`POLKA`: Towards Leakage-Resistant Post-Quantum CCA-Secure Public Key Encryption

Clément Hoffmann[1], Benoît Libert[2⋆], Charles Momin[1], Thomas Peters[1], François-Xavier Standaert[1]

[1] Crypto Group, ICTEAM Institute, UCLouvain, Louvain-la-Neuve, Belgium.
[2] Zama, France.

Abstract. As for any cryptographic algorithm, the deployment of post-quantum CCA-secure public key encryption schemes may come with the need to be protected against side-channel attacks. For existing post-quantum schemes that have not been developed with leakage in mind, recent results showed that the cost of these protections can make their implementations more expensive by orders of magnitude. In this paper, we describe a new design, coined `POLKA`, that is specifically tailored to reduce this cost. It leverages various ingredients in order to enable efficient side-channel protected implementations such as: (i) the rigidity property (which intuitively means that the 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 problems 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, the operations of `POLKA` can be protected against leakage in a cheaper way than state-of-the-art designs, opening the way towards schemes that are both quantum-safe and leakage-resistant.

1 Introduction

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) [42, 43]. The FO transformation and its variants are, for example, used in the NIST post-quantum finalists `KYBER` [4, 20]

⋆ This work was done when this author was a CNRS researcher at Laboratoire LIP (UMR CNRS - ENS Lyon - UCB Lyon 1 - INRIA 5668), Lyon, France.

and `SABER` [10, 31], 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 [63], 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 [63]. 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 of its choice [57, 60, 64], 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 $c_{kem} \overset{?}{=} \mathsf{Encaps}_{pk}(m, r)$ 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) [53].

In parallel, several pieces of work started to analyze masked implementations of `KYBER` and `SABER` [11, 17, 21, 22, 41]. 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 [5]. 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 [14].

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. For this purpose, we propose to combine the different seed ingredients. First, we leverage the *rigidity* property introduced by Bernstein and

Persichetti [15], 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 [5]. 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.[1] 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 [36, 40, 55]. 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 [39]. In short, the LWPR assumption is a physical version of the crypto dark matter introduced by Boneh *et al.* [19]. 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 of the DEM as an Encrypt-then-MAC scheme 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 as well. 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) [18]. 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) [52], 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 de-randomizing an IND-CPA system, 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

[1] 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.

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 [15] 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, 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 [14] 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. On the other hand, we describe a hardware architecture for these masked operations, which confirms these excellent features (e.g., simplicity to implement them securely, performance overheads that are linear in the number of shares). Overall, protecting the long-term secret of our leveled 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, which is directly obtained thanks to parallelism in hardware. Protecting the message confidentiality additionally requires protecting its symmetric cryptographic components (i.e., hash function and authenticated encryption) against DPA.

As a last result to confirm the generality of our findings, we also show in the ePrint report [46] that they apply to an LR variant of the NTRU cryptosystem [27], which satisfies the rigidity property, can be enhanced with a dummy mechanism and has internal computations that also generate LWPR samples.

2 Technical Overview & Cautionary Note

Technical overview. Our construction can be seen as a rigid and randomness-recovering version of the RLWE-based encryption scheme described in [52]. 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 [15] 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 [47]), this generally introduces one-more place of potential side-channel vulnerabilities, which is precisely exploited in [57,60,64]. 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 in order to remain 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 $\mu = c_2 - c_1 \cdot s \mod q = p^2 er - pe_1 s + e_2$. 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 [32], 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 [32]) is non-standard and its hardness is not known to be implied by RLWE.[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.[3] While our scheme could have relied on implicit rejection in a similar way, we chose to avoid additional computations involving an extra long-term secret z. The reason is that, if we were to introduce additional key material z, it should be DPA-protected with possibly heavy side-channel countermeasures (cf. Section 5.1).

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 modified decryption oracle is indistinguishable from the real one. After having modified the decryption oracle, we can adapt ideas from Saito *et al.* [62] in order to tightly relate the security of the hybrid scheme to the RLWE assumption.

[2] D'Anvers *et al.* [32] 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 $c_2 = c_2'$.

[3] 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 $\perp$.

As mentioned earlier, avoiding re-encryption does not suffice to ensure side-channel resistance. As another improvement, 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 5 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 [14] are a natural option. But we show an even cheaper one that leverages a key-homomorphic MAC.

Cautionary note. Advances in leakage-resistant cryptography usually combine progresses following two main movements. On the one hand, theoretical works aim to specify sufficient conditions of security in abstract models. On the other hand, practical works rather aim to study heuristic countermeasures against concrete attacks (i.e., necessary security conditions). The long-term goal of such researches is therefore to "meet in the middle", which can occur either by making sufficient security conditions empirically falsifiable or by making the heuristic study of countermeasures more and more general. Reaching this goal is challenging due to the continuous and device-specific nature of physical leakages. In the case of symmetric cryptography, such movements are for example witnessed by definitional efforts like [45] and instances of (initially) more heuristic designs like ISAP [34] or Ascon [35], while their match has been recently discussed in [14]. In the case of asymmetric post-quantum cryptography, it is expected to be even more challenging since algorithms come with more versatile building blocks that will in turn require a finer-grain analysis (than just relying on block ciphers or permutations, for example). Given the amplitude of the challenge, the approach we follow is a bottom-up one. That is, we aim to show that considering the need for side-channel countermeasures as a design criterion can lead to encryption schemes that are easier to protect. For this purpose, we focus our analysis on intuitive design tweaks (for which we can explain how they avoid certain attack vectors) and on their necessary security conditions. We hope the design of POLKA can serve as a trigger for more formal analyses leading to identify sufficient security conditions for (part of) its design or improvements thereof, and that this formal analysis will be easier than for encryption schemes that did not consider leakage to guide their design, like KYBER or SABER.

In this context, one can naturally wonder why we do not provide a comprehensive comparison of POLKA with KYBER or SABER. The short answer is that the current state-of-the-art does not allow such comparisons yet. That is, a sound comparison between post-quantum encryption schemes against side-channel attacks would require assessing their cost vs. security tradeoff. But while there are several works that evaluate the (high) cost of masking KYBER or SABER, none of them come with a quantitative security evaluation against worst-case adversaries (e.g., as done in [24] for the AES). Therefore, we are for now left with the more quantitative analysis of Section 5.1, where we identify the parts of POLKA that must be protected against DPA and the ones that only require protections against SPA, together with the observation of Section 5.2 and 5.3 that the DPA security of some critical operations in POLKA can be obtained with overheads that scale linearly in the number of shares (vs. quadratic for KYBER or SABER). As in the context of symmetric cryptography, it is naturally expected that POLKA comes with overheads in case leakage is not a concern. However, considering for simplicity that they are dominated by the cost of the NTTs and multiplications, the larger polynomials (e.g., $n = 1024$ vs. $n = 512$ for KYBER) and modulus (e.g., 16-bit vs. 12-bit for KYBER) of POLKA should not decrease performances by large factors. For example, assuming NTTs have complexity in $\mathcal{O}(n \log(n))$ and multiplications have complexity in $\mathcal{O}(\log(q)^2)$, while taking into account the number of such operations, the factor of overheads of POLKA over KYBER would be around two. Besides, POLKA makes a sparser use of symmetric cryptography and its design (without FO transform) should require less shares for its masked implementations to provide the same security level. We informally illustrate the cycle counts of POLKA and KYBER in function of the number of shares of their masked implementation in Figure 1, where the black (quadratic) curve is from [22] and the red (linear) ones assume POLKA is from twice to (a conservative) five times more expensive than KYBER without countermeasures. Turning this qualitative analysis into a quantitative one in order to determine the target security level (and number of shares) that makes POLKA or improvements thereof a relevant alternative to existing schemes is an interesting scope for further investigations. As for the aforementioned quest towards analyzing sufficient security conditions for leakage-resistant post-quantum encryption schemes, we therefore hope our results can serve as a trigger towards evaluating the worst-case side-channel security level of masked post-quantum encryption schemes.

Additional related works are discussed in the ePrint report [46].

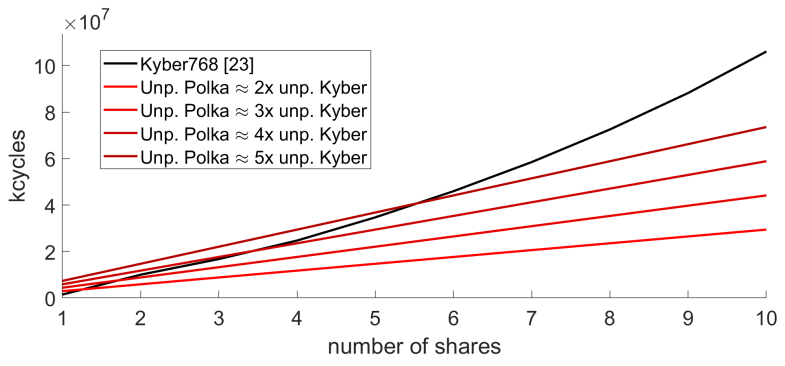

Fig. 1: Informal comparison between masked `POLKA` (n=1024, 16-bit modulus, ≈191 bits of security – see Section 4.3) and `KYBER768` (196 bits of security).

3 Background

3.1 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 $\rho_{\sqrt{\mathbf{\Sigma}},\mathbf{c}}(\mathbf{x}) = \exp(-\pi(\mathbf{x}-\mathbf{c})^\top \mathbf{\Sigma}^{-1}(\mathbf{x}-\mathbf{c}))$. 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 $D_{\Lambda,\sqrt{\mathbf{\Sigma}},\mathbf{c}}$ has probability mass $\Pr_{X \sim D_{\Lambda,\sqrt{\mathbf{\Sigma}},\mathbf{c}}}[X = \mathbf{x}] = \frac{\rho_{\sqrt{\mathbf{\Sigma}},\mathbf{c}}(\mathbf{x})}{\rho_{\sqrt{\mathbf{\Sigma}},\mathbf{c}}(\Lambda)}$ 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 ([56, Lemma 4.4]). *For $\sigma = \omega(\sqrt{\log n})$ there is a negligible function $\varepsilon = \varepsilon(n)$ such that $\Pr_{\mathbf{x} \sim D_{\mathbb{Z}^n,\sigma}}[\|\mathbf{x}\| > \sigma\sqrt{n}] \leq \frac{1+\varepsilon}{1-\varepsilon} \cdot 2^{-n}$.*

3.2 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 $r \rightarrow a \cdot r$. This implies that, for any $a, b \in R$, $\|a \cdot b\|_\infty \leq \|a\| \cdot \|b\|$ by the Cauchy-Schwartz inequality.

As in [50], for any lattice Λ, $D^{\mathsf{coeff}}_{\Lambda,\sigma}$ 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 [61]. 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 1, the number of samples k is made explicit.

Definition 1. *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 ([52]). *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 $\chi = D^{\mathsf{coeff}}_{\mathbb{Z}^n,r}$.*

4 `POLKA`: Rationale and Specifications

Our starting point is a variant of the LPR cryptosystem [52], which builds on a rigid randomness-recovering KEM. As in [52], 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 [52]), 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 [52] 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 [15] 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).

Our hybrid encryption scheme builds on a KEM with *explicit rejection* (as per [47, 59]), meaning that invalid encapsulations are rejected as soon as they are noticed in decryption. In the security proof, we will switch to an implicit rejection mechanism (as defined [59, 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.

4.1 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

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

Optionally, one can add b^{-1} in PK (to avoid computing an inverse in decryption).

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. The proof of Lemma 7 shows that, unless the RLWE assumption is false, a suitable b can be found after at most $\lceil \lambda / \log n \rceil$ iterations, except with negligible probability $2^{-\lambda}$. 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, $\mathsf{Decrypt}$ 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 $\mathsf{Encrypt}$ and $\mathsf{Decrypt}$ (by Lemma 1). At step 3, the decryptor obtains

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

where the last equality holds over R with overwhelming probability over the randomness of Keygen, $\mathsf{Encrypt}$ and $\mathsf{Decrypt}$. 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 $\mathsf{Encrypt}$ and $\mathsf{Decrypt}$, we also have $\|\bar{r}\|, \|\bar{e}_1\|, \|\bar{e}_2\| \leq 2\alpha q \sqrt{n}$. Then, the Cauchy-Schwartz inequality implies

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

Since $p/2 > 2\alpha q \sqrt{n}$, step 4 recovers $\bar{e}_2$ with overwhelming probability. Since $b \in R_q^\times$, $\mathsf{Decrypt}$ 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 (1).

Rigidity. By direct inspection of the decryption algorithm, given the KEM part (c_1, c_2), if it can extract a triple (r, e_1, e_2) in the domain, later called D_E, where each component has Euclidean norm bounded by B, the decapsulation is valid. If the decapsulation is valid we have $r = (c_2 - e_2)/b$ and $e_1 = c_1 - a \cdot r$ by definition (after removing the masks r', e'_1, e'_2). Hence, $c_1 = a \cdot r + e_1$ and $c_2 = b \cdot r + e_2$, which is the public encapsulation part on input (r, e_1, e_2).

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 a 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 [30, 33].

4.2 Black-Box Security Analysis

Our security proof uses ideas from Saito *et al.* [62, Section 4] to prove (tight) security in the QROM. Their approach exploits the implicit rejection mechanism of their KEM. Namely, when the incoming encapsulation (c_1, c_2) is found invalid upon decryption in [62], the decapsulated symmetric key K is replaced by a random-looking $K = H'(u, (c_1, c_2))$, where u is a random string included in the secret key and H' is an independent random oracle.

Here, in order to simplify the analysis of side-channel leakages in the real scheme, it is desirable to minimize the amount of secret key operations in the decryption algorithm and the amount of key material to protect against leakage. Therefore we refrain from introducing an additional secret key component u (cf. Section 5.1). Instead, our security proof will first switch (in Game_2) to a modified decryption algorithm where the rejection mechanism goes implicit and the decapsulation procedure computes K as a random function of (c_1, c_2). At this point, we will be able to apply the techniques from [62].

Since the implicit/explicit decapsulation mechanisms are used as part of a hybrid encryption system, we can argue that they are indistinguishable by relying on the ciphertext integrity of the symmetric encryption scheme. This is the reason why we are considering the CCA security of the hybrid combination as a whole, rather than that of its KEM component.[4] We note that similar ideas were

[4] The underlying explicit rejection KEM can be proven CCA-secure secure in the ROM but we do not prove it CCA-secure in the QROM as we only consider the CCA security of the hybrid PKE scheme.

previously used in the security proofs of hybrid PKE schemes [1,48], but usually in the opposite direction (to go from implicit rejection to explicit rejection).

For the rest, the proof in the ROM carries over to the QROM since it avoids ROM techniques that do not work in the QROM: we do not rely on the extraction of encryption randomness by inspecting the list of RO queries to answer decryption queries, which is not possible when queries are made on superpositions of inputs, and the RO is programmed identically for all queries.

Theorem 1. *If Π^{sym} is a symmetric authenticated encryption scheme, the construction specified in Section 4.1 provides IND-CCA security in the QROM under the ring learning-with-errors (*RLWE*) assumption.*

Proof. The proof considers a sequence of hybrid games, which is similar to that of [62, Theorem 4.2] from Game_3 to Game_5. For each i, we denote by W_i the event that the adversary wins (i.e., $d' = d$) in Game_i. We also denote by $\mathsf{Encaps}(PK, (r, e_1, e_2))$ the deterministic algorithm that takes as inputs PK and explicit randomness $(r, e_1, e_2) \in R^3$, and outputs $(c_1, c_2) = (a \cdot r + e_1, b \cdot r + e_2)$.

Game_0: This is the real IND-CCA game. The challenger faithfully answers (quantum) random oracle queries. All (classical) decryption queries are answered by running the real decryption algorithm. Note that a decryption query triggers a random oracle query at step 8 of $\mathsf{Decrypt}$. In the challenge phase, the adversary $\mathcal{A}$ outputs messages M_0, M_1 and obtains a challenge $C^\star = (c_0^\star, c_1^\star, c_2^\star)$, where $c_1^\star = a \cdot r^\star + e_1^\star$, $c_2^\star = b \cdot r^\star + e_2^\star$, with $r^\star, e_1^\star, e_2^\star \hookleftarrow D^{\mathsf{coeff}}_{\mathbb{Z}^n,\alpha q}$, and $c_0^\star = \mathsf{E}_{K^\star}(M_d)$ for some $d \hookleftarrow U(\{0,1\})$. Eventually, $\mathcal{A}$ outputs $b' \in \{0,1\}$ and its advantage is $\mathbf{Adv}(\mathcal{A}) := |\Pr[W_0] - 1/2|$.

Game_1: In this game, the challenger aborts and replaces $\mathcal{A}$'s output by a random bit $d' \in \{0,1\}$ in the event that $\|s\| > B$ or $\|e\| > B$ at step 3 of Keygen. By Lemma 1, we have $|\Pr[W_1] - \Pr[W_0]| \leq 2^{-\Omega(n)}$.

Game_2: We modify the decryption algorithm. Throughout the game, the challenger uses an independent random oracle $H_Q : R_q^2 \to \{0,1\}^\kappa$ that is only accessible to $\mathcal{A}$ via decryption queries (i.e., $\mathcal{A}$ has no direct access to H_Q). This random oracle is used to run the following decryption algorithm.

$\mathsf{Decrypt}_2$: Given $SK = s$ and $C = (c_0, c_1, c_2)$, initialize a Boolean variable $\mathsf{flag} = 0$. Then, do the following.

1. Sample $r', e_1', e_2' \hookleftarrow D^{\mathsf{coeff}}_{\mathbb{Z}^n,\alpha q}$. If $\|r'\| > B$, or $\|e_1'\| > B$, or $\|e_2'\| > B$, then set $\mathsf{flag} = 1$ and return $\perp$.[5] 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 $\|\bar{e}_2\| > 2B$, set $\mathsf{flag} = 1$.

[5] $\mathsf{Decrypt}_2$ still uses explicit rejection at step 1 because the secret key is not needed at this step and the goal of implicit rejection is to handle validity checks that depend on the secret key and the ciphertext.

5. Compute $\bar{r} = (\bar{c}_2 - \bar{e}_2) \cdot b^{-1} \in R_q$. If $\|\bar{r}\| > 2B$, set $\mathsf{flag} = 1$.
6. Compute $\bar{e}_1 = \bar{c}_1 - a \cdot \bar{r} \in R_q$. If $\|\bar{e}_1\| > 2B$, set $\mathsf{flag} = 1$.
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 set $\mathsf{flag} = 1$.
8. If $\mathsf{flag} = 0$, compute $K = H(r, e_1, e_2) \in \{0,1\}^{\kappa}$. Otherwise, compute $K = H_Q(c_1, c_2)$.
9. Compute and return $M = \mathsf{D}_K(c_0) \in \{0,1\}^{\ell_m} \cup \{\perp\}$.

Lemma 3 shows that, if the adversary can distinguish Game_2 from Game_1, we can turn it into an adversary against the ciphertext integrity of Π^{sym} (of which the definition is recalled in the ePrint report [46]).

Game_3: We now simulate the random oracle[6] $H : D_E \to \{0,1\}^{\kappa}$ as

$$H(r, e_1, e_2) = H_Q'\big(\mathsf{Encaps}(PK, (r, e_1, e_2))\big) \tag{2}$$

where $H_Q' : R_q^2 \to \{0,1\}^{\kappa}$ is another random oracle to which $\mathcal{A}$ has no direct access. At each decryption query, $\mathsf{Decrypt}_2$ consistently computes K as per (2) when $\mathsf{flag} = 0$. In the computation of $C^\star = (c_0^\star, c_1^\star, c_2^\star)$, the symmetric key $K^\star$ is similarly obtained as $K^\star = H_Q'\big(c_1^\star, c_2^\star\big)$, where $(c_1^\star, c_2^\star) = \mathsf{Encaps}(PK, (r^\star, e_1^\star, e_2^\star))$. Lemma 4 shows that, from $\mathcal{A}$'s view, Game_3 is identical to Game_2, so that we have $\Pr[W_3] = \Pr[W_2]$.

Game_4: This game is like Game_3 except that the random oracle H is now simulated as $H(r, e_1, e_2) = H_Q\big(\mathsf{Encaps}(PK, (r, e_1, e_2))\big)$, where $H_Q : R_q^2 \to \{0,1\}^{\kappa}$ is the random oracle introduced in Game_2. In the computation of the challenge ciphertext $C^\star = (c_0^\star, c_1^\star, c_2^\star)$, the symmetric key $K^\star$ is similarly obtained as $K^\star = H_Q\big(c_1^\star, c_2^\star\big)$, where $(c_1^\star, c_2^\star) = \mathsf{Encaps}(PK, (r^\star, e_1^\star, e_2^\star))$, and K is computed in the same way when $\mathsf{flag} = 0$ at step 8 of $\mathsf{Decrypt}_2$. That is, Game_4 is identical to Game_3 except that H_Q' has been replaced by H_Q in the simulation of H. Lemma 5 shows that $\Pr[W_4] = \Pr[W_3]$ as the two games are perfectly indistinguishable.

Game_5: This game is like Game_4 except that we modify the decryption oracle. At each query $C = (c_0, c_1, c_2)$, if $\mathsf{flag} = 0$ at the end of step 1, then the decryption oracle computes $K = H_Q(c_1, c_2)$ and returns $M = \mathsf{D}_K(c_0) \in \{0,1\}^{\ell_m} \cup \{\perp\}$ (i.e., it ignores steps 2-7 of $\mathsf{Decrypt}_2$ and jumps to step 8 after having set $\mathsf{flag} = 1$). Lemma 6 shows that $\Pr[W_5] = \Pr[W_4]$.

Game_6: We now remove the change introduced in Game_1. Namely, Game_6 is like Game_5, but we no longer replace $\mathcal{A}$'s output by a random bit if $\|s\| > B$ or $\|e\| > B$ at the end of Keygen. By Lemma 1, $|\Pr[W_6] - \Pr[W_5]| \leq 2^{-\Omega(n)}$.

In Game_6, we note that the decryption oracle does not use the secret s anymore.

Game_7: We modify the generation of PK. The challenger initially samples $a_1, \ldots, a_k \hookleftarrow U(R_q)$, $e_1, \ldots, e_k \hookleftarrow D^{\mathsf{coeff}}_{\mathbb{Z}^n, \alpha q}$, where $k = \lceil \lambda / \log n \rceil$, and computes $b_i = a_i \cdot s + e_i$ for each $i \in [k]$. If none of the obtained $\{b_i\}_{i=1}^k$ is

[6] We may assume that H outputs $\perp$ on input of a triple $(r, e_1, e_2) \notin D_E$. A hash function can always check domain membership before any computation.

invertible, the challenger aborts and replaces $\mathcal{B}$'s output by a random bit. Otherwise, it determines the first index $i \in [k]$ such that $b_i \in R_q^\times$ and defines the public key by setting $a = a_i$ and $b = p \cdot b_i$. Lemma 7 shows that, under the RLWE assumption, this modified key generation procedure does not affect $\mathcal{A}$'s view and we have $|\Pr[W_7] - \Pr[W_6]| \leq \mathbf{Adv}^{\mathsf{RLWE}}(\lambda) + 2^{-\lambda}$.

Game_8: We change again the generation of the public key. We replace the pseudorandom ring elements $\{b_i = a_i \cdot s + e_i\}_{i=1}^{k}$ of Game_7 by truly random $b_1, \ldots, b_k \hookleftarrow U(R_q)$ at the beginning of the game. Under the RLWE assumption, this change goes unnoticed and a straightforward reduction shows that $|\Pr[W_8] - \Pr[W_7]| \leq \mathbf{Adv}^{\mathsf{RLWE}}(\lambda)$. As a result, since $\gcd(p, q) = 1$, the public key is now distributed so that $a \sim U(R_q)$ and $b \sim U(R_q^\times)$.

Game_9: We change the generation of the challenge $C^\star = (c_0^\star, c_1^\star, c_2^\star)$. In this game, instead of computing $c_1^\star = a \cdot r^\star + e_1^\star$, $c_2^\star = b \cdot r^\star + e_2^\star$ with $r^\star, e_1^\star, e_2^\star \hookleftarrow D^{\mathsf{coeff}}_{\mathbb{Z}^n,\alpha q}$, we now sample $c_1^\star, c_2^\star \hookleftarrow U(R_q)$ uniformly. Then, we compute $c_0^\star$ as a symmetric encryption of M_d under the key $K^\star = H_Q(c_1^\star, c_2^\star)$. Lemma 8 shows that Game_9 is indistinguishable from Game_8 under the RLWE assumption.

In Game_9, $\mathcal{A}$ can no longer query H on short ring elements $(r^\star, e_1^\star, e_2^\star)$ that underlie $(c_1^\star, c_2^\star)$ (in which case we would have $H_Q(c_1^\star, c_2^\star) = H(r^\star, e_1^\star, e_2^\star)$). With overwhelming probability $1 - 2^{-\Omega(n)}$, there exist no $r^\star, e_1^\star, e_2^\star \in R$ of norm $\leq B$ such that $c_1^\star = a \cdot r^\star + e_1^\star$ and $c_2^\star = b \cdot r^\star + e_2^\star$. Since $\mathcal{A}$ has no direct access to $H_Q(\cdot)$, this means that $H_Q(c_1^\star, c_2^\star)$ is now independent of $\mathcal{A}$'s view.

Game_{10}: In this game, we modify the decryption oracle that now rejects all ciphertexts of the form $C = (c_0, c_1^\star, c_2^\star)$ with $c_0 \neq c_0^\star$ after the challenge phase. Game_{10} is identical to Game_9 until the event E_{10} that $\mathcal{A}$ queries the decryption of a ciphertext $C = (c_0, c_1^\star, c_2^\star)$ that would not have been rejected in Game_9. Since $c_0^\star = \mathsf{E}_{K^\star}(M_d)$ is encrypted under a random key $K^\star = H_Q(c_1^\star, c_2^\star)$ that is independent of $\mathcal{A}$'s view, E_{10} would imply an attack against the ciphertext integrity of Π^{sym} (as defined in the ePrint report [46]). We have $|\Pr[W_{10}] - \Pr[W_9]| \leq \Pr[E_{10}] \leq 2^{-\Omega(n)} + \mathbf{Adv}^{\mathsf{AE\text{-}INT}}(\lambda)$, where Q is the number of decryption queries.

In Game_{10}, the challenge $C^\star = (c_0^\star, c_1^\star, c_2^\star)$ is obtained by encrypting $c_0^\star$ under a random key $K^\star$ which is never used anywhere but in the computation of $c_0^\star = \mathsf{E}_{K^\star}(M_d)$. At this point, the adversary is essentially an adversary against the indistinguishability (under passive attacks) of the authenticated encryption scheme Π^{sym}. We have $|\Pr[W_{10}] - 1/2| \leq \mathbf{Adv}^{\mathsf{AE\text{-}IND}}(\lambda)$.

Putting the above altogether, we can bound the advantage of an IND-CCA adversary as

$$\mathbf{Adv}^{\mathrm{cca}}(\mathcal{A}) \leq \frac{3}{1 - 2^{-\lambda}} \cdot \mathbf{Adv}^{\mathcal{B},\mathsf{RLWE}}_{n,\lceil \lambda/\log n \rceil,q,\chi}(\lambda) + Q(Q+1) \cdot \mathbf{Adv}^{\mathsf{AE\text{-}INT}}(\lambda) + \mathbf{Adv}^{\mathsf{AE\text{-}IND}}(\lambda) + \frac{1}{2^{\Omega(n)}}, \tag{3}$$

where Q is the number of decryption queries. □

Lemma 3. Game_2 *is indistinguishable from* Game_1 *as long as the authenticated encryption scheme* Π^{sym} *provides ciphertext integrity. Concretely, we have the inequality* $|\Pr[W_2] - \Pr[W_1]| \leq \frac{Q\cdot(Q+1)}{2} \cdot \mathbf{Adv}^{\mathsf{AE\text{-}INT}}(\lambda)$.

Proof. See the ePrint report [46].

Lemma 4. *If* $q > 8p(\alpha q)^2 n$ *and* $p > 4\alpha q\sqrt{n}$, Game_3 *is perfectly indistinguishable from* Game_2.

Proof. See the ePrint report [46].

Lemma 5. Game_4 *is perfectly indistinguishable from* Game_3.

Proof. See the ePrint report [46].

Lemma 6. Game_5 *is perfectly indistinguishable from* Game_4.

Proof. See the ePrint report [46].

Lemma 7. *Under the* $\mathsf{RLWE}_{n,k,q,\chi}$ *assumption where* $\chi = D^{\mathsf{coeff}}_{\mathbb{Z}^n,\alpha q}$ *and* $k = \lceil \lambda / \log n \rceil$, Game_7 *is indistinguishable from* Game_6 *if* $q > n^2$. *Concretely, there is a PPT algorithm* $\mathcal{B}$ *such that* $|\Pr[W_7] - \Pr[W_6]| \leq \mathbf{Adv}^{\mathcal{B},\mathsf{RLWE}}_{n,k,q,\chi}(\lambda) + 2^{-\lambda}$.

Proof. the ePrint report [46].

Lemma 8. *Under the* RLWE *assumption,* Game_9 *is indistinguishable from* Game_8. *We have* $|\Pr[W_9] - \Pr[W_8]| \leq (1-2^{-\lambda})^{-1} \cdot \mathbf{Adv}^{\mathcal{B},\mathsf{RLWE}}_{n,k,q,\chi}(\lambda)$, *where* $\chi = D^{\mathsf{coeff}}_{\mathbb{Z}^n,\alpha q}$ *and* $k = 2\lceil \lambda / \log n \rceil$ *is the number of samples.*

Proof. See the ePrint report [46].

We note that bound (3) tightly relates the security of the scheme to the RLWE assumption. On the other hand, it loses a quadratic factor $O(Q^2)$ with respect to the ciphertext integrity of the symmetric authenticated encryption scheme. However, the term $Q(Q+1) \cdot \mathbf{Adv}^{\mathsf{AE\text{-}INT}}(\lambda)$ becomes statistically negligible if Π^{sym} is realized using an information-theoretically secure one-time MAC, as we discuss in the following instantiation section.

4.3 Parameters and Instantiations

PARAMETERS. In an instantiation in fully splitting rings R_q (which allows faster multiplications using the NTT in $\mathsf{Encrypt}$), 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 $4p(B^2+1) < q/2$, which satisfy the requirements of Lemma 4. 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. In order to get a more efficient implementation, we sample s, e, r, e_1, e_2 from a centered binomial distributions as previously suggested in [3, 20, 38] and set the parameters according to the concrete hardness against known attacks via the heuristic LWE estimator [2]. Precisely, we consider the noise distribution ψ_k^n defined over $\mathbb{Z}^n$ as $\{\sum_{i=1}^{k}(a_i - b_i) \mid a_i, b_i \hookleftarrow U(\{0,1\}^n)\}$ instead of a Gaussian distribution. As in [38], we use the distribution $\bar{\psi}_2^n$ obtained by reducing ψ_2^n mod 3. We then obtain ciphertexts of 4Kb by setting $n = 1024$, $p = 5$ and $q = 59393$, and an estimated security level of 191 bits. This modification of `POLKA` only entails minor modifications in the security proof, which are detailed in the ePrint report [46] and is next used as our main instance.

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 [38]. 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 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 [14]. Yet, `POLKA` encourages the following more efficient solution based on a key-homomorphic one-time MAC. 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 $\mathrm{GF}(2^{\ell_m})$. This specific MAC "annihilates" the quadratic term $O(Q^2)$ in the security bound (3). In the ciphertext integrity experiment (defined in the ePrint report [46]), the adversary's advantage can then be bounded as $(Q+1)/2^{\ell_m}$ if Q is the number of decryption queries. To make the term $Q(Q+1) \cdot \mathbf{Adv}^{\mathsf{AE\text{-}INT}}(\lambda)$ statistically negligible in (3), we can assume that $\ell_m \geq \lambda + 3\log^2\lambda$ in order to have $(Q+1)^2 Q/2^{\ell_m} < 2^{3\log^2\lambda}/2^{\ell_m} < 2^{-\lambda}$. Concretely, if we set $\lambda = 128$ and assume $Q < 2^{60}$, 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.

5 Side-Channel Security Analysis

We now discuss the leakage properties of POLKA. In Section 5.1 we introduce the general ideas supporting its leveled implementation and explain how its security requirements can be efficiently fulfilled. In Section 5.2, 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. In Section 5.3 we describe a hardware architecture for the most sensitive DPA target of POLKA. Our descriptions borrow the terminology introduced in [45] 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.

5.1 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 [14] and identify its SPA and DPA targets in decryption. The resulting leveled implementation of POLKA is represented in Figure 2. 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 (i.e., the key K can leak in full in case only leakage-resilience is needed). 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 [65].[7] A bit more concretely, this reference shows

[7] 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 [66].

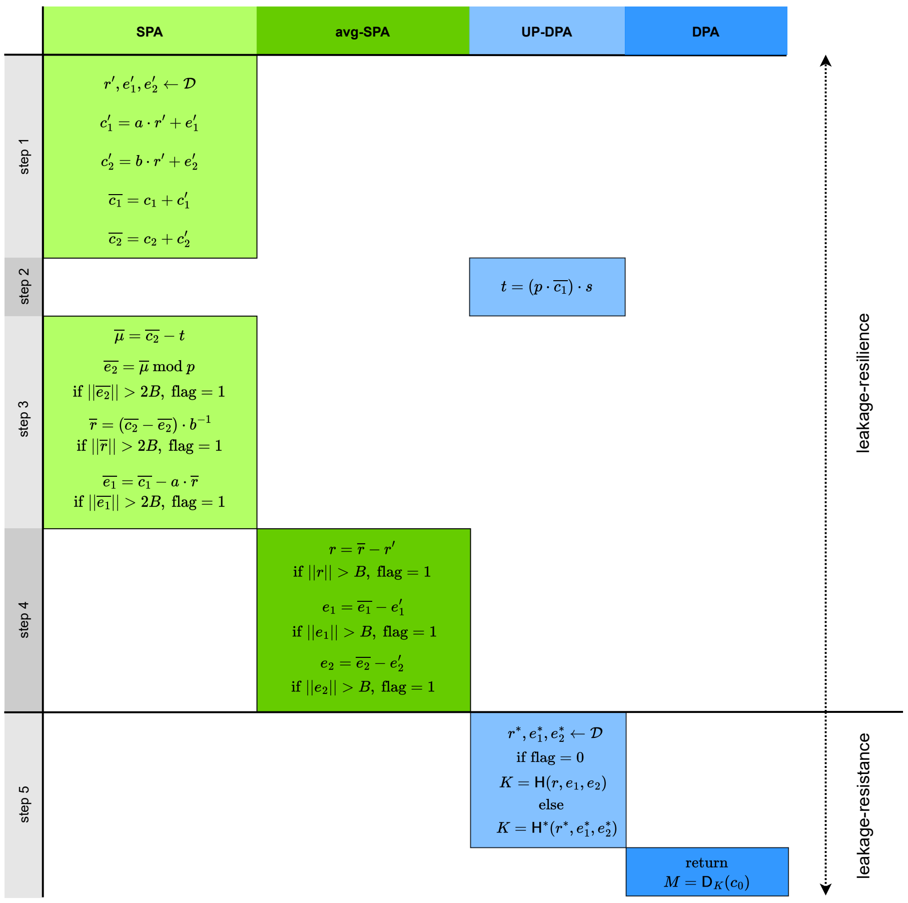
SPA
avg-SPA
UP-DPA
DPA
step 1
$r', e'_1, e'_2 \leftarrow \mathcal{D}$
$c'_1 = a \cdot r' + e'_1$
$c'_2 = b \cdot r' + e'_2$
$\overline{c_1} = c_1 + c'_1$
$\overline{c_2} = c_2 + c'_2$
step 2
$t = (p \cdot \overline{c_1}) \cdot s$
step 3
$\overline{\mu} = \overline{c_2} - t$
$\overline{e_2} = \overline{\mu} \bmod p$
if $||\overline{e_2}|| > 2B$, flag $= 1$
$\overline{r} = (\overline{c_2} - \overline{e_2}) \cdot b^{-1}$
if $||\overline{r}|| > 2B$, flag $= 1$
$\overline{e_1} = \overline{c_1} - a \cdot \overline{r}$
if $||\overline{e_1}|| > 2B$, flag $= 1$
step 4
$r = \overline{r} - r'$
if $||r|| > B$, flag $= 1$
$e_1 = \overline{e_1} - e'_1$
if $||e_1|| > B$, flag $= 1$
$e_2 = \overline{e_2} - e'_2$
if $||e_2|| > B$, flag $= 1$
step 5
$r^*, e^*_1, e^*_2 \leftarrow \mathcal{D}$
if flag $= 0$
$K = \mathsf{H}(r, e_1, e_2)$
else
$K = \mathsf{H}^*(r^*, e^*_1, e^*_2)$
return
$M = \mathsf{D}_K(c_0)$
leakage-resilience
leakage-resistance

Fig. 2: Leveled implementation of POLKA.

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 and 4 of Figure 2 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 [49]. Furthermore, implementing masking securely is a sensitive process, which requires dealing with composition issues [8,29], physical defaults such as glitches [54,58] or transitions [6,28] or even their combination [25,26].

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 [36, 39, 40, 55]. Namely, the operation $t = (p \cdot \overline{c_1}) \cdot s$ in step 2 of Figure 2 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 [9]. Second, their natural implementation offers strong immunity against composition issues and physical defaults [23]. 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 attack setting [16].

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 [12, 13]. Dziembowski et al. proposed a (more expensive) wPRF-based re-keying that is secure even if its output is leaked in full [40]. 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 [39]. 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 2, 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 [14], 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 4.3, 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 2 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 also remind that the implicit rejection used in schemes like KYBER or SABER generates a pseudorandom "garbage key" in case of invalid ciphertext, which implies the manipulation of additional long-term key material that must be secure against DPA. We avoid such a need by relying on an explicit rejection. Yet, it is an interesting open question to find out whether the same result could be obtained with other (implicit) rejection mechanisms or proof techniques.

Overall, Figure 2 highlights that the novelty of POLKA mostly lies in its leakage-resilient parts (i.e., steps 1 to 4). In order to help their understanding, we provide an open source piece of code (for now without SPA and DPA countermeasures that require lower level programming languages, neither ensuring the leakage-resistance of the authenticated encryption in step 5).[8] In general, further improving the leakage-resistance of POLKA so that it can be ensured with weaker side-channel security requirements is another interesting research direction.

We finally note that ensuring a constant-time implementation of POLKA requires running a dummy hash function when flag $= 1$. Without such a dummy hash, 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, by using a timing channel to detect whether a $\perp$ message is generated during step 4 or step 5 (by the authenticated encryption scheme). This also means that in order to avoid such a leakage on e_1, e_2 and r, it should be hard to distinguish whether the flag is 0 or 1 with SPA. We conjecture the latter is simpler/cheaper than protecting another long-term secret against DPA (as required with current implicit rejections), but as mentioned in introduction, POLKA could be adapted with an implicit rejection 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).

5.2 Learning With Physical Rounding Assumption

We now move to the main assumption that allows an efficient leveling of POLKA. Namely, we study the security of step 2 in Figure 2 after the recombination 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.[9] We start by recalling the LWPR problem introduced at CHES 2021 [39], then discuss its adaptation to polynomial multiplications used in POLKA. We finally propose security parameters together with cryptanalysis challenges.

[8] https://github.com/cmomin/polka_implem.

[9] As mentioned in subsection 5.1, 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.

A. The original LWPR problem can be viewed as an adaptation of the crypto dark matter proposed by Boneh et al. in [19], 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 2 (Learning with physical rounding [39]). *Let $q, x, y \in \mathbb{N}^*$, q prime, for a secret $\kappa \in \mathbb{F}_q^{x\times y}$. The $\mathsf{LWPR}_{\mathsf{L_g},q}^{x,y}$ sample distribution is given by:*

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

where $\mathsf{L_g} : \mathbb{F}_q^x \to \mathbb{R}^d$ is the physical rounding function. Given query access to $\mathcal{D}_{\mathsf{LWPR}_{\mathsf{L_g},q}^{x,y}}$ for a uniformly random κ, the $\mathsf{LWPR}_{\mathsf{L_g},q}^{x,y}$ 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 on its product with a random vector r. It corresponds to a learning problem similar to LWR [7], with the rounding function instantiated with a leakage function. Its security depends on the dimensions (q, x, y) and the leakage function considered. In [39], 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 binary representation) if the product is implemented in parallel. By this, we mean that $x \times \log_2(q)$ bits are produced per cycle by the implementation computing the LWPR samples. This problem can then be used as the basis of a fresh re-keying mechanism, producing an ephemeral key $\in \mathbb{F}_q^x$. The security analysis of Duval et al. 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. The first instance they propose 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). As can be seen in Figure 2, `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 [52], we now describe a ring version of the LWPR problem. Let us define $r := p \cdot \overline{c_1}$. 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 s (resp., r), we can write:

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

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 $r \cdot s$ can be represented as the following matrix-vector product:

$$
\begin{array}{cc}
 & \overset{n}{\longleftrightarrow} \\
x\updownarrow &
\begin{pmatrix}
\boxed{\begin{matrix} r_0 & -r_{n-1} & \dots & -r_2 & -r_1 \\ r_1 & r_0 & \ddots & & -r_2 \end{matrix}} \\
\begin{matrix} \vdots & \ddots & \ddots & \ddots & \vdots \\ r_{n-2} & & \ddots & \ddots & -r_{n-1} \\ r_{n-1} & r_{n-2} & \dots & r_1 & r_0 \end{matrix}
\end{pmatrix}
\cdot
\begin{pmatrix} s_0 \\ s_1 \\ \vdots \\ s_{n-2} \\ s_{n-1} \end{pmatrix}.
\end{array}
$$

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 s matrix can then be split in $\frac{n}{x}$ $(x \times n)$-submatrices, denoted $(B_u)_{0 \le u < \frac{n}{x}}$. The product can then be decomposed into $\frac{n}{x}$ subproducts (illustrated in blue) 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, $B_u^{i,j} = (2\mathbb{1}_{(ux+i-j)<0} - 1) \cdot r_{ux+i-j \pmod n}$ 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 in practice, 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 3 (Ring learning with Physical rounding). *Let $q, x, n \in \mathbb{N}$, q prime, for a secret $s \in R_q$. The $\mathsf{RLWPR}^{n,x}_{\mathsf{L_g},q}(s)$ sample distribution is given as:*

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

where $\mathsf{L_g}$ is the physical rounding function and the (B_u) are submatrices made of elements of r as defined above. Given query access to $\mathcal{D}_{\mathsf{RLWPR}^{n,x}_{\mathsf{L_g},q}(s)}$ for a uniformly random s, the $\mathsf{RLWPR}^{n,x}_{\mathsf{L_g},q}(s)$ 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 9. 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 [37]. 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 [52] for RLWE or [44, 51] for LPN variants) and we could not identify parts of the analyses in [39] that become easy in this case, the corresponding problems are less studied, justifying additional security margins to cover possible cryptanalysis improvements. 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 subsection 4.3, we propose the sets of parameters in Table 1 as interesting targets for cryptanalysis with a time complexity of less than 2^{128} and at most 2^{64} queries to a $\mathsf{RLWPR\text{-}b}^{n,x,y}_{\mathsf{HW_g},q}(s)$, assuming a Hamming weight leakage function.

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

Table 1: Proposed sets of parameters.

5.3 Hardware Performance Evaluation

We complete our results with a hardware prototype for the masked computation of t (i.e., step 2 in Figure 2), which is the most sensitive operation in POLKA. Due to place constraints, we defer the description of the hardware architecture we use and its FPGA implementation results in the ePrint report [46]. They confirm overheads that are linear in the number of shares d. Since based on similar or larger levels of parallelism as [39], these implementations are expected to provide similar or larger levels of security against higher-order DPA.

6 Conclusions

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, some of them being easy to mask. We reach this goal by mixing various ideas which we believe of independent interest. We also believe these techniques are quite generic and could be exploited for other schemes. For example, a leakage-resistant variant of the NTRU cryptosystem is discussed in the ePrint report [46].

Our results lead to a number of interesting research challenges. 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 direction for further investigations. The study 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. As hinted in [39], this context may require additional countermeasures like shuffling [66], in order to emulate the leakage of a parallel implementation. 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 [14]. So formalizing and proving the leakage security of POLKA with an appropriate set of physical assumptions and comparing the concrete security level of its implementations against the one of KYBER or SABER are necessary long-term goals.

Acknowledgments. The authors thank Tobias Schneider for useful feedback on the design of POLKA. Thomas Peters and François-Xavier Standaert are respectively research associate and senior research associate of the Belgian Fund for Scientific Research (F.R.S.-FNRS). This work has been funded in parts by the European Union through the ERC project 724725 (acronym SWORD) and the PROMETHEUS project (Horizon 2020 Research and Innovation Program, grant 780701), and by the Walloon Region Win2Wal project PIRATE.

References

1. M. Abe, R. Gennaro, K. Kurosawa, and V. Shoup. Tag-KEM/DEM: A new framework for hybrid encryption and a new analysis of Kurosawa-Desmedt KEM. In *Eurocrypt*, 2005.
2. M. Albrecht, R. Player, and S. Scott. On the concrete hardness of Learning with Errors. *Journal of Mathematical Cryptology*, 9(3), 2015.
3. E. Alkim, L. Ducas, T. Pöppelmann, and P. Schwabe. Post-quantum key exchange – a new hope. In *USENIX Security Symposium*, 2016.
4. R. Avanzi, J. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J. Schanck, P. Schwabe, G. Seiler, and D. Damien Stehlé. CRYSTALS-KYBER algorithm specifications and supporting documentation. *NIST PQC Round*, 3:42, 2020.
5. M. Azouaoui, O. Bronchain, C. Hoffmann, Y. Kuzovkova, T. Schneider, and F. Standaert. Systematic study of decryption and re-encryption leakage: The case of kyber. In *COSADE*, 2022.
6. J. Balasch, B. Gierlichs, V. Grosso, O. Reparaz, and F. Standaert. On the cost of lazy engineering for masked software implementations. In *CARDIS*, 2014.
7. A. Banerjee, C. Peikert, and A. Rosen. Pseudorandom functions and lattices. In *EUROCRYPT*, 2012.
8. G. Barthe, S. Belaïd, F. Dupressoir, P. Fouque, B. Grégoire, P. Strub, and R. Zucchini. Strong non-interference and type-directed higher-order masking. In *CCS*, 2016.
9. G. Barthe, F. Dupressoir, S. Faust, B. Grégoire, F. Standaert, and P. Strub. Parallel implementations of masking schemes and the bounded moment leakage model. In *Eurocrypt (1)*, 2017.
10. A. Basso, J. M. B. Mera, J.-P. D'Anvers, A. Karmakar, S. S. Roy, M. V. Beirendonck, and F. Vercauteren. SABER algorithm specifications and supporting documentation. *NIST PQC Round*, 3:44, 2020.

11. M. V. Beirendonck, J. D'Anvers, A. Karmakar, J. Balasch, and I. Verbauwhede. A side-channel-resistant implementation of SABER. *ACM J. Emerg. Technol. Comput. Syst.*, 17(2), 2021.
12. S. Belaïd, J. Coron, P. Fouque, B. Gérard, J. Kammerer, and E. Prouff. Improved side-channel analysis of finite-field multiplication. In *CHES*, 2015.
13. S. Belaïd, P. Fouque, and B. Gérard. Side-channel analysis of multiplications in GF(2128) - application to AES-GCM. In *ASIACRYPT (2)*, 2014.
14. D. Bellizia, O. Bronchain, G. Cassiers, V. Grosso, C. Guo, C. Momin, O. Pereira, T. Peters, and F. Standaert. Mode-level vs. implementation-level physical security in symmetric cryptography - A practical guide through the leakage-resistance jungle. In *Crypto (1)*, 2020.
15. D. J. Bernstein and E. Persichetti. Towards KEM unification. Cryptology ePrint Archive, Report 2018/526, 2018.
16. F. Berti, S. Bhasin, J. Breier, X. Hou, R. Poussier, F. Standaert, and B. Udvarhelyi. A finer-grain analysis of the leakage (non) resilience of OCB. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2022(1), 2022.
17. S. Bhasin, J. D'Anvers, D. Heinz, T. Pöppelmann, and M. V. Beirendonck. Attacking and defending masked polynomial comparison for lattice-based cryptography. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2021(3), 2021.
18. D. Boneh, O. Dagdelen, M. Fischlin, A. Lehmann, C. Schaffner, and M. Zhandry. Random oracles in a quantum world. In *Asiacrypt*, 2011.
19. D. Boneh, Y. Ishai, A. Passelègue, A. Sahai, and D. J. Wu. Exploring crypto dark matter: - new simple PRF candidates and their applications. In *TCC (2)*, 2018.
20. 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 EuroS&P* , 2018.
21. J. W. Bos, M. Gourjon, J. Renes, T. Schneider, and C. van Vredendaal. Masking KYBER: First- and higher-order implementations. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2021(4), 2021.
22. O. Bronchain and G. Cassiers. Bitslicing arithmetic/boolean masking conversions for fun and profit with application to lattice-based kems, 2022.
23. O. Bronchain, T. Schneider, and F. Standaert. Reducing risks through simplicity: high side-channel security for lazy engineers. *J. Cryptogr. Eng.*, 11(1):39–55, 2021.
24. O. Bronchain and F. Standaert. Breaking masked implementations with many shares on 32-bit software platforms or when the security order does not matter. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2021(3), 2021.
25. G. Cassiers, B. Grégoire, I. Levi, and F. Standaert. Hardware private circuits: From trivial composition to full verification. *IEEE Trans. Computers*, 70(10), 2021.
26. G. Cassiers and F. Standaert. Provably secure hardware masking in the transition- and glitch-robust probing model: Better safe than sorry. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2021(2):136–158, 2021.
27. C. Chen, O. Danba, J. Hoffstein, A. Hülsing, J. Rijneveld, J. Schanck, P. Schwabe, W. Whyte, Z. Zhang, T. Saito, T. Yamakawa, and K. Xagawa. NTRU algorithm specifications and supporting documentation. *NIST PQC Round*, 3:41, 2020.
28. J. Coron, C. Giraud, E. Prouff, S. Renner, M. Rivain, and P. K. Vadnala. Conversion of security proofs from one leakage model to another: A new issue. In *COSADE*, 2012.
29. J. Coron, E. Prouff, M. Rivain, and T. Roche. Higher-order side channel security and mask refreshing. In *FSE*, 2013.

30. 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.
31. 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.
32. J.-P. D'Anvers, E. Orsini, and F. Vercauteren. Error term checking: Towards chosen ciphertext security without re-encryption. In *AsiaPKC*, 2021.
33. 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.
34. C. Dobraunig, M. Eichlseder, S. Mangard, F. Mendel, B. Mennink, R. Primas, and T. Unterluggauer. Isap v2.0. *IACR Trans. Symmetric Cryptol.*, 2020(S1), 2020.
35. C. Dobraunig, M. Eichlseder, F. Mendel, and M. Schläffer. Ascon v1.2: Lightweight authenticated encryption and hashing. *J. Cryptol.*, 34(3), 2021.
36. C. Dobraunig, F. Koeune, S. Mangard, F. Mendel, and F. Standaert. Towards fresh and hybrid re-keying schemes with beyond birthday security. In *CARDIS*, 2015.
37. A. Duc, S. Faust, and F. Standaert. Making masking security proofs concrete - or how to evaluate the security of any leaking device. In *EUROCRYPT (1)*, 2015.
38. 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.
39. S. Duval, P. Méaux, C. Momin, and F. Standaert. Exploring crypto-physical dark matter and learning with physical rounding towards secure and efficient fresh rekeying. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2021(1), 2021.
40. S. Dziembowski, S. Faust, G. Herold, A. Journault, D. Masny, and F. Standaert. Towards sound fresh re-keying with hard (physical) learning problems. In *Crypto (2)*, 2016.
41. T. Fritzmann, M. V. Beirendonck, D. B. Roy, P. Karl, T. Schamberger, I. Verbauwhede, and G. Sigl. Masked accelerators and instruction set extensions for post-quantum cryptography. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2022(1), 2022.
42. E. Fujisaki and T. Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In *Crypto*, 1999.
43. E. Fujisaki and T. Okamoto. Secure integration of asymmetric and symmetric encryption schemes. *Jo. of Cryptology*, 226(21), 2013.
44. H. Gilbert, M. J. B. Robshaw, and Y. Seurin. Hb$^{\#}$: Increasing the security and efficiency of hb^{+}. In *EUROCRYPT*, 2008.
45. C. Guo, O. Pereira, T. Peters, and F. Standaert. Authenticated encryption with nonce misuse and physical leakage: Definitions, separation results and first construction - (extended abstract). In *LATINCRYPT*, 2019.
46. C. Hoffmann, B. Libert, C. Momin, T. Peters, and F. Standaert. Towards leakage-resistant post-quantum cca-secure public key encryption. *IACR Cryptol. ePrint Arch.*, page 873, 2022.
47. D. Hofheinz, K. Hövelmanns, and E. Kiltz. A modular analysis of the Fujisaki-Okamoto transformation. In *TCC*, 2017.
48. D. Hofheinz and E. Kiltz. Secure hybrid encryption from weakened key encapsulation. In *Crypto*, 2007.

49. Y. Ishai, A. Sahai, and D. A. Wagner. Private circuits: Securing hardware against probing attacks. In *CRYPTO*, 2003.
50. S. Katsumata and S. Yamada. Partitioning via non-linear polynomial functions: More compact IBEs from ideal lattices and bilinear maps. In *Asiacrypt*, 2016.
51. E. Kiltz, K. Pietrzak, D. Venturi, D. Cash, and A. Jain. Efficient authentication from hard learning problems. *J. Cryptol.*, 30(4):1238–1275, 2017.
52. V. Lyubashevsky, C. Peikert, and O. Regev. On ideal lattices and learning with errors over rings. In *Eurocrypt*, 2010.
53. S. Mangard, E. Oswald, and T. Popp. *Power analysis attacks - revealing the secrets of smart cards.* Springer, 2007.
54. S. Mangard, T. Popp, and B. M. Gammel. Side-channel leakage of masked CMOS gates. In *CT-RSA*, 2005.
55. M. Medwed, F. Standaert, J. Großschädl, and F. Regazzoni. Fresh re-keying: Security against side-channel and fault attacks for low-cost devices. In *Africacrypt*, 2010.
56. D. Micciancio and O. Regev. Worst-case to average-case reductions based on Gaussian measures. *SIAMJC*, 37(1):267–302, 2007.
57. K. Ngo, E. Dubrova, Q. Guo, and T. Johansson. A side-channel attack on a masked IND-CCA secure SABER KEM implementation. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2021(4), 2021.
58. S. Nikova, V. Rijmen, and M. Schläffer. Secure hardware implementation of non-linear functions in the presence of glitches. *J. Cryptol.*, 24(2):292–321, 2011.
59. E. Persichetti. *Improving the efficiency of code-based cryptography.* PhD thesis, Univ. of Auckland, 2012.
60. P. Ravi, S. S. Roy, A. Chattopadhyay, and S. Bhasin. Generic side-channel attacks on CCA-secure lattice-based PKE and KEMs. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2020(3), 2020.
61. O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In *STOC*, 2005.
62. T. Saito, K. Xagawa, and Y. Yamakawa. Tightly-secure key-encapsulation mechanism in the quantum random oracle model. In *Eurocrypt*, 2018.
63. V. Shoup. A proposal for an ISO standard for public key encryption. Manuscript, Dec. 2001.
64. R. Ueno, K. Xagawa, Y. Tanaka, A. Ito, J. Takahashi, and N. Homma. Curse of re-encryption: A generic power/EM analysis on post-quantum KEMs. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2022(1), 2022.
65. N. Veyrat-Charvillon, B. Gérard, and F. Standaert. Soft analytical side-channel attacks. In *ASIACRYPT (1)*, 2014.
66. N. Veyrat-Charvillon, M. Medwed, S. Kerckhof, and F. Standaert. Shuffling against side-channel attacks: A comprehensive study with cautionary note. In *ASIACRYPT*, 2012.