Post-quantum side-channel security from hard physical learning problems

Clément Hoffmann

October 2024

École polytechnique de Louvain
UCLouvain
Louvain-la-Neuve
Belgium

Thesis Committee:

Pr. François-Xavier Standaert	*UCLouvain*
Dr. Pierrick Méaux	*Luxembourg University*
Dr. Melissa Rossi	*ANSSI*
Dr. Mehdi Tibouchi	*NTT*
Pr. Thomas Peters	*UCLouvain*
Pr. David Bol	*UCLouvain*

Acknowledgements

Completing this thesis has been a challenging yet rewarding journey, and I am deeply thankful to everyone who accompanied me along the way, offering encouragement, advice, and support. Initially, I hoped to thank each person who contributed to this work by name, but I quickly realized the list would be far too extensive. From colleagues and collaborators to friends and family, so many have left an indelible mark on this journey in countless ways. Every discussion, piece of advice, and gesture of support has played a role in helping me reach this milestone, and for that, I am truly grateful.

My deepest thanks go to my supervisors, FX and Pierrick, who each contributed immensely in their own unique way. I am grateful for the time and effort they invested in guiding me through my discovery of the research environment. I am especially thankful to Pierrick for his technical expertise and unwavering support during our weekly discussions. I could not have imagined a better environment for my PhD.

I would also like to extend my gratitude to the members of my jury for dedicating their time to reviewing this manuscript and for the insightful discussions that arose from it. While pursuing a PhD often feels like a solitary endeavor, I truly valued these moments of honest and constructive feedback. I am also deeply thankful to my co-authors and all the collaborators I had the privilege to work with throughout this journey—your guidance and expertise have been invaluable, and I have learned a lot from each of you.

A heartfelt thank you goes to my colleagues for creating such a positive and inspiring work environment. I have learned a great deal through our discussions and collaborations, and the time we shared during beer meetings, squash matches, or long weekend bike rides has meant the world to me. I truly cherished the camaraderie and the experiences we shared. I am also grateful to my master's professors for introducing me to the fascinating field of cryptography, and I owe special thanks to Christophe Clavier for sparking my interest in side-channel attacks.

Je tiens à remercier profondément ma famille pour votre soutien sans faille, surtout pendant les moments les plus difficiles de cette thèse et tout au long de mon parcours. Avoir pu me concentrer pleinement sur

mes études a été un vrai privilège, et cela n'aurait pas été possible sans vous.

Enfin, et surtout, un immense merci à mes amis et à ma copine Clarisse. Merci d'être venus me voir en Belgique, d'avoir pris le temps d'écouter mes plaintes et d'avoir été là pour moi à chaque étape de ces quatre années. J'espère que nous aurons encore de nombreuses belles aventures à partager ensemble. Vous êtes tous les amours de ma vie, et je suis infiniment reconnaissant de vous avoir à mes côtés.

Abstract

The widespread adoption of public-key cryptography has enabled the secure transmission of information, particularly over the Internet. However, with the potential development of a quantum computer powerful enough to break current encryption standards, research into post-quantum cryptography—cryptographic protocols resistant to quantum attacks—has flourished over the past two decades. Meanwhile, side-channel attacks, which exploit physical information such as power consumption or electromagnetic emissions, have proven both practical and effective against implementations that lack proper countermeasures.

As post-quantum cryptography nears the final stages of its standardization process, addressing the need for secure key exchange mechanisms and digital signature algorithms, many important questions continue to drive ongoing research. Notably, since the initial design of standardized post-quantum schemes did not prioritize side-channel security, the systematic study of this issue remains an active and evolving area of research.

In this work, we first conduct a side-channel analysis of several post-quantum schemes currently undergoing standardization. Through this analysis, we observe that while some schemes have structural vulnerabilities to leakage attacks, others demonstrate promising properties for secure implementation. We then present insights on how to design post-quantum protocols that are more efficient to implement with protection against side-channel attacks.

We then examine the side-channel security of post-quantum schemes at the operation level, showing that the security of some of the most critical operations can be reduced to the hardness of certain physical learning problems. We provide several examples of this synergy, including the use of inexact computing to implement key operations as physical functions and the application of existing learning problems for side-channel analysis of post-quantum schemes.

Finally, after extending our hardness analysis of various physical learning problems, we take an initial step towards a generalized approach for learning problem reductions. We introduce a new generalized unstructured learning problem and provide the first reduction from this problem to the classical Learning with Errors (LWE) problem.

Contents

1 Introduction **1**
- 1.1 Outline of the thesis . . . 3
- 1.2 Other contributions . . . 6

2 Background **9**
- 2.1 Classical definitions in cryptography . . . 9
  - 2.1.1 Overview of cryptographic concepts . . . 10
  - 2.1.2 Security model in black-box cryptography . . . 13
- 2.2 Learning problems . . . 14
  - 2.2.1 Classical learning problems, variants & reductions. 15
  - 2.2.2 Hard physical learning problems. . . . 16
- 2.3 Side-channel attacks . . . 21

3 Physical security of post-quantum schemes **25**
- 3.1 Leakage analysis: the case of Kyber . . . 26
  - 3.1.2 Shortcut formulas for SPA and DPA . . . 32
  - 3.1.3 Generic intuitions . . . 34
  - 3.1.4 Applications to CRYSTALS-Kyber . . . 38
- 3.2 Secure and efficient implementation of Dilithium . . . 44
  - 3.2.2 Sensitivity analysis . . . 50
  - 3.2.3 Improved masked gadgets . . . 54
  - 3.2.4 Implementation . . . 62
  - 3.2.5 Benchmarks . . . 63
- 3.3 Designing `POLKA`, a PQC KEM tailored for physical security 70
  - 3.3.1 Technical Overview of `POLKA` & Related Works . . 74
  - 3.3.3 `POLKA`: Rationale and Specifications . . . 81
  - 3.3.4 Side-Channel Security Analysis . . . 86

4 Physical learning problems applications to side-channel analysis **97**
- 4.1 Physical security from inexact computing and learning problems reductions . . . 98
  - 4.1.1 LPPN physical imperfections . . . 101
  - 4.1.2 Reduction between LPN and LPPN . . . 106
  - 4.1.3 Masked LPPN . . . 113

4.2 Generalizing inexact computing to higher dimension . . . 115
4.2.2 Introducing LWPE with a design example 119
4.2.3 LWPE design Space 127
4.3 Learning with physical rounding for linear and quadratic leakage functions 132
4.3.2 Appetizer: threat model and serial LWPR 138
4.3.3 LWPR for linear leakage functions 140
4.3.4 Security against worst-case leakages 145
4.3.5 Empirical validation 148
4.3.6 Further generalizations 157
4.4 Towards a generalized approach for learning problems reductions 160
4.4.1 LWE-OD reduction to q-LWE 163
4.4.2 q-LWE reduction to LWE 169

5 Conclusion and future research **175**

1 Introduction

Cryptology is the science of communicating secure information in the presence of adversarial behavior. While it has been used at least since Antiquity to conceal sensitive messages, it truly evolved into a formal science with the advent of modern computers after Word War II. Today, cryptography plays a key role in various applications, such as ensuring secure communication through encryption or verifying the integrity and authenticity of documents through Digital Signature Algorithm (DSA). In every case, cryptography relies on mathematics to guarantee that no unauthorized third party intercepting data in transit can alter it or extract sensitive information.

Modern cryptography operates on realistic assumptions about the capabilities of an adversary. One fundamental principle is *Kerckhoffs' principle*, which asserts that the security of a cryptographic system should depend solely on the secrecy of a key, rather than the secrecy of the system's operations. Traditionally, cryptography also relied on the *black-box* model, in which adversaries can only observe the inputs and outputs of a system without access to its internal computations. Another classic assumtion is to give a *limited power of computation* to an adversary, commonly a polynomial number of operations on a classical Turing Machine. While Kerckhoffs' principle remains central to cryptographic design, advances in technology and research have eroded the black-box model and computational assumptions.

Towards the end of the 20th century, it was discovered that the physical characteristics of devices implementing cryptographic algorithms, such as power consumption and electromagnetic emissions, could reveal information about the internal computations. This led to the rise of *side-channel* attacks (SCA), where adversaries exploit physical leakage to bypass the black-box assumption and gather sensitive information.

To defend against these attacks, two complementary strategies are used. The first involves developing heuristic countermeasures against specific, known attacks. While this approach often leads to practical solutions and state-of-the-art protections, it can be difficult to generalize across different cryptographic schemes or to predict how these countermeasures interact with one another. The second strategy takes a theoretical approach, attempting to rigorously prove security against the

most realistic adversary model possible. However, this often requires simplifying assumptions, which can limit the applicability of the results in real-world scenarios.

The other assumption being challenged is the adversary's computational model. Recent progress in quantum computing have made the potential existence of quantum computers more tangible. These quantum machines could perform certain computations exponentially faster than classical computers, which poses a serious threat to most modern public-key cryptosystems. Although the timeline for building a sufficiently powerful quantum computer is still debated, there are strong reasons to begin preparing for this eventuality. First, a powerful quantum computer will likely be developed in secret, providing its creators—such as governmental agencies—a strategic advantage. Second, data encrypted today may still be valuable decades from now, and transitioning to post-quantum cryptography now provides protection against "harvest now, decrypt later" attacks.

As a result, post-quantum cryptography (PQC) has become an active area of research over the past two decades. Interest in this field has intensified with the National Institute of Standards and Technology (NIST) PQC Standardization Process, initiated in 2016 and now nearing completion. In the context of the black-box model, the underlying PQC problems are getting more and more well-known after years of cryptanalisys and instanciations.

Side-channel analysis of the simple (symmetric) cryptosystems is also maturing, as the gap between heuristic countermeasures and theoretical foundations begins to narrow [Bel+20]. In the black-box model, PQC is also reaching maturity, with some post-quantum schemes achieving performance on par with, or even surpassing, their pre-quantum counterparts. However, the interaction between post-quantum cryptography and side-channel leakage remains largely unexplored.

The aim of this thesis is to investigate the security of post-quantum cryptographic schemes in the context of side-channel attacks, and take a step toward systematically understanding the interactions between PQC and SCA.

First, we examine the newly established NIST standards. We identify structural weaknesses in the design of some post-quantum protocols but demonstrate that these flaws are not inherent to post-quantum cryptography. We show that other PQ schemes can be as efficient to implement as their pre-quantum counterparts, and we propose strategies to avoid the structural vulnerabilities to side-channel attacks found in certain designs.

Next, we conduct a side-channel analysis of post-quantum protocols at the operational level. We show that the physical security of some key operations, especially those involving the secret key, can be reduced to hard physical learning problems. We introduce several such physical learning problems and extend their hardness analysis. Lastly, we define a new generalized learning problem that encompasses most unstructured classical learning problems as well as the physical learning problems discussed earlier. We make an initial step toward reducing this generalized problem to the classical Learning With Errors (LWE) problem, establishing a sufficient condition for this reduction.

1.1 Outline of the thesis

The main focus of this thesis is the study of side-channel aspects of post-quantum cryptographic (PQC) schemes. While the manuscript is primarily centered on lattice-based cryptography, many of the concepts also apply to other areas of PQC, particularly code-based cryptography.

The manuscript is organized into two main chapters, each addressing the problem at a different level. The first chapter concentrates on scheme-level study, which includes leakage analysis, masked software implementations, and the design of a new scheme optimized for physical attack resistance. The second chapter shifts focus to the operation level, specifically exploring the connections between learning problems and the leakage issues that arise in such analyses.

Chapter 2 provides the necessary background. Each chapter is divided into sections based on peer-reviewed research papers (except Section 4.4 that is not yet published) I contributed to, with references to these papers included in the section titles. I have omitted parts of these papers in which I did not contribute, such as FPGA considerations in Sections 4.1, 4.2.2 and 4.3, and some security proofs in Section 3.3.

In Chapter 3, we analyze various lattice-based schemes using different approaches. The first two sections examine the physical security of two NIST PQC standards: Kyber in Section 3.1 and Dilithium in Section 3.2. These two studies, however, take almost opposite approaches.

In Section 3.1, we focus solely on leakage analysis, evaluating potential side-channel attack vectors in schemes using the Fujisaki-Okamoto (FO) transform. Applying these results to Kyber, we conclude that it is unlikely to perform efficiently in scenarios where leakage occurs.

In contrast, Dilithium proved to be much more suitable for efficient, physically secure implementations. Therefore, in Section 3.2, we provide a state-of-the-art software implementation that is protected against

side-channel attacks. We improved on two fronts: first, by performing a more refined sensitivity analysis to adjust the level of physical security for each intermediate computation; second, by improving masking techniques using new boolean-arithmetic conversion algorithms.

In Section 3.3, we give leads for solutions to the problems raised in Section 3.1 by propounding a new Public Key Encryption (PKE) scheme, `Polka`. Designed with leakage resistance in mind, `Polka` is optimized for efficiency in a masked implementation context. While it may not aim to become a standard, `Polka` introduces new ideas and techniques for achieving efficient side-channel protection.

In the analyses presented in Chapter 3, we frequently discuss the sensitivity of intermediate computations. Some computations can be left unprotected, as their security proofs guarantee they do not leak sensitive information (e.g., intermediate values retrievable from a public signature). For the remaining operations, two key questions arise: what level of physical security is required, and what is the best way to implement these operations? In Chapter 4, we discuss how studying learning problems can help to answer these questions and improve physical security at the operation level.

First, in Sections 4.1 and 4.2.2, we investigate how faulty inner products can be efficiently implemented in hardware using inexact computing, without generating errors. Since these faulty inner products directly involve the secret key, transforming them into key-homomorphic operations makes them easier to mask. By eliminating the need for a Pseudo Random Number Generator (PRNG), traditionally used to generate independent errors that will be added to a correct inner product, we reduce the attack surface for side-channel attacks and avoids unnecessary computations.

In Section 4.1, we study the implementation of a faulty inner product over $\mathbb{F}_2$, emulating the LPN problem. Using physical errors in the inexact computation gives rise to the Learning Parity with Physical Noise (LPPN) problem. Although this problem is not new [Kam+20b], we discovered that the error generated is output-dependent, meaning the error distribution is correlated with the correct value of the inner product. We modelize this output dependant vairant of LPN with a new problem called LPN with Output Dependencies (LPN-OD), formally defined in Definition 15. We provide a tight reduction from LPN-OD to LPN, along with a formula to precisely quantify the (small) security loss due to the asymmetry of the error. This provides an example of how theoretical work on learning problems can contribute to side-channel protected implementations.

In Section 4.2.2, we generalize this result by introducing and implementing the Learning with Physical Errors (LWPE) problem. After demonstrating that directly applying LPPN techniques leads to undesirable data dependencies, we propose design modifications to eliminate these dependencies. Additionally, we show that these tweaks also resolve the output dependencies observed in LPPN, mitigating the need for a new reduction.

In Section 4.3, we extend the heuristic hardness analysis of the Learning with Physical Rounding (LWPR) problem [Duv+21]. The samples in this problem are leakage measurements of an inner product between a secret key and public vectors. The leakage function behaves similarly to a rounding function, obfuscating the linear equations. The hardness of this problem can be applied to build rekeying operations and assist in analyzing the sensitivity of intermediate variables in post-quantum schemes. While the physical function varies by device and cannot be expressed as a precise mathematical expression, it can be approximated by a modelization. In [Duv+21], the hardness analysis was conducted modelizing the leakage as the Hamming weight function. We extend this analysis to leakages that are approximated as linear or quadratic functions (the functions variables being the bits the leaked value with bounded coefficients).

Throughout these sections, we illustrate how analyzing variants of learning problems can contribute to physically secure implementations. An interesting phenomenon is that these learning problems appear "in the wild": rather than being intentionally used as building blocks, their faulty inner product structures emerge while studying the leakage characteristics of implementations. To analyze these problems, we take a first steps toward building a generalized approach to learning problem reductions in Section 4.4. We introduce a generalized learning problem called Learning with Errors with Output Dependencies (LWE-OD), which encompasses LWE, LWR, and LWPR as particular cases. Using rejection sampling, we provide a proof of reduction from LWE-OD to LWE. The main result, found in Theorem 4, presents a system of equations involving the error distributions of LWE and LWE-OD, ensuring that if the system holds, the LWE-OD instance is at least as hard as the LWE instance. However, this result remains limited, as the system does not readily allow the identification of LWE instances that are at least as hard as LWE-OD instances, or vice versa. This reduction, if extended to more practical usecase, could benefit most others sections of this chapter, as LWE-OD encompasses the problems that raised from inexact computing in Sections 4.1 and 4.2.2 and the LWPR problem studied in Section 4.3.

1.2 Other contributions

We briefly describe the published results obtained during this thesis that are not discussed in detail in this manuscript.

These works focus on Fully Homomorphic Encryption (FHE), and more specifically, Hybrid Homomorphic Encryption (HHE). FHE schemes enable the evaluation of arbitrary functions on encrypted data, but they are often associated with significant performance overheads and ciphertext expansion. HHE addresses these issues by encrypting a symmetric key with the more computationally expensive FHE scheme, which is then sent to the server. The client can subsequently encrypt private data using a symmetric cipher and send it to the server, which then performs homomorphic decryption of the symmetric cipher and proceeds with the private computations.

Although, in theory, any symmetric encryption scheme can be evaluated homomorphically, using standard schemes like AES has been shown to be inefficient [GHS12]. As a result, the current state-of-the-art involves using specialized ciphers optimized for homomorphic evaluation. These FHE-friendly ciphers include, but are not limited to, LowMC [Alb+15], Kreyvium [Can+16], FiLIP [Méa+19], Rasta [Dob+18], Hera [Cho+21], Elisabeth [Cos+22a], and Rubato [Ha+22].

In a first work [HMR20], we implemented the FiLIP stream cipher [Méa+19] using the TFHE [Chi+20a] homomorphic protocol. While previous implementations of FiLIP had been achieved using second-generation FHE schemes (such as HELib [HS14]) in [Méa+19], this work represents the first implementation with a third-generation FHE scheme. We also explored various stream cipher variants and introduced optimizations in the FHE scheme to improve efficiency.

In [Cos+22a], we explored HHE using a case-optimized approach. Given the complexity of fully homomorphic decryption, we observed that existing HHE protocols often overlooked potential server-side use cases after HHE operations, typically supporting only limited server-side computations (e.g., a few multiplications). Our work was the first to introduce an HHE protocol enabling a practical real-world application, specifically neural network inference. To achieve this, we developed Elisabeth, a family of stream ciphers, and conducted its security analysis. We also implemented a homomorphic evaluation of an Elisabeth instance in Rust, along with the server-side use case.

After Elisabeth-4 (the instance we proposed in [Cos+22a]) was attacked in [Gil+23], we proposed different patches in [HMS23]. The security analysis of these patches required the implementation of efficient Boolean Function algorithms to recover the cryptographic criteria required for the security proofs.

Finally, in a preprint [Ara+24], we demonstrated that Margrethe, one of Elisabeth-4 patches, can be used for secure and efficient transciphering for FHE-based Multi Party Computation. We show that the requirements for a symmetric encryption scheme to be employed in such protocols can be reduced to its security against Related-Key Attacks (RKA). We then establish the RKA security of Margrethe and provide an optimized implementation for transciphering operations in this context.

2 Background

We present the standard background and notations used throughout the manuscript. Some sections may include additional background subsections to introduce notations and concepts relevant only to that specific section. However, most of the necessary background is covered in this chapter, and each subsequent section will reference the relevant material when needed.

Notations

We define the polynomial ring $R_q = \mathbb{Z}_q[X]/(X^n + 1)$. We denote polynomials with lowercase letters (e.g., p), a vector with lowercase bold letters (e.g., $\mathbf{v}$), and a matrix with uppercase bold letters (e.g., $\mathbf{M}$).

For $z, \alpha \in \mathbb{Z}$ we write $z \bmod^{\pm} \alpha$ to mean the unique integer z' in $]-\frac{\alpha}{2}, \frac{\alpha}{2}]$ with $z \equiv z' \bmod \alpha$. The notation $\mathbf{z} \bmod^{\pm} \alpha$ means that all the coefficients in $\mathbf{z}$ are reduced modulo α to the range $(-\frac{\alpha}{2}, \frac{\alpha}{2}]$ With this, we can define the following norms on $\mathbb{Z}_q$, R_q and R_q^k respectively:

$$\|z\|_\infty = |z \bmod^{\pm} q|, \quad \|p\|_\infty = \max_i \|p_i\|_\infty, \quad \|\mathbf{w}\|_\infty = \max_i \|\mathbf{w}_i\|_\infty,$$

with $z \in \mathbb{Z}_q$, $p \in R_q$, p_i being the i-th coefficient of p, $\mathbf{w} \in R_q^k$ and $\mathbf{w}_i$ being the i-th polynomial in $\mathbf{w}$.

Let X be a random variable, we denote by $X \leftarrow \mathcal{D}$ that X follows $\mathcal{D}$ if $\mathcal{D}$ is a probability distribution, or that X is chosen uniformly at random in $\mathcal{D}$ if $\mathcal{D}$ is a set.

2.1 Classical definitions in cryptography

In this section, we present several classical cryptographic definitions that are useful for understanding the works discussed in this thesis. These definitions are taken from the second edition of *Introduction to Modern Cryptography* by Katz and Lindell [KL14], which serves as a key reference for cryptographers. Although most of the work I have chosen to present does not heavily rely on these definitions—since my focus is primarily on studying leakage problems in existing schemes—they still provide valuable context and contribute to a deeper understanding of the environment in which my work is situated.

2.1.1 Overview of cryptographic concepts

We provide an high-level description along with a formal definition for the cryptographic concepts used in this manuscript.

Signature Scheme. A **digital signature scheme** ensures the authenticity and integrity of a message. It allows a sender to digitally "sign" a message using a private key, creating a signature that any recipient can verify using the sender's public key. Digital signatures provide:

- **Authenticity:** The signature confirms the message came from the claimed sender.
- **Integrity:** Any change to the message invalidates the signature.
- **Non-repudiation:** The sender cannot deny having signed the message, as only they possess the private key.

Signature schemes are foundational in securing online transactions, verifying software updates, and enabling protocols like SSL/TLS.

Definition 1 (Digital Signature Scheme)**.** *A* digital signature scheme $\mathcal{S} = (K, \mathit{Sign}, \mathit{Verify})$ *consists of three polynomial-time algorithms:*

- ***Key Generation*** *algorithm* K*:*

  $$K(1^k) \to (sk, pk),$$

  where 1^k *denotes the security parameter,* sk *is the private (signing) key, and* pk *is the public (verification) key. The key generation algorithm outputs a pair of keys: a private key* sk *and a public key* pk*.*

- ***Signing*** *algorithm Sign:*

  $$\mathit{Sign}(sk, m) \to \sigma,$$

  where m *is a message, and* σ *is the digital signature corresponding to the message. The signing algorithm takes the private key* sk *and a message* m*, and produces a signature* σ *for the message.*

- ***Verification*** *algorithm Verify:*

  $$\mathit{Verify}(pk, m, \sigma) \to \mathit{True/False},$$

  where pk *is the public key,* m *is the message, and* σ *is the signature. The verification algorithm checks whether* σ *is a valid signature for the message* m *under the public key* pk*. It outputs True if the signature is valid and False otherwise.*

A digital signature scheme is considered secure *if it satisfies the following two properties:*

- ***Unforgeability:*** *It should be computationally infeasible for an adversary to generate a valid signature for any message that has not been signed by the legitimate signer, even if they have access to the public key pk and some number of valid signatures for other messages.*
- ***Verifiability:*** *Given the public key pk, anyone should be able to verify the validity of a signature σ for a message m by using the verification algorithm. If σ is a valid signature on m, then it must be accepted as such.*

Formally, the security of a digital signature scheme is typically defined in terms of an adversary's ability to forge signatures. Specifically, we say that a digital signature scheme is strongly unforgeable under chosen message attack (CMA) *if no polynomial-time adversary can produce a valid signature for any new message with non-negligible probability, given access to the signing oracle.*

Encryption Scheme. An **encryption scheme** ensures the confidentiality of a message by transforming plaintext into ciphertext using a key. Only those possessing the correct decryption key can recover the original plaintext. Encryption schemes can be broadly categorized as:

- **Symmetric Encryption:** Uses a single shared secret key for both encryption and decryption (e.g., AES).
- **Asymmetric Encryption:** Uses a pair of keys—public for encryption and private for decryption (e.g., RSA, ElGamal).

Encryption schemes secure communications, protecting sensitive data like financial information, passwords, or personal data.

Definition 2 (Private-Key Encryption Scheme)**.** *A private-key encryption scheme is a tuple of probabilistic polynomial-time algorithms* (Gen, Enc, Dec) *such that:*

1. *The key-generation algorithm Gen takes as input* 1^n *(i.e., the security parameter written in unary) and outputs a key k; we write* $k \leftarrow Gen(1^n)$ *(emphasizing that Gen is a randomized algorithm). We assume without loss of generality that any key k output by* $Gen(1^n)$ *satisfies* $|k| \geq n$*.*

2. *The encryption algorithm Enc takes as input a key k and a plaintext message $m \in \{0,1\}^*$, and outputs a ciphertext c. Since Enc may be randomized, we write this as $c \leftarrow Enc_k(m)$.*

3. *The decryption algorithm Dec takes as input a key k and a ciphertext c, and outputs a message m or an error. We assume that Dec is deterministic, and so write $m := Dec_k(c)$ (assuming here that Dec does not return an error). We denote a generic error by the symbol $\perp$.*

It is required that for every n, every key k output by $Gen(1^n)$, and every $m \in \{0,1\}^*$, it holds that:

$$Dec_k(Enc_k(m)) = m.$$

If (Gen, Enc, Dec) is such that for k output by $Gen(1^n)$, algorithm Enc_k is only defined for messages $m \in \{0,1\}^{\ell(n)}$, then we say that (Gen, Enc, Dec) is a fixed-length private-key encryption scheme for messages of length $\ell(n)$.

Key Encapsulation Mechanism (KEM). A **key encapsulation mechanism (KEM)** is a cryptographic method used to securely exchange symmetric keys in public-key systems. Instead of encrypting the entire message using a public-key encryption scheme (which can be inefficient), KEM encrypts only the symmetric key that will be used for subsequent encryption of the message. KEM ensures:

- **Efficiency:** Public-key operations are limited to encrypting a small symmetric key.
- **Scalability:** The symmetric key can be used with fast symmetric encryption algorithms for large amounts of data.

Definition 3 (Key Encapsulation Mechanism (KEM))**.** *A* Key Encapsulation Mechanism (KEM) *consists of three algorithms:*

- ***Key Generation** algorithm K:*

  $$K(1^k) \to (pk, sk),$$

  where 1^k is the security parameter, pk is the public key, and sk is the secret key. The key generation algorithm outputs a key pair: a public key pk and a secret key sk.

- ***Encapsulation*** *algorithm Encaps:*

$$Encaps(pk) \rightarrow (c, k),$$

where pk is the public key and k is a key that is encapsulated. The encapsulation algorithm takes the public key pk and produces a ciphertext c and a shared secret key k. The key k is intended to be shared between the parties, while c is the encapsulated key (ciphertext).

- ***Decapsulation*** *algorithm Decaps:*

$$Decaps(sk, c) \rightarrow k,$$

where sk is the secret key and c is the ciphertext from the encapsulation algorithm. The decapsulation algorithm takes the secret key sk and the ciphertext c, and recovers the shared secret key k.

2.1.2 Security model in black-box cryptography

Modern cryptography emphasizes rigorous definitions and proofs to ensure a system remains secure under well-defined attack scenarios. Security against a chosen-ciphertext attack (CCA) is one of the strongest notions of security for encryption schemes. It ensures that even if an adversary can access a decryption oracle—a black-box system that decrypts arbitrary ciphertexts—the adversary cannot derive useful information about the plaintext of a target ciphertext. This robustness is essential for protecting communication protocols like HTTPS or VPNs, where adversaries may intercept and alter messages.

The formal definition of CCA-security stems from the notion of indistinguishability. A cryptographic scheme is CCA-secure if no polynomial-time adversary can distinguish between the encryptions of two chosen plaintexts, even with oracle access to decrypt other ciphertexts. This guarantees that even under active attacks, where an adversary can manipulate and analyze the system's response, the confidentiality of the encrypted messages is preserved. The formal definition of CCA-security is given below:

Definition 4 (CCA-Security)**.** *An encryption scheme $\mathcal{E} = (K, E, D)$ is* CCA-secure *if for all probabilistic polynomial-time (PPT) adversaries $\mathcal{A}$, the following holds. Let $\mathcal{A}$ be an adversary that is given access to an encryption oracle and a decryption oracle, denoted E and D, respectively, where the decryption oracle is restricted in that it may not be queried on the ciphertext that is returned as the challenge ciphertext.*

Let k be a security parameter. The adversary $\mathcal{A}$ is said to win the CCA game if it can distinguish between the two scenarios outlined below with non-negligible advantage. Formally, we define the game between an adversary $\mathcal{A}$ and a challenger as follows:

- *The challenger generates a random key $K \in \mathcal{K}$ using the key generation algorithm K.*
- *The challenger generates two messages $m_0, m_1 \in \{0,1\}^n$ and flips a coin $b \in \{0,1\}$. The challenger then computes the ciphertext $c = E_K(m_b)$.*
- *The adversary $\mathcal{A}$ is given access to the encryption oracle E and the decryption oracle D, where the decryption oracle is restricted from returning the plaintext corresponding to the challenge ciphertext c.*
- *The adversary can make polynomially many queries to the oracles, and eventually outputs a guess $\hat{b}$ of the value of b.*

The adversary $\mathcal{A}$ wins the game if $\hat{b} = b$. The encryption scheme $\mathcal{E}$ is considered CCA-secure if for all adversaries $\mathcal{A}$, the probability that $\mathcal{A}$ wins the game is negligibly greater than $1/2$, i.e.,

$$\left|\Pr[\hat{b} = b] - \frac{1}{2}\right| \leq \mathit{negligible}(k)$$

where the probability is taken over the coin flips of the challenger and the random choices made by $\mathcal{A}$.

2.2 Learning problems

We broadly define a *learning problem* as a computational problem that involves analyzing an inexact linear system of equations over a ring. The ring can be either an integer-based one (e.g., $\mathbb{F}_q$) or a polynomial-based one (e.g., R_q). The way the inexact equations are generated varies depending on the specific learning problem. In some cases, the errors are probabilistic and drawn from an error distribution, such as discrete Gaussian, rounded Gaussian, binomial, or uniform distributions. In other cases, the errors are deterministic, such as bit rounding.

Learning problems typically come in two forms: a search version and a decision version. In the *search version*, the adversary's goal is to solve the system and recover the secret key. In the *decision version*, the adversary must distinguish between noisy equations and uniformly random ones. For most learning problems, these two versions are equally difficult under certain parameter conditions.

2.2.1 Classical learning problems, variants & reductions.

The LPN problem can be specified as follows [LF06]::

Definition 5 (LPN problem)**.** *Let $\langle ., . \rangle$ denote the binary inner product, $\boldsymbol{k}$ be a random n-bit secret, $\epsilon \in [0, \frac{1}{2}]$ be a noise parameter, Ber_ϵ be the Bernoulli distribution with parameter ϵ (if $e \leftarrow \mathsf{Ber}_\epsilon$, then $\Pr[e = 1] = \epsilon$), and $D^\epsilon_{\boldsymbol{k}}$ be the distribution defined as:*

$$D^\epsilon_{\boldsymbol{k}} =: \{(\boldsymbol{x}, \langle \boldsymbol{x}, \boldsymbol{k} \rangle \oplus e) : \boldsymbol{x} \leftarrow \{0,1\}^n; e \leftarrow \mathsf{Ber}_\epsilon\}.$$

Let $\mathcal{O}^\epsilon_{\boldsymbol{k}}$ denote an oracle outputting independent samples according to the distribution $D^\epsilon_{\boldsymbol{k}}$. The LPN^n_ϵ problem is said to be (q, t, m, θ)-hard to solve if for any algorithm A,

$$\Pr[\boldsymbol{k} \leftarrow \{0,1\}^n : A^{\mathcal{O}^\epsilon_{\boldsymbol{k}}}(1^n) = \boldsymbol{k}] \leq \theta,$$

and A runs in time $< t$, with memory $< m$ and makes at most q queries to $\mathcal{O}^\epsilon_{\boldsymbol{k}}$.

We then introduce the LWE problem and its associated distribution. It is worth noting that the LPN problem is a special case of the LWE problem where the modulus q is equal to 2 and therefore χ is a Bernoulli distribution.

The standard "algebraically unstructured" LWE problem is specified as follows [Reg09a]:

Definition 6 (LWE problem)**.** *Let $n, q \in \mathbb{N}$, $\langle ., . \rangle$ denote the inner product and χ be a distribution over $\mathbb{Z}_q$. For a secret $\boldsymbol{s} \in \mathbb{Z}^n_q$, the LWE distribution $L_{\boldsymbol{s},\chi}$ is defined as:*

$$L_{\boldsymbol{s},\chi} =: \{(\boldsymbol{a}, \langle \boldsymbol{a}, \boldsymbol{s} \rangle + e) : \boldsymbol{a} \xleftarrow{\$} \mathbb{Z}^n_q; e \xleftarrow{\$} \chi\}.$$

Let $\mathcal{O}_{\boldsymbol{s},\chi}$ denote an oracle outputting independent samples according to the distribution $L_{\boldsymbol{s},\chi}$. The $LWE_{n,\chi}$ (search) problem is said to be (c, t, m, θ)-hard to solve if for any algorithm A,

$$\Pr[\boldsymbol{s} \xleftarrow{\$} \mathbb{Z}^n_q \ : A^{\mathcal{O}_{\boldsymbol{s},\chi}}(1^n) = \boldsymbol{s}] \leq \theta,$$

and A runs in time $< t$, with memory $< m$ and makes at most c queries to $\mathcal{O}_{\boldsymbol{s},\chi}$.

It has been shown that retrieving the secret vector $\boldsymbol{k}$ from public vectors $\boldsymbol{a}$ and noisy inner products is asymptotically hard [Reg09b].

Learning problems variants.

In PQC, the LWE problem is often replaced by its *structured* or *ring* variants. These structured versions operate on polynomials, typically in $\mathbb{R}_q$, or vectors of polynomials (such as in the case of MLWE, discussed below). In these cases, the inner product is replaced by polynomial multiplication. These variants are termed "structured" because polynomial multiplication in R_q can be viewed as an inner product over $\mathbb{F}_q$. In this case, the resulting public matrix, generated from a random polynomial, is not uniformly random but consists of negacirculant submatrices. While this structure has not yet led to more efficient attacks, it allows for polynomial operations to be more efficient than inner products, offering comparable security.

The Module LWE problem generalizes both the LWE and Ring LWE [LPR12] problems.

Definition 7 (MLWE problem [BGV12; LS12])**.** *Let $\mathbb{R}_q = \mathbb{Z}_q[x]/f(x)$ be the ring of integer polynomials modulo both $f(x)$ and q, where $f(x) = x^d + 1 \in \mathbb{Z}[x]$, the dimension of the ring d is a power of 2 and $q \in \mathbb{N}$. Let χ be a distribution over $\mathbb{R}_q$. For a secret $\boldsymbol{s} \in \mathbb{R}_q^n$, where n is called the rank of the module the MLWE distribution $M_{\boldsymbol{s},\chi}$ is defined as:*

$$M_{\boldsymbol{s},\chi} =: \left\{ \left(\boldsymbol{a}, \sum_{i=1}^{n} \boldsymbol{a}_i \cdot \boldsymbol{s}_i + e \right) : \boldsymbol{a} \xleftarrow{\$} R_q^n; e \xleftarrow{\$} \chi \right\},$$

where $\cdot$ denotes the multiplication in $\mathbb{R}_q$. Let $\mathcal{O}_{\boldsymbol{s},\chi}$ denotes an oracle outputting independent samples according to the distribution $M_{\boldsymbol{s},\chi}$. The $MLWE_{d,n,\chi}$ problem is said to be (c, t, m, θ)-hard to solve if for any algorithm A,

$$\Pr[\boldsymbol{s} \leftarrow U(R_q^n) \ : A^{\mathcal{O}_{\boldsymbol{s},\chi}}(1^{nd}) = \boldsymbol{s}] \leq \theta,$$

and A runs in time $< t$, with memory $< m$ and makes at most c queries to $\mathcal{O}_{\boldsymbol{s},\chi}$.

Note that LWE is the particular case of MLWE where $d = 1$ and the ring version of LWE (coined RLWE) is the particular case of MLWE where $n = 1$.

2.2.2 Hard physical learning problems.

Learning Parity with Physical Noise (LPPN)

The LPPN problem, introduced in [Kam+20b], is a variant of the LPN problem where a physical function directly computes incorrect inner products. Its specification requires the definition of physical function adapted from [Arm+11] which we recall next:

Definition 8 (Physical function)**.** *A physical function* $\mathsf{PF}_{d,\alpha}$ *is a probabilistic procedure based on a physical device* d*, which can be stimulated with an input* $\boldsymbol{x} \in \{0,1\}^{n_i}$*, making* d *respond with a (probabilistic) output* $\boldsymbol{y} \in \{0,1\}^{n_o}$*, with* α *a set of parameters.*

In the LPPN case, d will be an implementation of inexact inner product computations (implying $n_i = n$ and $n_o = 1$), which we define next, and α is the set of parameters determining its error distribution (e.g., the clock frequency or supply voltage).

Definition 9 (Physical inner product)**.** *Let* $\boldsymbol{k} \in \{0,1\}^n$ *be a random* n*-bit secret stored in a device* $d_{\boldsymbol{k}}$*. A physical function* $\mathsf{PF}_{d_{\boldsymbol{k}},\alpha}$ *is called an* ϵ*-Physical Inner Product (*ϵ*-PIP) if, on uniform public input* $\boldsymbol{x} \in \{0,1\}^n$*, it outputs* $\langle \boldsymbol{x}, \boldsymbol{k} \rangle$ *with estimated error probability:*

$$\hat{\Pr}[\mathsf{PF}_{d_{\boldsymbol{k}},\alpha}(\boldsymbol{x}) \neq \langle \boldsymbol{x}, \boldsymbol{k} \rangle] = \epsilon.$$

The LPPN problem can then be defined as follows:

Definition 10 (LPPN problem)**.** *Let* $\mathsf{PF}_{d_{\boldsymbol{k}},\alpha}$ *be an* ϵ*-PIP. The* $LPPN^{n,\epsilon}_{d_{\boldsymbol{k}},\alpha}$ *problem is said to be* (q,t,m,θ)*-hard to solve if for any algorithm* A *running in time* $< t$*, memory* $< m$ *and making at most* q *uniformly random queries to an* ϵ*-PIP, it holds that:*

$$\Pr[\boldsymbol{k} \leftarrow \{0,1\}^n : A^{\mathsf{PF}_{d_{\boldsymbol{k}},\alpha}}(1^n) = \boldsymbol{k}] \leq \theta.$$

The main difference between the LPN and LPPN problems is that assuming the LPN problem to be hard is a mathematical assumption, while assuming the LPPN problem to be hard is a physical assumption. In particular, we have no guarantee that the LPPN error distribution is exactly equal to the one specified in Definition 5. We show in Section 4.1 that concrete instances of physical functions can be affected by various defaults of the noise distribution, hence questioning the equivalence of these problems.

Learning With Physical Rounding (LWPR)

Fresh re-keying is a countermeasure against side-channel analysis where an ephemeral key is derived from a long-term key using a public random value. Popular instances of such schemes rely on key-homomorphic primitives, so that the re-keying process is easy to mask and the rest of the (e.g., block cipher) computations can run with cheaper countermeasures. The main requirement for these schemes to be secure is that the leakages of the ephemeral keys do not allow recovering the long-term key. The Learning with Physical Rounding (LWPR) problem formalizes

this security in a practically-relevant model where the adversary can observe noise-free leakages. It can be viewed as a physical version of the Learning With Rounding (LWR) problem [BPR12] where the rounding is performed by a leakage function and therefore does not have to be computed explicitly.

We formally define the LWPR assumption introduced in [Duv+21]. For this purpose, let us consider a secret matrix $\boldsymbol{K} \in \mathbb{F}_2^{m \times n}$ and a public vector $\boldsymbol{r} \in \mathbb{F}_2^n$. The work of Boneh et al. on cryptographic dark matter [Bon+18] showed that a low-complexity wPRF $\mathsf{F}_{\boldsymbol{K}}(\boldsymbol{r})$ can be obtained by computing the product $\boldsymbol{K} \cdot \boldsymbol{r}$, interpreting its output as a vector of 0/1 values over $\mathbb{F}_3$ and rounding it. Precisely:

$$\mathsf{F}_{\boldsymbol{K}}(\boldsymbol{r}) := \mathsf{map}(\boldsymbol{K} \cdot \boldsymbol{r}),$$

where the proposed mapping was the sum of this vector of 0/1 values modulo 3 [Bon+18]. The idea of Duval et al. is to replace this mathematical mapping by a physical one (the leakage function) that is formalized as follows.

First, it is assumed that the leaking device computes on binary-represented data: each value in $\mathbb{F}_p$ is therefore represented with (at least) $\lceil \log p \rceil$ bits. We denote as $\mathsf{g} : \mathbb{F}_p \to \{0,1\}^{\lceil \log p \rceil}$ the function associating to each element y of $\mathbb{F}_p$ the binary representation of its representative in $[0, p-1]$. We also define $\mathsf{g}_m : \mathbb{F}_p^m \to \{0,1\}^{m\lceil \log p \rceil}$ as: $\mathsf{g}_m(\boldsymbol{y}) := \mathsf{g}(\boldsymbol{y}_1)||\mathsf{g}(\boldsymbol{y}_2)||\ldots||\mathsf{g}(\boldsymbol{y}_m)$. This assumption is quite generic and captures the reality of most embedded computing devices deployed in current applications. Next, a more specialized assumption is required to define how the physical (noise-free) leakages depend on the $m\lceil \log p \rceil$ bits provided by g_m. This role will be played by the leakage function. We denote the leakage function computed on the binary representation of the manipulated data as $\mathsf{L}_{\mathsf{g}}(.)$. The generic LWPR problem is then defined as follows:

Definition 11 (Learning with physical rounding)**.** *Let $p, n, m \in \mathbb{N}^*$, p prime, for $\boldsymbol{K} \in \mathbb{F}_p^{m \times (n+1)}$. The $\mathrm{LWPR}^{n,m}_{\mathsf{L}_{\mathsf{g}},p}$ sample distribution is given by: $\mathcal{D}_{\mathrm{LWPR}^{n,m}_{\mathsf{L}_{\mathsf{g}},p}} := (\boldsymbol{r}, \mathsf{L}_{\mathsf{g}}(\boldsymbol{K} \boxdot \boldsymbol{r}))$ for $\boldsymbol{r} \in \mathbb{F}_p^n$ uniformly random, where $\boldsymbol{K} \boxdot \boldsymbol{r} = \boldsymbol{K} \cdot (\boldsymbol{r}, 1)$ and $\mathsf{L}_{\mathsf{g}} : \mathbb{F}_p^m \to \mathbb{R}^{n_d}$ is the physical rounding function. Given query access to $\mathcal{D}_{\mathrm{LWPR}^{n,m}_{\mathsf{L}_{\mathsf{g}},p}}$ for a uniformly random $\boldsymbol{K}$, the $\mathrm{LWPR}^{n,m}_{\mathsf{L}_{\mathsf{g}},p}$ problem is (q, τ, μ, ϵ)-hard to solve if after the observation of q LWPR samples, no adversary can recover the key $\boldsymbol{K}$ with time complexity τ, memory complexity μ and probability higher than ϵ.*

Duval et al. analyzed the case of Hamming weight leakages as a first step. The Hamming weight function is defined on any vector $\boldsymbol{v}$ of length

$t \in \mathbb{N}^*$ with coefficients in $\{0,1\}$ as $\mathsf{M}_{\mathsf{h}}(\boldsymbol{v}) = \sum_{i=1}^{t} \boldsymbol{v}_i$, where the sum is performed in $\mathbb{Z}$. They additionally considered two implementation contexts. In the serial case, the adversary is provided with m leakages on $\lceil \log p \rceil$ bits:

$$\mathsf{M}^s_{\mathsf{h}}(\boldsymbol{y}) : \boldsymbol{y} \mapsto \Big(\mathsf{M}_{\mathsf{h}}(\mathsf{g}(\boldsymbol{y}_1)), \mathsf{M}_{\mathsf{h}}(\mathsf{g}(\boldsymbol{y}_2)), \ldots, \mathsf{M}_{\mathsf{h}}(\mathsf{g}(\boldsymbol{y}_m))\Big).$$

In the parallel case, she receives the sum of these m values, which hardens the problem.

The LWPR instance proposed by Duval et al. uses a Mersenne prime $p = 2^{31} - 1$. It aims at the generation of 124-bit fresh keys in the parallel case and therefore considers $m = 4$. As for the main security parameter n, their analysis suggested $(n+1)\log(p) + 3\log(n) \geq 124$ as a lower bound. This led the authors to select $n = 4$ as a first target for further cryptanalytic investigations.

Related constructions

We now present a few illustrative use cases of LWE, MLWE, and RLWE problems, as utilized in recent post-quantum cryptographic primitives submitted to the NIST competition. More detailed descriptions of Kyber and Dilithium can be found in the sections dedicated to their analysis (Section 3.1for Kyber and Section 3.2 for Dilithium).

Frodo. The Frodo scheme [Alk+21] is tightly related to the hardness of a corresponding LWE problem. Its security depends on the lattice dimension n. Frodo-n with n equal to 640, 976, and 1344 targets different security levels specified in the NIST requirements. The main operations are simple matrix-vector products. The modulus q is always a power of two. This ensures that modular arithmetic operations do not require modulo reduction schemes such as Montgomery or Barrett. Instead, the reduction modulo q is computed by low-cost bit-masking or even at no cost if the data type of the target platform supports the number of bits required by the modulus.$q = 2^{15}$ or 2^{16}, depending on the targeted security level. Frodo requires to sample errors from an approximated discrete Gaussian distribution DGD_σ via lookup tables, where σ is the standard deviation (equal to 2.8, 2.3 and 1.4 for the different security levels). Frodo relies on the standard "algebraically unstructured" LWE problem and is therefore simple to implement. However, this simplicity comes at the cost of reduced efficiency (i.e., a high running time and large bandwidth).

NewHope. The NewHope cryptosystem [Alk+20] is based on the hardness of the RLWE problem. Its security depends on the polynomial ring

dimension d which has to be a power of two to maintain the security properties of RLWE and to support the efficient implementation of Number Theoretic Transform (NTT) algorithms (other rings allow efficient NTT, but are less commonly used). For bit security levels of 101 and 233, d is set to 512 and 1024, respectively. The modulus q is fixed to $12,289$ as it is the smallest prime for which it holds that $q \equiv 1 \mod 2d$ (so that the NTT can be performed efficiently). The noise in NewHope is sampled from a Central Binomial Distribution (CBD) with parameter $\eta = 8$, avoiding the aforementioned challenge to implement a discrete Gaussian sampler efficiently, and protecting against timing attacks. In contrast to Frodo, NewHope can be very efficient, both in terms of speed and in terms of key and ciphertext sizes. This is because it relies on a highly structured ideal lattice, though it is not easily scalable. For example, varying the hardness of an RLWE scheme requires the change of the ring dimension d which essentially means re-implementing all the operations.

CRYSTALS-Kyber. The security of CRYSTALS-Kyber [Ava+20] is based on the hardness of solving the MLWE problem. In all Kyber variants, the polynomial ring dimension is fixed to $d = 256$ and the security depends on the lattice dimension n. Kyber-$(d \times n)$ with n equal to 2, 3 and 4 targets Level 1, 3 and 5. The modulus $q = 3,329$ is a prime satisfying $q \equiv 1 \mod d$.[1] Similarly to the NewHope cryptosystem, the Kyber scheme uses the centered binomial distribution CBD_η as the noise distribution, with parameter η varying between 1 and 2 depending on the target security level. Since Kyber relies on the MLWE problem, it has a reduced structure compared to (RLWE-based) NewHope. However, it maintains similar performances (in the specific case of the MLWE parameters used in Kyber). Also, it offers better scalability because its security can be adapted by increasing the vector size k while keeping the ring modulus constant. As for NewHope, the need to implement fast polynomial multiplication algorithms to exploit the algebraic structure results in more complicated implementations than those of Frodo. Kyber has been selected as PQC standard KEM by NIST.

CRYSTALS-Dilithium. CRYSTALS-Dilithium[Duc+18] is a post-quantum signature scheme based on the Fiat-Shamir with aborts approach [Lyu09]. As CRYSTALS-Kyber [Ava+20], its security relies on the hardness of solving the MLWE problem. The signature scheme's

[1] The use of NTT algorithms requires that $q \equiv 1 \, mod \, 2d$. However, to reduce the bandwidth (by decreasing the modulus), version 2.0 of Kyber preprocesses the polynomial before applying the NTT so that a polynomial with degree d is divided into two low-dimensional polynomials of degree $d/2$.

implementation consists of symmetric primitives (SHAKE) and operations over a polynomial, which dimensions are given by the modulus $q = 8380417$ and $n = 256$. CRYSTAL-Dilithium is a third-round NIST candidate, its strengths are the small size of its signatures and public keys, and its simplicity of implementation compared to its rivals. CRYSTAL-Dilithium uses uniform noise in its instances of the MLWE problem in order to avoid the difficult implementation of Gaussian sampling, which can lead to side-channel attacks if performed incorrectly [Bru+16; Esp+17; PBY17]. However, using a physical rounded-Gaussian sampler as LWPE may benefit to Dilithium by allowing a secure implementation of a variant with Gaussian noise. Dilithium has been selected as a PQC standard DSA by NIST.

2.3 Side-channel attacks

This manuscript does not address the physical methods used to capture leakage. Instead, we focus on viewing leakage as a sequence of real values associated with the execution of an implementation. Each value corresponds to the internal variables processed by the device at discrete clock cycles and the transitions between those values. Practically, this leakage can be seen as a function of the intermediate variables, subject to a random noise, often modeled as a Gaussian error.

Since leakage is a physical phenomenon, accurately modeling it is extremely challenging. It depends not only on the specific implementation but also on the device it runs on and the physical conditions during execution (e.g., temperature, power supply fluctuations). However, it is sometimes beneficial to approximate and measure leakage using certain metrics. One approach is through information-theoretic metrics, as discussed below. Another method is to approximate the leakage with a simple function: for example, one common approximation models the leakage of a value x as $HW(x) + e$, where HM denotes the Hamming weight and e is a Gaussian error. More detailed discussions of leakage models and correlations with actual measurements can be found in Section 4.3.

All side-channel attacks (whether based on a divide-and-conquer or analytical strategy) include an extraction phase where they collect information about guessable intermediate computations. A standard strategy to estimate the number of traces required to recover leaking target intermediate values with confidence is to use information theoretic metrics such as the Mutual Information (MI) [SMY09; DFS15; Ché+19]. Next, we recall the simple relations that we are going to leverage to derive

shortcut formulas. We start with unprotected variables, continue with masked variables, and finally discuss the impact of attacks exploiting multiple intermediate computations.

Unprotected variables. To evaluate the number of independent leakages N required to recover a (sub)-secret X, one can use the relation:

$$N \approx \frac{c}{\mathtt{MI}(X; \boldsymbol{L})}\,, \tag{2.1}$$

where c is a small constant that depends on the size of the intermediate variable and the target success rate (we will set it to Shannon's entropy such that $c = \mathrm{H}(X)$ for simplicity), and $\mathtt{MI}(X; \boldsymbol{L})$ is the Mutual Information between the target variable X and the leakage vector $\boldsymbol{L}$.

Roughly, the attack complexity is inversely proportional to the MI, which is itself inversely proportional to the noise variance of the leakages [MOS11]. In the following, we will use the MI between the target intermediate computations and their leakage as a parameter of our evaluations, and denote it as λ.

Masked variables. To protect implementations against SCA, masking is a countermeasure that has been extensively studied. It consists in representing a sensitive variable X as d shares $(X^0, X^1, \ldots, X^{d-1})$ such that any set of $d-1$ shares remains independent of X. The operations are then applied to the shares of X instead of on X directly. When implemented securely (i.e., under some noise and independence conditions), it guarantees an exponential security increase at the cost of quadratic performance overheads [Cha+99; ISW03; PR13; DDF14; DFS15]. Concretely, this security amplification is reflected by a reduction of the MI:

$$\mathtt{MI}(X; \boldsymbol{L}) \approx \prod_{i=0}^{d-1} \mathtt{MI}(X^i; \boldsymbol{L}) = \lambda^d,$$

leading to an increas in attack data complexity captured by:

$$N \approx \frac{c}{\lambda^d}\,.$$

Splitting a value in d independent shares can be achieved in different ways. In PQC, most value can be seen either as arithmetic variables (e.g., polynomial with coefficients in $\mathbb{F}_q$) or bit sequences. For these, the most efficient maskings are either boolean or arithmetic.

Boolean masking consists in splitting a variable x into shares $x_0, \ldots x_{d-1}$ such that $x = x_0 \oplus \ldots \oplus x_{d-1}$. This method is particularly useful when

the variable is involved in symmetric operations, such as in XOFs or hash functions. Arithmetic masking consists in splitting a variable x into shares $x_0, \dots x_{d-1}$ such that $x = x_0 + \dots + x_{d-1}$, where $+$ denotes the addition over $\mathbb{F}_q$. This type of masking is well-suited for implementing masked arithmetic operations, like polynomial multiplication over $\mathbb{R}_q$.

Since PQC schemes often combine both Boolean-friendly and arithmetic-friendly operations, they require masking conversions between representations (either Boolean to arithmetic or arithmetic to Boolean). These representations and conversions are further discussed in Section 3.2.

Targeting multiple (independent) operations. Side-channel attacks are not limited to the exploitation of a single intermediate computation and its corresponding leakage. Multiple operations can be exploited jointly. The simplest way to do so is when multiple intermediate computations relate to the same guessable secret. In this case, a natural abstraction is to model their leakages as providing independent information on the secret, as captured by the Independent Operations' Leakages (IOL) proposed in [GS18]. It leads to the approximation:

$$\mathtt{MI}(X; \boldsymbol{L}) \approx \sum_j \mathtt{MI}(var_j(X); \boldsymbol{L}) \approx \#var \cdot \lambda^d,$$

where $var_j(X)$ is one of the $\#var$ variables depending only on X.

For example in the case of block-ciphers, an attack can be performed by exploiting leakage (e.g., with Gaussian templates) at the input of the Sbox and its output and combining information from both. This will double the information about X (hence $\#var = 2$) compared to an attack exploiting only its output (with $\#var = 1$). It turns out this simple setting will be quite frequently observed (and generalized) in our following investigations.

Cautionary notes. The above formulas are admittedly simplified and may ignore a part of the leakages. First, it is also possible to exploit the leakages of operations that are not exploitable via a divide-and-conquer DPA, for example using analytical strategies [VGS14], as considered by [PPM17; PP19] in the PQC setting. Second, masked multiplications with d shares imply quadratic overheads, e.g., they require computing d^2 partial products which may leak as well. Yet, as modeled in [Guo+20], the leakage of intermediate values that do not relate to the same guessable secret does not simply add up as in the previous IOL case and rather implies a constant gain in the attacks; and the leakage of partial products in masked multiplication is dominated by the leakage of

the tuples as the noise level of the implementation increases. So simply stated, while the following shortcut formulas could be refined to take into account advanced attacks, they are a convenient first step to study the large design space of masked PQ KEMs up to small constants.

3 Physical security of post-quantum schemes

In this chapter, we study post-quantum side-channel security at a scheme-level. Through three different schemes, we cover various aspects of the study of post-quantum schemes such as side-channel sensitivity analysis, physically secure implementation on micro-controllers or quantum-safe and leakage-resistant scheme design. Each section of this chapter is based on a peer-reviewed publication that I co-authored.

In Section 3.1, we study the Fujisaki Okamoto transform using the NIST standard KEM CRYSTALS-Kyber as an example. Many side-channel attacks have been published against post-quantum schemes such as [Rav+20b; Xu+20], the ones targeting the FO-transform [FO99] re-encryption step appearing to be very efficient [Uen+21]. DPAs against the decryption step are also possible [PPM17; PP19]. We propose a systematic and quantitative investigation of their impact for designers, using shortcut formulas in order to compare their respective strength in function of the noise level. Taking the case of Kyber for illustration, we then evaluate the (high) cost of preventing them with masking and the extent to which different parts of an implementation could benefit from varying security levels. We finally discuss tweaks to improve the situation and enable a better leveling of the countermeasures. Our conclusions confirm that current solutions for side-channel secure PQ key encapsulation schemes like Kyber are unlikely to be efficient in low-noise settings without (design or countermeasures) improvements.

We then continue in Section 3.2 by studying another NIST standard, the DSA CRYSTALS-Dilithium. We revisit the side-channel countermeasures of Dilithium in three directions. First, we improve its sensitivity analysis by classifying intermediate computations according to their physical security requirements. Second, we provide improved gadgets dedicated to Dilithium, taking advantage of recent advances in masking conversion algorithms. Third, we combine these contributions and report performance for side-channel protected Dilithium implementation. Our benchmarking results additionally put forward that the randomized

version of Dilithium can lead to significantly more efficient implementations (than its deterministic version) when side-channel attacks are a concern.

Finally, in Section 3.3, we describe a new CCA-secure quantum-safe KEM named `POLKA`. As we have seen in Section 3.1, implementions of post-quantum schemes that have not been developed with leakage in mind can become more expensive by orders of magnitude. On the other hand, we made `POLKA` design choices considering the side-channel perspective. It leverages various ingredients in order to enable efficient side-channel protected implementations such as:

(i) the rigidity property (which intuitively means that de-randomized encryption and decryption are injective functions) to avoid the very leaky re-encryption step of the Fujisaki-Okamoto transform,
(ii) the randomization of the decryption thanks to the incorporation of a dummy ciphertext, removing the adversary's control of its intermediate computations and making these computations ephemeral,
(iii) key-homomorphic computations that can be masked against side-channel attacks with overheads that scale linearly in the number of shares,
(iv) hard physical learning problem to argue about the security of some critical unmasked operations.

Furthermore, we use an explicit rejection mechanism (returning an error symbol for invalid ciphertexts) to avoid the additional leakage caused by implicit rejection. As a result, all the operations of `POLKA` can be protected against leakage in a much cheaper way than state-of-the-art designs, with a minimal overhead in a black-box framework, opening the way towards schemes that are both quantum-safe and leakage-resistant.

3.1 Leakage analysis: the case of Kyber, adapted from [Azo+22b]

Many Post-Quantum (PQ) Key Encapsulation Mechanisms (KEMs), including third-round finalists of the NIST post-quantum standardization effort, rely on the Fujisaki-Okamoto (FO) transform [FO99]. It allows building a Chosen-Ciphertext (CCA) secure scheme from a Chosen-Plaintext (CPA) secure Public-Key Encryption (PKE) scheme. This transform first decrypts the ciphertext c with the underlying CPA-secure PKE to retrieve the message m. Then, it re-encrypts (in a deterministic manner) m to obtain a ciphertext c'. By construction, any ciphertext c that has not been generated by the CPA-secure encryption scheme will

result in a case where $c' \neq c$ (up to a negligible probability). In such a case, the CCA-secure KEM returns a random message that cannot be exploited by the adversary. Yet, while the FO transform is well suited to reach black-box security, several recent works showed that the situation strongly differs when physical attacks are considered [Rav+20b; Xu+20; Uen+21; Ngo+21]. In such a context, leakage about intermediate computations makes it possible to circumvent black-box security guarantees. Roughly, an adversary can then use a chosen-ciphertext attack against the part of the scheme that is only CPA secure. For example in [Rav+20b], the adversary carefully crafts ciphertexts such that the decrypted message m leaks a bit of a secret key coefficient. Since this message m is used as input for the deterministic re-encryption, the adversary then only has to distinguish between an encryption of 0 or 1 given leakage of the computation. To do so, she can target all the intermediate computations within the (long) deterministic re-encryption jointly, which can include hundreds, thousands, or even millions of intermediate bytes/words. Furthermore, this leakage is easy to exploit, whether via standard techniques (e.g., template attacks with dimensionality reduction) or machine learning-based cryptanalysis.

Echoing the situation in symmetric cryptography, such an attack corresponds to the strongest (state comparison) one in the taxonomy of [Sta19, Slide 1.7]. In terms of design, [Sta19] also recalls that symmetric decryption ensuring CCA security with leakage requires a two-pass design where the validity of the ciphertexts is verified before being decrypted (e.g., thanks to a MAC).

In addition to these attacks targeting the re-encryption, Differential Power Analysis attacks (DPAs) targeting leakage in the first (guessable) parts of a secret computation are also possible. In the case of PQ KEMs, such attacks naturally apply to the part of the decryption that takes place before the re-encryption. As usual, they can be extended from a standard DPA to analytical attacks exploiting even the hard-to-guess parts of the computation thanks to belief propagation, leading to strong key recoveries [PPM17; PP19].

In parallel to the efforts regarding the identification of attack vectors, countermeasures against side-channel attacks have also been adapted to the PQ setting. As one of the leading protections, masking has received particular attention and, for example, first- and higher-order masked implementations of Saber and Kyber have been proposed [Bei+21; Bos+21; Hei+21; Fri+21]. To the best of our knowledge, masked implementations of PQ schemes so far mostly considered a uniform protection level, where all the parts of the computations embed the same num-

ber of shares. Again echoing the situation in symmetric cryptography, these works naturally question the possibility of considering so-called leveled implementations where different parts of the computations have different security levels (for example, based on the number of operations exploitable via side-channel leakage [Bel+20]).

Based on this state-of-the-art, the objectives of this section are threefold:

First, we propose a model for both (i.e., SPA and DPA) attack paths. For each of them, we derive a shortcut formula (i.e., a generic expression for the minimal number of traces needed for a successful attack) that takes as parameters the number of shares in the masking scheme, the level of noise in the leakage (measured as the mutual information between the shares and the leakage λ) and the (cipher-specific) amount of operations for which the leakage is exploitable. These shortcut formulas enable a first (simplified) approximation of attack complexities to compare side-channel attacks according to the above parameters.

Second, we illustrate these formulas in the example of CRYSTALS-Kyber [Ava+19]. This specific choice of KEM is motivated by the already large literature dedicated to its side-channel analysis and countermeasures. We derive our model parameters based on state-of-the-art implementation results from [Bos+21] . To enable a comparative study of the attack paths, we additionally express the cost needed to protect the different subparts of the Kyber computations.

Third, we use our results to discuss masked implementations of Kyber and the possibility of leveraging the leveled implementation concept for PQ KEMs. We show that for unmasked implementations, the re-encryption becomes the preferred attack vector as the noise level increases (since it enables very strong horizontal attacks). But somewhat surprisingly, we also show that as the number of shares (hence, the target security level) increases, the impact of the re-encryption in the overall security versus efficiency tradeoff tends to vanish, which significantly limits the interest in leveled implementations.

In conclusion, we first recall that getting rid of the FO-transform in a PQ KEM such as Kyber would require a way to identify "well structured ciphertexts" without re-encryption, as performed in the symmetric cryptographic setting thanks to a leakage-resilient MAC. We note that this is a hard problem in itself. At this stage, it is not clear whether simple filtering heuristics can be sufficient [Xu+20], and the more formal solution of relying upon a zero-knowledge proof (which we briefly discuss in Subsection 3.1.4) is quite expensive. In this respect, our results show that the performance margin available to (heuristically or formally) get rid of the FO-transform is quite limited. This conclusion

derives from the observation that as the number of shares increases, the cost of protecting the (CPA-secure) decryption and the re-encryption becomes increasingly balanced in Kyber. In other words, improving the efficiency of side-channel secure Kyber implementations would not only require to get rid of the FO-transform, but also to increase the performance gap between its protected (CPA-secure) decryption and re-encryption. Candidates for this purpose include improving the efficiency of this decryption (especially the compression part), reducing the complexity of its masking and taking advantage of hardware tweaks (e.g., exploiting different noise levels in the decryption and re-encryption). We believe a similar challenge appears in related PQ KEMs relying on the FO-transform (e.g., Saber [Bas+19]).

3.1.1 Background

We give here a short description of CRYSTALS-Kyber, as well as performances of tis latest masked implementation.

Arithmetic & notations. We remind that polynomial ring $\mathbb{R}_q$ is $\mathbb{Z}_q[X]/(X^n+1)$. We denote polynomials with lower case such that $f \in \mathbb{R}_q$. As a result, the polynomials are of the form:

$$f = f_0 + f_1 \cdot X^1 + \ldots + f_{n-1} \cdot f^{n-1}, \tag{3.1}$$

where $f_i \in \mathbb{Z}_q$ is the i-th coefficient of the polynomial. We denote vectors and matrices with bold letters. For example, $\boldsymbol{v} \in \mathbb{R}_q^k$ is a vector of k polynomials such that $\boldsymbol{v}[i] \in \mathbb{R}_q$ for all $0 \leq i < k$. Hence, $\boldsymbol{v}[i]_j$ refers to the j-th coefficient of the i-th polynomial in $\boldsymbol{v}$. Additionally, $\boldsymbol{A} \in \mathbb{R}_q^{k \times k}$ is a matrix of polynomials with size $k \times k$. For CRYSTALS-Kyber, the prime q is chosen such that polynomial multiplications can be performed very efficiently via the Number-Theoretic Transform (NTT). Concretely, the NTT is first applied to the polynomials such that:

$$\hat{f} = \mathrm{NTT}(f) = \hat{f}_0 + \hat{f}_1 \cdot X^1 + \ldots + \hat{f}_{n-1} \cdot X^{n-1}, \tag{3.2}$$

where $\hat{f}$ (resp., $\hat{f}_0$) denotes the NTT domain representation of f (resp., f_0). $\texttt{NTT}(\cdot)$ is efficiently applied with a butterfly algorithm [CT65; GS66]. The multiplication between two polynomials can then leverage their representations in the NTT domain such that:

$$c = a \cdot b = \mathrm{NTT}^{-1}(\hat{a} \circ \hat{b}), \tag{3.3}$$

where $\circ$ is the coefficient-wise multiplication.

CRYSTALS-Kyber PKE. We first detail the underlying CPA-secure PKE scheme denoted as Kyber.CPAPKE, which relies on the hardness of Module-LWE [LS15]. Kyber.CPAPKE encryption and decryption are recalled in Algorithm 1 and Algorithm 2, respectively. There, $\hat{\boldsymbol{s}}$ denotes a secret key, pk a public key containing a vector of polynomials $\hat{\boldsymbol{t}}$ and a random matrix of polynomials $\hat{\boldsymbol{A}}$, m the 256-bit message, σ a 256-bit random seed used to derive deterministically the randomness from a pseudo-random generator and $\texttt{CBD}_\eta$ denotes the sampling, from a uniform random string, of a centered binomial distribution with parameter η. $\texttt{Compress}_q$ and $\texttt{Decompress}_q$ are respectively compression and decompression functions such that $\texttt{Decompress}_q(\texttt{Compress}_q(x, d_x), d_x) \approx x$. Both PRF, KDF, G and H are hash functions with various output lengths based on the Keccak permutation. For more details about these algorithms, we refer to their specifications in [Ava+19].

Algorithm 1
Kyber.CPAPKE.Enc(pk,m,σ)

Input: Public key $pk := (\hat{\boldsymbol{t}}, \hat{\boldsymbol{A}})$ with $\hat{\boldsymbol{t}} \in \mathbb{R}_q^k$ and $\hat{\boldsymbol{A}} \in \mathbb{R}_q^{k \times k}$, message $m \in \{0,1\}^n$, random coin $\sigma \in \{0,1\}^{256}$.
Output: Ciphertext $c := (c_1, c_2)$.

for i in $[0, \ldots, k-1]$ **do**
  $\boldsymbol{r}[i] := \texttt{CBD}_{\eta_1}(\texttt{PRF}(\sigma, i))$
  $\boldsymbol{e}_1[i] := \texttt{CBD}_{\eta_2}(\texttt{PRF}(\sigma, i+k))$
$e_2 := \texttt{CBD}_{\eta_2}(\texttt{PRF}(\sigma, 2 \cdot k))$
$\hat{\boldsymbol{r}} := \texttt{NTT}(\boldsymbol{r})$
$\boldsymbol{u} := \texttt{NTT}^{-1}(\hat{\boldsymbol{A}}^T \circ \hat{\boldsymbol{r}}) + \boldsymbol{e}_1$
$v := \texttt{NTT}^{-1}(\hat{\boldsymbol{t}}^T \circ \hat{\boldsymbol{r}}) + e_2 + \texttt{Decompress}_q(m, 1)$
$c_1 := \texttt{Compress}_q(\boldsymbol{u}, d_u)$
$c_2 := \texttt{Compress}_q(v, d_v)$

return (c_1, c_2)

Algorithm 2
Kyber.CPAPKE.Dec($\hat{\boldsymbol{s}}$,c)

Input: Secret key $\hat{\boldsymbol{s}} \in \mathbb{R}_q^k$, ciphertext $c := (c_1, c_2)$.
Output: Message m.

$\boldsymbol{u} := \texttt{Decompress}_q(c_1, d_u)$
$v := \texttt{Decompress}_q(c_2, d_v)$
$\hat{z} = \hat{\boldsymbol{s}}^T \circ \texttt{NTT}(\boldsymbol{u})$
$w := v - \texttt{NTT}^{-1}(\hat{z})$
$m := \texttt{Compress}_q(w, 1)$

return m

CRYSTALS-Kyber KEM & FO-transform. CRYSTALS-Kyber leverages the FO-transform [FO99; HHK17b] to build a CCA-secure KEM from a CPA-secure PKE. The resulting Kyber.CCAKEM encapsulation and decapsulation algorithms are described respectively in Algorithm 3 and Algorithm 4. More precisely, Kyber.CCA KEM.Dec first retrieves a message candidate m' by decrypting the ciphertext c thanks to Kyber.CPAPKE.Dec and the secret key. Then, it computes c' by re-encrypting m' using the encryption $c' = \text{Kyber.CPAPKE.Enc}(pk, m', \sigma')$. It then only outputs the correct symmetric key material K if c is equal to c'.The resulting scheme is protected against chosen-ciphertext attacks:

as soon as an adversary attempts to use a forged ciphertext, the resulting re-encrypted c' will differ from c (up to a negligible probability).

We note that for such a construction to be correct, the randomness used during the (re)-encryption should be deterministically generated from a random coin associated with the plaintext. Otherwise, the ciphertexts c and c' could differ even though they are both correct ciphertexts of the same message m.

Algorithm 3
Kyber.CCAKEM.Enc(pk)

Input: Public key $pk := (\hat{\boldsymbol{t}}, \hat{\boldsymbol{A}})$
Output: Ciphertext c, encap. secret K.

1: $m \leftarrow \{0,1\}^{256}$
2: $m := \mathtt{H}(m)$
3: $(\bar{K}, \sigma) := \mathtt{G}(m||\mathtt{H}(pk))$
4: $c :=$ Kyber.CPAPKE.Enc(pk, m, σ)
5: $K := \mathtt{KDF}(\bar{K}||\mathtt{H}(c))$
6:
7: **return** (c, K)

Algorithm 4
Kyber.CCAKEM.Dec(c, sk)

Input: Ciphertext $c := (c_1, c_2)$, secret key $sk := (\hat{\boldsymbol{s}}, pk, \mathtt{H}(pk), z)$.
Output: Decap. secret K.

1: $m' :=$ Kyber.CPAPKE.Dec$(\hat{\boldsymbol{s}}, c)$
2: $(\bar{K}', \sigma') := \mathtt{G}(m'||\mathtt{H}(pk))$
3: $c' :=$ Kyber.CPAPKE.Enc(pk, m', σ')
4: **if** $c = c'$ **then**
5:
6: **return** $K := \mathtt{KDF}(\bar{K}'||\mathtt{H}(c))$
7: **else**
8:
9: **return** $K := \mathtt{KDF}(z||\mathtt{H}(c))$

Concrete parameters and masked implementation costs. We give the parameters used by CRYSTALS-Kyber for the third round of the NIST PQC standardization process [Ava+19] in Table 3.1, for the different versions corresponding to different security levels.[1]

	n	k	q	η_1	η_2	d_u	d_v
Kyber512	256	2	3329	3	2	10	4
Kyber768	256	3	3329	2	2	10	4
Kyber1024	256	4	3329	2	2	11	5

Table 3.1: Summary of Kyber parameters (from [Ava+19]).

In Table 3.2, we show the performance values for masked Kyber.CCAKEM.Dec for different masking orders based on [Bos+21] but with an optimized comparison. These values are provided for the STM32F407G (ARM Cortex-M4) MCU and up to order 6, since the maximum stack size on

[1] Namely, NIST security levels 1, 3 and 5 for which any attack requires comparable computational resources to attacks against AES-128,AES-196 and AES-256 [Ala+19; Ava+19].

the target MCU is 112 KiB (for SRAM1). Starting from *2nd* order, the implementation requires more than 13.5 KiB of stack size for each subsequent order. The *7th* order masked code requires at least 114 KiB of stack.

Operation	**Order**					
	1st	*2nd*	*3rd*	*4th*	*5th*	*6th*
`crypto_kem_dec`	3 178	57 141	97 294	174 220	258 437	350 529
`indcpa_dec`	200	4 203	7 047	13 542	20 323	27 230
`unpack`	24	30	36	43	50	56
`poly_arith`	89	119	148	598	713	790
`compression`	87	4 054	6 863	12 901	19 561	26 384
`indcpa_enc`	2 024	18 879	32 594	53 298	75 692	104 191
`decompression`	118	291	537	889	1 267	1 745
`gen_at`	391	391	391	391	391	391
`poly_getnoise`	1 217	17 727	31 069	49 390	70 856	98 435
`prf`	706	11 483	19 577	30 318	43 541	60 640
`cbd`	510	6,243	11 492	19 071	27 314	37 794
`poly_arith`	302	453	603	2 627	3 172	3 643
`ntt`	66	99	131	585	670	817
`invntt`	70	105	140	1 008	1 274	1 410
`comparison`	693	32 293	54 725	102 922	156 075	210 518
`hashg (SHA3-512)`	98	1 639	2 801	4 489	6 456	8 794
`hashh (SHA3-256)`	113	113	113	113	113	113
`kdf`	13	13	13	13	13	13

Table 3.2: Performance numbers for masked Kyber.CCAKEM.Dec and its subroutines (values are in kCycles).

3.1.2 Shortcut formulas for SPA and DPA

As stated in the introduction of Section 3.1, there exist two main attack paths against PQ KEMs based on the FO-transform. In this section, we derive shortcut formulas for their data complexity. For this purpose, we use the illustration of Figure 3.1, which is a simplified graphical representation of Algorithm 4.

$\mathcal{A}^{sk}_{\mathrm{DPA}}$: DPA against CPAPKE.Dec

Description. A DPA attack against PQ KEMs can be performed in a similar way as classical (divide-and-conquer) DPA against block ciphers. In this case, $\mathcal{A}^{sk}_{\mathrm{DPA}}$ targets directly each of the secret key coefficients $\hat{s}_i$

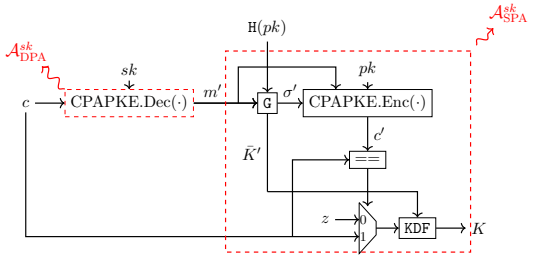


Figure 3.1: Attack paths against Kyber.CCAKEM.Dec(c, sk) (Algorithm 4).

independently. To do so, she asks for the decryption of legit ciphertexts c's and records the corresponding traces. She then exploits every operation that directly involves $\hat{s}_i$, as she would target the Sbox's input and output in the case of a block cipher.

For illustration, in the case of CRYSTALS-Kyber, this adversary targets each $\hat{\boldsymbol{s}}_i$ independently by observing the leakage of $\hat{\boldsymbol{s}}^T \circ \texttt{NTT}(\boldsymbol{u})$ (see Algorithm 2, line 3) where $\boldsymbol{u}$ is known and random. Indeed, she obtains leakage of the coefficient-wise operations $\hat{\boldsymbol{s}}_i \circ \hat{\boldsymbol{u}}_i$ as detailed in Equation 3.3.

Shortcut formula. Because this attack targets each of the coefficients independently, the resulting shortcut formula can leverage the approximations given in Section 2.3. Namely, we can estimate the data complexity of $\mathcal{A}^{sk}_{\text{DPA}}$ with:

$$N_{Dec} \approx \frac{\alpha_{Dec}}{\lambda^{d_{Dec}}}\,, \tag{3.4}$$

where d_{Dec} is the amount (#) of shares used to protect the CPAPKE.Dec and $\alpha_{Dec} = \frac{c}{\#var}$ is the cipher-dependent constant reflecting the # of operations that can be exploited via DPA (and the constant c), discussed for Kyber in Subsection 3.1.4.

Other DPA attacks. As already mentioned in Section 2.3, advanced adversaries could additionally exploit the leakages in $\texttt{NTT}^{-1}$ thanks to belief-propagation [PPM17; PP19]. By exploiting the leakage of hard-to-guess intermediate variables, such advanced attacks can reduce the data complexity by additional constant factors, which could be reflected by adapting α_{Dec} [Guo+20].

$\mathcal{A}^{sk}_{\mathrm{SPA}}$: SPA against re-encryption

Description. The SPA adversary $\mathcal{A}^{sk}_{\mathrm{SPA}}$ targets leakages from all operations that directly depend on the decrypted message m' [Rav+20b; Xu+20; Uen+21; Ngo+21]. In short, these attacks are chosen-ciphertext attacks against the part of the KEMs that are only CPA-secure. This is made possible despite the FO-transform by exploiting the leakage of m' as a plaintext-checking oracle. This can be done simply by observing whether m' is equal or not equal to a reference message. To recover the full secret key sk, multiple forged ciphertexts (and corresponding oracle accesses) are required. We denote their number as $\#\mathcal{O}$.

Shortcut formula. To predict the data complexity of such an SPA adversary, we first observe that by targeting the leakage of the re-encryption, it is possible to exploit multiple independent variables that directly depend on m'. We denote the number of these operations as $\#var$. Again leveraging the approximations of Section 2.3, the number of traces required to succeed in detecting whether m' is equal to the reference is then estimated as:

$$N_{Enc} \approx \frac{\alpha_{Enc}}{\lambda^{d_{Enc}}}\,, \tag{3.5}$$

with $\alpha_{Enc} = \frac{c' \cdot \#\mathcal{O}}{\#var}$. Here again, the number of exploitable operations $\#var$ is cipher-specific. The same holds for the number of required oracle accesses $\#\mathcal{O}$. Those values will be specified for the case of Kyber in the next section.

3.1.3 Generic intuitions

In this section, we discuss the case of masked implementations of PQ KEMs. We assume a designer aiming at an implementation that can resist attacks of complexity γ (which therefore corresponds to the security level) on a platform with physical security parameter λ (which, as outlined in Section Section 2.3, can be viewed as a measure of the implementation's noise level). More precisely, we first study the case where a designer selects the same number of shares to protect the decryption and re-encryption, and show that such a strategy requires a large number of shares. Then, we quantify the effect of using different number of shares for decryption and re-encryption and discuss its impact on the relative cost of the re-encryption in the overall implementation.

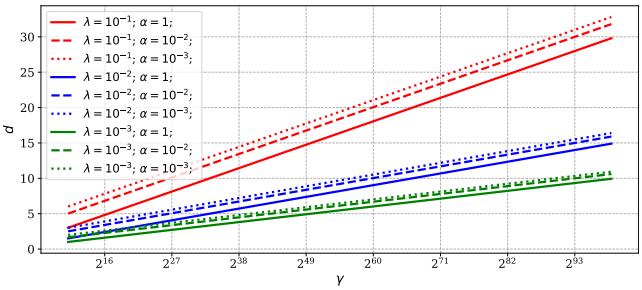


Figure 3.2: Impact of the attack parameters α and the noise level λ on the number of shares d in function of the target security level γ.

Masking can be (very) expensive

Next, we estimate the number of shares required to protect against both decryption and re-encryption attacks such that $d = d_{Enc} = d_{Dec}$. To do so, the target security level γ should be smaller than the attack complexities N derived in Subsection 3.1.2 such that:

$$\gamma \leq \alpha \cdot \frac{1}{\lambda^d}, \tag{3.6}$$

with $\alpha = \min(\alpha_{Enc}, \alpha_{Dec})$. As a result, the required number of shares to protect the entire implementation is defined by:

$$d \geq \log_\lambda \left(\frac{\alpha}{\gamma}\right) \tag{3.1}$$

$$\geq \frac{\log_{10}(\alpha) - \log_{10}(\gamma)}{\log_{10}(\lambda)}, \tag{3.2}$$

up to $\lceil \cdot \rceil$ to ensure that d is an integer (for readability, we omit the rounding for the rest of the section). Based on this equation, we report the number of shares for various settings in Figure 3.2. As expected from Equation 3.2, the parameter that has the strongest influence on the number of shares is the noise λ, which corresponds to the slope of the curves. For example, moving from $\lambda = 0.1$ to $\lambda = 0.01$ reduces the number of shares by a factor 2 since $|\log_{10}(\lambda)|$ is increased similarly. The number of shares also depends on the attack parameter α, but to a lower extent (since this factor is constant in d). For example, reducing α by a factor 10 implies the need for $1/\log_{10}(\lambda)$ additional shares to compensate.

The combination of these two effects is at the basis of the somewhat paradoxical conclusions in the following sections. On the one hand, a

large number of operations (which are observed for PQ KEMs) make side-channel attacks very strong in the absence of countermeasures. On the other hand, the impact of reducing this number of leaking operations vanishes as the target security level (and number of shares) increases, since it is then increasingly dominated by the amplified noise. For the rest, the figure also recalls that with low-noise devices, the number of shares needed to reach high security against side-channel attacks, and therefore the implementation overheads, can be prohibitive [BS21].

Leveling moderately helps

State-of-the-art analyses of PQ KEMs indicate that SPA attacks targeting re-encryption are very powerful. This is for example witnessed by the "curse of re-encryption" discussed in [Uen+21]. Concretely, the origin of this curse mainly lies in the different number of exploitable operations that decryption and re-encryption may have. A natural target to mitigate this issue is to use a different number of shares for these two parts of a KEM. By doing so, we could expect to reduce the overall cost of implementation by protecting its stronger (i.e., CPA decryption) part. To model such a leveling, we assume that masking leads to overheads that are quadratic in d and express the cost of protecting one component as $\zeta = \beta \cdot d^2$, where β is the cycle count of an unprotected implementation. Note that the quadratic cost overheads are naturally dependent on the fraction of linear and non-linear operations in the algorithm to protect. But concrete results recalled in Table 3.2 show that this trend is observed even for a low number of shares in the case of Kyber.[2] More precisely, we are interested in the proportion ζ_{Enc}/ζ_{Dec} of time that is spent to protect re-encryption and decryption. A large ratio means that the re-encryption is the dominating the overall cost. Hence, removing re-encryption would lead to significant performance improvements in this case. A small ratio means the opposite. Next, we study this ratio according to attack parameter α, target security level γ as well as the

[2]For both CPAPKE.Enc and CPAPKE.Dec, lattice arithmetic can be efficiently masked with linear complexity. However, both include polynomial comparison, polynomial compression and binomial noise sampling which require boolean-to-arithmetic and/or arithmetic-to-boolean masking conversions implying quadratic overheads. In the case of Kyber, these conversions dominate the overall cost of protected implementations even for small masking orders.

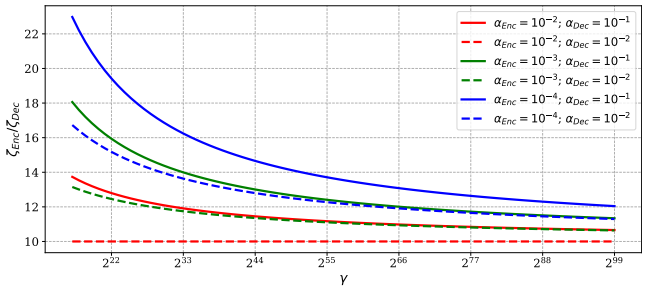


Figure 3.3: Ratio between the cost of decryption and re-encryption ($\beta_{Enc}/\beta_{Dec} = 10$).

implementation complexity β.[3] This proportion can be given by:

$$\zeta_{Enc}/\zeta_{Dec} = \frac{\beta_{Enc}}{\beta_{Dec}} \cdot \frac{d_{Enc}^2}{d_{Dec}^2} \tag{3.1}$$

$$= \frac{\beta_{Enc}}{\beta_{Dec}} \cdot \frac{(\log_{10}(\alpha_{Enc}) - \log_{10}(\gamma))^2}{(\log_{10}(\alpha_{Dec}) - \log_{10}(\gamma))^2}\,, \tag{3.2}$$

thanks to the expression of the number of shares in Equation 3.2. For increasing security target γ, this ratio converges such that:

$$\lim_{\gamma \to \infty} \zeta_{Enc}/\zeta_{Dec} = \frac{\beta_{Enc}}{\beta_{Dec}}\,. \tag{3.1}$$

We observe that as γ increases, the proportion of time spent between the two blocks converges towards the ratio of their performances when implemented without side-channel protections. In other words, the gap between the number of shares needed to compensate different numbers of exploitable operations vanishes with the security level. This equation is illustrated in Figure 3.3. For low security, the re-encryption dominates the overall cost of the secure implementation. By contrast, it is no longer the case for large security levels.

Interestingly, even if the idea of leveling we analyze in this work looks quite similar to the one leveraged in symmetric cryptography [Bel+20], its impact is much lower. The best factor that a designer could hope to gain in the case of a PQ KEM that is strongly protected against side-channel attacks is $\approx 1 + \beta_{Enc}/\beta_{Dec}$. But in fact, the explanation is also quite natural. The main factor making the leveling process very effective in the symmetric case is the possibility to amortize the cost of a highly masked implementation for long messages. In the case of

[3] β is strongly connected to α in the case of PQ KEMs where the leakage of most operations can be exploited. Yet, it could be more different in other cases.

a PQ KEM, the size of the messages is fixed. Admittedly again, these conclusions are based on generic formulas that may be oversimplifying and their concrete impact depends on the exact complexity of concrete PQ KEMs. We next show that a specialization in Kyber leads to these conclusions as well.

3.1.4 Applications to CRYSTALS-Kyber

We now aim to characterize the parameters of the SPA and DPA attack paths more concretely. For this purpose, we first introduce a slightly finer-grain analysis than the shortcut formulas in the previous section. Its goal is to enable a discussion of how stable our conclusions are in front of technology-dependent variations. We next quantify the operation counts in the specific case study of the software CRYSTALS-Kyber implementation from [Bos+21]. We finally use these concrete values to revisit the generic intuitions of the previous section.

Preliminaries. Before these discussions, we mention two preliminary steps towards specializing our shortcut formulas to Kyber. The first one is to observe that the c constant in Subsection 3.1.2 will be different for the two attack paths. In the $\mathcal{A}^{sk}_{\mathrm{DPA}}$ case, one aims to recover 12-bit values (key coefficients in the NTT domain), leading to $c = 12$. In the $\mathcal{A}^{sk}_{\mathrm{SPA}}$ case, one only wants to recover a 1-bit value (i.e., to distinguish the leakage of a message from the leakage of another message), leading to $c' = 1$. The second step is to determine the number of oracle accesses $\#\mathcal{O}$ required for the SPA to succeed. For Kyber, this number is worth $\#\mathcal{O} = k \cdot n \cdot 3$, as given in Table 3 of [Uen+21], based on the analysis in [HV20].[4]

Finer grain analysis

Equations 3.4 and 3.5 in the previous section implicitly assume that all the intermediate variables they target leak the same amount of information λ. However, it may not be the case in practice, for different (implementation-specific) reasons. The simplest one, that we will characterize next, is that the operations may not all manipulate the same amount of bits per cycle. For example, in a software implementation of Kyber, an efficient implementation of the hash function Keccak will typically be obtained by bitslicing. In this case, 32 bits would be manipulated per cycle. By contrast, the 12-bit arithmetic operations will

[4] The basic $\mathcal{A}^{sk}_{\mathrm{SPA}}$ aims to recover a single coefficient of $\boldsymbol{s}$ per oracle access. But it can be extended to target b coefficients of the secret at once, which essentially works by targeting b bits of the decrypted message rather than a single one.

generally lead to 12 bits manipulated by cycle. Assuming a Hamming weight leakage model, where one can roughly approximate $\mathtt{MI}(X; \boldsymbol{L})$ by $\log_2(n)$ when X is an n-bit value, it means that the leakage of 32-bit operations will be $f = \frac{\log_2(32)}{\log_2(12)}$ time larger than the one of 12-bit operations. A completely accurate fine-grain characterization of how the different operations of an implementation of Kyber leak is outside the scope of this work (and would go against the simplicity goal of shortcut formulas). Yet, in order to assess the potential impact of such a characterization, we will next consider two leakage parameters: λ_{32} for 32-bit operations and λ_{12} for 12-bit operations. By default, we will further assume that $\lambda_{32} = f \cdot \lambda_{12}$ with $f = \frac{\log_2(32)}{\log_2(12)}$. As a result, Equations 3.4 and 3.5 will be updated as:

$$N_{Dec} \approx \frac{12}{\#var_{Dec} \cdot \lambda_{12}^{d_{Dec}}}, \tag{3.2}$$

since there are no Keccak instances in Kyber.CPAKEM.Dec, and:

$$N_{Enc} \approx \frac{3 \cdot k \cdot n}{\#var_{Enc}^{12} \cdot \lambda_{12}^{d_{Enc}^{12}} + \#var_{Enc}^{32} \cdot \lambda_{32}^{d_{Enc}^{32}}} \cdot \tag{3.3}$$

In the next subsection, we therefore focus on evaluating the number of 12-bit and 32-bit operations that can be exploited in the different subcomputations that are found in the Kyber decryption and re-encryption algorithms.

Concrete attack parameters

DPA against decryption. The $\mathcal{A}_{\mathrm{DPA}}^{sk}$ adversary exploits the leakage generated by the first few operations in the decryption involving the private key $\hat{\boldsymbol{s}}$ (Algorithm 2 - Line 3). It aims to recover the NTT representation of the private key $\boldsymbol{s}$. More precisely, $\mathcal{A}_{\mathrm{DPA}}^{sk}$ exploits leakage from the product between the two vectors of polynomials $\hat{\boldsymbol{u}} \in \mathbb{R}_q^k$ and $\hat{\boldsymbol{s}} \in \mathbb{R}_q^k$ where $\hat{\boldsymbol{u}}$ is known by the adversary (since derived from the ciphertext). In the specific case of Kyber, the polynomial-wise base multiplications $\hat{\boldsymbol{u}}[i] \circ \hat{\boldsymbol{s}}[i]$ is performed by computing coefficient-wise operations $\hat{\boldsymbol{u}}[i]_{2j} \cdot \hat{\boldsymbol{s}}[i]_{2j}$, $\hat{\boldsymbol{u}}[i]_{2j} \cdot \hat{\boldsymbol{s}}[i]_{2j+1}$, $\hat{\boldsymbol{u}}[i]_{2j+1} \cdot \hat{\boldsymbol{s}}[i]_{2j}$ and $\hat{\boldsymbol{u}}[i]_{2j+1} \cdot \hat{\boldsymbol{s}}[i]_{2j+1}$ for $0 \leq i < k$ and $0 \leq j < n/2$. As a result by requesting one single decryption, $\mathcal{A}_{\mathrm{DPA}}^{sk}$ can directly exploit, thanks to dedicated attacks such as [BS21], the leakage from the multiplication of every coefficient in $\hat{s}$ with two know coefficients in $\hat{\boldsymbol{u}}$. Hence in this case $\#var_{Dec} = 2$.

SPA against re-encryption. As mentioned above, the $\mathcal{A}^{sk}_{\text{SPA}}$ adversary exploits the leakage of the deterministic re-encryption to gain knowledge on the message m' decrypted by Kyber.CPAPKE.Enc [Rav+20b]. Next, we evaluate the number of operations in Kyber of which the leakage can be jointly exploited to gain information on m' during the re-encryption. We first estimate the number of operations performed on 32-bit words (i.e., for hash functions). Then, we estimate the number of operations on 12-bit words (i.e., arithmetic operations).

- $\#var^{32}_{Enc}$. The re-encryption involves many calls to hash functions depending directly on the decrypted message m'. The first call to a hash function depending on m' is $\mathtt{G}$ in $\mathtt{G}(m'||\mathtt{H}(pk))$ (see Algorithm 4 - Line 2). The others are made within the re-encryption with Kyber.CPAKEM.Enc (Algorithm 1). There, $(1 + 2 \cdot k)$ calls to $\mathtt{PRF}(\cdot, \cdot)$ are performed in order to sample $\boldsymbol{e}_1$, $\boldsymbol{r}$ and e_2 with $\mathtt{CBD}_\eta(\cdot)$. As a result, $(2 + 2 \cdot k)$ calls to hash functions can be exploited by the $\mathcal{A}^{sk}_{\text{SPA}}$ adversary. All of them are based on the Keccak[1600] permutation [Ber+11].[5] For each of the calls to the hash functions, because of the input and output length, one single call to Keccak[1600] is performed.[6] Keccak[1600] requires 24 rounds alternating between linear and Sbox layers. Each round implies about 320 operations on 32-bit words. As a result, $\mathcal{A}^{sk}_{\text{SPA}}$ exploits jointly the leakage of $\#var^{32}_{Enc} \approx (2 + 2 \cdot k) \cdot 24 \cdot 320$.

- $\#var^{12}_{Enc}$. To model the information that can be extracted from the $\mathtt{NTT}$ and $\mathtt{NTT}^{-1}$ operations, we assume that each of the intermediate stages in the butterfly algorithm provides independent leakages. As a result, the adversary can exploit the leakage of $n \cdot \log_2 n$ operations for each of the calls to $\mathtt{NTT}$ or its inverse. Kyber.CPAPKE.Enc includes $2 + k$ $\mathtt{NTT}$ calls, leading to a total of $n \cdot \log_2 n \cdot (2 + k)$ operations on 12 bits from the $\mathtt{NTT}$.

  Besides, Kyber.CPAPKE.Enc also uses 12-bit element-wise operations such as base multiplication ($\circ$) and additions. The base multiplication implies $2 \cdot n$ operations and the polynomial addition only n. In Algorithm 1, the variable $\boldsymbol{u}$ is computed thanks to k^2 base multiplications and k^2 vector additions since $\hat{\boldsymbol{A}} \in \mathbb{R}_q^{k \times k}$ and $\hat{\boldsymbol{r}} \in \mathbb{R}_q^k$. Similarly, k base multiplications and $k + 2$ additions are needed to derive $\boldsymbol{v}$. Kyber.CPAPKE.Enc also includes $(2k + 1) \cdot n$

[5] Expect for the Kyber 90s versions.

[6] The only exception is for its call in $\mathtt{CBD}_\eta(\mathtt{PRF}(\cdot, \cdot))$ of Algorithm 1 if $\eta = 3$ (for Kyber512). Indeed, in such a case, $\mathtt{CBD}_3(\cdot)$ requires 1526 random bits. This requires two executions of Keccak[1600] since 1526 is larger than the rate of $\mathtt{PRF}(\cdot, \cdot)$.

calls to `CBD`, $(k+1) \cdot n$ calls to `Compress`. The final comparison consists of $(k+1) \cdot n$ coefficient-wise operations. Overall, the adversary can exploit a total of $(3k^2+8k+5) \cdot n$ leaking element-wise operations.

Hence in the following, we take $\#var^{12}_{Enc} \approx n \cdot (3k^2 + 8k + 5 + \log_2 n \cdot (2+k))$.

A look at unprotected implementations

In Figure 3.4, we report the data complexity of attacks against an unprotected ($d = 1$) implementation of Kyber768. The x-axis corresponds to physical noise parameter λ. The y-axis reports the data complexity N which is derived from Equation 3.4 for the $\mathcal{A}^{sk}_{\text{DPA}}$ adversary and from Equation 3.5 for the $\mathcal{A}^{sk}_{\text{SPA}}$ adversary, with the concrete parameters estimated above. We can observe that for low noise

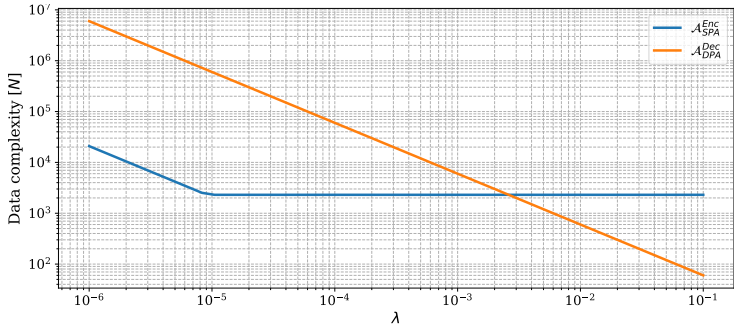


Figure 3.4: Data complexity of attacks against unprotected Kyber768.

levels (i.e., large λ values), the DPA attack path leads to stronger attacks. But as the noise increases, the SPA becomes the preferred attack path. Its complexity remains constant for a wide range of noise levels (up to λ values of 10^{-5} per leakage sample), as reflected by the plateau region of the curve. This is because a single oracle access is then sufficient to recover the key, leading to a successful attack in $\#\mathcal{O} = 3 \cdot n \cdot k$ traces. Concretely, for small λ values, the $\mathcal{A}^{sk}_{\text{SPA}}$ requires $(\#var^{32}_{Enc} \cdot f + \#var^{12}_{Enc}) \cdot 12/(\#var_{Dec} \cdot \#\mathcal{O}) \approx 2.8 \cdot 10^2$ less traces than the $\mathcal{A}^{sk}_{\text{DPA}}$, providing a quantitative view on the "curse of re-encryption".

Generic intuition revisited

In this subsection, we finally revisit the intuitions put forward in Subsection 3.1.3 with concrete parameters corresponding to a software implementation of Kyber. We aim to derive the implementation cost needed

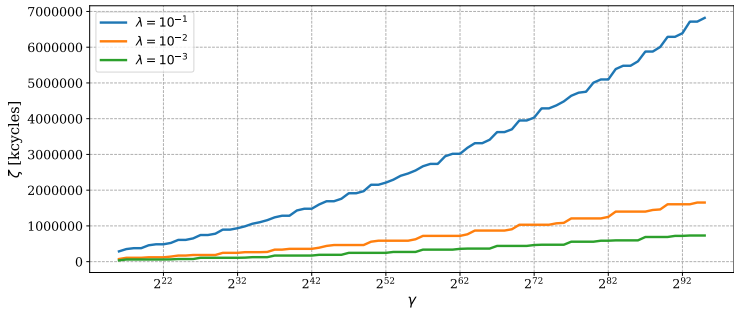


Figure 3.5: Cost (kcycles) of Kyber768 for noise parameter λ and security level γ.

to reach a given security level γ. Concretely, this requires to optimize the number of shares d_{Dec}, d_{Enc}^{12} and d_{Enc}^{32} in Equations 3.2 and 3.3 based on the numbers given in Table 3.2. We used a grid search for this purpose, considering only the integer number of shares.

Masking is (very) expensive.

The cycle counts of a protected implementation of Kyber768 is reported in Figure 3.5. It confirms the large overheads needed to reach high physical security, especially in the case of low noise levels. For example, 10^9 cycles are required to reach a 2^{32} security for a noise $\lambda = 0.1$. With a larger noise $\lambda = 0.01$, similar overheads would lead to a security level $\gamma = 2^{74}$. Our results consolidate and quantify the intuition that masking PQ KEMs like Kyber can only be done at realistic cost under sufficient noise levels. Those may not be directly available in low-end embedded microcontrollers (where $\lambda = 0.1$ is a typical value, see [BS21] Fig.10), and therefore suggest relying on secure microcontrollers with noise engines or hardware implementations for this purpose.

Levelling still moderately helps.

Figure 3.6 contains the estimated ratio between the cost of encryption and decryption for the minimal d_{Enc} and d_{Dec} leading to a given security level γ. The steps on the curves are due to the grid search on d_{Enc} and d_{Dec} which only considers an integer number of shares. Overall Figure 3.6 also confirms that the impact of the re-encryption on the overall implementation cost decreases as the security level increases. For example, for high-security levels, the protected re-encryption is about 12 times slower than the protected decryption for Kyber 768.

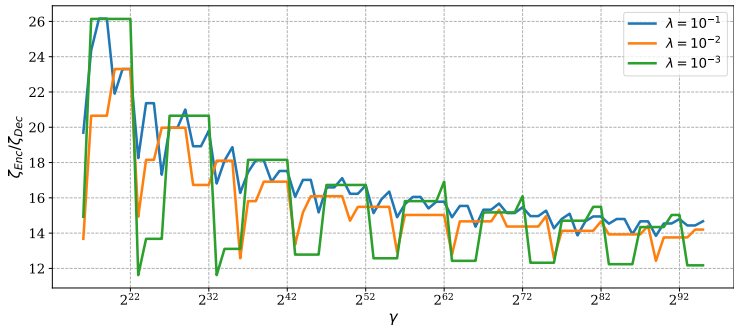


Figure 3.6: Ratio between the cost of decryption and re-encryption for Kyber768.

3.1.5 Discussion & challenges

The generic analysis in this work enables clarifying the challenges for improving the side-channel security of PQ KEMs. It first confirms that the re-encryption used in the FO transform is an important and hard-to-contain source of leakage. For an unprotected implementation, it gives rise to a very large amount of easy-to-exploit leakage samples. As a result, a SPA against this part of a PQ KEM becomes the best attack as soon as the noise in the implementation increases. Our results also show that the direct application of higher-order masking to prevent such attacks can lead to very expensive implementations. We are unaware of a CPA to CCA FO-like transform that would avoid this (or a similar) issue.

An intuitive design goal following these observations would be to limit the use or even to get rid of the FO transform. Heuristic solutions for this purpose, such as rejecting certain easy-to-exploit ciphertext structures (e.g., with low Hamming weight) or adding redundancy in the plaintext or the key are unlikely to solve the problem in a generic manner [Xu+20]. The standard solution used to obtain CCA security with leakage in symmetric cryptography (i.e., to MAC the ciphertext with careful verification mechanism [Bel+20]) is not applicable in the public-key setting. A generic replacement for this solution would be to rely on zero-knowledge proof techniques on top of the CPA encryption, to avoid using the secret key before verifying the validity of the resulting ciphertext during its decapsulation. By quantifying the performance gap between the decryption and re-encryption parts of a masked KEM, our results clarify the limited budget that such a zero-knowledge proof can use to beat the security vs. performance tradeoff of a uniformly masked

implementation. For Kyber, it roughly corresponds to a factor of 10 in clock cycles (which ignores the increase in the ciphertext size).

Overall, we therefore put forward the paradoxical observation that the FO transform leads to very strong attack vectors against unprotected PQ KEMs, but that the impact of getting rid of this transform to leverage a leveled implementation vanishes as the security level increases. The latter is mostly due to the eventually comparable complexities of a protected decryption and re-encryption, which cannot be amplified by an amortization over long messages like in the symmetric encryption setting. Hence, improving the side-channel security vs. efficiency tradeoff of PQ KEMs like Kyber will not only require to mitigate the leakages due to re-encryption, but also to find a way to make the protected decryption significantly more efficient. Our shortcut formulas directly indicate optimizations that could play a role in this direction, for example reducing the number of exploitable operations reflected by the α parameters in our formulas or leveraging different noise levels in decryption and re-encryption (i.e., playing with the f parameters of Section 3.1.4). Alternatively, finding solutions to optimize the masking complexity of the decryption would also be beneficial. In the absence of such advances, ensuring a sufficient noise level to limit the number of shares in PQ KEMs appear as the only practical option.

Eventually, while the interest of shortcut formulas lies in their ability to identify design trends, the counterpart of this genericity naturally lies in the limited details they provide about fine-grain implementation-specific issues. Evaluating the impact of the previous optimizations based on concrete case studies and experiments is therefore an interesting scope for further investigations.

3.2 Secure and efficient implementation of Dilithium, adapted from [Azo+22c]

The world's digital security infrastructure has always relied on a range of efficient and secure cryptographic primitives, including both symmetric and asymmetric solutions. In particular, for asymmetric cryptography, RSA and ECC are ubiquitous schemes in practice. However, with the anticipated advent of powerful and dedicated quantum computers, the established asymmetric cryptographic schemes, that we mainly use for key exchange and digital signatures, will no longer provide the desired security.

In 2016, the National Institute of Standards and Technology (NIST) launched a standardization effort for cryptographic schemes that can withstand quantum cryptanalysis [Nat]. Recently in 2022, the NIST announced the first Post-Quantum cryptography (PQC) schemes to be standardized. These include (CRYSTALS-)Kyber [Ava+19] for Key Encapsulation Mechanism (KEM), and (CRYSTALS-)Dilithium [Duc+17] for digital signatures. Both Kyber and Dilithium are lattice-based schemes, and in recent years the analysis of lattice-based PQC schemes and their implementations has become a prominent area of research. This is not only due to their widely accepted strong security but also because of their implementation efficiency in comparison to other PQC schemes.

Although a PQC scheme can be secure against classic and quantum adversaries, this is not sufficient to provide practical security in the embedded context. The implementations of cryptographic schemes on constrained devices can be targeted by physical attacks, which include Side-Channel Analysis (SCA) and Fault Injection (FI) attacks. Over the last years, PQC KEMs have attracted the most of the attention when it comes to SCA. Indeed, most KEM's in the NIST competition, including Kyber, relies on the Fujisaki-Okamoto (FO) transformation [FO99] which is a simple and generic technique to achieve IND-CCA security. Unfortunately, the leakage of the re-encryption step in the FO transformation leads to very powerful SCAs, demonstrated and analyzed in many recent works, including but not limited to [Rav+20b; Rav+22; Uen+22]. An adversary can also exploit leakage from the Number Theoretic Transformation (NTT) or from the Key Derivation Function (KDF) to extract the long-term secret key or the shared secret key [Rav+20a; Ham+21; KPP20; PPM17]. This variety of threats implies a large attack surface leading to significant overheads when protecting PQC KEM's [Azo+22b].

To the best of our knowledge, digital signatures, including Dilithium, have received less attention than KEMs concerning SCA. The main results include a work by Ravi et al. [Rav+18] that shows that to achieve existential forgery an attacker only requires knowledge of one part of the secret key in Dilithium, namely $\mathbf{s_1}$. Marzougui et al. [Mar+22] exploit leakage of the zero coefficients in the secret signing nonce $\mathbf{y}$ for multiple signatures and recover the secret key by leveraging least squares regression and integer linear programming. Liu et al. [Liu+21] also present an SCA on Dilithium, which can recover the secret key from the leakage of a single bit of the secret signing nonce $\mathbf{y}$ for multiple signatures. The authors use this side-channel information to define a problem called the Fiat-Shamir Integer LWE, and showed that it can be solved efficiently. This attack is very reminiscent of the well-known

lattice reduction attacks on (EC)DSA (and other Schnorr-like signature schemes) with partial nonce leakage, originally due to Howgrave-Graham and Smart [HS01] and recently improved by Sun et al. [Sun+22]. Liu et al. showed that their attack requires a relatively low number of signatures. This result, along with previous works and the fact that the side-channel analysis of Dilithium is quite a new research topic for the community, highlights the vital need to protect the future digital signature standard against these threats. The amount of published works appears to be even scarcer when it comes to protecting Dilithium against leakage. To the best of our knowledge, the main contribution comes from Migliore et al. [Mig+19] and presents masked gadgets for Dilithium and a power-of-two modulus masked version of it.

Contributions. In this work, we tackle the challenge of efficiently protecting Dilithium implementations on embedded devices. Our contributions are the following. First, we revisit the sensitivity analysis of Migliore et al. [Mig+19]. Interestingly, we notice that the authors do not consider some intermediate computations as sensitive even though they can be explicitly used to recover the secret key. Conversely, others were unnecessarily protected since they could be computed from the signature and the public key. These observations lead to improved security and to more efficient signature generation. To the best of our knowledge, our work presents the first masked Dilithium design compliant with the third round submission document for all parameter sets.

Second, and following the security requirements of our sensitivity analysis, we propose new and improved masked gadgets for the main operations of Dilithium (namely the bound check, the secret sampling and the decomposition) and for all NIST security levels.

Finally, we provide a complete benchmark for an ARM Cortex-M4 microcontroller, which includes the evaluation of individual components, and their comparison with the ones of Migliore et al., and performance results of full signature generation for deterministic and randomized versions of Dilithium. They highlight the advantages of randomized Dilithium compared to its deterministic variant in the context of physical attacks.

Cautionary note. In an earlier presentation of these results, at the NIST's 4th PQC Standardization Conference, a finer-grain sensitivity analysis distinguishing security against Simple Power Analysis and Differential Power Analysis was proposed [Azo+22c]. This analysis was conjectured to enable strongly leveled implementations of Dilithium, where

different parts of the implementation use different countermeasures (e.g., shuffling against SPA [Vey+12], masking against DPA [Cha+99; ISW03]). We clarify in Section 3.2.2 that the possibility to leverage a "hard physical learning problem" similar to [Duv+21] that would back up this conjecture does not hold. As a result, Dilithium has less potential for leveling and its sensitivity analysis can be simplified to a coarser-grain mix of sensitive operations that require DPA protections and non-sensitive ones that can leak in full. We discuss directions to partially restore the interest of a finer-grain leveling in conclusions. We also come back to this topic in the conclusion of the thesis, after introducing hard physical learning problems in Chapter 4.

3.2.1 Background

We now detail some additional notations used in the section and a description of the Dilithium signature scheme.

We define $S_\eta = \{w \in R : \|w\|_\infty \leq \eta\}$ and $\tilde{S}_\eta = \{w \bmod^{\pm} 2\eta : w \in R\}$. This means that the coefficients of an element in S_η or $\tilde{S}_\eta$ are in the range $[-\eta, \eta]$ or $]-\eta, \eta]$, respectively. We use the notation $x \leftarrow \mathcal{X}$ whenever we assign a uniformly random element of a set $\mathcal{X}$ to a variable x. The symbol $\|$ is used for the concatenation of two-bit strings, the function `H` is an expandable output function (`XOF`).

Dilithium

Dilithium is a digital signature scheme based on the MLWE (Module Learning With Errors) and the SelfTargetMSIS (Module Short Integer Solution) problems [LS15]. It is the primary algorithm selected by the NIST for quantum-safe digital signatures. Its main features are: random sampling from a uniform distribution instead of a discrete Gaussian distribution, a focus on keeping the public key and the signature as small as possible in terms of their bit size, and being easy to adjust for different security levels by only changing the dimensions of the matrices and vectors involved. For a comprehensive description of the algorithm, we refer to the proposal [Duc+17]. Note that the pseudocode presented in this work is a variation from the reference implementation described in [Duc+17, p.17]. The key differences are highlighted in Section 3.2.2. In this work, we refer to the implemented version if not stated otherwise. We describe the key generation and signature generation algorithms in the following paragraphs. We do not consider the verification, which does not involve long-term secret variables (hence, does not leak sensitive information). Table 3.3 provides the Dilithium parameters for different NIST security levels.

NIST Security level	2	3	5
q (modulus)	$2^{23} - 2^{13} + 1$	$2^{23} - 2^{13} + 1$	$2^{23} - 2^{13} + 1$
d (number of dropped bits from $\mathbf{t}$)	13	13	13
τ (# of $\pm$ 1's in c)	39	49	60
γ_1 ($\mathbf{y}$ coefficient range)	2^{17}	2^{19}	2^{19}
γ_2 (low order rounding range)	(q-1)/88	(q-1)/32	(q-1)/32
(k, l) (dimensions of $\mathbf{A}$)	(4,4)	(6,5)	(8,7)
η (secret key range)	2	4	2
β ($= \tau \cdot \eta$)	78	196	120
ω (max. # 1's in $\mathbf{h}$)	80	55	75
average number of signing iterations	4.25	5.1	3.85

Table 3.3: Dilithium parameters.

Key generation. The key generation is defined in [Duc+17, Fig 4.] and is recalled in Algorithm 5. Initially, a random bit string ζ is created and used to generate three seeds ρ, ς and K thanks to the hash function $\mathtt{H}$. A public matrix $\mathbf{A}$ for which all coefficients are uniform in $\mathbb{Z}_q$ is generated from ρ. Two secret vectors $\mathbf{s}_1 \in S_\eta^l$ and $\mathbf{s}_2 \in S_\eta^k$ are derived from ς. Then, the vector $\mathbf{t} = \mathbf{A}\mathbf{s}_1 + \mathbf{s}_2$ is calculated. This is an instance of MLWE, where $\mathbf{s}_1$ and $\mathbf{s}_2$ are hard to calculate given $\mathbf{A}$ and $\mathbf{t}$. Next, the bit representation of $\mathbf{t}$ is split up into high order bits $\mathbf{t}_1$ and low order bits $\mathbf{t}_0$. Only $\mathbf{t}_1$ will be part of the public key, to keep its size as small as possible. For the same reason, the matrix seed ρ is part of the output, rather than the whole matrix $\mathbf{A}$. Lastly, $\rho \| \mathbf{t}_1$ gets hashed to tr. The output is the public key $pk = (\rho, \mathbf{t}_1)$ and the secret key $sk = (\rho, K, tr, \mathbf{s}_1, \mathbf{s}_2, \mathbf{t}_0)$.

Algorithm 5 KeyGen.

1: $\zeta \leftarrow \{0,1\}^{256}$
2: $(\rho, \varsigma, K) = \mathtt{H}(\zeta)$ ▷ $(\rho, \varsigma, K) \in \{0,1\}^{256} \times \{0,1\}^{512} \times \{0,1\}^{256}$
3: $\mathbf{A} = \mathtt{ExpandA}(\rho)$ ▷ $\mathbf{A} \in R^{k \times l}$
4: $(\mathbf{s_1}, \mathbf{s_2}) = \mathtt{ExpandS}(\varsigma)$ ▷ $(\mathbf{s_1}, \mathbf{s_2}) \in S_\eta^l \times S_\eta^k$
5: $\mathbf{t} = \mathbf{A}\mathbf{s_1} + \mathbf{s_2}$
6: $(\mathbf{t_1}, \mathbf{t_0}) = \mathtt{Power2Round}(\mathbf{t}, d)$
7: $tr = \mathtt{H}(\rho \| \mathbf{t_1})$ ▷ $tr \in \{0,1\}^{256}$
8: **return** $pk = (\rho, \mathbf{t_1}), sk = (\rho, K, tr, \mathbf{s_1}, \mathbf{s_2}, \mathbf{t_0})$

Signature. Similarly, we now describe the signing procedure in Algorithm 6. We refer to [Duc+17, Fig 4.] for a more detailed description. The input is the secret key sk and a message M. The message is pre-

processed with $\mathtt{H}$ into a bit string μ of fixed length. For deterministic signing, μ is used together with K to produce a seed ρ'. For the randomized version the seed ρ' is generated randomly. This seed and a rejection counter κ (initially set to $\kappa = 0$) are used to sample the secret polynomial $\mathbf{y} \in \tilde{S}^l_{\gamma_1}$ with $\mathtt{ExpandMask}$. Then, the product $\mathbf{w} = \mathbf{Ay}$ is decomposed via division with remainder into $\mathbf{w}_1$ and $\mathbf{w}_0$. The challenge $\tilde{c}$ is the hash of $\mu \| \mathbf{w}_1$. For further calculations, $\tilde{c}$ is converted into a polynomial c that contains strictly τ coefficients set to ± 1 and the others set to zero. This polynomial is then used to calculate $\mathbf{z}$ and $\tilde{\mathbf{r}}$. To ensure the security and correctness of the scheme, two checks are performed:

$$\|\mathbf{z}\|_\infty < \gamma_1 - \beta, \qquad \|\tilde{\mathbf{r}}\|_\infty < \gamma_2 - \beta,$$

where $\beta = \eta \cdot \tau$. If any of the two conditions does not hold, κ is increased and the process starts over (beginning with the sampling of a new $\mathbf{y}$). After successful checks, a hint $\mathbf{h}$ is calculated. This is needed in the verification step to make up for the "lost" information of $\mathbf{t}_0$. Two more checks are performed on $c\mathbf{t}_0$ and $\mathbf{h}$. Again, if these conditions are not met, the signature is rejected and κ is increased. Otherwise, if all checks are successful, the signature $\sigma = (\tilde{c}, \mathbf{z}, \mathbf{h})$ can be output.

Algorithm 6 Sign(sk, M).

$\mathbf{A} = \mathtt{ExpandA}(\rho)$
$\mu = \mathtt{H}(tr \| M)$ ▷ $\mu \in \{0,1\}^{512}$
$\kappa = 0, (\mathbf{z}, \mathbf{h}) = \perp$
$\rho' = \mathtt{H}(K \| \mu)$ (or $\rho' \xleftarrow{\$} \{0,1\}^{512}$ for randomized signing) ▷ $\rho' \in \{0,1\}^{512}$
while $(\mathbf{z}, \mathbf{h}) = \perp$ **do**
    $\mathbf{y} = \mathtt{ExpandMask}(\rho', \kappa)$ ▷ $\mathbf{y} \in \tilde{S}^l_{\gamma_1}$
    $\mathbf{w} = \mathbf{Ay}$
    $(\mathbf{w_0}, \mathbf{w_1}) = \mathtt{Decompose}(\mathbf{w}, 2\gamma_2)$
    $\tilde{c} = \mathtt{H}(\mu \| \mathbf{w_1})$ ▷ $\tilde{c} \in \{0,1\}^{256}$
    $c = \mathtt{SampleInBall}(\tilde{c})$ ▷ $c \in B_\tau$
    $\mathbf{z} = \mathbf{y} + c\mathbf{s_1}$
    $\tilde{\mathbf{r}} = \mathbf{w_0} - c\mathbf{s_2}$
    if $\|\mathbf{z}\|_\infty \geq \gamma_1 - \beta$ or $\|\tilde{\mathbf{r}}\|_\infty \geq \gamma_2 - \beta$ **then** $(\mathbf{z}, \mathbf{h}) = \perp$
    else
        $\mathbf{h} = \mathtt{MakeHint}(\tilde{\mathbf{r}}, c, \mathbf{t_0}, \mathbf{w_1}, \gamma_2)$
        if $\|c\mathbf{t_0}\|_\infty \geq \gamma_2$ or the # of 1's in $\mathbf{h}$ is greater than ω **then** $(\mathbf{z}, \mathbf{h}) = \perp$
    $\kappa = \kappa + l$
return $\sigma = (\tilde{c}, \mathbf{z}, \mathbf{h})$

3.2.2 Sensitivity analysis

In this section, we analyze Dilithium's key generation and signature generation and discuss the sensitivity of all the variables and functions potentially leading to side-channel attacks. This sensitivity analysis indicates which operations/variables must be protected against leakage. As mentioned in the introduction, we use a coarse-grain taxonomy for this purpose, which is next reflected by color-coded diagrams: Figure 3.7 and Figure 3.8, where red (resp., blue) denotes sensitive variables/operations that need security against DPA (resp., variables/operations that do not require side-channel attack protection). Doing so we also compare our analysis to the one previously proposed in [Mig+19].

Starting with generalities, we first note that the public key can be leaked to the adversary over the whole scheme (since it is public). The public matrix $\mathbf{A}$ can also be leaked since it is deterministically derived from ρ. A similar status holds for some parts of the secret key $sk := (\rho, K, tr, \mathbf{s}_1, \mathbf{s}_2, \mathbf{t}_0)$, since similar variables are contained in the public key. Concretely, tr does not need to be protected since it is a hash of pk. We additionally note that the vector of polynomials $\mathbf{t}_0$ can be leaked as well. Indeed, the Dilithium security proofs consider $\mathbf{t}$ (hence $\mathbf{t}_0$ and $\mathbf{t}_1$) to be public [Duc+17].[7] M does not need to be protected, but $\mathbf{s}_1$ and $\mathbf{s}_2$, and K, must be protected to avoid side-channel attacks leading to a signature forgery. Next, we detail which other variables must be protected to avoid the leakage of long-term sensitive secrets. We start with their sensitivity analysis for the key generation followed by the signing procedure.

Key generation sensitivity

During key generation, the variable ζ has to be protected since it is the seed for all subsequent values (e.g., K). Similarly, ς has to be protected since it serves as a seed to deterministically generate the long-term secrets $\mathbf{s}_1$ and $\mathbf{s}_2$. All the other variables in the key generation can be leaked or are public, hence do not need side-channel protection.

Signature generation sensitivity

All the variables denoted in red in Figure 3.8 need to remain secret and hence must be secure against DPA. This naturally holds for both secret

[7] As a result $\mathbf{t}$ could be fully part of the public key. Alternatively, it can decomposed into $\mathbf{t}_0$ and $\mathbf{t}_1$ so that the public key only contains $\mathbf{t}_1$. This reduces the size of the public key by a factor close to two at the cost of an increased secret key size, which must then contain $\mathbf{t}_0$, and a slightly increased signature size.

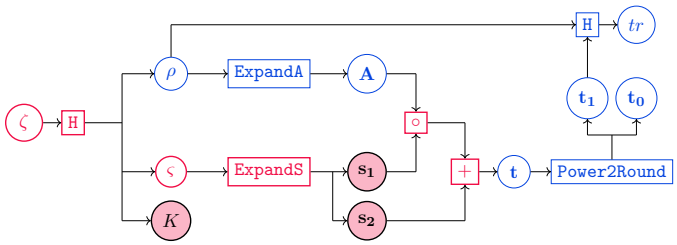


Figure 3.7: Graphical representation of the key generation. Output: $pk = (\rho, \mathbf{t_1})$, $sk = (\rho, K, tr, \mathbf{s_1}, \mathbf{s_2}, \mathbf{t_0})$. Red: sensitive. Blue: non-sensitive.

key components $\mathbf{s_1}$ and $\mathbf{s_2}$, to avoid trivial key recovery attacks leading to signature forgeries. Next, the vector of polynomials $\mathbf{y}$ is sensitive and must be protected. Indeed, given a valid signature $\sigma = (\tilde{c}, \mathbf{z}, \mathbf{h})$, the secret vector $\mathbf{s}_1$ can be recovered from $\mathbf{z} = \mathbf{y} + c\mathbf{s}_1$ for known or partial knowledge of $\mathbf{y}$ [Mar+22]. A similar analysis applies to $\mathbf{w}_0$ which can lead to the recovery of $\mathbf{s}_2$.[8] As a result, the vector of polynomials $\mathbf{w}$ must be protected: $\mathbf{w}_0$ is directly derived from $\mathbf{w}$, and it is possible to solve the system of equations $\mathbf{Ay} = \mathbf{w}$ for known $\mathbf{A}$ and $\mathbf{w}$ to recover $\mathbf{y}$ in most cases (see Section 3.2.2 for details). This equation is similar to a Learning With Rounding (LWR) instance, where w_1 would be the public rounded value and w_0 would be the error (which cannot leak). For the same reason, ρ' must be protected since it is used as a seed to obtain $\mathbf{y}$. The same holds for K in the deterministic signing case.

Next, when it comes to public or non-sensitive variables, both tr and $\mathbf{t_0}$, the message M, the seed ρ, the hash μ and the matrix $\mathbf{A}$ are public. The sensitivity of the vector of polynomials $\mathbf{w}_1$ in the signing procedure is more delicate to analyze, since it depends on the result of the boundary checks on $\mathbf{z}$ and $\tilde{\mathbf{r}}$. If the boundary checks pass, for example when a signature is accepted, then the zero-knowledge proof of security of Dilithium shows that $\mathbf{w}_1$ does not leak any information. Informally, $\mathbf{w}_1$ can be reconstructed from a valid signature, which in turn can be simulated in zero knowledge, and hence $\mathbf{w}_1$ contains no more information than the signature itself. As a result, Dilithium does not need an explicit LWR hardness assumption in this case, since it is at least as hard as its LWE assumption. When the boundary checks do not pass, the reduction from LWR to LWE does not apply immediately since the distribution on $\mathbf{w}_1$ changes slightly. Leaving $\mathbf{w}_1$ unmasked therefore requires the additional explicit assumption that the corresponding LWR problem is

[8] A recent ePrint report details an attack exploiting the leakage of $\mathbf{w}_0$ [Ber+23].

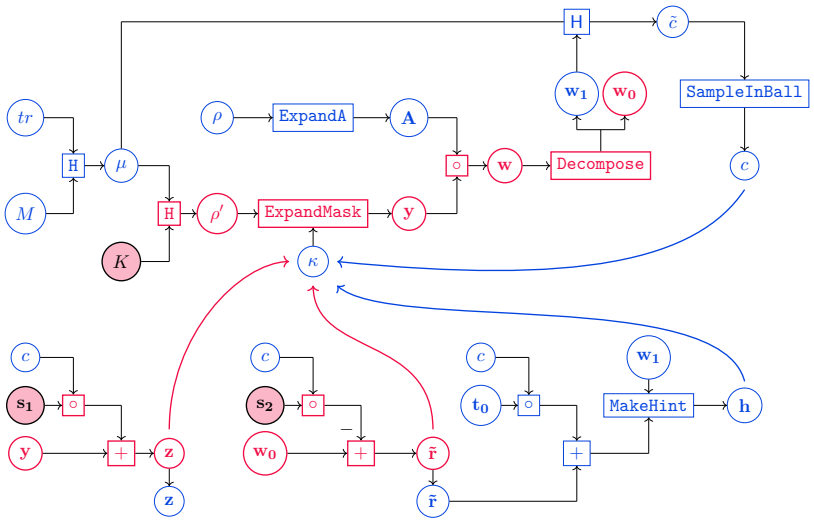


Figure 3.8: Graphical representation of the signature generation. Input: sk, M, Output: $\sigma = (\tilde{c}, \mathbf{z}, \mathbf{h})$. Curved arrows represent rejection checks. Red: sensitive. Blue: non-sensitive.

hard. Since the number of rounded bits is significantly higher than the error that is added in the LWE problem, we conjecture that $\mathbf{w}_1$ does not require side-channel countermeasures. The same expectation was also shared by Vadim Lyubashevsky in a personal communication and publicly during RWPQC23. We leave its further investigation, e.g., in the light of the recent [Dev+23], as an interesting scope for further investigations. To be conservative, we nevertheless study the option of additionally protecting $\mathbf{w}_1$ in Section 3.2.5. For the rest, the challenge c can be left unprotected since it is derived from a one-way hash of $\mathbf{w}_1$ and public inputs. And once the bound checks on $\mathbf{z}$ and $\tilde{\mathbf{r}}$ (discussed further in the next paragraph) have passed, the hint vector can be made public since it does not contain any sensitive information. Indeed, in the simplified version of Dilithium which does not involve the hints or the public key compression, all information that would be given by the hints is already contained in the returned valid signature and the public key. The checks on $c\mathbf{t_0}$ and $\mathbf{h}$ are needed for correctness only.

Finally, regarding $\mathbf{z}$ and $\tilde{\mathbf{r}}$, both must remain protected until the bound checks on both have passed. This implies the need for a secure bound check algorithm. After successful bound checks, they do not leak information about other sensitive values and can be leaked to the adversary.[9] For $\mathbf{z}$ this is trivial, as it is part of the signature. In the case

[9] This refers to the rejection checks on $\mathbf{z}$ and $\tilde{\mathbf{r}}$. The one on $\mathbf{h}$ is not sensitive.

of $\tilde{\mathbf{r}}$, this can be shown by the equation:

$$\mathbf{Az} - c\mathbf{t} = \mathbf{w} - c\mathbf{s}_2 = \alpha\mathbf{w}_1 + \tilde{\mathbf{r}}.$$

Indeed, for a valid signature, the values $\mathbf{A}$, $\mathbf{z}$, c, $\mathbf{t}$, and $\mathbf{w}_1$ are not sensitive and α is a known parameter of the algorithm. Therefore, $\tilde{\mathbf{r}}$ can be computed using only public values, so there is no need to keep it protected after a successful signing process. A public $\tilde{\mathbf{r}}$ is quite handy because it allows us to compute the hint $\mathbf{h}$ completely on public data.

Differences with [Mig+19]

Most of our claims made above do align with the ones made in [Mig+19]. However, our conclusions on $\mathbf{w}$ and $\tilde{\mathbf{r}}$ slightly differ, which we discuss in the following.

Protecting $\mathbf{w}$. First, we look at $\mathbf{w}$, in particular at the system of equations that produces it: $\mathbf{Ay} = \mathbf{w}$. It is possible to solve this system for $\mathbf{y}$, if the matrix $\mathbf{A}$ has one more row than columns. This is the case for NIST security levels 3 and 5, where $\mathbf{A}$ has dimensions 6×5 and 8×7 respectively. Even a simple solver is able to compute the sensitive $\mathbf{y}$ in less than two minutes on a laptop, with original Dilithium parameters.[10] For level 2, since the matrix $\mathbf{A}$ is square (of dimensions 4×4) and random, it is most likely invertible.[11] Hence, with knowledge of $\mathbf{w}$, $\mathbf{y}$ can be computed simply as $\mathbf{y} = \mathbf{A}^{-1} \cdot \mathbf{w}$. This shows that $\mathbf{w}$ must be protected, contrary to what was previously proposed in [Mig+19].

Unmasking $\tilde{\mathbf{r}}$. Before we look at $\tilde{\mathbf{r}}$ we need to address a variation of the signing procedure in Dilithium. The original pseudocode for the signing algorithm described in [Duc+17] only keeps the output $\mathbf{w}_1$ from $\texttt{Decompose}(\mathbf{w})$. Then, instead of $\tilde{\mathbf{r}} = \mathbf{w}_0 - c\mathbf{s}_2$ it computes $\mathbf{r} = \mathbf{w} - c\mathbf{s}_2$. The rejection check on $\tilde{\mathbf{r}}$ is done on $\mathbf{r}_0$, which comes from $\texttt{Decompose}(\mathbf{r})$.

[10] First, the polynomials of the first column of $\mathbf{A}$ are reduced to monic form via Gaussian elimination on the coefficients. Then, the fact that $a_{ij}X^k \cdot X^{n-k} = -a_{ij}X^0 (\text{mod } X^n + 1)$ is used to eliminate all but one polynomial in the first column. The remaining polynomial then allows us to eliminate all others in the first row, from column 2 onwards. The procedure is repeated on the second column, etc. The solver was implemented in Matlab and tested with the parameters given in [Duc+17] for security levels 3 and 5.

[11] More precisely, since the entries of $\mathbf{A}$ are elements of R, and R contains $q^n = 8380417^{256}$ elements, it is highly unlikely that two rows of $\mathbf{A}$ are linearly dependent. Therefore, it almost always holds that the determinant of $\mathbf{A}$ is non-zero and an inverse matrix exists.

Also, the `MakeHint` function works slightly different and takes $\mathbf{r}, c, \mathbf{t}_0$ as input (but produces the same exact output $\mathbf{h}$). In [Mig+19], this $\mathbf{r}$-version is used while the $\tilde{\mathbf{r}}$-version is never mentioned. However, considering the equation:

$$\mathbf{r} = \mathbf{w} - c\mathbf{s}_2 = \alpha\mathbf{w}_1 + \mathbf{w}_0 - c\mathbf{s}_2 = \alpha\mathbf{w}_1 + \tilde{\mathbf{r}}.$$

we can see that $\mathbf{r}$ and $\tilde{\mathbf{r}}$ can be calculated from each other using the public values $\mathbf{w}_1$ and α. So any consideration regarding the sensitivity classification of one of these values automatically applies to the other one as well. In [Mig+19], the value $\mathbf{r}$ is never unmasked which means that the calculation of the hint $\mathbf{h}$ must be protected against side-channel attacks. But as we explained above, $\tilde{\mathbf{r}}$ can be recreated from public values after a valid signature output. So we consider $\tilde{\mathbf{r}}$ as public after the checks on $\mathbf{z}$ and $\tilde{\mathbf{r}}$.

Differences with [Azo+22c]

In a previous version of those results, a finer-grain sensitivity analysis was proposed, suggesting that the computation of $c\mathbf{s}_1 + \mathbf{y} = \mathbf{z}$ could only require security against SPA. It was in particular conjectured that an intermediate attack path targeting this computation would be hard, the investigation of which being left as an open problem. However, it appears that the analogy between the multiplication $c\mathbf{s}_1$ and a variant of "hard physical learning problem" similar to [Duv+21], which would back up this conjecture, does not hold.

The problem, already glimpsed in [Azo+22c], is that c and $\mathbf{s}_1$ are not uniformly distributed and have small norms, while the hardness of hard physical learning problems leverages uniform secrets so that computing the multiplication leads to modular reductions. Combined with the fact that when the signature is correct, the knowledge on $\mathbf{z}$ can be used to directly transfer information on $\mathbf{y}$ into information on the output of the multiplication, it implies that $\mathbf{y}$ and $\mathbf{z}$ need to be protected against DPA. As a result, Dilithium has fewer opportunities for leveling and its sensitivity analysis can be simplified into the mix of unprotected and DPA-protected operations that we now use.

However further sensitivity may be conducted with tools introduce in Chapter 4. Those optimizations will be discussed in the conclusion of this manuscript.

3.2.3 Improved masked gadgets

In this section, we describe the techniques used for masking Dilithium. First, we recall some standard notions of masking. Then, we provide a

set of new gadgets dedicated to Dilithium operations. For each of them, we justify their correctness and discuss their probing security. Finally, we discuss their instantiation in the case of the different parameter sets of Dilithium.

Concretely, these gadgets are essentially relying on standard approaches tailored to the Dilithium use case and most of our optimizations are obtained from carefully selecting the type of masking (i.e., Boolean or arithmetic) We in particular rely on the recently improved masking conversion proposed in [BC22a] for this purpose.

Masking background

Masking is a popular countermeasure against side-channel attacks. It consists of splitting any sensitive variable x into d shares [Cha+99]. Concretely, $d-1$ shares are chosen uniformly at random. Hence, any subset of $d-1$ shares remains independent of the secret x, forcing the adversary to exploit d shares simultaneously to extract sensitive information. This property must be maintained during the entire execution of the masked circuit. This is formalized in the probing model, ensuring that the adversary learns no information about the secret by having access to $d-1$ intermediate variables [ISW03].

In lattice-based cryptography, two types of masking are used. The first one is Boolean masking. In such a case, the sharing of a k-bit Boolean variable x is written as $\boldsymbol{x}^{B,k}$ and satisfies the property that $x = \bigoplus_{i=0}^{d-1} \boldsymbol{x}_i^{B,k}$ where $\boldsymbol{x}_i^{B,k}$ is the i-th share of x. The notation $\boldsymbol{x}^{B,k}[j]$ denotes the sharing of the j-th bit of x. Boolean masking is typically used for protecting symmetric primitives such as hash functions. The second one is arithmetic masking. In such a case, the sharing of a variable $x \in \mathbb{Z}_q$ is expressed as $\boldsymbol{x}^{A_q}$ such that $x = \sum_{i=0}^{d-1} \boldsymbol{x}_i^{A_q} \bmod q$ where $\boldsymbol{x}_i^{A_q}$ is the i-th share of x. Arithmetic masking is typically used to perform polynomial operations such as additions and multiplications. Since both arithmetic and Boolean masking are used to protect lattice-based cryptography, gadgets are required to convert masking from one type to another. To convert from arithmetic to Boolean masking, we use $\texttt{SecA2BModp}_q^d$. Similarly to converting from Boolean masking to arithmetic masking, we use $\texttt{SecB2AModp}_q^d$. Eventually, we also leverage the gadget $\texttt{SecAddModp}_q^d$ that performs a modular addition operating on inputs protected with Boolean masking. We refer to [BC22a] for the implementation of these algorithms.

In this work, probing security is ensured thanks to the Probe Isolating Non Interference (PINI) security notion [CS20]. Fulfilling PINI ensures

probing security and the composition of PINI gadgets is PINI as well. This means that PINI gadgets can be composed (without refresh) and the resulting circuit is probing secure. Since the new gadgets we propose can be expressed as a composition of PINI gadgets previously proposed by Bronchain and Cassiers, it directly implies that they are PINI and therefore probing secure.

`SecLeq`

We first introduce $\texttt{SecLeq}_{\psi}^{d}\left(\boldsymbol{x}^{B,k}\right)$ described in Algorithm 8. It outputs a bit b equal to 1 if the input Boolean sharing of the k-bit variable x is less than or equal to a bound ψ.

Algorithm 7 $\texttt{SecUnMask}_{k}^{d}\left(\boldsymbol{x}^{B,k}\right)$

Input: Boolean sharing $\boldsymbol{x}^{B,k}$ with $0 \leq x < 2^k$.
Output: Output the k-bit unmasked value x.

1: $\boldsymbol{y}^{B,k} \leftarrow \texttt{Refresh}_{k}^{d}\left(\boldsymbol{x}^{B,k}\right)$ ▷ Refresh based on the ISW multiplication [CS21b, Algorithm 3].
2: $x \leftarrow \bigoplus_{i=0}^{d-1} \boldsymbol{x}_i^{B,k}$

Algorithm 8 $\texttt{SecLeq}_{\psi}^{d}\left(\boldsymbol{x}^{B,k}\right)$

Input: Boolean sharing $\boldsymbol{x}^{B,k}$ with $0 \leq x < 2^k$ and $\psi \geq 0$.
Output: For $0 \leq \psi < 2^k - 1$, public bit b with $b = 1$ if $x \leq \psi$ and $b = 0$ otherwise. If $\psi \geq 2^k - 1$, trivially returns $b = 1$.

1: $\boldsymbol{x'}^{B,k+1} \leftarrow \texttt{SecAdd}_{k+1}^{d}\left(\boldsymbol{x}^{B,k}, 2^{k+1} - \psi - 1\right)$
2: $b \leftarrow \texttt{SecUnMask}_{1}^{d}\left(\boldsymbol{x'}^{B,k+1}[k]\right)$

Correctness. We next detail the correctness of Algorithm 8 for the case of $0 \leq \psi < 2^k - 1$. The first step in `SecLeq` consists of doing an addition of x with the $(k+1)$-bit two's complement representation of $-(\psi+1)$ to obtain $x' = x - \psi - 1$. As a result, the output b must be set to 1 only if x' is strictly negative. Because of the input conditions $0 \leq x < 2^k$ and $0 \leq \psi < 2^k - 1$, the resulting x' is included in $-2^k \leq x < 2^k$ which fits in a $k+1$-bit two complement representation, hence no overflow occurs in the subtraction. The second step consists of unmasking the $(k+1)$-th bit of x' which corresponds to the sign bit of the two's complement

representation. Eventually, the case of $\psi \geq 2^k - 1$ is trivial. Indeed, $x \leq 2^k - 1$ hence x is always smaller or equal to ψ.

Proposition 1. *Algorithm 8 is PINI if b is public.*

Proof. `SecUnMask` is PINI as a consequence of [Cor+23, Lemma 2].[12]. Therefore, if b is public, Algorithm 8 is a composition of PINI gadgets. □

Usage in Dilithium. `SecLeq` is not a high-level component of Dilithium but is instead used as a building block in Algorithm 9 and Algorithm 10. We note that the $\texttt{SecAdd}^d_{k+1}$ in `SecLeq` can be generically implemented with the full-adder-based addition proposed in [BC22a, Algorithm 6]. In the context of Dilithium, the added constant is public and fixed by the parameter set, enabling possible optimization of the adder taking into account the bits of the constant as well as the fact that only the sign bit (and so all the intermediate carries) must be explicitly computed. These optimizations depend on the constant and can lead to the saving of multiple `SecAnd`'s and XOR's.

`SecBoundCheck`

Algorithm 9 describes $\texttt{SecBoundCheck}^d_{q,\lambda_0,\lambda_1}\left(\boldsymbol{x}^{A_q}\right)$ which returns a bit b if the input arithmetic sharing $\boldsymbol{x}^{A_q}$ satisfies the property $-\lambda_0 \leq x \leq \lambda_1 \bmod q$.

Algorithm 9 $\texttt{SecBoundCheck}^d_{q,\lambda_0,\lambda_1}\left(\boldsymbol{x}^{A_q}\right)$

Input: Arithmetic sharing $\boldsymbol{x}^{A_q}$, integer $q < 2^k$ and $\lambda_0 + \lambda_1 < q$ with $\lambda_0 \geq 0$ and $\lambda_1 \geq 0$.
Output: Bit b with $b = 1$ if $-\lambda_0 \leq x \leq \lambda_1 \bmod q$, $b = 0$ otherwise.

1: $\boldsymbol{x'}_0^{A_q} \leftarrow \boldsymbol{x}_0^{A_q} + \lambda_0 \bmod q$ ▷ $b = 1$ iff $0 \leq x' \leq \lambda_1 + \lambda_0 \bmod q$
2: $\boldsymbol{x'}^{B,k} \leftarrow \texttt{SecA2BModp}^d_q\left(\boldsymbol{x'}^{A_q}\right)$
3: $b \leftarrow \texttt{SecLeq}^d_{\lambda_0+\lambda_1}\left(\boldsymbol{x'}^{B,k}\right)$

Correctness The first step in Algorithm 9 is to add λ_0 to the input sharing of x resulting in a sharing of x'. As a result, the output bit b will

[12] `Refresh` is $(d-1)$-free-SNI [CS21b] Therefore, all its outputs and any set of at most t probes inside the gadget can be simulated by knowing its output and t of its input shares.

be set to one if and only if $0 \leq x' \leq \lambda_0 + \lambda_1 \bmod q$. The second step is to check that condition thanks to `SecLeq`. To do so, the arithmetic sharing $\boldsymbol{x'}^{A_q}$ of x' is converted to a Boolean sharing $\boldsymbol{x'}^{B,k}$ thanks to `SecA2BModp`. The resulting sharing fulfills the input conditions of `SecLeq`. Indeed, $(x' \bmod q) < q$ and $q < 2^k$ implies $x' < 2^k$. Additionally, since λ_0 and λ_1 are positive integers, we ensure that $\lambda_0 + \lambda_1 \geq 0$. The returned bit by `SecBoundCheck` is the one returned by `SecLeq`.

Proposition 2. *Algorithm 9 is PINI.*

Proof. The first addition is applied only on the first share hence PINI. $\texttt{SecA2BModp}_q^d$ is PINI by [BC22a, Proposition 4], and Algorithm 8 is PINI by 1. Hence, Algorithm 9 is PINI since it is the composition of PINI gadgets. □

Usage in Dilithium. `SecBoundCheck` can be used to perform both rejection checks $\|\mathbf{z}\|_\infty < \gamma_1 - \beta$ and $\|\tilde{\mathbf{r}}\|_\infty < \gamma_2 - \beta$, where γ_1, γ_2 and β are defined by Dilithium specifications (see Table 3.3). In the first case, `SecBoundCheck` is instantiated with $\lambda_0 = \lambda_1 = \gamma_1 - \beta - 1$, where the -1 is due to the strict inequality in the norm check. Similarly, in the second case, `SecBoundCheck` is instantiated with $\lambda_0 = \lambda_1 = \gamma_2 - \beta - 1$.

`SecSampleModp`

Algorithm 10 describes `SecSampleModp` which samples uniformly x over the range $-\phi_0 \leq x \leq \phi_1 \bmod p$ and outputs an arithmetic sharing when provided with a masked uniform randomness stream $(\boldsymbol{x}_0^{B,k}, \boldsymbol{x}_1^{B,k}, \dots)$.

Algorithm 10 $\texttt{SecSampleModp}_{q,\phi_0,\phi_1}^d\left(\boldsymbol{x}_0^{B,k}, \boldsymbol{x}_1^{B,k}, \dots\right)$

Input: Bounds ϕ_0 and ϕ_1 with $\phi_0 \geq 0$, $\phi_1 \geq 0$ and $\phi_0 + \phi_1 < q$.
Output: Arithmetic sharing $\boldsymbol{x}^{A_q}$ with uniformly distributed x such that $-\phi_0 \leq x \leq \phi_1 \bmod q$.

1: $k \leftarrow \lceil \log_2(\phi_0 + \phi_1 + 1) \rceil$
2: $i \leftarrow 0$
3: **while** $\neg\texttt{SecLeq}_{\phi_0+\phi_1}^d\left(\boldsymbol{x}_i^{B,k}\right)$ **do**
4: $\quad i \leftarrow i + 1$
5: $\boldsymbol{x}^{A_q} \leftarrow \texttt{SecB2AModp}_q^d\left(\boldsymbol{x}_i^{B,k}\right)$
6: $\boldsymbol{x}^{A_q}[0] \leftarrow \boldsymbol{x}^{A_q}[0] - \phi_0 \bmod q$

Correctness. We first note that the output sharing should be uniform on a continuous range $[-\phi_0, \phi_1]$ which contains $\phi_0 + \phi_1 + 1$ integers.

This range can be represented with k-bits such that $\phi_0 + \phi_1 + 1 \leq 2^k$. The first step in Algorithm 10 is to convert its uniformly distributed input bits into shares $\boldsymbol{x}_i^{B,k}$ while the obtained x is strictly larger than $\phi_0 + \phi_1$. This inequality is checked by leveraging `SecLeq` described in Algorithm 8. Once the inequality is not satisfied, the obtained x is uniformly distributed on the range $0 \leq x \leq \phi_0 + \phi_1$. The Boolean sharing of x is then converted into an arithmetic sharing $\boldsymbol{x}^{A_q}$. Finally, $-\phi_0 \bmod q$ is added to this arithmetic sharing, resulting in x being uniformly distributed over $-\phi_0 \leq x \leq \phi_1 \bmod q$.

Proposition 3. *Algorithm 10 is PINI assuming that whether each x_i satisfies $x_i > \phi_0 + \phi_1$ is public information and that $x_{i^*} \leq \phi_0 + \phi_1$ for some integer i^*.*

Proof. The assumptions imply that the output value of the `SecLeq` gadget calls are public and that the gadget terminates with the number of iterations being public. Therefore, the gadget `SecSampleModp` can be viewed as a circuit composed of PINI gadgets, hence Algorithm 10 is itself PINI. □

Usage in Dilithium. `SecSampleModp` is used both for key generation and signing. First, during `ExpandS` in key generation, a secret key coefficient x in $\mathbf{s_1}$ or $\mathbf{s_2}$ is sampled such that $-\eta \leq x \leq \eta$ where $\eta \in \{2, 4\}$ depending on the Dilithium parameter set. This sampling can be masked with $\texttt{SecSampleModp}^d_{q,\eta,\eta}(\cdot)$. For these parameters, rejections can occur and a fresh x passes the `SecLeq` check with probability $\frac{5}{8}$ and $\frac{9}{16}$, respectively. Second, during `ExpandMask` signature generation, a coefficient x of $\mathbf{y}$ is sampled such that $-\gamma_1 < x \leq \gamma_1$ where γ_1 is a power of two such that $\gamma_1 \in \{2^{17}, 2^{19}\}$ depending on the parameter set. As a result, this sampling can be masked thanks to $\texttt{SecSampleModp}^d_{q,\gamma_1-1,\gamma_1}(\cdot)$. For these parameters, no resampling of x is required. Indeed, $\texttt{SecLeq}^d_{2\gamma_1-1}(\cdot)$ is used which satisfies the trivial condition $\phi \geq 2^k - 1$ since $\phi = 2\gamma_1 - 1$. The k must not be evaluated at run time since it is directly derived from the Dilithium parameter set. As an example for the `ExpandMask` execution during signature generation, $k \in \{18, 20\}$ depends on the parameter set. We note that in both `ExpandS` and `ExpandMask`, the x is sampled from the output of a hash function, which is most efficiently protected using Boolean masking. This explains why we consider only sampling in the Boolean domain.[13] Moreover, whether the samples have to be rejected is public information in the original security proof of Dilithium.

[13] An alternative solution is to leverage arithmetic to arithmetic conversion. It would require to first generate a random sharing $\boldsymbol{x}^{A_{\phi_0+\phi_1}}$ resulting in a uniform x modulus $\phi_0 + \phi_1$. Then, that sharing of x must be converted to arith-

`SecDecompose`

The `SecDecompose` gadget presented in Algorithm 11 enables to compute the decomposition (w_1, w_0) of a coefficient w such that $w = \alpha w_1 + w_0 \bmod q$ with $w_0 = w \bmod^{\pm} \alpha$. Concretely, we leverage the fact that w_1 can be leaked to the adversary since it is computed during signature verification, and hence must not be protected against side-channel attacks. The first step of our gadget is to derive w_1 from $\boldsymbol{w}^{A_q}$. Then, $\boldsymbol{w_0}^{A_q}$ is obtained by computing $\boldsymbol{w_0}^{A_q} = \boldsymbol{w}^{A_q} - \alpha \cdot w_1 \bmod q$. To the best of our knowledge, there is no generic and efficient method for a masked division to compute $\boldsymbol{w_0}^{A_q}$ divided by α to get w_1. Hence, we next specialize the extraction of w_1 to the different parameter sets of Dilithium.

Algorithm 11 $\texttt{SecDecompose}^{d}_{q,\alpha}\left(\boldsymbol{w}^{A_q}\right)$

Input: Arithmetic sharing $\boldsymbol{w}^{A_q}$, prime integer q with $q < 2^k$ integer α with $0 < \alpha < q$ and $\alpha = 2\gamma_2$.
Output: Arithmetic sharing $\boldsymbol{w_0}^{A_q}$ and integer w_1 such that $w = \alpha w_1 + w_0 \bmod q$.

1: **if** NIST Level 3 or Level 5 **then**
2: $\quad \boldsymbol{b}^{A_q} \leftarrow \boldsymbol{w}^{A_q} + \gamma_2 \bmod q$
3: $\quad \boldsymbol{b'}^{A_q} \leftarrow \alpha^{-1} \cdot \boldsymbol{b}^{A_q} - 1 \bmod q$
4: $\quad \boldsymbol{b'}^{B,k} \leftarrow \texttt{SecA2BModp}^{d}_{q}\left(\boldsymbol{b'}^{A_q}\right)$
5: $\quad \boldsymbol{w_1}^{B,k'} \leftarrow \boldsymbol{b'}^{B,k}[[0, k'[]$
6: **else** ▷ NIST Level 2
7: $\quad \boldsymbol{w_1}^{B,k'} \leftarrow \texttt{SecCompress}^{d}_{q,-\alpha^{-1}}\left(\boldsymbol{w}^{A_q}\right)$
8: $w_1 \leftarrow \texttt{SecUnMask}^{d}_{k'}\left(\boldsymbol{w_1}^{B,k'}\right)$
9: $\boldsymbol{w_0}^{A_q} \leftarrow \boldsymbol{w}^{A_p} - \alpha \cdot w_1 \bmod q$

Correctness Level 2. For the NIST level 2 parameters of Dilithium, we have $\alpha = (q-1)/44$. Hence, $\alpha^{-1} = -44 \bmod q$. For these parameters, w_1 can be extracted by performing a division with its reminder such

metic sharing with modulus q such as $\boldsymbol{x}^{A_q}$. Yet to our knowledge, there exists no arithmetic-to-arithmetic conversion more efficient than the combination of `SecLeq` and `SecB2AModp`.

that:

$$\lfloor \frac{\alpha w_1 + w_0}{\alpha} \rceil = \lfloor (\alpha w_1 + w_0) \cdot \frac{-\alpha^{-1}}{q-1} \rceil, \tag{3.1}$$

$$\approx \lfloor (\alpha w_1 + w_0) \frac{-\alpha^{-1}}{q} \rceil, \tag{3.2}$$

which in turn can be performed by using the `Compress` function defined as:

$$\texttt{Compress}(x, \delta, q) = \lfloor \frac{x \cdot \delta}{q} \rceil \bmod \delta, \tag{3.3}$$

for which a masked version at any order is presented in [Cor+23]. The `Compress` function can be used since $-\alpha^{-1} \ll q$. Hence, the error does not have an impact on the results. This fact has been checked exhaustively for all possible values of $w \bmod q$.[14]

Correctness Level 3 & Level 5. Next, we check the correctness of `SecDecompose` for NIST level 3 and level 5 parameters. In such cases, $\alpha = (q-1)/16$. Hence, $\alpha^{-1} = -16 \bmod q$. The first steps in Algorithm 11 execute the following processing to w in order to derive b' such that;

$$b' = \alpha^{-1} \cdot ((\alpha w_1 + w_0) + \frac{\alpha}{2}) - 1 \bmod q, \tag{3.1}$$

which can be alternatively expressed as:

$$b = w_1 + \alpha^{-1} \cdot (w_0 - \frac{\alpha}{2}) \bmod q, \tag{3.1}$$

$$= w_1 + 16 \cdot (\frac{\alpha}{2} - w_0) \bmod q. \tag{3.2}$$

There, we note that $\frac{\alpha}{2} - w_0$ is strictly positive thanks to the definition of `Decompose`. Indeed, it follows from $-\alpha/2 \leq w_0 \leq \alpha/2$. As a result, w_1 can simply be contained in the 4 LSBs of the binary representation of b'. This is done thanks to the combination of $\texttt{SecA2BModp}^d_q$ and just keeping the 4 LSBs of the output.[15]

Proposition 4. *Algorithm 11 is PINI if w_1 is public.*

Proof. If w_1 is public, Algorithm 11 is the composition of PINI gadgets hence it is PINI. □

[14] [Cor+23] only considers δ equals a power of two. We take δ as an arbitrary positive number.

[15] We note that only the LSBs of the $\texttt{SecA2BModp}^d_q$ have to be explicitly computed. As a result, this can save several `SecAnd` when the $\texttt{SecA2BModp}^d_q$ from [BC22a] is used.

3.2.4 Implementation

We now discuss the different designs we compare later in Subsection 3.2.5. We describe the implementations for both the deterministic and the randomized versions of Dilithium. All our results are based on modified versions of the Dilithium implementations provided by the PQM4 project [Kan+]. These are C implementations with optimized assembly for polynomial arithmetic and hash functions. To prevent side-channel attacks, we follow our previous sensitivity analysis that distinguishes sensitive and non-sensitive operations, and we make use of masking with the gadgets presented in Subsection 3.2.3 and the underlying masked additions and conversion gadgets such as `SecA2BModp`, `SecAdd`, `SecAddModp` or `SecB2AModp` for all sensitive ones. We rely on their state-of-the-art bitsliced implementations introduced by Bronchain and Cassiers, which offer (to the best of our knowledge) the best performances on Cortex-M4 [BC22a]. Eventually, we use the same masked `Keccak` as the one provided in [BC22a]. We additionally leverage arithmetic masking with q modulus for all the polynomial operations and then apply share-wise the optimized polynomial arithmetic from the PQM4 implementations. Interestingly, the smaller modulus q' approach for the `NTT` in $\mathbf{s_1} \circ c$ proposed in [Abd+22] could also be used. However, it requires arithmetic to arithmetic masking conversion from q' to q to perform the addition with masked $\mathbf{y}$. We leave the study of such a trade-off for future works.

Deterministic Dilithium

We use a masked Keccak for $\mathtt{H}(K||\mu)$ and within `ExpandMask`. In `ExpandMask`, the randomness generation in `SecSampleModp` (see Algorithm 10 Line-3) is performed with a call to the masked `XOF`. The multiplication $\mathbf{Ay}$ is performed on each of the shares of $\mathbf{y}$ independently by leveraging the optimized arithmetic operations in [Kan+]. For the $\mathbf{w}$ decomposition, we leverage the new gadget `SecDecompose` from Algorithm 11 with the appropriate parameters given in Section 3.2.3. The protected rejections $\|\mathbf{z}\|_\infty < \gamma_1 - \beta$ and $\|\tilde{\mathbf{r}}\|_\infty < \gamma_2 - \beta$ are implemented thanks to `SecBoundCheck` presented in Algorithm 9. Eventually, similarly to `SecLeq` in Algorithm 9, we unmask the public signature $\mathbf{z}$ and $\tilde{\mathbf{r}}$ using the `SecUnMask` gadget once all the bound checks are passed, to maintain probing security.

Randomized Dilithium

The implementation we consider of the randomized version is similar to the deterministic one previously described. The main difference lies in the sampling of the randomness $\mathbf{y}$ within `ExpandMask`. The randomized version of Dilithium enables more freedom in the generation of the uniform polynomial vector $\mathbf{y}$ compared to the deterministic one. For the deterministic version, the randomness sampling in Algorithm 10 is based on secured `XOF` (`Keccak`) which can be a performance bottleneck (as detailed in Table 3.4). For the randomized version, a first option (which follows the specifications but does not bring performance improvements) is to generate ρ' with a TRNG and to use the masked `XOF` to derive $\mathbf{y}$. Alternatively, one can directly generate the shares of $\mathbf{y}$ with the TRNG. This does not follow the specifications of Dilithium but saves the cost of a masked `XOF` and does not weaken the security of Dilithium as $\mathbf{y}$ remains uniform. We next evaluate this option.[16]

3.2.5 Benchmarks

In this section, we report the performances of the Dilithium implementations are described in Subsection 3.2.4. We first detail the benchmarking setup used for this purpose. Second, we report the performance improvements provided by the new gadgets of Subsection 3.2.3 compared to the ones of [Mig+19]. Then, we evaluate the cost of each operation in Dilithium's signature generation (without considering the rejections). Based on this, we compare the performance of both deterministic and randomized versions when side-channel countermeasures are required. Performances are given for Dilithium with Level-3 parameters (see Table 3.3), but the general conclusions apply to all security levels.

Benchmarking setup

To evaluate the execution time of our implementations, we use a similar benchmarking setup as the one provided in [BC22a], which itself is based on the `PQM4` benchmarking initiative for PQC signatures and KEMs [Kan+]. More precisely, the benchmarks are performed with the NUCLEO-L4R5ZI demonstration board. The cycle counts are measured thanks to the cycle-accurate counter `DWT_CCYCNT`. With the considered clock configuration, the TRNG of the microcontroller provides 32 fresh random bits every 53 Cycles. This TRNG is used for the generation of

[16] Yet another alternative, in case of weak TRNG, is to generate shares of $\mathbf{y}$ by applying the unmasked `XOF` to TRNG outputs. This option saves the cost of masking the `XOF`.

the randomness masking as well as for the `ExpandMask` in the randomized version of Dilithium that we evaluate.

Gadgets improvements

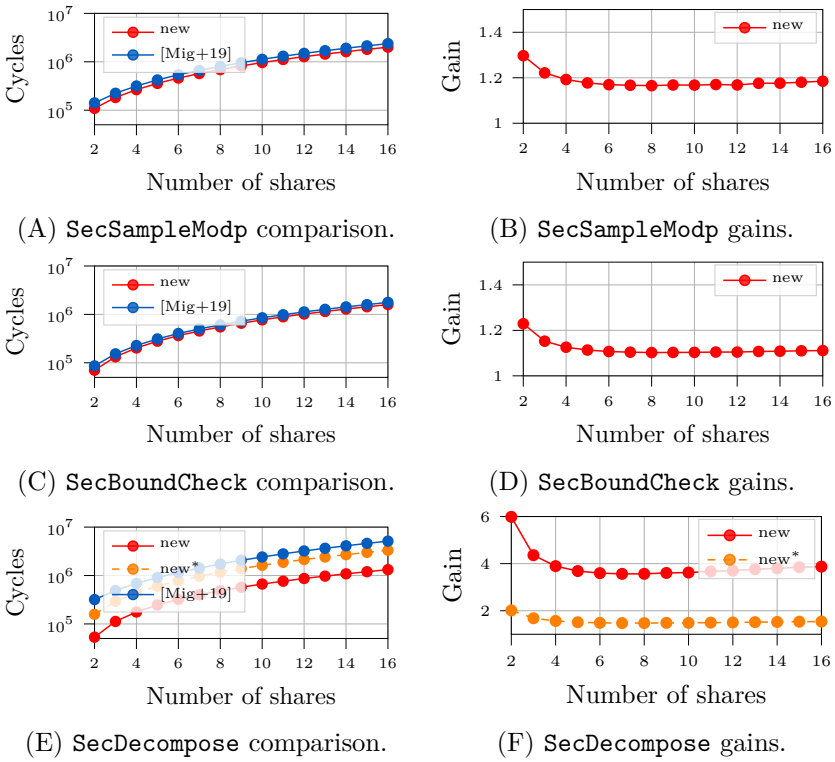
We first compare the gadgets presented in Subsection 3.2.3 and the ones proposed by Migliore et al. in [Mig+19], and we report the results in Figure 3.9. To enable a fair comparison, we implemented the gadgets as described in [Mig+19] by leveraging the PINI property and the bitslice gadgets from [BC22a] for `SecAdd`, `SecAddModp`, `SecA2BModp` and `SecB2AModp`. As a result, the implementation of [Mig+19] we consider does not contain extra refresh gadgets (as it is PINI). We note that [Mig+19] uses a parameter w reflecting the bus width of the target CPU, which implies that operations might be performed on more bits than necessary. In our implementation, we do not use that parameter w as the operations are performed on the exact necessary number of bits as allowed by bitslicing. Similarly, [Mig+19] performed masked `SecAnd` with public values to isolate bits on secret variables. Individual bits are isolated in our implementations thanks to bitslicing.

`SecSampleModp`. Both versions of `SecSampleModp` as used in the signature generation are similar. The only difference is that the subtraction with ϕ_0 is performed with Boolean masking in [Mig+19] and with arithmetic masking in Algorithm 10. As a result, our new gadget saves the cost of one `SecAdd` by replacing it with a share-wise addition. This results in a speedup of an approximate factor 1.2, as highlighted in Figure 3.9B.

`SecBoundCheck`. Our `SecBoundCheck` also simplifies the one proposed in [Mig+19] where a `SecA2BModp` is performed followed by two `SecAdd`'s.[17] Our construction replaces one of these additions with one arithmetically masked addition, which is almost free. This leads to a performance improvement by a factor ≈ 1.1, as reported in Figure 3.9D.

`SecDecompose`. Finally, we compare the two implementations of `SecDecompose`. Interestingly, the main improvement comes from the fact that we first extract $\mathbf{w}_1$ efficiently and then unmask it to compute $\mathbf{w}_0$. This improvement relies on the fact that the higher order bits (e.g., $\mathbf{w}_1$) of the `SecDecompose` gadget are considered as sensitive by Migliore et al. while

[17] Only `SecBoundCheck` is described for Boolean sharing in [Mig+19]. Here we assume that a `SecA2BModp` is performed before the end to match Algorithm 9 specifications.



(A) `SecSampleModp` comparison. (B) `SecSampleModp` gains.

(C) `SecBoundCheck` comparison. (D) `SecBoundCheck` gains.

(E) `SecDecompose` comparison. (F) `SecDecompose` gains.

Figure 3.9: Comparison between our new gadgets and [Mig+19] for NIST Level-3 parameters. The label `new`* corresponds to the version where $\mathbf{w}_1$ remains masked.

it is not necessary as detailed in the revisited sensitivity analysis detailed in Section 3.2.2. In short, the implementation based on [Mig+19] starts with a `SecA2BModp`, continues with several (≈ 10) additions and finally performs a `SecB2AModp` to obtain the arithmetic sharing of $\mathbf{w}_0$. The new gadget only requires a single `SecA2BModp` and some share-wise operations with arithmetic masking, and runs ≈ 3.8 times faster.

We note that for Level-2 parameters, the α changes and the gadget from Migliore et al. does not apply. Our implementation of `SecDecompose` for Level-2 parameters is slightly slower than for Level-3 and Level-5. Indeed, in the `SecCompress`, the `SecA2BModp` must be performed on a slightly larger modulus increasing the cost by a factor ≈ 1.2.

Deterministic vs. randomized performances

The performances of each operation within both versions of Dilithium are reported in Table 3.4. We observe that the randomized signature generation is more efficient than the deterministic one. For two shares, 24 005 kCycles are needed for the deterministic version vs. only 14 282 kCycles for the randomized one. Hence, randomization offers an improvement by a factor ≈ 1.68. Similarly, for 8 shares, the randomized version is $\approx 1.77\times$ faster than the deterministic one. The run time of unprotected Dilithium3 signature generation is 3224 kCycles [Kan+]. Hence, the two-share version is $4.3\times$ slower than the unprotected implementation in the randomized case and $7.32\times$ in the deterministic case. Most of the difference between the two Dilithium versions is due to `ExpandMask`, which is composed of two parts as detailed in Algorithm 10. The first one is sampling the uniform $\mathbf{y}$ in Boolean masking. This operation is performed with a masked `XOF` in the deterministic case, and with the on-board TRNG in the randomized case. The second part is to perform a `SecB2AModp` to produce an arithmetic sharing. This operation is similar for both cases. In the deterministic case, the `ExpandMask` represents 56 % of the total run time from which 74 % are due to the masked `XOF`. As the randomized Dilithium does not require this masked `XOF`, only 25 % are imputable to `ExpandMask`.

The cost of the other operations are similar for both versions. Concretely, the overall cost is dominated by the operations that have quadratic overheads in the number of shares (even for $d = 2$). These operations are $\texttt{H}(K||\mu)$, `ExpandMask`, `SecDecompose`, `SecBoundCheck` and `UnMask`. We note that for the deterministic version, the most expensive operation is `ExpandMask`, while it is `SecBoundCheck` for the randomized version. Additionally, the cost of polynomial arithmetic ($\texttt{NTT}(s)$, $\mathbf{y} + \mathbf{s}_1 c$ and $\mathbf{w}_0 - \mathbf{s}_2 c$) is limited. As expected, the cost of the public matrix expansion `ExpandA` remains constant with the number of shares. Eventually, we note that the `SecB2AModp` is slightly more expensive in the deterministic case, as it includes the linear overheads needed to map the output of the masked `XOF` into the correct bitslice representation. This operation is not needed in the randomized case as the output of the TRNG already has the appropriate layout.

More generally, we also stress that the randomized version does not allow the adversary to average traces for the same inputs, which is beneficial for security. The combination of these observations and performance gains naturally calls for considering the randomized Dilithium in application contexts where side-channel attacks are a concern.

Table 3.4: Performances of the masked Dilithium Level-3 components for randomized and deterministic versions: number of clock cycles are given when running on a `STM32L4R5` and using the TRNG for generating the masking randomness (32-bit randomness every 53 Cycles). Reported numbers are in `kCycles`. The numbers are for a single execution of the component (and do not consider repetitions due to rejections). Rand. *: The vector polynomial $\mathbf{y}$ is sampled from the TRNG and not from a `XOF`.

d	2		4		6		8	
	Deter.	Rand.*	Deter.	Rand.*	Deter.	Rand.*	Deter.	Rand.*
`Sign`	23,613.8	13,891.4	68,557.9	39,064.5	128,280.9	73,905.4	199,567.4	111,515.4
`NTT`(s)	185.4	185.4	370.7	370.7	556.3	556.2	741.7	741.7
`ExpandA`	2,160.7	2,160.7	2,160.7	2,160.7	2,160.7	2,160.7	2,160.7	2,160.7
`H`($K\|\|\mu$)	370.1	0.0	1,126.7	0.0	2,082.8	0.0	3,377.8	0.0
`ExpandMask`	12,551.1	3,198.6	38,068.8	9,702.2	70,788.2	18,514.6	113,185.0	28,498.6
`SecB2AModp`	3,350.2	3,104.8	9,993.8	9,516.7	18,967.1	18,235.5	29,081.0	28,128.4
Sampling	9,193.9	79.7	28,062.1	165.9	51,802.5	253.7	84,079.6	339.0
Ay	382.2	382.3	764.4	764.4	1,146.7	1,146.6	1,528.8	1,528.8
`Decompose`	2,369.0	2,369.1	8,142.3	8,142.3	16,382.8	16,376.8	25,076.9	25,080.8
`H`($\mu\|\|\mathbf{w}_1$)	94.9	94.9	94.9	94.9	94.9	94.9	94.9	94.9
$\mathbf{y} + \mathbf{s}_1 c$	174.9	174.9	349.7	349.7	524.6	524.6	699.5	699.5
$\mathbf{w}_0 - \mathbf{s}_2 c$	209.9	210.0	419.6	419.7	629.4	629.5	839.3	839.4
`SecBoundCheck`	4,354.6	4,354.6	15,472.8	15,472.8	31,234.4	31,221.3	48,376.4	48,386.4
`UnMask`	378.3	378.4	1,185.7	1,185.7	2,260.7	2,260.6	3,046.8	3,047.0

Protecting $\mathbf{w}_1$ with masking

We finally discuss the cost of a more conservative implementation keeping $\mathbf{w}_1$ masked before the bound checks on $\mathbf{z}$ and $\tilde{\mathbf{r}}$ are passed. The resulting block diagram for signature generation is given in Figure 3.10, where the only difference with Figure 3.8 is that both $\mathbf{w}_1$ and `H`$(\mu||\mathbf{w}_1)$ are in red (i.e., sensitive). This change implies that $\mathbf{w}_1$ must remain masked in `SecDecompose` (see Algorithm 11). Hence, `SecDecompose` is adapted to keep $\mathbf{w}_1$ masked by performing a `SecB2AModp` on its Boolean sharing (instead of unmasking). The computation of $\mathbf{w}_0$ can then be applied share-wise similarly as in Algorithm 11, Line-9. The resulting increase in the run time of `SecDecompose` is detailed in Figure 3.9.

The impact of protecting $\mathbf{w}_1$ on the full signature is detailed in Table 3.5. The main difference with Table 3.4 is that the cost of `SecDecompose` increases as discussed above. A second difference is that the hash function `H`$(\mu||\mathbf{w}_1)$ now also needs to be masked. As a result, this hashing represents a total of 9 % of the run time for deterministic signatures, and 13.7 % for randomized ones. Hence, protecting $\mathbf{w}_1$ decreases the benefit of randomized signatures, from 66 % of the run time when $\mathbf{w}_1$ does not need to be protected to 56 % when it has. Overall, the run time

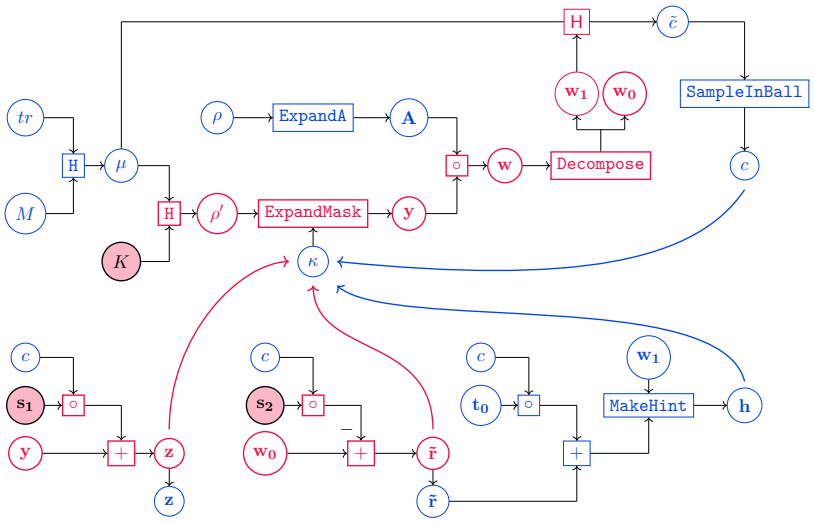


Figure 3.10: Graphical representation of the signing procedure with masked $\mathbf{w}_1$, taking as input sk, M and outputting $\sigma = (\tilde{c}, \mathbf{z}, \mathbf{h})$. Curved arrows represent rejection checks. Red filled circles: sensitive variables. Blue: no side-channel protection required.

of deterministic (resp., randomized) signatures increases by a factor 1.3 (resp., 1.53) when we need to protect $\mathbf{w}_1$ with masking.

3.2.6 Conclusion and open problems

In this work, we analyzed side-channel protected implementations of Dilithium by mixing different contributions. First, we presented an updated sensitivity analysis for its key generation and signing algorithms. Our results show that a previous work in this direction was slightly flawed, with some parts leading to insecurities and other parts leading to inefficiencies. Second, our new masking gadgets improve over the state-of-the-art, leading to performance gains of factors up to 3.8. They also fill gaps for which it was previously unknown how to efficiently apply masking and we propose the first masking gadgets that are compatible with all the Dilithium parameter sets. Overall, our analysis and benchmark highlight that the randomized variant of Dilithium evaluated in this section provides notably better performances, thanks to additional flexibility in the sampling of random values. In addition, it also offers a smaller side-channel attack surface as signatures cannot be repeated. We therefore believe that it should be the default variant for embedded devices when side-channel leakage needs to be taken into account.

Table 3.5: Performance of the masked Dilithium Level-3 components for randomized and deterministic versions with masked $\mathbf{w}_1$: number of clock cycles when running on a `STM32L4R5` and using the TRNG for generating the masking randomness (32-bit randomness every 53 Cycles). Reported numbers are in `kCycles`. The numbers are for a single execution of the component (does not consider repetitions due to rejections). Rand. *: The vector polynomial $\mathbf{y}$ is sampled from the TRNG and not from a `XOF`.

d	2		4		6		8	
	Deter.	Rand.*	Deter.	Rand.*	Deter.	Rand.*	Deter.	Rand.*
`Sign`	30,254.8	20,532.1	88,912.9	59,512.8	166,748.8	112,397.1	259,743.1	171,692.1
`NTT`(s)	185.4	185.4	370.7	370.8	556.2	556.2	741.6	741.7
`ExpandA`	2,160.7	2,160.7	2,160.7	2,160.7	2,160.7	2,160.7	2,160.7	2,160.7
`H`$(K\|\|\mu)$	370.1	0.0	1,126.7	0.0	2,082.8	0.0	3,377.8	0.0
`ExpandMask`	12,551.3	3,198.6	38,057.4	9,712.6	70,783.2	18,514.4	113,185.7	28,495.5
`SecB2AModp`	3,350.3	3,104.8	9,982.3	9,528.9	18,962.0	18,235.4	29,081.7	28,126.1
Sampling	9,193.9	79.9	28,062.1	164.1	51,802.5	253.7	84,079.6	338.4
Ay	382.3	382.2	764.4	764.4	1,146.6	1,146.6	1,528.7	1,528.8
`Decompose`	6,527.5	6,527.4	20,767.0	20,813.1	40,448.2	40,447.9	61,779.3	61,790.4
`H`$(\mu\|\|\mathbf{w}_1)$	2,576.9	2,576.9	7,860.8	7,860.8	14,514.4	14,514.5	23,560.8	23,560.8
$\mathbf{y} + \mathbf{s}_1 c$	174.9	174.9	349.7	349.7	524.5	524.6	699.5	699.4
$\mathbf{w}_0 - \mathbf{s}_2 c$	209.9	209.9	419.7	419.8	629.5	629.5	839.3	839.3
`SecBoundCheck`	4,354.7	4,354.7	15,447.0	15,471.9	31,221.1	31,221.1	48,380.3	48,389.1
`UnMask`	584.8	584.7	1,832.4	1,832.6	3,493.6	3,493.6	4,708.8	4,708.9

These results lead to many natural open problems. First, they highlight that for now, the leveling concept (i.e., the idea of protecting different parts of an implementation with different countermeasures, in order to limit the overheads) cannot be fully exploited for Dilithium, as initially thought (see the cautionary note in this section's introduction. For example, most of the signature operations in our implementations need to be secure against DPA, which requires (expensive) masked gadgets. Hence it is a natural open question to find out whether more leveled implementations could be obtained, which could be considered in different fashions. A light leveling option, directly applicable to Dilithium, would be to try exploiting that even when masking, all operations may not leak similarly. For example, one could try leveraging the recent observation that prime masking is more resilient to low-noise leakages than Boolean masking, and study whether the number of shares in the Boolean and arithmetic masking used for protecting Dilithium against leakage could be leveled [Mas+23]. A more ambitious direction would be to study how to enable a stronger leveling again (i.e., mixing operations that require security against SPA and operations that require security against DPA). One potential direction could be to rely on hard physical learning problems like introduced in [Duv+21], possibly at the cost of some design tweaks in Dilithium (e.g., to deal with the challenges raised by the ma-

nipulation of non-uniform and low-norm secrets). In general, designing a PQ signature scheme with a better performance vs. side-channel security tradeoff appears as an interesting long-term goal. Besides, our work only focuses on side-channel attacks and therefore raises the question of how to additionally protect Dilithium against fault attacks, and whether its randomized version also provides (security or performance) benefits in this context, which is yet another interesting direction for further research.

The source codes of the implementations described in this work are available from the following link: `https://github.com/uclcrypto/pqm4_masked/tree/master`.

3.3 Designing POLKA, a PQC KEM tailored for physical security, adapted from [Hof+22]

Recent research efforts showed that designing post-quantum chosen-ciphertext-secure public-key encryption (PKE) schemes that allow efficient implementations offering side-channel security guarantees is extremely challenging with existing techniques. One well-documented issue arises from the Fujisaki-Okamoto (FO) transform that is frequently used for building key encapsulation mechanisms (KEMs) with chosen-ciphertext (IND-CCA) security from PKE schemes or KEMs that only provide weak security notions like one-wayness under passive attacks (OW-CPA security) [FO99; FO13]. The FO transformation and its variants are, for example, used in the NIST post-quantum finalists KYBER [Bos+18a; Ava+20] and SABER [D'A+18a; Bas+19], where the CCA-secure KEM is combined with a secret-key (authenticated) encryption scheme into a hybrid PKE system.

Recall that a KEM system $(\mathsf{Keygen}, \mathsf{Encaps}, \mathsf{Decaps})$ is a PKE scheme that does not take any plaintext as input, but rather computes an encryption of a random symmetric key K. To encrypt a plaintext M via the hybrid KEM/DEM framework [Sho01], the Encaps algorithm often samples a random m, which is used to derive a symmetric key K and random coins r from a random oracle $(K, r) \leftarrow H(m)$ before deterministically encapsulating K as $c_{kem} = \mathsf{Encaps}_{pk}(m, r)$. Next, a secret-key scheme (E, D) (a.k.a. data encapsulation mechanism, or DEM) is used to compute $c_{sym} = \mathsf{E}_K(M)$ in order to obtain a hybrid PKE ciphertext $c = (c_{kem}, c_{sym})$. The receiver can then recover $m = \mathsf{Decaps}_{sk}(c_{kem})$ and $(K, r) \leftarrow H(m)$ before obtaining $M = \mathsf{D}_K(c_{sym})$. It is known that the hybrid construction provides IND-CCA security if the underlying KEM is itself IND-CCA-secure and if the DEM satisfies a similar security notion in the secret-key setting [Sho01]. In order to secure the KEM part

against chosen-ciphertext attacks, the FO transform usually checks the validity of the incoming c_{kem} by testing if $c_{kem} = \mathsf{Encaps}_{pk}(m, r)$ (a step known as "re-encryption") after having recovered the random coins r from $(K, r) \leftarrow H(m)$ upon decryption.

In the FO transform, the first computation during a decryption attempt is $\mathsf{Decaps}_{sk}(c_{kem})$, where Decaps is the underlying decapsulation of the OW-CPA secure KEM. While this has no impact in a black-box security analysis, in the context of side-channel chosen-ciphertext attacks the adversary remains able to target this component using many c_{kem} values [Rav+20b; Ngo+21; Uen+22] of its choice, leaving an important source of vulnerabilities. Indeed, the adversary is free to adaptively feed Decaps_{sk} with (invalid) ciphertexts and craft c_{kem} in such a way that an internal message m with only few unknown bits is re-encrypted via the FO transform. This allows side-channel attacks to directly exploit the leakage of these bits obtained during the re-encryption test $c_{kem} \stackrel{?}{=} \mathsf{Encaps}_{pk}(m, r)$ to infer information about sk. This task is surprisingly easy since all the leakage samples of the deterministic re-encryption can be exploited for this purpose (i.e., much more than the few rounds of leakage that are typically exploited in divide-and-conquer side-channel attacks against symmetric encryption schemes) [MOP07].

In parallel, several pieces of work started to analyze masked implementations of KYBER and SABER [Bei+21; Bha+21; Bos+21; Fri+22]. These works typically indicate large overheads when high security levels are required, which can be directly connected to a large amount of leaking intermediate computations [Azo+22a]. In particular, these implementations all consider a uniform protection level for all their operations, that is in contrast with the situation of symmetric cryptography where so-called leveled implementations, in which different (more or less sensitive) parts of a mode of operation are protected with different (more or less expensive) side-channel countermeasures, can lead to important performance gains [Bel+20].

In this paper, we therefore initiate the study of quantum-safe CCA-secure public-key encryption schemes that have good features for leakage-resistant (LR) implementations. The seed ingredients we propose for this purpose are threefold. First, we leverage the *rigidity* property introduced by Bernstein and Persichetti [BP18], as it allows building CCA-secure encryption schemes without relying on re-encryption nor on the FO transform. Despite removing an important source of leakage, getting rid of the FO transform is not yet sufficient to enable leveled implementations for KYBER (or SABER), since the rest of their operations remains expensive to protect [Azo+22a]. Therefore, we also propose to

randomize the decryption process by incorporating a "dummy ciphertext". It brings the direct benefit of removing the adversary's control on all intermediate computations that are dummied, while making these computations ephemeral, which is in general helpful against leakage. This second step already allows an interesting leveling between computations that require security against simple power analysis (SPA) and differential power analysis (DPA) attacks.[18] Eventually, we observe that the structure of the KEM's remaining DPA target shares similarities with the key-homomorphic re-keying schemes used in symmetric cryptography to prevent side-channel attacks [Med+10; Dob+15; Dzi+16]. Building on this observation, we propose to implement this DPA target such that only its key-homomorphic parts are (efficiently) protected thanks to masking, by relying on the recently introduced Learning With Physical Rounding (LWPR) assumption [Duv+21]. In short, the LWPR assumption is a physical version of the crypto dark matter introduced by Boneh *et al.* [Bon+18]. The latter assumes that low-complexity PRFs can be obtained by mixing linear mappings over different small moduli. LWPR further leverages the possibility that one of these mappings is computed by a leakage function.

We additionally observe that by carefully instantiating the symmetric authenticated encryption scheme of the DEM as an Encrypt-then-MAC one with a one-time key-homomorphic MAC, the overheads due to the side-channel countermeasures can be reduced to linear in the number of shares used for masking for this part of the computation. And we finally combine these different ingredients into a new efficient post-quantum CCA-secure public-key encryption scheme, called POLKA (standing for POst-quantum Leakage-resistant public Key encryption Algorithm), that *simultaneously* provides excellent features against leakage and a proof of IND-CCA security (in the sense of the standard definition without leakage) under the standard RLWE assumption.

Without leakage, we show that POLKA provides CCA security in the quantum random oracle model (QROM) [Bon+11]. Our construction is a hybrid KEM-DEM encryption scheme built upon a variant of a public-key encryption scheme due to Lyubashevsky, Peikert and Regev (LPR) [LPR10], which is well-known to provide IND-CPA security under the ring learning-with-errors (RLWE) assumption. In order to obtain a KEM, we modify the LPR system so as to recover the sender's random coins upon decryption. In contrast with the FO transformation and its variants, this is achieved without derandomizing an IND-CPA system,

[18] Informally, SPAs are side-channel attacks where the adversary can only observe the leakage of a few inputs to the target operation for a given secret. DPAs are attacks where the adversary can observe the leakage of many such inputs.

by deriving the sender's random coins. Instead of encrypting a random message m to derive our symmetric key K, we always "encrypt" 0 and hash the random coins consisting of a tuple $(r, e_1, e_2) \in R$ of small-norm ring elements sampled from the noise distribution. These elements (r, e_1, e_2) are then encoded into a pair $c_{kem} = (a \cdot r + e_1, b \cdot r + e_2) \in R_q^2$, where $a, b \in R_q$ are random-looking elements included in the public key. Using its secret key, the decryptor can extract (r, e_1, e_2) from c_{kem} and check their smallness. This verification/extraction step is designed in such a way that decapsulation natively provides rigidity [BP18] without relying on re-encryption. Namely, due to the way to recover (r, e_1, e_2) from c_{kem}, we are guaranteed that deterministically re-computing $c_{kem} = (a \cdot r + e_1, b \cdot r + e_2)$ would yield the incoming ciphertext. This allows dispensing with the need to explicitly re-compute c_{kem} in the real scheme, thus eliminating an important source of side-channel vulnerability that affects KYBER and SABER.

In a black-box security analysis (that will not appear in this manuscript but can be found in the ePrint version of this work), our KEM can be seen as an injective trapdoor function that maps $(r, e_1, e_2) \in R^3$ to $(a \cdot r + e_1, b \cdot r + e_2)$. As long as we sample (r, e_1, e_2) from a suitable distribution, c_{kem} is pseudorandom under the RLWE assumption. However, to ease the use of efficient side-channel countermeasures upon decryption, we also leverage the fact that our injection satisfies a (bounded) form of additive homomorphism for appropriate parameters. That is, if we generate what we call a *dummy* ciphertext c'_{kem} by having the decryptor honestly run the basic encapsulation step using its own random coins, the decapsulation of $\bar{c}_{kem} = c_{kem} + c'_{kem}$ should give the sum of the random coins chosen by the sender and the receiver. Then, we can easily remove the additional *dummy* random coins after some additional tests. Introducing $\bar{c}_{kem}$ in the decryption process removes the adversary's freedom of forcing the computation of $\mathsf{Decaps}_{sk}(c_{kem})$ to take place on a c_{kem} under its control, which helps us protecting the secret key. Moreover, the underlying coins of c_{kem} are now split into two shares upon decryption and they are only recombined in a step where we can safely derive K. To implement this idea, we prove the CCA security of our scheme in its variant endowed with a probabilistic decryption algorithm.

With leakage, we argue that POLKA offers a natural path towards efficient leveled implementations secured against side-channel attacks. For this purpose, we first use a methodology inspired from [Bel+20] to identify the level of security required for all its intermediate computations. We then focus on how to secure the polynomial multiplication used in POLKA against DPA, by combining masking for its key-homomorphic

parts and a variant of the aforementioned LWPR assumption after the shares recombination. Our contributions in this respect are twofold. On the one hand, we define the LWPR variant on which POLKA relies and discuss its difference from the original one. Given that LWPR is an admittedly recent assumption and in view of the important performance gains it can lead to, we additionally specify instances to serve as cryptanalysis targets. Overall, protecting the long-term secret of our prototype implementation only needs to combine the masking of key-homomorphic computations (which has linear overheads in the number of shares) with SPA security for other computations, that is quite directly/cheaply obtained thanks to parallelism in hardware. An hardware architecture implementing these masked operations can be found in the ePrint version of this work, confirming the simplicity to implement and the linear overhead in the number of shares). This implementation is left out of this manuscript because slightly out of scope of my personal contributions on this paper. Protecting the message confidentiality additionally requires protecting its symmetric cryptographic components (i.e., hash function and authenticated encryption).

We insist that our leakage analysis is not (yet) about sufficient security conditions but about necessary conditions that implementers must fulfill to avoid critical attack paths. Such an analysis has been shown sufficiently informative to compare designs in the symmetric setting. We see no reason why things would significantly differ here. Formally proving the leakage-resistance of POLKA is a natural next step and an important open problem. So our goal is to show that (i) by considering the need for side-channel countermeasures as a design criterion, we can considerably limit the attack vectors, and (ii) by combining design tweaks (e.g., rigidity, dummy operations, LWPR, key-homomorphism) we can make the cost of countermeasures significantly lower than, say KYBER or SABER.

As a last result to confirm the generality of our findings, we also show that they can naturally apply to an LR variant of the NTRU cryptosystem [Che+20], which already satisfies the rigidity property, can be enhanced with a dummy mechanism and has internal computations that also generate LWPR samples.

3.3.1 Technical Overview of POLKA & Related Works

Technical overview. Our construction can be seen as a rigid and randomness-recovering version of the RLWE-based encryption scheme described in [LPR10]. By "randomness-recovering," we mean that the decryption procedure recovers the message *and* the sender's random

coins. A randomness-recovering encryption scheme is rigid [BP18] if, when the decryptor obtains a message m and randomness r, running the encryption algorithm on input of (m, r) necessarily yields the incoming ciphertext. While rigidity can always be achieved by adding a re-encryption step (as pointed out in [HHK17a]), this generally introduces one-more place of potential side-channel vulnerabilities, which is precisely exploited in [Rav+20b; Ngo+21; Uen+22]. In order to eliminate the need for an explicit re-encryption step, it is thus desirable to have a decryption algorithm which is natively injective (when seen as a deterministic function). The first difficulty is thus to build a rigid, randomness-recovering PKE/KEM under the standard RLWE assumption. Our goal is to achieve this without sacrificing the efficiency of the original LPR system while remaining reasonably competitive with NIST finalists.

The LPR cryptosystem is not randomness-recovering. In a cyclotomic ring $R = \mathbb{Z}[X]/(X^n + 1)$, it involves a public key containing a pair $(a, b = a \cdot s + e)$, where $a \in R/(qR)$ is uniform, $s \in R$ is the secret key and e is a noise. To encrypt $m \in R/(pR)$ (for some moduli $p < q$), the sender chooses small-norm randomness $r, e_1, e_2 \in R$ and computes $(c_1, c_2) = (a \cdot r + e_1, b \cdot r + e_2 + m \cdot \lfloor q/p \rceil)$, so that the receiver can obtain $c_2 - c_1 \cdot s \bmod q = m \cdot \lfloor q/p \rceil + \mathsf{small}$. While m is then computable, there is no way to recover (r, e_1, e_2) from the "decryption error" term small. To address this problem, a folklore solution is to introduce distinct powers of p. Suppose we want to build a randomness-recovering encryption of 0 (which is sufficient to build a KEM). The sender can then compute $(c_1, c_2) = (p^2 \cdot a \cdot r + p \cdot e_1, p^2 \cdot b \cdot r + e_2)$, which allows the receiver to obtain $\mu = c_2 - c_1 \cdot s \bmod q = p^2 er - pe_1 s + e_2$. Since the right-hand-side member is small, the receiver can efficiently decode $(r, e_1, e_2) \in R^3$ from μ. Unfortunately, the latter construction is not rigid. Suppose that an adversary can somehow compute a non-trivial pair $(u, u \cdot s) \in R^2$ given $(a, a \cdot s + e)$. It can then faithfully compute $(c_1, c_2) = (p^2 \cdot a \cdot r + p \cdot e_1, p^2 \cdot b \cdot r + e_2)$ and turn it into $(c_1', c_2') = (c_1 + u, c_2 + u \cdot s)$, which yields a "decryption collision" $\mu = c_2 - c_1 \cdot s = c_2' - c_1' \cdot s$. Besides, as shown in [DOV21], computing a pair $(u, u \cdot s)$ (for an arbitrary, possibly non-invertible $u \neq 0$) given $(a, a \cdot s + e)$ can only be hard in rings $R/(qR) \cong \mathbb{Z}_q[X]/(\Phi_1(X)) \times \cdots \times \mathbb{Z}_q[X]/(\Phi_t(X))$ that have no small-degree factors, which rules out NTT-friendly rings. Even for rings where $\Phi(X) = X^n + 1$ splits into degree-$n/2$ factors, the problem (called SIP-LWE in [DOV21]) is non-standard and its hardness is not known to be implied by RLWE.[19]

[19] D'Anvers *et al.* [DOV21] defined a homogeneous variant of SIP-LWE which is unconditionally hard, even in fully splitting rings. Still, relying on this variant incurs a partial re-encryption to enforce the equality $c_2 = c_2'$.

Here, we take a different approach since we aim at rigidity without relying on stronger assumptions than RLWE and without forbidding fully splitting rings.

We modify the original LPR system in the following way. The public key contains a random $a \in R/(qR)$ and a pseudorandom $b \in R/(qR)$, which is now of the form $b = p \cdot (a \cdot s + e)$, for small secrets $s, e \in R$ and a public integer p such that $\|e\|_\infty < p/2$. We also require b to be invertible over $R/(qR)$, so that the key generation phase must be repeated with new candidates (s, e) until b is a unit. To compute an encapsulation, we sample Gaussian ring elements $r, e_1, e_2 \in R$ and compute $c_{kem} = (c_1, c_2) = (a \cdot r + e_1, b \cdot r + e_2)$, where $K = H(r, e_1, e_2)$ is the encapsulated key. Decapsulation is performed by using $s \in R$ to compute $\mu = c_2 - p \cdot c_1 \cdot s \bmod q$, which is a small-norm element $\mu = e_2 + p \cdot \mathsf{small} \in R$ that reveals $e_2 = \mu \bmod p$. Given e_2, the receiver then obtains $r = (c_2 - e_2) \cdot b^{-1}$ and $e_1 = c_1 - a \cdot r$, and checks the smallness of (r, e_1, e_2). The decapsulation phase is natively *rigid* (without re-encryption) as it outputs small-norm $(r, e_1, e_2) \in R^3$ *if and only if* $(c_1, c_2) = (a \cdot r + e_1, b \cdot r + e_2)$.

Our hybrid encryption scheme builds on a variant of the above KEM with *explicit rejection*, where the decapsulation phase returns an error symbol $\perp$ on input of an invalid c_{kem}. It thus departs from NIST finalists that all rely on KEMs with implicit rejection, where the decapsulation algorithm never outputs $\perp$, but rather handles invalid encapsulations c_{kem} by outputting a random key $K' \leftarrow H(z, c_{kem})$ derived from an independent long-term secret z.[20] While our scheme could have relied on implicit rejection in a similar way, we chose to avoid additional computations involving extra secret key components z. The reason is that, if we were to introduce additional key material z, it should also be DPA-protected with possibly heavy side-channel countermeasures.

When it comes to proving security in the QROM, the use of an explicit-rejection KEM introduces some difficulty as it is not clear how to deal with invalid ciphertexts. While the classical ROM allows inspecting all random oracle queries and determining if one of them explains a given ciphertext, we cannot use this approach in the QROM because RO-queries are made on superpositions of inputs. Our solution is to use an implicit-rejection KEM only in the security proof. In a sequence of games, we first modify the decryption oracle so as to make the rejection process implicit. Then, we argue that, as long as the DEM component is realized using an authenticated symmetric encryption scheme, the

[20] When the hybrid KEM-DEM framework is instantiated with an implicit rejection KEM, invalid ciphertexts are usually rejected during the symmetric decryption step as decrypting c_{sym} with a random key K' yields $\perp$.

modified decryption oracle is indistinguishable from the real one. After having modified the decryption oracle, we can adapt ideas from Saito *et al.* [SXY18] in order to tightly relate the security of the hybrid scheme to the RLWE assumption.

As mentioned earlier, avoiding re-encryption does not suffice to ensure side-channel resistance. As a first countermeasure, we modify the decapsulation step and add a dummy ciphertext $(c'_1, c'_2) = (a \cdot r' + e'_1, b \cdot r' + e'_2)$ for fresh receiver-chosen randomness r', e'_1, e'_2 to (c_1, c_2) before proceeding with the decapsulation of $(\bar{c}_1, \bar{c}_2) = (c_1 + c'_1, c_2 + c'_2)$. This simple trick prevents the adversary from controlling the ring elements that multiply the secret key s at the only step where it is involved. We even show in Section 3.3.4 how this computation can be protected against DPA with minimum overheads by combination the masking countermeasure and a LWPR assumption. Additionally, the choice of (c'_1, c'_2) as an honestly generated encapsulation allows continuing the decryption process as if $(\bar{c}_1, \bar{c}_2)$ was the ciphertext computed from the (still) small-norm coins $(\bar{r}, \bar{e}_1, \bar{e}_2) = (r + r', e_1 + e'_1, e_2 + e'_2)$. That is, we do not have to remove the noise terms as we can retrieve $\bar{r}$, $\bar{e}_1$ and $\bar{e}_2$ and test their smallness. Since r', e'_1 and e'_2 do have small norm, if the decryption succeeds until this step, then r, e_1 and e_2 must be small as well (with a small constant slackness factor 3). Therefore, the dummy ciphertext/KEM makes it possible to eliminate an exponential amount of invalid ciphertexts without having ever tried to re-compute the correct (r, e_1, e_2). In case of an early rejection, and because the secret key s is now protected with a hidden and pseudorandom $(\bar{c}_1, \bar{c}_2)$, the leakage only provides limited information related to the ephemeral values in (r', e'_1, e'_2) which were sampled independently of the adversary's view. If no rejection occurs, (r', e'_1, e'_2) has components of (small but) sufficiently large norm to hide (at least most of the bits of) (r, e_1, e_2) if the adversary gets the full leakage of $(\bar{r}, \bar{e}_1, \bar{e}_2)$. At that time, we can safely recover (r, e_1, e_2) and check their norm (to eliminate the slackness) for technical reasons. This computation can only be repeated through many decryption queries on fixed inputs, and therefore only require SPA security (with averaging), which is cheaper to ensure than DPA security. As for the DEM, the general solutions outlined in [Bel+20] are a natural option. But we show an even cheaper one that leverages a key-homomorphic MAC.

Related work. As a KEM variant of LPR, `POLKA` bears high-level resemblance with NewHope [Alk+16a] and Peikert's KEM [Pei14] in that its encapsulation procedure computes $(c_1, c_2) = (a \cdot r + e_1, b \cdot r + e_2)$. The main differences are that its public key is of the form $b = p \cdot (as + e)$ (instead of $b = as + e$ in [Pei14; Alk+16a]) and c_2 is sent along with c_1,

whereas [Pei14; Alk+16a] send a compressed version of c_2 from which a key is extractable using s (and a "reconciliation" technique from [DXL14; Pei14]). Also, decapsulation first recovers e_2 (instead of a noisy version of ars in [Pei14; Alk+16a]) in order to enforce rigidity and randomness recovery. In [Pei14; Alk+16a], the compression of c_2 makes it hard to combine both properties without using Fujisaki-Okamoto.

The NTRU cryptosystem [HPS98] can be made rigid, as observed in [Che+20, Section 2.3] where a re-encryption-free variant of an NTRU-KEM due to Saito *et al.* [SXY18] was proposed. While the latter construction admits highly efficient instantiations (as well as a tight proof in the QROM [SXY18]), these rely on a less standard assumption than RLWE. In a polynomial ring $R = \mathbb{Z}[X]/(\Phi(x))$, NTRU involves a public key of the form $h = p \cdot g/f \in R/(qR)$, for some moduli $p < q$ and small-norm polynomials $f, g \in R$. Its most efficient variants rely on the assumption (often called "NTRU assumption") that $h = p \cdot g/f$ is indistinguishable from a random invertible element of $R/(qR)$, even when f and g are sampled from a narrow distribution over R. In fact, some NTRU-based NIST candidates even sample f and g from a distribution of polynomials over $\mathbb{Z}[X]/(\Phi(X))$ with ternary coefficients in $\{-1, 0, 1\}$. Other works [LATV12] sample them from a wider, Gaussian distribution with standard deviation $\sigma < q^{1/2}$.

Despite recent progress [PMD21], the NTRU problem is still less understood than RLWE. It is in fact asymptotically easier for small-magnitude f, g: Kirchner and Fouque [KF15] gave a heuristic slightly sub-exponential algorithm in the setting where $q < \deg(\Phi)$ and $\|f\|_\infty, \|g\|_\infty = O(1)$. There is even a parameter regime [ABD16] where NTRU is easy and RLWE remains hard when q is larger than the secrets by a sub-exponential factor in $n = \deg(\Phi)$.

Stehlé and Steinfeld [SS11] showed that, when f and g are sampled from a discrete Gaussian with standard deviation $\sigma > \mathsf{poly}(n) \cdot q^{1/2}$, the NTRU problem is information-theoretically hard as the distribution of g/f is statistically uniform over the units of $R/(qR)$. Using this property, Stehlé and Steinfeld gave an IND-CPA-secure variant of NTRU that solely relies on the RLWE assumption. On the downside, sampling f and g from such a wide Gaussian requires a significantly larger modulus q to ensure correctness. We show in the ePrint version of this work that the re-encryption-free hybrid NTRU candidate of [Che+20, Section 2.3] can be made side-channel-resistant by applying the same countermeasures as in `POLKA`. Nevertheless, if we tune it so as to only rely on the RLWE assumption by applying the result of [SS11], it is significantly less efficient than `POLKA`. Asymptotically, it requires a modulus $q = \tilde{O}(n^7)$ while `POLKA` can make do with $q = O(n^3)$ when the noise is sampled

from a Gaussian of standard deviation $\alpha q = \Omega(\sqrt{n})$.

Recently, Duman *et al.* [Dum+] gave a rigorous analysis of the security of optimized NTRU-based KEMs. While some of their constructions are rigid, no discussion on side-channel resistance was given in [Dum+]. In fact, only their third construction seems compatible with our dummy-ciphertext technique. If we similarly set the parameters of POLKA according to the concrete hardness of RLWE against known attacks [APS15], our scheme competes fairly well with the most efficient NTRU candidates of [Dum+]. In a fully optimized instantiation, ciphertexts and keys fit within 3Kb. Moreover, in contrast with [Dum+], we can avoid the use of assumptions that posit the pseudorandomness of small-polynomial ratios.

D'Anvers *et al.* [DOV21] introduced a technique that obviates the need for re-encryption when Fujisaki-Okamoto is applied to specific KEMs based on Module LWE [LS14]. They somehow relaxed the rigidity property of [BP18] by showing that, as long as the SIP-LWE problem is hard, a cheaper test on error terms allows dispensing with the need to re-compute the entire ciphertext. Although their error-term-checking technique offers noticeable computational savings in some NIST candidates [Alk+16a; Ham19; Lu+19], it does not (and was not claimed to) protect against side-channel leakage, as observed in [Uen+22]. POLKA departs from their approach as it relies neither on the FO transform, nor on SIP-LWE.

The ideas we use regarding polynomial operations build on the early observation that they have good features for efficient masking: see [Rep+16b; Rep+15] for examples of key randomization (which we leverage for the parts of POLKA that require DPA security) and [Rep+16a] for an example of message randomization (which we leverage for the parts of POLKA that require SPA security), respectively. As also observed in these previous works, it is generally the additional operations besides the polynomial multiplication that make post-quantum schemes challenging to protect against leakage, and it is precisely this issue that POLKA aims to contribute to, by taking advantage of the concept of leveled implementations.

Eventually, the interest of an explicit rejection mechanism was also mentioned in the recent independent work of Hövelmanns et al. [HHM22].

3.3.2 Background

Lattices and Discrete Gaussian Distributions

An n-dimensional lattice $\Lambda \subseteq \mathbb{R}^n$ is the set $\Lambda = \{\sum_{i=1}^{n} z_i \cdot \mathbf{b}_i \mid \mathbf{z} \in \mathbb{Z}^n\}$ of all integer linear combinations of a set of linearly independent basis vectors $\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\} \subseteq \mathbb{R}^n$. Let $\mathbf{\Sigma} \in \mathbb{R}^{n \times n}$ be a symmetric positive definite matrix, and $\mathbf{c} \in \mathbb{R}^n$. The n-dimensional Gaussian function on $\mathbb{R}^n$ is defined as $\rho_{\mathbf{\Sigma},\mathbf{c}}(\mathbf{x}) = \exp(-\pi(\mathbf{x}-\mathbf{c})^\top \mathbf{\Sigma}^{-1}(\mathbf{x}-\mathbf{c}))$. In the special case where $\mathbf{\Sigma} = \sigma^2 \cdot \mathbf{I}_n$ and $\mathbf{c} = \mathbf{0}$, we denote it by ρ_σ. For any lattice $\Lambda \subset \mathbb{R}^n$, the discrete Gaussian distribution $D_{\Lambda,\mathbf{\Sigma},\mathbf{c}}$ has probability mass $\Pr_{X \sim D_{\Lambda,\mathbf{\Sigma},\mathbf{c}}}[X = \mathbf{x}] = \frac{\rho_{\mathbf{\Sigma},\mathbf{c}}(\mathbf{x})}{\rho_{\mathbf{\Sigma},\mathbf{c}}(\Lambda)}$ for any $\mathbf{x} \in \Lambda$. When $\mathbf{c} = \mathbf{0}$ and $\mathbf{\Sigma} = \sigma^2 \cdot \mathbf{I}_n$ we denote it by $D_{\Lambda,\sigma}$.

Lemma 1 ([MR07, Lemma 4.4])**.** *For $\sigma = \omega(\sqrt{\log n})$ there is a negligible function $\varepsilon = \varepsilon(n)$ such that $\Pr_{\mathbf{x} \sim D_{\mathbb{Z}^n,\sigma}}[\|\mathbf{x}\| > \sigma\sqrt{n}] \leq \frac{1+\varepsilon}{1-\varepsilon} \cdot 2^{-n}$.*

Rings and Ideal Lattices.

Let n a power of 2 and define the rings $R = \mathbb{Z}[X]/(X^n + 1)$ and $R_q = R/qR$. Each element of R is a $(n-1)$-degree polynomial in $\mathbb{Z}[X]$ and can be interpreted as an element of $\mathbb{Z}[X]$ via the natural coefficient embedding that maps the polynomial $a = \sum_{i=0}^{n-1} a_i X^i \in R$ to $(a_0, a_1, \ldots, a_{n-1}) \in \mathbb{Z}^n$. An element of R_q can similarly be viewed as a degree-$(n-1)$ polynomial over $\mathbb{Z}_q[X]$ and represented as an n-dimensional vector with coefficients in the range $\{-(q-1)/2, \ldots, (q-1)/2\}$.

The Euclidean and infinity norms of an element of $a \in R$ are defined by viewing elements of R as elements of $\mathbb{Z}^n$ via the coefficient embedding.

The ring R can also be identified as the subring of anti-circulant matrices in $\mathbb{Z}^{n \times n}$ by viewing each $a \in R$ as a linear transformation $r \to a \cdot r$. This implies that, for any $a, b \in R$, $\|a \cdot b\|_\infty \leq \|a\| \cdot \|b\|$ by the Cauchy-Schwartz inequality.

As in [KY16], for any lattice Λ, $D_{\Lambda,\sigma}^{\mathsf{coeff}}$ denotes the distribution of a ring element $a = \sum_{i=0}^{n-1} a_i X^i \in R$ of which the coefficient vector $(a_0, \ldots, a_{n-1})^\top \in \mathbb{Z}^n$ is sampled from the discrete Gaussian distribution $D_{\Lambda,\sigma}$.

We now recall the ring variant of the Learning-With-Errors assumption [Reg05a]. The ring LWE (RLWE) problem is to distinguish between a polynomial number of pairs of the form $(a_i, a_i \cdot s + e_i)$, where $a_i \sim U(R_q)$ and $s, e_i \in R$ are sampled from some distribution χ of

bounded-magnitude ring elements, and random pairs $(a_i, b_i) \sim U(R_q^2)$. In Definition 12, the number of samples k is made explicit.

Definition 12. *Let $\lambda \in \mathbb{N}$ a security parameter. Let positive integers $n = n(\lambda)$, $k = k(\lambda)$, and a prime $q = q(n) > 2$. Let an error distribution $\chi = \chi(n)$ over R. The $\mathsf{RLWE}_{n,k,q,\chi}$ assumption says that the following distance is a negligible function for any PPT algorithm $\mathcal{A}$,*

$$\mathbf{Adv}^{\mathcal{A},\mathsf{RLWE}}_{n,k,q,\chi}(\lambda) := \big|\Pr[\mathcal{A}(1^\lambda, \{(a_i, v_i)\}_{i=1}^k) = 1] - \Pr[\mathcal{A}(1^\lambda, \{(a_i, a_i s + e_i)\}_{i=1}^k) = 1]\big|,$$

where $a_1, \ldots, a_i, v_1, \ldots, v_k \hookleftarrow U(R_q)$, $s \hookleftarrow \chi$, $e_1, \ldots, e_i \hookleftarrow \chi$.

For suitable parameters, the RLWE assumption is implied by the hardness of worst-case instances of the approximate shortest vector problem in ideal lattices.

Lemma 2 ([LPR10])**.** *Let n a power of 2. Let $\Phi_m(X) = X^n + 1$ the m-th cyclotomic polynomial where $m = 2n$, and $R = \mathbb{Z}[X]/(\Phi_m(X))$. Let $q = 1 \bmod 2n$. Let also $r = \omega(\sqrt{\log n})$ Then, there is a randomized reduction from $2^{\omega(\log n)} \cdot (q/r)$-approximate R-SVP to $\mathsf{RLWE}_{n,\mathsf{poly}(n),q,\chi}$ where $\chi = D^{\mathsf{coeff}}_{\mathbb{Z}^n,r}$.*

3.3.3 POLKA: Rationale and Specifications

Our starting point is a variant of the LPR cryptosystem [LPR10], which builds on a rigid randomness-recovering KEM. As in [LPR10], the public key contains a random ring element $a \in R_q$ and a pseudorandom $b \in R_q$. Here, b is of the form $b = p \cdot (a \cdot s + e)$ (instead of $b = a \cdot s + e$ as in [LPR10]), for secret $s, e \in R$ sampled from the noise distribution and where p is an integer such that $\|e\|_\infty < p$. Another difference with [LPR10] is that decryption requires b to be invertible over R_q.

The encryptor samples ring elements $r, e_1, e_2 \in R$ from a Gaussian distribution and uses them to derive a symmetric key $K = H(r, e_1, e_2)$. The latter is then encapsulated by computing a pair $(c_1, c_2) = (a \cdot r + e_1, b \cdot r + e_2)$. The decryption algorithm uses $s \in R$ to compute $\mu = c_2 - p \cdot c_1 \cdot s \in R_q$, which is a small-norm ring element $\mu = e_2 + p \cdot (er - e_1 s) \in R$. This allows recovering $e_2 = \mu \bmod p$, which in turn reveals $r = (c_2 - e_2) \cdot b^{-1} \in R_q$ and $e_1 = c_1 - a \cdot r \in R_q$. After having checked the smallness of (r, e_1, e_2), the decryption procedure obtains $K = H(r, e_1, e_2)$. The scheme provides the *rigidity* property of [BP18] as the decryptor obtains $(r, e_1, e_2) \in R^3$ such that $\|r\|, \|e_1\|, \|e_2\| \leq B$, for some norm bound B, if and only if $(c_1, c_2) = (a \cdot r + e_1, b \cdot r + e_2)$.

This ensures that no re-encryption is necessary to check the validity of the input pair (c_1, c_2).

In this section, our hybrid encryption scheme builds on a KEM with *explicit rejection* (in the terminology of [Per12; HHK17a]), meaning that invalid encapsulations are rejected as soon as they are noticed in the decryption algorithm. In the security proof, we will switch to an implicit rejection mechanism (as defined [Per12, Section 5.3]), where the decapsulation algorithm outputs a random key on input of an invalid encapsulation. The rejection of malformed encapsulations is then deferred to the symmetric decryption step.

The Scheme With an Additive Mask

We now describe a version of the scheme that has good features for side-channel resistant implementation, where the decryption algorithm first adds a "dummy ciphertext" to (c_1, c_2) before proceeding with the actual decryption.

Keygen(1^λ)**:** Given a security parameter $\lambda \in \mathbb{N}$,

1. Choose a dimension $n \in \mathbb{N}$, a prime modulus $q = 1 \bmod 2n$. Let the rings $R = \mathbb{Z}[X]/(X^n+1)$ and $R_q = R/(qR)$ such that $\Phi(X) = X^n + 1$ splits into linear factors over R_q. Let $R_q^\times$ the set of units in R_q.
2. Choose a noise parameter $\alpha \in (0, 1)$, and let a norm bound $B = \alpha q\sqrt{n}$. Choose an integer $p \in \mathbb{N}$ such that $4B < p < \frac{q}{8(B^2+1)}$.
3. Sample $a \hookleftarrow U(R_q)$ and $s, e \hookleftarrow D^{\mathsf{coeff}}_{\mathbb{Z}^n,\alpha q}$ and compute $b = p \cdot (a \cdot s + e)$. If $b \notin R_q^\times$, restart step 3.
4. Choose an authenticated symmetric encryption scheme $\Pi^{sym} = (\mathsf{K}, \mathsf{E}, \mathsf{D})$ with key length $\kappa \in \mathsf{poly}(\lambda)$ and message space $\{0,1\}^{\ell_m}$.
5. Let a domain $D_E := \{(r, e_1, e_2) \in R^3 : \|r\|, \|e_1\|, \|e_2\| \leq B\}$. Choose a hash function $H : D_E \to \{0,1\}^\kappa$ modeled as a random oracle.

Return the key pair (PK, SK) where

$$PK := \Big(n,\ q,\ p,\ \alpha,\ a \in R_q,\ b \in R_q^\times,\ \Pi^{sym},\ H,\ B\Big) \text{ and } SK := s \in R.$$

Encrypt(PK, M)**:** Given a public key PK and a message $M \in \{0,1\}^{\ell_m}$:

1. Sample $r, e_1, e_2 \hookleftarrow D^{\mathsf{coeff}}_{\mathbb{Z}^n,\alpha q}$ and compute
$$c_1 = a \cdot r + e_1 \in R_q, \qquad c_2 = b \cdot r + e_2 \in R_q$$
together with $K = H(r, e_1, e_2) \in \{0,1\}^\kappa$.
2. Compute $c_0 = \mathsf{E}_K(M)$.

Output the ciphertext $C = (c_0, c_1, c_2)$.

Decrypt(SK, C)**:** Given $SK = s \in R$ and $C = (c_0, c_1, c_2)$, do the following:

1. Sample $r', e'_1, e'_2 \hookleftarrow D^{\mathsf{coeff}}_{\mathbb{Z}^n,\alpha q}$ and return $\perp$ if $\|r'\| > B$, or $\|e'_1\| > B$, or $\|e'_2\| > B$. Otherwise, compute $c'_1 = a \cdot r' + e'_1$ and $c'_2 = b \cdot r' + e'_2$.
2. Compute $\bar{c}_1 = c_1 + c'_1$ and $\bar{c}_2 = c_2 + c'_2$.
3. Compute $\bar{\mu} = \bar{c}_2 - p \cdot \bar{c}_1 \cdot s$ over R_q.
4. Compute $\bar{e}_2 = \bar{\mu} \bmod p$. If $\|\bar{e}_2\| > 2B$, return $\perp$.
5. Compute $\bar{r} = (\bar{c}_2 - \bar{e}_2) \cdot b^{-1} \in R_q$. If $\|\bar{r}\| > 2B$, return $\perp$.
6. Compute $\bar{e}_1 = \bar{c}_1 - a \cdot \bar{r} \in R_q$. If $\|\bar{e}_1\| > 2B$, return $\perp$.
7. Compute $r = \bar{r} - r'$, $e_1 = \bar{e}_1 - e'_1$ and $e_2 = \bar{e}_2 - e'_2$. If $\|r\| > B$, or $\|e_1\| > B$, or $\|e_2\| > B$, then return $\perp$.
8. Compute $K = H(r, e_1, e_2) \in \{0,1\}^\kappa$ and return
$$M = \mathsf{D}_K(c_0) \in \{0,1\}^{\ell_m} \cup \{\perp\}.$$

The use of fully splitting rings may require multiple attempts to find an invertible $b \in R_q^\times$ as step 3 of Keygen. Under the RLWE assumption, a suitable b can be found after at most $\lceil \lambda / \log n \rceil$ iterations, except with negligible probability $2^{-\lambda}$ (this property is proven in the process of the security proof of the scheme, that can be found in the ePrint version). In practice, a small number of attempts suffices since a random ring element is invertible with probability $1 - n/q$, which is larger than $1 - 1/n$ with our choice of parameters.

Correctness. Let $\bar{r} = r + r'$, $\bar{e}_1 = e_1 + e'_1$ and $\bar{e}_2 = e_2 + e'_2$ over R. At step 2, Decrypt computes $\bar{c}_1 = a \cdot \bar{r} + \bar{e}_1$, $\bar{c}_2 = b \cdot \bar{r} + \bar{e}_2$ over R_q, where

$\|\bar{r}\|, \|\bar{e}_1\|, \|\bar{e}_2\| \leq 2B$ with probability $1 - 2^{-\Omega(n)}$ over the randomness of Encrypt and Decrypt (by Lemma 1). At step 3, the decryptor obtains

$$\begin{aligned}\bar{\mu} &= \bar{c}_2 - p \cdot \bar{c}_1 \cdot s \bmod q \\ &= (b \cdot \bar{r} + \bar{e}_2) - p \cdot (a \cdot \bar{r} + \bar{e}_1) \cdot s \bmod q \\ &= p \cdot (as + e) \cdot \bar{r} + \bar{e}_2 - p \cdot a\bar{r}s - p \cdot \bar{e}_1 s \bmod q \\ &= \bar{e}_2 + p \cdot e\bar{r} - p \cdot \bar{e}_1 s,\end{aligned}$$

where the last equality holds over R with overwhelming probability over the randomness of Keygen, Encrypt and Decrypt. Indeed, Lemma 1 implies that $\|s\|, \|e\| \leq \alpha q\sqrt{n}$ with probability $1 - 2^{-\Omega(n)}$ over the randomness of Keygen. With probability $1 - 2^{-\Omega(n)}$ over the randomness of Encrypt and Decrypt, we also have $\|\bar{r}\|, \|\bar{e}_1\|, \|\bar{e}_2\| \leq 2\alpha q\sqrt{n}$. Then, the Cauchy-Schwartz inequality implies

$$\begin{aligned}\|\bar{e}_2 + p \cdot e\bar{r} - p \cdot \bar{e}_1 s\|_\infty &< 2\alpha q\sqrt{n} + 4p \cdot (\alpha q)^2 n \\ &< 4p(B^2 + 1) < q/2.\end{aligned} \tag{3.3}$$

Since $p/2 > 2\alpha q\sqrt{n}$, step 4 recovers $\bar{e}_2$ with overwhelming probability. Since $b \in R_q^\times$, Decrypt obtains $\bar{r}$ at step 5 and $\bar{e}_1$ at step 6. Therefore, it also recovers (r, e_1, e_2) at step 7 and the correct symmetric key $K = H(r, e_1, e_2)$ at step 8. Correctness thus follows from the correctness of Π^{sym}.

Remark 1. We note that correctness is guaranteed whenever $\|s\|, \|e\| \leq B$ and $\|\bar{r}\|, \|\bar{e}_1\|, \|\bar{e}_2\| \leq 2B$, as it is a sufficient condition to have inequalities (3.3). This will be used in the security proof.

On Decryption Failures. Due to the rigidity and randomness recovery properties of the scheme, the probability of decryption failure does not depend on the specific secret key s in use as long as $\|s\| \leq B$. If $(r, e_1, e_2) \in D_E$ and $\|s\| \leq B$, we always have $\|\bar{\mu}\|_\infty \leq q/2$, where $\bar{\mu} = \bar{c}_2 - p \cdot \bar{c}_1 \cdot s \bmod q = p \cdot (e \cdot \bar{r} - \bar{e}_1 \cdot s) + \bar{e}_2$ unless the ciphertext is rejected at step 1 (which does not depend on s). If $(r, e_1, e_2) \notin D_E$, then either: (i) We still have $\|\bar{\mu}\|_\infty \leq q/2$ and Decrypt obtains (r, e_1, e_2), which are necessarily rejected; or (ii) $\|\bar{\mu}\|_\infty > q/2$ but the extracted $(r^\dagger, e_1^\dagger, e_2^\dagger)$ cannot land in D_E since, otherwise, the rigidity property would imply $(c_1, c_2) = (a \cdot r^\dagger + e_1^\dagger, b \cdot r^\dagger + e_2^\dagger)$, in which case we would have $\|\bar{\mu}\|_\infty \leq q/2$ unless the ciphertext is rejected at step 1. Hence, if Decrypt does not return $\perp$ at step 1, it computes $(r, e_1, e_2) \in D_E$ if and only if $(c_1, c_2) = (a \cdot r + e_1, b \cdot r + e_2)$ no matter which s of norm $\|s\| \leq B$ is used at step 2.

In contrast, when $m = 0$ is encrypted in the LPR cryptosystem, we

have $\mu = c_2 - c_1 \cdot s \bmod q = e \cdot r - e_1 \cdot s + e_2$. An adversary can then fix small (r, e_2) and play with many e_1's until it triggers a decryption failure when $\|\mu\|_\infty > q/2$. The probability that this happens depends on the secret s (as a different s' may not cause rejection for a fixed (r, e_1, e_2)). In KYBER and SABER, the FO transform allows restricting the adversary's control over e_1 (which is derived from a random message m using a random oracle and re-computed for verification upon decryption) so as to make such attacks impractical. The FO transform is thus crucial to offer a sufficient security margin against attacks like [D'A+19; DRV20].

Black-box security analysis. A detailed security analysis can be found in the ePrint version of this work. It is interesting to highlight that the design choices made regarding side-channel security are fully part of the scheme and must be taken into account in the security proofs.

Parameters and Instantiations

Parameters. In an instantiation in fully splitting rings R_q (which allows faster multiplications using the NTT in Encrypt), we use $\Phi(X) = X^n + 1$, where n is a power of 2, with a modulus $q = 1 \bmod 2n$.

For correctness, we need to choose $\alpha \in (0, 1)$, q and p such that $p/2 > 2B$ and $4p(B^2 + 1) < q/2$. To apply Lemma 2, we can set $\alpha \in (0, 1)$ so that $\alpha q = \Omega(\sqrt{n})$. To satisfy all conditions, we may thus set $p = \Theta(n)$, $q = \Theta(n^3)$ and $\alpha^{-1} = \Theta(n^{2.5})$.

Concrete proposals. Around the 128-bit security level, n has to be somewhere between 512 and 1024. When r, e_1, e_2 are sampled from $D^{\mathsf{coeff}}_{\mathbb{Z}^n, \alpha q}$, we relax the constraint $\alpha q = \Omega(\sqrt{n})$ but still choose $\alpha q \approx \lambda^{1/2}$ in order to apply Lemma 1 and taking into account the constraints from the security proof. If we set $n = 1024$ so as to have a power of 2, we can use $q = 1813307393$, $\alpha q = 12$, and $p = 1537$, so that the pair (c_1, c_2) fits within 7.75Kb.

In order to obtain a more optimized instantiation, we can further sample s, e, r, e_1, e_2 from a centered binomial distributions as suggested in [Alk+16a; Bos+18a; Dum+] and set the parameters according to the concrete hardness against known attacks via the LWE estimator [APS15]. As in [Dum+], we use the distribution $\bar{\psi}_2^n$ obtained by reducing ψ_2^n mod 3, where ψ_k^n is defined over $\mathbb{Z}^n$ as the distribution $\{\sum_{i=1}^k (\boldsymbol{a}_i - \boldsymbol{b}_i) \mid \boldsymbol{a}_i, \boldsymbol{b}_i \hookleftarrow U(\{0,1\}^n)\}$. This modification only entails minor modifications in the security proof, which are detailed in the ePrint version of this work. We then obtain ciphertexts of 4Kb by setting $n = 1024$, $p = 5$ and $q = 59393$.

In order to push optimizations even further, we could use rings where n does not have to be a power of 2, as suggested in [Dum+]. For example,

the cyclotomic polynomial $\Phi_{3n} = X^n - X^{n/2} + 1$ with $q = 1 \bmod 3n$ and $n = 2^i 3^j$ still gives a fully splitting $R_q = \mathbb{Z}_q[X]/(\Phi_{3n})$. This allows choosing $n = 768$, $p = 5$ (so that $\|\bar{e}_2\|_\infty \leq (p-1)/2$), and $q = 28 \cdot 3n + 1 = 64513$. Correctness is ensured since we have $\|a \cdot b\|_\infty \leq 2\|a\|\|b\|$ for any $a, b \in R = \mathbb{Z}[X]/(\Phi_{3n})$, so that $\|\bar{e}_2 + p \cdot (e \cdot \bar{r} - \bar{e}_1 \cdot s)\|_\infty \leq (p-1)/2 + 8 \cdot p \cdot n \leq (q-1)/2$. We would then obtain a ciphertext (c_1, c_2) that only takes 3Kb to represent.

In all instantiations, the public key can compressed down to roughly 50% of the ciphertext size if we derive the random $a \in R_q$ from a hash function modeled as a random oracle (as considered by many NIST candidates).

Instantiating the DEM component. The symmetric authenticated encryption scheme Π^{sym} can in general be instantiated with a a leakage-resistant Enc-then-MAC mode of operation. Candidates for this purpose that rely on a masked block cipher or permutation can be found in [Bel+20]. Yet, `POLKA` encourages the following more efficient solution based on a key-homomorphic one-time MAC. Specifically, if ℓ_m is the message length, we can use a key length $\kappa = \lambda + 2\ell_m$ and a pseudorandom generator $G : \{0,1\}^\lambda \to \{0,1\}^{\ell_m}$. To encrypt $M \in \{0,1\}^{\ell_m}$, we parse $K = H(r, e_1, e_2) \in \{0,1\}^\kappa$ as a triple $K = (K_0, K_1, K_2) \in \{0,1\}^\lambda \times (\{0,1\}^{\ell_m})^2$. Then, we compute a ciphertext $c = (\bar{c}, \tau) = (M \oplus G(K_0), K_1 \cdot \bar{c} + K_2)$, where the one-time MAC $\tau = K_1 \cdot \bar{c} + K_2$ is computed over $\mathrm{GF}(2^{\ell_m})$. We can choose $\ell_m \geq 308$.

In terms of leakage, the computation $K_1 \cdot \bar{c} + K_2$ is linear in the key and can therefore be masked with overheads that are linear in the number of shares (rather than quadratic for a block cipher or permutation). The constraint on the message length could be relaxed by hashing the message at the cost of an additional idealized assumption, which we leave as a scope for further research.

3.3.4 Side-Channel Security Analysis

We now discuss the leakage properties of `POLKA`. In Section 3.3.4 we introduce the general ideas supporting its leveled implementation and explain how its security requirements can be efficiently fulfilled. In Section 3.3.4, we focus on its most novel part, namely the variant of the LWPR assumption on which this implementation relies. We also provide cryptanalysis challenges to motivate further research on hard physical learning problems. Our descriptions borrow the terminology introduced in [Guo+19] for symmetric cryptography. Namely, we denote as leakage-resilient implementations of which confidentiality guarantees may vanish in the presence of leakage, but are restored once leakage is removed from

the adversary's view; and we denote as leakage-resistant implementations that preserve confidentiality against leakage even for the challenge encryption.

Leveled Implementation and Design Goals

The high-level idea behind leveled implementations is that it may not be necessary to protect all the parts of implementation with equally strong (and therefore expensive) side-channel countermeasures. In the following, we describe how `POLKA` could be implemented in such a leveled manner. For this purpose, and as a first step, we follow the heuristic methodology introduced in [Bel+20] and identify its SPA and DPA targets in decryption. The resulting leveled implementation of `POLKA` is represented in Figure 3.11. The lighter green-colored (dummied) operations need to be protected against SPA. The darker green-colored operations need to be protected against SPA with averaging (avg-SPA), which is a SPA where the adversary can repeat the measurement of a fixed target intermediate computation in order to remove the leakage noise. The lighter blue-colored operations need to be protected against DPA with unknown (dummied) inputs (UP-DPA). The darker blue-colored operations must be protected against DPA. Operations become generically more difficult to protect against side-channel attacks when moving from the left to the right of the figure. Eventually, securing the first four steps in the figure is needed to protect the long-term secret of `POLKA`, and therefore to ensure leakage-resilience. By contrast, securing the fifth step is only needed to ensure leakage-resistance (these values can leak in full in case only leakage-resilience is required). Next, we first explain how these security requirements can be efficiently satisfied by hardware designers. We then discuss the advantages of this implementation over a uniformly protected one.

SPA and DPA protections. All the operations requiring SPA protection (with or without averaging) can be efficiently implemented thanks to parallelism in hardware. Typically, we expect that an implementation manipulating 128 bits or more in parallel is currently difficult to attack via SPA, even when leveraging advanced analytical strategies [VGS14].[21] A bit more concretely, this reference shows that single-trace attacks are possible for Signal-to-Noise Ratios (SNRs) higher than one. Adversaries targeting a 128-bit secret based on 8-bit (resp., 32-bit) hypotheses would face an SNR of $\frac{1}{16}$ (resp., $\frac{1}{4}$). Securing the computations in steps 1, 3

[21] As will be clear in conclusions, software implementations are left as an interesting open problem. In this case, the typical option to obtain security against SPA would be to emulate parallelism thanks to the shuffling countermeasure [Vey+12].

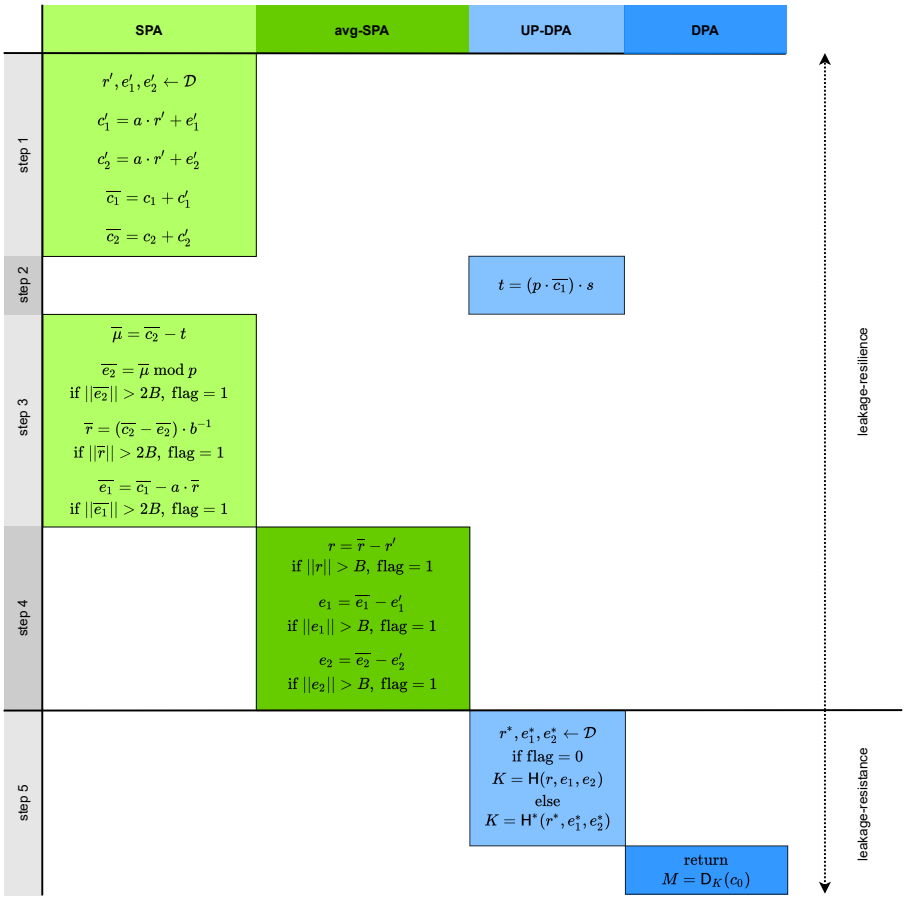


Figure 3.11: Leveled implementation of POLKA.

and 4 of Figure 3.11 against side-channel attacks should therefore lead to limited overheads. Note that the dependency on a dummy ciphertext in step 3 prevents the adversary to control the intermediate computations (and for example to try canceling the algorithmic noise for those sensitive operations).

Security against DPA is in general expected to be significantly more expensive to reach. The standard approach for this purpose is to mask all the operations that can be targeted, which leads to (roughly) quadratic performance overheads [ISW03]. Furthermore, implementing masking securely is a sensitive process, which requires dealing with composition issues [Cor+13; Bar+16], physical defaults such as glitches [MPG05;

NRS11] or transitions [Cor+12; Bal+14] or even their combination [Cas+21; CS21a].

The main observation we leverage in POLKA is that its most critical DPA sensitive operation shares similarities with the key-homomorphic re-keying schemes used in symmetric cryptography to prevent side-channel attacks [Med+10; Dob+15; Dzi+16; Duv+21]. Namely, the operation $t = (p \cdot \overline{c_1}) \cdot s$ in step 2 of Figure 3.11 can indeed be viewed as the product between a long-term secret s and an ephemeral (secret) value $(p \cdot \overline{c_1})$. As a result, it can be directly computed as $t = \sum_{i=1}^{d} (p \cdot \overline{c_1}) \cdot s^i$, where $s = s^1 + s^2 + \ldots s^d$ and the s^i's are the additive shares of the long-term secret s. Besides the linear (rather than quadratic) overheads that such a solution enables, key-homomorphic primitives have two important advantages for masking. First, their long-term secret can be refreshed with linear randomness requirements [Bar+17]. Second, their natural implementation offers strong immunity against composition issues and physical defaults [BSS21]. On top of this, the fact that the variable input of $(p \cdot \overline{c_1}) \cdot s^i$ is dummied (hence unknown) implies that it will need one less share than in a known input setting [Ber+22].

Even more importantly, and as discussed in the aforementioned papers on fresh re-keying, it is then possible to re-combine the shares and to perform the rest of the computations on unshared values, hence extending the interest of a leveled approach. Various models have been introduced for this purpose in the literature, depending on the type of multiplication to perform. The first re-keying schemes considered multiplications in binary fields that require a sufficient level of noise to be secure [BFG14; Bel+15]. Dziembowski et al. proposed a (more expensive) wPRF-based re-keying that is secure even if its output is leaked in full [Dzi+16]. Duval et al. proposed an intermediate solution that only requires the (possibly noise-free) leakage function to be surjective and "incompatible" with the field multiplication: they for example show that this happens when combining multiplications in prime fields with the Hamming weight leakage function, which they formalized as the LWPR assumption [Duv+21]. Given that the multiplication of POLKA is based on prime moduli, we next focus on this last model, which provides a nice intermediate between efficiency and weak physical assumptions.

As for the operations of step 5 of Figure 3.11, we first observe that despite the inputs of H being ephemeral, it is possible that an adversary obtains a certain level of control over them by incrementally increasing c_1 or c_2. This explains why it must be secure against DPA (with unknown plaintexts since r, e_1 and e_2 are unknown as long as steps 3 and 4 are secure against SPA). Finally, the protection of the authenticated encryption is somewhat orthogonal to POLKA since it is needed for any

DEM. The standard option for this purpose would be to use a leakage-resistant mode of operation that ensures side-channel security with decryption leakage. As discussed in [Bel+20], state-of-the-art modes allow the authenticated encryption scheme to be leveled (i.e., to mix SPA-secure operations with DPA-secure ones), like the rest of POLKA. But as mentioned in Section 3.3.3, an even more efficient solution is to use an Enc-then-MAC scheme with a one-time key-homomorphic MAC that is linear in the key and therefore easy to mask.

Discussion. The main advantage of POLKA is that its structure allows avoiding the costly implementation of uniformly protected operations based on masking. In this respect, it is worth recalling that: (i) the removal of the dummy ciphertext takes place as late as possible in the process (i.e., just before the hashing and symmetric decryption), and (ii) if only the long-term secret s must be protected (i.e., if only leakage-resilience is required), step 5 of Figure 3.11 does not need countermeasures. Overall, these design tweaks strongly limit the side-channel attack surface and the need to mask non-linear operations compared to algorithms like KYBER or SABER, at the cost of an admittedly provocative LWPR assumption.

We note that the explicit rejection process allows avoiding the need to manipulate additional long-term secret material. This is in contrast with an implicit rejection mechanism where a pseudorandom "garbage key" is generated in case of invalid ciphertext, the generation of which must be protected against DPA. Yet, ensuring the leakage-resistance of POLKA against timing attacks still requires running a dummy hash function when flag $= 1$ in order to ensure constant time. Otherwise, the same granular increase of c_1 or c_2 as mentioned to justify the DPA security requirements of H could leak information on e_1, e_2 and r if a timing channel allowed detecting whether a $\perp$ message is generated during step 4 or step 5 (by the authenticated encryption scheme). This also means that it should be hard to distinguish whether the flag is 0 or 1 with SPA to ensure leakage-resistance. We conjecture the latter is simpler/cheaper than protecting another long-term secret against DPA (as required with implicit rejection), but as mentioned in introduction, POLKA could be adapted with an implicit rejection mechanism as well (in which case, it should also be hard to distinguish whether the key used to decrypt is a garbage one or not thanks to SPA).

Learning With Physical Rounding Assumption

We now move to the main assumption that allows POLKA to be masked with linear overheads. Namely, we study the security of step 2 in Fig-

ure 3.11 after the recombinations of the shares. In other words, we study the security of the long-term secret s assuming that the adversary can observe the leakage of the (unmasked) output t.[22] We start by recalling the LWPR problem introduced at CHES 2021 [Duv+21], then discuss its adaptation to polynomial multiplications needed for POLKA. We finally propose parameters & cryptanalysis challenges.

A. The original LWPR problem.

This problem can be viewed as an adaptation of the crypto dark matter proposed by Boneh et al. in [Bon+18], which showed that low-complexity PRFs can be obtained by mixing linear functions over different small moduli. Duval et al. observed that letting one of these functions being implicitly computed by a leakage function can lead to strong benefits for masking against side-channel attacks. Intuitively, it implies that a designer only has to implement a key-homomorphic function securely (i.e., the first crypto dark matter mapping), since the second (physical) mapping never has to be explicitly computed: it is rather the leakage function that provides its output to the adversary. The formal definition of the resulting LWPR problem is given next.

Definition 13 (Learning with physical rounding [Duv+21])**.** *Let $q, x, y \in \mathbb{N}^*$, q prime, for a secret $\kappa \in \mathbb{F}_q^{x \times y}$. The $\mathit{LWPR}^{x,y}_{\mathsf{L_g},q}$ sample distribution is given by:*

$$\mathcal{D}_{\mathit{LWPR}^{x,y}_{\mathsf{L_g},q}} \coloneqq (\boldsymbol{r}, \mathsf{L_g}(\kappa \cdot \boldsymbol{r})) \text{ for } \boldsymbol{r} \in \mathbb{F}_q^y \text{ uniformly random,}$$

where $\mathsf{L_g} : \mathbb{F}_q^x \to \mathbb{R}^d$ is the physical rounding function. Given query access to $\mathcal{D}_{\mathit{LWPR}^{x,y}_{\mathsf{L_g},q}}$ for a uniformly random κ, the $\mathit{LWPR}^{x,y}_{\mathsf{L_g},q}$ problem is $(\chi, \tau, \mu, \epsilon)$-hard to solve if after the observation of χ LWPR samples, no adversary can recover the key κ with time complexity τ , memory complexity μ and probability $\geq \epsilon$.

Concretely, the LWPR problem consists in trying to retrieve a secret key matrix κ using the information leakage emitted during its product with a random vector r. It corresponds to a learning problem similar to LWR [BPR12], with the rounding function instantiated with a leakage function. Its security depends on the dimensions (q, x, y) and the leakage function considered. In [Duv+21], it is argued that this problem is hard in the case of the Hamming weight leakage function that is frequently encountered in practice (with a regular binary representation). This

[22] As mentioned in Section 3.3.4, the security of the internal computations of $t = \sum_{i=1}^{d} (p \cdot \overline{c_1}) \cdot s^i$ is obtained thanks to masking. So here, we only need to argue that the leakage of the recombined t does not lead to strong attacks.

problem can then be used as the basis of a fresh re-keying mechanism, producing an ephemeral key $\in \mathbb{F}_q^x$ thanks to the aforementioned product. It can be implemented serially (by setting the x parameter to 1), in which case words of $\log_2(q)$ bits of ephemeral key will be produced one by one, or in a parallel manner by increasing x, therefore producing $x \times \log_2(q)$ bits of ephemeral key at once.

The security analysis of Duval et al. first shows that the complexity of various (algebraic and statistical) attacks against such a fresh re-keying scheme grows exponentially with the (main) security parameter y. Due to the inevitably heuristic nature of their cryptanalysis, they next use the parallelism parameter x as a way to obtain security margins. Denoting the words of the ephemeral key as z_i, a serial implementation will leak independent $\mathsf{HW}(\mathsf{g}(z_i))$ values to the adversary, while a parallel one will leak $\sum_{i=1}^{x} \mathsf{HW}(\mathsf{g}(z_i))$ values, with $\mathsf{g}(z_i)$ denoting the binary representation of z_i. As a result, the amount of mutual information per word that such Hamming weight leakages provide rapidly decreases as $\frac{\log_2(x \log_2(q))}{x \log_2(q)}$, which is expected to make attacks more challenging.

The instance proposed by Duval et al. uses a 31-bit prime modulus $p = 2^{31} - 1$ with parameters $x = 4$ and $y = 4$ (i.e., it assumes that four $\log_2(p)$-bit multiplications can be performed in parallel). It claims 128 bits of security.

As can be seen in Figure 3.11, step 2 of `POLKA` shares strong similarities with the aforementioned fresh-re-keying scheme based on LWPR, by simply viewing the intermediate value t as an ephemeral key. We next discuss the differences between the original LWPR assumption and the one needed for `POLKA`.

B. Ring-LWPR. Leveraging the fact that ring variants of learning problems are common [LPR10], we now describe a ring version of the LWPR problem. Let $r \coloneqq p \cdot \overline{c_1}$. Seeing s as a long-term secret (similar to κ in the original LWPR problem), the t value can be re-written as $t = r \cdot s$. Further denoting s_i (resp., r_i) the coefficients of $\boldsymbol{s}$ (resp., $\boldsymbol{r}$), we can write:

$$r \cdot s = \left(\sum_{i=0}^{n-1} s_i X^i \right) \cdot \left(\sum_{i=0}^{n-1} r_i X^i \right) = \sum_{i=0}^{2n-2} \left(\sum_{j=max(0,i-n+1)}^{min(i,n-1)} s_j r_{i-j} \right) X^i,$$

$$= \sum_{i=0}^{n-2} \left(\sum_{j=0}^{i} s_j r_{i-j} \right) X^i + \left(\sum_{j=0}^{n-1} s_j r_{i-j} \right) X^{n-1} + \sum_{i=n}^{2n-2} \left(\sum_{j=i-n+1}^{n-1} s_j r_{i-j} \right) X^i,$$

$$= \sum_{i=0}^{n-2} \left(\sum_{j=0}^{i} s_j r_{i-j} \right) X^i + \left(\sum_{j=0}^{n-1} s_j r_{i-j} \right) X^{n-1} + \sum_{i=0}^{n-2} \left(\sum_{j=i+1}^{n-1} s_j r_{n+i-j} \right) X^{n+i},$$

$$= \sum_{i=0}^{n-2} \left(\sum_{j=i+1}^{n-1} s_j r_{i-j} - \sum_{j=0}^{i} s_j r_{n+i-j} \right) X^i + \left(\sum_{j=0}^{n-1} s_j r_{i-j} \right) X^{n-1}.$$

The above equation highlights the matrix representation of the polynomial multiplication carried out in `POLKA`. If we represent polynomials as n-dimension vectors, where the i-th coefficient is the polynomial's i-th coefficient, the product $\boldsymbol{r} \cdot \boldsymbol{s}$ can be represented as the following matrix-vector product:

$$\begin{pmatrix} r_0 & -r_{n-1} & \dots & -r_2 & -r_1 \\ r_1 & r_0 & \ddots & & -r_2 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ r_{n-2} & & \ddots & \ddots & -r_{n-1} \\ r_{n-1} & r_{n-2} & \dots & r_1 & r_0 \end{pmatrix} \cdot \begin{pmatrix} s_0 \\ s_1 \\ \vdots \\ s_{n-2} \\ s_{n-1} \end{pmatrix},$$

The key is represented as the vector (rather than the matrix) in order to optimize memory usage when splitting it into shares. This product can therefore be seen as a large LWPR instance, with two significant differences. First, a circulant matrix is used instead of one having independent coefficients (which we will discuss when selecting parameters in the next subsection). Second, the size of the matrix is (much) larger than the one in the original LWPR. Concretely, this second difference implies that in practice, these products are unlikely to be performed in one step: they will rather be decomposed into several submatrix-subvector products. For this purpose, let $x \in \mathbb{N}$ be a divider of n, the $\boldsymbol{s}$ matrix can then be split in $\frac{n}{x}$ $(x \times n)$-submatrices, denoted $(B_u)_{0 \leq u < \frac{n}{x}}$. The product can then be decomposed into $\frac{n}{x}$ subproducts between $\boldsymbol{r}$ and the x-line submatrices of $\boldsymbol{s}$.

with x serving as a parameter to adapt the security vs. performance tradeoff, as in the original LWPR. For a given k, one can explicitly obtain the coefficient i, j of B_u. For a proposition $\mathcal{P}$, denote $\mathbb{1}_{\mathcal{P}} := \begin{cases} 1 \text{ if } \mathcal{P} \\ 0 \text{ else} \end{cases}$. Then, $B_u^{i,j} = (2\mathbb{1}_{(ux+i-j)<0} - 1) \cdot r_{ux+i-j \,(\mathrm{mod}\ n)}$ and the $(x \times n)$-submatrices are therefore Toeplitz, determined by their first line and first column, each other value being equal to their top-left neighbor. Concretely, the x parameter sets the number of coefficients that

are computed in parallel, so that an adversary will be granted access to $\frac{n}{x}$ samples given by the leakage function applied to $x \log_2(q)$ bit-values.

Definition 14 (Ring learning with Physical rounding)**.** *Let $q, x, n \in \mathbb{N}$, q prime, for a secret $s \in R_q$. The $\mathsf{RLWPR}^{n,x}_{\mathsf{L_g},q}(\boldsymbol{s})$ sample distribution is given as:*

$$\mathcal{D}_{\mathsf{RLWPR}^{n,x}_{\mathsf{L_g},q}(\boldsymbol{s})} \coloneqq \left(r, \left(\mathsf{L_g}\left(B_u \cdot s\right)\right)_{0 \leq u < \frac{n}{x}}\right),$$

where $\mathsf{L_g}$ is the physical rounding function and the (B_u) are submatrices made of elements of $\boldsymbol{r}$ as defined above. Given query access to $\mathcal{D}_{\mathsf{RLWPR}^{n,x}_{\mathsf{L_g},q}(\boldsymbol{s})}$ for a uniformly random $\boldsymbol{s}$, the $\mathsf{RLWPR}^{n,x}_{\mathsf{L_g},q}(\boldsymbol{s})$ problem is $(\chi, \tau, \mu, \epsilon)$-hard to solve if after the observation of χ RLWPR samples, no adversary can recover the key s with time complexity τ, memory complexity μ and probability higher than ϵ.

Note that an implementer can also split each $(x \times n)$ submatrice into $\frac{n}{y}$ pieces (e.g., to further trade circuit size for cycles in hardware) but this has no impact on the security of the RLWPR assumption, since the internal computations are assumed to be secure thanks to masking, as per Footnote 22. By contrast, more parallel implementations (reflected by a large y) may increase the level of noise in the measurements and therefore the security of the masked computations [DFS15]. So overall, the security of the above RLWPR problem only depends on n and x. For a similar reason, the polynomial multiplication can be implemented naively or in the NTT domain, as long as the inverse NTT is applied on every share before recombination. A more efficient solution for the NTT case would be to recombine the shares in the NTT domain (so that the inverse NTT is computed only once). This would provide the adversary with leakages having a slightly different structure than in the above RLWPR problem. We leave the security analysis of this variant as an interesting scope for further research.

C. Choice of parameters and cryptanalysis challenges. Applying the security analysis of LWPR described in Part A of this subsection to RLWPR, we could choose instances based on the main security parameter n using the parallelism parameter x to obtain security margins (a necessary condition to reach λ bits of security is that $(n+1)\log_2 q + 3\log_2 n \geq \lambda$). However, as mentioned in Part B of this subsection, the RLWPR problem is not exactly the same as the LWPR one. Negatively, the $(x \times n)$ submatrices are Toeplitz and they are not independent. While using structured matrices is not unusual in the context of hard learning problems (see for example [LPR10] for RLWE or [GRS08a; Kil+17] for LPN variants) and we could not identify parts of the analyses in [Duv+21] that become significantly easier in this case,

the corresponding problems are less studied. It is therefore natural to consider additional security margins. As for the non-independence issue, considering that the security of the full RLWPR is at least as strong as the security of one of its subproducts, we can conservatively assume that one RLWPR sample will generate at most $\frac{n}{x}$ leakages about this subsecret. So parameters' choices covering that the data complexity of attacks against RLWPR is reduced by this factor compared to attacks against LWPR should be safe. Positively, the secret s in the leveled implementation of POLKA is not multiplied with a public r since this r value is dummied. So concretely, the side-channel adversary will only be provided with the leakage of this ephemeral value.

Putting things together, and considering the instances proposed in Section 3.3.3, we propose the following sets of parameters as interesting targets for cryptanalysis with a time complexity of less than 2^{128} and at most 2^{64} queries to a $\mathsf{RLWPR\text{-}b}^{n,x,y}_{\mathsf{HW}_\mathsf{g},q}(\boldsymbol{s})$, assuming a Hamming weight leakage function.

	$\log_2(q)$	n	x
Set 1	31	1024	8
Set 2	31	1024	4
Set 3	16	1024	8

We leave the investigation of more aggressive parameters (especially reducing the level of parallelism x) as interesting directions for further research.

3.3.5 Discussion and open problems

The uniform protection of all the operations in recent post-quantum CCA-secure public key encryption schemes against side-channel attacks is known to be very expensive. To the best of our knowledge, POLKA is the first scheme for which a protected implementation can be leveled, mixing operations that only require SPA security with a few operations that require DPA security but can be efficiently masked. We reach this goal by mixing various ideas which we believe of independent interest.[23] We also believe these techniques are generic and could be exploited for other schemes. For example, a leakage-resistant variant of the NTRU cryptosystem is discussed in the ePrint version of this work (alongside other parameters with binomial errors and details on AE).

[23] For example, even a POLKA design that does not rely on the RLWPR assumption to unmask some computations would remain interesting compared to the NIST post-quantum finalists, thanks to the significant reduction of masked symmetric cryptographic primitives that dominate their performance figures [BC22b].

Our results also lead to a number of interesting open problems. First, the RLWPR assumption on which a part of POLKA's physical security relies is an admittedly recent one. So further cryptanalysis (e.g., generalized to wide classes of realistic leakage functions) is an important scope for further research. The investigation of such hard physical learning problems in increasingly serial implementations is another promising direction, as it could lead to their exploitation in a software context. The same holds for a NTT-LWPR variant of RLWPR that would allow re-combining shares in the NTT domain, therefore leading to more efficient multiplications and, in general, for efforts towards a more unified / less specialized view of hard physical learning problems. More related to the high-level design of POLKA, it would be interesting to study options to further improve its potential for leveling (e.g., by removing the possibility of DPA against the hash function of step 5). From a theoretical viewpoint, evaluating whether post-quantum and leakage-resistant schemes could take advantage of ciphertext compression would be relevant as well. Eventually, the first leakage analysis we provide in this work is based on the heuristic (attack-based) approach of [Bel+20]. So formalizing and proving the leakage security of POLKA with an appropriate set of physical assumptions is a necessary long-term goal.

4 Physical learning problems applications to side-channel analysis

In this chapter, we explore the connection between the side-channel security of post-quantum cryptographic schemes and physical learning problems. We illustrate this link through various examples and examine the proven or heuristic hardness of these problems. The first three sections of this chapter are based on a peer-reviewed publication I co-authored, while the last one is yet to be published.

In Section 4.1, we investigate the implementation of the Learning Parity with Physical Noise (LPPN) problem, which arises when using inexact computing to implement LPN instances. In this context, inexact computing allows the noisy inner product in LPN-based schemes to be computed without explicitly generating the error. Instead, the noise obscuring the inner product is introduced by the faulty execution of the hardware implementing it. In this work, we do not delve into hardware implementation but focus on revealing new output data dependencies in the error probabilities that LPPN samples may suffer from. We formally prove that LPPN instances with such dependencies retain the same hardness as the standard LPN problem.

In Section 4.2.2, we extend the LPPN concept to the Learning With Errors (LWE) problem by introducing Learning With Physical Errors (LWPE). We first demonstrate that directly applying the design principles of LPPN prototypes leads to new mathematical data dependencies in the error distributions, potentially weakening the security of the underlying problem. We then show how specific design modifications can mitigate this issue, ensuring that LWPE samples are inherently robust against such dependencies. Moreover, we argue that these design ideas create a broad space for innovation, making hard physical learning problems applicable to a wide range of use cases.

In Section 4.3, we analyze the heuristic hardness of the LWPR problem (presented in Section 2.2.2). Initially, we reinforce the notion that LWPR cannot be secure in a serial implementation context without additional countermeasures due to practical attacks that exploit worst-case

leakages with realistic data complexity. We then extend this analysis to parallel implementations. On one hand, we generalize LWPR's resilience to cryptanalysis that leverages any form of leakage, not just worst-case scenarios. Previous research claimed security in the context of Hamming weight leakage functions; we clarify the conditions necessary to maintain this security, particularly regarding the leakage function's degree and the precision of its coefficients. On the other hand, we demonstrate that parallelism naturally enhances security against attacks targeting worst-case leakages. We further validate these theoretical findings through experimental results from an example implementation.

Finally, in Section 4.4, we introduce and examine a generalized unstructured learning problem called LWE-OD. This problem is defined to encompass LWE, LWR, and more exotic variants like LWPR. We provide a reduction from LWE-OD to LWE, formulated as a system of equations in which the unknowns are the error distributions of the LWE and LWE-OD samples.

4.1 Physical security from inexact computing and learning problems reductions, adapted from [Bel+21]

For more than two decades, learning in the presence of errors has been considered as an interesting source of computationally hard problems [Kea93; Hås97], which have in turn found many applications in the design of provably secure cryptographic schemes [Reg10; Pie12]. Among the possible instances of such hard problems, the Learning Parity with Noise (LPN) problem is probably the simplest one. It has given rise to a wide literature about efficient authentication protocols, starting with the scheme of Hopper and Blum (HB) [HB01] and extended in various directions [JW05; GRS08a; Kil+11; Dod+12; Hey+12]. It can also be used to build one way-functions [Blu+93], secret-key encryption schemes [GRS08b], public-key encryption schemes [DV13], pseudorandom functions [YS16] or hash functions [Yu+19]. Extensions of the LPN problem turn out to be even more versatile. For example, the Learning With Errors (LWE) problem introduced by Regev [Reg05b] can additionally be used to build identity-based encryption [GPV08; Cas+10] or fully homomorphic encryption [Gen09; BV11a]. Its variants are the basis for many post-quantum cryptographic schemes [BL17] and are in particular the core assumption for CRYSTALS-Dilithium [Duc+18], CRYSTALS-Kyber [Bos+18b], which are both finalists of the NIST

Post-Quantum (PQ) cryptography competition.[1] Finally, large-modulus LPN has recently found advanced applications in secure multi-party computation [Boy+18; Boy+19], arithmetic cryptography [AAB17] and indistinguishability obfuscation [JLS20].

Whenever such primitives become practical to the point of being deployed, their secure and efficient implementation naturally becomes a critical problem to solve. This is for example the case of LNP-based authentication protocols and PQ schemes. In this respect, various results showed that the conceptual simplicity of LPN-based protocols does not directly translate into concrete advantages [BL12; AHM14]. This is in part due to the cost of generating random numbers that those solutions require. Similarly, while it could be expected that the simple algebraic structure of LPN-based protocols could directly lead to implementations secure against side-channel attacks, it turned out the need to protect its (additive) Bernoulli noise implies significant overheads as well [GLS14].

As a result of these limitations, Kamel et al. observed that by leveraging recent advances in inexact computing (a trend that is of independent interest in view of the miniaturization of electronic devices [GR10]), one could implement LPN-based schemes without explicitly generating random numbers [Kam+20b]. They formalized the corresponding problem as the Learning with Physical Noise (LPPN) problem, and highlighted its potential advantages in terms of implementation cost (since it removes the need to explicitly generate randomness for the additive errors) and implementation security (since it prevents the trivial attack probing the randomness in a leaking implementation). A first ASIC prototype of LPPN processor was then described in [Kam+18] and its side-channel security has been evaluated in [Kam+20a]. Nevertheless, and despite these promising and quite unique features, the understanding of the security and applicability of LPPN remains limited. A detailed background on LPPN was introduced in the above background 2.2.2.

On the one hand, LPPN is a physical assumption (in contrast with the mathematical LPN assumption), raising the question whether the errors generated thanks to inexact computing (which may not exactly follow a Bernoulli distribution) are leading to hard problems? For example, the investigations in [Kam+20b; Kam+18] showed that the error probability of LPPN samples may depend on the Hamming weight of the corresponding challenges, and provided heuristic evidence that these dependencies can be limited.

On the other hand, the prototype implementations in these previous works were heavily relying on an accurate control of frequency and volt-

[1] `https://csrc.nist.gov/projects/post-quantum-cryptography`.

age over-scaling, raising the question whether such ideas are applicable with off-the-shelf devices like FPGAs?

Our contributions to these questions are threefold:

First, we further study the physical imperfections that can affect the security of the LPPN problem. In particular, we show that besides the dependencies of the LPPN error probability on its input challenges, LPPN prototypes can also suffer from output dependencies. That is, the probability of error of LPPN samples can depend on the (correct) value of its output. We also show that certain types of implementations lead to "structured errors" (i.e., the error at cycle i depends on the output at cycle i-1). We then discuss options to mitigate these issues, which turn out to be easy to fix by design for the structured errors, and hard to completely cancel for the output dependencies. Hence, these observations question the equivalence between LPPN and LPN.

Second, and since it is well known that tweaking the LPN problem can make it weaker [AG10], we formalize the exhibited output dependencies as a (new) LPN problem with Output Dependencies (LPN-OD) and we show that solving the LPN-OD problem is as hard as solving an LPN problem with adapted security parameters (depending on the output dependencies). This reduction allows us to gain provable confidence that certain types of physical imperfections do not completely ruin the hardness of LPPN problems. It also opens the way towards formalizing other imperfections with similar reductions.

As a side-result, we finally discuss how much masking the LPPN implementation, which is required in order to reach high security levels against side-channel attacks, can also be used to mitigate the error dependencies of LPPN samples.[2] This observation strengthens the relevance of the LPPN assumption, since it implies additional concrete advantages in the practically important context of side-channel secure implementations.

Overall, the LPPN assumption is admittedly a provocative one, and a lot of work remains needed to gain confidence in its security and usability. Besides, its strongest potential use case is for the side-channel secure implementation of PQ cryptographic algorithms and most of them rely on more general LWE problems (for which a physical counterpart is also outlined in [Kam+20b]). Yet, we believe understanding the security of LPPN in front of possible physical imperfections (like input or output er-

[2] The high-level intuition that masking helps against such dependencies is already given in [Kam+20b] but we further detail it and connect it with side-channel security requirements.

ror dependencies) is a necessary first step in this direction. We hope our results can be used as a seed towards both the theoretical investigation of other hard physical learning problems and the practical investigation of inexact computing in this context. We also believe such a risky research path is justified by the high importance of obtaining secure and efficient PQ cryptographic implementations in the future.

4.1.1 LPPN physical imperfections

It has already been put forward in [Kam+20b; Kam+18] that the errors of an ϵ-PIP may exhibit input data dependencies. For example, denoting the bitwise AND between $\boldsymbol{x}$ and $\boldsymbol{k}$ as $\boldsymbol{x} \cdot \boldsymbol{k}$ and the Hamming weight function as $\mathsf{HW}(.)$, it is known that the error probability of a serial implementation of ϵ-PIP based on frequency or voltage over-scaling decreases with $\mathsf{HW}(\boldsymbol{x} \cdot \boldsymbol{k})$. It has also been shown that such input dependencies are significantly reduced in the case of parallel ϵ-PIP architectures. Given the easier error control of serial ϵ-PIP architectures, this has led the authors of [Kam+20b; Kam+18] to consider mixed (parallel then serial) ϵ-PIP implementations as a good tradeoff for secure LPPN instances. In this section, we complement these first analyses of LPPN physical defaults by analyzing the output data dependencies and possible structure of physical errors. For this purpose, we simulate an ASIC design that is essentially similar to the one in [Kam+18] and is comprised of three parts: an inner product architecture, a variable delay line and an error controller. By controlling the output sampling clock thanks to the variable delay line, this design allows accurately controlling the error rate. Implementation details are not necessary for the understanding of this section, but are given in published version of this work.

Output data dependencies of the errors and mitigation

In the LPN setting, the error $e \leftarrow \mathsf{Ber}_\epsilon$ is completely independent on the output of the inner product, since it is generated separately thanks to a Random number Generator (RNG) or Pseudo-Random Number Generator (PRNG). By contrast, in the LPPN setting, inexact computations are deployed to implement a noisy inner product with embedded errors. Since these errors are generated by sampling the output incorrectly, it is therefore natural to investigate whether the error probability depends on the output data.

Evaluation settings.

Our goal in this first subsection is to show that LPPN samples can be affected by output dependencies, and that these dependencies can to

some extent be mitigated thanks to adequate architectural choices. For this purpose, we next analyze various solutions to compute 32-bit LPPN samples and compare the output dependencies of their error probabilities The cases we consider are listed next and illustrated in Figure 4.1:

- Case A (parallel architecture). The 32-bit ϵ-PIP is implemented with a tree of depth 6 (1 AND stage & 5 XOR stages). The glitches are minimized since there are limited imbalances between the routing paths of parallel architecture. The variable delay line (used to control the output sampling clock) is fed with an ideal clock source.

- Case B (serial architecture). The 32-bit ϵ-PIP is implemented with 32 sequential stages (1 AND stage & 31 XOR stages). The amount of glitches increases with the logic depth of the combinatorial design because the imbalances between the routing paths are maximized. Minimum sized gates provided by the foundry are used (next denoted as 0.5X). The variable delay line is fed with an ideal clock source.

- Case C (parallel architecture with jitter). This is the same parallel architecture as described above (in case A), but with a jittery clock source feeding the variable delay line such that the output is sampled with some random inaccuracies. As a result, the errors are due to both deterministic and probabilistic (jitter) effects.

- Case D (serial architecture with power gating). This case uses the same serial architecture as described above (in case B) but we add some power gating cells to the ground net in order to reduce the current that drives a zero. It allows balancing the propagation time of the $0 \leftarrow 1$ and $1 \leftarrow 0$ transitions and implies a more balanced generation of glitches towards 0 or 1, aimed to reduce the output dependencies.

- Case E (serial architecture with balanced gates and power gating). This is the same as case D but we use bigger and more balanced gates (next denoted as 4X).

Our case studies were implemented in a 65nm TSMC CMOS technology using the regular voltage threshold (RVT) flavor. Pre-layout simulations were performed using ULTRASIM within the CADENCE environment. We used 10^4 uniformly distributed random 32-bit input $\boldsymbol{x}$ vectors and a single uniformly distributed random 32-bit secret key $\boldsymbol{k}$. The supply voltage is fixed at nominal 1.2V and the clock frequency

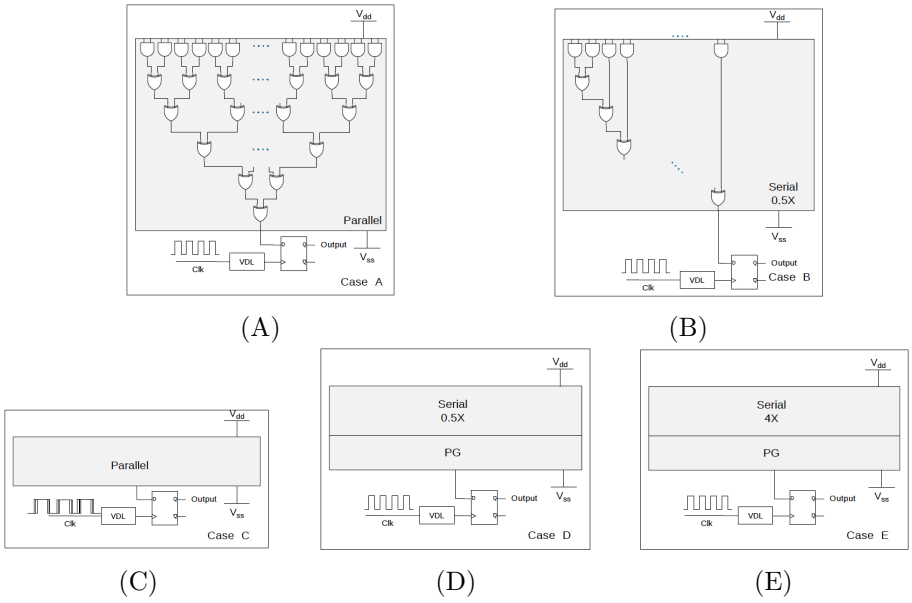


Figure 4.1: ϵ-PIP architectures.

is 2MHz. For each case, an adapted variable delay line is designed in order to sample the output of the inner product incorrectly, aiming to achieve an error probability close to 0.25. In the parallel with jitter case, we used a clock source with 200ps RMS jitter (the latter refers to the standard deviation of the jitter's Gaussian distribution). Finally, we took advantage of the readily available power gating standard cells of the 65nm TSMC CMOS technology in order to implement the serial architecture with power gating (i.e., case D).

In our evaluations, we estimate the error probability of our ϵ-PIPs in two parts. First the error probability ϵ_0 if $\langle \boldsymbol{x}, \boldsymbol{k} \rangle = 0$, second the error probability ϵ_1 if $\langle \boldsymbol{x}, \boldsymbol{k} \rangle = 1$. Then, we compute the following normalized difference between the error probabilities as:

$$\Delta = \frac{\left| \hat{\Pr}[e \leftarrow \epsilon_0\text{-PIP}] - \hat{\Pr}[e \leftarrow \epsilon_1\text{-PIP}] \right|}{2}.$$

Output data dependencies results.
The output data dependencies captured by Δ for all the case studies are shown in Figure 4.2 and lead to a number of useful observations that we list next:

1. The error probability of the parallel architecture (case A) shows significant data dependencies ($\Delta \approx 4.2\%$). Their main explana-

tion is that the errors in this case are mostly generated by the incorrect sampling of the previous correct output bit by an ideal clock source. In other words, these are structured errors (as will be discussed in the next subsection) and they are mostly deterministic, since they depend on the propagated data. The fact that the first stage of the ϵ-PIP is an AND then favors an output zero rather than one. Hence, the probability of having an error given that the correct output is one is higher than in the opposite case, leading to an imbalance.

2. The errors of the serial architecture (case B) suffer from significant data dependencies as well ($\Delta \approx 3.7\%$). However, the reason is quite different than in the parallel case. In the serial architecture the errors are generated by sampling the deterministic glitches. We posit that imbalanced glitches are in cause in this case. Indeed, we used minimum sized gates that provide different propagation times in case the gate output goes from high to low or the inverse. More precisely, these gates switch their outputs faster to zero than they do to one. As a result, we observe a higher probability of error given that the correct output is one than in the opposite case.

3. Adding a jittery clock source to the parallel architecture (case C) is a nice solution to reduce the errors' data dependencies below 1%. Intuitively, a jittery clock implies more probabilistic (random) inaccuracies in the sampling process of the output. As a result, even though the error is still the result of sampling the previous correct bit, the use of a jittery clock masks the data dependency to a good extent.

4. As an attempt to mitigate the errors' data dependencies for the serial architecture, we considered a gate-level solution that reduces the current that drives a zero by means of power gating cells applied only to the ground net (case D). As it is clear from Figure 4.2, this proposal reduced the errors' data dependencies ($\Delta \approx 1.6\%$).

5. Finally, we explored differently sized gates that have more driving strength (therefore are bigger) and more balanced propagation time (i.e., the time needed for the gate's output to go from high to low is close to the time it needs to go from low to high) along with power gating applied to the ground net (case E). Clearly, this solution significantly reduces the errors' data dependencies so that Δ is below 1%.

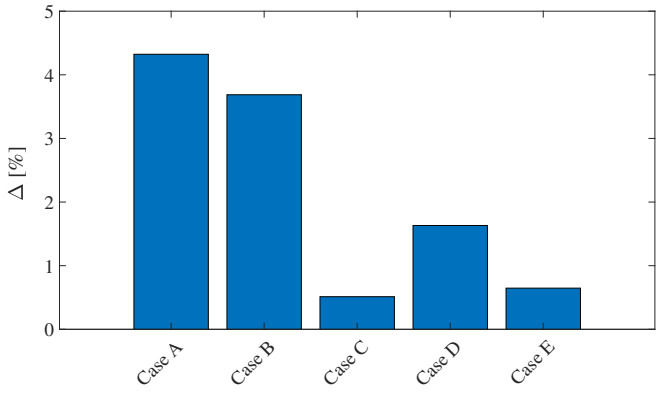


Figure 4.2: Output data dependencies of the probability of error of different ϵ-PIP architectures. Case A: parallel architecture. Case B: serial architecture with minimum sized gates. Case C: parallel architecture with jitter. Case D: serial architecture with minimum sized gates and power gating. Case E: serial architecture with bigger gates and power gating.

We conclude from these preliminary simulations that significant output dependencies can be observed in LPPN samples. These dependencies can be reduced thanks to ASIC design tweaks and their detailed analysis and improvement is an interesting scope for further research. However, it is unlikely to cancel such physical defaults completely. We use this result as an important motivation for analyzing these dependencies and proving that they do not cancel the hardness of the LPPN problem in Section 4.1.2.

Structured errors in parallel implementations and mitigation

Besides the output data dependencies of the error distribution, another physical default which is quite specific to parallel ϵ-PIPs is the fact that LPPN errors can be structured. More precisely, when two consecutive (parallel) inner products are computed, namely $\langle \boldsymbol{x1}, \boldsymbol{k} \rangle$ and $\langle \boldsymbol{x2}, \boldsymbol{k} \rangle$, the result of the second inner product can either be correct (i.e., equal to $\langle \boldsymbol{x2}, \boldsymbol{k} \rangle$) or incorrect and equal to the previous correct value $\langle \boldsymbol{x1}, \boldsymbol{k} \rangle$ (but wether it is one or the other option that is observed remains unknow to the adversary).

To demonstrate this physical default, we estimated the probability that the second error corresponds to the correct output value given by the first inner product computation, that is $\Pr[e_2 = \langle \boldsymbol{x1}, \boldsymbol{k} \rangle]$, for both

the parallel (A) case and serial (B) case of the previous subsection. For the parallel implementation, this probability was worth 0.98 while it was of 0.53 for the serial one. The (much) less structured errors of the serial implementation are explained by the fact that errors are then sampled from glitches and therefore, there is a nearly balanced chance that the glitch is equal to the previous correct output.

We note that such structured errors are easy to prevent at the design level. One option is to output one every two inner product computations (which halves the throughput). This way the neglected output conceals the previous correct one. A more efficient solution would be to use a dual-edge clock implementation, where the inputs are sampled at both the rising and falling edges of the clock, and only considering the output due to one of the clock edges. Most interestingly, this problem also vanishes in case we mask the implementation since inner product computations are randomized in this case. Since structured errors are quite specific to parallel implementations and easy to fix by design, we do not formalize them further and next focus on the more critical issue of output data-dependent errors.

4.1.2 Reduction between LPN and LPPN

The previous section showed that the ϵ-PIPs generating LPPN samples may be affected by physical defaults that can create output dependencies in the error distribution. Due to their physical nature, it is unlikely to completely cancel these dependencies at the design level. In this section, we therefore aim to demonstrate that despite these dependencies, the security provided by the corresponding LPPN samples does not fundamentally differ from the security of LPN samples. For this purpose, and as usual when reasoning about physical primitives, we face a trade-off between the generality of security claims and their practical relevance. That is, in theory, there are as many LPPN instances with output dependencies as there are physical functions. But providing a separate formal treatment for each physical function is not possible. We deal with this issue by specifying a family of LPN problems that covers classes of imperfections that may be observed in practice, which we denote as the LPN with Output Dependencies (LPN-OD) problem. We then show a self-reduction of the LPN-OD problem, which gives in particular an equivalence between the LPN-OD problem and the standard LPN problem, with given parameters. The equivalence ensures the intractability of LPN-OD from the one of LPN. We show how to use it to estimate parameters for our physical problem at the end of the section.

Notations

We first define the LPN problem with output error dependencies as follows:

Definition 15 (LPN problem with output dependencies)**.** *Let $\langle ., . \rangle$ denote the binary inner product, $\boldsymbol{k}$ be a random n-bit secret, $\epsilon \in [0, \frac{1}{2}]$ be a noise parameter, Ber_ϵ be the Bernoulli distribution with parameter ϵ and $D_{\boldsymbol{k}}^{\epsilon_0,\epsilon_1}$ be the distribution defined as:*

$$D_{\boldsymbol{k}}^{\epsilon_0,\epsilon_1} =: \{(\boldsymbol{x}, \langle \boldsymbol{x}, \boldsymbol{k} \rangle \oplus e) : \boldsymbol{x} \leftarrow \{0,1\}^n; e \leftarrow \begin{cases} \mathsf{Ber}_{\epsilon_0} \text{ if } \langle \boldsymbol{x}, \boldsymbol{k} \rangle = 0 \\ e \leftarrow \mathsf{Ber}_{\epsilon_1} \text{ otherwise} \end{cases} \}.$$

Let $\mathcal{O}_{\boldsymbol{k}}^{\epsilon_0,\epsilon_1}$ denote an oracle outputting independent samples according to the $D_{\boldsymbol{k}}^{\epsilon_0,\epsilon_1}$. The LPN-$OD^n_{\epsilon_0,\epsilon_1}$ problem is said to be (q, t, m, θ)-hard to solve if for any algorithm A,

$$\Pr[\boldsymbol{k} \leftarrow \{0,1\}^n : A^{\mathcal{O}_{\boldsymbol{k}}^{\epsilon_0,\epsilon_1}}(1^n) = \boldsymbol{k}] \leq \theta,$$

and A runs in time $< t$, with memory $< m$ and makes at most q queries to $\mathcal{O}_{\boldsymbol{k}}^{\epsilon_0,\epsilon_1}$.

Proposition 5 (LPN-OD alternative characterization)**.** *Let $n \in \mathbb{N}, \boldsymbol{k} \in \mathbb{F}_2^n \setminus \{0\}, \epsilon_0, \epsilon_1 \in [0, 1/2]$. Samples $(\boldsymbol{a}, b) \in \mathbb{F}_2^n \times \mathbb{F}_2$ follow $D_{\boldsymbol{k}}^{\epsilon_0,\epsilon_1}$ if and only if $\boldsymbol{a}$ follows a uniform distribution and the following equations hold:*

$$\begin{cases} Pr[b = 1] = \dfrac{1}{2} + \dfrac{\epsilon_0 - \epsilon_1}{2}, \\ Pr[\langle \boldsymbol{a}, \boldsymbol{k} \rangle = 0 \,|\, b = 0] = \dfrac{1 - \epsilon_0}{1 - \epsilon_0 + \epsilon_1}, \\ Pr[\langle \boldsymbol{a}, \boldsymbol{k} \rangle = 0 \,|\, b = 1] = \dfrac{\epsilon_0}{1 + \epsilon_0 - \epsilon_1}. \end{cases}$$

Proof. Let $(\boldsymbol{a}, b) \in \mathbb{F}_2^n \times \mathbb{F}_2$ be sampled from $D_{\boldsymbol{k}}^{\epsilon_0,\epsilon_1}$ with $b = \langle \boldsymbol{a}, \boldsymbol{k} \rangle + e$. First, we prove that, for $b \in \{0,1\}, \Pr[\langle \boldsymbol{a}, \boldsymbol{k} \rangle = b] = \frac{1}{2}$.

Let $j = \min\{i \in [n] \,|\, \boldsymbol{k}_i = 1\}$ (j is well defined since $\boldsymbol{k} \neq 0$), therefore $\langle \boldsymbol{a}, \boldsymbol{k} \rangle = \left(\sum_{i \neq j} \boldsymbol{a}_i \boldsymbol{k}_i \right) + \boldsymbol{a}_j$. Since $\boldsymbol{a}_j$ is uniformly random over $\mathbb{F}_2$ and $b \in \{0,1\}$, it gives $\Pr[\langle \boldsymbol{a}, \boldsymbol{k} \rangle = b] = \frac{1}{2}$. We can then show:

$$\begin{aligned} \Pr[b = 1] &= \Pr[\langle \boldsymbol{a}, \boldsymbol{k} \rangle = 0 \cap e = 1] + \Pr[\langle \boldsymbol{a}, \boldsymbol{k} \rangle = 1 \cap e = 0] \\ &= \frac{1}{2}(\epsilon_0 + 1 - \epsilon_1) = \frac{1}{2} + \frac{\epsilon_0 - \epsilon_1}{2}, \end{aligned}$$

$$\Pr\left[\langle \boldsymbol{a}, \boldsymbol{k}\rangle = 0 \,|\, b = 0\right] = \Pr\left[b = 0 \,|\, \langle \boldsymbol{a}, \boldsymbol{k}\rangle = 0\right] \frac{\Pr\left[\langle \boldsymbol{a}, \boldsymbol{k}\rangle = 0\right]}{\Pr\left[b = 0\right]} = \frac{1-\epsilon_0}{1-\epsilon_0+\epsilon_1},$$

$$\Pr\left[\langle \boldsymbol{a}, \boldsymbol{k}\rangle = 0 \,|\, b = 1\right] = \Pr\left[b = 1 \,|\, \langle \boldsymbol{a}, \boldsymbol{k}\rangle = 0\right] \frac{\Pr\left[\langle \boldsymbol{a}, \boldsymbol{k}\rangle = 0\right]}{\Pr\left[b = 1\right]} = \frac{\epsilon_1}{1+\epsilon_0-\epsilon_1}.$$

For the reverse implication, since $\boldsymbol{a}$ is uniformly distributed and the three probabilities $\Pr[b = 1]$, $\Pr[\langle \boldsymbol{a}, \boldsymbol{k}\rangle = 0 \,|\, b = 1]$ and $\Pr[\langle \boldsymbol{a}, \boldsymbol{k}\rangle = 0 \,|\, b = 1]$ uniquely define the pair $\Pr[b = 1 \,|\, \langle \boldsymbol{a}, \boldsymbol{k}\rangle = 0] = \epsilon_0$ and $\Pr[b = 0 \,|\, \langle \boldsymbol{a}, \boldsymbol{k}\rangle = 1] = \epsilon_1$, it allows to conclude. □

Remark. LPN-OD is a more general problem than LPN. Indeed, for $\epsilon_0 = \epsilon_1$, $\text{LPN-OD}^n_{\epsilon_0,\epsilon_1}(\boldsymbol{k})$ is equivalent to $\text{LPN}^n_{\epsilon_0}(\boldsymbol{k})$. Besides, the two problems defined above are so-called *search* problems, where the goal of the adversary is to recover $\boldsymbol{k}$. One could also define a *decision* version of these two problems, that will not be studied in this paper.

Reduction

We next show that the standard LPN (resp., LPN-OD) problem is as hard as the LPN-OD (resp., standard LPN) problem and there is both a polynomial reduction from LPN-OD to LPN and from LPN to LPN-OD. To do so, we use a self-reduction of the LPN-OD problem and show that we can transform in polynomial time an instance of LPN-OD into a noisier one with the same key and a possibly different balance between the two noise parameters. Precisely, we show that a probabilistic polynomial time algorithm can produce an instance of $\text{LPN-OD}^n_{\epsilon'_0,\epsilon'_1}(\boldsymbol{k})$ from an instance of $\text{LPN-OD}^n_{\epsilon_0,\epsilon_1}(\boldsymbol{k})$ (for noise parameters satisfying restrictions specified later). Fixing $\epsilon'_0 = \epsilon'_1$ gives a reduction from LPN-OD to LPN (breaking LPN breaks LPN-OD) and fixing $\epsilon_0 = \epsilon_1$ gives a reduction from LPN to LPN-OD (breaking LPN-OD breaks LPN). The algorithm has access to an oracle sampling $D_{\boldsymbol{k}}^{\epsilon_0,\epsilon_1}$ with ϵ_0 and ϵ_1 known, and the reduction is based on two main ideas: the control of the (un)-balancedness of the b part, and the addition of two Bernouilli distributions.

First, as seen in Proposition 5, the probability $\Pr[b = 1]$ depends on the noise parameters. Hence, the algorithm uses a method of rejection sampling in order to produce samples with b having the target probability of samples from $D_{\boldsymbol{k}}^{\epsilon'_0,\epsilon'_1}$. Second, a standard idea to turn LPN samples into noisier samples in a black-box manner is to add a Bernoulli on the b component. Using this idea, LPN-OD samples have two noise parameters linked with the value of the correct inner product computation and the algorithm acts differently on these two parameters, adding

two Bernouilli's next denoted as t_0 and t_1, and choosing which one to add based on the value given by the rejection sampling.

We first let SampleLPN-OD$^n_{\epsilon_0,\epsilon_1}(\boldsymbol{k})$ describe the oracle $\mathcal{O}^{\epsilon_0,\epsilon_1}_{\boldsymbol{k}}$ (from Definition 15) that returns samples from the distribution $D^{\epsilon_0,\epsilon_1}_{\boldsymbol{k}}$ for a key $\boldsymbol{k}$ in Algorithm 12.

Algorithm 12 SampleLPN-OD$^n_{\epsilon_0,\epsilon_1}(\boldsymbol{k})$

Output: LPN-OD sample $(\boldsymbol{a}, b)$

$\boldsymbol{a} \xleftarrow{\$} \mathbb{F}_2^n, e \xleftarrow{\$} \mathsf{Ber}_{\epsilon_i}$ where $i = \langle \boldsymbol{a}, \boldsymbol{k} \rangle$
$(\boldsymbol{a}, b) \leftarrow (\boldsymbol{a}, \langle \boldsymbol{a}, \boldsymbol{k} \rangle + e)$

We then use this sampler in Algorithm 13. The idea is to start by fixing the b of the output in order to ensure that it follows the good distribution. We then control the output probabilities by adding different Bernoulli errors t_0, t_1 depending on the fixed value of b. The correctness of our algorithm is given with the following lemma:

Lemma 3. *For any non-zero $\boldsymbol{k} \in \mathbb{F}_2^n$, $m \in \mathbb{N}$, error parameters $\epsilon_0, \epsilon_1 \in \left[0, \frac{1}{2}\right]$, $t_0, t_1 \in \left[0, \frac{1}{2}\right]$ such that $t_0 + t_1 \neq 1$, if both $\begin{cases} \epsilon'_0 = \frac{2(\epsilon_0(2t_1-1)-t_1)t_0-\epsilon_0(2t_1-1)+t_1}{1-t_0-t_1} \\ \epsilon'_1 = \frac{(2\epsilon_1(2t_1-1)-2t_1+1)t_0-\epsilon_1(2t_1-1)}{1-t_0-t_1} \end{cases}$ remain in $\left[0, \frac{1}{2}\right]$, then the output of AltSampleLPN-OD$^n_{\epsilon_0,\epsilon_1,t_0,t_1,m}(\boldsymbol{k})$ follows the distribution $D^{\epsilon'_0,\epsilon'_1}_{\boldsymbol{k}}$, which is the same distribution as SampleLPN-OD$^n_{\epsilon'_0,\epsilon'_1}(\boldsymbol{k})$'s output.*

Proof. We study the sample output probabilities of algorithm 13 and use Proposition 5 to show that it follows the distribution $D^{\epsilon'_0,\epsilon'_1}_{\boldsymbol{k}}$. First, note that $\boldsymbol{a}$ is chosen uniformly. We aim at comparing the output probabilities with those of a regular sampler.

Keeping the notations of the algorithm, we have that $\Pr[b = 1] = \Pr[b' = 1] = \frac{1}{2} + \frac{\epsilon'_0 - \epsilon'_1}{2}$. Therefore, it verifies the first equality of Proposition 5.

We now need to express $\Pr[\langle \boldsymbol{a}, \boldsymbol{k} \rangle = 0 \,|\, b = 0]$ and $\Pr[\langle \boldsymbol{a}, \boldsymbol{k} \rangle = 0 \,|\, b = 1]$ with the inputs and parameters of Algorithm 13: $\epsilon_0, \epsilon_1, t_0, t_1$. Note that the sum of two Bernoulli variables of parameters ϵ, ϵ' is a Bernoulli variable of parameter $\epsilon + \epsilon' - 2\epsilon\epsilon'$. With that in mind, for $i, j \in \{0, 1\}$, we introduce new error parameters $E_{i,j} = \epsilon_i + t_j - 2\epsilon_i t_j$. Since $b = \langle \boldsymbol{a}, \boldsymbol{k} \rangle + e + e'$ is computed as the sum of a LPN-OD sample and a Bernoulli, where $e \xleftarrow{\$} \mathsf{Ber}_{\epsilon_{\langle \boldsymbol{a}, \boldsymbol{k} \rangle}}$ and $e' \xleftarrow{\$} \mathsf{Ber}_{t_{b'}}$, and inside the loop, $(\boldsymbol{a}, b)$ follows an independent distribution $D^{E_{0,b'}, E_{1,b'}}_{\boldsymbol{k}}$, applying Proposi-

Algorithm 13 AltSampleLPN-OD$^n_{\epsilon_0,\epsilon_1,t_0,t_1,m}(\boldsymbol{k})$

Input: A sampling upper limit $m \in \mathbb{N}$, error parameters $\epsilon_0, \epsilon_1 \in \left[0, \frac{1}{2}\right]$, Bernoulli parameters $t_0, t_1 \in \left[0, \frac{1}{2}\right]$ with $t_0 + t_1 \neq 1$

Output: A sample $(\boldsymbol{a}, b) \in \mathbb{F}_2^n \times \mathbb{F}_2$ following the distribution $D_{\boldsymbol{k}}^{\epsilon'_0,\epsilon'_1}$

$\epsilon'_0 = \frac{2\,\epsilon_0 t_0 - (2\,(2\,\epsilon_0 - 1)t_0 - 2\,\epsilon_0 + 1)t_1 - \epsilon_0}{t_0 + t_1 - 1}$
$\epsilon'_1 = \frac{(2\,\epsilon_1 - 1)t_0 - 2\,((2\,\epsilon_1 - 1)t_0 - \epsilon_1)t_1 - \epsilon_1}{t_0 + t_1 - 1}$
if $\epsilon'_0 \notin \left[0, \frac{1}{2}\right]$ or $\epsilon'_1 \notin \left[0, \frac{1}{2}\right]$ **then**
  return $\perp_0$
$b' \xleftarrow{\$} \mathsf{Ber}_{\frac{1}{2} + \frac{\epsilon'_0 - \epsilon'_1}{2}}$
cnt := 0
do
  $(\boldsymbol{a}, b'') \xleftarrow{\$}$ SampleLPN-OD$^n_{\epsilon_0,\epsilon_1}(\boldsymbol{k})$
  $e' \xleftarrow{\$} \mathsf{Ber}_{t_{b'}}$
  $(\boldsymbol{a}, b) \leftarrow (\boldsymbol{a}, b'' + e')$
  cnt $\leftarrow$ cnt +1
while $b \neq b'$ and $cnt \leq m$
if cnt = m **then**
  return $\perp$
return $(\boldsymbol{a}, b)$

tion 5 gives us $\begin{cases} \Pr[\langle \boldsymbol{a}, \boldsymbol{k} \rangle = 0 \,|\, b = 0] = \frac{1 - E_{0,b'}}{1 - E_{0,b'} + E_{1,b'}} \\ \Pr[\langle \boldsymbol{a}, \boldsymbol{k} \rangle = 0 \,|\, b = 1] = \frac{E_{0,b'}}{1 + E_{0,b'} - E_{1,b'}} \end{cases}$. After the loop condition filtering $b = b'$, for $i \in \{0, 1\}$, we get $\Pr[\langle \boldsymbol{a}, \boldsymbol{k} \rangle = 0 \,|\, b = i] = \Pr[\langle \boldsymbol{a}, \boldsymbol{k} \rangle = 0 \,|\, b' = i]$, and:

$$\begin{cases} \Pr[\langle \boldsymbol{a}, \boldsymbol{k} \rangle = 0 \,|\, b = 0] = \frac{1 - E_{0,0}}{1 - E_{0,0} + E_{1,0}}, \\ \Pr[\langle \boldsymbol{a}, \boldsymbol{k} \rangle = 0 \,|\, b = 1] = \frac{E_{0,1}}{1 + E_{0,1} - E_{1,1}}. \end{cases}$$

We can now put in relation those three equations with the ones given by Proposition 3 for the output of a LPN-OD sampler (Algorithm 12):

$$\begin{cases} \frac{1 - E_{0,0}}{1 - E_{0,0} + E_{1,0}} = \frac{1 - \epsilon'_0}{1 - \epsilon'_0 + \epsilon'_1} \\ \frac{E_{0,1}}{1 + E_{0,1} - E_{1,1}} = \frac{\epsilon'_0}{1 + \epsilon'_0 - \epsilon'_1} \end{cases} \iff \begin{cases} \epsilon'_0 = \frac{2\,(\epsilon_0(2\,t_1 - 1) - t_1)t_0 - \epsilon_0(2\,t_1 - 1) + t_1}{1 - t_0 - t_1}, \\ \epsilon'_1 = \frac{(2\,\epsilon_1(2\,t_1 - 1) - 2\,t_1 + 1)t_0 - \epsilon_1(2\,t_1 - 1)}{1 - t_0 - t_1}. \end{cases}$$

The equivalence allows us to conclude that the output of $\text{AltSampleLPN-OD}^n_{\epsilon_0,\epsilon_1,t_0,t_1,m}(\boldsymbol{k})$ follows $D^{\epsilon'_0,\epsilon'_1}_{\boldsymbol{k}}$, the same distribution as $\text{SampleLPN-OD}^n_{\epsilon'_0,\epsilon'_1}(\boldsymbol{k})$'s output. □

Remark. Using the lemma, one can either:

- Use a LPN sampler to simulate an asymmetric LPN-OD one.
- Use a LPN-OD sampler to simulate a symmetric LPN one.

Theorem 1. *The LPN problem is polynomially equivalent to the LPN-OD problem. More precisely LPN^n_ϵ is exactly $\text{LPN-OD}^n_{\epsilon,\epsilon}$, and there is a polynomial reduction from $\text{LPN-OD}^n_{\epsilon_0,\epsilon_1}$ to $\text{LPN-OD}^n_{\epsilon'_0,\epsilon'_1}$ for all $\epsilon_0, \epsilon_1, \epsilon'_0, \epsilon'_1$ verifying the equations given by Lemma 3.*

Proof. From Lemma 3, Algorithm 13 allows to simulate $D^{\epsilon'_0,\epsilon'_1}_{\boldsymbol{k}}$ from $D^{\epsilon_0,\epsilon_1}_{\boldsymbol{k}}$ with the appropriate parameters. Hence, we must show that, with m polynomial in n, Algorithm 13 is a probabilistic polynomial time algorithm, and it returns $\perp$ with negligible probability.

First, note that Lemma 3 requires $\epsilon'_0, \epsilon'_1 \in \left[0, \frac{1}{2}\right]$. Hence, the probability of returning $\perp_0$ is 0. Then, since the number of calls to SampleLPN is bounded by m and the sampler Algorithm 12 runs in constant time, for m polynomial in n the execution time is polynomial in n. The algorithm returns $\perp$ only if the samples given by the sampler have the wrong value of b for m consecutive trials, hence we can study this probability using a binomial law. The probability of returning $\perp$ is $\Pr[b' = 1] \cdot \Pr[b = 0 \mid b' = 1]^m + \Pr[b' = 0] \cdot \Pr[b = 1 \mid b' = 0]^m$. Let $i \in \{0, 1\}$, we show that the probabilities $\Pr[b' = i]$ and $\Pr[b = i \mid b' = 1 - i]$ always belong to $[\frac{1}{4}, \frac{3}{4}]$. The value of b' is given by a Bernouilli of parameter $\frac{1}{2} + \frac{\epsilon'_0 - \epsilon'_1}{2}$ where $\epsilon'_0, \epsilon'_1 \in [0, \frac{1}{2}]$. Based on the analysis in the proof of Lemma 3, $\Pr[b = i \mid b' = 1 - i]$ is given by a Bernouilli of parameter $\frac{1}{2} + \frac{E_{0,b'} - E_{1,b'}}{2}$, and we show that $E_{0,b'}, E_{1,b'}$ are also in $[0, \frac{1}{2}]$. These quantities are obtained as $x + y - 2xy$ where $x, y \in [0, \frac{1}{2}]$. Fixing y, we define the function $f_y(x)$ on $[0, \frac{1}{2}]$ as $f_y(x) = x(1 - 2y) + y$, its derivative $(1 - 2y)$ is positive for $y \neq \frac{1}{2}$ and null otherwise. Hence the maximum of $f_y(x)$ is reached in $\frac{1}{2}$, and the minimum in 0. It allows concluding that $0 \leq x + y - 2xy \leq \frac{1}{2}$. Finally, the probability of failure is at most $2\left(\frac{3}{4}\right)^{m+1}$, therefore negligible in n. □

Concrete security estimation

Theorem 1 gives a general self-reduction of the LPN-OD problem, encompassing the equivalence between LPN-OD and LPN. But in order

to quantify the security we can expect from the real samples we obtain from a LPPN processor, we only need the reduction from LPN-OD to LPN. For this purpose, we simply consider that these samples are the ones of a LPN-OD problem (Definition 15) and we determine an instance of a classical LPN problem which can be converted to it. Since we can assume that an adversary recovering the key from LPPN samples could break $\mathrm{LPN}_\epsilon^n(\boldsymbol{k})$, we can estimate the security of the instances of a LPPN processor with the complexity of the best known attacks on LPN with a noise parameter ϵ (assuming the polynomial reduction has a negligible cost compared to this complexity). This (LPN-OD to LPN) reduction is a particular case of Theorem 1. We state it in the following corollary. This result shows that for noise parameters of LPN-OD centered around the same value, the less these two parameters differ, the higher is the parameter of the LPN problem it reduces to.

Corollary 1. *For any $n, m > 1$, non-zero $\boldsymbol{k} \in \mathbb{F}_2^n$, ϵ' and Δ error parameters such that $0 \leq \epsilon' - \Delta \leq \epsilon' + \Delta \leq \frac{1}{2}$, if an adversary is able to retrieve the key $\boldsymbol{k}$ given m samples from LPN-OD$^n_{\epsilon'-\Delta,\epsilon'+\Delta}(\boldsymbol{k})$ then a polynomially equivalent adversary can retrieve the key from LPN$^n_\epsilon(\boldsymbol{k})$ using m samples, where $\epsilon = \frac{\epsilon'-\Delta}{1-2\Delta}$.*

Proof. It corresponds to the particular case where the adversary uses the polynomial reduction from LPN-OD$^n_{\epsilon_0,\epsilon_1}$ to LPN-OD$^n_{\epsilon'_0,\epsilon'_1}$ with $\epsilon_0 = \epsilon_1 = \frac{\epsilon'-\Delta}{1-2\Delta}$, $\epsilon'_0 = \epsilon' - \Delta$ and $\epsilon'_1 = \epsilon' + \Delta$. We check if these values of $\epsilon_0, \epsilon_1, \epsilon'_0$ and ϵ'_1 verify the equations of Lemma 3. The relations $0 \leq \epsilon' - \Delta \leq \epsilon' + \Delta \leq \frac{1}{2}$ give $\epsilon' \in [0, \frac{1}{2}]$ and $\Delta \in [0, \frac{1}{2}[$, hence $\epsilon_0, \epsilon_1 \in [0, \frac{1}{2}]$. Then, taking $t_0 = \frac{2\Delta}{2\Delta+1}$ and $t_1 = 0$ we get $t_0, t_1 \in [0, \frac{1}{2}]$, $t_0 + t_1 \neq 1$ and:

$$\begin{cases} \epsilon'_0 = \frac{2\,(\epsilon_0(2\,t_1-1)-t_1)t_0-\epsilon_0(2\,t_1-1)+t_1}{1-t_0-t_1} = \epsilon' - \Delta, \\ \epsilon'_1 = \frac{(2\,\epsilon_1(2\,t_1-1)-2\,t_1+1)t_0-\epsilon_1(2\,t_1-1)}{1-t_0-t_1} = \epsilon' + \Delta, \end{cases}$$

$\epsilon' \pm \Delta \in [0, \frac{1}{2}]$ hence the 4 parameters verify the equations. In conclusion, Theorem 1 gives a polynomial reduction from $\mathrm{LPN}_\epsilon^n(\boldsymbol{k})$ to LPN-OD$^n_{\epsilon'-\Delta,\epsilon'+\Delta}(\boldsymbol{k})$. Since the reduction conserves the number of samples, m, it gives the final result: an adversary breaking LPN-OD$^n_{\epsilon'-\Delta,\epsilon'+\Delta}(\boldsymbol{k})$ with m samples implies the existence of a (polynomially equivalent) adversary breaking $\mathrm{LPN}_\epsilon^n(\boldsymbol{k})$ with m samples. □

Remark. By ensuring a smaller output unbalancing parameter Δ (e.g., thanks to implementation tweaks), we manage to reduce the LPN-OD problem to a stronger LPN problem (with higher parameter). Indeed, for all $\epsilon' \in \left[0, \frac{1}{2}\right]$, we can introduce the function $f_{\epsilon'}$ defined for $\Delta \in \left[0, \frac{1}{2}\right[$

as $f_{\epsilon'}(\Delta) = \frac{\epsilon'-\Delta}{1-2\Delta}$. This function returns the parameters of the LPN problem our LPN-OD instance reduces to. Its derivative is such that $f'_{\epsilon'}(\Delta) = -\frac{1-2\epsilon'}{(1-2\Delta)^2} \leq 0$, which means that f increases when Δ decreases. Accordingly, the smaller the Δ, the bigger the ϵ, and the more secure the corresponding LPN problem is. Note that a similar corollary applies for $\text{LPN-OD}^n_{\epsilon'+\Delta,\epsilon'-\Delta}$ and LPN^n_ϵ, by inverting the roles of t_0 and t_1 in the proof.

We conclude by evaluating the security that a LPPN processor concretely provides.

Assuming the LPPN processor provides LPN-OD samples with error parameters $0.25 \pm \Delta$ and $0.25 \mp \Delta$, Corollary 1 shows that retrieving the key from these samples is at least as hard as solving LPN with parameter ϵ where $\epsilon = \frac{0.25-\Delta}{1-2\Delta}$. We therefore base the security of $\text{LPN-OD}^{512}_{0.25\pm\Delta,0.25\mp\Delta}(\boldsymbol{k})$ on the complexity of the best known attacks on $\text{LPN}^{512}_{\frac{0.25-\Delta}{1-2\Delta}}(\boldsymbol{k})$. Indeed, breaking our instances with a lower complexity would imply better attacks on the well established LPN problem. Concretely, our LPPN processor relies on the security of $\text{LPN}^{512}_{0.217}(\boldsymbol{k})$ and on $\text{LPN}^{512}_{0.200}(\boldsymbol{k})$ for $\Delta = 8.2\%$ (these value being the one empirically exhibited in the published version of this work). For such parameters, we estimate a security of 80 bits. According to the work [BTV16], LF2 is the best attack against LPN instances with $n = 512$ and $\epsilon \approx 0.25$. It gives a theoretical time complexity of 99 bits (for a query complexity of 88 bits). To find these numbers, we used the formulas and methods of [BTV16]. To conduct LF2, an adversary can fix two parameters a and b such that $a \cdot b > n$, where n is the size of our key (512). The best attacks were reached for $a = 6$ and $b = 86$. We take a safety margin over the theoretical security of 99 bits to cover possible gaps between theoretical and practical results for LF2.

4.1.3 Masked LPPN

The previous section showed that the LPN-OD problem is hard. Yet, it also suggests that its hardness increases if designers of LPPN-based implementations are able to limit the output data dependencies of the error probability. In this section, we briefly discuss how much masking an ϵ-PIP can help to mitigate such data dependencies. Admittedly, the high-level idea behind our observations was already given in [Kam+20b], and we only specialize it to LPN-OD. In view of the general difficulty to model physical leakages, our following discussion is not based on security reductions but on attack-based arguments. We leave their formalization

as an open problem and use it to argue that masking can significantly mitigate "filtering attacks" exploiting output-dependent errors.

Our starting observation for this purpose is the following one. Imagine that an ϵ-PIP is implemented in a masked manner. That is, rather than computing $y = \epsilon\text{-PIP}(\boldsymbol{x}, \boldsymbol{k})$ in an incorrect manner, one would compute $y = \epsilon\text{-PIP}(\boldsymbol{x}, \boldsymbol{k}_1) + \langle \boldsymbol{x}, \boldsymbol{k_2} \rangle + \ldots + \langle \boldsymbol{x}, \boldsymbol{k}_d \rangle$, where $d-1$ shares $\boldsymbol{k}_1, \boldsymbol{k}_2, \ldots, \boldsymbol{k}_{d-1}$ are picked up uniformly at random and $\boldsymbol{k} = \boldsymbol{k}_1 + \boldsymbol{k}_2 + \ldots + \boldsymbol{k}_d$. In this case, an adversary willing to filter the (incorrect) outputs in order to take advantage of the data-dependent errors would not have access to the output of the ϵ-PIP causing the errors, let's denote it as $z = \epsilon\text{-PIP}(\boldsymbol{x}, \boldsymbol{k}_1)$, but only to its leakage, denoted as $\boldsymbol{l}$.

Besides the fact that such a masking improves security against Differential Power Analysis (DPA), as discussed in [Kam+20a], it also reduces the Δ of the corresponding LPN-OD problem as follows. First, assume that the incorrect computation of z indeed suffers from output data dependencies of the error probability such that $\Pr[e = 1 | \langle \boldsymbol{x}, \boldsymbol{k}_1 \rangle = 0] = \epsilon_0$ and $\Pr[e = 1 \,|\, \langle \boldsymbol{x}, \boldsymbol{k}_1 \rangle = 1] = \epsilon_1$. Using the notations of Section 4.1.1, it means that $\epsilon_0 = \epsilon + \Delta$ and $\epsilon_1 = \epsilon - \Delta$ so that $|\epsilon_0 - \epsilon_1| = 2\Delta$. Then assume that the leakage does only allow recovering z with a certain probability given by $\Pr[z = 0 \,|\, \boldsymbol{l}] = \frac{1}{2} + \delta$ and $\Pr[z = 1 \,|\, \boldsymbol{l}] = \frac{1}{2} + \delta$.[3]

As a result, we directly have that the actual imbalance between the probabilities of error that the adversary will be able to exploit to filter is given by:

$$\epsilon'_0 = \epsilon_0 \cdot \left(\frac{1}{2} + \delta\right) + \epsilon_1 \cdot \left(\frac{1}{2} - \delta\right),$$
$$\epsilon'_1 = \epsilon_1 \cdot \left(\frac{1}{2} + \delta\right) + \epsilon_0 \cdot \left(\frac{1}{2} - \delta\right).$$

If $\delta = \frac{1}{2}$, then the recovery of z is perfect so that we have $\epsilon'_0 = \epsilon_0$ and $\epsilon'_1 = \epsilon_1$. If $\delta = 0$, then there is no leakage so that we have $\epsilon'_0 = \epsilon'_1 = \epsilon$. So the latter shows that in case an implementation is masked and the adversary does not exploit the leakage, then exploiting the output data dependencies with a filtering attack is not possible.

Quite naturally, the situation when an adversary exploits the leakage will fall in between (i.e., actual δ values will be between 0 and $\frac{1}{2}$). But it is important to see that the secret z to recover is an ephemeral one, so this adversary will be limited to Simple Power Analysis (SPA). Concrete

[3] As usual in side-channel analysis, we assume a leakage function with additive and data-independent noise so that the complexity of recovering z is the same whether it is zero or one.

values are discussed in the FPGA section of the published version of this work. Note that the proposal in this section is aimed at hiding the output error dependencies, since it needs to be addressed for current prototypes of LPPN processors. But in case input dependencies become critical, the same idea can be used to mask $\boldsymbol{x}$. Note also that the proposal in this section induces the errors on a single share, which is interesting since it maintains the level of control needed on the error probability equal to the one of an unprotected implementation. If the error was spread over multiple shares, its probability would be lower on each share (as per the piling up lemma), leading to a more challenging calibration of the error probability.[4]

4.2 Generalizing inexact computing to higher dimension, adapted from [Bel+22]

The Learning With Errors (LWE) problem is an important computational assumption which is the basis of many modern cryptographic constructions [Reg09a]. It has in particular been used to design several Fully Homomorphic Encryption (FHE) schemes [BV11a; BGV12; GSW13] and is the core of many Post-Quantum (PQ) algorithms such as Frodo [Alk+21], New Hope [Alk+20], CRYSTALS-Kyber [Ava+20] or CRYSTALS-Dilithium [Duc+21]. In a standard implementation of LWE-based cryptographic primitives, one typically generates errors following a given distribution explicitly, which are then added to some correct computations. Yet, it has been shown that the cost of generating the random errors with a TRNG or with cryptographically strong PRNG can become a performance bottleneck, especially if side-channel attacks are a concern [Sch+19; Bos+21].

In parallel, a line of works initiated by Kamel et al. has shown that leveraging inexact computing in order to directly compute erroneous computations following the appropriate distribution can lead to conceptual advantages for the implementation of primitives based on the Learning Parity with Noise (LPN) problem [Kam+20b]. The LPN problem is a special case of the LWE problem where computations are performed in GF(2). Its physical variant has been formalized as the Learning Parity with Physical Noise (LPPN) problem. It has been shown that it is possible to control the error distribution of LPPN prototypes with a

[4] The side-channel based filtering attacks discussed in this section, which target the leakage of an ϵ-PIP's (unbalanced) incorrect output, are different from side-channel attacks targeting the correct output of LPN implementations, formalized in [Dzi+16] with the Learning with Leakage (LPL) problem.

sufficient accuracy [Kam+18] and that the resulting (key-homomorphic) implementations have interesting features for side-channel security via masking [Kam+20a].

More recently, Bellizia et al. investigated the (lack of) gap between the (mathematical) LPN problem and the (physical) LPPN problem [Bel+21]. They first showed that LPPN implementations can suffer from output data dependencies, where the probability of the errors depends on the correct computations (while they are independent in LPN). They next showed that these dependencies can be reduced thanks to implementation tweaks, despite not being able to make them negligible. As a result, they finally gave a reduction from the standard LPN problem to a (new) LPN-OD problem with such output dependencies, allowing one to quantity the limited security loss that such dependencies imply.

In terms of applicability, the main open question raised by these different previous works is whether generalized hard physical learning problems could be instantiated for use in actual applications. Our contributions in this respect are threefold.

First, we define such a generalization, next coined Learning With Physical Errors (LWPE), and discuss the new implementation challenges it raises. In particular, we show that directly applying the techniques used in the LPPN context leads to a new type of (mathematical) dependencies in the error distributions (hence the bad news part of our title). In short, these dependencies are due to the fact that the errors introduced affect bits rather than field elements. Yet, we also show that it is possible to get rid of them by inserting the errors early in the inner product computations of a LWPE prototype.

Second, we discuss the LWPE design space and the extent to which LWPE could be used for different lattice structures, multiplication algorithms, modulus, and error distributions. We conclude that hard physical learning problems formalized by LWPE have a broad applicability while also being more challenging in certain specific contexts.

Cautionary note. Leveraging the LWPE assumption is only possible for PQ schemes that do not require deterministic errors in their decryption which, for example, is not the case of Key Encapsulation Mechanisms (KEMs) based on the Fujisaki-Okamoto (FO) transform [FO99]. Yet, it is applicable in any context where LWE is used without such an additional requirement (e.g., IND-CPA public key encryption or signature schemes that do not need to be de-randomized). Besides, given that the FO transform is the root of important weaknesses against side-channel attacks [Uen+22] while LWPE actually gains interest when side-channel

attacks are a concern, our results give further incentives for designing new PQ cryptographic schemes that do not rely on such transforms.

4.2.1 Background

In this section, we provide the necessary background on the hard learning problems needed in this work and introduce statistical tests that we will need for our experimental investigations.

The LWPE problem

Bellizia et al. introduced LPPN as a variant of the LPN problem where a physical function directly computes incorrect inner products. Analogously, we introduce the LWPE problem, which relies on a physical function (Definition 8) *pd*. *pd* will be an implementation of an inexact inner product computations modulo q (implying $n_i = n$ represented in $\lceil log_2(q) \rceil$ bits and $n_o = 1$ represented in $\lceil log_2(q) \rceil$ bits) and α is the set of parameters determining its error distribution.

Definition 16 (Physical inner product)**.** *Let $\boldsymbol{s} \in \mathbb{Z}_q^n$ be a uniform random secret stored in a keyed device $pd_{\boldsymbol{s}}$. A physical function $\mathsf{PF}_{pd_{\boldsymbol{s}},\alpha}$ is called a χ-Physical Inner Product (χ-PIP) if, on uniform public input $\boldsymbol{a} \in \mathbb{Z}_q^n$, it outputs $\langle \boldsymbol{a}, \boldsymbol{s} \rangle$ with an additional error following a distribution χ such that:*

$$\forall i \in \mathbb{Z}_q, \hat{\Pr}[\mathsf{PF}_{pd_{\boldsymbol{s}},\alpha}(\boldsymbol{a}) - \langle \boldsymbol{a}, \boldsymbol{s} \rangle = i] = Pr[X = i] \text{ where } X \xleftarrow{\$} \chi.$$

The LWPE problem can then be specified as follows:

Definition 17 (LWPE problem)**.** *Let $\mathsf{PF}_{pd_{\boldsymbol{s}},\alpha}$ be a χ-PIP. The $LWPE^{n,\chi}_{pd_{\boldsymbol{s}},\alpha}$ problem is said to be (c, t, m, θ)-hard to solve if for any algorithm A running in time $< t$, memory $< m$ and making at most c uniformly random queries to a χ-PIP, it holds that:*

$$\Pr[\boldsymbol{s} \xleftarrow{\$} \mathbb{Z}_q^n : A^{\mathsf{PF}_{pd_{\boldsymbol{s}},\alpha}}(1^n) = \boldsymbol{s}] \leq \theta.$$

Noise distributions

The noise distribution χ of the LWE/MLWE error/secret are sampled either from a discrete Gaussian distribution DGD_σ or from a centered binomial distribution CBD_η.

Discrete Gaussian distribution. The discrete Gaussian distribution DGD_σ parametrized by the Gaussian parameter $\sigma \in \mathbb{R}^+$ is defined by assigning a weight proportional to $exp(\frac{-x^2}{2\sigma^2})$ to all integers x. Note that

it can be challenging to implement a discrete Gaussian sampler efficiently and protected against timing attacks [Bos+15; Alk+16b].

Centered binomial distribution The binomial distribution with parameters n and p is the discrete probability distribution of the number of successes in a sequence of n independent experiments with individual success of probability p. Its probability mass function is defined as $\binom{n}{x}p^x(1-p)^{n-x}$ with mean $= np$ and variance $= np(1-p)$. The centered binomial distribution is obtained by subtracting the mean from this distribution. We next focus on a sub-family that has the particularity to give symmetric distributions. For η a non negative integer we denote CBD_η the centered binomial distribution with parameters $p = 1/2$ and $n = 2\eta$. CBD_η has mean 0 and variance equal to $\eta/2$, and its samples can be computed as $\Sigma_{i=0}^{\eta-1} a_i - b_i$ where $a_i, b_i \in \{0, 1\}$ are uniform independent bits. Therefore, sampling from the centered binomial distribution is easy and does not require high-precision computations or large tables like Gaussian sampling does.

Statistical tests

In this paper, we will need statistical tests to detect potential data dependencies in the LWPE samples distributions due to physical imperfections. In general, such tests use probability theory to reject (or not) a hypothesis, commonly called as the null hypothesis, thanks to a set of samples. Therefore, statistical tests can produce two types of errors:

- Type I errors (or false positives, or alpha risk). These errors capture the risk of rejecting the null hypothesis when it is true. Most statistical tests (and the ones we will use) quantify the likelihood of such errors and name it p-value.

- Type II errors (or false negatives, or beta risk). These errors capture the risk of not rejecting the null hypothesis when it is false. They cannot be quantified in general. However, the bigger the set of samples used, the lower this error becomes.

Concretely, we used the Kruskal-Wallis test (or one-way ANOVA on ranks), which is a nonparametric test used for testing if independent samples originate from the same distribution [KW52]. It works under the null hypothesis that the empirical distributions are identical, which should be the case if there are no data-dependent errors.

4.2.2 Introducing LWPE with a design example

We now introduce the main design ideas needed to instantiate practically-relevant LWPE instances. For this purpose, and for the sake of simplicity, we will first focus on a small LWE modulus ($q = 2^8$) and a small lattice dimension $n = 128$. Based on these parameters, we will describe how to implement an LWPE processor whose errors' distributions approximate a centered binomial distribution (CBD_η) with parameter $\eta \in \{2, 3\}$. We investigate two approaches for this purpose, and provide rationale about the resulting error distribution and its security. In **Design A**, we directly extend the LPPN prototype in [Kam+18] and sample the two Least Significant Bits (LSBs) of the LWPE processor's output incorrectly, so that the combined errors approximately follow a CBD_3. Yet, we show that such an approach is not successful as it leads to (mathematical) data dependencies in the error distributions. Instead of sampling at the final stage just before outputting the result, **Design B** then generates the errors by sampling the LSB of η bytes at a designated internal stage. Interestingly, we will show that this approach cancels the previously mentioned mathematical data dependencies. Both designs are implemented in a 65-nm technology using a fully digital design approach. Pre-layout simulations are performed using ULTRASIM within the CADENCE environment. We used 2^{16} uniformly distributed random 128-byte inputs $\boldsymbol{a}$ and a single uniformly distributed random 128-byte secret key $\boldsymbol{s}$. The supply voltage is fixed at nominal 1.2V and the clock frequency is 500MHz.

Note that, as will be detailed in Section 4.2.2, the shape of the noise distribution is not a critical security requirement. So the selection of a CBD in our next designs is just used as an illustrative example that can be found in PQ cryptographic schemes.

Design A

We first present the architecture of the LWPE_A processor and demonstrate the feasibility of generating samples following an approximated CBD_η ($\eta = 3$) distribution.

LWPE_A implementation. Our first approach to implement a χ-PIP is illustrated in Figure 4.3. The inner product block takes 128-byte vectors of the secret $\boldsymbol{s}$ and the challenge $\boldsymbol{a}$ as inputs, and outputs a single byte y. Two specific sampling clocks *clkD0, clkD1* are generated by two variable delay lines (VDL0 and VDL1) such that the error probability of the sampled output bits provides a good approximation of the target error distribution.

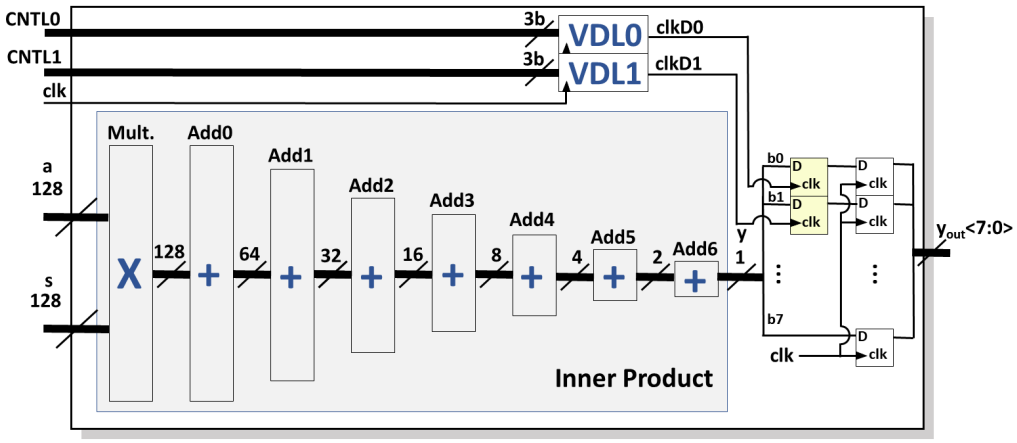


Figure 4.3: Architecture of the LWPE_A processor.

The inner product block comprises 128 parallel 8-bit multipliers used to process the byte vectors $\boldsymbol{a}$ and $\boldsymbol{s}$. The sum of these products is implemented with a seven-stage tree of multiple-size parallel adders. Two bits (b_0 and b_1) of the inner product block output byte y are sampled by two flip-flops clocked thanks to delayed versions of the system clock, namely $clkD0$ and $clkD1$. These flip-flops are responsible for sampling the designated output bits during their glitchy period before they stabilize. The output of these two flip-flops along with the remaining bits of the inner product output y ($\langle b_7 : b_2 \rangle$) are sampled with the system clock clk to synchronize and produce the LWPE_A output y_{out}.

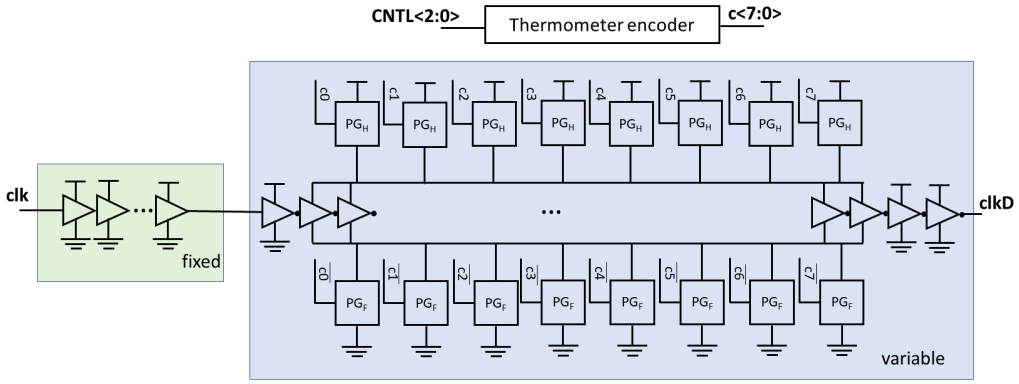


Figure 4.4: Block diagram of the Variable Delay Line (VDL).

To ensure that the generated errors follow a CBD_η distribution, the error probability on each bit needs to be bounded. As in LPPN prototypes, the key aspect to guarantee such bounds is to design a variable delay line and a control circuitry per bit, such that they can set the delay of the sampling clocks $clkD0$ and $clkD1$ to capture the non-stable state of the designated bits. Concretely, this typically works in two phases: calibration and generation [Kam+18]. Since the calibration we use is identical to the LPPN case, we do not describe it nor implement it for simplicity. Instead, the control signals $CNTL0$ and $CNTL1$ (which are 3-bits each) are entered as inputs to the LWPE_A module. Each variable delay line consists of two parts: a fixed delay to compensate for the computation of the inner product until the final adder stage, and a variable delay to sample incorrectly the relevant output bit. Figure 4.4 demonstrates the block diagram of the VDL. The fixed delay part uses multiple delay cells provided by the standard cell library of the foundry. The implementation of the variable delay part is based on the current starved inverter concept [MNS05]. It comprises a chain of inverters whose supply and drain currents are digitally controlled by a power gate header and footer cells (also provided by the standard cell library of the foundry). The number of stages in the variable delay block is adjusted such that both the tuning range and the delay steps provide reliable control over the required probabilities of error. Finally, an encoder is needed to convert the binary $CNTL\langle 2:0\rangle$ signals into its corresponding thermometer encoded $c\langle 7:0\rangle$ signal to control how much power gate cells are activated, and thus the delay.

In order for the error distribution of the proposed LWPE_A processor to approximate a CBD_η (with $\eta = 3$ in our example), we explored all the error probability combinations on b_0 and b_1 of the LWPE_A processor output, and computed the statistical distance ($\Delta = \Sigma_{j=-\eta}^{\eta} |CBD_\eta(j) - \chi(j)|$) between the CBD_3 distribution and the one of the LWPE_A error, with χ taken over multiple samples by computing the error $e = y - \langle \boldsymbol{a}.\boldsymbol{s} \rangle$. After mathematically searching all such possible combinations using a Python script, the minimum statistical distance (0.128) was achieved by setting the error probability on b_0 to 0.5 and on b_1 to 0.37. Figure 4.5 shows the resulting error distribution of the LWPE_A processor compared to the required CBD_3, highlighting a first limitation: the errors generated with the best fitted parameters do not exactly follow a CBD_3 distribution.[5]

[5] Another limitation of this approach is the difficulty to approximate a CBD_η distribution with $\eta \neq 2^l - 1$. With $\eta = 3$, manipulating the last two LSBs of the LWPE output directly supports an error range $\{-3, \ldots, +3\}$. However, if $\eta = 2$, the required support range of the error distribution is $\{-2, \ldots, +2\}$ and to achieve

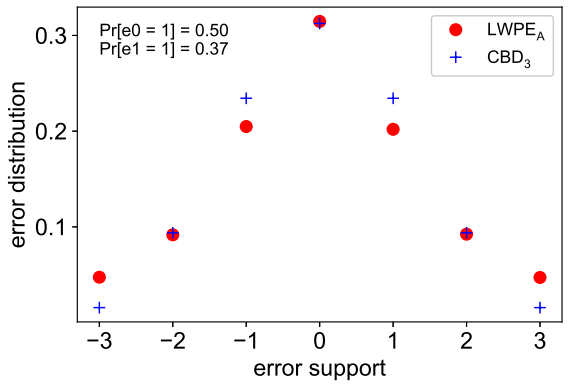


Figure 4.5: LWPE_A: Error distribution approximating CBD_3.

Mathematical output data-dependencies of the errors. As previously mentioned, exactly approximating a CBD is not a critical security requirement for LWPE. By contrast, a more sensitive implementation challenge when instantiating hard physical learning problems is to limit the data dependencies of their error distributions. For example, it has already been put forward in [Bel+21] that the errors of an ϵ-PIP generating LPPN samples can exhibit significant output data dependencies due to physical imperfections. In this section, we highlight that the previous generalization of an LPPN design towards an LWPE one leads to another source of strong (mathematical) output data dependencies that neither LWE samples nor LPPN samples exhibit.

In summary, since the error is not added (in the sense that it is sampled from a distinct CBD_η distribution) as in a standard LWE sample $t = \langle \boldsymbol{a}.\boldsymbol{s} \rangle + e$, but generated by sampling incorrectly the last two LSBs of the LWPE_A output, this error does not propagate to higher-order bits. In fact, the incorrect sampling of the LSBs corresponds to an XOR operation between the output and a binary vector equivalent to 0, 1, 2, or 3 (since two bits of noise are considered in our example). As a result, the error range is not equal across different values of the correct LWPE output. This means that the error of an LWPE processor will depend on the correct value of the last two LSBs of the output. We plot in Figure 4.6 the error ranges as well as the output of both LWE and LWPE_A samples for our example where $q = 16$ and $\eta = 3$, where the mathematical dependencies are clear.

this we need to reduce the probability of occurrence of ± 3 errors to a negligible level.

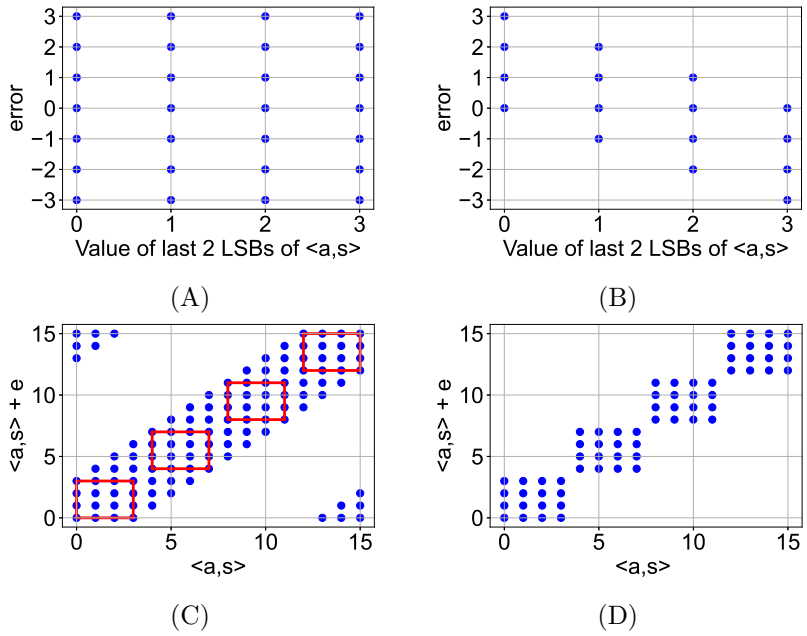


Figure 4.6: Errors of (a) LWE and (b) LWPE_A versus the correct value of the last two LSBs of $\langle a, s\rangle$, and incorrect output of (c) LWE and (d) LWPE_A versus the correct output for an example where $q = 16$ and $\eta = 3$. The LWPE output values obtained in (d) are superimposed on the ones of the LWE in (c) as red rectangles.

We note that the errors of the ϵ-PIP generating LPPN samples did not feature such mathematical output dependencies in [Bel+21] (only physical output dependencies were observed). The reason is twofold. First, these errors were generated by sampling a single bit (i.e., the output). Second, these errors (almost) followed a Bernoulli distribution that does not take into consideration whether the actual value of the error was $+1$ or -1.

We conclude that generalizing LPPN prototypes into LWPE ones to approximate practically-relevant noise distributions (e.g., for PQ applications) using this approach is not ideal. First, the error distribution of an LWPE_A instance does not approximate well the target (e.g., CBD_3) distribution. More critically, the approximated distributions exhibit mathematical dependencies that may affect the security of the assumption. We next discuss an alternative approach to LWPE prototypes addressing these limitations.

Design B

To mitigate the previous mathematical error dependencies, we next propose to generate the errors by sampling the LWPE processor incorrectly at an internal stage rather than at its output. We first discuss the implementation details of the resulting LWPE_B processor followed by simulation results. Then, we explain why this approach allows canceling the mathematical output data dependencies of the error distributions.

LWPE_B implementation

Figure 4.7 illustrates the proposed architecture of the LWPE_B processor. It is quite similar to the LWPE_A design where the inner product block takes 128-byte vectors of the secret $\boldsymbol{s}$ and the challenge $\boldsymbol{a}$, multiplies them (using 128 parallel 8-bit multipliers), and then adds them two by two in a seven stage adder tree to output a single byte y. However, there are two main differences between the two approaches. The first one is that the flip-flops responsible for the incorrect sampling are placed after an internal adder stage (Add3 in this case) instead of at the output of the LWPE_A processor. The second difference is that the errors are generated from sampling the LSB of η bytes ($\eta = 2$ in the current example of Figure 4.7) rather than multiple LSBs of a single byte (as in the LWPE_A processor). Indeed it is easier to ensure that the generated errors of the LWPE_B processor follow a CBD_η distribution by sampling only the LSB of multiple bytes (b_0 and not higher-order bits), in which case the error probability needs to be equal to 0.5 for all the erroneous bits.

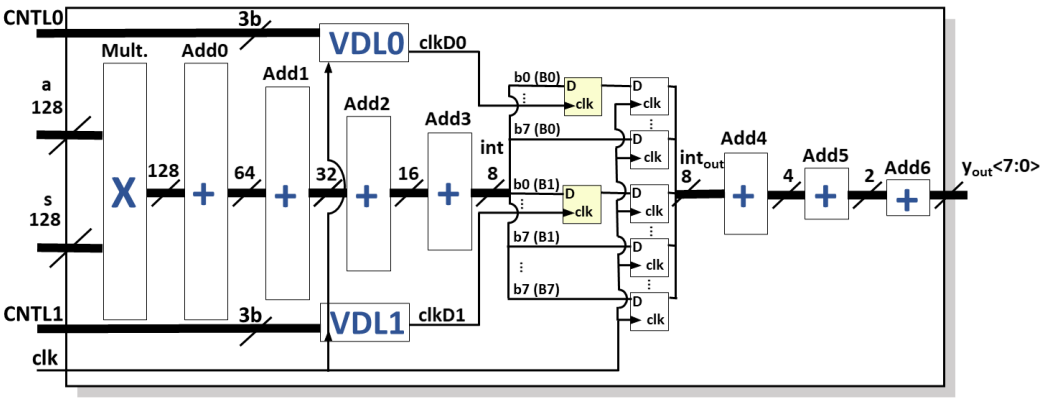
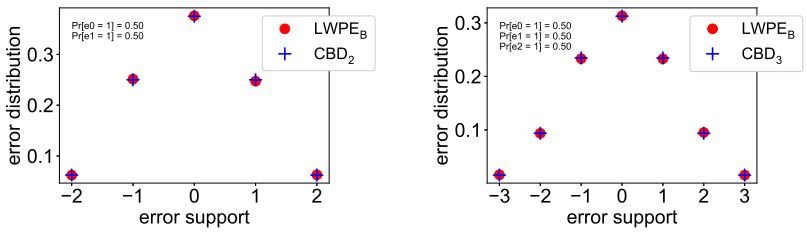


Figure 4.7: Architecture of the LWPE_B processor.

Next, the output of the incorrect sampling flip-flops along with the remaining bits of the concerned stage is sampled with the system clock *clk* to synchronize them before being passed to the subsequent adder

stages to produce the final LWPE_B output y_{out}. The incorrect sampling flip-flops are clocked by delayed versions of the system clock. Similar to the LWPE_A processor, the current approach uses variable delay lines with control signals to adjust the delay of the sampling clocks. Figure 4.8 shows the error distribution of the LWPE_B processor compared to the required CBD_2 and CBD_3 distributions. Clearly, the current design of the LWPE_B processor leads to a much tighter fit of the output error distribution compared to that of the previously described LWPE_A processor.[6]



(A) (B)

Figure 4.8: LWPE_B: Error distribution approximating (a) CBD_2 and (b) CBD_3.

We note that sampling incorrectly at the output of other internal stages is possible. However, opting for a later stage in the adder chain is in general a good approach since it reduces the need to add registers (and therefore decreases the area and power consumption).

Mathematical output data-dependencies of the errors. Sampling incorrectly at an internal stage of the LWPE_B processor causes the errors in the sampled output of this specific stage to encounter the same mathematical data dependencies as previously explained in section 4.2.2. However, since this is not the final output and the remaining bytes of the same and subsequent stages are yet to be added, the mathematical data dependencies are expected to be reduced thanks to diffusion and eventually canceled out, which we illustrate with the next example. Let $n = 128$, $q = 256$ and $\eta = 2$. Let also the error be generated by sampling incorrectly the output of the fourth adder (Add3) at b_0 of bytes B_0 and B_1. We show in Figure 4.9 three different scenarios (marked by distinct colored rectangles) in case the correct output is equal to 125 as highlighted in yellow.

[6] The statistical distances with CBD_2 and CBD_3 are 3.5×10^{-3} and 5.1×10^{-3}, respectively.

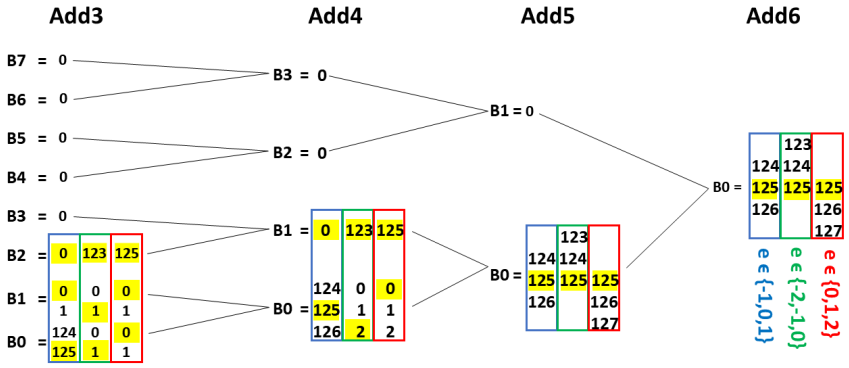


Figure 4.9: Illustration of the reduced output data dependencies in LWPE_B, for an exemplary case where $\eta = 2$ and the errors are introduced at the Add3 stage of the processor.

The error at bytes B_0 and B_1 of Add3 $\in$ either to $\{-1, 0\}$ or to $\{0, 1\}$, depending on the value of their LSB being either 1 or 0, respectively. Therefore, if $B_2 = 0$, the correct values of $B_1 = 0$ and the one of $B_0 = 125$, then the error of the final output $\in \{-1, 0, 1\}$ (marked in blue). Certainly, multiple combinations of byte values at Add3 will lead to the same output (125 in this example). We will consider two cases such that the error at the final output indeed $\in \{-2, \ldots, 2\}$. If $B_2 = 123$, the correct values of $B_1 = 1$ and the one of $B_0 = 1$, then the error of the final output $\in \{-2, 0, 1\}$ (marked in green). Finally, if $B_2 = 125$, the correct values of $B_1 = 0$ and the one of $B_0 = 0$, then the error of the final output $\in \{0, 1, 2\}$ (marked in red). Hence, the error at the final output can take values from the full error support of a CBD_2 distribution (despite it does not for the intermediate stages) and these scenarios may occur with other combinations of byte values. redSimply stated, the adder tree is such that in the final stage, an erroneous byte X is added with an independent and uniformly distributed byte Y to produce the output. So even if the error distribution is dependent on X, it is not dependent on the output $X + Y$. As a result, without side-channel leakage on Y, there are no exploitable data-dependencies in the LWPE_B samples.

We complement this example with the results of Figure 4.10, where the outputs of the LWPE_B processor targeting a CBD_2 are plotted versus the correct value of $(\langle \boldsymbol{a}, \boldsymbol{s} \rangle)$. As can be seen from the inset plots, the

outputs extend across the full error support range of the CBD_2 for all correct values, indicating no mathematical output dependencies.

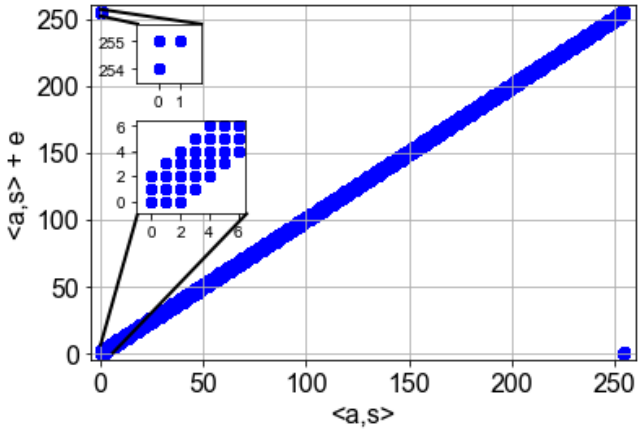


Figure 4.10: LWPE_B: incorrect outputs versus correct outputs in case $\eta = 2$.

Security requirements

Despite a physical inner product may not produce an error that exactly follows a CBD (or any other target distribution), the assumption that "the performance of the best-known attacks against LWE-based encryption does not depend on the exact distribution of noise, but rather on its standard deviation" [Ava+20] is commonly admitted. More precisely, the property that is required for LWE to be hard is that the error distribution is B-bounded [BGV12]. It is assumed that a B-bounded error distribution provides security in both the QROM [Reg09a] and the ROM [Pei09] for rounded Gaussians. The use of other B-bounded distributions, with sufficient standard deviation, does not seem to lead to better attacks [BGV12; DT14]. In the LWPE_B case, the CBD_η distribution is approximated by flipping η LSBs, which guarantees the resulting distribution is B-bounded [BGV12]. Concretely, the empirical standard deviation of the error distribution has been calculated using a prototype hardware implementation of LWPE and can be found in the published version of this work.

4.2.3 LWPE design Space

The previous section put forward the feasibility to design a χ-PIP where χ is a CBD_η distribution, in order to generate useful LWPE samples.

In this section, we further expand our design space and explore the possibility of applying inexact computing to other scenarios. Figure 4.11 summarizes the dimensions of this design space (marked in different color codes). Elements highlighted in green imply high-level compatibility with the proposed concept of inexact computing. We mark in yellow the parts of the design space that may be less efficiently implemented thanks to incorrect sampling. Finally, elements emphasized in orange indicate stronger implementation challenges still to be solved. We next discuss these different dimensions of the design space in more detail.

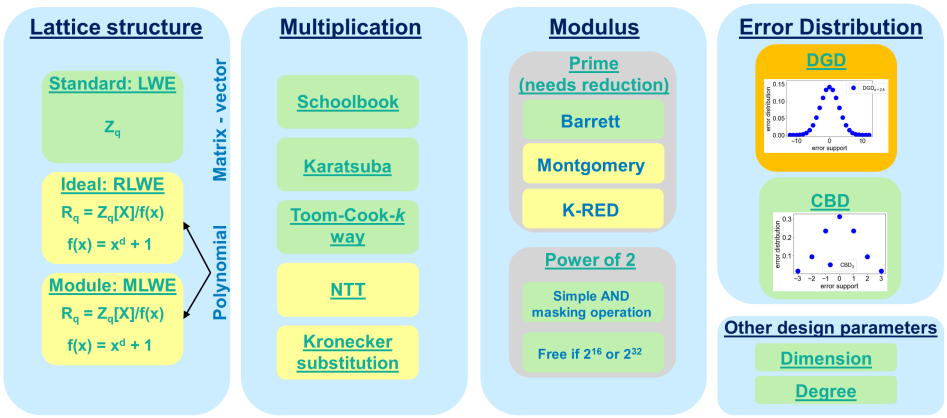


Figure 4.11: LWPE design space.

Lattice structures. LWE-based cryptosystems can rely on different variants: the standard LWE, the ring-LWE or the module-LWE problem. The standard (also called plain) LWE is a hard learning problem relying on algebraically unstructured lattices. Consequently, the main operations are simple matrix-vector products in ring $\mathbb{Z}_q$. On the other hand, algebraically structured lattices such as RLWE and MLWE use a polynomial ring $R_q = \mathbb{Z}_q[x]/(x^d + 1)$, where d is a power of 2, instead of $\mathbb{Z}_q$. In theory, inexact computing can be used in both cases to generate erroneous samples. Yet, the efficient implementation of RLWE or MLWE schemes may require to exploit more advanced algorithms for the polynomial operations, especially the multiplication, as we discuss next.

Multiplication algorithms. Various algorithms can be used to perform the multiplication operations of LWE-based schemes. In case the lattice-based cryptosystem relies on the standard LWE problem, the main arith-

metic operations are simple matrix-vector products. This enables the use of simple Schoolbook multiplication which directly supports the previously described use of inexact computing to generate errors.

On the other hand, for algebraically structured lattices, multiplications take place in R_q. This generally necessitates the use of fast polynomial multiplication algorithms such as the Number Theoretic Transform (NTT) if the cryptosystem parameters are NTT-friendly. In this case, the polynomial coefficients are first converted to the NTT domain before pointwise multiplications take place. The result is then converted out of the NTT domain (i.e., we compute $\mathrm{NTT}^{-1}(\mathrm{NTT}(\boldsymbol{a}) \times \mathrm{NTT}(\boldsymbol{s}))$). In cases where a polynomial matrix is multiplied by a polynomial vector (e.g., in Kyber), further addition operations are done before transforming the final result to polynomial domain (i.e., $\mathrm{NTT}^{-1}(\Sigma_{i=0}^{n-1}\ \mathrm{NTT}(\boldsymbol{a_i}) \times \mathrm{NTT}(\boldsymbol{s}))$).[7] Consequently, error generation via incorrect sampling is possible at one of the (multiple) stages of the multiplication. As previously, one can sample the final output after the inverse NTT operation. However, this approach leads to errors featuring mathematical output dependencies, because the incorrect sampling takes place at the LSB of the output and will not propagate to higher-order bits, as previously discussed in Section 4.2.2. It is also feasible to sample incorrectly at an intermediate stage, between two addition operations (i.e., $\mathrm{NTT}(noisy(\mathrm{NTT}^{-1}(\mathrm{NTT}(\boldsymbol{a_0}) \times \mathrm{NTT}(\boldsymbol{s})))) + \Sigma_{i=1}^{n-1}\ \mathrm{NTT}(\boldsymbol{a_i}) \times \mathrm{NTT}(\boldsymbol{s}))$. In that case, the mathematical output dependencies are expected to be diffused thanks to the subsequent addition operation(s), as previously explained in section 4.2.2.

Other popular multiplication algorithms include Karatsuba [KO63], Toom-Cook-k-way [Too63]and the Kronecker substitution. Unlike the NTT, these multiplication algorithms do not require any restrictions on the cryptosystem modulus: they apply to any prime or power of two modulus. For the rest, the Kronecker substitution leads to similar challenges as the NTT. They both use pointwise multiplication after evaluating the polynomial using predefined points. As a result, it does not directly support the use of inexact computing to generate errors and requires the multiplication result to be converted back to polynomial. Both the Karatsuba and Toom-Cook algorithms are based on a divide-and-conquer strategy. The Karatsuba multiplication splits each of the multiplicand polynomials into two degree-$d/2$ polynomials, whereas the

[7] In Kyber, the result of the polynomial multiplication is kept in NTT domain to further compute the public key and ciphertext. As a result, an extra NTT^{-1} operation is necessary to sample incorrectly in the polynomial domain as well as an extra NTT to go back to NTT domain if necessary.

Toom-Cook-k-way one splits them into k equal parts. Further splitting is necessary to reduce the number of multiplication steps. Both algorithms are compatible with our incorrect sampling techniques to generate errors since the coefficients are multiplied without any transformation.

We note that it is possible to combine algorithms as in [Alb+19], where the authors used Kronecker substitution (with improved techniques [Har09]) together with either Karatsuba or Schoolbook multiplications for implementing Kyber. Another example in [MKV20], where the authors carry out the polynomial multiplications of Saber [D'A+18b] (a public key encryption scheme, NIST Round 3 finalist, that relies on the hardness of the module learning with rounding problem), by combining Toom-Cook and Karatsuba with an evaluation step (similar to the Kronecker substitution) to perform pointwise multiplication. In yet another work, Karatsuba is used with NTT in order to reduce the number of multiplications needed [DMG21; XL21]. Whether generating errors with inexact computing is supported or not by these approaches largely depends on the individual algorithms of such combinations, which we leave as a scope for further investigations.

Modulus. Modular computations used in lattice-based cryptosystems either use a prime or a power of two modulus. Prime moduli are usually chosen to satisfy either $q \equiv 1 \, mod \, d$ (as in Kyber) or $q \equiv 1 \, mod \, 2d$ (as in NewHope), where d is the ring dimension in both cases. This is required to enable the fast NTT-based polynomial multiplication employed in several post-quantum cryptosystems. Accordingly, modular operations necessitate the use of reduction algorithms such as Barrett [Bar86], Montgomery [Mon85] or K-RED [LN16]. The Barrett algorithm optimizes the modular reduction by replacing the expensive divisions (used in classical modular reduction) with multiplications and bit shifts. Concretely, to compute $(a \times b) \, mod \, q$ we need to evaluate $(a \times b) - ((a \times b \times p) \gg l) \times q$, where $p = \lfloor \frac{2^l}{q} \rfloor$, $\gg$ is the right bit shift operation and l needs to be chosen to reduce the difference between $\frac{1}{q}$ and $\frac{p}{2^l}$. Such a reduction technique is compatible with the inexact computing concept without any extra cost. The Montgomery reduction rather avoids expensive division operations by transforming the input multiplicands to a "Montgomery form" (i.e., $\bar{a} = (a \times R) \; mod \; q$ and $\bar{b} = (b \times R) \; mod \; q$), which is basically the residue class of the multiplicands, using a constant $R > q$ that is coprime to q, and is always a power of two. Thus the only division necessary is the division by R which is a simple bit shift. The Montgomery reduction is then applied to evaluate the result of the modular multiplication which is in Montgomery form (i.e., $\bar{a} \times \bar{b} \times R^{-1} \; mod \; q$). In cryptosystems such

as Kyber, several multiplications are performed in a row, so the intermediate results can be left in Montgomery form while only the final output is transformed out of it. Consequently, sampling incorrectly to generate errors is not directly possible in Montgomery form. To solve this issue, we need to convert the product (or a subsequent result of modular operations in the processor implementing a χ-PIP) out of the Montgomery form (using the Montgomery reduction) before the incorrect sampling step. This comes at a slight increase in the overall implementation cost. Eventually, the K-RED algorithm is somewhat similar to Montgomery as it operates on the residue class of the input. However, K-RED outputs a value that is congruent to a multiple of the input. So again, the K-RED algorithm does not support error generation via incorrect sampling unless the output is converted out of the K-RED representation.

We note that a power-of-two modulus facilitates the operations since reduction modulo q can then be computed by low-cost bit masking, or even at no cost if the size of the target platform matches the number of bits of the modulus (e.g., 16-bit or 32-bit). Such moduli facilitate the use of inexact computing to generate errors as well.

Error distribution. Natively, many LWE-based cryptosystems require the generation of errors from the (discrete) Gaussian distribution [Bra+13]. However, various works have shown that it is challenging to implement a discrete Gaussian sampler efficiently [Bos+15], especially if the implementation has to be protected against timing attacks (see for example [PBY17]). Therefore, PQ cryptosystems such as Frodo use a distribution that is very close to it and at the same time easier to implement via pre-computed lookup tables. We posit that a similar conclusion holds for implementations based on inexact computing. Namely, we expect the generation of discrete Gaussian errors to be significantly more challenging, especially due to the tail of the distribution. As a result, the centered binomial distribution (CBD) is gaining popularity, thanks to its easy, efficient, and secure implementation features [Alk+16b]. It is used in both the Kyber and NewHope cryptosystems. From a security viewpoint, this choice is backed up by the argument that the performances of the best attacks against LWE-based encryption schemes do not depend on the exact distribution of the error, but rather on its standard deviation [Ava+20].

Other design parameters. The ring dimension and the polynomial degree are not influencing the possibility to leverage inexact computing to generate erroneous samples.

4.2.4 Conclusions and further research

Hard physical learning problems have been proposed as a provocative idea to circumvent certain implementation challenges in recent cryptosystems based on hard learning problems. First progresses in this direction were focused on the simpler LPN/LPPN case, and their generalization to LWE/LWPE was therefore an important open question (since the LWE assumption is much more versatile than the LPN one). In this paper, we showed that such a generalization is possible, although instances of LWPE require to tweak some of the design ideas used in LPPN prototypes. Namely, LWPE designs highly benefit from the introduction of the errors in an intermediate computation stage rather than at its output, in order to reduce the output data-dependencies of the errors' distributions.

These results lead to interesting scopes for further investigations. First, one could study whether the design tweaks we used for LWPE could be used in LPPN protoypes as well. This may not be direct since the simplicity of the LPPN computations leaves less room (i.e., critical path) to introduce errors in intermediate stages while keeping a sufficient control of the error probability. But combining this idea with longer serial parts than in [Bel+21] could lead to positive outcomes. Second, we showed that the output data-dependencies of our LWPE prototype can be made small. But even if the exploitation of such small dependencies is not a direct security threat, a formal reduction from the standard LWE problem to a LWE-OD problem with output dependencies remains of theoretical interest (and may be motivated by other LWPE designs). Eventually, the understanding of the performances, reliability and side-channel security of hard physical learning problems remains at an early stage. So further challenging these ideas in concrete case studies would be beneficial to better specify the contexts in which they are useful.

4.3 Learning with physical rounding for linear and quadratic leakage functions, adapted from [Hof+23]

Hard learning problems have been shown to be an important source of cryptographic hardness [Reg10; Pie12]. Popular instances of such problems include Learning Parity with Noise (LPN), Learning With Errors (LWE), which is a generalization to larger moduli [Reg05b], and Learning With Rounding (LWR), which is a deterministic version [BPR12]. They have found many applications for the design of basic cryptographic primitives such as efficient authentication protocols [HB01; Hey+12] or

pseudorandom functions [YS16], and advanced cryptographic primitives such as identity-based [GPV08; Cas+10] or fully homomorphic [Gen09; BV11b] encryption. They have also gained particular interest in the context of post-quantum cryptography, as witnessed by the signature scheme CRYSTALS-Dilithium [Duc+18] and the encryption scheme CRYSTALS-Kyber [Bos+18b], which have been selected by the NIST to become future standards.[8] Furthermore, their algebraic structure makes a (key-homomorphic) part of their implementation quite amenable to masking against side-channel attacks. Yet, protecting the error generation part remains a challenge, whether in the case of probabilistic (e.g., LPN-based) or deterministic (LWR-based) designs. This is because the noise generation itself can become the target of a side-channel attack [GLS14], and the rounding functions needed for the deterministic designs to be secure are non-linear and therefore more difficult to mask [Bre+14].

Hard *physical* learning problems have been introduced as a possible remedy to these implementation issues. Conceptually, their goal is to leverage the physical features of an implementation in order to generate errors or to round, as required to implement primitives based on hard learning problems. A typical application of such physical problems is fresh re-keying against side-channel attacks [Med+10], of which the goal is to generate an ephemeral (e.g., block cipher) key thanks to an easy-to-protect operation. Various pieces of work studied the cryptanalysis of such schemes instantiated with a binary field multiplication [BFG14; Bel+15; PM16; GJ19]. Dziembowski et al. proved the security of binary re-keying based on a Learning Parity with Leakage (LPL) problem that reduces to the standard LPN problem [Dzi+16]. Unfortunately, the concrete level of noise in the leakages needed for their proof that LPL is secure was shown to be quite high. More recently, Duval et al. showed that it is possible to design re-keying schemes based on a Learning With Physical Rounding (LWPR) problem [Duv+21]. Here, the main observation is that many practical leakage functions, such as the Hamming weight one [MOP07], are non-injective. It is therefore tempting to let the leakage function perform the rounding, removing the hassle of computing this rounding explicitly and protecting it against side-channel attacks. Non-injectivity alone (i.e., without noise nor further constraints on the field in which computations take place) is known to be insufficient. For example, in binary fields the Hamming weight of a secret value provides a linear equation of its bits. But combining a non-injective leakage function with computations in a prime field was

[8] `https://csrc.nist.gov/Projects/post-quantum-cryptography`.

suggested as a possible source of "crypto-physical" hardness, emulating Boneh et al.'s cryptographic dark matter [Bon+18].

It is easy to see that practically-relevant secure LWPR instances could be a game changer for the secure implementations of cryptographic algorithms. First, masked block ciphers usually incur performance overheads that are quadratic in the number of shares [ISW03; GR17]. By contrast, the cost of masked key-homomorphic primitives scales linearly in this number and can benefit from simple refreshing schemes (with linear randomness requirements) [Bar+17]. Second, the analysis of key-homomorphic primitives in the probing model is trivial and does not raise composability issues since, as for linear functions, their computation can be implemented share by share and is therefore probe-isolating non-interferent [CS20]. Third, they intrinsically mitigate the risks of security flaws caused by physical defaults like glitches [MPG05] or transitions [Cor+12] that can lead to shares re-combinations, thanks to the independent manipulation of the shares they allow [CS21a]. Eventually, is was recently hinted that they also have good potential to significantly improve the performance vs. side-channel security tradeoff of CCA-secure public key encryption schemes [Hof+22]. The latter is well-motivated given the acknowledged difficulty to protect schemes like CRYSTALS-Kyber against side-channel attacks [Rav+20b; Ngo+21; Uen+22].

All these reasons give a strong incentive for understanding the security of the LWPR assumption under realistic leakage functions. The CHES 2021 work of Duval et al. made a first step in this direction by analyzing LWPR in the meaningful though specific case of Hamming weight leakages, and proposed parallel (FPGA) implementations processing 124 bits per cycle as a first cryptanalysis target. This investigation therefore suggested as fundamental problem the analysis of leakage functions that (even slightly) deviate from the Hamming weight case, while also leaving doubts regarding the possibility to ensure the security of LWPR in a serial (e.g., 32-bit software) implementation context.

In this paper, we contribute to these problems in three main directions.

As an appetizer, we confirm the intuition that securing LWPR in a serial implementation context requires additional side-channel countermeasures. This is because attacks exploiting so-called worst-case leakages can then be mounted with practical data complexity. By worst-case leakages, we mean the most informative values provided by the leakage function (e.g., the extreme ones in the Hamming weight case). This observation suggests the use of shuffling as a natural complement to serial LWPR which, if well implemented, has the effect of emulating a parallel

implementation [HOM06; Vey+12; UBS21]. It also allows us to focus the rest of our investigations on parallel LWPR instances.

Next, we generalize the security of the LWPR assumption beyond the Hamming weight function. For this purpose, we first consider attacks taking advantage of any value of the leakage function, which we denote as "any-case leakages" in the paper. We show that security against algebraic cryptanalysis is preserved in the practical context of linear leakage functions (i.e., functions that can be expressed as a weighted sum of the bits manipulated by an implementation) [SLP05], under reasonable conditions on the accuracy of the weights. We also argue that these guarantees can be extended towards higher-order (e.g., quadratic) leakage functions. We then show that attacks exloiting worst-case leakages (used to rule out unprotected serial LWPR instances) can be prevented in the parallel case, with similar assumptions as to resist any-case leakages.

We combine these investigations with experiments confirming the practical relevance of our assumptions for realistic hardware implementations. As a result, we make a crucial step towards making deterministic hard physical learning problems a credible alternative to secure cryptographic implementations. And we conclude the paper by identifying interesting scopes for further research.

We note that the security of LWPR with linear or quadratic leakage functions relies on different arguments than previously used for similar constructions. For example in [Duv+21], the high-level idea behind the hardness of LWPR with Hamming weight leakages is that the composition of linear functions in different structures is sufficient to provide security. This holds since the composition gives functions with good cryptographic parameters in both initial structures. It corresponds to the main conclusion of Boneh et al.'s cryptographic dark matter [Bon+18] which proposes PRF constructions by only combining additions modulus p and modulus q for any pair of different primes p and q. The same general idea has been reused recently in [Din+21]. In the case of LWPR with linear or quadratic leakage functions, additionally to the composition of functions over two different structures, good cryptographic properties are obtained directly from the non-injectivity of one of the functions. This new insight allows us to generalize the results to more functions and to strengthen their connection to practice, since the non-injectivity of a leakage function is typically something that can be characterized by existing side-channel evaluation setups & tools.

4.3.1 Background

In this section, we refer to the background on LWPR provided in Section 2.2.2.

Regression-based side-channel analysis

Linear regression is among the most popular tools for profiling a side-channel leakage model. Originally introduced by Schindler et al. [SLP05], it has been rapidly established as an efficient alternative to template attacks [GLP06]. Its main idea is to approximate the deterministic part of the leakage function as a weighted sum of n_b well-chosen basis function that we denote as β_i:

$$\mathsf{M_r}(\boldsymbol{v}) = \sum_{i=1}^{n_b} a_i \cdot \beta_i(\boldsymbol{v}),$$

with the a_i's $\in \mathbb{R}$. For a t-bit value $\boldsymbol{v}$, a typical choice is to consider $n_b = t$ and to use the bits of $\boldsymbol{v}$ as basis functions.[9] We will denote the resulting class of leakage functions as linear models $\mathsf{M_{r1}}(\boldsymbol{v})$, which generalizes Hamming weight leakages where $a_i = 1 \ \forall \ 1 \leq i \leq n_b$. The model can be refined by adding more basis functions. Typically, we can add the $\binom{t}{\delta}$ basis functions of degree δ, which we denote as quadratic leakage models $\mathsf{M_{r2}}(\boldsymbol{v})$ when $\delta = 2$, cubic leakage models $\mathsf{M_{r3}}(\boldsymbol{v})$ when $\delta = 3, \ldots$, until the bijective model $\mathsf{M_{r}}_t$ where $\delta = t$ and $n_b = 2^t$.

Profiling a leakage model then essentially boils down to estimate the vector of coefficients $\boldsymbol{a}$, by applying the least square method. For this purpose, the evaluator first collects a vector of n_p profiling traces $\boldsymbol{l} = [l^1, l^2, \ldots, l^{n_p}]$, corresponding to values $\boldsymbol{v}^1, \boldsymbol{v}^2, \ldots, \boldsymbol{v}^{n_p}$. She then builds a $n_p \times n_b$ matrix:

$$B = \begin{pmatrix} \beta_1(\boldsymbol{v}^1) & \beta_2(\boldsymbol{v}^1) & \cdots & \beta_{n_b}(\boldsymbol{v}^1) \\ \beta_1(\boldsymbol{v}^2) & \beta_2(\boldsymbol{v}^2) & \cdots & \beta_{n_b}(\boldsymbol{v}^2) \\ \vdots & \vdots & \ddots & \vdots \\ \beta_1(\boldsymbol{v}^{n_p}) & \beta_2(\boldsymbol{v}^{n_p}) & \cdots & \beta_{n_b}(\boldsymbol{v}^{n_p}) \end{pmatrix}.$$

Eventually, the vector of coefficients minimizing the means square error is:

$$\hat{\boldsymbol{a}} = [\hat{a_1}, \hat{a_2}, \ldots, \hat{a_{n_b}}] = (B^T \cdot B)^{-1} \cdot B^T \cdot \boldsymbol{l}.$$

We note that the analysis in [SLP05] considers noisy leakages and therefore adds a probabilistic component to the previous deterministic function. We do not detail this part since LWPR (conservatively) assumes noise-free leakages.

[9] Or $n_b = t + 1$, with a constant term to capture DC effects in the measurements.

Correlation-based security evaluations

Given a leakage model, the main question of a side-channel security evaluation is to determine the number of attack traces needed to recover a key [SMY09]. In the context of univariate leakage functions that we consider in this paper, Pearson's correlation is among the most popular evaluation tools [BCO04]. We next describe the basic approximations we will use in our empirical analyses.

Let us first denote a vector of n_a attack values $\overline{\boldsymbol{v}} = [\boldsymbol{v}^1, \boldsymbol{v}^2, \ldots, \boldsymbol{v}^{n_a}]$. From those values, an evaluator can compute the modeled leakage vector:

$$\boldsymbol{m} = [\mathsf{M}_{\mathsf{r}}(\boldsymbol{v}^1), \mathsf{M}_{\mathsf{r}}(\boldsymbol{v}^2), \ldots, \mathsf{M}_{\mathsf{r}}(\boldsymbol{v}^{n_a})].$$

Similarly, she can measure a vector of actual leakage traces $\boldsymbol{l} = [l^1, l^2, \ldots, l^{n_a}]$. Viewing these vectors as n_a samples of random variables M and L, the data complexity of a Correlation Power Analysis (CPA) is given by [Man04]:

$$N = \frac{c}{\hat{\rho}(M, L)^2},$$

where c is a small constant and $\hat{\rho}$ denotes Pearson's correlation coefficient.

In our experimental evaluations, we will in particular be interested by the comparison between different leakage models, with the goal to evaluate whether simplifying a model (e.g., by limiting its degree or quantizing it) results in a significant information loss. Pearson's correlation is convenient for this purpose. For example, let us assume a model M_1 that is a simplification of a model M_2, leading to model errors captured by a random variable E so that $M_1 = M_2 + E$. Then, thanks to the correlation chain rule given in [Sta+06] we have:

$$\hat{\rho}(M_1, L) = \hat{\rho}(M_1, M_2) \cdot \hat{\rho}(M_2, L).$$

As a result, the increase of data complexity that is due to simplifying the model M_2 into M_1 is reflected by the factor $f = \frac{1}{\hat{\rho}(M_1, M_2)^2}$: if $f \approx 1$ the attack using the simplified model is close to the one using the complex model; if $f > 1$ the simpler model misses some leakage features and the attack using this model requires f times more traces than the attack using the more complex model.

Cryptographic criteria & p-ary functions

Since the re-keying scheme we consider in this paper is linear over $\mathbb{F}_p$, we will study the properties of the leakage function embedded in $\mathbb{F}_p$ that

a side-channel adversary can obtain. For this purpose, we adapt tools from the analysis of cryptographic criteria for Boolean functions from $\mathbb{F}_2^n$ to $\mathbb{F}_2$ (e.g., [Car21]) and study the cryptographic properties of functions from $\mathbb{F}_p^n$ to $\mathbb{F}_p$.

Definition 18 (p-ary function)**.** *For p a prime, a p-ary function f in n variables (an n-variable p-ary function) is a function from $\mathbb{F}_p^n$ to $\mathbb{F}_p$. The set of all p-ary functions in n variables is denoted by $\mathcal{F}_{p,n}$, and $|\mathcal{F}_{p,n}| = p^{p^n}$.*

Definition 19 (Algebraic normal form and degree (e.g., [Hou18]))**.** *We call Algebraic Normal Form (ANF) of a p-ary function f its n-variable polynomial representation over $\mathbb{F}_p$ belonging to $\mathbb{F}_p[x_1, \ldots, x_n]/(x_1^p - x_1, \ldots, x_n^p - x_n)$:*

$$f(x) = \sum_{S \subset [0,p-1]^n} a_S \left(\prod_{i \in [1,n]} x_i^{S_i} \right) = \sum_{S \subset [0,p-1]^n} a_S x^S,$$

where $a_S \in \mathbb{F}_p$. The ANF of f is unique, and the algebraic degree of f is defined as $\mathsf{deg}(f) = \max_{\{S \mid a_S \neq 0\}} \sum_{i=1}^{n} S_i$ (with the convention that $\mathsf{deg}(0) = 0$).

Definition 20 (Nonlinearity)**.** *The nonlinearity $\mathsf{nl}(f)$ of a p-ary function $f \in \mathcal{F}_{p,n}$, is the minimum Hamming distance between f and all the affine functions in $\mathcal{F}_{p,n}$:*

$$\mathsf{nl}(f) = \min_{g,\, \mathsf{deg}(g) \leq 1} \{d_H(f, g)\},$$

where $d_H(f, g)$ is the Hamming distance $|\{x \in \mathbb{F}_p^n \mid f(x) \neq g(x)\}|$ between f and g.

4.3.2 Appetizer: threat model and serial LWPR

Before describing our technical contributions, we quickly recall the threat model we consider, which is illustrated in Figure 4.12. We also come back on the impact of an attack against serial LWPR instances outlined in the appendix of [Duv+21], by specializing it to the case of a Hamming weight function for simplicity. We use this example for two purposes: first, confirming that securing serial LWPR instances requires additional countermeasures; second: introducing the difference between attacks using worst-case leakages and any-case leakages.

In brief, the idea of fresh re-keying schemes is that it is possible to efficiently mask their key-homomorphic part (i.e., the product between

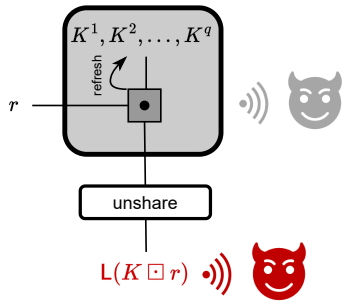


Figure 4.12: Fresh re-keying with LWPR: threat model.

the secret matrix $\boldsymbol{K}$ and the public vector $\boldsymbol{r}$) and to recombine the output of this product for which the adversary can only observe the leakage. In other words, the multiplication we consider in Definition 11 will be performed on the shares $\boldsymbol{K}^1, \boldsymbol{K}^2, \ldots, \boldsymbol{K}^q$ and then recombined such that the fresh key is $\boldsymbol{K} \boxdot \boldsymbol{r} = \sum_{i=1}^{q} (\boldsymbol{K}^i \boxdot \boldsymbol{r})$.

This fresh key will then be used without masking for a single (e.g., block cipher) encryption, which is expected to be significantly harder to break than if using the long-term key multiple times with the same cipher. As a result, there are two natural attack paths against such re-keying schemes. The first one (depicted in gray on Figure 4.12) is to target the masked product. Evaluating it can be done with standard side-channel analysis techniques, enjoying the advantages listed in introduction (linear overheads, no composability issues, resistance against glitches). The analysis of this attack path given in [Duv+21] remains identical and we therefore do not repeat it. The second option (depicted in red on Figure 4.12) is to target the output of the product after its shares have been re-combined. Evaluating it requires less standard techniques and can be done in different models. Binary fresh re-keying schemes like [Med+10] require noisy leakages to be secure. We consider a more conservative model where the adversary obtains noise-free leakages. As a result, the analysis of this attack path becomes conceptually similar to the analysis of a stream cipher, the product playing the role of an LFSR and the leakage function the one of a filter.

Given this threat model, the attack against serial LWPR implementations is pretty simple. Say we have an instance of LWPR with $p = 2^{31} - 1$ where the adversary can observe the Hamming weight of every 31-bit word of $\boldsymbol{K} \boxdot \boldsymbol{r}$. Then, she can filter the leakages and retain the ones with (worst-case) value 0, which has a single preimage. Every such event gives a linear equation in the key elements of the corresponding line of $\boldsymbol{K}$, and each such line can be recovered with $(n + 1)$ linear equations.

Since p is only mid-size, these events happen with a concretely reachable probability $\frac{1}{p}$. As a result, after the generation $p\,(n+1)$ LWPR samples, the full key is compromised with good probability.

Since it is hard to rule out that (close to) Hamming weight leakages can be observed in practice (as Section 4.3.5 will confirm experimentally), this attack suggests that serial instances of LWPR cannot be secure without additional side-channel countermeasures. Various candidates can be considered for this purpose. As it is argued in [Duv+21], exploiting extreme Hamming weights becomes difficult when the level of parallelism increases (since they become exponentially less likely), and shuffling therefore appears as the most natural one. Indeed, if well implemented, shuffling makes the m intermediate computations of a serial implementation hard to distinguish so that the adversary does not gain more information than if she was provided with the sum of these leakages, which emulates a parallel implementation [HOM06; Vey+12; UBS21].[10] We show in Section 4.3.4 that attacks exploiting worst-case leakages can be generalized, and that a security argument can be given based on the degree of the (linear) leakage function and the accuracy of its coefficients. Beforehand, we consider another important class of (algebraic) attacks where the adversary exploits any-case leakages (i.e., where the leakages are not filtered and all the measurements are used). For this purpose, we generalize the former security analysis of Duval et al. from the specialized Hamming weight leakage function towards any linear leakage function. Interestingly, for those attacks we are even able to give a security argument for serial implementations, which then easily extends to parallel ones.

4.3.3 LWPR for linear leakage functions

While reasonable as a first step, considering security against Hamming weight leakages is oversimplifying since the different bits of an implementation can consume more or less power, due to different load capacitances. This is precisely what is modeled by regression-based attacks [SLP05]. In this section, we therefore analyze the security of LWPR in this generalized context.

For this purpose, we first observe that the product $\boldsymbol{K} \boxdot \boldsymbol{r}$ is linear over $\mathbb{F}_p$. As a result, we focus on the (linear invariant) cryptographic criteria of the remaining function L_{g} considered as a function over $\mathbb{F}_p$. If the adversary can approximate well the real function L_{g} by a p-ary

[10] Increasing the size of p could provide similar security benefits at higher cost.

function, she obtains a system of equations over $\mathbb{F}_p$ where the unknowns are (affine combinations of) the key elements.

This gives a modeling of $\mathsf{L}_{\mathsf{g}}(\boldsymbol{K} \boxdot \boldsymbol{r})$ as equations over $\mathbb{F}_p$ where the only unknowns are the $\mathbb{F}_p$ elements of $\boldsymbol{K}$. We next carry out such an analysis for a linear leakage model in the serial case, by considering these models as functions over $\mathbb{F}_p$, which we define as s-bounded pseudo-linear functions.

Definition 21 (s-bounded pseudo-linear functions)**.** *We denote as* $\mathsf{F}_{\boldsymbol{a}}$ *the function characterized by* $\boldsymbol{a} \in \mathbb{F}_p^t$*, where* $t = \lceil \log p \rceil$ *and defined as:*

$$\mathsf{F}_{\boldsymbol{a}} : \ \mathbb{F}_p \to \mathbb{F}_p, \ y \mapsto \sum_{i=0}^{t-1} a_i \cdot \mathsf{g}(y)_i. \tag{4.1}$$

We call $\mathsf{F}_{\boldsymbol{a}}$ *s-bounded pseudo-linear if each* a_i *belongs to* $[0, s]$*, with* $s \in \mathbb{F}_p$ *a security parameter, and we call* C_1^s *the class of s-bounded pseudo-linear functions.*

We recall that the g function associates elements of $\mathbb{F}_p$ to their binary representation. The name pseudo-linear comes from the fact that this function is linear in the bits of the binary representation of $\boldsymbol{a}$. However, it corresponds to a much higher degree p-ary function. This increase of the degree when projecting a binary representation into a prime field is the root of the LWPR hardness. We next consider that an s-bounded $\mathsf{F}_{\boldsymbol{a}}$ is the p-ary function used to approximate L_{g}.

We note that this definition implicitly relies on two assumptions: one on the degree of the physical leakage function and another one on the number of values taken by the coefficients a_i. The practical relevance of these two assumptions is discussed in Section 4.3.5. We also note that the proposed analysis captures an important class of (algebraic) attacks against LWPR. We will show that the non-injectivity of a one-variable p-ary function is sufficient to prove a lower bound on its degree and its nonlinearity. On the one hand, the high algebraic degree of a p-ary function allows thwarting the attacks based on solving a linearized algebraic system. On the other hand, the high nonlinearity prevents the attacks based on solving noisy linear algebraic systems, which use the best approximation of a p-ary function by an affine function. But as usual in cryptanalysis, nothing prevents that other characterizations or attack vectors lead to better results.

Definition 22 (Preimage sets & main preimage)**.** *Let* $f \in \mathcal{F}_{p,m}$*, we call:*

- $\mathcal{A}_f(y) = \{x \in \mathbb{F}_p^m \mid f(x) = y\}$ *for* $y \in \mathbb{F}_p$*, the set of preimages of* y *through* f*.*
- $\mathsf{u}_f \in \mathbb{F}_p$ *a main image of* f *defined as* $\forall y \in \mathbb{F}_p, |\mathcal{A}_f(\mathsf{u}_f)| \geq |\mathcal{A}_f(y)|$*.*
- $\mathsf{v}_f \in [1, p^m]$ *the main preimage size of* f*, defined as* $\mathsf{v}_f = |\mathcal{A}_f(\mathsf{u}_f)|$*.*
- $\mathsf{w}_f \in [0, p-1]$ *the no preimage set's size of* f*, defined as*

$$\mathsf{w}_f = |\{y \in \mathbb{F}_p \mid \mathcal{A}_f(y) = \emptyset\}|$$

We first show a bound on the degree of f based on its main preimage size.

Proposition 6 (Main preimage size and algebraic degree)**.** *Let* $f \in \mathcal{F}_{p,1}$*, if its main preimage* $\mathsf{v}_f < p$ *then its algebraic degree* $\mathsf{deg}(f) \geq \mathsf{v}_f$*.*

Proof. Since $\mathsf{v}_f < p$, f is not a constant function and $f - \mathsf{u}_f$ is not the null function. Accordingly, the function $f - \mathsf{u}_f$ admits v_f different zeros in $\mathbb{F}_p$, and there exists v_f terms $(x - z_i)$ dividing f. Therefore its degree is at least v_f and, since the algebraic degree is affine invariant, $\mathsf{deg}(f) = \mathsf{deg}(f - \mathsf{u}_f) \geq \mathsf{v}_f$. □ □

Furthermore, when f is not enough non-injective, composing the function can be used to determine better its degree as shown in the following corollary:

Corollary 2. *Let* $f \in \mathcal{F}_{p,1}$ *and* $r \in \mathbb{N}^*$*, we denote* $f^{\circ r} = f \circ f \circ \cdots \circ f$ *the composition of* f r *times. If* $\mathsf{v}_{f^{\circ r}} < p$ *then* $\mathsf{deg}(f) \geq (\mathsf{v}_{f^{\circ r}})^{1/r}$*.*

We next show a bound on the nonlinearity of f.

Proposition 7 (Main preimage size, no preimage set's size and nonlinearity)**.** *Let* $f \in \mathcal{F}_{p,1}$ *such that its main preimage size* $\mathsf{v}_f < p$*, then:*

$$\mathsf{nl}(f) \geq \min\left(p - \mathsf{v}_f, \max\left(\mathsf{v}_f - 1, \mathsf{w}_f\right)\right).$$

Proof. We derive the bound on the nonlinearity by first considering the distance to constant functions, and then to any affine function. Since $\mathsf{v}_f < p$, f is not constant, and it will agree on at most v_f values with a constant function c. Hence $d_H(f, c) \geq p - \mathsf{v}_f$. Besides, p-ary affine (not constant) functions in one variable are bijective. Accordingly, each element of $\mathbb{F}_p$ is the image of exactly one element. Therefore, for any affine (not constant) function in $\mathcal{F}_{p,1}$, there exists at least $\mathsf{v}_f - 1$ elements of $\mathbb{F}_p$ where the image of f is different from the affine function's one (all except one of the preimages of u_f). Moreover, since w_f elements are not in the image of f, f disagrees at least w_f times with any affine (not constant) function, which give the final bound. □ □

Finally, we bound the main preimage size of s-bounded pseudo-linear functions.

Proposition 8 (Properties of s-bounded pseudo-linear functions)**.** *Let $f \in \mathsf{C}_1^s$ with $ts < p$, where $t = \lceil \log p \rceil$, then the following holds:*

- $\mathsf{v}_f \geq \lceil \frac{p}{ts+1} \rceil$,
- $\mathsf{w}_f \geq p - ts - 1$.

And assuming $\mathsf{v}_f \neq p$, we further have:

- $\mathsf{deg}(f) \geq \lceil \frac{p}{ts+1} \rceil$,
- $\mathsf{nl}(f) \geq \min\left(p - \mathsf{v}_f, \max\left(\lceil \frac{p}{ts+1} \rceil - 1, p - ts - 1\right)\right)$.

Proof. Since $ts < p$, we obtain that for all $x \in \mathbb{F}_p$, $0 \leq f(x) \leq ts$, hence $\mathsf{w}_f \geq p - ts - 1$. Thereafter, $f(x)$ can take at most $ts + 1$ values, hence there exists at least a preimage set with cardinal $\lceil \frac{p}{ts+1} \rceil$ or more. The last two items are obtained using Proposition 6, Proposition 7 and two first items. □ □

Note that the condition $ts < p$ guarantees that no reduction takes place independent of the input of the pseudo-linear leakage function in Equation 4.1.

For a given $\boldsymbol{a} \in \mathbb{F}_p^t$, the output of $\mathsf{F}_{\boldsymbol{a}}$ gives a system of equations where the key elements are the unknowns. Therefore, solving the algebraic system given by images of $\mathsf{F}_{\boldsymbol{a}}$ allows retrieving the key. We next explore two methods for solving such a system over $\mathbb{F}_p$, and demonstrate that the time and data complexity of these methods are sufficiently high for our instances in Section 4.3.5 .

Concretely, an adversary can either solve the system of k variables directly or use the higher-order correlation approach presented in [Cou02]. Higher-order correlation attacks consist in approximating $\mathsf{F}_{\boldsymbol{a}}$ by a degree-d function h, and then solving systems of equations until one is such that $\mathsf{F}_{\boldsymbol{a}}$ and h coincide on all these equations. An adversary can create additional equations if needed, for instance through a linear combination of the existing ones. Hence, the time complexity of solving a noisy degree-d system of equations over $\mathbb{F}_p$ can be written as $C(1-\varepsilon)^{-D}$ with C the time complexity to solve a degree-d system of equations in k' variables over $\mathbb{F}_p$, $(1-\varepsilon)$ the probability of the approximation to be correct for one equation, k' the number of variables (i.e., at least the number of key variables k, but it can be more if techniques introducing new

variables are used), and D the quantity of data necessary (at least the number of variables k').

Exact algebraic system attack path. Let us for now fix the security parameter s to 2^{12} (the practical relevance of this value will be discussed in Section 4.3.5). Trying to solve the system directly given by the output of the function $\mathsf{F}_{\boldsymbol{a}}$ with this value and a modulus $p = 2^{31} - 1$, we find $\mathsf{v}_f \geq \lceil \frac{2^{31}-1}{31 \cdot 2^{12}+1} \rceil \geq 2^{14}$ (since $t = \lceil \log(p) \rceil$). Therefore, the adversary would have to solve a system of equations of degree at least 2^{14} in $n+1$ variables over $\mathbb{F}_{2^{31}-1}$ to recover the first row of $\boldsymbol{K}$. There are usually two main families of algorithms considered to solve polynomial systems of equations: the XL algorithms and Gröbner bases algorithms such as F4 [Fau99] or F5 [Fau02]. The complexities of these families of algorithms are studied in different works. We refer to [YC04] for the XL family and to [BFS15] for F4/F5. For clarity, we first discuss the results given by the simpler approach of linearization. Let V_d denote the number of monomials over $\mathbb{F}_p$ with degree at most d in the k variables of the system. The linearization attacks consists in considering each of these monomials as a new variable, and then in inverting the linear system using Gaussian elimination. Thereafter, the corresponding time complexity to solve a system with this approach is simply $\mathcal{O}\left(V_d^3\right)$. XL algorithms or Gröbner bases algorithms should improve over this complexity. However, Gaussian elimination algorithms give a convenient estimate of the complexity to solve a linear system, that is sufficient for our purposes and which can be combined with mild security margins if needed. For example, XL and F4/F5 can be much faster when more polynomials are available, leading to a time complexity reduced to $\mathcal{O}\left(V_d^\omega\right)$, where $2 \leq \omega \leq 3$ is a linear algebra constant that depends on the algorithm used. We make the conservative choice to consider the complexity $\mathcal{O}\left(V_d^2\right)$ for our concrete security estimations.

Noisy linear system attack path. We also consider the complexity of linearizing the d-degree system, then solving the resulting linear system. For this purpose, we use a similar argument as the one given in [Duv+21]. Namely, we first observe that the number of variables after linearization is $V_d = |\{\boldsymbol{v} \in [0, p-1]^k, 0 \leq \sum_{i=1}^k \boldsymbol{v}_i \leq d\}| = \sum_{l=0}^d \binom{l+k-1}{l} = \binom{k+d}{k}$ (using the stars and bars theorem) and again assume $C = \mathcal{O}\left(V_d^\omega\right)$. The noisy linear system approach then just implies considering a probability of error $\varepsilon = \mathsf{nl}(\mathsf{F}_{\boldsymbol{a}})/p$ for the equation approximation. In this case, with a degree $d = 1$, we have $V_1 = k + 1$ and the time complexity is at least $\mathcal{O}\left((k+1)^\omega (p/(p - \mathsf{nl}(\mathsf{F}_{\boldsymbol{a}})))^k\right)$. Using the bound of Proposition 8 therefore provides the required security estimation.

These theoretical bounds will be made concrete in Section 4.3.5, where actual instances of $\mathsf{F}_{\boldsymbol{a}}$ are given, with their associated values v_f and w_f.

Adaptation to the parallel case. In the serial case above, we studied the properties of the function an adversary can obtain from the leakage corresponding to one ($\mathbb{F}_p$) element of the ephemeral key, each element of $\boldsymbol{y}$ depending on a different row of the long term key. In the parallel case, the adversary gets less information since she obtains the leakage on the whole $\boldsymbol{y}$ and not independently on each coefficient. In this case, the leakage is the sum of the leakages obtained in the serial case. Therefore, in terms of p-ary functions, calling f_i the functions considered in the serial case, the one in the parallel case is $f' \in \mathcal{F}_{p,m}$:

$$f'(\boldsymbol{y}) = \sum_{i=1}^{m} f_i(y_i).$$

Since f' is a direct sum of functions (sum of functions acting on different variables) its properties can be derived from the ones of the f_i's. Namely, $\mathsf{deg}(f_i)$ and $\mathsf{nl}(f_i)$ can be obtained from Proposition 8, $\mathsf{deg}(f') = \max_i \mathsf{deg}(f_i)$ and the nonlinearity of f' can be derived from the following result:

Property 1 (Adapted from [Cos+22b], Proposition 1)**.** *Let $\mathbb{G}$ be a group, Let $h = f+g$ be the direct sum of the functions f and g with n and m $\mathbb{G}$-variables respectively. Then,* $\mathsf{nl}_{\mathbb{G}}(h) \geq \max(|\mathbb{G}|^n \mathsf{nl}_{\mathbb{G}}(g), |\mathbb{G}|^m \mathsf{nl}_{\mathbb{G}}(f))$.

Accordingly, we also obtain $\mathsf{nl}(f') \geq \max_i p^{m-1}\mathsf{nl}(f_i)$. The complexity of the attacks against parallel LWPRG implementations is therefore increased thanks to this higher nonlinearity and the higher number of variables km.

4.3.4 Security against worst-case leakages

We now extend the approach of [Duv+21] for attacks taking advantage of worst-case leakages (intuitively outlined in Section 4.3.2). Precisely, we investigate the complexity of attacks that benefit from the existence of leakages with small preimage sets. In the parallel case, the adversary obtains images z such that:

$$z = f'(\boldsymbol{y}) = \sum_{i=1}^{m} f_i(y_i) = \sum_{i=1}^{m} f_i(\boldsymbol{k}_i \boxdot \boldsymbol{r}),$$

where $\boldsymbol{k}_i$ denotes the i-th row of $\boldsymbol{K}$. For $z = f'(\boldsymbol{y})$, the m-variable p-ary function f' has $|\mathcal{A}_{f'}(z)|$ preimages that we denote $(y_1^{(j)}, \ldots, y_m^{(j)})$, for

$j \in [1, |\mathcal{A}_{f'}(z)|]$. From an image z, the adversary obtains that a linear system of m equations, namely $\{\boldsymbol{k}_i \boxdot \boldsymbol{r} = y_i^{(j)}$, for $i \in [1, m]\}$, is correct for one j. She can then collect $n+1$ samples $z_1, \ldots, z_{n+1}$ and solve the $\prod_{i=1}^{n+1} |\mathcal{A}_{f'}(z_i)|$ possible linear systems in $m\,(n+1)$ variables (in order to retrieve $\boldsymbol{K}$). Only one of these systems is the correct one, and the correct solution can be verified with more samples. Since by definition $|\mathcal{A}_{f'}(z_i)| \leq \mathsf{v}_{f'}$, the time complexity of this attack is upper bounded by $\mathcal{O}(\mathsf{v}_{f'}^{n+1} \cdot (m(n+1))^{\omega})$ (following the analysis of Section 4.3.3).

We extend this attack by considering algebraic systems of higher degree rather than linear systems. When z gives $|\mathcal{A}_{f'}(z)|$ preimages, instead of considering $m \cdot |\mathcal{A}_{f'}(z)|$ linear equations, we can build the m equations:

$$\prod_{j=1}^{|\mathcal{A}_{f'}(z)|} (\boldsymbol{k}_i \boxdot \boldsymbol{r} - y_i^{(j)}) = 0, \text{ for } i \in [1, m],$$

where the m equations have degree $|\mathcal{A}_{f'}(z)|$. We first observe that the cases we previously described, with $m \cdot |\mathcal{A}_{f'}(z)|$ equations of degree 1 and $m \cdot 1$ equations of degree $|\mathcal{A}_{f'}(z)|$, are two extreme cases. We can also create $m \cdot b$ equations of degree c such that $bc = |\mathcal{A}_{f'}(z)|$, where only one of the b system of equations is correct. As a result, the attack becomes a tradeoff between the number of polynomial systems of fixed degree and their algebraic degree, since it corresponds to the resolution of b^{n+1} polynomial systems of degree c, with $bc = |\mathcal{A}_{f'}(z)|$.

The complexity of the attack relies on two main factors: first, the size of the preimage sets considered $|\mathcal{A}_{f'}(z)|$, intervening in the number of systems and the degree of the polynomial systems; second, the probability to get samples z_i with small preimage sets. We argue that this attack is not a threat since either the time complexity is higher than the security level targeted when $\mathcal{A}_{f'}(z)$ is large, or the data complexity is too high when $\mathcal{A}_{f'}(z)$ is small. To do so, we first give the expression of $|\mathcal{A}_{f'}(z)|$ when f' is the direct sum of m p-ary functions. Then, we show how the complexity of the attack can be bounded from the values of $|\mathcal{A}_{f'}(z_i)|$ from the samples z_i an adversary obtains. Finally, we give a proposition to bound the probability of getting z_i such that the preimage set is small, and we conclude by applying these results to a practical example.

Concretely, when f' is the direct sum of m p-ary functions f_i, the expression of $|\mathcal{A}_{f'}(z)|$ is given by:

$$|\mathcal{A}_{f'}(z)| = \sum_{\substack{y_1,\dots,y_m \in \mathbb{F}_p \\ y_1+\cdots+y_m=z}} \prod_{i=1}^{m} |\mathcal{A}_{f_i}(y_i)|.$$

Then, since the time complexity of the attack comes from the resolution of b^{n+1} polynomial systems of degree c such that $bc = |\mathcal{A}_{f'}(z)|$, using the analysis of Section 4.3.3 the complexity is $\mathcal{O}(b^{n+1} \cdot (mV_c)^\omega)$. We rewrite $b^{n+1} \cdot (mV_c)^\omega$ to express it relatively to $|\mathcal{A}_{f'}(z)|$ as follows:

$$\begin{aligned} b^{n+1} \cdot (mV_c)^\omega &= \left(\frac{|\mathcal{A}_{f'}(z)|}{c}\right)^{n+1} \binom{n+1+c}{n+1}^{\omega} m^\omega, \\ &\geq \left(\frac{|\mathcal{A}_{f'}(z)|}{c}\right)^{n+1} \frac{c^{n+1}}{(n+1)!} \binom{n+1+c}{n+1}^{\omega-1} m^\omega, \\ &\geq \frac{|\mathcal{A}_{f'}(z)|^{n+1}}{(n+1)!}. \end{aligned}$$

Finally, we use the following proposition in order to bound the probability of getting images coming from small preimage sets:

Proposition 9 (Probability of getting images from small preimage sets)**.** *Let $f' \in \mathcal{F}_{p,m}$ and $B \in \mathbb{N}$ such that $B < p^m/|\operatorname{Im}(f')|$ where $\operatorname{Im}(\cdot)$ denotes the image, the probability $p_{f',B}$ of choosing at random $z \in \mathbb{F}_p^m$ such that $|\mathcal{A}_{f'}(z)| \leq B$ follows:*

$$p_{f',B} \leq \frac{(|\operatorname{Im}(f')| - 1)B}{p^m}.$$

In particular, if f' is the direct sum of m p-ary functions with C_1^s and $mts < p$ (where $t = \lceil \log p \rceil$), the probability becomes:

$$p_{f',B} \leq \frac{mstB}{p^m}.$$

Proof. First, we show the result in the general case. By definition, the preimage sets $\mathcal{A}_{f'}(z)$ for $z \in \mathbb{F}_p$ are a partition of $\mathbb{F}_p^m$, Therefore:

$$p^m = \sum_{z \in \mathbb{F}_p} |\mathcal{A}_{f'}(z)| = \sum_{z \in \operatorname{Im}(f')} |\mathcal{A}_{f'}(z)|.$$

Since $B < \frac{p^m}{|\operatorname{Im}(f')|}$, at least one of the images is such that $|\mathcal{A}_{f'}(z)| > B$. Hence, at most $|\operatorname{Im}(f')| - 1$ images are such that $|\mathcal{A}_{f'}(z)| \leq B$, and the

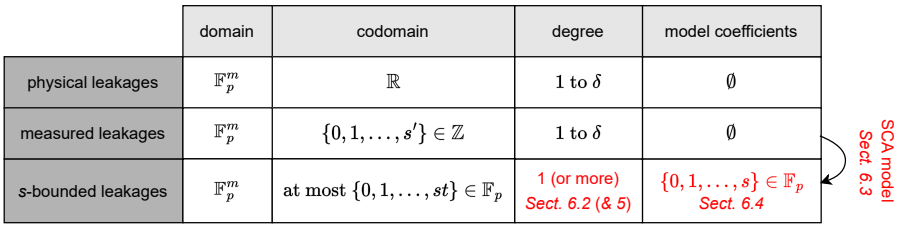


Figure 4.13: Adversarial strategy & leakage assumptions.

corresponding number of corresponding preimages is therefore at most $(|\operatorname{Im}(f')|-1)B$, which allows us to conclude on $p_{f',B}$. For the particular case where f' is a direct sum of m p-ary functions from C_1^s, since $mst < p$ we have that $f(y) \in [0, st]$ since $f \in \mathsf{C}_1^s$ and $f'(\boldsymbol{y}) \in [0, mst]$, hence $|\operatorname{Im}(f')| \leq mst + 1$. □

□

Using the same parameters as in Section 4.3.4, namely $p = 2^{31}-1$, $s = 2^{12}$, $n = 4$ and $m = 4$, we obtain that preimage sizes $|\mathcal{A}_{f'}(z)|$ larger than $\sqrt[5]{5!2^{124}} \approx 2^{26}$ lead to a time complexity larger than 2^{124}. We can then take the value $B = 2^{26}$ and apply Proposition 9, leading to a probability to get a sample z with at most B preimages lower than $\frac{4 \cdot 2^{12} \cdot 31 \cdot 2^{26}}{(2^{31}-1)^4)} \approx 2^{-79}$. Hence, our analysis shows that attacks exploiting worst-case leakages cannot succeed for adversaries with a data complexity bounded by 2^{79}, whereas side-channel security generally requires preventing attacks with data complexity in the 2^{50} range. As a result, we conclude that such attacks are not a concrete threat to the proposed instance.

4.3.5 Empirical validation

The previous sections showed that the LWPR assumption can remain secure in the generalized context of linear leakage functions. Yet, as illustrated in Figure 4.13, it is based on an adversarial strategy that interprets the leakages in $\mathbb{F}_p$ and security against adversaries following this strategy relies on two assumptions. Namely, it requires a bound on the degree of the leakage function and another bound on the quantization of the coefficients of this leakage function.

In the following, we therefore evaluate the validity of our assumptions based on a prototype hardware implementation (in Section 4.3.5 and Section 4.3.5) and discuss the applicability and relevance of our attack strategy (in Section 4.3.5). We conclude in Section 4.3.5, by confirming that the concrete leakage models we estimated from real measurements

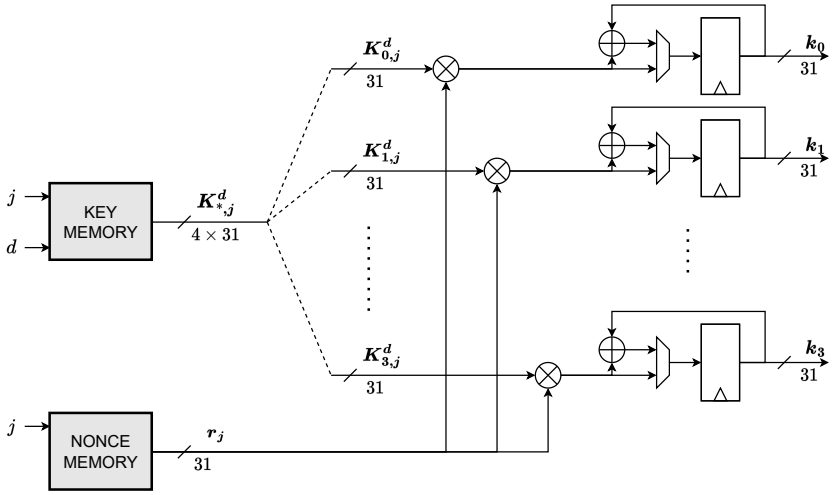


Figure 4.14: Masked LWPR-based re-keying: prototype hardware implementation.

are covered by our security analysis. Beforehand, we describe the prototype implementation we considered in Section 4.3.5

Prototype hardware implementation

We measured a parallel hardware implementation operating on 31-bit words running on a FPGA. The architecture of the computational core of our masked LWPR-based re-keying scheme is depicted in Figure 4.14. Concretely, the complete computation is performed by processing each share sequentially and the matrix multiplication for the share index d is split in $(N+1) \times M = (4+1) \times 4 = 20$ modular multiplications over $\mathbb{F}_p$. More precisely, the core implements 4 modular multiplications in parallel (i.e., the results of the multiplications between the j-th column of a key share denoted $K^d_{*,0 \leq j < 5}$ and the j-th element of the nonce denoted r_j, where $r_4 = 1$). We used the efficient modular multiplication architecture based on DSPs proposed in [Kop+17] for this purpose. The results obtained for the multiplications are then added in 4 accumulators (i.e., one for each row) dedicated to the reconstruction of the resulting fresh key.

As shown in Figure 4.15, the architecture of the key memory is designed to limit the impact of physical defaults on the implementation's security. In particular, the key shares are stored in independent physical memory units (i.e., BRAMs in the context of our FPGA implementation) and a register barrier is used before selecting the appropriate

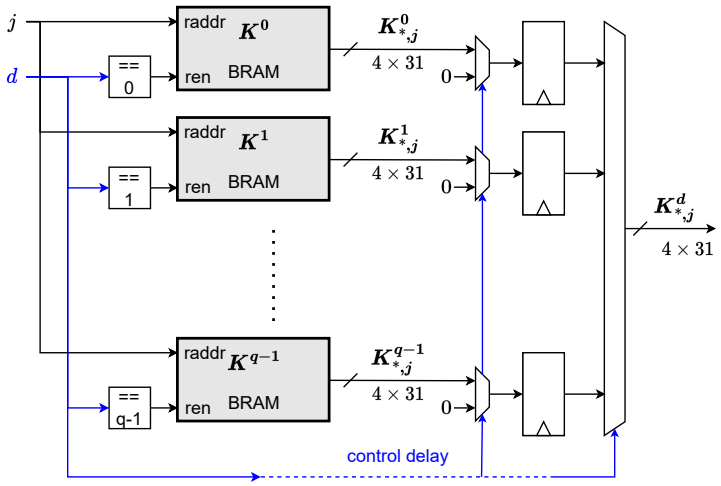


Figure 4.15: Key material memory architecture.

memory output. The latter is needed to avoid shares' recombinations that may be caused by glitches. Finally, a dedicated multiplexer at the input of the register barrier is used to drive the shares' values that are not currently processed to zero. Together with the fact that the shares are processed sequentially, this mechanism ensures the independence of the key shares.

Overall, our prototype implementation matches the parallelism requirements of our security analysis since it recombines a full (124-bit) fresh key in a single clock cycle, the leakage of which we analyze in the rest of this section. Besides, we note that when analyzing the second (red) attack path described in the threat model of Figure 4.12, we are only interested in the value of the recombined fresh key.[11] Concretely, we therefore choose to measure an implementation with a single share (i.e., $q = 1$). This gives the most favorable leakages to the evaluator and therefore puts us as close as possible to the noise-free setting considered in our analyzes. This is because the implementation with $q = 1$ is the smallest one and a limited use of resources in turn reduces the noise in the leakages.

Our measurements were performed on the Sakura-G evaluation board, which embeds a XC6SLX75-2CSG484C Spartan6 FPGA. The traces were collected using a Picoscope 5244D with a Tektronix CT-1 current probe, following the guidelines given in [BUS21]. In order to take into

[11] The first attack path (leveraging the leakages of the shared multiplications) was analyzed in [Duv+21] and the arguments of this previous work apply similarly.

account the effect of congestion that can take place during the place-and-route step of a dense design, we performed three different syntheses using the ISE14.7 design toolchain. The first one is unconstrained. The second one is constrained in order to achieve 95% of resources' utilization density (using a PBLOCK area constraint). The last one further constraints the design by (artificially) enforcing that higher-capacity routing elements are included in the path of some bits. We next refer to these different syntheses as unconstrained, constrained and amplified. The layout of the unconstrained and constrained syntheses are illustrated in Figure 4.16.

Bounded degree of the leakage function

The first (and main) physical assumption used to rule out algebraic attacks in our previous analysis is that the leakage function has a bounded degree. More precisely, we claim security for linear leakage functions in Sections 4.3.3 and 4.3.4 and will generalize to higher-order (e.g., quadratic) functions in Section 4.3.6.

We next study whether this assumption can be reasonably fulfilled in practice. As detailed in Section 4.3, regression-based models are a natural tool for this purpose, since they can be estimated for different, more or less complex, basis functions. A simple case, considered in the previous sections, is to take the bits of the target intermediate value as basis functions. But higher-degree models can be built, culminating with the exhaustive model which corresponds to a (worst-case) profiling where all the mean leakage values are exhaustively estimated [CRR02]. Our goal is to therefore show that a simple linear model does not lead to significant information losses compared to the exhaustive one.

For this purpose, we analyzed the 4 leakage functions f_i of our target implementation. We analyzed them independently to reduce the noise. They all gave similar results. For the exhaustive model, and since building 2^{31} templates is computationally intensive, we rather estimated a subset of n_a average leakage traces (with $n_a = 10,000$), and stored them in a vector of average leakage traces $\bar{\boldsymbol{l}} = [\overline{l^1}, \overline{l^2}, \ldots, \overline{l^{n_a}}]$. We then computed the vectors corresponding to our regression-based linear models for the three different syntheses, evaluated on the same values. In each case, we first estimated the 31-element vector of coefficients $\hat{\boldsymbol{a}}$ using $n_p = 10,000$ profiling traces (meaning $\approx \frac{10,000}{31}$ traces per coefficient, which was sufficient for good estimation given the low noise level of our measurements). As a result, we obtained model predictions:

$$\boldsymbol{m}^u_{\mathsf{r1}} = [\mathsf{M}^u_{\mathsf{r1}}(\boldsymbol{v}^1), \mathsf{M}^u_{\mathsf{r1}}(\boldsymbol{v}^2), \ldots, \mathsf{M}^u_{\mathsf{r1}}(\boldsymbol{v}^{n_a})],$$

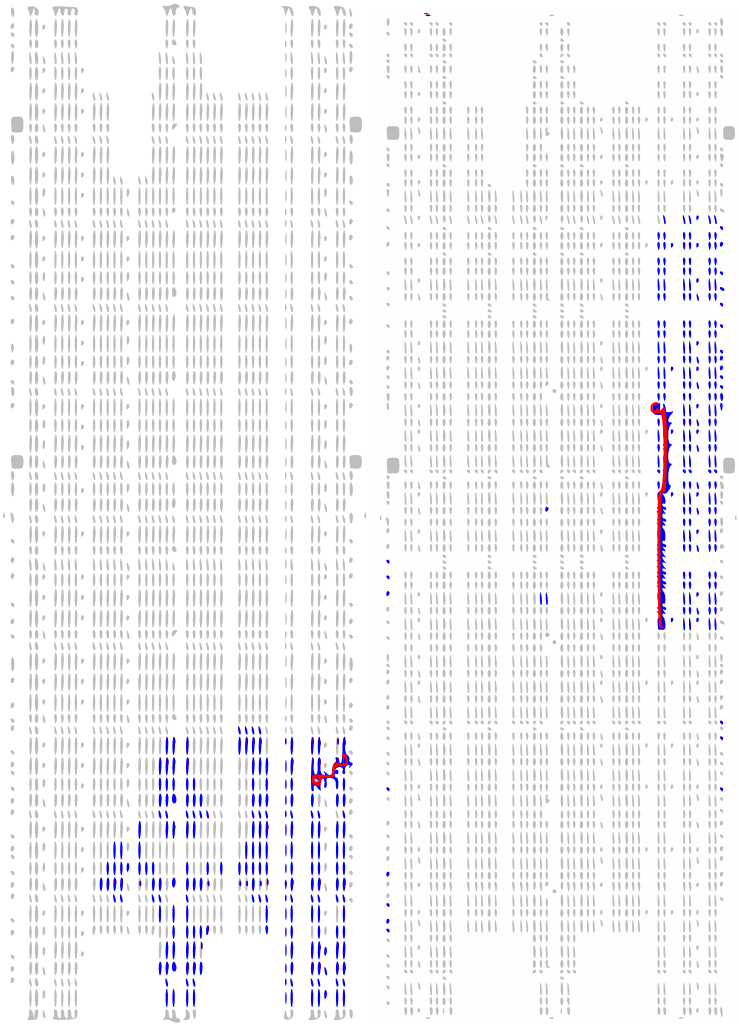


Figure 4.16: Layout of unconstrained (left) and constrained (right) syntheses. Unused resources are in white. Used resources are in blue. Routing is in red.

for the unconstrained synthesis, and similarly $\boldsymbol{m}^c_{\mathsf{r}1}$ & $\boldsymbol{m}^a_{\mathsf{r}1}$ for the constrained and amplified syntheses. Viewing these vectors as the samples of random variables $\overline{L}$, $M^u_{\mathsf{r}1}$, $M^c_{\mathsf{r}1}$ and $M^a_{\mathsf{r}1}$, we finally computed the correlation between the true (averaged) leakages and the models: $\hat{\rho}(M^u_{\mathsf{r}1}, \overline{L})$, $\hat{\rho}(M^c_{\mathsf{r}1}, \overline{L})$ and $\hat{\rho}(M^a_{\mathsf{r}1}, \overline{L})$.

These correlation coefficients are reported in Figure 4.17 in function of the number of traces used to obtain the average traces $\overline{l^i}$. We additionally report the estimates obtained when correlating the real leakage vec-

tors with a Hamming weight leakage model, and each curve in the figure is accompanied by a 95% bootstrap confidence interval. These results confirm that unconstrained syntheses (at the top of the figure) lead to high correlations already with the Hamming weight leakage model. By contrast, for constrained and amplified syntheses reflecting congestioned designs (at the middle and bottom of the figure) the correlation with the Hamming weight leakage model decreases while it remains close to one for the linear models. These experiments show that the generalization we propose indeed captures practically-relevant implementations. Admittedly, reaching such high correlation values does not preclude that higher-degree terms can be characterized and exploited. But it shows that most of the information leaked by our target implementation can be extracted with a simple linear model. So combined with (*i*) the generalization of Section 4.3.6 showing that security does not collapse as long as the leakage function remains sufficiently non-injective, and (*ii*) the fact

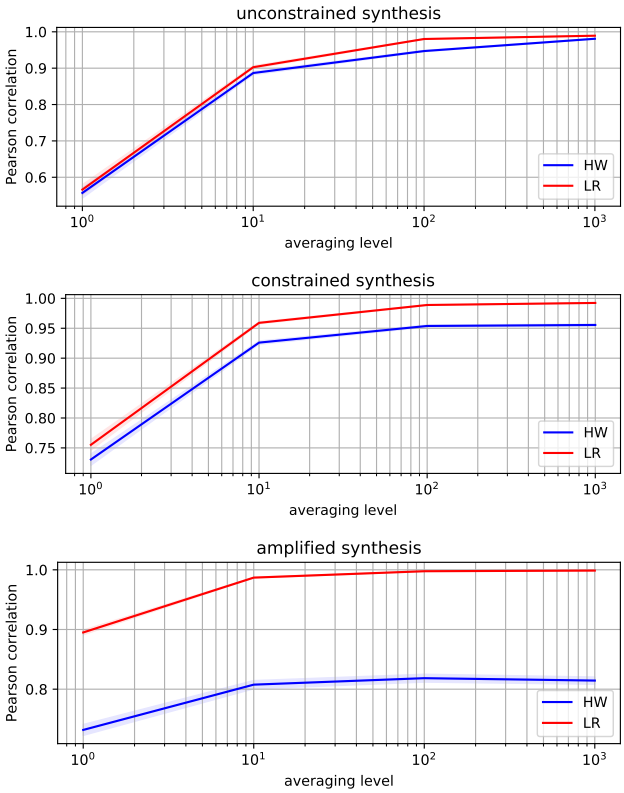


Figure 4.17: Correlation between real traces with different levels of averaging (on the X axis) and Hamming weight vs. regression-based linear models.

that our modeling is conservative (since it assumes noise-free leakages), we conjecture that the bounded-degree assumption we require can be matched sufficiently well to provide practically secure LWPR instances.

Practical application of the attack

The attacks we study in this paper work by interpreting physical leakages as functions in $\mathbb{F}_p$. So far, we showed that such attacks are hard if the leakage function has a bounded degree and its coefficients have limited accuracy. We also assessed empirically that the bounded degree assumption can be reasonably matched in practice. We now describe how to mount these attacks thanks to standard side-channel analysis tools, which will incidentally lead us to provide good indications that limiting the coefficient's accuracy is also reasonable.

Precisely, and as illustrated in Figure 4.13, we show how to move from measured leakages to s-bounded leakages thanks to the linear regression-based model of Section 4.3. For this purpose, we re-use the 31-element vectors of coefficients $\hat{\boldsymbol{a}}$ estimated using $n_p = 10,000$ profiling traces, for each of our three syntheses. We next observe that the value of Pearson's correlation is not affected if one of its variables is multiplied by a constant value. As a result, the following simple discretization process for the coefficients $\hat{\boldsymbol{a}}$ can be used:

$$\hat{\boldsymbol{a}}_{\mathsf{d}} = \left\lceil \hat{\boldsymbol{a}} \cdot \frac{s}{\mathsf{max}(\hat{\boldsymbol{a}})} \right\rceil .$$

We finally illustrate the coefficients of the linear models estimated from the unconstrained synthesis measurements together with the coefficients of the corresponding s-bounded functions in Figure 4.18. We can observe that already for $s = 2^8$ the matching between both is quite accurate. We further estimated the correlation $\hat{\rho}(M^s_{\mathsf{r}1}, M_{\mathsf{r}1})$ which was worth $1-\phi$ with $\phi < 10^{-6}$ for $s = 2^8$.[12] We conclude that a quantization corresponding to $s = 2^{12}$ offers a sufficient granularity to discretize our linear leakage models nearly perfectly. In other words, while we cannot rule out that other strategies than interpreting the leakages in $\mathbb{F}_p$ as we consider can lead to better results, this section shows that if this is the best strategy, then the move from measured leakages to s-bounded ones does not imply a significant information loss for the values of s (e.g., 2^{12}) that our theoretical investigations tolerate. The results for the quantization of the models for moderately constrained and amplified syntheses

[12] With a relative error of below 1%, which is easy to reach since the estimation is performed from modeled samples (rather than measured ones in Section 4.3.5).

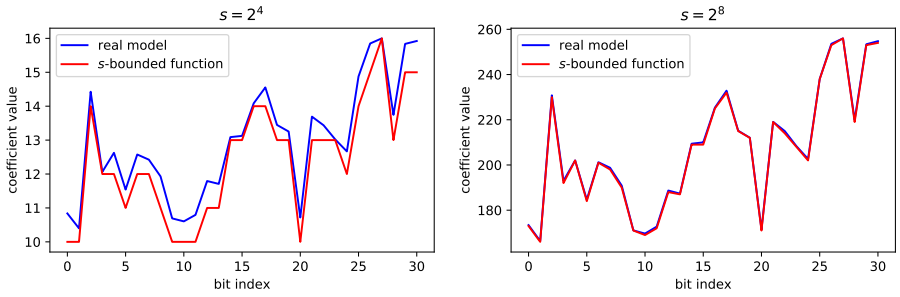


Figure 4.18: Discretization of the linear regression models (unconstrained synth.).

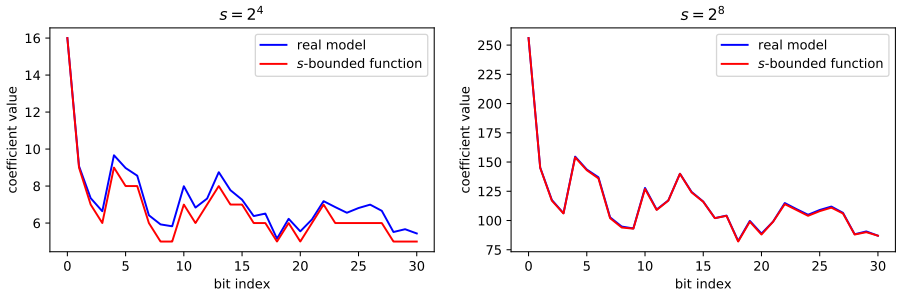


Figure 4.19: Discretization of the linear regression models (constrained synthesis).

are similar. We illustrate their respective coefficients in Figures 4.19 and 4.20 for completeness.

Bounded measurement accuracy

The previous sections suggested that using linear regression-based models is sufficient to accurately capture concretely-relevant leakage functions and that increasing the coefficient accuracy of the s-bounded pseudo-linear functions beyond 2^{12} does not lead to significant improvements of their informativeness in our case study. Quite naturally, such assessments remain limited by their heuristic nature. For the analysis of the degree, this is unavoidable (since absence of evidence is not evidence of absence) and the only option to make it more robust is to consider higher-degree functions that could theoretically show up, which we will do in Section 4.3.6. But for the bounded coefficient accuracy, there is a complementary argument that could be given. For this purpose, the starting observation is that physical leakages are measured by converting an analog signal into a digital one. Concretely, the "vertical resolution" s' of modern oscilloscopes typically ranges from 8 to 12 bits, meaning

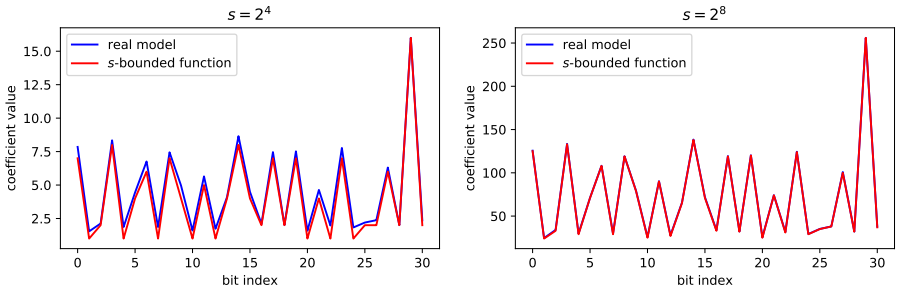


Figure 4.20: Discretization of the linear regression models (amplified synthesis).

that the measured leakage can take between 2^8 and 2^{12} values (possibly a bit more thanks to oversampling, at the cost of a reduced "horizontal" resolution). So if the increase of the s parameter from Definition 21 leads to a pseudo-linear function having a codomain with more than s' elements, it implies that the s-bounded leakage function is more informative than the measured one, which contradicts the data processing inequality. So for such leakage functions, the bounded coefficient accuracy can be directly connected to an assumption on the adversary's measurement apparatus.

However, increasing the s parameter does not always increase the cardinality of the leakage functions (e.g., think about the case where $t = 1$ for an increasing s). So one cannot directly bound the coefficient accuracy s based on the resolution of the oscilloscope s'. Yet, we note that most of the security claims in Sections 4.3.3 (resp., 4.3.4) depend on the product ts (resp., mts) which bounds the cardinality of the functions' co-domain and where t corresponds to the $\binom{t}{1}$ basis functions of a linear leakage model. So if the security analysis could be improved in order to rely only on this cardinality of the leakage functions, independent of the value of s (i.e., getting rid of the $ts < p$ and $mts < p$ conditions), we could replace the assumption on the bounded coefficient accuracy by an assumption on the resolution of the oscilloscope used to mount the attack. We leave the investigation of such an improved analysis as an interesting open problem.

Concrete security level

The three aforementioned implementations gave us three sets of real parameters, namely $\hat{\boldsymbol{a}}^u, \hat{\boldsymbol{a}}^c$ and $\hat{\boldsymbol{a}}^a$. Each of them can be discretized using the method of Section 4.3.5, resulting in three sets of $\mathbb{F}_p$ coef-

ficients $\boldsymbol{a}_{\mathsf{d}}^{u}, \boldsymbol{a}_{\mathsf{d}}^{c}$ and $\boldsymbol{a}_{\mathsf{d}}^{a}$, themselves associated to an s-bounded pseudo-linear function $\mathsf{F}_{\boldsymbol{a}_{\mathsf{d}}^{u}}, \mathsf{F}_{\boldsymbol{a}_{\mathsf{d}}^{c}}$ and $\mathsf{F}_{\boldsymbol{a}_{\mathsf{d}}^{a}}$.

	$v_{\mathsf{F}_{\boldsymbol{a}_{\mathsf{d}}^{*}}}$	$w_{\mathsf{F}_{\boldsymbol{a}_{\mathsf{d}}^{*}}}$	$\mathsf{deg}(\mathsf{F}_{\boldsymbol{a}_{\mathsf{d}}^{*}})$	$\mathsf{nl}(\mathsf{F}_{\boldsymbol{a}_{\mathsf{d}}^{*}})$
Unconstrained	101748	1073657388	≥ 101748	≥ 2147381899
Constrained	213852	1073692455	≥ 213852	≥ 2147269795
Amplified	160593	1073709374	≥ 160593	≥ 2147323054

Table 4.1: $\mathsf{deg}(\mathsf{F}_{\boldsymbol{a}_{\mathsf{d}}^{*}})$ and $\mathsf{nl}(\mathsf{F}_{\boldsymbol{a}_{\mathsf{d}}^{*}})$ bounds of s-bounded pseudo-linear functions for the different implementations (computed using Proposition 8).

The results of Table 4.1 show the very high values of the degree and nonlinearity reached by these s-bounded pseudo-linear functions. As a result, algebraic attacks appear to be impracticable against the corresponding LWPR instances. The time complexities being several order of magnitude larger than the 124 bits of security we target, even more advanced approach (e.g., solving algebraic systems with Gröbner bases, using code-based techniques to approximate by smaller-degree functions) should not lead to successful attacks.

4.3.6 Further generalizations

The previous section confirmed the security that can be offered by concrete LWPR instances. It also showed that linear models are good abstractions to capture the features of actual leakage measurements. In this section, we show that the positive results obtained for s-bounded pseudo-linear functions extend naturally to higher-order leakage models, taking the quadratic case as an example. Despite not directly motivated by practice (as linear leakage models offered high level of correlation with real leakages in the previous section), we deem this result important to confirm that even in the hypothetical presence of such higher-order dependencies, the security of LWPR would not collapse.

Definition 23 (s-bounded pseudo-quadratic functions)**.** *We denote* $\mathsf{Q}_{\boldsymbol{a},\boldsymbol{b}}$ *the function defined by a vector* $\boldsymbol{a} \in \mathbb{F}_p^t$ *and a stricly upper triangular matrix* $\boldsymbol{b} \in \mathbb{F}_p^{t \times t}$*, where* $t = \lceil \log p \rceil$*, as:*

$$\begin{aligned} \mathsf{Q}_{\boldsymbol{a},\boldsymbol{b}} :& \mathbb{F}_p \to \mathbb{F}_p \\ & y \mapsto \sum_{i=0}^{t-1} a_i \mathsf{g}(y)_i + \sum_{0 \leq i < j < t} b_{i(t+1)+j} \mathsf{g}(y)_i \mathsf{g}(y)_j. \end{aligned}$$

We call $\mathsf{Q}_{\boldsymbol{a},\boldsymbol{b}}$ *s-bounded pseudo-quadratic when each* $a_i, b_{i,j}$ *belongs to* $[0, s]$*, with security parameter* $s \in \mathbb{F}_p$ C_2^s *the class of s-bounded pseudo-quadratic functions.*

We give the following result for attacks exploiting any-case leakages.

Proposition 10 (Properties of s-bounded pseudo-quadratic functions)**.** *Let* $f \in \mathsf{C}_2^s$ *with* $\frac{t(t+1)}{2}s < p$ *and* $t = \lceil \log p \rceil$*, then the following holds:*

- $\mathsf{v}_f \geq \left\lceil \frac{p}{st(t+1)/2+1} \right\rceil$,
- $\mathsf{w}_f \geq p - st(t+1)/2 - 1$.

And assuming $\mathsf{v}_f \neq p$*, we further have:*

- $\mathsf{deg}(f) \geq \left\lceil \frac{p}{st(t+1)/2+1} \right\rceil$,
- $\mathsf{nl}(f) \geq \min\left(p - \mathsf{v}_f, \max\left(\left\lceil \frac{p}{st(t+1)/2+1} \right\rceil - 1, p - st(t+1)/2 - 1\right)\right)$.

Proof. Since $\frac{t(t+1)}{2}s < p$, we obtain that for all $x \in \mathbb{F}_p$, $0 \leq f(x) \leq \frac{t(t+1)}{2}s$, hence $\mathsf{w}_f \geq p - \frac{t(t+1)}{2}s - 1$. Thereafter, $f(x)$ can take at most $\frac{t(t+1)}{2}s+1$ values, hence there exists at least a preimage set with cardinal $\left\lceil \frac{p}{st(t+1)/2+1} \right\rceil$ or more. Finally, the last items are obtained by using Proposition 6 and Proposition 7 respectively, combined with the two first items. □ □

Note that the condition $ts < p$ which avoids the reduction when computing the leakage function in the analysis of the linear case is replaced by a condition $\frac{t(t+1)}{2}s < p$, where $\frac{t(t+1)}{2} = \binom{t}{1} + \binom{t}{2}$ is the number of basis functions in the quadratic model. This last result shows that s-bounded pseudo-quadratic functions behave similarly to s-bounded pseudo-linear functions. The only difference is that the security parameter s is a bit more constrained, so that $\frac{t(t+1)}{2}s < p$. Yet, with $p = 2^{31} - 1$ and $t = \lceil \log p \rceil$, security remains guaranteed for $s = 2^{12}$ (with a large security margin). In a similar manner, it could be shown that the estimated cost of algebraic attacks against LWPR is maintained for larger degrees, if the leakage function it relies on remains sufficiently non-injective, with a constraint on s evolving with the number of basis functions in the model.

As for attacks using worst-case leakages as studied in Section 4.3.4, the condition is slightly more restricted since we need $m\frac{t(t+1)}{2}s < p$ in order to apply Proposition 9. It nevertheless remains sufficient in the

quadratic case. Besides, it could again be that an improved analysis leads to relexed conditions (or tighter bounds), which we leave as an interesting scope for further research.

4.3.7 Conclusions and open problems

By showing that the LWPR assumption remains secure for a wide class of practically-relevant leakage functions, this work makes an important step in improving its usability. As a result, it strengthens the motivation for using re-keying schemes operating in prime fields, since they provide significant security improvements over their binary counterparts. Concretely, our investigations suggest that parallel instances of LWPR have good potential to be secure. The generalization of Section 4.3.6 further shows that increasing the degree of the leakage function is not a direct threat, so that the main assumption for LWPR to be hard is the quite natural requirement that the leakages are sufficiently non-injective.

Our results open several interesting avenues for further research. First, Section 4.3.2 recalled the challenge of securing serial LWPR implementations against leakage and suggested shuffling as a possible option for this purpose. Its concrete investigation is therefore a natural next step. Second, our analyzes could be improved in order to rely only on the degree of the leakage functions and the cardinality of their co-domain. As discussed in Section 4.3.5, this could lead to formally connect the bouned coefficient accuracy hypothesis with the resolution of the adversary's measurement apparatus. Third, we for now consider univariate leakages. They reasonably match the hardware implementation context we evaluated, where it is possible to implement the un-masking of the ephemeral key in a single cycle. But multivariate generalizations are an important scope for further investigations and will be especially relevant in a software implementation context where (for example) the reduction that has to take place after the un-masking may take several cycles. Dealing with more leaky software implementations (even with shuffling) may also motivate stepping back to a less conservative model, where security is argued based on the difficulty for the adversary to infer leakage values without errors, possibly formalized by a Learning With Physical Rounding and Errors problem. LWPR being an aggressive and new physical assumption, improving the understanding of the best cryptanalysis techniques to break it naturally remains needed as well. For example, our current attacks do not exploit key guessing strategies (which should at least imply a slight increase of the security parameters). Eventually, it would be worth studying the conditions upon which hard

physical learning problems can be connected and reduced to standard (mathematical) hard learning problems.

4.4 Towards a generalized approach for learning problems reductions

Solving approximate systems of linear equations has been considered useful computationally hard problem [Kea93; Hås97] to build cryptographic schemes. The most elementary of these problems is probably Learning Parity with Noise (LPN), which operates on binary values. This problem is the core of various protocols and cryptographic tools [Pie12]. Nowadays, many lattice-based post-quantum protocols are built on learning problem variants. At first, the unstructured problems Learning With Errors (LWE) [Reg05b], using a Gaussian error, and Learning With Rounding (LWR) [BPR12], using deterministic rounding, were most common. In recent lattice-based schemes such as [Duc+18; Bas+19], they are often replaced by their structured variants (Ring-LWE, Module-LWR , etc.,) for efficiency reasons. Learning problems are also the core of the first Fully Homomorphic Encryption (FHE) scheme [Gen09], and most of the current [Chi+20b; Che+17] FHE schemes.

Some approaches also use more unconventional variations of learning problems. A common usage is to sample keys from small spaces (e.g., binary keys [MP13; Bra+13], sparse keys [JLS24]). FHE also introduces new learning problems, as some manipulated ciphertexts follow learning problems distribution with Gaussian mixture as the error. Also, the Hint-LWE [Dac+20] approach introduces a high number of learning problem variants, initially to security in the context of study side-channel information.

In the context of leakage studies, we are starting to see various exotic learning problems emerging "in the wild." For instance, Bellizia et al. [Bel+21] observed a variant of LPN with asymmetric error while exploring the use of inexact computing for more efficient and secure hardware implementations. Another example is the Learning With Physical Rounding (LWPR) problem [Duv+21], where the leakage of the inner product is treated as a rounding function. LWPR has potential applications in creating rekeying operations, adding an extra layer to existing protocols, and also surfaces in the analysis of leakage in current PQC schemes.

There are many reductions between the different variants of learning problems, as mentioned in Chapter 1. "In the wild" Learning problems are approached in different ways: Bellizia et al. managed to

provide a complete reduction of the LPN-OD problem they introduce ([Bel+21], Section 4), while the analysis of LWPR's difficulty remains heuristic [Hof+23]. Nevertheless, the wide range of learning problems motivates a more generalized approach.

Therefore, the first contribution of this work is the introduction of a new learning problem, named Learning With Error with Output Dependencies (LWE-OD) in Definition 25. Similarly to LWE, the LWE-OD problem consists of solving an approximate linear system over $\mathbb{F}_q$. The key difference is that in LWE-OD, the error distribution can vary based on the value of the inner product. An LWE-OD instance is defined by q error distributions, denoted $\varphi_0, \ldots, \varphi_{q-1}$, and LWE-OD samples take the form $(\boldsymbol{a}, \langle \boldsymbol{a}, \boldsymbol{k} \rangle + e)$ where e is sampled according to $\varphi_{\langle \boldsymbol{a}, \boldsymbol{k} \rangle}$. LWE-OD instances encompass LWE, LWR, LWPR, and most other learning problems operating on $\mathbb{F}_q$. Indeed, an LWE-OD instance where all the error distributions are equal ($\varphi_0 = \cdots = \varphi_{q-1} = \psi$) is equivalent to LWE with error distribution ψ. LWE-OD with deterministic error distributions that simulate rounding corresponds exactly to LWR (more details are given in Section 4.4.1).

The second contribution is a reduction from certain instances of LWE-OD to LWE in Theorem 4. More specifically, we provide conditions on the error distributions of both LWE-OD and LWE for the reduction to apply. While those conditions do not enable to find LWE parameters from every interesting LWE-OD instance, we can find new proofs of previously known results, namely [AA16] reduction from LWR to LWE and [Bel+21] reduction from LPN-OD to LPN.

Notations and definitions

For the rest of the paper, $n \in \mathbb{N}$ is a dimension parameter, $q \in \mathbb{N}$ is a modulus polynomial in n. $\mathbb{F}_q$-distributions refer to probability distributions over F_q. Let φ be a $\mathbb{F}_q$-distribution, $X \leftarrow \varphi$, $\varphi(i)$ denotes the probability $\Pr[X = i]$. For φ, χ $\mathbb{F}_q$-distributions, and X, Y random variables such that $X \leftarrow \varphi$, and $Y \leftarrow \varphi$, we denote by $\varphi + \chi$ the $\mathbb{F}_q$-distribution followed by $X + Y$.

Definition 24 (Walsh transform)**.** *Let ω be a q-th root of unity. For any $\mathbb{F}_q$ distribution φ, we define its Walsh transform $\widehat{\varphi}$ as*

$$\begin{aligned}\widehat{\varphi}:&\mathbb{Z}\to\mathbb{C}\\ &i\mapsto\sum_{k=0}^{q-1}\omega^{ik}\varphi(k)\end{aligned}$$

We give definitions of the distributions, associated with learning problems, used specifically in this Section.

Definition 25 (LWE-OD distribution)**.** *Let $n\in\mathbb{N}^*, q\in\mathcal{P}, \boldsymbol{k}\in\mathbb{F}_q^n\setminus\{0\}, \varphi_0,\dots,\varphi_{q-1}$ be probability distributions over $\mathbb{F}_q$. We denote as the LWE-OD$^n_{\varphi_0,\cdots,\varphi_{q-1}}(\boldsymbol{k})$ distribution the probability distribution over $\mathbb{F}_q^n\times\mathbb{F}_q$ such that:*

$$\textit{LWE-OD}^n_{\varphi_0,\cdots,\varphi_{q-1}}(\boldsymbol{k})=\left\{(\boldsymbol{a},\langle\boldsymbol{a},\boldsymbol{k}\rangle+e)\mid \boldsymbol{a}\text{ such that }\langle\boldsymbol{a},\boldsymbol{k}\rangle\text{ is uniform}, e\leftarrow\varphi_{\langle\boldsymbol{a},\boldsymbol{k}\rangle}\right\}.$$

We denote a LWE-OD$^n_{\varphi_0,\cdots,\varphi_{q-1}}(\boldsymbol{k})$ sample any vector $(\boldsymbol{a},b)$ following the LWE-OD$^n_{\varphi_0,\cdots,\varphi_{q-1}}(\boldsymbol{k})$ distribution.

Let $\mathcal{O}^{\varphi_0,\dots,\varphi_{q-1}}_{n,\boldsymbol{k}}$ denote an oracle outputting independent samples according to the LWE-OD$^n_{\varphi_0,\cdots,\varphi_{q-1}}(\boldsymbol{k})$ distribution. The LWE-OD$^n_{\varphi_0,\cdots,\varphi_{q-1}}(\boldsymbol{k})$ problem is said to be (Q,t,m,θ)-hard to solve if for any algorithm A,

$$\Pr[\boldsymbol{k}\leftarrow\mathbb{F}_q^n: A^{\mathcal{O}^{\varphi_0,\dots,\varphi_{q-1}}_{n,\boldsymbol{k}}}(1^n)=\boldsymbol{k}]\leq\theta,$$

and A runs in time $< t$, with memory $< m$ and makes at most Q queries to $\mathcal{O}^{\varphi_0,\dots,\varphi_{q-1}}_{n,\boldsymbol{k}}$.

Finally, we introduce the q-LWE distribution. Intuitively, this distribution is the combination of q LWE ones with (potentially) different error distributions but under the same key. q-LWE$^n_{\chi_0,\cdots,\chi_{q-1}}(\boldsymbol{k})$ is at most as secure as the weakest LWE$^n_{\chi_i}(\boldsymbol{k})$, and is only used as a milestone towards a reduction to LWE in Section 4.4.2.

Definition 26 (q-LWE distribution)**.** *Let $n\in\mathbb{N}^*, q\in\mathcal{P}, \boldsymbol{k}\in\mathbb{F}_q^n\setminus\{0\}$, $\chi_0,\dots,\chi_{q-1}$ be $\mathbb{F}_q$ distributions. We denote as the q-LWE$^n_{\chi_0,\cdots,\chi_{q-1}}(\boldsymbol{k})$ distribution the probability distribution over $\left(\mathbb{F}_q^n\times\mathbb{F}_q\right)^q$ such that:*

$$q\textit{-LWE}^n_{\chi_0,\cdots,\chi_{q-1}}(\boldsymbol{k})=\left\{(\boldsymbol{a_i},\langle\boldsymbol{a_i},\boldsymbol{k}\rangle+e_i)_{0\leq i<q}\mid \boldsymbol{a_i}\leftarrow\mathbb{F}_q^n, e_i\leftarrow\chi_i\right\}.$$

We denote a q-LWE$^n_{\chi_0,\cdots,\chi_{q-1}}(\boldsymbol{k})$ sample any vector $(\boldsymbol{a_i},b_i)$ following the q-LWE$^n_{\chi_0,\cdots,\chi_{q-1}}(\boldsymbol{k})$ distribution.

4.4 Towards a generalized approach for learning problems reductions

Let $\mathcal{O}_{n,\boldsymbol{k}}^{\chi_0,\cdots,\chi_{q-1}}$ denote an oracle outputting independent samples according to the q-LWE$^n_{\chi_0,\cdots,\chi_{q-1}}(\boldsymbol{k})$ distribution. The q-LWE$^n_{\chi_0,\cdots,\chi_{q-1}}(\boldsymbol{k})$ problem is said to be (Q,t,m,θ)-hard to solve if for any algorithm A,

$$\Pr[\boldsymbol{k} \leftarrow \mathbb{F}_q^n : A^{\mathcal{O}_{n,\boldsymbol{k}}^{\chi_0,\cdots,\chi_{q-1}}}(1^n) = \boldsymbol{k}] \leq \theta,$$

and A runs in time $< t$, with memory $< m$ and makes at most Q queries to $\mathcal{O}_{n,\boldsymbol{k}}^{\chi_0,\cdots,\chi_{q-1}}$.

We now introduce algorithms that can be used to sample from LWE and the distribution previously introduced. Let $\psi, (\chi_i)_{0\leq i<q}, (\varphi_i)_{0\leq i<q}$ be random distributions over $\mathbb{F}_q$.

Algorithm 14 SampleLWE$^n_\psi(\boldsymbol{k})$

Output: A LWE$^n_\psi(\boldsymbol{k})$ sample $(\boldsymbol{a}, b)$.
$\boldsymbol{a} \leftarrow \mathbb{F}_q^n, e \leftarrow \psi$
$(\boldsymbol{a}, b) \leftarrow (\boldsymbol{a}, \langle \boldsymbol{a}, \boldsymbol{k}\rangle + e)$
return $(\boldsymbol{a}, b)$

Algorithm 15 SampleLWE-OD$^n_{\varphi_0,\cdots,\varphi_{q-1}}(\boldsymbol{k})$

Output: A LWE-OD$^n_{\varphi_0,\cdots,\varphi_{q-1}}(\boldsymbol{k})$ sample $(\boldsymbol{a}, b)$.
$\boldsymbol{a} \leftarrow \mathbb{F}_q^n, e \leftarrow \varphi_i$ where $i = \langle \boldsymbol{a}, \boldsymbol{k}\rangle$
$(\boldsymbol{a}, b) \leftarrow (\boldsymbol{a}, \langle \boldsymbol{a}, \boldsymbol{k}\rangle + e)$
return $(\boldsymbol{a}, b)$

Note that according to LWE-OD$^n_{\varphi_0,\cdots,\varphi_{q-1}}(\boldsymbol{k})$ definition, only $\langle \boldsymbol{a}, \boldsymbol{k}\rangle$ is required to be uniform. Sampling $\boldsymbol{a}$ uniformly ensures that.

Algorithm 16 Sample-q-LWE$^n_{\chi_0,\cdots,\chi_{q-1}}(\boldsymbol{k})$

Output: A q-LWE$^n_{\chi_0,\cdots,\chi_{q-1}}(\boldsymbol{k})$ sample $(\boldsymbol{a}, b)$.
for i in range(q) **do**
    $(\boldsymbol{a_i}, b_i) \leftarrow$ SampleLWE$^n_{\psi_i}(\boldsymbol{k})$
return $(\boldsymbol{a_i}, b_i)_{0\leq i<q}$

4.4.1 LWE-OD reduction to q-LWE

LWE-OD to q-LWE reduction

In this section, we give a condition under which the LWE-OD$^n_{\varphi_0,\cdots,\varphi_{q-1}}(\boldsymbol{k})$ search problem is at least as hard as the q-LWE$^n_{\chi_0,\cdots,\chi_{q-1}}(\boldsymbol{k})$ search prob-

lem. After giving a characterization of the LWE-OD$^n_{\varphi_0,\cdots,\varphi_{q-1}}(\boldsymbol{k})$ distribution, we show that it is possible to create LWE-OD$^n_{\varphi_0,\cdots,\varphi_{q-1}}(\boldsymbol{k})$ samples using a q-LWE$^n_{\chi_0,\cdots,\chi_{q-1}}(\boldsymbol{k})$ oracle. Finally, we use this sampler to establish a security reduction between q-LWE and LWE-OD.

First, we provide a characterization of the LWE-OD samples. This characterization can serve as an alternative definition and will be used in the proofs in the following section.

Proposition 11 (LWE-OD$^n_{\varphi_0,\cdots,\varphi_{q-1}}(\boldsymbol{k})$ distribution characterization)**.** *Let $q \in \mathbb{N}, n \in \mathbb{N}^*, \boldsymbol{k}^n, \varphi_0, \ldots, \varphi_{q-1}\mathbb{F}_q$-distributions. A sample $(\boldsymbol{a}, b)$ follows the LWE-OD$^n_{\varphi_0,\cdots,\varphi_{q-1}}(\boldsymbol{k})$ if and only if:*

$$\forall i,j \in \mathbb{F}_q, \begin{cases} Pr[b=j] = \frac{1}{q}\sum\limits_{i=0}^{q-1}\varphi_i(j-i) \\ Pr[\langle \boldsymbol{a},\boldsymbol{k}\rangle = i | b=j] = \frac{\varphi_i(j-i)}{\sum\limits_{k=0}^{q-1}\varphi_k(j-k)} \end{cases}$$

Proof. Let $(\boldsymbol{a}, b) \leftarrow$ LWE-OD$^n_{\varphi_0,\cdots,\varphi_{q-1}}(\boldsymbol{k})$, then:

$$\begin{aligned} \Pr[b=j] &= \Pr[\langle \boldsymbol{a},\boldsymbol{k}\rangle + e = j] \\ &= \sum_{i=0}^{q-1} \Pr[\langle \boldsymbol{a},\boldsymbol{k}\rangle + e = j | \langle \boldsymbol{a},\boldsymbol{k}\rangle = i] \cdot \Pr[\langle \boldsymbol{a},\boldsymbol{k}\rangle = i] \\ &= \frac{1}{q}\cdot\sum_{i=0}^{q-1} \Pr[\langle \boldsymbol{a},\boldsymbol{k}\rangle + e = j | \langle \boldsymbol{a},\boldsymbol{k}\rangle = i] \\ &= \frac{1}{q}\cdot\sum_{i=0}^{q-1} \varphi_i(j-i) \end{aligned}$$

$$\begin{aligned} \Pr[\langle \boldsymbol{a},\boldsymbol{k}\rangle = i | b=j] &= \frac{\Pr[\langle \boldsymbol{a},\boldsymbol{k}\rangle = i]}{\Pr[b=j]}\Pr[b=j | \langle \boldsymbol{a},\boldsymbol{k}\rangle = i] \\ &= \frac{\frac{1}{q}}{\frac{1}{q}\sum\limits_{k=0}^{q-1}\varphi_k(j-k)}\varphi_i(j-i) \\ &= \frac{\varphi_i(j-i)}{\sum\limits_{k=0}^{q-1}\varphi_k(j-k)} \end{aligned}$$

Reciprocally, let $(\boldsymbol{a}, b)$ such that $\forall i, j \in \mathbb{F}_q$, $\begin{cases} \Pr[b = j] = \frac{1}{q} \sum_{i=0}^{q-1} \varphi_i(j-i) \\ \Pr[\langle \boldsymbol{a}, \boldsymbol{k} \rangle = i | b = j] = \frac{\varphi_i(j-i)}{\sum_{k=0}^{q-1} \varphi_k(j-k)} \end{cases}$,

then:

$$\begin{aligned} \Pr[\langle \boldsymbol{a}, \boldsymbol{k} \rangle = i] &= \sum_{j=0}^{q-1} \Pr[b = j] \cdot \Pr[\langle \boldsymbol{a}, \boldsymbol{k} \rangle = i | b = j] \\ &= \frac{1}{q} \sum_{j=0}^{q-1} \varphi_i(j-i) \\ &= \frac{1}{q} \end{aligned}$$

$$\begin{aligned} \Pr[e = j - i | \langle \boldsymbol{a}, \boldsymbol{k} \rangle = i] &= \Pr[b = j | \langle \boldsymbol{a}, \boldsymbol{k} \rangle = i] \\ &= \sum_{i=0}^{q-1} \varphi_i(j-i) \cdot \frac{\varphi_i(j-i)}{\sum_{k=0}^{q-1} \varphi_k(j-k)} \\ &= \varphi_i(j-i) \end{aligned}$$

□

Note that, with output dependencies, the distribution of the last coefficient b is not necessarily uniform. While this allows to distinguish LWE-OD distributions from uniform ones, this does not immediatly translate to an attack on the search version of the problem.

We introduce an alternative sampler and perform an analysis-synthesis to prove its output follows a LWE-OD$^n_{\varphi_0, \cdots, \varphi_{q-1}}(\boldsymbol{k})$ distribution.

Proposition 12 (Algorithm 17 output characterization)**.** *Let $(\boldsymbol{a}, b) = (\boldsymbol{a}, \langle \boldsymbol{a}, \boldsymbol{k} \rangle + e)$ be the different from $\perp$ output of Algorithm 17. The following holds:*

$$\forall i, j \in \mathbb{F}_q \text{ such that } \rho(j) \neq 0, \begin{cases} Pr[b = j] = \rho(j) \\ Pr[\langle \boldsymbol{a}, \boldsymbol{k} \rangle = i | b = j] = \chi_j(j-i) \end{cases}.$$

Proof. b is sampled according to ρ at the beginning of Algorithm 17, and is not modified before being output.

$\forall i, j$ such that $\Pr[\rho(j)] = \Pr[b = j] \neq 0$, $\Pr[\langle \boldsymbol{a}, \boldsymbol{k} \rangle = i | b = j]$ is well defined and

$$\begin{aligned} \Pr[\langle \boldsymbol{a}, \boldsymbol{k} \rangle = i | b = j] &= \Pr[e = j - i | b = j] \\ &= \chi_j(j-i). \end{aligned}$$

□

Algorithm 17 AltSampleLWE-OD

Input: A sampling upper limit m, a $\mathbb{F}_q$-distribution ρ, a q-LWE$^n_{\chi_0,\cdots,\chi_{q-1}}(\boldsymbol{k})$ sampler
Output: A sample $(\boldsymbol{a}, b) \in \mathbb{F}_q^n \times \mathbb{F}_q$.

```
b ← ρ
cnt := 0
do
    (a, b') ← Sample-q-LWE^n_{χ_0,···,χ_{q-1}}(k)[b]   ▷ Only keep the b-th sample, which follows a LWE^n_{χ_b}(k) distribution
    cnt ← cnt +1
while b ≠ b' and cnt ≤ m
if cnt = m then
    return ⊥
return (a, b)
```

Proposition 13 (Algorithm 17 termination and complexity analysis)**.** *Let $n \in \mathbb{N}^*$, let $q \in \mathbb{N}$ and $m \in \mathbb{N}^*$ integers of polynomial size in n, let Sample-q-LWE a q-LWE$^n_{\chi_0,\cdots,\chi_{q-1}}(\boldsymbol{k})$ sampler (for any $\chi_0, \ldots, \chi_{q-1}$). Algorithm 17 with input q, m, Sample-q-LWE runs in polynomial time and returns $\perp$ with negligible probability.*

Proof. First, since the number of calls to Algorithm 16 is bounded by m and every other operation runs in constant time, for m polynomial in n the execution time is polynomial in n.

On each iteration of the loop in Algorithm 17, the probability of satisfying the exit condition is $\frac{1}{q}$ since b', output by Algorithm 16, is uniform. Denoting by X the number of iterations of this loop and since each call to Algorithm 16 is independent, X follows a geometric law of parameter $\frac{1}{q}$. Then, $\Pr[\perp] = \Pr[X > m] = \left(\frac{q-1}{q}\right)^{m-1}$, therefore negligible in n for a polynomial q. □

We now give the main theorem of the section, which gives a reduction of LWE-OD to q-LWE for given parameters. The high-level idea of its proof is showing that an adversary can use Algorithm 17 to turn q-LWE samples into LWE-OD ones in polynomial time. With this transformation possible, the existence of an attack against LWE-OD would enable an attack against q-LWE.

Theorem 2 (LWE-OD reduction to q-LWE)**.** *The LWE-OD$^n_{\varphi_0,\cdots,\varphi_{q-1}}(\boldsymbol{k})$ problem is at least as hard as the q-LWE$^n_{\chi_0,\cdots,\chi_{q-1}}(\boldsymbol{k})$ one, for $\chi_0, \ldots, \chi_{q-1}$*

4.4 Towards a generalized approach for learning problems reductions

such that:

$$\forall i,j \in \mathbb{F}_q \text{ such that } \sum_{k=0}^{q-1} \varphi_k(j-k) \neq 0, \chi_j(j-i) = \frac{\varphi_i(j-i)}{\sum_{k=0}^{q-1} \varphi_k(j-k)}.$$

Proof. Let $\varphi_0, \ldots, \varphi_{q-1}$ be $\mathbb{F}_q$-distributions, let $(\boldsymbol{a}, b)$ be the (non-$\perp$ with overwhelming probability) output of Algorithm 17 with input ρ such that $\forall j \in \mathbb{F}_q, \rho(j) = \frac{1}{q}\sum_{k=0}^{q-1} \varphi_k(j-k)$ and a q-LWE$^n_{\chi_0,\cdots,\chi_{q-1}}(\boldsymbol{k})$ sampler such that $\forall i,j \in \mathbb{F}_q$ such that $\rho(j) \neq 0$, $\chi_j(j-i) = \frac{\varphi_i(j-i)}{\sum_{k=0}^{q-1} \varphi_k(j-k)}$.

Then, using Proposition 12, we obtain the following:

$$\forall i,j \in \mathbb{F}_q \text{ such that } \rho(j) \neq 0, \begin{cases} \Pr[b=j] = \rho(j) = \frac{1}{q}\sum_{i=0}^{q-1} \varphi_i(j-i) \\ \Pr[\langle \boldsymbol{a}, \boldsymbol{k} \rangle = i | b = j] = \chi_j(j-i) = \frac{\varphi_i(j-i)}{\sum_{k=0}^{q-1} \varphi_k(j-k)} \end{cases}$$

Therefore, under the theorem assumptions, the output of Algorithm 17 $(\boldsymbol{a}, b)$ is a LWE-OD$^n_{\varphi_0,\cdots,\varphi_{q-1}}(\boldsymbol{k})$ sample. The algorithm can then be used to obtain LWE-OD$^n_{\varphi_0,\cdots,\varphi_{q-1}}(\boldsymbol{k})$ samples from q-LWE$^n_{\chi_0,\cdots,\chi_{q-1}}(\boldsymbol{k})$ oracle (both problems sharing the same key) in a polynomial time per Proposition 13. An adversary can then use any attack against the LWE-OD instance to break the q-LWE one, making the LWE-OD$^n_{\varphi_0,\cdots,\varphi_{q-1}}(\boldsymbol{k})$ problem at least as hard as the q-LWE$^n_{\chi_0,\cdots,\chi_{q-1}}(\boldsymbol{k})$ one. □

Note that the theorem conditions do not put constraints on the whole characteristic functions of the distributions χ_i, the remaining values are left free as long as the χ_i fulfill the distributions' properties.

Learning With Rounding (LWR) particular case

The LWR distribution can be seen as a particular case of LWE-OD distribution. Plugging this distribution into Theorem 2, we can make a reduction from LWR to a remarkable q-LWE, which itself directly reduces to LWE. By doing so, we get the same result as [Bog+16], Section 4, which provides a proof that relies on extremely similar ideas.

Definition 27 (LWR distribution)**.** *Let $n \in \mathbb{N}^*, q \in \mathbb{N}, p \in \mathbb{N}, \boldsymbol{k} \in \mathbb{F}_q^n \setminus \{0\}, \varphi_0, \ldots, \varphi_{q-1}$ be probability distributions over $\mathbb{F}_q$. We denote as*

the $LWR_p^n(\boldsymbol{k})$ *distribution the probability distribution over* $\mathbb{F}_q^n \times \mathbb{F}_q$ *such that:*

$$LWR_p^n(\boldsymbol{k}) = \left\{ \left(\boldsymbol{a}, \lfloor \langle \boldsymbol{a}, \boldsymbol{k} \rangle \rfloor_p = \lfloor (p/q) \cdot \langle \boldsymbol{a}, \boldsymbol{k} \rangle \rfloor \right) \mid \boldsymbol{a} \leftarrow \mathbb{F}_q^n \right\}.$$

We denote a $LWR_p^n(\boldsymbol{k})$ *sample any vector* $(\boldsymbol{a}, b)$ *following the* $LWR_p^n(\boldsymbol{k})$ *distribution.*

Let $\mathcal{O}_{n,\boldsymbol{k}}^p$ *denote an oracle outputting independent samples according to the* $LWR_p^n(\boldsymbol{k})$ *distribution. The* $LWR_p^n(\boldsymbol{k})$ *problem is said to be* (Q, t, m, θ)*-hard to solve if for any algorithm* A*,*

$$\Pr[\boldsymbol{k} \leftarrow \mathbb{F}_q^n : A^{\mathcal{O}_{n,\boldsymbol{k}}^p}(1^n) = \boldsymbol{k}] \leq \theta,$$

and A *runs in time* $< t$*, with memory* $< m$ *and makes at most* Q *queries to* $\mathcal{O}_{n,\boldsymbol{k}}^p$*.*

LWR is sometimes defined with a rounding approximation, and sometimes with a truncation (the formest being more efficient to implement as, for a power of 2 modulus, it consists of discarding the least significant bits). We choose the latter in this paper, all the results about this problem can be easily adapted to the other definition.

Remark 1 (LWR as a LWE-OD)**.** *The* $LWR_p^n(\boldsymbol{k})$ *distribution is a particular case of the* $LWE\text{-}OD_{\varphi_0,\cdots,\varphi_{q-1}}^n(\boldsymbol{k})$ *distribution, with* φ_i *such that:*

$$\forall i, j \in \mathbb{F}_q, \varphi_i(j) = \begin{cases} 1 \text{ if } q \mid (i \bmod \frac{q}{p}) + j \\ 0 \text{ else} \end{cases}$$

From here, we denote as $\text{LWE-OD}_{\varphi_0,\cdots,\varphi_{q-1}}^n(\boldsymbol{k})$ the LWE-OD distribution corresponding to a $\text{LWR}_p^n(\boldsymbol{k})$ one.

It is then possible to plug this LWE-OD distribution into Theorem 2 to get a reduction from LWR to q-LWE. The specific q-LWE distribution LWR reduces to have the same

Corollary 3. *With* $p \mid q$*, the* $LWR_p^n(\boldsymbol{k})$ *problem is at least as hard as the* $LWE_\psi^n(\boldsymbol{k})$ *one, for* $\psi = \mathcal{U}\left(\left]-\frac{q}{p}, 0\right]\right)$*.*

Proof. Using Theorem 2, we get that the $\text{LWR}_p^n(\boldsymbol{k})$ problem is at least as hard as the $q\text{-LWE}_{\chi_0,\cdots,\chi_{q-1}}^n(\boldsymbol{k})$ one, where $\forall i \in , \chi_i = \mathcal{U}\left(\left]-\frac{q}{p}, 0\right]\right)$. The other q-LWE error distributions are left condition-free by the theorem and can be anything.

LWR then reduces to a q-LWE problem whose all relevant distributions are the same. This $q\text{-LWE}_{\chi_0,\cdots,\chi_{q-1}}^n(\boldsymbol{k})$ samples can be easily simulated using $\frac{q}{p}$ calls to a $\text{LWE}_\psi^n(\boldsymbol{k})$ sampler, with $\psi = \mathcal{U}\left(\left]-\frac{q}{p}, 0\right]\right)$, thus giving the final reduction. □

Link to [Bog+16]. Our result is similar to the one in [Bog+16]. The more general approach results in an overhead in data complexity of $q \times \frac{q}{p}$ instead of $\frac{q}{p}$. Indeed, the fact that the last coefficient b of LWR samples is uniformly distributed, the algorithm can be adapted by not sampling b before the rejection loop, and accepting any sample with a b that belongs to the support of the LWR distribution, leading to p times less calls to the LWE sampler. Moreover, the q-LWE sampler can be directly replaced by a LWE one, leading to $\frac{q}{p}$ times fewer calls to the LWE sampler.

Explicitly, their sampling method corresponds to the following algorithm using our notations:

Algorithm 18 [Bog+16]LWR sampler

Input: A $\mathrm{LWE}_{\psi}^{n}(\boldsymbol{k})$ sampler
Output: A $\mathrm{LWR}_{p}^{n}(\boldsymbol{k})$ sample $(\boldsymbol{a}, b) \in \mathbb{F}_q^n \times \mathbb{F}_p$.
do
$(\boldsymbol{a}, b) \leftarrow \text{Sample-LWE}_{\psi}^{n}(\boldsymbol{k})$
while $\frac{q}{p} \nmid b$
return $(\boldsymbol{a}, b \times (p/q))$

4.4.2 q-LWE reduction to LWE

In this section, we show that the $q\text{-LWE}_{\chi_0,\cdots,\chi_{q-1}}^{n}(\boldsymbol{k})$ is at least as hard as $\mathrm{LWE}_{\psi}^{n}(\boldsymbol{k})$, under given conditions on the error distributions. This result is given in Theorem 3, and can then be combined with Theorem 2 to obtain the full reduction in Theorem 4.

The outline of the proof is quite similar to that in the previous section. We demonstrate that an adversary can convert $\mathrm{LWE}_{\psi}^{n}(\boldsymbol{k})$ samples into $q\text{-LWE}_{\chi_0,\cdots,\chi_{q-1}}^{n}(\boldsymbol{k})$ samples without knowing the key $\boldsymbol{k}$ and within polynomial time. This conversion enables any attack on $q\text{-LWE}_{\chi_0,\cdots,\chi_{q-1}}^{n}(\boldsymbol{k})$ to be applied to $\mathrm{LWE}_{\psi}^{n}(\boldsymbol{k})$, showing that the qLWE problem is at least as hard as the LWE problem.

We first give the sampler that uses the LWE sampler as a black-box, and give the conditions on the input distributions such that $\mathrm{LWE}_{\psi}^{n}(\boldsymbol{k})$ samples are indeed converted into $q\text{-LWE}_{\chi_0,\cdots,\chi_{q-1}}^{n}(\boldsymbol{k})$ ones.

Proposition 14 (Algorithm 19 output characterization)**.** *Let $\mathit{SampleLWE}_{\psi}^{n}(\boldsymbol{k}), \tau_0, \ldots, \tau_{q-1}$ be the input of Algorithm 19, then the output of the algorithm follows a $q\text{-}\mathit{LWE}_{\chi_0,\cdots,\chi_{q-1}}^{n}(\boldsymbol{k})$ distribution if and only if:*

Algorithm 19 AltSample-q-LWE

Input: $\mathbb{F}_q$-distributions $\tau_0, \ldots, \tau_{q-1}$, a $\mathrm{LWE}^n_\psi(\boldsymbol{k})$ sampler

Output: A sample $(\boldsymbol{a_i}, b_i)_{0 \leq i < q} \in \left(\mathbb{F}_q^n \times \mathbb{F}_q\right)^q$.

```
for i in range(q) do
    (a_i, b'_i) ← SampleLWE^n_ψ(k)
    e ← τ_i
    b_i ← b'_i + e
return (a_i, b_i)_{0≤i<q}
```

$$\forall i, j \in \mathbb{F}_q, \begin{cases} \tau_i(j) \in \mathbb{R} \text{ and } 0 \leq \tau_i(j) \leq 1 \\ \tau_i(j) = \sum_{k=0}^{q-1} \frac{\widehat{\chi_i}(k)}{\widehat{\psi}(k)} \omega^{-jk} \end{cases}$$

where $\omega = exp\left(\frac{2i\pi}{q}\right)$ is an (arbitrary) q-th root of unity and $\widehat{\chi_i}$ [resp. $\widehat{\psi}$] is the Walsh transform of χ_i [resp. ψ].

Proof. Adding two $\mathbb{F}_q$-distribution χ and τ gives a convolution product. Indeed, let $X \leftarrow \chi, Y \leftarrow \tau$:

$$\begin{aligned} \forall i \in \mathbb{F}_q, (\chi + \tau)(i) &= \Pr[X + Y = i] \\ &= \sum_{j \in \mathbb{F}_q} \Pr[X + Y = i | X = j] \cdot \Pr[X = j] \\ &= \sum_{j \in \mathbb{F}_q} \tau(i - j) \cdot \chi(j) \end{aligned}$$

We then show that $\widehat{\chi + \tau}(i) = \widehat{\chi}(i)\widehat{\tau}(i)$. Let $i \in F_q$:

$$\begin{aligned} \widehat{\chi + \tau}(i) &= \sum_{k \in \mathbb{F}_q} \omega^{ik} \left(\sum_{j \in \mathbb{F}_q} \chi(j)\tau(k - j) \right) \\ &= \sum_{j \in \mathbb{F}_q} \sum_{k \in \mathbb{F}_q} \omega^{ik} \chi(j)\tau(k - j) \\ &= \sum_{j \in \mathbb{F}_q} \sum_{k \in \mathbb{F}_q} \omega^{i(k+j)} \chi(j)\tau(k) \\ &= \sum_{k \in \mathbb{F}_q} \left(\omega^{ik} \tau(k) \sum_{j \in \mathbb{F}_q} \omega^{ij} \chi(j) \right) \\ &= \widehat{\chi}(i)\widehat{\tau}(i) \end{aligned}$$

We therefore have $\widehat{\tau(i)} = \frac{\widehat{\chi+\tau(i)}}{\widehat{\chi}(0)}$, which is well defined since, χ being a probability distribution, $\widehat{\chi}(k)$ is non null for all $k \in \mathbb{Z}$.

Moreover,

$$\begin{aligned}\widehat{\tau(0)} &= \frac{\widehat{\chi+\tau}(0)}{\widehat{\chi}}(0) \\ &= \frac{\sum\limits_{k\in\mathbb{F}_q}(\chi+\tau)(k)}{\sum\limits_{k\in\mathbb{F}_q}\chi(k)} \\ &= 1\end{aligned}$$

Hence $\sum\limits_{k\in\mathbb{F}_q}\tau(k) = 1$, and τ is a $\mathbb{F}_q$-distribution if and only if $\forall i \in \mathbb{F}_q, \tau(i) \in \mathbb{R}$ and $0 \leq \tau(i) \leq 1$.

τ can be obtained from its transform $\widehat{\tau}$ using the inversion formula $\tau(j) = \frac{1}{q}\sum\limits_{k\in\mathbb{F}_q}\widehat{\tau}(j)\omega^{-jk}$. Indeed:

$$\begin{aligned}\sum_{k\in\mathbb{F}_q}\widehat{\tau}(j)\omega^{-jk} &= \sum_{k\in\mathbb{F}_q}\sum_{l\in\mathbb{F}_q}\omega^{k(l-j)}\tau(l) \\ &= \sum_{l\in\mathbb{F}_q}\tau(l)\sum_{k\in\mathbb{F}_q}\omega^{k(l-j)} \\ &= q\tau(j)\end{aligned}$$

The last equation comes from the fact that
$\sum\limits_{k\in\mathbb{F}_q}\omega^k = \begin{cases} q \text{ if } k = 0 \\ 0 \text{ else (as the sum of a geometric sequence formula)}\end{cases}$

We can now apply these results to the algorithm. If the above conditions hold for all τ_i, defined as $\forall i, j \in \mathbb{F}_q, \tau_i(j) = \frac{1}{q}\sum\limits_{k\in\mathbb{F}_q}\frac{\widehat{\chi_i}(j)}{\widehat{\psi}(j)}\omega^{-jk}$, then $\forall i \in \mathbb{F}_q, \psi + \tau_i = \chi_i$ and therefore the output of Algorithm 19 follows a q-$\mathrm{LWE}^n_{\chi_0,\cdots,\chi_{q-1}}(\boldsymbol{k})$ distribution.

□

We can now use this lemma to give the following reduction:

Theorem 3 (q-LWE reduction to LWE)**.** *Let $\psi, \chi_0, \ldots, \chi_{q-1}$ be F_q-distributions. If $\forall i, j \in \mathbb{F}_q$, $\sum\limits_{k\in\mathbb{F}_q}\frac{\widehat{\chi_i}(j)}{\widehat{\psi}(j)}\omega^{-jk} \in [0,1] \subset \mathbb{R}$, then the q-$LWE^n_{\chi_0,\cdots,\chi_{q-1}}(\boldsymbol{k})$ problem is at least as hard as the $LWE^n_\psi(\boldsymbol{k})$ one.*

Proof. As shown in Proposition 14, the output of Algorithm 19 follows a q-LWE$^n_{\chi_0,\cdots,\chi_{q-1}}(\boldsymbol{k})$ distribution under the theorem assumptions. Algorithm 19 runs in constant time.

Therefore, under the theorem assumptions, the output of Algorithm 19 $(\boldsymbol{a_i}, b_i)_{0\leq i<q}$ is a q-LWE$^n_{\chi_0,\cdots,\chi_{q-1}}(\boldsymbol{k})$ sample. The algorithm can then be used to obtain q-LWE$^n_{\chi_0,\cdots,\chi_{q-1}}(\boldsymbol{k})$ samples from LWE$^n_\psi(\boldsymbol{k})$ oracle (both problems sharing the same key) in a polynomial time. An adversary can then use any attack against the LWE-OD instance to break the q-LWE one, making the q-LWE$^n_{\chi_0,\cdots,\chi_{q-1}}(\boldsymbol{k})$ problem at least as hard as the LWE$^n_\psi(\boldsymbol{k})$ one. □

Finally, we obtain the complete theorem by combining Theorem 3 and Theorem 2:

Theorem 4 (LWE-OD reduction to LWE)**.** *LWE-OD$^n_{\varphi_0,\cdots,\varphi_{q-1}}(\boldsymbol{k})$ is at least as hard as LWE$^n_\psi(\boldsymbol{k})$ if:*

$$\forall i,j \in \mathbb{F}_q \text{ such that } \sum_{k=0}^{q-1} \varphi_k(j-k) \neq 0,$$
$$\sum_{k=0}^{q-1} \frac{\sum_{l=0}^{q-1} \omega^{jl} \varphi_{i-l}(l)}{\widehat{\psi}(k) \sum_{l=0}^{q-1} \varphi_l(i-l)} \in [0,1] \subset \mathbb{R}$$

Proof. We get this result by combining Theorem 3 and Theorem 2.

First, using Theorem 2, we get that LWE-OD$^n_{\varphi_0,\cdots,\varphi_{q-1}}(\boldsymbol{k})$ reduces to q-LWE$^n_{\chi_0,\cdots,\chi_{q-1}}(\boldsymbol{k})$, where the χ_i are such that $\forall i,j \in \mathbb{F}_q$ such that $\sum_{k=0}^{q-1} \varphi_k(j-k) \neq 0, \chi_j(j-i) = \frac{\varphi_i(j-i)}{\sum_{k=0}^{q-1} \varphi_k(j-k)}$. For any LWE-OD error distributions, this q-LWE exists and is well defined (asserting that the χ_j verifying $\sum_{k=0}^{q-1} \varphi_k(j-k) = 0$ are any probability distribution).

We then use Theorem 3 to assert that, given the existence of $\mathbb{F}_q$-distributions τ_i such that $\psi + \tau_i = \chi_i$, we can reduce the obtained q-LWE to a LWE distribution, therefore obtaining the complete reduction. □

Theorem 4 provides a condition for efficiently verifying whether an LWE-OD instance reduces to an LWE instance, given the two instances. However, the theorem has certain limitations: for any given LWE instance, it is not straightforward to find non-trivial LWE-OD instances

that are at least as hard as the LWE instance. Similarly, for any given LWE-OD instance, it is not immediately clear how to find an LWE distribution that it reduces to.

This arises from the fact that the equations in Theorem 2 are only easy to solve when the LWE-OD error distributions are known, and the q-LWE error distributions are the unknowns. This means that for any LWE-OD instance, it is easy to find (if one exists) a q-LWE distribution to reduce to. However, finding an LWE-OD instance from a given q-LWE one is not straightforward using the equations.

Similarly, the equations in Theorem 3 make it straightforward to find q-LWE distributions that are at least as hard as a given LWE instance. However, starting with a q-LWE distribution, it is more difficult, using these equations, to identify an LWE instance it reduces to.

Therefore, when using Theorem 4 to find an LWE instance that a given LWE-OD instance reduces to, we can easily identify the intermediate q-LWE distribution, but there is no immediate way to find the corresponding LWE instance. The other way round, when attempting to find an LWE-OD instance that a given LWE instance reduces to, we can easily identify the intermediate q-LWE distribution, but there is no immediate way to find the corresponding LWE-OD instance.

4.4.3 Conclusion and open problems

In this work, we introduce and provide an initial analysis of a generalized learning problem called Learning with Errors with Output Dependencies (LWE-OD). This problem is defined in such a way that it encompasses traditional problems like LWE and LWR, as well as more unconventional ones, such as LPN-OD and LWPR. In our effort to establish a reduction from LWE-OD to LWE, we opted for an approach based on probabilities and rejection sampling rather than lattice-based methods. This approach led to the result in Theorem 4, which presents a system of equations involving error distributions $\psi, \varphi_0, \ldots, \varphi_{q-1}$. If this system holds, it guarantees the reduction from $\text{LWE-OD}^n_{\varphi_0,\cdots,\varphi_{q-1}}(\boldsymbol{k})$ to $\text{LWE}^n_{\psi}(\boldsymbol{k})$.

However, this result remains incomplete, as the system of equations cannot be easily solved to find LWE instances that are at least as hard as a given LWE-OD instance, or vice versa. Nevertheless, the theorem can still be applied in specific cases, such as in the rediscovery of the reduction from LWR to LWE by [Bog+16].

There are several directions for future research stemming from this work. One limitation of the current system of equations approach is that the reduction holds only when the equations are satisfied exactly.

A promising approach might be to combine our main theorem with a lattice-based method, such as using Rényi divergence [Pre17] to overcome this limitation.

5 Conclusion and future research

In this manuscript, we analyzed various aspects of leakage analysis in post-quantum schemes.

In Chapter 3, we focused on a scheme-level analysis, studying the NIST standards. In Section 3.1, we observed that many post-quantum cryptographic schemes employ the FO transform, which introduces structural weaknesses against leakage.

In Section 3.3, we proposed optimizations that could enhance implementation efficiency and introduced a new KEM scheme called `Polka`. First, we recommended eliminating the FO transform and leveraging the rigidity property [BP18] to achieve CCA security. Additionally, we introduced ciphertext randomization, enabling a leveled implementation where randomized values allow for less expensive countermeasures against leakage attacks. We also discussed integrating hard physical learning problems (covered in more detail in Chapter 4).

While the FO transform introduces a structural weakness against SCA for the PQC schemes that utilize it, we also demonstrated that leakage vulnerability is not inherent to post-quantum cryptography itself. For instance, Dilithium, a NIST standard for digital signatures, offers strong side-channel protection properties. In Section 3.2, we provided a secure and efficient implementation of Dilithium using refined leakage analysis and innovative gadgets, taking advantage of state-of-the-art masking conversion algorithms.

In Chapter 4, we explored various examples of physical learning problems and their synergy with PQC leakage analysis and secure implementation. In Sections 4.1 and 4.2.2 we examined how inexact computing can be applied in PQC implementations. In Section 4.1, after noticing an output depency in existing LPPN implementations, we proposed a mathematical reduction to prove that this implementation was still secured. We then proceed in Section 4.2.2, we extended LPPN to higher dimensions, providing an FPGA prototype that implements LWPE, a physical variant of LWE. This method can be used in PQC schemes that do not require deterministic randomness. Inexact computing eliminates the need for a PRNG to generate the error, reducing the attack surface for SCA and transforming sensitive operations into key-homomorphic

ones. This property allows for a linear overhead when applying masking, minimizing the cost of side-channel security in the implementation.

We further studied the empirical hardness of the LWPR problem in Section 4.3, where previous analysis was limited to leakage modeled as Hamming weight. We extended this to linear and quadratic leakage models, generalizing Hamming weight, and demonstrated the inefficiency of attacks under these models. Additionally, we showed that such models closely correlate with the real leakage observed when implementing LWPR on FPGA hardware.

Finally, in Section 4.4, we aimed to establish a proof framework for generalized unstructured learning problems. We introduced the LWE-OD problem (Definition 25), which is a generalization of LWE, LWR or LWPR. We then proceeds to give a reduction from LWE-OD to LWE. This reduction takes the form of a system of equation involving parameters of the LWE and LWE-OD distribution. However, this result remains limited, as the system does not allow the identification of LWE instances that are at least as hard as LWE-OD instances, or vice versa. Nevertheless, this work represents a step toward the theoretical understanding necessary for applying physical learning problems to PQC, offering a different approach from traditional reductions in learning problems.

At the beginning of this thesis, I approached my initial projects with the intuition that the FO transform represented a structural vulnerability for physical security. I also anticipated that achieving physical security would be particularly challenging for post-quantum cryptography (PQC), even more so than for classical schemes.

While this intuition held true for the FO transform, I discovered through my work that this black-box security method is only relevant for KEM and that other schemes did not present equivalent flaws. Interestingly, while implementing Dilithium, I found that secure implementation of certain PQC schemes was not necessarily unattainable. In fact, Dilithium's signature scheme turned out to be relatively conventional to secure, exhibiting no inherent structural weaknesses or unusual masking synergies, at least up to our knowledge.

However, exploring hard physical learning problems revealed a promising synergy between leakage and modular arithmetic, used in lattice-based and code-based cryptogrtaphy. This insight suggests that secure PQC implementations could potentially be more efficient than their classical counterparts. The unique hardness properties of these problems enabled the application of prior leakage-resilient strategies, such as leveling, to PQC implementations.

Moreover, the recognition that leakage problems, whether in PQC or sometimes even in symmetric cryptography (where LWPR has been applied as a rekeying block), can be framed as learning problems typically associated with PQC introduces an exciting possibility. Expertise in these learning problems and their reductions may offer a structured framework for studying leakage. In turn, advances in PQC research could significantly benefit the field of leakage analysis.

The study of leakage impacts on PQC is still in its early stages and can currently be divided into two approaches. Theoretical work focuses on establishing sufficient security conditions within abstract models, which are often disconnected from real-world observations. Practical work, on the other hand, seeks to develop effective countermeasures based on heuristic knowledge, but it frequently lacks formal security proofs. A long-term goal is to bridge this gap, achieving a "meet-in-the-middle" synthesis of these approaches.

I hope that my thesis represents a meaningful step toward this objective. In my view, both approaches are complementary and mutually reinforcing. For example, while Polka employed a bottom-up approach, aiming to provide a concrete proposal with heuristic countermeasures, it still benefited from the theoretical groundwork on LWPR and the study of symmetric cryptography leakage. Conversely, attempting to implement this construction in a real-world system raised important questions about its variants, limitations, and usage constraints.

Looking ahead, I hope to continue building connections between my projetcs and further expanding upon these insights. I aim to continue exploring both perspectives and finding ways to combine them. On the practical side, one potential direction is to implement Polka efficiently in software to validate our claims about its performance relative to Kyber. Additionally, further improvements to Dilithium's implementation could leverage the security properties of hard physical learning problems.

From my perspective, the most intriguing avenue for future work lies in formally proving the security of these hard physical learning problems. Specifically, I hope to delve deeper into the lattice proof literature to explore applying these techniques to establish the security of LWPR and LWE-OD.

Related Publications

[Azo+22b] Melissa Azouaoui, Olivier Bronchain, Clément Hoffmann, Yulia Kuzovkova, Tobias Schneider, and François-Xavier Standaert. "Systematic Study of Decryption and Re-encryption Leakage: The Case of Kyber." In: *COSADE*. Vol. 13211. Lecture Notes in Computer Science. Springer, 2022, pp. 236–256.

[Azo+22c] Melissa Azouaoui et al. "Leveling Dilithium against Leakage: Revisited Sensitivity Analysis and Improved Implementations." In: *Fourth PQC Standardization Conference* (2022).

[Bel+21] Davide Bellizia, Clément Hoffmann, Dina Kamel, Hanlin Liu, Pierrick Méaux, François-Xavier Standaert, and Yu Yu. "Learning Parity with Physical Noise: Imperfections, Reductions and FPGA Prototype." In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2021.3 (2021), pp. 390–417. DOI: 10.46586/tches.v2021.i3.390-417. URL: https://doi.org/10.46586/tches.v2021.i3.390-417.

[Bel+22] Davide Bellizia, Clément Hoffmann, Dina Kamel, Pierrick Méaux, and François-Xavier Standaert. "When Bad News Become Good News Towards Usable Instances of Learning with Physical Errors." In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2022.4 (2022), pp. 1–24. DOI: 10.46586/TCHES.V2022.I4.1-24. URL: https://doi.org/10.46586/tches.v2022.i4.1-24.

[Hof+22] Clément Hoffmann, Benoît Libert, Charles Momin, Thomas Peters, and François-Xavier Standaert. *Towards Leakage-Resistant Post-Quantum CCA-Secure Public Key Encryption*. Cryptology ePrint Archive, Paper 2022/873. https://eprint.iacr.org/2022/873. 2022. URL: https://eprint.iacr.org/2022/873.

[Hof+23] Clément Hoffmann, Pierrick Méaux, Charles Momin, Yann Rotella, François-Xavier Standaert, and Balazs Udvarhelyi. "Learning with Physical Rounding for Linear and Quadratic

Leakage Functions." In: *Advances in Cryptology - CRYPTO 2023 - 43rd Annual International Cryptology Conference, CRYPTO 2023, Santa Barbara, CA, USA, August 20-24, 2023, Proceedings, Part III*. Ed. by Helena Handschuh and Anna Lysyanskaya. Vol. 14083. Lecture Notes in Computer Science. Springer, 2023, pp. 410–439. DOI: 10.1007/978-3-031-38548-3_14. URL: https://doi.org/10.1007/978-3-031-38548-3_14.

Other Publications

[Ara+24] Diego F. Aranha, Antonio Guimarães, Clément Hoffmann, and Pierrick Méaux. *Secure and efficient transciphering for FHE-based MPC*. Cryptology ePrint Archive, Paper 2024/1702. 2024. URL: https://eprint.iacr.org/2024/1702.

[Cos+22b] Orel Cosseron, Clément Hoffmann, Pierrick Méaux, and François-Xavier Standaert. "Towards Case-Optimized Hybrid Homomorphic Encryption - Featuring the Elisabeth Stream Cipher." In: *ASIACRYPT (3)*. Vol. 13793. Lecture Notes in Computer Science. Springer, 2022, pp. 32–67.

[HMR20] Clément Hoffmann, Pierrick Méaux, and Thomas Ricosset. "Transciphering, Using FiLIP and TFHE for an Efficient Delegation of Computation." In: *Progress in Cryptology - INDOCRYPT 2020 - 21st International Conference on Cryptology in India, Bangalore, India, December 13-16, 2020, Proceedings*. Ed. by Karthikeyan Bhargavan, Elisabeth Oswald, and Manoj Prabhakaran. Vol. 12578. Lecture Notes in Computer Science. Springer, 2020, pp. 39–61. DOI: 10.1007/978-3-030-65277-7_3. URL: https://doi.org/10.1007/978-3-030-65277-7_3.

[HMS23] Clément Hoffmann, Pierrick Méaux, and François-Xavier Standaert. "The Patching Landscape of Elisabeth-4 and the Mixed Filter Permutator Paradigm." In: *Progress in Cryptology - INDOCRYPT 2023 - 24th International Conference on Cryptology in India, Goa, India, December 10-13, 2023, Proceedings, Part I*. Ed. by Anupam Chattopadhyay, Shivam Bhasin, Stjepan Picek, and Chester Rebeiro. Vol. 14459. Lecture Notes in Computer Science. Springer, 2023, pp. 134–156. DOI: 10.1007/978-3-031-56232-7_7. URL: https://doi.org/10.1007/978-3-031-56232-7_7.

Bibliography

[AA16] Jacob Alperin-Sheriff and Daniel Apon. "Dimension-Preserving Reductions from LWE to LWR." In: *IACR Cryptol. ePrint Arch.* (2016), p. 589. URL: http://eprint.iacr.org/2016/589.

[AAB17] Benny Applebaum, Jonathan Avron, and Chris Brzuska. "Arithmetic Cryptography." In: *J. ACM* 64.2 (2017), 10:1–10:74. DOI: 10.1145/3046675. URL: https://doi.org/10.1145/3046675.

[ABD16] M. Albrecht, S. Bai, and L. Ducas. "A subfield lattice attack on overstretched NTRU assumptions: Cryptanalysis of some FHE and Graded Encoding Schemes." In: *Crypto*. 2016.

[Abd+22] Amin Abdulrahman, Vincent Hwang, Matthias J. Kannwischer, and Daan Sprenkels. "Faster Kyber and Dilithium on the Cortex-M4." In: *ACNS*. Vol. 13269. Lecture Notes in Computer Science. Springer, 2022, pp. 853–871.

[AG10] Sanjeev Arora and Rong Ge. "Learning Parities with Structured Noise." In: *Electron. Colloquium Comput. Complex.* 17 (2010), p. 66.

[AHM14] Frederik Armknecht, Matthias Hamann, and Vasily Mikhalev. "Lightweight Authentication Protocols on Ultra-Constrained RFIDs - Myths and Facts." In: *RFIDSec*. Vol. 8651. Lecture Notes in Computer Science. Springer, 2014, pp. 1–18.

[Ala+19] Gorjan Alagic et al. *Status report on the first round of the NIST post-quantum cryptography standardization process.* National Institute of Standards and Technology, 2019.

[Alb+15] Martin R. Albrecht, Christian Rechberger, Thomas Schneider, Tyge Tiessen, and Michael Zohner. "Ciphers for MPC and FHE." In: *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*. Ed. by

Elisabeth Oswald and Marc Fischlin. Vol. 9056. Lecture Notes in Computer Science. Springer, 2015, pp. 430–454. DOI: 10.1007/978-3-662-46800-5_17. URL: https://doi.org/10.1007/978-3-662-46800-5_17.

[Alb+19] Martin R. Albrecht, Christian Hanser, Andrea Höller, Thomas Pöppelmann, Fernando Virdia, and Andreas Wallner. "Implementing RLWE-based Schemes Using an RSA Co-Processor." In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2019.1 (2019), pp. 169–208. DOI: 10.13154/tches.v2019.i1.169-208. URL: https://doi.org/10.13154/tches.v2019.i1.169-208.

[Alk+16a] E. Alkim, L. Ducas, T. Pöppelmann, and P. Schwabe. "Post-quantum key exchange – A new hope." In: *USENIX Security Symposium*. 2016.

[Alk+16b] Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. "Post-quantum Key Exchange - A New Hope." In: *25th USENIX Security Symposium, USENIX Security 16, Austin, TX, USA, August 10-12, 2016*. Ed. by Thorsten Holz and Stefan Savage. USENIX Association, 2016, pp. 327–343. URL: https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/alkim.

[Alk+20] Erdem Alkim et al. *NewHope Algorithm Specifications and Supporting Documentation*. https://newhopecrypto.org/data/NewHope_2020_04_10.pdf. 2020-04-10. 2020.

[Alk+21] Erdem Alkim et al. *FrodoKEM Learning With Errors Key Encapsulation*. https://frodokem.org/files/FrodoKEM-specification-20210604.pdf. 2021-06-04. 2021.

[APS15] M. Albrecht, R. Player, and S. Scott. "On the concrete hardness of Learning with Errors." In: *Journal of Mathematical Cryptology* 9.3 (2015).

[Ara+24] Diego F. Aranha, Antonio Guimarães, Clément Hoffmann, and Pierrick Méaux. *Secure and efficient transciphering for FHE-based MPC*. Cryptology ePrint Archive, Paper 2024/1702. 2024. URL: https://eprint.iacr.org/2024/1702.

[Arm+11] Frederik Armknecht, Roel Maes, Ahmad-Reza Sadeghi, François-Xavier Standaert, and Christian Wachsmann. "A Formalization of the Security Features of Physical Functions." In: *IEEE Symposium on Security and Privacy*. IEEE

Computer Society, 2011, pp. 397–412. DOI: 10.1109/SP.2011.10. URL: https://doi.org/10.1109/SP.2011.10.

[Ava+19] Roberto Avanzi et al. "CRYSTALS-Kyber algorithm specifications and supporting documentation." In: *NIST PQC Round* 3 (2019), p. 4.

[Ava+20] Roberto Avanzi et al. "CRYSTALS-KYBER algorithm specifications and supporting documentation." Version 3.02. In: *NIST PQC Round* 3 (2020). 2021-08-04, p. 42.

[Azo+22a] Melissa Azouaoui, Olivier Bronchain, Clément Hoffmann, Yulia Kuzovkova, Tobias Schneider, and François-Xavier Standaert. *Systematic Study of Decryption and Re-Encryption Leakage: the Case of KYBER*. Cryptology ePrint Archive, Report 2022/036. https://ia.cr/2022/036. 2022.

[Azo+22b] Melissa Azouaoui, Olivier Bronchain, Clément Hoffmann, Yulia Kuzovkova, Tobias Schneider, and François-Xavier Standaert. "Systematic Study of Decryption and Re-encryption Leakage: The Case of Kyber." In: *COSADE*. Vol. 13211. Lecture Notes in Computer Science. Springer, 2022, pp. 236–256.

[Azo+22c] Melissa Azouaoui et al. "Leveling Dilithium against Leakage: Revisited Sensitivity Analysis and Improved Implementations." In: *Fourth PQC Standardization Conference* (2022).

[Bal+14] Josep Balasch, Benedikt Gierlichs, Vincent Grosso, Oscar Reparaz, and François-Xavier Standaert. "On the Cost of Lazy Engineering for Masked Software Implementations." In: *CARDIS*. Vol. 8968. Lecture Notes in Computer Science. Springer, 2014, pp. 64–81.

[Bar+16] Gilles Barthe, Sonia Belaïd, François Dupressoir, Pierre-Alain Fouque, Benjamin Grégoire, Pierre-Yves Strub, and Rébecca Zucchini. "Strong Non-Interference and Type-Directed Higher-Order Masking." In: *CCS*. ACM, 2016, pp. 116–129.

[Bar+17] Gilles Barthe, François Dupressoir, Sebastian Faust, Benjamin Grégoire, François-Xavier Standaert, and Pierre-Yves Strub. "Parallel Implementations of Masking Schemes and the Bounded Moment Leakage Model." In: *EUROCRYPT (1)*. Vol. 10210. Lecture Notes in Computer Science. 2017, pp. 535–566.

[Bar86] Paul Barrett. "Implementing the Rivest Shamir and Adleman Public Key Encryption Algorithm on a Standard Digital Signal Processor." In: *CRYPTO*. Ed. by Andrew M. Odlyzko. Vol. 263. Lecture Notes in Computer Science. Springer, 1986, pp. 311–323. DOI: 10.1007/3-540-47721-7_24. URL: https://doi.org/10.1007/3-540-47721-7_24.

[Bas+19] Andrea Basso, Jose Maria Bermudo Mera, Jan-Pieter D'Anvers, Angshuman Karmakar, Sujoy Sinha Roy, Michiel Van Beirendonck, and Frederik Vercauteren. "Saber algorithm specifications and supporting documentation." In: *NIST PQC Round* 3 (2019), p. 44.

[BC22a] Olivier Bronchain and Gaëtan Cassiers. "Bitslicing Arithmetic/Boolean Masking Conversions for Fun and Profit with Application to Lattice-Based KEMs." In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2022.4 (2022), pp. 553–588.

[BC22b] Olivier Bronchain and Gaëtan Cassiers. *Bitslicing Arithmetic/Boolean Masking Conversions for Fun and Profit with Application to Lattice-Based KEMs*. Cryptology ePrint Archive, Report 2022/158. 2022.

[BCO04] Eric Brier, Christophe Clavier, and Francis Olivier. "Correlation Power Analysis with a Leakage Model." In: *CHES*. Vol. 3156. Lecture Notes in Computer Science. Springer, 2004, pp. 16–29.

[Bei+21] Michiel Van Beirendonck, Jan-Pieter D'Anvers, Angshuman Karmakar, Josep Balasch, and Ingrid Verbauwhede. "A Side-Channel-Resistant Implementation of SABER." In: *ACM J. Emerg. Technol. Comput. Syst.* 17.2 (2021), 10:1–10:26. DOI: 10.1145/3429983. URL: https://doi.org/10.1145/3429983.

[Bel+15] Sonia Belaïd, Jean-Sébastien Coron, Pierre-Alain Fouque, Benoît Gérard, Jean-Gabriel Kammerer, and Emmanuel Prouff. "Improved Side-Channel Analysis of Finite-Field Multiplication." In: *CHES*. Vol. 9293. Lecture Notes in Computer Science. Springer, 2015, pp. 395–415.

[Bel+20] Davide Bellizia, Olivier Bronchain, Gaëtan Cassiers, Vincent Grosso, Chun Guo, Charles Momin, Olivier Pereira, Thomas Peters, and François-Xavier Standaert. "Mode-Level vs. Implementation-Level Physical Security in Symmetric Cryptography - A Practical Guide Through the

Leakage-Resistance Jungle." In: *Advances in Cryptology - CRYPTO 2020 - 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17-21, 2020, Proceedings, Part I*. Ed. by Daniele Micciancio and Thomas Ristenpart. Vol. 12170. Lecture Notes in Computer Science. Springer, 2020, pp. 369–400. DOI: 10.1007/978-3-030-56784-2_13. URL: https://doi.org/10.1007/978-3-030-56784-2_13.

[Bel+21] Davide Bellizia, Clément Hoffmann, Dina Kamel, Hanlin Liu, Pierrick Méaux, François-Xavier Standaert, and Yu Yu. "Learning Parity with Physical Noise: Imperfections, Reductions and FPGA Prototype." In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2021.3 (2021), pp. 390–417. DOI: 10.46586/tches.v2021.i3.390-417. URL: https://doi.org/10.46586/tches.v2021.i3.390-417.

[Bel+22] Davide Bellizia, Clément Hoffmann, Dina Kamel, Pierrick Méaux, and François-Xavier Standaert. "When Bad News Become Good News Towards Usable Instances of Learning with Physical Errors." In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2022.4 (2022), pp. 1–24. DOI: 10.46586/TCHES.V2022.I4.1-24. URL: https://doi.org/10.46586/tches.v2022.i4.1-24.

[Ber+11] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. *The KECCAK reference*. 2011. URL: https://keccak.team/files/Keccak-reference-3.0.pdf.

[Ber+22] Francesco Berti, Shivam Bhasin, Jakub Breier, Xiaolu Hou, Romain Poussier, François-Xavier Standaert, and Balazs Udvarhelyi. "A Finer-Grain Analysis of the Leakage (Non) Resilience of OCB." In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2022.1 (2022), pp. 461–481.

[Ber+23] Alexandre Berzati, Andersson Calle Viera, Maya Chartouni, Steven Madec, Damien Vergnaud, and David Vigilant. "A Practical Template Attack on CRYSTALS-Dilithium." In: *IACR Cryptol. ePrint Arch.* (2023), p. 50. URL: https://eprint.iacr.org/2023/050.

[BFG14] Sonia Belaïd, Pierre-Alain Fouque, and Benoît Gérard. "Side-Channel Analysis of Multiplications in GF(2128) - Application to AES-GCM." In: *ASIACRYPT (2)*. Vol. 8874. Lecture Notes in Computer Science. Springer, 2014, pp. 306–325.

[BFS15] Magali Bardet, Jean-Charles Faugère, and Bruno Salvy. "On the complexity of the F5 Gröbner basis algorithm." In: *J. Symb. Comput.* 70 (2015), pp. 49–70.

[BGV12] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. "(Leveled) fully homomorphic encryption without bootstrapping." In: *ITCS*. ACM, 2012, pp. 309–325.

[Bha+21] Shivam Bhasin, Jan-Pieter D'Anvers, Daniel Heinz, Thomas Pöppelmann, and Michiel Van Beirendonck. "Attacking and Defending Masked Polynomial Comparison for Lattice-Based Cryptography." In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2021.3 (2021), pp. 334–359.

[BL12] Daniel J. Bernstein and Tanja Lange. "Never Trust a Bunny." In: *RFIDSec*. Vol. 7739. Lecture Notes in Computer Science. Springer, 2012, pp. 137–148.

[BL17] Daniel J. Bernstein and Tanja Lange. "Post-quantum cryptography." In: *Nat.* 549.7671 (2017), pp. 188–194.

[Blu+93] Avrim Blum, Merrick L. Furst, Michael J. Kearns, and Richard J. Lipton. "Cryptographic Primitives Based on Hard Learning Problems." In: *CRYPTO*. Vol. 773. Lecture Notes in Computer Science. Springer, 1993, pp. 278–291.

[Bog+16] Andrej Bogdanov, Siyao Guo, Daniel Masny, Silas Richelson, and Alon Rosen. "On the Hardness of Learning with Rounding over Small Modulus." In: *Theory of Cryptography - 13th International Conference, TCC 2016-A, Tel Aviv, Israel, January 10-13, 2016, Proceedings, Part I*. Ed. by Eyal Kushilevitz and Tal Malkin. Vol. 9562. Lecture Notes in Computer Science. Springer, 2016, pp. 209–224. DOI: 10.1007/978-3-662-49096-9_9. URL: https://doi.org/10.1007/978-3-662-49096-9_9.

[Bon+11] D. Boneh, O. Dagdelen, M. Fischlin, A. Lehmann, C. Schaffner, and M. Zhandry. "Random Oracles in a Quantum World." In: *Asiacrypt*. 2011.

[Bon+18] Dan Boneh, Yuval Ishai, Alain Passelègue, Amit Sahai, and David J. Wu. "Exploring Crypto Dark Matter: - New Simple PRF Candidates and Their Applications." In: *TCC (2)*. Vol. 11240. Lecture Notes in Computer Science. Springer, 2018, pp. 699–729.

[Bos+15] Joppe W. Bos, Craig Costello, Michael Naehrig, and Douglas Stebila. "Post-Quantum Key Exchange for the TLS Protocol from the Ring Learning with Errors Problem." In: *2015 IEEE Symposium on Security and Privacy, SP 2015, San Jose, CA, USA, May 17-21, 2015*. IEEE Computer Society, 2015, pp. 553–570. DOI: 10.1109/SP.2015.40. URL: https://doi.org/10.1109/SP.2015.40.

[Bos+18a] J. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J. Schanck, P. Schwabe, G. Seiler, and D. Stehlé. "CRYSTALS - Kyber: A CCA-Secure Module-Lattice-Based KEM." In: *IEEE European Symposium on Security and Privacy, EuroS&P*. 2018.

[Bos+18b] Joppe W. Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehlé. "CRYSTALS - Kyber: A CCA-Secure Module-Lattice-Based KEM." In: *EuroS&P*. IEEE, 2018, pp. 353–367.

[Bos+21] Joppe W. Bos, Marc Gourjon, Joost Renes, Tobias Schneider, and Christine van Vredendaal. "Masking Kyber: First- and Higher-Order Implementations." In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2021.4 (2021), pp. 173–214. DOI: 10.46586/tches.v2021.i4.173-214. URL: https://doi.org/10.46586/tches.v2021.i4.173-214.

[Boy+18] Elette Boyle, Geoffroy Couteau, Niv Gilboa, and Yuval Ishai. "Compressing Vector OLE." In: *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018*. Ed. by David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang. ACM, 2018, pp. 896–912. DOI: 10.1145/3243734.3243868. URL: https://doi.org/10.1145/3243734.3243868.

[Boy+19] Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, Peter Rindal, and Peter Scholl. "Efficient Two-Round OT Extension and Silent Non-Interactive Secure Computation." In: *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS 2019, London, UK, November 11-15, 2019*. Ed. by Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz. ACM, 2019, pp. 291–308. DOI: 10.1145/

3319535.3354255. URL: https://doi.org/10.1145/3319535.3354255.

[BP18] Daniel J. Bernstein and Edoardo Persichetti. *Towards KEM Unification*. Cryptology ePrint Archive, Report 2018/526. https://ia.cr/2018/526. 2018.

[BPR12] Abhishek Banerjee, Chris Peikert, and Alon Rosen. "Pseudorandom Functions and Lattices." In: *EUROCRYPT*. Vol. 7237. Lecture Notes in Computer Science. Springer, 2012, pp. 719–737.

[Bra+13] Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. "Classical hardness of learning with errors." In: *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*. Ed. by Dan Boneh, Tim Roughgarden, and Joan Feigenbaum. ACM, 2013, pp. 575–584. DOI: 10.1145/2488608.2488680. URL: https://doi.org/10.1145/2488608.2488680.

[Bre+14] Hai Brenner, Lubos Gaspar, Gaëtan Leurent, Alon Rosen, and François-Xavier Standaert. "FPGA Implementations of SPRING - And Their Countermeasures against Side-Channel Attacks." In: *CHES*. Vol. 8731. Lecture Notes in Computer Science. Springer, 2014, pp. 414–432.

[Bru+16] Leon Groot Bruinderink, Andreas Hülsing, Tanja Lange, and Yuval Yarom. "Flush, Gauss, and Reload - A Cache Attack on the BLISS Lattice-Based Signature Scheme." In: *Cryptographic Hardware and Embedded Systems - CHES 2016 - 18th International Conference, Santa Barbara, CA, USA, August 17-19, 2016, Proceedings*. Ed. by Benedikt Gierlichs and Axel Y. Poschmann. Vol. 9813. Lecture Notes in Computer Science. Springer, 2016, pp. 323–345. DOI: 10.1007/978-3-662-53140-2_16. URL: https://doi.org/10.1007/978-3-662-53140-2_16.

[BS21] Olivier Bronchain and François-Xavier Standaert. "Breaking Masked Implementations with Many Shares on 32-bit Software Platforms or When the Security Order Does Not Matter." In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2021.3 (2021), pp. 202–234.

[BSS21] Olivier Bronchain, Tobias Schneider, and François-Xavier Standaert. "Reducing risks through simplicity: high side-channel security for lazy engineers." In: *J. Cryptogr. Eng.* 11.1 (2021), pp. 39–55.

[BTV16] Sonia Bogos, Florian Tramer, and Serge Vaudenay. "On solving LPN using BKW and variants." In: *Cryptography and Communications* 8.3 (2016), pp. 331–369.

[BUS21] Davide Bellizia, Balazs Udvarhelyi, and François-Xavier Standaert. "Towards a Better Understanding of Side-Channel Analysis Measurements Setups." In: *CARDIS*. Vol. 13173. Lecture Notes in Computer Science. Springer, 2021, pp. 64–79.

[BV11a] Zvika Brakerski and Vinod Vaikuntanathan. "Efficient Fully Homomorphic Encryption from (Standard) LWE." In: *FOCS*. IEEE Computer Society, 2011, pp. 97–106.

[BV11b] Zvika Brakerski and Vinod Vaikuntanathan. "Fully Homomorphic Encryption from Ring-LWE and Security for Key Dependent Messages." In: *CRYPTO*. Vol. 6841. Lecture Notes in Computer Science. Springer, 2011, pp. 505–524.

[Can+16] Anne Canteaut, Sergiu Carpov, Caroline Fontaine, Tancrède Lepoint, María Naya-Plasencia, Pascal Paillier, and Renaud Sirdey. "Stream Ciphers: A Practical Solution for Efficient Homomorphic-Ciphertext Compression." In: *Fast Software Encryption - 23rd International Conference, FSE 2016, Bochum, Germany, March 20-23, 2016, Revised Selected Papers*. Ed. by Thomas Peyrin. Vol. 9783. Lecture Notes in Computer Science. Springer, 2016, pp. 313–333. DOI: 10.1007/978-3-662-52993-5_16. URL: https://doi.org/10.1007/978-3-662-52993-5_16.

[Car21] Claude Carlet. *Boolean Functions for Cryptography and Coding Theory*. Cambridge University Press, 2021. DOI: 10.1017/9781108606806.

[Cas+10] David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. "Bonsai Trees, or How to Delegate a Lattice Basis." In: *EUROCRYPT*. Vol. 6110. Lecture Notes in Computer Science. Springer, 2010, pp. 523–552.

[Cas+21] Gaëtan Cassiers, Benjamin Grégoire, Itamar Levi, and François-Xavier Standaert. "Hardware Private Circuits: From Trivial Composition to Full Verification." In: *IEEE Trans. Computers* 70.10 (2021), pp. 1677–1690.

[Cha+99] Suresh Chari, Charanjit S. Jutla, Josyula R. Rao, and Pankaj Rohatgi. "Towards Sound Approaches to Counteract Power-Analysis Attacks." In: *CRYPTO*. Vol. 1666. Lecture Notes in Computer Science. Springer, 1999, pp. 398–412.

[Che+17] Jung Hee Cheon, Andrey Kim, Miran Kim, and Yongsoo Song. "Homomorphic encryption for arithmetic of approximate numbers." In: *Advances in Cryptology–ASIACRYPT 2017: 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I 23*. Springer. 2017, pp. 409–437.

[Ché+19] Eloi de Chérisey, Sylvain Guilley, Olivier Rioul, and Pablo Piantanida. "Best Information is Most Successful Mutual Information and Success Rate in Side-Channel Analysis." In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2019.2 (2019), pp. 49–79.

[Che+20] Cong Chen et al. "NTRU algorithm specifications and supporting documentation." In: *NIST PQC Round* 3 (2020), p. 41.

[Chi+20a] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. "TFHE: Fast Fully Homomorphic Encryption Over the Torus." In: *J. Cryptol.* 33.1 (2020), pp. 34–91. DOI: 10.1007/S00145-019-09319-X. URL: https://doi.org/10.1007/s00145-019-09319-x.

[Chi+20b] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. "TFHE: fast fully homomorphic encryption over the torus." In: *Journal of Cryptology* 33.1 (2020), pp. 34–91.

[Cho+21] Jihoon Cho, Jincheol Ha, Seongkwang Kim, ByeongHak Lee, Joohee Lee, Jooyoung Lee, Dukjae Moon, and Hyojin Yoon. "Transciphering Framework for Approximate Homomorphic Encryption." In: *Advances in Cryptology - ASIACRYPT 2021 - 27th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 6-10, 2021, Proceedings, Part*

III. Ed. by Mehdi Tibouchi and Huaxiong Wang. Vol. 13092. Lecture Notes in Computer Science. Springer, 2021, pp. 640–669. DOI: 10.1007/978-3-030-92078-4_22. URL: https://doi.org/10.1007/978-3-030-92078-4_22.

[Cor+12] Jean-Sébastien Coron, Christophe Giraud, Emmanuel Prouff, Soline Renner, Matthieu Rivain, and Praveen Kumar Vadnala. "Conversion of Security Proofs from One Leakage Model to Another: A New Issue." In: *COSADE*. Vol. 7275. Lecture Notes in Computer Science. Springer, 2012, pp. 69–81.

[Cor+13] Jean-Sébastien Coron, Emmanuel Prouff, Matthieu Rivain, and Thomas Roche. "Higher-Order Side Channel Security and Mask Refreshing." In: *FSE*. Vol. 8424. Lecture Notes in Computer Science. Springer, 2013, pp. 410–424.

[Cor+23] Jean-Sébastien Coron, François Gérard, Simon Montoya, and Rina Zeitoun. "High-order Polynomial Comparison and Masking Lattice-based Encryption." In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2023.1 (2023), pp. 153–192.

[Cos+22a] Orel Cosseron, Clément Hoffmann, Pierrick Méaux, and François-Xavier Standaert. "Towards Case-Optimized Hybrid Homomorphic Encryption - Featuring the Elisabeth Stream Cipher." In: *Advances in Cryptology - ASIACRYPT 2022 - 28th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, December 5-9, 2022, Proceedings, Part III*. Ed. by Shweta Agrawal and Dongdai Lin. Vol. 13793. Lecture Notes in Computer Science. Springer, 2022, pp. 32–67. DOI: 10.1007/978-3-031-22969-5_2. URL: https://doi.org/10.1007/978-3-031-22969-5_2.

[Cos+22b] Orel Cosseron, Clément Hoffmann, Pierrick Méaux, and François-Xavier Standaert. "Towards Case-Optimized Hybrid Homomorphic Encryption - Featuring the Elisabeth Stream Cipher." In: *ASIACRYPT (3)*. Vol. 13793. Lecture Notes in Computer Science. Springer, 2022, pp. 32–67.

[Cou02] Nicolas T. Courtois. "Higher Order Correlation Attacks, XL Algorithm and Cryptanalysis of Toyocrypt." In: *Information Security and Cryptology - ICISC 2002, 5th International Conference Seoul, Korea, November 28-29, 2002, Revised Papers*. Ed. by Pil Joong Lee and Chae Hoon Lim. Vol. 2587. Lecture Notes in Computer Science. Springer,

2002, pp. 182–199. DOI: 10.1007/3-540-36552-4_13. URL: https://doi.org/10.1007/3-540-36552-4_13.

[CRR02] Suresh Chari, Josyula R. Rao, and Pankaj Rohatgi. "Template Attacks." In: *CHES*. Vol. 2523. Lecture Notes in Computer Science. Springer, 2002, pp. 13–28.

[CS20] Gaëtan Cassiers and François-Xavier Standaert. "Trivially and Efficiently Composing Masked Gadgets With Probe Isolating Non-Interference." In: *IEEE Trans. Inf. Forensics Secur.* 15 (2020), pp. 2542–2555.

[CS21a] Gaëtan Cassiers and François-Xavier Standaert. "Provably Secure Hardware Masking in the Transition- and Glitch-Robust Probing Model: Better Safe than Sorry." In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2021.2 (2021), pp. 136–158.

[CS21b] Jean-Sébastien Coron and Lorenzo Spignoli. "Secure Wire Shuffling in the Probing Model." In: *CRYPTO (3)*. Vol. 12827. Lecture Notes in Computer Science. Springer, 2021, pp. 215–244.

[CT65] James W. Cooley and John W. Tukey. "An algorithm for the machine calculation of complex Fourier series." In: *Mathematics of Computation* 19 (1965), pp. 297–301.

[D'A+18a] J.-P. D'Anvers, A. Karmakar, S.-S. Roy, and F. Vercauteren. "Saber: Module-LWR Based Key Exchange, CPA-Secure Encryption and CCA-Secure KEM." In: *Africacrypt*. 2018.

[D'A+18b] Jan-Pieter D'Anvers, Angshuman Karmakar, Sujoy Sinha Roy, and Frederik Vercauteren. "Saber: Module-LWR Based Key Exchange, CPA-Secure Encryption and CCA-Secure KEM." In: *AFRICACRYPT*. Vol. 10831. Lecture Notes in Computer Science. Springer, 2018, pp. 282–305.

[D'A+19] J.-P. D'Anvers, Q. Guo, T. Johansson, A. Nilsson, F. Vercauteren, and I. Verbauwhede. "Decryption Failure Attacks on IND-CCA Secure Lattice-Based Schemes." In: *PKC*. 2019.

[Dac+20] Dana Dachman-Soled, Léo Ducas, Huijing Gong, and Mélissa Rossi. "LWE with Side Information: Attacks and Concrete Security Estimation." In: *Advances in Cryptology - CRYPTO 2020 - 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17-21, 2020, Proceedings, Part II*. Ed. by Daniele

Micciancio and Thomas Ristenpart. Vol. 12171. Lecture Notes in Computer Science. Springer, 2020, pp. 329–358. DOI: 10.1007/978-3-030-56880-1_12. URL: https://doi.org/10.1007/978-3-030-56880-1_12.

[DDF14] Alexandre Duc, Stefan Dziembowski, and Sebastian Faust. "Unifying Leakage Models: From Probing Attacks to Noisy Leakage." In: *EUROCRYPT*. Vol. 8441. Lecture Notes in Computer Science. Springer, 2014, pp. 423–440.

[Dev+23] Julien Devevey, Pouria Fallahpour, Alain Passelègue, and Damien Stehlé. "A Detailed Analysis of Fiat-Shamir with Aborts." In: *IACR Cryptol. ePrint Arch.* (2023), p. 245. URL: https://eprint.iacr.org/2023/245.

[DFS15] Alexandre Duc, Sebastian Faust, and François-Xavier Standaert. "Making Masking Security Proofs Concrete - Or How to Evaluate the Security of Any Leaking Device." In: *EUROCRYPT (1)*. Vol. 9056. Lecture Notes in Computer Science. Springer, 2015, pp. 401–429.

[Din+21] Itai Dinur, Steven Goldfeder, Tzipora Halevi, Yuval Ishai, Mahimna Kelkar, Vivek Sharma, and Greg Zaverucha. "MPC-Friendly Symmetric Cryptography from Alternating Moduli: Candidates, Protocols, and Applications." In: *CRYPTO (4)*. Vol. 12828. Lecture Notes in Computer Science. Springer, 2021, pp. 517–547.

[DMG21] BaKhacTrieu Dang, Kamyar Mohajerani, and Kris Gaj. "High-Speed Hardware Architectures and Fair FPGA Benchmarking." In: *Third PQC Standardization Conference, June 7 - 9, 2021*. 2021.

[Dob+15] Christoph Dobraunig, François Koeune, Stefan Mangard, Florian Mendel, and François-Xavier Standaert. "Towards Fresh and Hybrid Re-Keying Schemes with Beyond Birthday Security." In: *CARDIS*. Vol. 9514. Lecture Notes in Computer Science. Springer, 2015, pp. 225–241.

[Dob+18] Christoph Dobraunig, Maria Eichlseder, Lorenzo Grassi, Virginie Lallemand, Gregor Leander, Eik List, Florian Mendel, and Christian Rechberger. "Rasta: A Cipher with Low ANDdepth and Few ANDs per Bit." In: *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part I*. Ed. by Hovav Shacham and

Alexandra Boldyreva. Vol. 10991. Lecture Notes in Computer Science. Springer, 2018, pp. 662–692. DOI: 10.1007/978-3-319-96884-1_22. URL: https://doi.org/10.1007/978-3-319-96884-1_22.

[Dod+12] Yevgeniy Dodis, Eike Kiltz, Krzysztof Pietrzak, and Daniel Wichs. "Message Authentication, Revisited." In: *EUROCRYPT*. Vol. 7237. Lecture Notes in Computer Science. Springer, 2012, pp. 355–374.

[DOV21] J.-P. D'Anvers, E. Orsini, and F. Vercauteren. "Error Term Checking: Towards Chosen Ciphertext Security without Re-encryption." In: *AsiaPKC*. 2021.

[DRV20] J.-P. D'Anvers, M. Rossi, and F. Virdia. "(One) Failure Is Not an Option: Bootstrapping the Search for Failures in Lattice-Based Encryption Schemes." In: *Eurocrypt*. 2020.

[DT14] Constantin Catalin Dragan and Ferucio Laurentiu Tiplea. "Efficient Key-policy Attribute-based Encryption for General Boolean Circuits from Multilinear Maps." In: *IACR Cryptol. ePrint Arch.* (2014), p. 462. URL: http://eprint.iacr.org/2014/462.

[Duc+17] Léo Ducas, Tancrède Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. "CRYSTALS - Dilithium: Digital Signatures from Module Lattices." In: *IACR Cryptol. ePrint Arch.* (2017), p. 633.

[Duc+18] Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. "CRYSTALS-Dilithium: A Lattice-Based Digital Signature Scheme." In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2018.1 (2018), pp. 238–268.

[Duc+21] Léo Ducas, Tancrede Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. "CRYSTALS-Dilithium Algorithm Specifications And Supporting Documentation." Version 3.1. In: (2021). 2021-02-08.

[Dum+] J. Duman, K. Hövelmanns, E. Kiltz, V. Lyubashevsky, G. Seiler, and D. Unruh. *A Thorough Treatment of Highly-Efficient NTRU Instantiations*. Cryptology ePrint Archive: Report 2021/1352.

[Duv+21] Sébastien Duval, Pierrick Méaux, Charles Momin, and François-Xavier Standaert. "Exploring Crypto-Physical Dark Matter and Learning with Physical Rounding Towards Secure and Efficient Fresh Re-Keying." In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2021.1 (2021), pp. 373–401.

[DV13] Alexandre Duc and Serge Vaudenay. "HELEN: A Public-Key Cryptosystem Based on the LPN and the Decisional Minimal Distance Problems." In: *AFRICACRYPT*. Vol. 7918. Lecture Notes in Computer Science. Springer, 2013, pp. 107–126.

[DXL14] J. Ding, X. Xie, and X. Lin. *simple provably secure key exchange scheme based on the Learning With Errors problem*. Cryptology ePrint Archive: Report 2012/688. 2014.

[Dzi+16] Stefan Dziembowski, Sebastian Faust, Gottfried Herold, Anthony Journault, Daniel Masny, and François-Xavier Standaert. "Towards Sound Fresh Re-keying with Hard (Physical) Learning Problems." In: *CRYPTO (2)*. Vol. 9815. Lecture Notes in Computer Science. Springer, 2016, pp. 272–301.

[Esp+17] Thomas Espitau, Pierre-Alain Fouque, Benoît Gérard, and Mehdi Tibouchi. "Side-Channel Attacks on BLISS Lattice-Based Signatures: Exploiting Branch Tracing against strongSwan and Electromagnetic Emanations in Microcontrollers." In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*. Ed. by Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu. ACM, 2017, pp. 1857–1874. DOI: 10.1145/3133956.3134028. URL: https://doi.org/10.1145/3133956.3134028.

[Fau02] Jean Charles Faugère. "A New Efficient Algorithm for Computing Gröbner Bases without Reduction to Zero (F5)." In: *Proceedings of the 2002 International Symposium on Symbolic and Algebraic Computation, ISSAC 2002*. 2002, 75–83.

[Fau99] Jean-Charles Faugère. "A new efficient algorithm for computing Groebner bases." In: *Journal of Pure and Applied Algebra* (1-3 1999), pp. 61–88.

[FO13] E. Fujisaki and T. Okamoto. "Secure Integration of Asymmetric and Symmetric Encryption Schemes." In: *Jo. of Cryptology* 226.21 (2013).

[FO99] Eiichiro Fujisaki and Tatsuaki Okamoto. "Secure Integration of Asymmetric and Symmetric Encryption Schemes." In: *CRYPTO*. Vol. 1666. Lecture Notes in Computer Science. Springer, 1999, pp. 537–554.

[Fri+21] Tim Fritzmann, Michiel Van Beirendonck, Debapriya Basu Roy, Patrick Karl, Thomas Schamberger, Ingrid Verbauwhede, and Georg Sigl. "Masked Accelerators and Instruction Set Extensions for Post-Quantum Cryptography." In: *IACR Cryptol. ePrint Arch.* (2021), p. 479. URL: https://eprint.iacr.org/2021/479.

[Fri+22] Tim Fritzmann, Michiel Van Beirendonck, Debapriya Basu Roy, Patrick Karl, Thomas Schamberger, Ingrid Verbauwhede, and Georg Sigl. "Masked Accelerators and Instruction Set Extensions for Post-Quantum Cryptography." In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2022.1 (2022), pp. 414–460.

[Gen09] Craig Gentry. "Fully homomorphic encryption using ideal lattices." In: *STOC*. ACM, 2009, pp. 169–178.

[GHS12] Craig Gentry, Shai Halevi, and Nigel P Smart. "Homomorphic evaluation of the AES circuit." In: *Annual Cryptology Conference*. Springer. 2012, pp. 850–867.

[Gil+23] Henri Gilbert, Rachelle Heim Boissier, Jérémy Jean, and Jean-René Reinhard. "Cryptanalysis of Elisabeth-4." In: *Advances in Cryptology - ASIACRYPT 2023 - 29th International Conference on the Theory and Application of Cryptology and Information Security, Guangzhou, China, December 4-8, 2023, Proceedings, Part III*. Ed. by Jian Guo and Ron Steinfeld. Vol. 14440. Lecture Notes in Computer Science. Springer, 2023, pp. 256–284. DOI: 10.1007/978-981-99-8727-6_9. URL: https://doi.org/10.1007/978-981-99-8727-6_9.

[GJ19] Qian Guo and Thomas Johansson. "A new birthday-type algorithm for attacking the fresh re-keying countermeasure." In: *Inf. Process. Lett.* 146 (2019), pp. 30–34.

[GLP06] Benedikt Gierlichs, Kerstin Lemke-Rust, and Christof Paar. "Templates vs. Stochastic Methods." In: *CHES*. Vol. 4249. Lecture Notes in Computer Science. Springer, 2006, pp. 15–29.

[GLS14] Lubos Gaspar, Gaëtan Leurent, and François-Xavier Standaert. "Hardware Implementation and Side-Channel Analysis of Lapin." In: *CT-RSA*. Vol. 8366. Lecture Notes in Computer Science. Springer, 2014, pp. 206–226.

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. "Trapdoors for hard lattices and new cryptographic constructions." In: *STOC*. ACM, 2008, pp. 197–206.

[GR10] Swaroop Ghosh and Kaushik Roy. "Parameter Variation Tolerance and Error Resiliency: New Design Paradigm for the Nanoscale Era." In: *Proc. IEEE* 98.10 (2010), pp. 1718–1751.

[GR17] Dahmun Goudarzi and Matthieu Rivain. "How Fast Can Higher-Order Masking Be in Software?" In: *EUROCRYPT (1)*. Vol. 10210. Lecture Notes in Computer Science. 2017, pp. 567–597.

[GRS08a] Henri Gilbert, Matthew J. B. Robshaw, and Yannick Seurin. "HB$^{\#}$: Increasing the Security and Efficiency of HB^{+}." In: *EUROCRYPT*. Vol. 4965. Lecture Notes in Computer Science. Springer, 2008, pp. 361–378.

[GRS08b] Henri Gilbert, Matthew J. B. Robshaw, and Yannick Seurin. "How to Encrypt with the LPN Problem." In: *ICALP (2)*. Vol. 5126. Lecture Notes in Computer Science. Springer, 2008, pp. 679–690.

[GS18] Vincent Grosso and François-Xavier Standaert. "Masking Proofs Are Tight and How to Exploit it in Security Evaluations." In: *EUROCRYPT (2)*. Vol. 10821. Lecture Notes in Computer Science. Springer, 2018, pp. 385–412.

[GS66] W. Morven Gentleman and G. Sande. "Fast Fourier Transforms: for fun and profit." In: *American Federation of Information Processing Societies: Proceedings of the AFIPS '66 Fall Joint Computer Conference, November 7-10, 1966, San Francisco, California, USA*. Vol. 29. AFIPS Conference Proceedings. AFIPS / ACM / Spartan Books, Washington D.C., 1966, pp. 563–578. DOI: 10.1145/1464291.

1464352. URL: https://doi.org/10.1145/1464291.1464352.

[GSW13] Craig Gentry, Amit Sahai, and Brent Waters. "Homomorphic Encryption from Learning with Errors: Conceptually-Simpler, Asymptotically-Faster, Attribute-Based." In: *CRYPTO (1)*. Vol. 8042. Lecture Notes in Computer Science. Springer, 2013, pp. 75–92.

[Guo+19] Chun Guo, Olivier Pereira, Thomas Peters, and François-Xavier Standaert. "Authenticated Encryption with Nonce Misuse and Physical Leakage: Definitions, Separation Results and First Construction - (Extended Abstract)." In: *LATINCRYPT*. 2019.

[Guo+20] Qian Guo, Vincent Grosso, François-Xavier Standaert, and Olivier Bronchain. "Modeling Soft Analytical Side-Channel Attacks from a Coding Theory Viewpoint." In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2020.4 (2020), pp. 209–238.

[Ha+22] Jincheol Ha, Seongkwang Kim, ByeongHak Lee, Jooyoung Lee, and Mincheol Son. "Rubato: Noisy Ciphers for Approximate Homomorphic Encryption." In: *Advances in Cryptology - EUROCRYPT 2022 - 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30 - June 3, 2022, Proceedings, Part I*. Ed. by Orr Dunkelman and Stefan Dziembowski. Vol. 13275. Lecture Notes in Computer Science. Springer, 2022, pp. 581–610. DOI: 10.1007/978-3-031-06944-4_20. URL: https://doi.org/10.1007/978-3-031-06944-4_20.

[Ham19] M. Hamburg. *Three Bears*. Technical report, National Institute of Standards and Technology. 2019.

[Ham+21] Mike Hamburg, Julius Hermelink, Robert Primas, Simona Samardjiska, Thomas Schamberger, Silvan Streit, Emanuele Strieder, and Christine van Vredendaal. "Chosen Ciphertext k-Trace Attacks on Masked CCA2 Secure Kyber." In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2021.4 (2021), pp. 88–113.

[Har09] David Harvey. "Faster polynomial multiplication via multipoint Kronecker substitution." In: *J. Symb. Comput.* 44.10 (2009), pp. 1502–1510. DOI: 10.1016/j.jsc.2009.05.004. URL: https://doi.org/10.1016/j.jsc.2009.05.004.

[Hås97] Johan Håstad. "Some Optimal Inapproximability Results." In: *STOC*. ACM, 1997, pp. 1–10.

[HB01] Nicholas J. Hopper and Manuel Blum. "Secure Human Identification Protocols." In: *ASIACRYPT*. Vol. 2248. Lecture Notes in Computer Science. Springer, 2001, pp. 52–66.

[Hei+21] Daniel Heinz, Matthias J. Kannwischer, Georg Land, Thomas Pöppelmann, Peter Schwabe, and Daan Sprenkels. *First-Order Masked Kyber on ARM Cortex-M4 (work in progress)*. 2021. URL: https://csrc.nist.gov/CSRC/media/Presentations/first-order-masked-kyber-on-arm-cortex-m4/images-media/session-4-heinz-first-order-masked-kyber.pdf.

[Hey+12] Stefan Heyse, Eike Kiltz, Vadim Lyubashevsky, Christof Paar, and Krzysztof Pietrzak. "Lapin: An Efficient Authentication Protocol Based on Ring-LPN." In: *FSE*. Vol. 7549. Lecture Notes in Computer Science. Springer, 2012, pp. 346–365.

[HHK17a] D. Hofheinz, K. Hövelmanns, and E. Kiltz. "A Modular Analysis of the Fujisaki-Okamoto Transformation." In: *TCC*. 2017.

[HHK17b] Dennis Hofheinz, Kathrin Hövelmanns, and Eike Kiltz. "A Modular Analysis of the Fujisaki-Okamoto Transformation." In: *TCC (1)*. Vol. 10677. Lecture Notes in Computer Science. Springer, 2017, pp. 341–371.

[HHM22] Kathrin Hövelmanns, Andreas Hülsing, and Christian Majenz. "Failing gracefully: Decryption failures and the Fujisaki-Okamoto transform." In: *IACR Cryptol. ePrint Arch.* (2022), p. 365.

[HMR20] Clément Hoffmann, Pierrick Méaux, and Thomas Ricosset. "Transciphering, Using FiLIP and TFHE for an Efficient Delegation of Computation." In: *Progress in Cryptology - INDOCRYPT 2020 - 21st International Conference on Cryptology in India, Bangalore, India, December 13-16, 2020, Proceedings*. Ed. by Karthikeyan Bhargavan, Elisabeth Oswald, and Manoj Prabhakaran. Vol. 12578. Lecture Notes in Computer Science. Springer, 2020, pp. 39–61. DOI: 10.1007/978-3-030-65277-7_3. URL: https://doi.org/10.1007/978-3-030-65277-7_3.

[HMS23] Clément Hoffmann, Pierrick Méaux, and François-Xavier Standaert. "The Patching Landscape of Elisabeth-4 and the Mixed Filter Permutator Paradigm." In: *Progress in Cryptology - INDOCRYPT 2023 - 24th International Conference on Cryptology in India, Goa, India, December 10-13, 2023, Proceedings, Part I*. Ed. by Anupam Chattopadhyay, Shivam Bhasin, Stjepan Picek, and Chester Rebeiro. Vol. 14459. Lecture Notes in Computer Science. Springer, 2023, pp. 134–156. DOI: 10.1007/978-3-031-56232-7_7. URL: https://doi.org/10.1007/978-3-031-56232-7_7.

[Hof+22] Clément Hoffmann, Benoît Libert, Charles Momin, Thomas Peters, and François-Xavier Standaert. *Towards Leakage-Resistant Post-Quantum CCA-Secure Public Key Encryption*. Cryptology ePrint Archive, Paper 2022/873. https://eprint.iacr.org/2022/873. 2022. URL: https://eprint.iacr.org/2022/873.

[Hof+23] Clément Hoffmann, Pierrick Méaux, Charles Momin, Yann Rotella, François-Xavier Standaert, and Balazs Udvarhelyi. "Learning with Physical Rounding for Linear and Quadratic Leakage Functions." In: *Advances in Cryptology - CRYPTO 2023 - 43rd Annual International Cryptology Conference, CRYPTO 2023, Santa Barbara, CA, USA, August 20-24, 2023, Proceedings, Part III*. Ed. by Helena Handschuh and Anna Lysyanskaya. Vol. 14083. Lecture Notes in Computer Science. Springer, 2023, pp. 410–439. DOI: 10.1007/978-3-031-38548-3_14. URL: https://doi.org/10.1007/978-3-031-38548-3_14.

[HOM06] Christoph Herbst, Elisabeth Oswald, and Stefan Mangard. "An AES Smart Card Implementation Resistant to Power Analysis Attacks." In: *ACNS*. Vol. 3989. Lecture Notes in Computer Science. 2006, pp. 239–252.

[Hou18] Xiang-dong Hou. *Lectures on Finite Fields*. Vol. 190. American Mathematical Society, 2018.

[HPS98] J. Hoffstein, J. Pipher, and J. H. Silverman. "NTRU: A ring-based public key cryptosystem." In: *ANTS*. 1998.

[HS01] Nick Howgrave-Graham and Nigel P. Smart. "Lattice Attacks on Digital Signature Schemes." In: *Des. Codes Cryptogr.* 23.3 (2001), pp. 283–290.

[HS14] Shai Halevi and Victor Shoup. "Algorithms in HElib." In: *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I*. Ed. by Juan A. Garay and Rosario Gennaro. Vol. 8616. Lecture Notes in Computer Science. Springer, 2014, pp. 554–571. DOI: 10.1007/978-3-662-44371-2_31. URL: https://doi.org/10.1007/978-3-662-44371-2_31.

[HV20] Loïs Huguenin-Dumittan and Serge Vaudenay. "Classical Misuse Attacks on NIST Round 2 PQC - The Power of Rank-Based Schemes." In: *Applied Cryptography and Network Security - 18th International Conference, ACNS 2020, Rome, Italy, October 19-22, 2020, Proceedings, Part I*. Ed. by Mauro Conti, Jianying Zhou, Emiliano Casalicchio, and Angelo Spognardi. Vol. 12146. Lecture Notes in Computer Science. Springer, 2020, pp. 208–227. DOI: 10.1007/978-3-030-57808-4_11. URL: https://doi.org/10.1007/978-3-030-57808-4_11.

[ISW03] Yuval Ishai, Amit Sahai, and David A. Wagner. "Private Circuits: Securing Hardware against Probing Attacks." In: *CRYPTO*. Vol. 2729. Lecture Notes in Computer Science. Springer, 2003, pp. 463–481.

[JLS20] Aayush Jain, Huijia Lin, and Amit Sahai. "Indistinguishability Obfuscation from Well-Founded Assumptions." In: *IACR Cryptol. ePrint Arch.* 2020 (2020), p. 1003. URL: https://eprint.iacr.org/2020/1003.

[JLS24] Aayush Jain, Huijia Lin, and Sagnik Saha. "A Systematic Study of Sparse LWE." In: *Advances in Cryptology - CRYPTO 2024 - 44th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2024, Proceedings, Part III*. Ed. by Leonid Reyzin and Douglas Stebila. Vol. 14922. Lecture Notes in Computer Science. Springer, 2024, pp. 210–245. DOI: 10.1007/978-3-031-68382-4_7. URL: https://doi.org/10.1007/978-3-031-68382-4_7.

[JW05] Ari Juels and Stephen A. Weis. "Authenticating Pervasive Devices with Human Protocols." In: *CRYPTO*. Vol. 3621. Lecture Notes in Computer Science. Springer, 2005, pp. 293–308.

[Kam+18] Dina Kamel, Davide Bellizia, François-Xavier Standaert, Denis Flandre, and David Bol. "Demonstrating an LPPN Processor." In: *ASHESCCS*. ACM, 2018, pp. 18–23.

[Kam+20a] Dina Kamel, Davide Bellizia, Olivier Bronchain, and François-Xavier Standaert. "Side-channel analysis of a learning parity with physical noise processor." In: *J. Cryptogr. Eng.* (2020), pp. 1–9.

[Kam+20b] Dina Kamel, François-Xavier Standaert, Alexandre Duc, Denis Flandre, and Francesco Berti. "Learning with Physical Noise or Errors." In: *IEEE Trans. Dependable Secur. Comput.* 17.5 (2020), pp. 957–971. DOI: 10.1109/TDSC.2018.2830763. URL: https://doi.org/10.1109/TDSC.2018.2830763.

[Kan+] Matthias J. Kannwischer, Joost Rijneveld, Peter Schwabe, and Ko Stoffelen. *PQM4: Post-quantum crypto library for the ARM Cortex-M4*. https://github.com/mupq/pqm4.

[Kea93] Michael J. Kearns. "Efficient noise-tolerant learning from statistical queries." In: *STOC*. ACM, 1993, pp. 392–401.

[KF15] P. Kirchner and P.-A. Fouque. "An Improved BKW Algorithm for LWE with Applications to Cryptography and Lattices." In: *Crypto*. 2015.

[Kil+11] Eike Kiltz, Krzysztof Pietrzak, David Cash, Abhishek Jain, and Daniele Venturi. "Efficient Authentication from Hard Learning Problems." In: *EUROCRYPT*. Vol. 6632. Lecture Notes in Computer Science. Springer, 2011, pp. 7–26.

[Kil+17] Eike Kiltz, Krzysztof Pietrzak, Daniele Venturi, David Cash, and Abhishek Jain. "Efficient Authentication from Hard Learning Problems." In: *J. Cryptol.* 30.4 (2017), pp. 1238–1275.

[KL14] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography, Second Edition*. CRC Press, 2014. ISBN: 9781466570269. URL: https://www.crcpress.com/Introduction-to-Modern-Cryptography-Second-Edition/Katz-Lindell/p/book/9781466570269.

[KO63] Anatoly A. Karatsuba and Y. Ofman. "Multiplication of multidigit numbers on automata." In: *Soviet Physics Doklady* 7 (1963). URL: http://cr.yp.to/bib/entries.html#1963/karatsuba, pp. 595–596. ISSN: 0038–5689.

[Kop+17] Philipp Koppermann, Fabrizio De Santis, Johann Heyszl, and Georg Sigl. "Automatic generation of high-performance modular multipliers for arbitrary mersenne primes on FPGAs." In: *HOST*. IEEE Computer Society, 2017, pp. 35–40.

[KPP20] Matthias J. Kannwischer, Peter Pessl, and Robert Primas. "Single-Trace Attacks on Keccak." In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2020.3 (2020), pp. 243–268.

[KW52] William H. Kruskal and W. Allen Wallis. "Use of Ranks in One-Criterion Variance Analysis." In: *Journal of the American Statistical Association* 47.260 (1952), pp. 583–621. DOI: 10.1080/01621459.1952.10483441. eprint: https://www.tandfonline.com/doi/pdf/10.1080/01621459.1952.10483441. URL: https://www.tandfonline.com/doi/abs/10.1080/01621459.1952.10483441.

[KY16] S. Katsumata and S. Yamada. "Partitioning via Non-linear Polynomial Functions: More Compact IBEs from Ideal Lattices and Bilinear Maps." In: *Asiacrypt*. 2016.

[LATV12] A. Lopez-Alt, E. Tromer, and V. Vaikuntanathan. "On-the-Fly Multiparty Computation on the Cloud via Multikey Fully Homomorphic Encryption." In: *STOC*. 2012.

[LF06] Éric Levieil and Pierre-Alain Fouque. "An Improved LPN Algorithm." In: *SCN*. Vol. 4116. Lecture Notes in Computer Science. Springer, 2006, pp. 348–359.

[Liu+21] Yuejun Liu, Yongbin Zhou, Shuo Sun, Tianyu Wang, Rui Zhang, and Jingdian Ming. "On the Security of Lattice-Based Fiat-Shamir Signatures in the Presence of Randomness Leakage." In: *IEEE Trans. Inf. Forensics Secur.* 16 (2021), pp. 1868–1879. DOI: 10.1109/TIFS.2020.3045904. URL: https://doi.org/10.1109/TIFS.2020.3045904.

[LN16] Patrick Longa and Michael Naehrig. "Speeding up the Number Theoretic Transform for Faster Ideal Lattice-Based Cryptography." In: *Cryptology and Network Security - 15th International Conference, CANS 2016, Milan, Italy, November 14-16, 2016, Proceedings*. Ed. by Sara Foresti and Giuseppe Persiano. Vol. 10052. Lecture Notes in Computer Science. 2016, pp. 124–139. DOI: 10.1007/978-3-319-48965-0_8. URL: https://doi.org/10.1007/978-3-319-48965-0_8.

[LPR10] V. Lyubashevsky, C. Peikert, and O. Regev. "On Ideal Lattices and Learning with Errors Over Rings." In: *Eurocrypt.* 2010.

[LPR12] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. "On Ideal Lattices and Learning with Errors Over Rings." In: *IACR Cryptol. ePrint Arch.* (2012), p. 230. URL: http://eprint.iacr.org/2012/230.

[LS12] Adeline Langlois and Damien Stehlé. "Worst-Case to Average-Case Reductions for Module Lattices." In: *IACR Cryptol. ePrint Arch.* (2012), p. 90. URL: http://eprint.iacr.org/2012/090.

[LS14] A. Langlois and D. Stehlé. "Worst-case to average-case reductions for module lattices." In: *Designs, Code & Cryptography* (2014).

[LS15] Adeline Langlois and Damien Stehlé. "Worst-case to average-case reductions for module lattices." In: *Des. Codes Cryptogr.* 75.3 (2015), pp. 565–599.

[Lu+19] X. Lu et al. *LAC*. Technical report, National Institute of Standards and Technology. 2019.

[Lyu09] Vadim Lyubashevsky. "Fiat-Shamir with Aborts: Applications to Lattice and Factoring-Based Signatures." In: *Advances in Cryptology - ASIACRYPT 2009, 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6-10, 2009. Proceedings*. Ed. by Mitsuru Matsui. Vol. 5912. Lecture Notes in Computer Science. Springer, 2009, pp. 598–616. DOI: 10.1007/978-3-642-10366-7_35. URL: https://doi.org/10.1007/978-3-642-10366-7_35.

[Man04] Stefan Mangard. "Hardware Countermeasures against DPA ? A Statistical Analysis of Their Effectiveness." In: *CT-RSA*. Vol. 2964. Lecture Notes in Computer Science. Springer, 2004, pp. 222–235.

[Mar+22] Soundes Marzougui, Vincent Ulitzsch, Mehdi Tibouchi, and Jean-Pierre Seifert. "Profiling Side-Channel Attacks on Dilithium: A Small Bit-Fiddling Leak Breaks It All." In: *IACR Cryptol. ePrint Arch.* (2022), p. 106. URL: https://eprint.iacr.org/2022/106.

[Mas+23] Loïc Masure, Pierrick Méaux, Thorben Moos, and François-Xavier Standaert. "Effective and Efficient Masking with Low Noise Using Small-Mersenne-Prime Ciphers." In: *EUROCRYPT (4)*. Vol. 14007. Lecture Notes in Computer Science. Springer, 2023, pp. 596–627.

[Méa+19] Pierrick Méaux, Claude Carlet, Anthony Journault, and François-Xavier Standaert. "Improved Filter Permutators for Efficient FHE: Better Instances and Implementations." In: *Progress in Cryptology - INDOCRYPT 2019 - 20th International Conference on Cryptology in India, Hyderabad, India, December 15-18, 2019, Proceedings*. Ed. by Feng Hao, Sushmita Ruj, and Sourav Sen Gupta. Vol. 11898. Lecture Notes in Computer Science. Springer, 2019, pp. 68–91. DOI: 10.1007/978-3-030-35423-7_4. URL: https://doi.org/10.1007/978-3-030-35423-7_4.

[Med+10] Marcel Medwed, François-Xavier Standaert, Johann Großschädl, and Francesco Regazzoni. "Fresh Re-keying: Security against Side-Channel and Fault Attacks for Low-Cost Devices." In: *AFRICACRYPT*. Vol. 6055. Lecture Notes in Computer Science. Springer, 2010, pp. 279–296.

[Mig+19] Vincent Migliore, Benoît Gérard, Mehdi Tibouchi, and Pierre-Alain Fouque. "Masking Dilithium - Efficient Implementation and Side-Channel Evaluation." In: *ACNS*. Vol. 11464. Lecture Notes in Computer Science. Springer, 2019, pp. 344–362.

[MKV20] Jose Maria Bermudo Mera, Angshuman Karmakar, and Ingrid Verbauwhede. "Time-memory trade-off in Toom-Cook multiplication: an application to module-lattice based cryptography." In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2020.2 (2020), pp. 222–244. DOI: 10.13154/tches.v2020.i2.222-244. URL: https://doi.org/10.13154/tches.v2020.i2.222-244.

[MNS05] M. Maymandi-Nejad and M. Sachdev. "A Monotonic Digitally Controlled Delay Element." In: *IEEE Journal of Solid-State Circuits* 40.11 (2005), pp. 2212–2219. DOI: 10.1109/JSSC.2005.857370.

[Mon85] Peter L. Montgomery. "Modular multiplication without trial division." In: *Mathematics of Computation* 44 (1985), pp. 519–521. ISSN: 0025–5718.

[MOP07] Stefan Mangard, Elisabeth Oswald, and Thomas Popp. *Power analysis attacks - revealing the secrets of smart cards*. Springer, 2007.

[MOS11] Stefan Mangard, Elisabeth Oswald, and François-Xavier Standaert. "One for all - all for one: unifying standard differential power analysis attacks." In: *IET Inf. Secur.* 5.2 (2011), pp. 100–110.

[MP13] Daniele Micciancio and Chris Peikert. "Hardness of SIS and LWE with Small Parameters." In: *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I*. Ed. by Ran Canetti and Juan A. Garay. Vol. 8042. Lecture Notes in Computer Science. Springer, 2013, pp. 21–39. DOI: 10.1007/978-3-642-40041-4_2. URL: https://doi.org/10.1007/978-3-642-40041-4_2.

[MPG05] Stefan Mangard, Thomas Popp, and Berndt M. Gammel. "Side-Channel Leakage of Masked CMOS Gates." In: *CT-RSA*. Vol. 3376. Lecture Notes in Computer Science. Springer, 2005, pp. 351–365.

[MR07] D. Micciancio and O. Regev. "Worst-case to Average-case Reductions based on Gaussian Measures." In: *SIAMJC* 37.1 (2007), pp. 267–302.

[Nat] National Institute of Standards and Technology. *Post-Quantum Cryptography Standardization*. https://csrc.nist.gov/Projects/Post-Quantum-Cryptography/Post-Quantum-Cryptography-Standardization.

[Ngo+21] Kalle Ngo, Elena Dubrova, Qian Guo, and Thomas Johansson. "A Side-Channel Attack on a Masked IND-CCA Secure Saber KEM Implementation." In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2021.4 (2021), pp. 676–707. DOI: 10.46586/tches.v2021.i4.676-707. URL: https://doi.org/10.46586/tches.v2021.i4.676-707.

[NRS11] Svetla Nikova, Vincent Rijmen, and Martin Schläffer. "Secure Hardware Implementation of Nonlinear Functions in the Presence of Glitches." In: *J. Cryptol.* 24.2 (2011), pp. 292–321.

[PBY17] Peter Pessl, Leon Groot Bruinderink, and Yuval Yarom. "To BLISS-B or not to be: Attacking strongSwan's Implementation of Post-Quantum Signatures." In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*. Ed. by Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu. ACM, 2017, pp. 1843–1855. DOI: 10.1145/3133956.3134023. URL: https://doi.org/10.1145/3133956.3134023.

[Pei09] Chris Peikert. "Public-key cryptosystems from the worst-case shortest vector problem: extended abstract." In: *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*. Ed. by Michael Mitzenmacher. ACM, 2009, pp. 333–342. DOI: 10.1145/1536414.1536461. URL: https://doi.org/10.1145/1536414.1536461.

[Pei14] C. Peikert. "Lattice Cryptography for the Internet." In: *PQCrypto*. 2014.

[Per12] E. Persichetti. "Improving the efficiency of code-based cryptography." PhD thesis. Univ. of Auckland, 2012.

[Pie12] Krzysztof Pietrzak. "Cryptography from Learning Parity with Noise." In: *SOFSEM*. Vol. 7147. Lecture Notes in Computer Science. Springer, 2012, pp. 99–114.

[PM16] Peter Pessl and Stefan Mangard. "Enhancing Side-Channel Analysis of Binary-Field Multiplication with Bit Reliability." In: *CT-RSA*. Vol. 9610. Lecture Notes in Computer Science. Springer, 2016, pp. 255–270.

[PMD21] A. Pellet-Mary and Stehlé D. "On the hardness of the NTRU problem." In: *Asiacrypt*. 2021.

[PP19] Peter Pessl and Robert Primas. "More Practical Single-Trace Attacks on the Number Theoretic Transform." In: *LATINCRYPT*. Vol. 11774. Lecture Notes in Computer Science. Springer, 2019, pp. 130–149.

[PPM17] Robert Primas, Peter Pessl, and Stefan Mangard. "Single-Trace Side-Channel Attacks on Masked Lattice-Based Encryption." In: *CHES*. Vol. 10529. Lecture Notes in Computer Science. Springer, 2017, pp. 513–533.

[PR13] Emmanuel Prouff and Matthieu Rivain. "Masking against Side-Channel Attacks: A Formal Security Proof." In: *EUROCRYPT*. Vol. 7881. Lecture Notes in Computer Science. Springer, 2013, pp. 142–159.

[Pre17] Thomas Prest. "Sharper bounds in lattice-based cryptography using the Rényi divergence." In: *Advances in Cryptology–ASIACRYPT 2017: 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I 23*. Springer. 2017, pp. 347–374.

[Rav+18] Prasanna Ravi, Mahabir Prasad Jhanwar, James Howe, Anupam Chattopadhyay, and Shivam Bhasin. "Side-channel Assisted Existential Forgery Attack on Dilithium - A NIST PQC candidate." In: *IACR Cryptol. ePrint Arch.* (2018), p. 821. URL: https://eprint.iacr.org/2018/821.

[Rav+20a] Prasanna Ravi, Romain Poussier, Shivam Bhasin, and Anupam Chattopadhyay. "On Configurable SCA Countermeasures Against Single Trace Attacks for the NTT - A Performance Evaluation Study over Kyber and Dilithium on the ARM Cortex-M4." In: *SPACE*. Vol. 12586. Lecture Notes in Computer Science. Springer, 2020, pp. 123–146.

[Rav+20b] Prasanna Ravi, Sujoy Sinha Roy, Anupam Chattopadhyay, and Shivam Bhasin. "Generic Side-channel attacks on CCA-secure lattice-based PKE and KEMs." In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2020.3 (2020), pp. 307–335.

[Rav+22] Prasanna Ravi, Martianus Frederic Ezerman, Shivam Bhasin, Anupam Chattopadhyay, and Sujoy Sinha Roy. "Will You Cross the Threshold for Me? Generic Side-Channel Assisted Chosen-Ciphertext Attacks on NTRU-based KEMs." In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2022.1 (2022), pp. 722–761.

[Reg05a] O. Regev. "On lattices, learning with errors, random linear codes, and cryptography." In: *STOC*. 2005.

[Reg05b] Oded Regev. "On lattices, learning with errors, random linear codes, and cryptography." In: *Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, May 22-24, 2005*. Ed. by Harold N. Gabow and Ronald Fagin. ACM, 2005, pp. 84–93. DOI:

10.1145/1060590.1060603. URL: https://doi.org/10.1145/1060590.1060603.

[Reg09a] Oded Regev. "On lattices, learning with errors, random linear codes, and cryptography." In: *J. ACM* 56.6 (2009), 34:1–34:40. DOI: 10.1145/1568318.1568324. URL: http://doi.acm.org/10.1145/1568318.1568324.

[Reg09b] Oded Regev. "On lattices, learning with errors, random linear codes, and cryptography." In: *Journal of the ACM (JACM)* 56.6 (2009), pp. 1–40.

[Reg10] Oded Regev. "The Learning with Errors Problem (Invited Survey)." In: *Computational Complexity Conference*. IEEE Computer Society, 2010, pp. 191–204.

[Rep+15] Oscar Reparaz, Sujoy Sinha Roy, Frederik Vercauteren, and Ingrid Verbauwhede. "A Masked Ring-LWE Implementation." In: *Cryptographic Hardware and Embedded Systems - CHES 2015 - 17th International Workshop, Saint-Malo, France, September 13-16, 2015, Proceedings*. Ed. by Tim Güneysu and Helena Handschuh. Vol. 9293. Lecture Notes in Computer Science. Springer, 2015, pp. 683–702. DOI: 10.1007/978-3-662-48324-4_34. URL: https://doi.org/10.1007/978-3-662-48324-4_34.

[Rep+16a] Oscar Reparaz, Ruan de Clercq, Sujoy Sinha Roy, Frederik Vercauteren, and Ingrid Verbauwhede. "Additively Homomorphic Ring-LWE Masking." In: *Post-Quantum Cryptography - 7th International Workshop, PQCrypto 2016, Fukuoka, Japan, February 24-26, 2016, Proceedings*. Ed. by Tsuyoshi Takagi. Vol. 9606. Lecture Notes in Computer Science. Springer, 2016, pp. 233–244. DOI: 10.1007/978-3-319-29360-8_15. URL: https://doi.org/10.1007/978-3-319-29360-8_15.

[Rep+16b] Oscar Reparaz, Sujoy Sinha Roy, Ruan de Clercq, Frederik Vercauteren, and Ingrid Verbauwhede. "Masking ring-LWE." In: *J. Cryptogr. Eng.* 6.2 (2016), pp. 139–153. DOI: 10.1007/s13389-016-0126-5. URL: https://doi.org/10.1007/s13389-016-0126-5.

[Sch+19] Tobias Schneider, Clara Paglialonga, Tobias Oder, and Tim Güneysu. "Efficiently Masking Binomial Sampling at Arbitrary Orders for Lattice-Based Crypto." In: *Public Key Cryptography (2)*. Vol. 11443. Lecture Notes in Computer Science. Springer, 2019, pp. 534–564.

[Sho01] V. Shoup. *A proposal for an ISO standard for public key encryption*. Manuscript. Dec. 2001.

[SLP05] Werner Schindler, Kerstin Lemke, and Christof Paar. "A Stochastic Model for Differential Side Channel Cryptanalysis." In: *CHES*. Vol. 3659. Lecture Notes in Computer Science. Springer, 2005, pp. 30–46.

[SMY09] François-Xavier Standaert, Tal Malkin, and Moti Yung. "A Unified Framework for the Analysis of Side-Channel Key Recovery Attacks." In: *EUROCRYPT*. Vol. 5479. Lecture Notes in Computer Science. Springer, 2009, pp. 443–461.

[SS11] D. Stehlé and R. Steinfeld. "Making NTRU as Secure as Worst-Case Problems over Ideal Lattices." In: *Eurocrypt*. 2011.

[Sta+06] François-Xavier Standaert, Eric Peeters, Gaël Rouvroy, and Jean-Jacques Quisquater. "An Overview of Power Analysis Attacks Against Field Programmable Gate Arrays." In: *Proc. IEEE* 94.2 (2006), pp. 383–394.

[Sta19] François-Xavier Standaert. "Towards and Open Approach to Secure Cryptographic Implementations (Invited Talk)." In: *EUROCRYPT I*. Vol. 11476. LNCS. 2019, xv, `https://www.youtube.com/watch?v=KdhrsuJT1sE`.

[Sun+22] Chao Sun, Thomas Espitau, Mehdi Tibouchi, and Masayuki Abe. "Guessing Bits: Improved Lattice Attacks on (EC)DSA with Nonce Leakage." In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2022.1 (2022), pp. 391–413. DOI: `10.46586/tches.v2022.i1.391-413`. URL: `https://doi.org/10.46586/tches.v2022.i1.391-413`.

[SXY18] T. Saito, K. Xagawa, and Y. Yamakawa. "Tightly-Secure Key-Encapsulation Mechanism in the Quantum Random Oracle Model." In: *Eurocrypt*. 2018.

[Too63] Andrei L. Toom. "The complexity of a scheme of functional elements realizing the multiplication of integers." In: *Soviet Mathematics Doklady* 3 (1963). http://www.de.ufpe.br/ toom/my-articles/engmat/MULT-E.PDF, pp. 714–716. ISSN: 0197–6788.

[UBS21] Balazs Udvarhelyi, Olivier Bronchain, and François-Xavier Standaert. "Security Analysis of Deterministic Re-keying with Masking and Shuffling: Application to ISAP." In:

COSADE. Vol. 12910. Lecture Notes in Computer Science. Springer, 2021, pp. 168–183.

[Uen+21] Rei Ueno, Keita Xagawa, Yutaro Tanaka, Akira Ito, Junko Takahashi, and Naofumi Homma. "Curse of Re-encryption: A Generic Power/EM Analysis on Post-Quantum KEMs." In: *IACR Cryptol. ePrint Arch.* (2021), p. 849. URL: https://eprint.iacr.org/2021/849.

[Uen+22] Rei Ueno, Keita Xagawa, Yutaro Tanaka, Akira Ito, Junko Takahashi, and Naofumi Homma. "Curse of Re-encryption: A Generic Power/EM Analysis on Post-Quantum KEMs." In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2022.1 (2022), pp. 296–322.

[Vey+12] Nicolas Veyrat-Charvillon, Marcel Medwed, Stéphanie Kerckhof, and François-Xavier Standaert. "Shuffling against Side-Channel Attacks: A Comprehensive Study with Cautionary Note." In: *ASIACRYPT*. Vol. 7658. Lecture Notes in Computer Science. Springer, 2012, pp. 740–757.

[VGS14] Nicolas Veyrat-Charvillon, Benoît Gérard, and François-Xavier Standaert. "Soft Analytical Side-Channel Attacks." In: *ASIACRYPT (1)*. Vol. 8873. Lecture Notes in Computer Science. Springer, 2014, pp. 282–296.

[XL21] Yufei Xing and Shuguo Li. "A Compact Hardware Implementation of CCA-Secure Key Exchange Mechanism CRYSTALS-KYBER on FPGA." In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2021.2 (2021), pp. 328–356. DOI: 10.46586/tches.v2021.i2.328-356. URL: https://doi.org/10.46586/tches.v2021.i2.328-356.

[Xu+20] Zhuang Xu, Owen Pemberton, Sujoy Sinha Roy, and David F. Oswald. "Magnifying Side-Channel Leakage of Lattice-Based Cryptosystems with Chosen Ciphertexts: The Case Study of Kyber." In: *IACR Cryptol. ePrint Arch.* 2020 (2020), p. 912.

[YC04] Bo-Yin Yang and Jiun-Ming Chen. "All in the XL Family: Theory and Practice." In: *Information Security and Cryptology - ICISC 2004, 7th International Conference, Seoul, Korea, December 2-3, 2004, Revised Selected Papers*. Ed. by Choonsik Park and Seongtaek Chee. Vol. 3506. Lecture Notes in Computer Science. Springer, 2004, pp. 67–86.

[YS16] Yu Yu and John P. Steinberger. "Pseudorandom Functions in Almost Constant Depth from Low-Noise LPN." In: *EUROCRYPT (2)*. Vol. 9666. Lecture Notes in Computer Science. Springer, 2016, pp. 154–183.

[Yu+19] Yu Yu, Jiang Zhang, Jian Weng, Chun Guo, and Xiangxue Li. "Collision Resistant Hashing from Sub-exponential Learning Parity with Noise." In: *ASIACRYPT (2)*. Vol. 11922. Lecture Notes in Computer Science. Springer, 2019, pp. 3–24.