this experiment, we flip a coin $b \xleftarrow{\$} \{0,1\}$ and also sample $i \xleftarrow{\$} [\ell]$. We simulate the 1-round game with $\mathcal{A}_1$, where the challenger answers a query $\mathbf{a}$ with $(g_1^r, \ldots, g_{i-1}^r, (-1)^{b \cdot a_i} \cdot g_i^r, (-1)^{a_{i+1}} \cdot g_{i+1}^r, \ldots, (-1)^{a_\ell} \cdot g_\ell^r)$. The experiment succeeds if $b' = b$.

  A standard argument shows that

$$\frac{\mathrm{AV}_\ell\mathrm{Adv}[\mathcal{A}_1]}{2\ell} = \left|\Pr[H_0] - \frac{1}{2}\right| \ .$$

- **Hybrid $H_1$.** In this hybrid we replace $g_i$ (which is a uniform square) with $(-g_i)$. We get that there exists $\mathcal{B}$ such that $|\Pr[H_1] - \Pr[H_0]| \le 2 \cdot \mathrm{QRAdv}[\mathcal{B}]$.

  We note that in this hybrid, the adversary's query is answered with $(g_1^r, \ldots, g_{i-1}^r, (-1)^{b \cdot a_i} \cdot (-g_i)^r, (-1)^{a_{i+1}} \cdot g_{i+1}^r, \ldots, (-1)^{a_\ell} \cdot g_\ell^r)$.

- **Hybrid $H_2$.** In this hybrid, the only change is that now $r \xleftarrow{\$} \mathbb{Z}_{2N'}$ (recall that $N' = \frac{\varphi(N)}{4}$) rather than $U([N^2])$. By Lemma 4.3.1 it follows that $|\Pr[H_2] - \Pr[H_1]| \le 1/N$. We note that while $N'$ is not explicitly known to any entity, the argument here is statistical and there is no requirement that this hybrid is efficiently simulated.

  We denote $r_1 = (r \bmod 2)$ and $r_2 = (r \bmod N')$. Since $N'$ is odd, the Chinese Remainder Theorem implies that $r_1, r_2$ are uniform in $\mathbb{Z}_2, \mathbb{Z}_{N'}$, respectively, and are independent. The answer to the query in this scenario is therefore

$$(g_1^r, \ldots, g_{i-1}^r, (-1)^{b \cdot a_i} \cdot (-g_i)^r, (-1)^{a_{i+1}} \cdot g_{i+1}^r, \ldots, (-1)^{a_\ell} \cdot g_\ell^r) = \\ (g_1^{r_2}, \ldots, g_{i-1}^{r_2}, (-1)^{b \cdot a_i + r_1} \cdot g_i^{r_2}, (-1)^{a_{i+1}} \cdot g_{i+1}^{r_2}, \ldots, (-1)^{a_\ell} \cdot g_\ell^{r_2}) \ .$$

  However since $r_1$ is a uniform bit, the answer is independent of $b$. It follows that $\Pr[H_2] = \frac{1}{2}$.

It follows that $\mathrm{AV}_\ell\mathrm{Adv}[\mathcal{A}_1] \le 4\ell \cdot \mathrm{QRAdv}[\mathcal{B}] + 2\ell/N$, and the result follows. □

## 4.7 Security Under Quadratic Residuosity

### 4.7.1 CPA Security

We start by proving the basic property of CPA security for our QR-based scheme. This will serve as a warm-up towards proving the stronger properties. We prove the CPA-security of $\mathcal{E}[\ell]$, for $\ell \ge \log N + \omega(\log k)$. Proof of CPA security under subgroup indistinguishability is provided in Section 4.8.3.

Intuitively, CPA security follows since the view of the adversary contains that public key $(g_0, \mathbf{g})$ and the ciphertext $((-1)^m \cdot g_0^r, \mathbf{g}^r)$. If $g_0$ was uniform, then this would have been identical to its view in a 1-round adaptive $(\ell+1)$-vector game, and security would follow. To complete the proof, therefore, we use the leftover hash lemma to prove that $g_0$ is statistically indistinguishable from uniform. A formal statement and proof follows.

**Theorem 4.7.1.** *Let $\mathcal{A}$ be a* CPA*-adversary for $\mathcal{E}[\ell]$, then there exists an adversary $\mathcal{B}$ such that*

$$\mathrm{CPAAdv}[\mathcal{A}] \le 4(\ell+1) \cdot \mathrm{QRAdv}[\mathcal{B}] + \sqrt{N \cdot 2^{-\ell}} + O(\ell/N) \ .$$

The theorem implies that taking $\ell = \log N + \omega(\log k)$ is sufficient to obtain CPA-security.

*Proof.* The proof proceeds by a series of hybrids.

- **Hybrid $H_0$.** This hybrid is identical to the CPA game with $b = 0$. By definition $\Pr_{H_0}[b' = 1] = \Pr[b' = 1 | b = 0]$.

- **Hybrid $H_1$.** In this hybrid, we change the distribution of $g_0$, which will now be sampled from $U(\mathbb{QR}_N)$. By Lemma 4.3.4, combined with Lemma 4.3.2, $(g_0, \mathbf{g})$ is $\sqrt{\frac{N'}{2^{\ell+2}}} \le \sqrt{\frac{N}{2^{\ell+2}}}$-uniform. Thus

$$\left| \Pr_{H_1}[b' = 1] - \Pr_{H_0}[b' = 1] \right| \le \sqrt{\frac{N}{2^{\ell+2}}} \ .$$

- **Hybrid $H_2$.** In this hybrid, we change the way the challenger answers queries. Now instead of answering $(c_0, \mathbf{c}) = ((-1)^m \cdot g_0^r, \mathbf{g}^r))$, the challenger answers $(c_0, \mathbf{c}) = (g_0^r, \mathbf{g}^r)$. The difference between $H_1$ and $H_2$ is now a 1-query adaptive $(\ell + 1)$-vector game and thus by Lemma 4.6.1,

$$\left|\Pr_{H_2}[b' = 1] - \Pr_{H_1}[b' = 1]\right| \leq 4(\ell+1) \cdot \mathrm{QRAdv}[\mathcal{B}] + O(\ell/N) \ ,$$

  for some $\mathcal{B}$.

- **Hybrid $H_3$.** We now revert the distribution of $g_0$ back to the original $\prod_{i \in [\ell]} g_i^{-s_i}$. Similarly to $H_1$, we have

$$\left|\Pr_{H_3}[b' = 1] - \Pr_{H_2}[b' = 1]\right| \leq \sqrt{\frac{N}{2^{\ell+2}}} \ .$$

  However, hybrid $H_3$ is identical to the CPA game with $b = 1$, as all queries are answered by encryptions of 0: $\Pr_{H_3}[b' = 1] = \Pr[b' = 1 | b = 1]$. Summing the terms above, the result follows. □

### 4.7.2 KDM Security

In this section, we discuss the KDM-security related properties of our scheme. We only discuss our QR-based encryption scheme in this section, in the interest of clarity. We prove the KDM$^{(1)}$-security of $\mathcal{E}[\ell]$, for $\ell \geq \log N + \omega(\log k)$, in Section 4.7.2.1. Then, in Section 4.7.2.2, we state and prove that for $\ell \geq n \cdot \log N + \omega(\log k)$, $\mathcal{E}[\ell]$ is also KDM$^{(n)}$-secure. Finally, extensions beyond affine functions are stated without proof in Section 4.7.2.3.

A presentation and analysis of the general case, including all relevant proofs, is provided in Section 4.8.4.

Throughout this section, we define $\mathcal{F}_{\text{aff}}$ to be the class of affine functions over $\mathbb{Z}_2$, namely the class of all functions of the form $f_{a_0,\mathbf{a}}(\mathbf{x}) = a_0 + \sum a_i x_i$, where $a_i, x_i \in \mathbb{Z}_2$, and arithmetics are also over $\mathbb{Z}_2$.

#### 4.7.2.1 KDM$^{(1)}$-Security

The intuition behind the KDM$^{(1)}$-security of $\mathcal{E}[\ell]$ is as follows. Consider a public key $(g_0 = \prod g_i^{-s_i}, \mathbf{g})$ that corresponds to a secret key $\mathbf{s}$, and a function $f_{a_0,\mathbf{a}} \in \mathcal{F}_{\text{aff}}$. The encryption of $f_{a_0,\mathbf{a}}(\mathbf{s}) = (-1)^{a_0 + \sum a_i s_i}$ is

$$(c_0, \mathbf{c}) = ((-1)^{a_0 + \sum a_i s_i} \cdot g_0^r, \mathbf{g}^r) = ((-1)^{a_0} \cdot \prod((-1)^{a_i} \cdot g_i^r)^{-s_i}, \mathbf{g}^r) \ .$$

We notice that if $\mathbf{s}, a_0, \mathbf{a}$ are known, then $c_0$ is completely determined by $\mathbf{c} = \mathbf{g}^r$. Therefore, if we replace $\mathbf{g}^r$ with $(-1)^{\mathbf{a}} \cdot \mathbf{g}^r$ (an indistinguishable vector, even given the public key, by an adaptive vector game), we see that $(c_0, \mathbf{c})$ is indistinguishable from $(c_0', \mathbf{c}') = ((-1)^{a_0} \cdot g_0^r, (-1)^{\mathbf{a}} \cdot \mathbf{g}^r)$, even when the secret key and the message are known. Applying the same argument again, taking into account that $g_0$ is close to uniform, implies that $(c_0', \mathbf{c}')$ is computationally indistinguishable from $(g_0^r, \mathbf{g}^r)$, which is an encryption of 0. A formal statement and analysis follow.

**Theorem 4.7.2.** *Let $\mathcal{A}$ be a $\mathrm{KDM}^{(1)}_{\mathcal{F}_{\mathrm{aff}}}$-adversary for $\mathcal{E}[\ell]$ that makes at most $t$ queries, then there exists an adversary $\mathcal{B}$ such that*

$$\mathrm{KDM}^{(1)}_{\mathcal{F}_{\mathrm{aff}}}\mathrm{Adv}[\mathcal{A}] \leq 4t(2\ell+1) \cdot \mathrm{QRAdv}[\mathcal{B}] + \sqrt{N \cdot 2^{-\ell}} + O(t\ell/N) \ .$$

The theorem implies that taking $\ell = \log N + \omega(\log k)$ is sufficient to obtain $\mathrm{KDM}^{(1)}$-security.

*Proof.* The proof proceeds by a series of hybrids.

- **Hybrid $H_0$.** This hybrid is identical to the $\mathrm{KDM}^{(1)}$ game with $b = 0$. By definition $\Pr_{H_0}[b' = 1] = \Pr[b' = 1 | b = 0]$.
- **Hybrid $H_1$.** In this hybrid, we change the way the challenger answers the adversary's queries. Recall that in hybrid $H_0$, the query $f_{a_0,\mathbf{a}} \in \mathcal{F}_{\mathrm{aff}}$ was answered by $(c_0, \mathbf{c}) = ((-1)^{a_0+\sum a_i s_i} \cdot g_0^r, \mathbf{g}^r)$. In hybrid $H_1$, it will be answered by $(c_0, \mathbf{c}) = ((-1)^{a_0} \cdot g_0^r, (-1)^{\mathbf{a}} \cdot \mathbf{g}^r)$.

  We prove that

  $$\left|\Pr_{H_1}[b' = 1] - \Pr_{H_0}[b' = 1]\right| \leq \mathrm{AV}_\ell\mathrm{Adv}[\mathcal{A}'] \leq 4t\ell \cdot \mathrm{QRAdv}[\mathcal{B}_1] + O(t\ell/N) \ ,$$

  for some $\mathcal{A}', \mathcal{B}_1$, even when $\mathbf{s}$ is fixed and known.

  To see this, we notice that in both hybrids $c_0 = (-1)^{a_0} \cdot \prod_{i\in[\ell]}((-1)^{a_i} \cdot c_i^{-1})^{s_i}$ and $g_0 = \prod_{i\in[\ell]} g_i^{-s_i}$. Therefore an adversary $\mathcal{A}'$ for the adaptive $\ell$-vector game can simulate $\mathcal{A}$, sampling $\mathbf{s}$ on its own and using $\mathbf{g}$ to generate $g_0$ and "translate" the challenger answers. Applying Lemma 4.6.1, the result follows.
- **Hybrid $H_2$.** In this hybrid, we change the distribution of $g_0$, which will now be sampled from $U(\mathbb{QR}_N)$. By Lemma 4.3.4, combined with Lemma 4.3.2, $(g_0, \mathbf{g})$ is $\sqrt{\frac{N'}{2^{\ell+2}}} \leq \sqrt{\frac{N}{2^{\ell+2}}}$-uniform. Thus

  $$\left|\Pr_{H_2}[b' = 1] - \Pr_{H_1}[b' = 1]\right| \leq \sqrt{\frac{N}{2^{\ell+2}}} \ .$$
- **Hybrid $H_3$.** In this hybrid, we again change the way the challenger answers queries. Now instead of answering $(c_0, \mathbf{c}) = ((-1)^{a_0} \cdot g_0^r, (-1)^{\mathbf{a}} \cdot \mathbf{g}^r))$, the challenger answers $(c_0, \mathbf{c}) = (g_0^r, \mathbf{g}^r)$. The difference between $H_2$ and $H_3$ is now a $t$-query adaptive $(\ell+1)$-vector game and thus by Lemma 4.6.1,

  $$\left|\Pr_{H_3}[b' = 1] - \Pr_{H_2}[b' = 1]\right| \leq 4t(\ell+1) \cdot \mathrm{QRAdv}[\mathcal{B}_2] + O(t\ell/N) \ ,$$

  for some $\mathcal{B}_2$.
- **Hybrid $H_4$.** We now revert the distribution of $g_0$ back to the original $\prod_{i\in[\ell]} g_i^{-s_i}$. Similarly to $H_2$, we have

  $$\left|\Pr_{H_4}[b' = 1] - \Pr_{H_3}[b' = 1]\right| \leq \sqrt{\frac{N}{2^{\ell+2}}} \ .$$

However, hybrid $H_4$ is identical to the $\mathrm{KDM}^{(1)}$ game with $b = 1$, as all queries are answered by encryptions of 0: $\Pr_{H_4}[b' = 1] = \Pr[b' = 1|b = 1]$. Summing the terms above, the result follows (where $\mathcal{B}$ is, say, a weighted average between $\mathcal{B}_1$ and $\mathcal{B}_2$). □

#### 4.7.2.2 $\mathrm{KDM}^{(n)}$-Security

A formal statement and proof for the QR case follows. For the statement and proof in the general case, see Section 4.8.4.3.

**Theorem 4.7.3.** *Let $\mathcal{A}$ be a $\mathrm{KDM}^{(n)}_{\mathcal{F}_{\mathrm{aff}}}$-adversary for $\mathcal{E}[\ell]$ that makes at most $t$ queries, then there exists an adversary $\mathcal{B}$ such that*

$$\mathrm{KDM}^{(n)}_{\mathcal{F}_{\mathrm{aff}}}\mathrm{Adv}[\mathcal{A}] \le 4nt(2\ell + 1) \cdot \mathrm{QRAdv}[\mathcal{B}] + (N \cdot 2^{-\ell/n})^{n/2} + O(nt\ell/N) \ .$$

The theorem implies that taking $\ell = n \cdot \log N + \omega(\log k)$ is sufficient for $\mathrm{KDM}^{(n)}$-security.

*Proof.* Let us first introduce the notation used in the proof: We now consider functions in $\mathcal{F}_{\mathrm{aff}}$ that are applied to a concatenated vector of $n$ secret keys. We will denote such functions by $f_{a_0,\mathbf{a}_1,\ldots,\mathbf{a}_n}$, where $a_0 \in \{0, 1\}$ and $\mathbf{a}_j \in \{0, 1\}^\ell$, and such that for all $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \{0, 1\}^\ell$ the function is defined by

$$f_{a_0,\mathbf{a}_1,\ldots,\mathbf{a}_n}(\mathbf{x}_1, \ldots, \mathbf{x}_n) = a_0 + \sum_{j\in[n],i\in[\ell]} a_{j,i} \cdot x_{j,i} \ .$$

All arithmetic operations are over $\mathbb{Z}_2$.

The proof follows the outline of the proof of Theorem 4.7.2 with slight modifications to the hybrids.

- **Hybrid $H_0$.** This hybrid is identical to the $\mathrm{KDM}^{(n)}$ game with $b = 0$. Let $\{sk_i = \mathbf{s}_i\}_{i\in[n]}$ denote the generated secret keys and $\{pk_i = (g_{0,i}, \mathbf{g}_i)\}_{i\in[n]}$ denote the public keys produced by the challenger. By definition $\Pr_{H_0}[b' = 1] = \Pr[b' = 1|b = 0]$.

- **Hybrid $H_1$.** We change the way the challenger answers the adversary's queries. For each query $((a_0, \{\mathbf{a}_j\}_{j\in[n]}), i)$ made by $\mathcal{A}$, the challenger does as follows.

  Define $\mathbf{y}_{i,j} = \mathbf{s}_i \oplus \mathbf{s}_j$ (the binary XOR operation). Given $\{\mathbf{y}_{i,j}\}_{i,j}$, the challenger finds $(a'_0, \mathbf{a}')$ such that $f_{a'_0,\mathbf{a}'}(\mathbf{s}_i) = f_{a_0,\{\mathbf{a}_j\}_{j\in[n]}}(\mathbf{s}_1, \ldots, \mathbf{s}_n)$. This is possible to do without knowing the values of the $\{\mathbf{s}_i\}$, only $\{\mathbf{y}_{i,j}\}$: Consider the element $s_{j,i'}$ (the $i'^{\mathrm{th}}$ element of $\mathbf{s}_j$). We know that $s_{j,i'} = s_{i,i'} \oplus (\mathbf{y}_{i,j})_{i'}$. Therefore we know that if $(\mathbf{y}_{i,j})_{i'} = 0$ then $s_{j,i'} = s_{i,i'}$ and if $(\mathbf{y}_{i,j})_{i'} = 1$ then $s_{j,i'} = 1 - s_{i,i'}$. We can thus replace the $a_{j,i'} \cdot s_{j,i'}$ element in the description of the function $f$ with either $a_{j,i'} \cdot s_{i,i'}$ or $a_{j,i'} \cdot (1 - s_{i,i'})$, depending on the (known) value of $(\mathbf{y}_{i,j})_{i'}$. Doing this for one variable after another, results in an affine function of only $\mathbf{s}_i$. We again stress that we only used $\{\mathbf{y}_{i,j}\}_{i,j}$ for this transformation.

  The challenger in this hybrid answers with $(c_0, \mathbf{c}) = ((-1)^{a'_0} \cdot g^r_{0,i}, (-1)^{\mathbf{a}'} \cdot \mathbf{g}^r_i)$ instead of $(c_0, \mathbf{c}) = ((-1)^{a'_0+\sum_{j\in[\ell]} a'_j\cdot s_{i,j}} \cdot g^r_{0,i}, \mathbf{g}^r_i)$.

  It holds that

  $$\left|\Pr_{H_1}[b' = 1] - \Pr_{H_0}[b' = 1]\right| \le 4nt\ell \cdot \mathrm{QRAdv}[\mathcal{B}_1] + O(nt\ell/N) \ ,$$

since, as in the proof of Theorem 4.7.2, the difference between the hybrids can be viewed as a $t$-round adaptive $(n\ell)$-vector game, considering $(\mathbf{g}_1, \dots, \mathbf{g}_n)$ as the first message, and now the simulated adversary only needs a part of the answer for each query (the one that is respective to the user $i$ for which the query was made).

- **Hybrid $H_2$.** Following the proof outline of Theorem 4.7.2, we change the distributions of $g_{0,i}$ for *all* $i \in [n]$, to $U(\mathbb{QR}_N)$. The challenger still needs to know $\{\mathbf{y}_{i,j}\}_{i,j\in[n]}$ in order to answer the queries, so we need to prove that even fixing $\{\mathbf{y}_{i,j}\}_{i,j\in[n]}$, hybrid $H_2$ is close to $H_1$. Intuitively speaking, this will require us to "extract" $n$ uniform elements of $\mathbb{QR}_N$ out of a single $\mathbf{s}$ (because once one of them is specified, all others are determined by the values of $\mathbf{y}_{i,j}$). Therefore, for security to hold, we have to require that $\ell$ is proportional to $n$.

  We now wish to apply Lemma 4.3.2 to claim that the hybrids are statistically close. To do that, we consider the following family of hash functions (defined for a fixed value of $\{\mathbf{y}_{i,j}\}$)

  $$z_{\mathbf{g}_1,\dots,\mathbf{g}_n}(\mathbf{s}_1) = \left(\prod g_{1,i}^{s_{1,i}}, \prod g_{2,i}^{s_{1,i}\oplus(\mathbf{y}_{1,2})_i}, \dots, \prod g_{n,i}^{s_{1,i}\oplus(\mathbf{y}_{1,n})_i}\right) .$$

  This family is 2-universal (by a similar argument to Lemma 4.3.4, using the fact that the vectors $\mathbf{g}_i$ are independent). The output describes the distribution of $g_{0,i}$ in the case where all $\mathbf{y}_{i,j}$ are known, but $\mathbf{s}_1$ is not. We can now apply Lemma 4.3.2 respective to this family and conclude that

  $$\left|\Pr_{H_2}[b'=1] - \Pr_{H_1}[b'=1]\right| \le \sqrt{\frac{(N')^n}{2^{\ell+2}}} \le \sqrt{\frac{N^n}{2^{\ell+2}}} = \frac{1}{2}\cdot\left(N\cdot 2^{-\ell/n}\right)^{n/2} .$$

- **Hybrid $H_3$.** Note that at this point the public keys are distributed uniformly and independently of $\mathbf{y}_{i,j}$'s. We again change the way the challenger answers queries, along the lines of the proof of Theorem 4.7.2. Instead of answering with $(c_0, \mathbf{c}) = ((-1)^{a'_0} \cdot g_{0,i}^r, (-1)^{\mathbf{a}'} \cdot \mathbf{g}_i^r)$, the challenger now answers with $(c_0, \mathbf{c}) = (g_{0,i}^r, \mathbf{g}_i^r)$. This can be viewed as a $t$-round adaptive $n(\ell+1)$-vector game (similarly to the previous hybrid) and thus

  $$\left|\Pr_{H_3}[b'=1] - \Pr_{H_2}[b'=1]\right| \le 4nt(\ell+1)\cdot \mathrm{QRAdv}[\mathcal{B}_2] + O(nt\ell/N) .$$

- **Hybrid $H_4$.** We revert the distributions of the $g_{0,i}$'s to the original one. As in hybrid $H_2$ we have $|\Pr_{H_4}[b'=1] - \Pr_{H_3}[b'=1]| \le \frac{1}{2}\cdot(N\cdot 2^{-\ell/n})^{n/2}$.

  Hybrid $H_4$ is identical to the $\mathrm{KDM}^{(n)}$ game with $b = 1$ as all queries are answered by encryptions of 0 and the claim follows. □

### 4.7.2.3 Beyond Affine Functions

Two building blocks have been suggested in [BGK11, BHHI10] to obtain KDM-security w.r.t. a larger class of functions. The work of [BHHI10] was later generalized and simplified by [App11]. Our scheme has the properties required to apply both constructions, yielding the following corollaries.

The first corollary is derived using [BGK11, Theorem 1.1]. A set of functions $\mathcal{H} = \{h_1, \dots, h_\ell : h_i : \{0,1\}^\kappa \to \{0,1\}\}$ is *entropy preserving* if the function $f(x) = (h_1(x)\|\cdots\|h_\ell(x))$ is injective (the operator $\|$ represents string concatenation).

**Corollary 4.7.4.** *Consider $\mathcal{E}[\ell]$ and let $\kappa$ be polynomial in the security parameter such that $\kappa \geq \log N + \omega(\log k)$. Then for any entropy preserving set $\mathcal{H} = \{h_1, \ldots, h_\ell : h_i \in \{0,1\}^\kappa \to \{0,1\}\}$ of efficiently computable functions, with polynomial cardinality (in the security parameter), there exists a* KDM$^{(1)}$*-secure scheme under the* QR*-assumption w.r.t. the class of functions*

$$\mathcal{F} = \left\{ f(\mathbf{x}) = a_0 + \sum a_i h_i(\mathbf{x}) : (a_0, \mathbf{a}) \in \mathbb{Z}_2 \times \mathbb{Z}_2^\ell \right\} .$$

The second corollary is derived using [App11, Proposition 4.9].[13]

**Corollary 4.7.5.** *Based on the* QR *assumption, for any polynomial $p$ there exists a* KDM$^{(1)}$*-secure encryption scheme w.r.t. all functions computable by circuits of size $p(k)$ (where $k$ is the security parameter).*

These results can be generalized to any SG assumption. For the generalized statements and proofs, see Section 4.8.4.4.

### 4.7.3 Leakage Resilience

We prove that the scheme $\mathcal{E}[\ell]$ (our QR based scheme) is resilient to a leakage of up of $\lambda = \ell - \log N - \omega(\log k)$ bits. This implies that taking $\ell = \omega(\log N)$, achieves $(1 - o(1))$ leakage rate.

Intuitively, to prove leakage resilience, we consider the case where instead of outputting the challenge ciphertext $((-1)^m \cdot g_0^r, \mathbf{g}^r)$, we output $((-1)^m \cdot (-1)^{\sum \sigma_i s_i} \cdot g_0^r, (-1)^{\boldsymbol{\sigma}} \cdot \mathbf{g}^r)$, for a random vector $\boldsymbol{\sigma} \overset{\$}{\leftarrow} \mathbb{Z}_2^\ell$. The views of the adversary in the two cases are indistinguishable (by an adaptive vector game).[14] Using the leftover hash lemma, so long as $\mathbf{s}$ has sufficient min-entropy, even given $g_0$ and the leakage, then $\sum \sigma_i s_i$ is close to uniform. In other words, the ciphertexts generated by our scheme are computationally indistinguishable from ones that contain a strong extractor (whose seed is the aforementioned $\boldsymbol{\sigma}$), applied to the secret key. This guarantees leakage resilience.[15] The result in the QR case is formally stated and proven below. The general result is stated and proven in Section 4.8.5.

**Theorem 4.7.6.** *Let $\mathcal{A}$ be a $\lambda$-leakage adversary for $\mathcal{E}[\ell]$. Then there exists an adversary $\mathcal{B}$ such that*

$$\text{Leak}_\lambda\text{Adv}[\mathcal{A}] \leq 8\ell \cdot \text{QRAdv}[\mathcal{B}] + \sqrt{N \cdot 2^{\lambda - \ell}} + O(\ell/N) .$$

*Proof.* We prove by a series of hybrids (experiments). Each experiment defines a binary random variable (one can think of a value of 1 as a "success" in the experiment).

- **Hybrid $H_0$.** This hybrid describes the following experiment: a challenger flips a coin $b \overset{\$}{\leftarrow} \{0,1\}$ and plays the $\lambda$-leakage game with $\mathcal{A}$. It returns 1 if and only if $b' = b$, where $b'$ is the value returned by $\mathcal{A}$. By definition

$$\left| \Pr[H_0 = 1] - \frac{1}{2} \right| = \frac{\text{Leak}_\lambda\text{Adv}[\mathcal{A}]}{2} .$$

[13] A more complicated proof, based on [BHHI10, Theorem 4.1], appeared in an earlier version of this work.

[14] Of course the latter ciphertext can only be generated using the secret key, but the indistinguishability holds even when the secret key is known.

[15] In the spirit of [NS09], we can say that our scheme defines a new hash proof system that is universal with high probability over illegal ciphertexts, a property which is sufficient for leakage resilience.

- **Hybrid $H_1$.** We change the encryption algorithm. In this hybrid, the challenger encrypts the message $m_b$ by first computing $\mathbf{c} = \mathbf{g}^r$ and then using $\mathbf{s}$ to produce $c_0 = (-1)^{m_b} \cdot \prod_{i \in [\ell]} c_i^{-s_i}$. The ciphertext distribution does not change and hence $\Pr[H_1 = 1] = \Pr[H_0 = 1]$.

- **Hybrid $H_2$.** Again we change the encryption. This time the challenger samples $\boldsymbol{\sigma} \overset{\$}{\leftarrow} \{0,1\}^\ell$ and uses $\mathbf{c} = (-1)^{\boldsymbol{\sigma}} \cdot \mathbf{g}^r$ instead of $\mathbf{c} = \mathbf{g}^r$. Note that the difference between $H_1$ and $H_2$ is exactly a 1-round adaptive $\ell$-vector game and thus by Lemma 4.6.1, there exists an adversary $\mathcal{B}$ such that
$$|\Pr[H_2 = 1] - \Pr[H_1 = 1]| \le 4\ell \cdot \mathrm{QRAdv}[\mathcal{B}] + O(\ell/N) \ .$$

- **Hybrid $H_3$.** We notice that in $H_2$, the distribution of $c_0$ is
$$c_0 = (-1)^{m_b} \cdot \prod_{i \in [\ell]} (-1)^{s_i \cdot \sigma_i} \cdot \Big( \prod_{i \in [\ell]} g_i^{-s_i} \Big)^r = (-1)^{m_b + \sum s_i \cdot \sigma_i} \cdot \Big( \prod_{i \in [\ell]} g_i^{-s_i} \Big)^r \ .$$
In hybrid $H_3$ we change this distribution. The challenger samples $u \overset{\$}{\leftarrow} \{0,1\}$ and sets
$$c_0 = (-1)^{m_b + u} \cdot \Big( \prod_{i \in [\ell]} g_i^{-s_i} \Big)^r \ .$$
To analyze this hybrid, we recall that $\prod_{i \in [\ell]} g_i^{-s_i} \in \mathbb{QR}_N$ and use Lemma 4.3.4 and Lemma 4.3.3 to conclude that for any value of $\mathbf{g}$, it holds that $(\boldsymbol{\sigma}, \sum s_i \sigma_i, \prod_{i \in [\ell]} g_i^{-s_i}, f(\mathbf{s}))$ is $\frac{1}{2} \cdot \sqrt{N \cdot 2^{\lambda - \ell}}$-close to $(\boldsymbol{\sigma}, u, \prod_{i \in [\ell]} g_i^{-s_i}, f(\mathbf{s}))$. It follows that
$$|\Pr[H_3 = 1] - \Pr[H_2 = 1]| \le \frac{1}{2} \cdot \sqrt{N \cdot 2^{\lambda - \ell}} \ .$$

- **Hybrid $H_4$.** We further change $c_0$ and now set it to be
$$c_0 = (-1)^u \cdot \Big( \prod_{i \in [\ell]} g_i^{-s_i} \Big)^r \ .$$
Since $u$ is uniform, it is distributed identically to $m_b + u$ (note that the arithmetics here are over $\mathbb{Z}_2$) and thus $\Pr[H_4 = 1] = \Pr[H_3 = 1]$. In $H_4$, however, the ciphertext distribution is independent of $b$. Therefore $\Pr[H_4 = 1] = \frac{1}{2}$.

Combining all of the above, the result follows. □

### 4.7.4 Auxiliary-Input Security

As in previous work, we start by proving weak auxiliary-input security in Lemma 4.7.7 below and then derive general auxiliary-input security for sub-exponentially hard functions in Corollary 4.7.8. The complete proofs for the general case appear in Section 4.8.6.

**Lemma 4.7.7.** *Let $\epsilon(\ell)$ and $f$ be such that $\epsilon$ is negligible and $f$ is $\epsilon$-weakly uninvertible function (more precisely, family of functions). Let $\mathcal{A}$ be an $f$-auxiliary input adversary for $\mathcal{E}[\ell]$. Then there exists an adversary $\mathcal{B}$ such that*

$$\mathrm{Aux}_f^{\mathrm{weak}}\mathrm{Adv}[\mathcal{A}] \leq 8\ell \cdot \mathrm{QRAdv}[\mathcal{B}] + O(\ell/N) + \mathrm{negl}(k) \ .$$

We note that the above may seem confusing, since it appears to imply auxiliary-input security, and thus also semantic security, regardless of the value of $\ell$. However, we recall that if $\ell$ is too small, then we may be able to retrieve $\mathbf{s}$ from $pk$ without the presence of any auxiliary input. Therefore, the value of $\ell$ must be large enough in order for $f$ to be weakly uninvertible.

We only provide a proof sketch of Lemma 4.7.7, for a full proof (for the general case) see Lemma 4.8.9 in Section 4.8.6.

*Proof sketch.* The proof is almost identical to that of Theorem 4.7.6. The only difference is that now we argue that $|\Pr[H_3 = 1] - \Pr[H_2 = 1]| = \mathrm{negl}(k)$ by applying Theorem 4.3.5 (the Goldreich-Levin theorem) to the uninvertible function $f'(\mathbf{s}) = (\mathbf{g}, \prod g_i^{-s_i}, f(\mathbf{s}))$. □

An immediate corollary (see [DGK+10, Lemma 4]) enables us to state that $\mathcal{E}[\ell]$ is $(\epsilon/N)$-auxiliary input secure for any negligible $\epsilon$, this is because the only part of the public key that depends on the secret key is $g_0$, whose value can be "guessed" with probability $1/N$. For a formal proof, see Corollary 4.8.10 in Section 4.8.6. Note that in order for $(\mathrm{negl}(k)/N)$-hard to invert functions to even exist, it must be that $\ell \geq \log N + \omega(\log k)$, since any function of $\ell$ input bits is trivially invertible with probability at least $2^{-\ell}$.

We can derive the following corollary (for proof, see Corollary 4.8.11 in Section 4.8.6).

**Corollary 4.7.8.** *Assuming that a subgroup indistinguishability assumption holds, then for any constant $\delta > 0$ there exists a $2^{-\ell^{\delta}}$-auxiliary input secure encryption scheme.*

## 4.8 Constructions and Proofs for General Subgroup Indistinguishability Assumptions

### 4.8.1 Description of the Encryption Scheme

The scheme $\mathcal{E}[\mathbb{G}_U, \ell]$, which is a generalization of $\mathcal{E}[\ell]$ presented in Section 4.5, is defined as follows.

- **Parameters.** The scheme is parameterized by a group $\mathbb{G}_U$, as described in Section 4.4. Namely, a probabilistic algorithm that given the security parameter $1^k$, produces an instance $I_{\mathbb{G}_U}$ of $\mathbb{G}_U$.

  An additional parameter is the value $\ell$, which is polynomial in the security parameter, but its exact value is determined based on the specific application. The message space for the encryption scheme is $\mathcal{M} = \mathbb{G}_M$.

- **Key generation.** First, sample an instance $I_{\mathbb{G}_U}$ of subgroup indistinguishability (which can be used as a joint public parameter for all users, similarly to $N$ in our QR-based scheme). Then, sample $\mathbf{s} \xleftarrow{\$} \{0,1\}^\ell$ and set $sk = \mathbf{s}$.[16] Lastly, sample $\mathbf{g} \xleftarrow{\$} \mathbb{G}_L{}^\ell$ and set $g_0 = (\prod_{i\in[\ell]} g_i^{s_i})^{-1}$. The public key is $pk = (g_0, \mathbf{g})$ (and, implicitly, also $I_{\mathbb{G}_U}$).

- **Encryption.** Given a public key $pk = (g_0, \mathbf{g})$ and a message $m \in \mathbb{G}_M$, sample $r \xleftarrow{\$} [T^2]$ and compute $\mathbf{c} = \mathbf{g}^r$ and $c_0 = m \cdot g_0^r$. Outputs the ciphertext $(c_0, \mathbf{c})$.
- **Decryption.** Given a secret key $sk = \mathbf{s}$ and a ciphertext $(c_0, \mathbf{c})$, compute $m = c_0 \cdot \prod_{i\in[\ell]} c_i^{s_i}$.

As in our QR-based scheme, correctness is by definition, and security will be proven in the following sections. As before, correctness holds regardless of the distribution of the vector $\mathbf{s}$, and in Section 4.8.7 we discuss the use secret keys which are not binary strings and its effect on the various notions of security.

### 4.8.2 The Adaptive Vector Game

The adaptive $\ell$-vector game, for a general SG assumption, is defined as follows (see Section 4.6 for an explicit presentation of the QR case).

- **Initialize.** Let $\mathbb{G}_U$ be as defined in Section 4.4.1. The challenger samples $b \xleftarrow{\$} \{0,1\}$ and also generates an instance $I_{\mathbb{G}_U}$ and a vector $\mathbf{g} \xleftarrow{\$} \mathbb{G}_L{}^\ell$. It sends $I_{\mathbb{G}_U}$ and $\mathbf{g}$ to the adversary.
- **Query.** The adversary adaptively makes queries where each query is a vector $\mathbf{a} \in \mathbb{G}_M{}^\ell$. For each query $\mathbf{a}$, the challenger samples $r \xleftarrow{\$} [T^2]$ and returns $\mathbf{a}^{(1-b)} \cdot \mathbf{g}^r$. Namely, if $b = 0$ it returns $\mathbf{a} \cdot \mathbf{g}^r$ and if $b = 1$ it returns $\mathbf{g}^r$.
- **Finish.** The adversary outputs a guess $b' \in \{0,1\}$.

The advantage of an adversary $\mathcal{A}$ in the game is defined to be

$$\mathrm{AV}_\ell\mathrm{Adv}[\mathcal{A}] = \left|\Pr[b' = 1|b = 0] - \Pr[b' = 1|b = 1]\right| \ .$$

We show that under the SG assumption, no polynomial time adversary can obtain a non-negligible advantage in the game.

**Lemma 4.8.1.** *Let $\mathcal{A}$ be an adversary for the adaptive $\ell$-vector game that makes at most $t$ queries, then there exists an adversary $\mathcal{B}$ for* SG *such that*

$$\mathrm{AV}_\ell\mathrm{Adv}[\mathcal{A}] \le 4t\ell \cdot \mathrm{SGAdv}[\mathcal{B}] + 2t\ell/T \ .$$

*Proof.* A standard hybrid argument implies the existence of $\mathcal{A}_1$, which is an adversary for a 1-round game ($t = 1$ in our notation) such that $\mathrm{AV}_\ell\mathrm{Adv}[\mathcal{A}] \le t \cdot \mathrm{AV}_\ell\mathrm{Adv}[\mathcal{A}_1]$.

We consider the following hybrids (experiments). For each hybrid $H_i$, we let $\Pr[H_i]$ denote the probability that the experiment "succeeds" (an event we define below).

[16] Alternatively, set $sk = h^{\mathbf{s}}$, and extract $\mathbf{s}$ via discrete logarithm during the decoding process (which is easy since $s_i \in \{0,1\}$). This alternative representation will lead to a different representation of the class of functions for which we get KDM security. See Section 4.8.4.1 for more details.

- **Hybrid** $H_0$**.** In this experiment, we flip a coin $b \xleftarrow{\$} \{0,1\}$ and also sample $i \xleftarrow{\$} [\ell]$. We simulate the 1-round game with $\mathcal{A}_1$, where the challenger answers a query $\mathbf{a} \in \mathbb{G}_M{}^\ell$ with $(g_1^r, \ldots, g_{i-1}^r, a_i^b \cdot g_i^r, a_{i+1} \cdot g_{i+1}^r, \ldots, a_\ell \cdot g_\ell^r)$. The experiment succeeds if $b' = b$.

  A standard argument shows that
  $$\frac{\mathrm{AV}_\ell\mathrm{Adv}[\mathcal{A}_1]}{2\ell} = \left|\Pr[H_0] - \frac{1}{2}\right| .$$

- **Hybrid** $H_1$**.** In this hybrid we replace $g_i$ (which is uniform in $\mathbb{G}_L$) with $h \cdot g_i$. Namely, the adversary's query is now answered with
  $$(g_1^r, \ldots, g_{i-1}^r, a_i^b \cdot (h \cdot g_i)^r, a_{i+1} \cdot g_{i+1}^r, \ldots, a_\ell \cdot g_\ell^r) .$$
  In a straightforward manner, there exist $\mathcal{B}', \mathcal{B}$ such that
  $$|\Pr[H_1] - \Pr[H_0]| \le \mathrm{SG}'\mathrm{Adv}[\mathcal{B}'] \le 2 \cdot \mathrm{SGAdv}[\mathcal{B}] .$$

- **Hybrid** $H_2$**.** In this hybrid, the only change is that now $r \xleftarrow{\$} \mathbb{Z}_{M \cdot L}$ rather than $U([T^2])$. By Lemma 4.3.1, it follows that $|\Pr[H_2] - \Pr[H_1]| \le 1/T$.

  We denote $r_1 = (r \bmod M)$ and $r_2 = (r \bmod L)$. By the Chinese Remainder Theorem, it holds that $r_1, r_2$ are uniform in $\mathbb{Z}_M, \mathbb{Z}_L$ respectively and are independent. The answer to the query in this scenario is therefore
  $$(g_1^r, \ldots, g_{i-1}^r, a_i^b \cdot (h \cdot g_i)^r, a_{i+1} \cdot g_{i+1}^r, \ldots, a_\ell \cdot g_\ell^r) = (g_1^{r_2}, \ldots, g_{i-1}^{r_2}, a_i^b \cdot h^{r_1} \cdot g_i^{r_2}, a_{i+1} \cdot g_{i+1}^{r_2}, \ldots, a_\ell \cdot g_\ell^{r_2}) .$$
  However since $h^{r_1}$ is uniform in the cyclic group $\mathbb{G}_M$ then $a_i^b \cdot h^{r_1}$ is also uniform and is independent of $b$. It follows that $\Pr[H_2] = \frac{1}{2}$.

It follows that $\mathrm{AV}_\ell\mathrm{Adv}[\mathcal{A}_1] \le 4\ell \cdot \mathrm{SGAdv}[\mathcal{B}] + 2\ell/T$, and the result follows. □

### 4.8.3 CPA Security

In this section, we state and prove the CPA security of our general scheme $\mathcal{E}[\mathbb{G}_U, \ell]$. We show that for $\ell \ge \log L + \omega(\log k)$, $\mathcal{E}[\mathbb{G}_U, \ell]$ is CPA-secure. The high level idea is the same as in the QR-based scheme as described in Section 4.7.1.

**Theorem 4.8.2.** *Let $\mathcal{A}$ be a* CPA*-adversary for $\mathcal{E}[\mathbb{G}_U, \ell]$, then there exists an adversary $\mathcal{B}$ such that*
$$\mathrm{CPAAdv}[\mathcal{A}] \le 4(\ell + 1) \cdot \mathrm{SGAdv}[\mathcal{B}] + \sqrt{L \cdot 2^{-\ell}} + O(\ell/T) .$$

The theorem implies that taking $\ell = \log L + \omega(\log k)$ is sufficient to obtain CPA-security.

*Proof.* The proof proceeds by a series of hybrids.

- **Hybrid** $H_0$**.** This hybrid is identical to the $\mathrm{KDM}^{(1)}$ game with $b = 0$. By definition $\Pr_{H_0}[b' = 1] = \Pr[b' = 1 | b = 0]$.

- **Hybrid $H_1$.** In this hybrid, we change the distribution of $g_0$, which will now be sampled from $U(\mathbb{G}_L)$. By Lemma 4.3.4, combined with Lemma 4.3.2, $(g_0, \mathbf{g})$ is $\sqrt{\frac{L}{2^{\ell+2}}}$-uniform. Thus
$$\left|\Pr_{H_1}[b' = 1] - \Pr_{H_0}[b' = 1]\right| \leq \sqrt{\frac{L}{2^{\ell+2}}} \ .$$

- **Hybrid $H_2$.** In this hybrid, we change the way the challenger answers queries. Now instead of answering $(c_0, \mathbf{c}) = (m \cdot g_0^r, \mathbf{g}^r)$, the challenger answers $(c_0, \mathbf{c}) = (g_0^r, \mathbf{g}^r)$. The difference between $H_2$ and $H_1$ is now a 1-query adaptive $(\ell+1)$-vector game and thus by Lemma 4.8.1,
$$\left|\Pr_{H_2}[b' = 1] - \Pr_{H_1}[b' = 1]\right| \leq 4(\ell+1) \cdot \mathrm{SGAdv}[\mathcal{B}] + O(\ell/T) \ ,$$
for some $\mathcal{B}$.

- **Hybrid $H_3$.** We now revert the distribution of $g_0$ back to the original $\prod_{i\in[\ell]} g_i^{-s_i}$. Similarly to $H_1$, we have
$$\left|\Pr_{H_3}[b' = 1] - \Pr_{H_2}[b' = 1]\right| \leq \sqrt{\frac{L}{2^{\ell+2}}} \ .$$
However, hybrid $H_4$ is identical to the CPA game with $b = 1$ as all queries are answered by encryptions of 0: $\Pr_{H_3}[b' = 1] = \Pr[b' = 1|b = 1]$. Summing the terms above, the result follows. □

### 4.8.4 KDM **Security**

In this section, we state and prove the KDM-related properties of our general scheme $\mathcal{E}[\mathbb{G}_U, \ell]$. We start by defining the class $\mathcal{F}_{\mathrm{aff}}$ of affine functions over $\mathbb{G}_M$ in Section 4.8.4.1. Then, in Section 4.8.4.2, we show that for $\ell \geq \log L + \omega(\log k)$, $\mathcal{E}[\mathbb{G}_U, \ell]$ is $\mathrm{KDM}^{(1)}_{\mathcal{F}_{\mathrm{aff}}}$-secure. We proceed, in Section 4.8.4.3, to show that for $\ell \geq n \log L + \omega(\log k)$, it holds that $\mathcal{E}[\mathbb{G}_U, \ell]$ is $\mathrm{KDM}^{(n)}_{\mathcal{F}_{\mathrm{aff}}}$-secure. Section 4.8.4.4 explains how to extend the KDM results of the preceding sections beyond affine functions, by using either the techniques of [BGK11] or those of [BHHI10, App11]. This section contains the general versions and full proofs of the statements in Section 4.7.2.

#### 4.8.4.1 The Class of Affine Functions

Recall that $\mathbb{G}_M$ is a cyclic group of order $M$ with generator $h$. The class of affine functions over $\mathbb{G}_M$ is the class of affine functions over $\mathbb{Z}_M$ "in the exponent". If the discrete logarithm problem in $h$ is easy, then the two are computationally equivalent.

Formally, the class $\mathcal{F}_{\mathrm{aff}}$ with input length $\ell$ is the set of functions $\mathcal{F}_{\mathrm{aff}} = \{f_{\boldsymbol{\alpha},\beta} : \mathbb{Z}_M^\ell \to \mathbb{G}_M\}_{\beta\in\mathbb{G}_M, \boldsymbol{\alpha}\in\mathbb{G}_M{}^\ell}$, where
$$f_{\boldsymbol{\alpha},\beta}(\mathbf{x}) = \beta \cdot \prod_{i\in[\ell]} \alpha_i^{x_i} \ .$$
The function $f_{\boldsymbol{\alpha},\beta}$ is represented by $(\boldsymbol{\alpha}, \beta)$.

In Section 4.8.4.3, we consider affine functions for input dimension $n \cdot \ell$, in which case we will "break" $\boldsymbol{\alpha}$ into $n$ parts and denote

$$\mathcal{F}_{\text{aff}} = \{f_{\boldsymbol{\alpha}_1,\ldots,\boldsymbol{\alpha}_n,\beta} : (\mathbb{Z}_M^\ell)^n \to \mathbb{G}_M\}_{\beta \in \mathbb{G}_M, \boldsymbol{\alpha}_i \in \mathbb{G}_M{}^\ell} \ ,$$

where

$$f_{\boldsymbol{\alpha}_1,\ldots,\boldsymbol{\alpha}_n,\beta}(\mathbf{x}_1,\ldots,\mathbf{x}_n) = \beta \cdot \prod_{i\in[n],j\in[\ell]} \alpha_{i,j}^{x_{i,j}} \ .$$

We remark that [BHHO08] used a "dual" definition of this function class. In their formulation, the secret key of the encryption scheme is $h^{\mathbf{s}}$, and the decryption algorithm needs to recover $\mathbf{s}$ as a first step. A description of an affine function using their formulation contains a vector $\mathbf{a} \in \mathbb{Z}_M$ such that $h^{\mathbf{a}}$ corresponds to $\boldsymbol{\alpha}$ in our formulation. Our proofs (as well as theirs) can be presented using either definition, we prefer ours for for aesthetic reasons.[17]

#### 4.8.4.2 KDM$^{(1)}$-Security

We can now prove KDM$^{(1)}$-security for $\mathcal{E}[\mathbb{G}_U, \ell]$. The high level idea is the same as in the QR-based scheme described in Section 4.7.2.1.

**Theorem 4.8.3.** *Let $\mathcal{A}$ be a $\mathrm{KDM}^{(1)}_{\mathcal{F}_{\text{aff}}}$-adversary for $\mathcal{E}[\mathbb{G}_U, \ell]$ that makes at most $t$ queries, then there exists an adversary $\mathcal{B}$ such that*

$$\mathrm{KDM}^{(1)}_{\mathcal{F}_{\text{aff}}}\mathrm{Adv}[\mathcal{A}] \le 4t(2\ell+1) \cdot \mathrm{SGAdv}[\mathcal{B}] + \sqrt{L \cdot 2^{-\ell}} + O(t\ell/T) \ .$$

The theorem implies that taking $\ell = \log L + \omega(\log k)$ is sufficient to obtain KDM$^{(1)}$-security.

*Proof.* The proof proceeds by a series of hybrids.

- **Hybrid $H_0$.** This hybrid is identical to the KDM$^{(1)}$ game with $b = 0$. By definition $\Pr_{H_0}[b' = 1] = \Pr[b' = 1|b = 0]$.

- **Hybrid $H_1$.** In this hybrid, we change the way the challenger answers the adversary's queries. Recall that in hybrid $H_0$, the query $(\boldsymbol{\alpha}, \beta) \in \mathcal{F}_{\text{aff}}$ was answered by $(c_0, \mathbf{c}) = (\beta \cdot \prod_{i\in[\ell]} \alpha_i^{s_i} \cdot g_0^r, \mathbf{g}^r)$. In hybrid $H_1$, it will be answered by $(c_0, \mathbf{c}) = (\beta \cdot g_0^r, \boldsymbol{\alpha} \cdot \mathbf{g}^r)$.

  We prove that

  $$\left|\Pr_{H_1}[b' = 1] - \Pr_{H_0}[b' = 1]\right| \le \mathrm{AV}_\ell\mathrm{Adv}[\mathcal{A}'] \le 4t\ell \cdot \mathrm{SGAdv}[\mathcal{B}_1] + O(t\ell/T) \ ,$$

  for some $\mathcal{A}', \mathcal{B}_1$, even when $\mathbf{s}$ is fixed and known.

  To see this, we notice that in both hybrids $c_0 = \beta \cdot \prod_{i\in[\ell]} (\alpha_i \cdot c_i^{-1})^{s_i}$ and $g_0 = \prod_{i\in[\ell]} g_i^{-s_i}$. Therefore, an adversary $\mathcal{A}'$ for the adaptive $\ell$-vector game can simulate $\mathcal{A}$, sampling $\mathbf{s}$ on its own and using $\mathbf{g}$ to generate $g_0$ and "translate" the challenger answers. Applying Lemma 4.8.1, the result follows.

[17] Special care needs to be taken when the group $\mathbb{G}_M$ is not efficiently recognizable (i.e. one cannot tell between members of the group and non-members). In such case, we require that the description of the function also contains a proof that it indeed belongs to the prescribed class. This is to prevent the adversary from querying illegal functions that "pretend" to be legal.

- **Hybrid $H_2$.** In this hybrid, we change the distribution of $g_0$, which will now be sampled from $U(\mathbb{G}_L)$. By Lemma 4.3.4, combined with Lemma 4.3.2, $(g_0, \mathbf{g})$ is $\sqrt{\frac{L}{2^{\ell+2}}}$-uniform. Thus
$$\left|\Pr_{H_2}[b' = 1] - \Pr_{H_1}[b' = 1]\right| \leq \sqrt{\frac{L}{2^{\ell+2}}} \ .$$

- **Hybrid $H_3$.** In this hybrid, we again change the way the challenger answers queries. Now instead of answering $(c_0, \mathbf{c}) = (\beta \cdot g_0^r, \boldsymbol{\alpha} \cdot \mathbf{g}^r)$, the challenger answers $(c_0, \mathbf{c}) = (g_0^r, \mathbf{g}^r)$. The difference between $H_2$ and $H_3$ is now a $t$-query adaptive $(\ell + 1)$-vector game and thus by Lemma 4.8.1,
$$\left|\Pr_{H_3}[b' = 1] - \Pr_{H_2}[b' = 1]\right| \leq 4t(\ell+1) \cdot \mathrm{SGAdv}[\mathcal{B}_2] + O(t\ell/T) \ ,$$
for some $\mathcal{B}_2$.

- **Hybrid $H_4$.** We now revert the distribution of $g_0$ back to the original $\prod_{i\in[\ell]} g_i^{-s_i}$. Similarly to $H_2$, we have
$$\left|\Pr_{H_4}[b' = 1] - \Pr_{H_3}[b' = 1]\right| \leq \sqrt{\frac{L}{2^{\ell+2}}} \ .$$
However, hybrid $H_4$ is identical to the $\mathrm{KDM}^{(1)}$ game with $b = 1$ as all queries are answered by encryptions of 0: $\Pr_{H_4}[b' = 1] = \Pr[b' = 1|b = 1]$. Summing the terms above, the result follows (where $\mathcal{B}$ is, say, a weighted average between $\mathcal{B}_1$ and $\mathcal{B}_2$). □

### 4.8.4.3 $\mathrm{KDM}^{(n)}$-Security

We go on to prove $\mathrm{KDM}^{(n)}$-security for our scheme. Unlike the schemes of [BHHO08, ACPS09], we do not achieve a single scheme that is secure w.r.t. any polynomial $n$. We do, however, prove that increasing the value of $\ell$ enables supporting more users. The result is stated in the following theorem.

**Theorem 4.8.4.** *Let $\mathcal{A}$ be a $\mathrm{KDM}^{(n)}_{\mathcal{F}_{\mathrm{aff}}}$-adversary for $\mathcal{E}[\mathbb{G}_U, \ell]$ that makes at most $t$ queries, then there exists an adversary $\mathcal{B}$ such that*
$$\mathrm{KDM}^{(n)}_{\mathcal{F}_{\mathrm{aff}}}\mathrm{Adv}[\mathcal{A}] \leq 4nt(2\ell+1) \cdot \mathrm{SGAdv}[\mathcal{B}] + (L \cdot 2^{-\ell/n})^{n/2} + O(nt\ell/T) \ .$$

The theorem implies that taking $\ell = n \log L + \omega(\log k)$ is sufficient for $\mathrm{KDM}^{(n)}$-security.

*Proof.* The proof follows the outline of the proof of Theorem 4.8.3 with slight modifications to the hybrids.

- **Hybrid $H_0$.** This hybrid is identical to the $\mathrm{KDM}^{(n)}$ game with $b = 0$. Let $\{sk_i = \mathbf{s}_i\}_{i\in[n]}$ denote the generated secret keys and $\{pk_i = (g_{0,i}, \mathbf{g}_i)\}_{i\in[n]}$ denote the public keys produced by the challenger. By definition $\Pr_{H_0}[b' = 1] = \Pr[b' = 1|b = 0]$.

- **Hybrid $H_1$.** We change the way the challenger answers the adversary's queries. For each query $((\{\boldsymbol{\alpha}_j\}_{j\in[n]}, \beta), i)$ made by $\mathcal{A}$, the challenger does as follows.

  Define $\mathbf{y}_{i,j} = \mathbf{s}_i \oplus \mathbf{s}_j$ (the binary XOR operation). Given $\{\mathbf{y}_{i,j}\}_{i,j}$, the challenger finds $(\boldsymbol{\alpha}', \beta')$ such that $f_{\boldsymbol{\alpha}',\beta'}(\mathbf{s}_i) = f_{\{\boldsymbol{\alpha}_j\}_{j\in[n]},\beta}(\mathbf{s}_1, \ldots, \mathbf{s}_n)$. This is possible to do without knowing the values of the $\{\mathbf{s}_i\}$, only $\{\mathbf{y}_{i,j}\}$: Consider the element $s_{j,i'}$ (the $i'^{\text{th}}$ element of $\mathbf{s}_j$). We know that $s_{j,i'} = s_{i,i'} \oplus (\mathbf{y}_{i,j})_{i'}$. Therefore we know that if $(\mathbf{y}_{i,j})_{i'} = 0$ then $s_{j,i'} = s_{i,i'}$ and if $(\mathbf{y}_{i,j})_{i'} = 1$ then $s_{j,i'} = 1 - s_{i,i'}$. We can thus replace the $\alpha_{j,i'}^{s_{j,i'}}$ element in the description of the function $f$ with either $\alpha_{j,i'}^{s_{i,i'}}$ or $\alpha_{j,i'}^{1-s_{i,i'}}$, depending on the (known) value of $(\mathbf{y}_{i,j})_{i'}$. Doing this for one variable after another, results in an affine function of only $\mathbf{s}_i$. We again stress that we only use $\{\mathbf{y}_{i,j}\}_{i,j}$ for this transformation.

  The challenger in this hybrid answers with $(c_0, \mathbf{c}) = (\beta' \cdot g_{0,i}^r, \boldsymbol{\alpha}' \cdot \mathbf{g}_i^r)$ instead of $(c_0, \mathbf{c}) = (\beta' \cdot \prod_{j\in[\ell]} {\alpha'}_j^{s_{i,j}} \cdot g_{0,i}^r, \mathbf{g}_i^r)$.

  It holds that
  $$\left|\Pr_{H_1}[b'=1] - \Pr_{H_0}[b'=1]\right| \le 4nt\ell \cdot \mathrm{SGAdv}[\mathcal{B}_1] + O(nt\ell/T) \ ,$$
  since, as in the proof of Theorem 4.8.3, the difference between the hybrids can be viewed as a $t$-round adaptive $(n\ell)$-vector game, considering $(\mathbf{g}_1, \ldots, \mathbf{g}_n)$ as the first message, and now the simulated adversary only needs a part of the answer for each query (the one that is respective to the user $i$ for which the query was made).

- **Hybrid $H_2$.** Following the proof outline of Theorem 4.8.3, we change the distributions of $g_{0,i}$ for *all* $i \in [n]$, to $U(\mathbb{G}_L)$. The challenger still needs to know $\{\mathbf{y}_{i,j}\}_{i,j\in[n]}$ in order to answer the queries so we need to prove that even fixing $\{\mathbf{y}_{i,j}\}_{i,j\in[n]}$, hybrid $H_2$ is close to $H_1$. Intuitively speaking, this will require us to "extract" $n$ uniform elements of $\mathbb{G}_L$ out of a single $\mathbf{s}$ (because once one of them is specified, all others are determined by the values of $\mathbf{y}_{i,j}$). Therefore, for security to hold, we have to require that $\ell$ is proportional to $n$.

  We now wish to apply Lemma 4.3.2 to claim that the hybrids are statistically close. To do that, we consider the following family of hash functions (defined for a fixed value of $\{\mathbf{y}_{i,j}\}$)
  $$z_{\mathbf{g}_1,\ldots,\mathbf{g}_n}(\mathbf{s}_1) = \left(\prod g_{1,i}^{s_{1,i}}, \prod g_{2,i}^{s_{1,i}\oplus(\mathbf{y}_{1,2})_i}, \ldots, \prod g_{n,i}^{s_{1,i}\oplus(\mathbf{y}_{1,n})_i}\right) \ .$$
  This family is 2-universal (by a similar argument to Lemma 4.3.4, using the fact that the vectors $\mathbf{g}_i$ are independent). The output describes the distribution of $g_{0,i}$ in the case where all $\mathbf{y}_{i,j}$ are known, but $\mathbf{s}_1$ is not. We can now apply Lemma 4.3.2 respective to this family and conclude that
  $$\left|\Pr_{H_2}[b'=1] - \Pr_{H_1}[b'=1]\right| \le \sqrt{\frac{L^n}{2^{\ell+2}}} = \frac{1}{2} \cdot (L \cdot 2^{-\ell/n})^{n/2} \ .$$

- **Hybrid $H_3$.** Note that at this point, the public keys are distributed uniformly and independently of $\mathbf{y}_{i,j}$'s. We again change the way the challenger answers queries, along the lines of the proof of Theorem 4.8.3. Instead of answering with $(c_0, \mathbf{c}) = (\beta' \cdot g_{0,i}^r, \boldsymbol{\alpha}' \cdot \mathbf{g}_i^r)$, the challenger now answers with $(c_0, \mathbf{c}) = (g_{0,i}^r, \mathbf{g}_i^r)$. This can be viewed as a $t$-round adaptive $n(\ell+1)$-vector game (similarly to the previous hybrid) and thus

$$\left|\Pr_{H_3}[b' = 1] - \Pr_{H_2}[b' = 1]\right| \leq 4nt(\ell + 1) \cdot \mathrm{SGAdv}[\mathcal{B}_2] + O(nt\ell/T) \ .$$

- **Hybrid** $H_4$**.** We revert the distributions of the $g_{0,i}$'s to the original one. As in hybrid $H_2$, we have $|\Pr_{H_4}[b' = 1] - \Pr_{H_3}[b' = 1]| \leq \frac{1}{2} \cdot (L \cdot 2^{-\ell/n})^{n/2}$.

  Hybrid $H_4$ is identical to the $\mathrm{KDM}^{(n)}$ game with $b = 1$ as all queries are answered by encryptions of 0 and the claim follows. □

#### 4.8.4.4 Beyond Affine Functions

Two approaches have been suggested to obtain KDM-security w.r.t. a larger family of functions. The first is due to [BGK11] and the second is due to [BHHI10, App11]. In this section we show that both extend to the SG assumption.

**Entropy-$\kappa$ Security.** The notion of entropy-$\kappa$ KDM-security was introduced in [BGK11] as a way to extend KDM-security beyond affine functions. In their work [BGK11, Definitions 3.1, 3.2], an encryption scheme is called *projective* if the key-generation can be described as follows: First, the secret key is uniformly sampled from some set $\mathcal{S}$, and then the public key is computed as a (possibly randomized) efficient function of the secret key. A projective scheme is *entropy-$\kappa$* $\mathrm{KDM}^{(n)}$*-secure* if for any distribution $\mathcal{D}$ with $\mathbf{H}_\infty(\mathcal{D}) \geq \kappa$ supported inside $\mathcal{S}$, the scheme obtained by sampling the secret key from $\mathcal{D}$ rather than from $\mathcal{S}$, is $\mathrm{KDM}^{(n)}$-secure.

We show that $\mathcal{E}[\mathbb{G}_U, \ell]$ is entropy-$\kappa$ $\mathrm{KDM}^{(1)}$-secure for $\kappa \geq \log L + \omega(\log k)$ (note that the scheme is clearly projective).

**Lemma 4.8.5.** *Let $\mathcal{D}$ be a distribution on $\{0,1\}^\ell$ with $\mathbf{H}_\infty(\mathcal{D}) \geq \kappa$, let $\mathcal{E}_\mathcal{D}[\mathbb{G}_U, \ell]$ denote the encryption scheme that samples the secret key from $\mathcal{D}$ rather than $U(\{0,1\}^\ell)$.*

*Let $\mathcal{A}$ be a* $\mathrm{KDM}^{(1)}$*-adversary for $\mathcal{E}_\mathcal{D}[\mathbb{G}_U, \ell]$ that makes at most $t$ queries, then there exists an adversary $\mathcal{B}$ such that*

$$\mathrm{KDM}^{(1)}\mathrm{Adv}[\mathcal{A}] \leq t(2\ell + 1) \cdot \mathrm{SGAdv}[\mathcal{B}] + \sqrt{L \cdot 2^{-\kappa}} + O(t\ell/T) \ .$$

*Proof.* The proof is identical to that of Theorem 4.8.3. The only difference is in the transitions from hybrid $H_1$ to $H_2$ and from hybrid $H_3$ to $H_4$ (which are the same transition in reverse order). The difference here is that now when we invoke Lemma 4.3.2, we have $|X| = 2^\kappa$ rather than $2^\ell$. The result immediately follows. □

The following corollary combines the above lemma with [BGK11, Theorem 1.1]. A set of functions $\mathcal{H} = \{h_1, \ldots, h_\ell : h_i : \{0,1\}^\kappa \to \{0,1\}\}$ is *entropy preserving* if the function $f(x) = (h_1(x)\| \cdots \|h_\ell(x))$ is injective (the operator $\|$ represents string concatenation).

**Corollary 4.8.6.** *Consider $\mathcal{E}[\mathbb{G}_U, \ell]$ and let $\kappa$ be polynomial in the security parameter such that $\kappa \geq \log L + \omega(\log k)$. Then for any entropy preserving set $\mathcal{H} = \{h_1, \ldots, h_\ell : h_i : \{0,1\}^\kappa \to \{0,1\}\}$*

*of efficiently computable functions with polynomial cardinality (in the security parameter), there exists a* KDM$^{(1)}$*-secure scheme under the* SG*-assumption w.r.t. the class of functions*

$$\mathcal{F} = \left\{ f(\mathbf{x}) = \beta \cdot \prod_{i \in [\ell]} \alpha_i^{h_i(\mathbf{x})} : (\boldsymbol{\alpha}, \beta) \in {\mathbb{G}_M}^{\ell} \times \mathbb{G}_M \right\} .$$

**Completeness Theorem for** KDM **Security.** The completeness theorem in [App11, Proposition 4.9] in conjunction with Theorem 4.8.3 (or Theorem 4.8.4 if KDM$^{(n)}$ is wanted), immediately implies the following corollary. (A more complicated proof, that appeared in an earlier version of this work, uses [BHHI10, Theorem 4.1].)

**Corollary 4.8.7.** *Based on the* SG *assumption, for any polynomial $p$ there exists a* KDM$^{(1)}$*-secure encryption scheme w.r.t. all functions computable by circuits of size $p(k)$ (where $k$ is the security parameter).*

### 4.8.5 Leakage Resilience

We prove that the scheme $\mathcal{E}[\mathbb{G}_U, \ell]$ is resilient to a leakage of up of $\lambda = \ell - \log(ML) - \omega(\log k)$ bits. The result is formally stated below, for overview see Section 4.7.3.

**Theorem 4.8.8.** *Let $\mathcal{A}$ be a $\lambda$-leakage adversary for $\mathcal{E}[\mathbb{G}_U, \ell]$. Then there exists an adversary $\mathcal{B}$ such that*

$$\mathrm{Leak}_\lambda\mathrm{Adv}[\mathcal{A}] \leq 8\ell \cdot \mathrm{SGAdv}[\mathcal{B}] + \sqrt{ML \cdot 2^{\lambda - \ell}} + O(\ell/T) .$$

*Proof.* We prove by a series of hybrids (experiments). Each experiment represents a process with a single binary value (one can think of 1 as a "success" in the experiment).

- **Hybrid $H_0$.** This hybrid describes the following experiment: a challenger flips a coin $b \xleftarrow{\$} \{0,1\}$ and plays the $\lambda$-leakage game with $\mathcal{A}$. It returns 1 if and only if $b' = b$, where $b'$ is the value returned by $\mathcal{A}$. By definition

$$\left| \Pr[H_0 = 1] - \frac{1}{2} \right| = \frac{\mathrm{Leak}_\lambda\mathrm{Adv}[\mathcal{A}]}{2} .$$

- **Hybrid $H_1$.** We change the encryption algorithm. In this hybrid, the challenger encrypts a message $m$ by first computing $\mathbf{c} = \mathbf{g}^r$, and then using $\mathbf{s}$ to produce $c_0 = m_b \cdot \prod_{i \in [\ell]} c_i^{-s_i}$. The ciphertext distribution does not change and hence $\Pr[H_1 = 1] = \Pr[H_0 = 1]$.

- **Hybrid $H_2$.** Again we change the encryption. This time the challenger samples $\boldsymbol{\sigma} \xleftarrow{\$} {\mathbb{G}_M}^{\ell}$ and uses $\mathbf{c} = \boldsymbol{\sigma} \cdot \mathbf{g}^r$ instead of $\mathbf{c} = \mathbf{g}^r$. Note that the difference between $H_1$ and $H_2$ is exactly a 1-round adaptive $\ell$-vector game, and thus by Lemma 4.8.1, there exists an adversary $\mathcal{B}$ such that

$$|\Pr[H_2 = 1] - \Pr[H_1 = 1]| \leq 4\ell \cdot \mathrm{SGAdv}[\mathcal{B}] + O(\ell/T) .$$

- **Hybrid $H_3$.** We notice that in $H_2$, the distribution of $c_0$ is
$$c_0 = m_b \cdot \prod_{i \in [\ell]} \sigma_i^{-s_i} \cdot \Big( \prod_{i \in [\ell]} g_i^{-s_i} \Big)^r \ .$$
In hybrid $H_3$ we change this distribution. The challenger samples $u \overset{\$}{\leftarrow} \mathbb{G}_M$ and sets
$$c_0 = m_b \cdot u^{-1} \cdot \Big( \prod_{i \in [\ell]} g_i^{-s_i} \Big)^r \ .$$
To analyze this hybrid, we recall that $\prod_{i \in [\ell]} g_i^{-s_i} \in \mathbb{G}_L$, and use Lemma 4.3.4 and Lemma 4.3.3 to conclude that for any value of $\mathbf{g}$ it holds that
$$\Big(\boldsymbol{\sigma}, \prod_{i \in [\ell]} \sigma_i^{s_i}, \prod_{i \in [\ell]} g_i^{-s_i}, f(\mathbf{s})\Big) \qquad \text{and} \qquad \Big(\boldsymbol{\sigma}, u, \prod_{i \in [\ell]} g_i^{-s_i}, f(\mathbf{s})\Big)$$
are $\frac{1}{2} \cdot \sqrt{LM \cdot 2^{\lambda - \ell}}$-close.
It follows that
$$|\Pr[H_3 = 1] - \Pr[H_2 = 1]| \le \frac{1}{2} \cdot \sqrt{LM \cdot 2^{\lambda - \ell}} \ .$$
- **Hybrid $H_4$.** We further change $c_0$ and now set it to be
$$c_0 = u \cdot \Big( \prod_{i \in [\ell]} g_i^{-s_i} \Big)^r \ .$$
Since $u$ is uniform, it is distributed identically to $m_b \cdot u^{-1}$ and thus $\Pr[H_4 = 1] = \Pr[H_3 = 1]$. In $H_4$, however, the ciphertext distribution is independent of $b$. Therefore $\Pr[H_4 = 1] = \frac{1}{2}$.

Combining all of the above, the result follows. □

### 4.8.6 Auxiliary-Input Security

We present the general result, showing the auxiliary-input security of $\mathcal{E}[\mathbb{G}_U, \ell]$, following the same line of proof as in Section 4.7.4. The main difference is that we use a *generalized* Goldreich-Levin theorem (Theorem 4.3.6) instead of the classic Theorem 4.3.5. Since the parameters of the generalized version are slightly worse, this leads to slightly worse parameters in the auxiliary input security.

The following lemma establishes weak auxiliary-input security.

**Lemma 4.8.9.** *Let $\epsilon(\ell) = M^{-\omega(\log \ell)}$ and let $f : \{0,1\}^\ell \to \{0,1\}^*$ be any $\epsilon$-weakly uninvertible function (more precisely, family of functions). Let $\mathcal{A}$ be an $f$-auxiliary input adversary for $\mathcal{E}[\mathbb{G}_U, \ell]$. Then there exists an adversary $\mathcal{B}$ such that*
$$\mathrm{Aux}_f^{\mathrm{weak}}\mathrm{Adv}[\mathcal{A}] \le 8\ell \cdot \mathrm{SGAdv}[\mathcal{B}] + O(\ell/T) + \mathrm{negl}(k) \ .$$

We remark that when $M$ is polynomial, as in the case of the QR-based scheme described in Section 4.7.4, it is sufficient to require that $\epsilon$ is negligible, and thus *any* function that is weakly uninvertible in polynomial time meets the requirement.

The proof (below) follows the same steps as the proof of Theorem 4.8.8. The only change is that we use the (generalized) Goldreich-Levin theorem instead of the leftover hash lemma.

*Proof.* We use the exact same hybrids as in the proof of Theorem 4.8.8 (with the exception that $f$ is now an $\epsilon$-weakly uninvertible function rather than a length bounded one). The exact same arguments imply that

$$\begin{aligned}
\left|\Pr[H_0 = 1] - \frac{1}{2}\right| &= \frac{\mathrm{Aux}_f\mathrm{Adv}[\mathcal{A}]}{2} \\
\Pr[H_1 = 1] &= \Pr[H_0 = 1] \\
|\Pr[H_2 = 1] - \Pr[H_1 = 1]| &\le 4\ell \cdot \mathrm{SGAdv}[\mathcal{B}] + O(\ell/T) \\
\Pr[H_4 = 1] &= \Pr[H_3 = 1] \\
\Pr[H_4 = 1] &= \frac{1}{2} \ .
\end{aligned}$$

Therefore, it remains to prove a bound on $|\Pr[H_3 = 1] - \Pr[H_2 = 1]|$. Assume towards contradiction that there exists a polynomial $t(k)$ such that $|\Pr[H_3 = 1] - \Pr[H_2 = 1]| \ge 1/t(k)$.

Consider the function $f' : \{0,1\}^\ell \to \{0,1\}^*$ defined as follows: $f'(\mathbf{s})$ uses $sk = \mathbf{s}$ as a secret key for $\mathcal{E}[\mathbb{G}_U, \ell]$ and computes a corresponding $pk$. It then computes $y \leftarrow f(sk, pk)$ and outputs $(y, pk)$. Since $f$ is $\epsilon$-weak uninvertible, it follows that for any adversary $\mathcal{C}$, $\Pr[\mathcal{C}(f'(\mathbf{s})) = \mathbf{s}] < \epsilon$, where the probability is over $\mathbf{s}$ and over the coin-tosses of $f'$ and $\mathcal{C}$.

We notice that the hybrids $H_2$ and $H_3$ can be represented as an efficient (randomized) function of the distributions $(f'(\mathbf{s}), \boldsymbol{\tau}, \langle \boldsymbol{\tau}, \mathbf{s}\rangle)$ and $(f'(\mathbf{s}), \boldsymbol{\tau}, v)$ respectively, where $(\boldsymbol{\tau}, v)$ are the discrete logarithms of $(\boldsymbol{\sigma}, u)$, respectively. Namely $\sigma_i = h^{\tau_i}$ and $u = h^v$. Note that $\{\tau_i\}, v$ are uniform in $\mathbb{Z}_M$. Our assumption implies, therefore, that these distributions are distinguishable with advantage $1/t(k)$. In this case, it follows from Theorem 4.3.6 that there exists a $\mathcal{C}$ whose running time is at most $\mathrm{poly}(\ell, t(k)) = \mathrm{poly}(k)$, such that $\Pr[\mathcal{C}(f'(\mathbf{s})) = \mathbf{s}] \ge t(k) \cdot M^{-(1+\log(8\ell t^2(k)))}/8 = M^{-O(\log k)} > \epsilon$. We reached a contradiction and the claim, therefore, follows. □

An immediate corollary (see also [DGK+10, Lemma 4]) enables us to state that $\mathcal{E}[\mathbb{G}_U, \ell]$ is $(M^{-\omega(\log \ell)}/L)$-auxiliary input secure. Note that in order for such functions to exist, it must be that $\ell \ge \log L + (\log M) \cdot \omega(\log k)$, since any function on $\{0,1\}^\ell$ is trivially invertible with probability at least $2^{-\ell}$.

**Corollary 4.8.10.** *Let $\epsilon(\ell) = \frac{M^{-\omega(\log \ell)}}{L}$ and let $f$ be any $\epsilon$-uninvertible function (more precisely, family of functions). Let $\mathcal{A}$ be an $f$-auxiliary input adversary for $\mathcal{E}[\mathbb{G}_U, \ell]$. Then there exists an adversary $\mathcal{B}$ such that*

$$\mathrm{Aux}_f\mathrm{Adv}[\mathcal{A}] \le 8\ell \cdot \mathrm{SGAdv}[\mathcal{B}] + O(\ell/T) + \mathrm{negl}(k) \ .$$

*Proof.* Recall that the public key of $\mathcal{E}[\mathbb{G}_U, \ell]$ is $(g_0 = \prod g_i^{-s_i}, \mathbf{g})$. Since $\mathbf{g}$ does not depend on the secret key, it can be treated as a public parameter of the scheme and not as a part of the public key. Since $g_0 \in \mathbb{G}_L$, then any $\epsilon$-uninvertible function is also $(\epsilon L)$-weakly uninvertible. The result follows. □

We can derive the following corollary, which is a restatement of Corollary 4.7.8.

**Corollary 4.8.11.** *Assuming that a subgroup indistinguishability assumption holds, then for any constant $\delta > 0$ there exists a $2^{-\ell^{\delta}}$-auxiliary input secure encryption scheme.*

*Proof.* This follows immediately from Corollary 4.8.10 by using $\mathcal{E}[\mathbb{G}_U, (\log T \cdot \omega(\log k))^{1/\delta}]$. (Recall that $T$ is our upper bound on $(ML)$.) □

## 4.8.7 Using Our Scheme with Non-Binary Secret Keys

In all constructions so far in this chapter, the secret keys were always a uniform binary vector. Indeed, this distribution is simple and makes the proofs go through more easily. However, in some cases, using secret keys that come from a larger domain can be useful. Specifically, the parameter $\ell$ that governs the efficiency of our scheme is chosen in many cases so that the secret key has sufficient entropy. Using distributions over larger domains, we can get the same entropy using smaller values of $\ell$, and hence improve the efficiency of our scheme. The correctness of our schemes is trivially preserved even for non-binary secret keys, as we mentioned in the relevant sections. In this section, we show under which conditions security is preserved.

We consider the generic scheme from Section 4.8.1 with one change: The secret key vector $\mathbf{s}$ will now be sampled from a distribution $\mathcal{S}$ over $\mathbb{Z}^{\ell}$. All other parameters of the scheme remain unchanged. We denote this generalized scheme by $\mathcal{E}[\mathbb{G}_U, \ell, \mathcal{S}]$. Again, correctness does not depend on the secret key distribution at all, and will hold for any $\mathcal{S}$.

Let us now go over the various security notions and see how they are affected by this change. We note that no change to the adaptive vector game is required.

**CPA Security.** Looking at the proof of Theorem 4.8.2, we see that the secret key distribution only comes into play in hybrids $H_1$, $H_3$, where we change the distribution of $g_0$ from $\prod_i g_i^{-s_i}$ to uniform and vice versa. We used the fact that if $\mathbf{s}$ is binary, then indistinguishability follows from the leftover hash lemma. Let us distill the exact property we need for the proof to go through.

**Definition 4.8.1** (hashing groups)**.** *Let $\mathbb{G}$ be a multiplicative commutative group, let $\ell \in \mathbb{N}$ and let $\mathcal{S}$ be a distribution supported on $\mathbb{Z}^{\ell}$. We say that $\mathbb{G}$ $\epsilon$-hashes $\mathcal{S}$, for $\epsilon \geq 0$, if letting $g_1, \ldots, g_\ell \xleftarrow{\$} \mathbb{G}$ and $\mathbf{s} \xleftarrow{\$} \mathcal{S}$, it holds that the distribution $\left(g_1, \ldots, g_\ell, \prod_{i\in[\ell]} g_i^{s_i}\right)$ is $\epsilon$-uniform in $\mathbb{G}^{\ell+1}$.*

If $\mathcal{S}$ is the uniform distribution over $\{0,1\}^{\ell}$, as we had before, then by Lemma 2.2.1 it holds that $\mathbb{G}_L$ $\epsilon$-hashes $\mathcal{S}$, for $\epsilon = \sqrt{\frac{L}{2^{\ell+2}}}$.

Indeed, all we need in order to obtain CPA security with secret key distribution $\mathcal{S}$, is a guarantee that $\mathbb{G}_L$ hashes $\mathcal{S}$. This is formalized in the following theorem

**Theorem 4.8.12.** *Consider the scheme $\mathcal{E}[\mathbb{G}_U, \ell, \mathcal{S}]$ such that $\mathbb{G}_L$ $\epsilon$-hashes $\mathcal{S}$, then letting $\mathcal{A}$ be a* CPA-*adversary for $\mathcal{E}[\mathbb{G}_U, \ell, \mathcal{S}]$, there exists an adversary $\mathcal{B}$ such that*

$$\mathrm{CPAAdv}[\mathcal{A}] \leq 4(\ell+1) \cdot \mathrm{SGAdv}[\mathcal{B}] + 2\epsilon + O(\ell/T) \ .$$

And CPA security follows so long as $\epsilon$ is negligible.

One way to prove the hashing property is using the leftover hash lemma: if $\prod_i g_i^{s_i}$ is (almost) 2-universal over the support of $\mathcal{S}$, and is sufficiently shrinking, then the hashing property follows. For example, if $\mathbb{G}_L$ is of prime order (or if its order is a product of large primes), then (almost) 2-universality will hold for *any* $\mathcal{S}$

In fact, the hashing property can be proven using weaker tools than the leftover hash lemma, since we only require hashing w.r.t. the specific distribution $\mathcal{S}$, and not *all* high-entropy distributions. Such instantiation is used in [MTY11] to show a scheme that is secure under the DCR assumption with $\ell = 1$ and a proper key distribution $\mathcal{S}$.

KDM-**Security.** From the proof of Theorem 4.8.3, it is apparent that the requirement of $\mathcal{S}$ in this case is identical to the case of CPA security. Hybrids $H_2$, $H_4$ in this proof are identical to hybrids $H_1$, $H_3$ of Theorem 4.8.2. A similar theorem, therefore, follows.

**Theorem 4.8.13.** *Consider the scheme $\mathcal{E}[\mathbb{G}_U, \ell, \mathcal{S}]$ such that $\mathbb{G}_L$ $\epsilon$-hashes $\mathcal{S}$, then letting $\mathcal{A}$ be a $\mathrm{KDM}^{(1)}_{\mathcal{F}_{\mathrm{aff}}}$-adversary for $\mathcal{E}[\mathbb{G}_U, \ell, \mathcal{S}]$ that makes at most $t$ queries, there exists an adversary $\mathcal{B}$ such that*

$$\mathrm{KDM}^{(1)}_{\mathcal{F}_{\mathrm{aff}}}\mathrm{Adv}[\mathcal{A}] \leq 4t(2\ell+1) \cdot \mathrm{SGAdv}[\mathcal{B}] + 2\epsilon + O(t\ell/T) \ .$$

And KDM security follows so long as $\epsilon$ is negligible.

An extension of the above for the case of $\mathrm{KDM}^{(n)}$ is possible, in the same spirit, but is more complicated.

As a final remark, we warn that while KDM security is well defined for any key distribution over $\mathbb{Z}^\ell$, the class of affine functions might not be very meaningful on some distributions. For example, using our QR-based scheme with keys in $\mathbb{Z}^\ell$, we get KDM security w.r.t. the class of affine functions over the least significant bits of the elements of $\mathbf{s}$.

**Leakage Resilience.** Let us carefully examine now the proof of Theorem 4.8.8. Again we see that the role of the secret key distribution is confined to hybrid $H_3$. Now one needs to prove that

$$\big(\mathbf{g}, \boldsymbol{\sigma}, \prod_i \sigma_i^{s_i}, \prod g_i^{-s_i}, f(\mathbf{s})\big) \qquad \text{and} \qquad \big(\mathbf{g}, \boldsymbol{\sigma}, h_0, g_0, f(\mathbf{s})\big)$$

are statistically close, where $h_0$ is uniform in $\mathbb{G}_M$, and $g_0$ is uniform in $\mathbb{G}_L$. In other words, we need the property that the group $\mathbb{G}_U$ (not $\mathbb{G}_L$ as before) hashes $\mathcal{S}$, even given the leakage value:

**Definition 4.8.2** (hashing with leakage)**.** *Let $\mathbb{G}$ be a multiplicative commutative group, let $\ell \in \mathbb{N}$ and let $\mathcal{S}$ be a distribution supported on $\mathbb{Z}^\ell$. Let $\epsilon \geq 0$, $\lambda \geq 0$. We say that $\mathbb{G}$ $\epsilon$-hashes $\mathcal{S}$ with $\lambda$-leakage if letting $g_1, \ldots, g_\ell, g_0 \xleftarrow{\$} \mathbb{G}$ and $\mathbf{s} \xleftarrow{\$} \mathcal{S}$ and letting $f : \mathbb{Z}^\ell \to \{0,1\}^\lambda$ be any function, it holds that the distributions*

$$\big(g_1, \ldots, g_\ell, \prod_{i\in[\ell]} g_i^{s_i}, f(\mathbf{s})\big) \qquad \textit{and} \qquad \big(g_1, \ldots, g_\ell, g_0, f(\mathbf{s})\big) \ ,$$

*are $\epsilon$-close.*

We again note that if $\mathcal{S}$ is the uniform distribution on $\{0,1\}^\ell$, then Lemma 4.3.3 implies that $\mathbb{G}_U$ $\epsilon$-hashes $\mathcal{S}$ with $\lambda$ leakage, for $\epsilon = \left(\frac{1}{2}\sqrt{\frac{M \cdot L}{2^{\ell-\lambda}}}\right)$.

Plugging the above into the proof of Theorem 4.8.8 will imply the following.

**Theorem 4.8.14.** *Consider the scheme $\mathcal{E}[\mathbb{G}_U, \ell, \mathcal{S}]$ such that $\mathbb{G}_U$ $\epsilon$-hashes $\mathcal{S}$ with $\lambda$-leakage, then letting $\mathcal{A}$ be a $\lambda$-leakage adversary for $\mathcal{E}[\mathbb{G}_U, \ell, \mathcal{S}]$, there exists an adversary $\mathcal{B}$ such that*

$$\mathrm{Leak}_\lambda\mathrm{Adv}[\mathcal{A}] \leq 8\ell \cdot \mathrm{SGAdv}[\mathcal{B}] + 2\epsilon + O(\ell/T) \ .$$

As in the previous case, if one can prove the 2-universality of both $\prod_i \sigma_i^{s_i}$ and $\prod_i g_i^{s_i}$, then Definition 4.8.2 will follow. Unlike the previous case, however, we now require that hashing works w.r.t. a family of distributions (corresponding to all possible leakage functions) and we do not know of a simpler way to achieve this property.

**Auxiliary-Input Security.** We can similarly isolate the property that is required in the proof of Lemma 4.8.9, and obtain a more general statement for any $\mathcal{S}$. This property is a computational analogue of Definition 4.8.2. Due to the added complication, we choose not to go into detail on this property.

# Chapter 5

# Discussion and Open Problems

We presented a number of theoretical solutions aimed at facing the challenges of the new cloud computation era. We showed how to construct better fully homomorphic encryption schemes, how to protect against continual memory leakage and how to achieve circular security. Of course, this work is not the end of the story, and many questions still await an answer.

**Efficiency.** Recalling that our schemes need to run in real-time on very weak platforms, the issue of efficiency becomes crucial to whether they can be used in practice.

For fully homomorphic encryption, previous candidates had fairly reasonable encryption and decryption running times, but key generation and homomorphic evaluation were far from practical. In our construction, the key generation becomes more efficient, but homomorphic evaluation remains very exhausting: while encryption and decryption are performed with the smaller parameters $k, p$, homomorphic evaluation is performed with $n, q$. In addition, the use of bootstrapping significantly slows down performance. While this is somewhat improved in [BGV11], we are still orders of magnitude far from practical, even considering the fact that homomorphic evaluation is potentially performed by a very powerful server.

Improving the efficiency of the homomorphic evaluation (perhaps by replacing bootstrapping with a more efficient procedure) is thus a very intriguing open problem in the field.

More efficient constructions for continual leakage resilient and circular secure encryption are also a goal for additional research. Specifically since all known leakage resilient schemes (including ours), achieve high leakage rate at the cost of "blowing up" the parameters and decreasing efficiency.

**Achieving all properties simultaneously.** In this thesis, we presented three schemes, each having some desirable property. The ultimate goal, however, is one scheme that simultaneously has all properties. Specifically, finding a fully homomorphic encryption scheme that is also leakage resilient (even in a non-continual sense) is an interesting open problem. We remark that our third contribution is both circular secure and bounded memory leakage resilient (this also holds for previous works based on different assumptions).

**Diversity in assumptions.** Diversifying the assumptions for circular security was a main motivations for Chapter 4, as well as a number of other works. However, for fully homomorphic encryption and for continual memory leakage, we do not have this diversity.

For fully homomorphic encryption, our contribution improved the state of affairs by showing that the primitive can be based on the hardness of short-vector problems in arbitrary lattices. However, there is no known constructions based on assumptions such as discrete logarithm or factoring. This calls for additional research.

A similar situation exists for continual memory leakage, where all known constructions are based on variants of the linear assumption in bilinear groups. Finding constructions under new assumptions is called for in this case as well.

**Addressing additional challenges of the cloud era.** We show in this thesis how to achieve public-key encryption with desirable properties. However, many of the challenges in this new era are not in the domain of encryption. For example, how to verify that the cloud indeed perform the prescribed operation (this is known as "the delegation problem"), how to maintain data integrity and how to properly authenticate in this setting. These questions are raising a lot of interest in recent years and a lot of progress has been made. Still many problems are still open.

**Fully homomorphic encryption without circular security assumption.** We explained that all known candidate fully homomorphic encryption schemes require explicitly making a circular security assumption in order to keep the size of the evaluation key independent of the depth of the evaluated circuit. Removing this undesired feature is a central open problem in the research of fully homomorphic encryption.

**Validity of the continual memory leakage model.** The continual memory leakage model is another step in the effort to theoretically model generic side channel attacks. However, this model is also not free of assumptions: It assumes that perfect erasure is possible and that the bit length of the leakage at every time period can be bounded. Whether these assumptions are realistic and whether there exist schemes that are secure in a more general model is a direction for further research.

**Circular secure encryption for any function.** We want our schemes to be secure with regards to encrypting arbitrary functions of the secret key. However, based on a standard assumption, we do not know of any scheme that has this property. Coming up with such scheme is an interesting open problem. We remark that in the works of [BHHI10, App11] it is shown that such fully circular secure scheme is equivalent (under some definition) to circular-secure fully homomorphic encryption.

**Is circular security universal?** In the current state of knowledge, we cannot rule out the possibility that any semantically secure scheme that encrypts messages in a bit-by-bit manner is also circular secure. Finding a counter-example or proving this claim is a fundamental open problem in the field.

# Bibliography

[ABB10] S. Agrawal, D. Boneh, and X. Boyen. Efficient lattice (h)ibe in the standard model. In *EUROCRYPT*, pages 553–572, 2010.

[ABHS05] P. Adão, G. Bana, J. Herzog, and A. Scedrov. Soundness of formal encryption in the presence of key-cycles. In S. D. C. di Vimercati, P. F. Syverson, and D. Gollmann, editors, *ESORICS*, volume 3679 of *Lecture Notes in Computer Science*, pages 374–396. Springer, 2005.

[ACPS09] B. Applebaum, D. Cash, C. Peikert, and A. Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In S. Halevi, editor, *CRYPTO*, volume 5677 of *Lecture Notes in Computer Science*, pages 595–618. Springer, 2009.

[ADN$^+$10] J. Alwen, Y. Dodis, M. Naor, G. Segev, S. Walfish, and D. Wichs. Public-key encryption in the bounded-retrieval model. In H. Gilbert, editor, *EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 113–134. Springer, 2010.

[ADW09] J. Alwen, Y. Dodis, and D. Wichs. Leakage-resilient public-key cryptography in the bounded-retrieval model. In S. Halevi, editor, *CRYPTO*, volume 5677 of *Lecture Notes in Computer Science*, pages 36–54. Springer, 2009.

[AGH10] A. Akavia, S. Goldwasser, and C. Hazay. Distributed public key schemes secure against continual leakage. In submission, 2010.

[AGV09] A. Akavia, S. Goldwasser, and V. Vaikuntanathan. Simultaneous hardcore bits and cryptography against memory attacks. In O. Reingold, editor, *TCC*, volume 5444 of *Lecture Notes in Computer Science*, pages 474–495. Springer, 2009.

[Ajt98] M. Ajtai. The shortest vector problem in $_2$ is $p$-hard for randomized reductions (extended abstract). In *STOC*, pages 10–19, 1998.

[AKS01] M. Ajtai, R. Kumar, and D. Sivakumar. A sieve algorithm for the shortest lattice vector problem. In *STOC*, pages 601–610, 2001.

[And97] R. Anderson. Two remarks on public-key cryptology, 1997. Invited lecture, ACM CCCS 1997. Available at `http://www.cl.cam.ac.uk/ftp/users/rja14/forwardsecure.pdf`.

[App11] B. Applebaum. Key-dependent message security: Generic amplification and completeness. In K. G. Paterson, editor, *EUROCRYPT*, volume 6632 of *Lecture Notes in Computer Science*, pages 527–546. Springer, 2011. Specific pointers refer to full version at `http://eprint.iacr.org/2010/513`.

[BBN$^+$09] M. Bellare, Z. Brakerski, M. Naor, T. Ristenpart, G. Segev, H. Shacham, and S. Yilek. Hedged public-key encryption: How to protect against bad randomness. In M. Matsui, editor, *ASIACRYPT*, volume 5912 of *Lecture Notes in Computer Science*, pages 232–249. Springer, 2009.

[BBS04] D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In M. K. Franklin, editor, *CRYPTO*, volume 3152 of *Lecture Notes in Computer Science*, pages 41–55. Springer, 2004.

[BFO08] A. Boldyreva, S. Fehr, and A. O'Neill. On notions of security for deterministic encryption, and efficient constructions without random oracles. In D. Wagner, editor, *CRYPTO*, volume 5157 of *Lecture Notes in Computer Science*, pages 335–359. Springer, 2008.

[BG09] Z. Brakerski and O. Goldreich. From absolute distinguishability to positive distinguishability. *Electronic Colloquium on Computational Complexity (ECCC)*, 2009. Also in Studies in Complexity and Cryptography, 2011.

[BG10] Z. Brakerski and S. Goldwasser. Circular and leakage resilient public-key encryption under subgroup indistinguishability - (or: Quadratic residuosity strikes back). In T. Rabin, editor, *CRYPTO*, volume 6223 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2010. Full version at `http://eprint.iacr.org/2010/226`.

[BGK11] Z. Brakerski, S. Goldwasser, and Y. T. Kalai. Black-box circular-secure encryption beyond affine functions. In Y. Ishai, editor, *TCC*, volume 6597 of *Lecture Notes in Computer Science*, pages 201–218. Springer, 2011. Specific pointers refer to full version at `http://eprint.iacr.org/2009/485`.

[BGN05] D. Boneh, E.-J. Goh, and K. Nissim. Evaluating 2-dnf formulas on ciphertexts. In J. Kilian, editor, *TCC*, volume 3378 of *Lecture Notes in Computer Science*, pages 325–341. Springer, 2005.

[BGRV09] Z. Brakerski, S. Goldwasser, G. N. Rothblum, and V. Vaikuntanathan. Weak verifiable random functions. In O. Reingold, editor, *TCC*, volume 5444 of *Lecture Notes in Computer Science*, pages 558–576. Springer, 2009.

[BGV11] Z. Brakerski, C. Gentry, and V. Vaikuntanathan. Manuscript in preparation, 2011.

[BHHI10] B. Barak, I. Haitner, D. Hofheinz, and Y. Ishai. Bounded key-dependent message security. In H. Gilbert, editor, *EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 423–444. Springer, 2010.

[BHHO08] D. Boneh, S. Halevi, M. Hamburg, and R. Ostrovsky. Circular-secure encryption from decision diffie-hellman. In D. Wagner, editor, *CRYPTO*, volume 5157 of *Lecture Notes in Computer Science*, pages 108–125. Springer, 2008.

[BK10] Z. Brakerski and Y. T. Kalai. A framework for efficient signatures, ring signatures and identity based encryption in the standard model. Cryptology ePrint Archive, Report 2010/086, 2010. `http://eprint.iacr.org/`.

[BK11] Z. Brakerski and Y. T. Kalai. A parallel repetition theorem for leakage resilience. Cryptology ePrint Archive, Report 2011/250, 2011. `http://eprint.iacr.org/`.

[BKKV10] Z. Brakerski, Y. T. Kalai, J. Katz, and V. Vaikuntanathan. Overcoming the hole in the bucket: Public-key cryptography resilient to continual memory leakage. In *FOCS*, pages 501–510. IEEE Computer Society, 2010. Full version at `http://eprint.iacr.org/2010/278`.

[BKSY11] Z. Brakerski, J. Katz, G. Segev, and A. Yerukhimovich. Limits on the power of zero-knowledge proofs in cryptographic constructions. In Y. Ishai, editor, *TCC*, volume 6597 of *Lecture Notes in Computer Science*, pages 559–578. Springer, 2011.

[BPS09a] Z. Brakerski and B. Patt-Shamir. Distributed discovery of large near-cliques. In S. Tirthapura and L. Alvisi, editors, *PODC*, pages 324–325. ACM, 2009.

[BPS09b] Z. Brakerski and B. Patt-Shamir. Distributed discovery of large near-cliques. In I. Keidar, editor, *DISC*, volume 5805 of *Lecture Notes in Computer Science*, pages 206–220. Springer, 2009.

[BRS02] J. Black, P. Rogaway, and T. Shrimpton. Encryption-scheme security in the presence of key-dependent messages. In K. Nyberg and H. M. Heys, editors, *Selected Areas in Cryptography*, volume 2595 of *Lecture Notes in Computer Science*, pages 62–75. Springer, 2002.

[BS11] Z. Brakerski and G. Segev. Better security for deterministic public-key encryption: The auxiliary-input setting. In *CRYPTO*, volume 6841, page 539, 2011.

[BSW11] E. Boyle, G. Segev, and D. Wichs. Fully leakage-resilient signatures. In K. G. Paterson, editor, *EUROCRYPT*, volume 6632 of *Lecture Notes in Computer Science*, pages 89–108. Springer, 2011.

[BV11a] Z. Brakerski and V. Vaikuntanathan. Fully homomorphic encryption from ring-LWE and security for key dependent messages. In *CRYPTO*, volume 6841, page 501, 2011.

[BV11b] Z. Brakerski and V. Vaikuntanathan. Efficient fully homomorphic encryption from (standard) lwe. Cryptology ePrint Archive, Report 2011/344, 2011. To appear in FOCS 2011.

[CDH$^+$00] R. Canetti, Y. Dodis, S. Halevi, E. Kushilevitz, and A. Sahai. Exposure-resilient functions and all-or-nothing transforms. In *EUROCRYPT*, pages 453–469, 2000.

[CGKS95] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan. Private information retrieval. In *FOCS*, pages 41–50, 1995.

[CHKP10] D. Cash, D. Hofheinz, E. Kiltz, and C. Peikert. Bonsai trees, or how to delegate a lattice basis. In *EUROCRYPT*, pages 523–552, 2010.

[CKGS98] B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan. Private information retrieval. *J. ACM*, 45(6):965–981, 1998.

[CL01] J. Camenisch and A. Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In B. Pfitzmann, editor, *EUROCRYPT*, volume 2045 of *Lecture Notes in Computer Science*, pages 93–118. Springer, 2001.

[CMS99] C. Cachin, S. Micali, and M. Stadler. Computationally private information retrieval with polylogarithmic communication. In *EUROCRYPT*, pages 402–414, 1999.

[CS02] R. Cramer and V. Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In L. R. Knudsen, editor, *EUROCRYPT*, volume 2332 of *Lecture Notes in Computer Science*, pages 45–64. Springer, 2002.

[DGK$^+$10] Y. Dodis, S. Goldwasser, Y. T. Kalai, C. Peikert, and V. Vaikuntanathan. Public-key encryption schemes with auxiliary inputs. In D. Micciancio, editor, *TCC*, volume 5978 of *Lecture Notes in Computer Science*, pages 361–381. Springer, 2010.

[DH76] W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, November 1976.

[DHLW10] Y. Dodis, K. Haralambiev, A. López-Alt, and D. Wichs. Cryptography against continuous memory attacks. In *FOCS*, pages 511–520. IEEE Computer Society, 2010.

[DJ01] I. Damgård and M. Jurik. A generalisation, a simplification and some applications of paillier's probabilistic public-key system. In K. Kim, editor, *Public Key Cryptography*, volume 1992 of *Lecture Notes in Computer Science*, pages 119–136. Springer, 2001.

[DKL09] Y. Dodis, Y. T. Kalai, and S. Lovett. On cryptography with auxiliary input. In M. Mitzenmacher, editor, *STOC*, pages 621–630. ACM, 2009.

[DLWW11] Y. Dodis, A. Lewko, B. Waters, and D. Wichs. Storing secrets on continually leaky devices. Cryptology ePrint Archive, Report 2011/369, 2011. To appear in FOCS 2011.

[DOPS04] Y. Dodis, S. J. Ong, M. Prabhakaran, and A. Sahai. On the (im)possibility of cryptography with imperfect randomness. In *FOCS*, pages 196–205. IEEE Computer Society, 2004.

[DP08] S. Dziembowski and K. Pietrzak. Leakage-resilient cryptography. In *FOCS*, pages 293–302. IEEE Computer Society, 2008.

[DS05] Y. Dodis and A. Smith. Correcting errors without leaking partial information. In H. N. Gabow and R. Fagin, editors, *STOC*, pages 654–663. ACM, 2005.

[FKPR10] S. Faust, E. Kiltz, K. Pietrzak, and G. N. Rothblum. Leakage-resilient signatures. In D. Micciancio, editor, *TCC*, volume 5978 of *Lecture Notes in Computer Science*, pages 343–360. Springer, 2010.

[FRR$^+$10] S. Faust, T. Rabin, L. Reyzin, E. Tromer, and V. Vaikuntanathan. Protecting circuits from leakage: the computationally-bounded and noisy cases. In H. Gilbert, editor, *EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 135–156. Springer, 2010.

[Gen09a] C. Gentry. A fully homomorphic encryption scheme. PhD thesis, Stanford University, 2009. `crypto.stanford.edu/craig`.

[Gen09b] C. Gentry. Fully homomorphic encryption using ideal lattices. In *STOC*, pages 169–178, 2009.

[Gen10] C. Gentry. Toward basing fully homomorphic encryption on worst-case hardness. In *CRYPTO*, pages 116–137, 2010.

[GGM86] O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. *J. ACM*, 33(4):792–807, 1986.

[GH11a] C. Gentry and S. Halevi. Fully homomorphic encryption without squashing using depth-3 arithmetic circuits. Cryptology ePrint Archive, Report 2011/279, 2011. To appear in FOCS 2011.

[GH11b] C. Gentry and S. Halevi. Implementing gentry's fully-homomorphic encryption scheme. In K. G. Paterson, editor, *EUROCRYPT*, volume 6632 of *Lecture Notes in Computer Science*, pages 129–148. Springer, 2011.

[GHV10a] C. Gentry, S. Halevi, and V. Vaikuntanathan. $i$-hop homomorphic encryption and rerandomizable Yao circuits. In *CRYPTO*, pages 155–172, 2010.

[GHV10b] C. Gentry, S. Halevi, and V. Vaikuntanathan. A simple BGN-type cryptosystem from LWE. In *EUROCRYPT*, pages 506–522, 2010.

[GL89] O. Goldreich and L. A. Levin. A hard-core predicate for all one-way functions. In *STOC*, pages 25–32. ACM, 1989.

[GM82] S. Goldwasser and S. Micali. Probabilistic encryption and how to play mental poker keeping secret all partial information. In *STOC*, pages 365–377. ACM, 1982.

[GM84] S. Goldwasser and S. Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984.

[Gol04] O. Goldreich. Foundations of Cryptography - Basic Applications. Cambridge University Press, 2004.

[GPV08] C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In C. Dwork, editor, *STOC*, pages 197–206. ACM, 2008.

[GR05] C. Gentry and Z. Ramzan. Single-database private information retrieval with constant communication rate. In L. Caires, G. F. Italiano, L. Monteiro, C. Palamidessi, and M. Yung, editors, *ICALP*, volume 3580 of *Lecture Notes in Computer Science*, pages 803–815. Springer, 2005.

[GR10] S. Goldwasser and G. N. Rothblum. Securing computation against continuous leakage. In T. Rabin, editor, *CRYPTO*, volume 6223 of *Lecture Notes in Computer Science*, pages 59–79. Springer, 2010.

[HO09] B. Hemenway and R. Ostrovsky. Lossy trapdoor functions from smooth homomorphic hash proof systems. *Electronic Colloquium on Computational Complexity (ECCC)*, 16(127), 2009. `http://eccc.uni-trier.de/report/2009/127/`.

[HSH$^+$08] J. A. Halderman, S. D. Schoen, N. Heninger, W. Clarkson, W. Paul, J. A. Calandrino, A. J. Feldman, J. Appelbaum, and E. W. Felten. Lest we remember: Cold boot attacks on encryption keys. In P. C. van Oorschot, editor, *USENIX Security Symposium*, pages 45–60. USENIX Association, 2008.

[ILL89] R. Impagliazzo, L. A. Levin, and M. Luby. Pseudo-random generation from one-way functions (extended abstracts). In *STOC*, pages 12–24. ACM, 1989.

[IP07] Y. Ishai and A. Paskin. Evaluating branching programs on encrypted data. In S. P. Vadhan, editor, *TCC*, volume 4392 of *Lecture Notes in Computer Science*, pages 575–594. Springer, 2007.

[IPSW06] Y. Ishai, M. Prabhakaran, A. Sahai, and D. Wagner. Private circuits ii: Keeping secrets in tamperable circuits. In *EUROCRYPT*, pages 308–327, 2006.

[ISW03] Y. Ishai, A. Sahai, and D. Wagner. Private circuits: Securing hardware against probing attacks. In D. Boneh, editor, *CRYPTO*, volume 2729 of *Lecture Notes in Computer Science*, pages 463–481. Springer, 2003.

[JV10] A. Juma and Y. Vahlis. Protecting cryptographic keys against continual leakage. In T. Rabin, editor, *CRYPTO*, volume 6223 of *Lecture Notes in Computer Science*, pages 41–58. Springer, 2010.

[KJJ99] P. C. Kocher, J. Jaffe, and B. Jun. Differential power analysis. In M. J. Wiener, editor, *CRYPTO*, volume 1666 of *Lecture Notes in Computer Science*, pages 388–397. Springer, 1999.

[KKS11] Y. T. Kalai, B. Kanukurthi, and A. Sahai. Cryptography with tamperable and leaky memory. In *CRYPTO*, volume 6841, page 367, 2011.

[KO97] E. Kushilevitz and R. Ostrovsky. Replication is not needed: Single database, computationally-private information retrieval. In *FOCS*, pages 364–373, 1997.

[Koc96] P. C. Kocher. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In N. Koblitz, editor, *CRYPTO*, volume 1109 of *Lecture Notes in Computer Science*, pages 104–113. Springer, 1996.

[KPSY09] E. Kiltz, K. Pietrzak, M. Stam, and M. Yung. A new randomness extraction paradigm for hybrid encryption. In A. Joux, editor, *EUROCRYPT*, volume 5479 of *Lecture Notes in Computer Science*, pages 590–609. Springer, 2009.

[KV09] J. Katz and V. Vaikuntanathan. Signature schemes with bounded leakage resilience. In M. Matsui, editor, *ASIACRYPT*, volume 5912 of *Lecture Notes in Computer Science*, pages 703–720. Springer, 2009.

[LC03] P. Laud and R. Corin. Sound computational interpretation of formal encryption with composed keys. In J. I. Lim and D. H. Lee, editors, *ICISC*, volume 2971 of *Lecture Notes in Computer Science*, pages 55–66. Springer, 2003.

[Lip05] H. Lipmaa. An oblivious transfer protocol with log-squared communication. In J. Zhou, J. Lopez, R. H. Deng, and F. Bao, editors, *ISC*, volume 3650 of *Lecture Notes in Computer Science*, pages 314–328. Springer, 2005.

[LLL82] A. K. Lenstra, H. W. Lenstra, and L. Lovsz. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261:515–534, 1982. 10.1007/BF01457454.

[LLW11] A. B. Lewko, M. Lewko, and B. Waters. How to leak on key updates. In L. Fortnow and S. P. Vadhan, editors, *STOC*, pages 725–734. ACM, 2011.

[LP11] R. Lindner and C. Peikert. Better key sizes (and attacks) for LWE-based encryption. In A. Kiayias, editor, *CT-RSA*, volume 6558 of *Lecture Notes in Computer Science*, pages 319–339. Springer, 2011.

[LPR10] V. Lyubashevsky, C. Peikert, and O. Regev. On ideal lattices and learning with errors over rings. In *EUROCRYPT*, pages 1–23, 2010. Draft of full version was provided by the authors.

[LRW11] A. B. Lewko, Y. Rouselakis, and B. Waters. Achieving leakage resilience through dual system encryption. In Y. Ishai, editor, *TCC*, volume 6597 of *Lecture Notes in Computer Science*, pages 70–88. Springer, 2011.

[MGH10] C. A. Melchor, P. Gaborit, and J. Herranz. Additively homomorphic encryption with $d$-operand multiplications. In *CRYPTO*, pages 138–154, 2010.

[Mic00] D. Micciancio. The shortest vector in a lattice is hard to approximate to within some constant. *SIAM J. Comput.*, 30(6):2008–2035, 2000.

[Mic10] D. Micciancio. A first glimpse of cryptography's holy grail. *Commun. ACM*, 53:96–96, March 2010.

[MR04] S. Micali and L. Reyzin. Physically observable cryptography (extended abstract). In M. Naor, editor, *TCC*, volume 2951 of *Lecture Notes in Computer Science*, pages 278–296. Springer, 2004.

[MR09] D. Micciancio and O. Regev. Lattice-based cryptography. In *Post-Quantum Cryptography.* Springer, 2009.

[MTVY11] T. Malkin, I. Teranishi, Y. Vahlis, and M. Yung. Signatures resilient to continual leakage on memory and computation. In Y. Ishai, editor, *TCC*, volume 6597 of *Lecture Notes in Computer Science*, pages 89–106. Springer, 2011.

[MTY11] T. Malkin, I. Teranishi, and M. Yung. Efficient circuit-size independent public key encryption with kdm security. In K. G. Paterson, editor, *EUROCRYPT*, volume 6632 of *Lecture Notes in Computer Science*, pages 507–526. Springer, 2011.

[MV10] D. Micciancio and P. Voulgaris. A deterministic single exponential time algorithm for most lattice problems based on voronoi cell computations. In L. J. Schulman, editor, *STOC*, pages 351–358. ACM, 2010.

[NS09] M. Naor and G. Segev. Public-key cryptosystems resilient to key leakage. In S. Halevi, editor, *CRYPTO*, volume 5677 of *Lecture Notes in Computer Science*, pages 18–35. Springer, 2009.

[OS07] R. Ostrovsky and W. E. Skeith III. A survey of single-database private information retrieval: Techniques and applications. In T. Okamoto and X. Wang, editors, *Public Key Cryptography*, volume 4450 of *Lecture Notes in Computer Science*, pages 393–411. Springer, 2007.

[Pai99] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *EUROCRYPT*, pages 223–238, 1999.

[Pei09] C. Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In *STOC*, pages 333–342, 2009.

[Pie09] K. Pietrzak. A leakage-resilient mode of operation. In A. Joux, editor, *EUROCRYPT*, volume 5479 of *Lecture Notes in Computer Science*, pages 462–482. Springer, 2009.

[PSP$^+$08] C. Petit, F.-X. Standaert, O. Pereira, T. Malkin, and M. Yung. A block cipher based pseudo random number generator secure against side-channel key recovery. In M. Abe and V. D. Gligor, editors, *ASIACCS*, pages 56–65. ACM, 2008.

[PVW08] C. Peikert, V. Vaikuntanathan, and B. Waters. A framework for efficient and composable oblivious transfer. In D. Wagner, editor, *CRYPTO*, volume 5157 of *Lecture Notes in Computer Science*, pages 554–571. Springer, 2008.

[PW08] C. Peikert and B. Waters. Lossy trapdoor functions and their applications. In C. Dwork, editor, *STOC*, pages 187–196. ACM, 2008.

[RAD78] R. Rivest, L. Adleman, and M. Dertouzos. On data banks and privacy homomorphisms. In *Foundations of Secure Computation*, pages 169–177. Academic Press, 1978.

[Reg05] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In H. N. Gabow and R. Fagin, editors, *STOC*, pages 84–93. ACM, 2005.

[Riv97] R. L. Rivest. All-or-nothing encryption and the package transform. In E. Biham, editor, *FSE*, volume 1267 of *Lecture Notes in Computer Science*, pages 210–218. Springer, 1997.

[RS10] M. Rückert and M. Schneider. Estimating the security of lattice-based cryptosystems. Cryptology ePrint Archive, Report 2010/137, 2010. `http://eprint.iacr.org/`.

[SS10] D. Stehlé and R. Steinfeld. Faster fully homomorphic encryption. In M. Abe, editor, *ASIACRYPT*, volume 6477 of *Lecture Notes in Computer Science*, pages 377–394. Springer, 2010.

[SV10] N. P. Smart and F. Vercauteren. Fully homomorphic encryption with relatively small key and ciphertext sizes. In P. Q. Nguyen and D. Pointcheval, editors, *Public Key Cryptography*, volume 6056 of *Lecture Notes in Computer Science*, pages 420–443. Springer, 2010.

[SYY99] T. Sander, A. Young, and M. Yung. Non-interactive cryptocomputing for $NC^1$. In *FOCS*, pages 554–567, 1999.

[vDGHV10] M. van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan. Fully homomorphic encryption over the integers. In *EUROCRYPT*, pages 24–43, 2010. Full Version in `http://eprint.iacr.org/2009/616.pdf`.

# Appendix

# Other Research Projects

## A Parallel Repetition Theorem for Leakage Resilience [BK11]

A leakage resilient encryption scheme is one which stays secure even against an attacker that obtains a bounded amount of side information on the secret key (say $\lambda$ bits of "leakage"). A fundamental question is whether parallel repetition amplifies leakage resilience. Namely, if we secret share our message, and encrypt the shares under two independent keys, will the resulting scheme be resilient to $2\lambda$ bits of leakage?

Surprisingly, Lewko and Waters (FOCS 2010) showed that this is false. They gave an example of a public-key encryption scheme that is resilient to $\lambda$ bits of leakage, and yet its 2-repetition is not resilient to even $(1+\epsilon)\lambda$ bits of leakage. In their counter-example, the repeated schemes share secretly generated public parameters.

In this work, we show that under a reasonable strengthening of the definition of leakage resilience (one that captures known proof techniques for achieving non-trivial leakage resilience), parallel repetition *does* in fact amplify leakage. In particular, if fresh public parameters are used for each copy of the Lewko-Waters scheme, then their negative result does not hold, and leakage is amplified by parallel repetition.

More generally, we show that given $t$ schemes that are resilient to $\lambda_1, \ldots, \lambda_t$ bits of leakage, respectfully, their direct product is resilient to $\sum(\lambda_i - 1)$ bits. We present our amplification theorem in a general framework that applies other cryptographic primitives as well.

## Fully Homomorphic Encryption from Ring-LWE and Security for Key Dependent Messages [BV11a]

We present a somewhat homomorphic encryption scheme that is both very simple to describe and analyze, and whose security (quantumly) reduces to the worst-case hardness of problems on ideal lattices. We then transform it into a fully homomorphic encryption scheme using standard "squashing" and "bootstrapping" techniques introduced by Gentry (STOC 2009).

One of the obstacles in going from "somewhat" to full homomorphism is the requirement that the somewhat homomorphic scheme be circular secure. Namely, the scheme can be used to securely encrypt its own secret key. For all known somewhat homomorphic encryption schemes, this requirement was not known to be achievable under any cryptographic assumption, and had to be explicitly assumed. We take a step forward towards removing this additional assumption by proving that our scheme is in fact secure when encrypting polynomial functions of the secret key.

Our scheme is based on the ring learning with errors (RLWE) assumption that was recently introduced by Lyubashevsky, Peikert and Regev (Eurocrypt 2010). The RLWE assumption is reducible to worst-case problems on ideal lattices, and allows us to completely abstract out the lattice interpretation, resulting in an extremely simple scheme. For example, our secret key is $s$, and our public key is $(a, b = as + 2e)$, where $s, a, e$ are all degree $(n - 1)$ integer polynomials whose coefficients are independently drawn from easy to sample distributions.

## Better Security for Deterministic Public-Key Encryption: The Auxiliary-Input Setting [BS11]

Deterministic public-key encryption, introduced by Bellare, Boldyreva, and O'Neill (CRYPTO '07), provides an alternative to randomized public-key encryption in various scenarios where the latter exhibits inherent drawbacks. A deterministic encryption algorithm, however, cannot satisfy any meaningful notion of security when the plaintext is distributed over a small set. Bellare et al. addressed this difficulty by requiring semantic security to hold only when the plaintext has high min-entropy from the adversary's point of view.

In many applications, however, an adversary may obtain auxiliary information that is related to the plaintext. Specifically, when deterministic encryption is used as a building block of a larger system, it is rather likely that plaintexts do not have high min-entropy from the adversary's point of view. In such cases, the framework of Bellare et al. might fall short from providing robust security guarantees.

We formalize a framework for studying the security of deterministic public-key encryption schemes with respect to auxiliary inputs. Given the trivial requirement that the plaintext should not be efficiently recoverable from the auxiliary input, we focus on *hard-to-invert* auxiliary inputs. Within this framework, we propose two schemes: the first is based on the decisional Diffie-Hellman (and, more generally, on the $d$-linear) assumption, and the second is based on a rather general class of subgroup indistinguishability assumptions (including, in particular, quadratic residuosity and Paillier's composite residuosity). Our schemes are secure with respect to any auxiliary input that is subexponentially hard to invert (assuming the *standard* hardness of the underlying computational assumptions). In addition, our first scheme is secure even in the multi-user setting, where related plaintexts may be encrypted under multiple public keys. Constructing a scheme that is secure in the multi-user setting (even without considering auxiliary inputs) was identified by Bellare et al. as an important open problem.

## Limits on the Power of Zero-Knowledge Proofs in Cryptographic Constructions [BKSY11]

For more than 20 years, black-box impossibility results have been used to argue the infeasibility of constructing certain cryptographic primitives (e.g., key agreement) from others (e.g., one-way functions). A widely recognized limitation of such impossibility results, however, is that they say nothing about the usefulness of (known) *non*black-box techniques. This state of affairs is unsatisfying as we would at least like to rule out constructions using the set of techniques we have at our disposal.

With this motivation in mind, we suggest a new framework for black-box constructions that encompasses constructions with a nonblack-box flavor, specifically, those that rely on *zero-knowledge proofs* relative to some oracle. We show that our framework is powerful enough to capture the Naor-Yung/Sahai paradigm for building a (shielding) CCA-secure public-key encryption scheme from a CPA-secure one, something ruled out by a prior black-box separation result. On the other hand, we show that several of the known black-box impossibility results still hold even in a setting that allows for zero-knowledge proofs.

### Black-Box Circular-Secure Encryption Beyond Affine Functions [BGK11]

We show how to achieve public-key encryption schemes that can securely encrypt nonlinear functions of their own secret key. Specifically, we show that for any constant $d \in \mathbb{N}$, there exists a public-key encryption scheme that can securely encrypt any function $f$ of its own secret key, assuming $f$ can be expressed as a polynomial of total degree $d$. Such a scheme is said to be key-dependent message (KDM) secure w.r.t. degree-$d$ polynomials. We also show that for any constants $c, e$, there exists a public-key encryption scheme that is KDM secure w.r.t. all Turing machines with description size $c \log \lambda$ and running time $\lambda^e$, where $\lambda$ is the security parameter. The security of such public-key schemes can be based either on the standard decision Diffie-Hellman (DDH) assumption or on the learning with errors (LWE) assumption (with certain parameter settings).

In the case of functions that can be expressed as degree-$d$ polynomials, we show that the resulting schemes are also secure with respect to *key cycles* of any length. Specifically, for any polynomial number $n$ of key pairs, our schemes can securely encrypt a degree-$d$ polynomial whose variables are the collection of coordinates of all $n$ secret keys. Prior to this work, it was not known how to achieve this for nonlinear functions.

Our key idea is a general transformation that amplifies KDM security. The transformation takes an encryption scheme that is KDM secure w.r.t. *some* functions even when the secret keys are weak (i.e. chosen from an arbitrary distribution with entropy $k$), and outputs a scheme that is KDM secure w.r.t. a *richer* class of functions. The resulting scheme may no longer be secure with weak keys. Thus, in some sense, this transformation converts security with weak keys into amplified KDM security.

### A Framework for Efficient Signatures, Ring Signatures and Identity Based Encryption in the Standard Model [BK10]

In this work, we present a generic framework for constructing efficient signature schemes, ring signature schemes, and identity based encryption schemes, all in the standard model (without relying on random oracles).

We start by abstracting the recent work of Hohenberger and Waters (Crypto 2009), and specifically their "prefix method". We show a transformation taking a signature scheme with a very weak security guarantee (a notion that we call a-priori-message unforgeability under static chosen message attack) and producing a fully secure signature scheme (i.e., existentially unforgeable under adaptive chosen message attack). Our transformation uses the notion of chameleon hash functions, defined by Krawczyk and Rabin (NDSS 2000) and the "prefix method". Constructing such weakly secure schemes seems to be significantly easier than constructing fully secure ones, and we present *simple* constructions based on the RSA assumption, the *short integer solution* (SIS) assumption, and the *computational Diffie-Hellman* (CDH) assumption over bilinear groups.

Next, we observe that this general transformation also applies to the regime of ring signatures. Using this observation, we construct new (provably secure) ring signature schemes: one is based on the *short integer solution* (SIS) assumption, and the other is based on the CDH assumption over bilinear groups. As a building block for these constructions, we define a primitive that we call *ring trapdoor functions*. We show that ring trapdoor functions imply ring signatures under a weak definition, which enables us to apply our transformation to achieve full security.

Finally, we show a connection between ring signatures and identity based encryption (IBE) schemes. Using this connection, and using our new constructions of ring signature schemes, we obtain two IBE schemes: The first is based on the *learning with error* (LWE) assumption, and is similar to recently introduced LWE-based IBE schemes; The second is based on the $d$-linear assumption over bilinear groups.

### Hedged Public-Key Encryption: How to Protect Against Bad Randomness [BBN$^+$09]

Public-key encryption schemes rely for their IND-CPA security on per-message fresh randomness. In practice, randomness may be of poor quality for a variety of reasons, leading to failure of the schemes. Expecting the systems to improve is unrealistic. What we show in this paper is that we can, instead, improve the cryptography to offset the lack of possible randomness. We provide public-key encryption schemes that achieve IND-CPA security when the randomness they use is of high quality, but, when the latter is not the case, rather than breaking completely, they achieve a weaker but still useful notion of security that we call IND-CDA. This hedged public-key encryption provides the best possible security guarantees in the face of bad randomness. We provide simple RO-based ways to make in-practice IND-CPA schemes hedge secure with minimal software changes. We also provide non-RO model schemes relying on lossy trapdoor functions (LTDFs) and techniques from deterministic encryption. They achieve adaptive security by establishing and exploiting the anonymity of LTDFs which we believe is of independent interest.

### Distributed Discovery of Large Near-Cliques [BPS09a, BPS09b]

Given an undirected graph and $0 \leq \epsilon \leq 1$, a set of nodes is called $\epsilon$-near clique if all but an $\epsilon$ fraction of the pairs of nodes in the set have a link between them. In this paper we present a fast synchronous network algorithm that uses small messages and finds a near-clique. Specifically, we present a constant-time algorithm that finds, with constant probability of success, a linear size $\epsilon$-near clique if there exists an $\epsilon^3$-near clique of linear size in the graph. The algorithm uses messages of $O(\log n)$ bits. The failure probability can be reduced to $n^{-\Omega(1)}$ in $O(\log n)$ time factor, and the algorithm also works if the graph contains a clique of size $\Omega(n/\log^{\alpha}\log n)$ for some $\alpha \in (0, 1)$. Our approach is based on a new idea of adapting property testing algorithms to the distributed setting.

## From Absolute Distinguishability to Positive Distinguishability [BG09]

We study methods of converting algorithms that distinguish pairs of distributions with a gap that has an absolute value that is noticeable into corresponding algorithms in which the gap is always positive. Our focus is on designing algorithms that, in addition to the tested string, obtain a fixed number of samples from each distribution. Needless to say, such algorithms can not provide a very reliable guess for the sign of the original distinguishability gap, still we show that even guesses that are noticeably better than random are useful in this setting.

## Weak Verifiable Random Functions [BGRV09]

Verifiable random functions (VRFs), introduced by Micali, Rabin and Vadhan, are pseudorandom functions in which the owner of the seed produces a public-key that constitutes a commitment to all values of the function. The owner can then produce, for any input $x$, a proof that the function has been evaluated correctly on $x$, preserving pseudorandomness for all other inputs. No public-key (even a falsely generated one) should allow for proving more than one value per input.

VRFs are both a natural and a useful primitive, and previous works have suggested a variety of constructions and applications. Still, there are many open questions in the study of VRFs, especially their relation to more widely studied cryptographic primitives and constructing them from a wide variety of cryptographic assumptions.

In this work we define a natural relaxation of VRFs that we call weak verifiable random functions, where pseudorandomness is required to hold only for randomly selected inputs. We conduct a study of weak VRFs, focusing on applications, constructions, and their relationship to other cryptographic primitives. We show:

- **Constructions.** We present constructions of weak VRFs based on a variety of assumptions, including general assumptions such as (enhanced) trapdoor permutations, as well as constructions based on specific number-theoretic assumptions such as the Diffie-Hellman assumption in bilinear groups.

- **Separations.** Verifiable random functions (both weak and standard) cannot be constructed from one-way permutations in a black-box manner. This constitutes the first result separating (standard) VRFs from any cryptographic primitive.

- **Applications.** Weak VRFs capture the essence of constructing non-interactive zero-knowledge proofs for all **NP** languages.