Classical Misuse Attacks on NIST Round 2 PQC

The Power of Rank-Based Schemes

Loïs Huguenin-Dumittan and Serge Vaudenay

LASEC, EPFL, Switzerland
{lois.huguenin-dumittan,serge.vaudenay}@epfl.ch

Abstract. The US National Institute of Standards and Technology (NIST) recently announced the public-key cryptosystems (PKC) that have passed to the second round of the post-quantum standardization process. Most of these PKC come in two flavours: a weak IND-CPA version and a strongly secure IND-CCA construction. For the weaker scheme, no level of security is claimed in the plaintext-checking attack (PCA) model. However, previous works showed that, for several NIST candidates, only a few PCA queries are sufficient to recover the secret key. In order to create a more complete picture, we design new key-recovery PCA against several round 2 candidates. Our attacks against CRYSTALS-Kyber, HQC, LAC and SABER are all practical and require only a few thousand queries to recover the full secret key. In addition, we present another KR-PCA attack against the rank-based scheme RQC, which needs roughly $O(2^{38})$ queries. Hence, this type of scheme seems to resist better than others to key recovery. Motivated by this observation, we prove an interesting result on the rank metric. Namely, that the learning problem with the rank distance is hard for some parameters, thus invalidating a common strategy for reaction attacks.

1 Introduction

As quantum computers are becoming a credible threat to standard public-key cryptography, the US National Institute of Standards and Technology (NIST) launched a standardization process for post-quantum cryptosystems. Many submissions were received at the first deadline in 2017. In January 2019, the second round candidates were announced, resulting in a smaller batch of 26 algorithms. Only a few types of schemes were proposed and most of them belong to three categories: lattice-based, code-based and multivariate-based. In addition, most lattice-based algorithms follow the same pattern, as shown in [3].

Most round 2 candidates share a similar structure: first, the authors present a CPA-secure public-key encryption scheme, which allows only for ephemeral keys. Then, this CPA construction is transformed into a strongly secure Key Exchange Mechanism (KEM) using the well-known Fujisaki-Okamoto (FO) transform or a variant [14,15,20,31].

While the CPA scheme is not meant to be secure if the secret key is used more than once, it is usually simpler and more efficient than its strongly secure counterpart. As a result, we think that the threat of misuse of the weaker

construction by non-experts in the implementation stage is high. Moreover, it was mentioned in [22] that badly implemented KEMs could leak information about the underlying CPA construction via side channels. More precisely, these implementations leaked whether the decryption of a ciphertext was correct or not and several timing attacks exploiting this flaw were subsequently proposed (e.g. [10,6]). This motivates our study of the key-reuse resilience of several NIST round 2 candidates.

In the security model we considered, the adversary can query a plaintext and ciphertext pair to an oracle, which returns whether the ciphertext decrypts to the given plaintext or not. The goal of the attacker is then to recover the secret key. This model makes sense in the side-channel scenario mentioned above. In addition, it also corresponds to the real-life setting where a malicious participant can attempt to establish a secure connection with a server. In this case, the malicious party can send erroneous ciphertexts and observe the reaction of the server (e.g. whether the secure channel can be established or not). This kind of attack is often called *reaction attack* in the literature.

Related work. Reaction attacks is an old topic in cryptography and one of the most famous examples is Bleichenbacher's attack against RSA published in 1998 [7]. The term *reaction attack* was probably first mentioned in [19]. In that paper, the authors showed that in the McEliece scheme, an adversary can recover a plaintext by observing decryption results of erroneous ciphertexts. In 2003, Howgrave-Graham et al. presented a reaction attack against the NTRU cryptosystem, which recovers the secret key [21]. More recently, several key-reuse and reaction attacks against post-quantum cryptosystems were published. See for example attacks against QC-MDPC [18], LEDApkc [12], NewHope [4], HILA5 [5], etc. In 2016, Fluhrer [13] and Ding et al. [11] showed how key-reuse can be exploited against Ring-LWE based schemes.

In 2019, Băetu et al. [3] introduced a framework capturing the similar structure shared by lattice-based proposals. In the same paper, the notion of key-recovery under plaintext-checking attack (KR-PCA) was presented, which formalized the concept of reaction attacks. More notably, the authors designed several misuse attacks against NIST candidates. It was shown that with a few thousand queries, many proposals can be broken if the secret key is reused. The algorithms attacked were (R.)EMBLEM, Frodo, KINDI, LIMA, LOTUS and Titanium. However, results against several NIST round 2 candidates are still missing. One of our goals is to get a more complete picture.

The same paper [3] also introduced the concept of *learning problem*. In this model, an adversary tries to recover a secret value, having access to an oracle that returns whether the distance between the secret and a given value is below some threshold. It was shown that an efficient learning algorithm was sufficient to design a practical KR-PCA attack in most cases. Interestingly, many key-reuse attacks solve an instance of the learning problem in one way or another in order to recover the key (e.g. [3,4,11]).

Finally, in an independent and concurrent work, Qin et al. [28] presented a reaction attack against Kyber similar to ours. Their paper is focused only on

Kyber while we target many schemes. The performance of their best attack is similar to ours, even if our algorithm seems to perform slightly better on average, at least for Kyber512.

Our contributions. In this paper, we present several key-reuse attacks in the KR-PCA model defined in [3]. More precisely, we design KR-PCA attacks against the following NIST round 2 proposals: HQC, LAC, CRYSTALS-Kyber, SABER and RQC. In our attacks (except for RQC), only a few thousands queries to the oracle are needed to recover the private key. Moreover, the complexity is polynomial in the size of the parameters. The only exception is RQC [25], a rank-metric proposal, for which our best attack is exponential (but still practical for the proposed parameters). We report our and other existing results against round 2 candidates in Table 1. We included external results only when the attack was in the same model as ours and targeted explicitly a version of a cryptosystem submitted to the NIST process. This does not mean that other round 2 candidates are not vulnerable to existing reaction attacks. Actually, apart from the schemes targeted in this paper, nearly all round 2 candidates have existing reaction attacks against them or similar schemes (e.g. the attack in [18] probably works on BIKE, [29] on ROLLO, [12] on LEDACrypt, [21] on NTRU, etc.).
For each scheme, we indicate the number of unknowns in the secret key in $\mathbb{Z}_q$, the maximal and expected number of queries necessary to recover the key. Concretely, the number of oracle calls can be seen as the number of times the key must be reused before the adversary can recover it. As a proof-of-concept, we also implemented the attacks against CRYSTALS-Kyber and SABER. As the attack against HQC is a straightforward application of the attack against Lepton from [3], we defer its description to Appendix C.

In addition, we show that the learning problem is hard in the rank-metric setting for some parameters. As most key-reuse attacks solve an instance of the learning problem in order to recover the key, this result demonstrates that such a strategy is not applicable to rank-based schemes. We stress that this result does not prove that efficient KR-PCA are impossible in the rank-metric but that common techniques are not applicable, which is still significant. From a more information-theoretical point of view, this confirms the intuition that the rank distance between a secret and a given value leaks much less information on the secret than other distances such as Hamming.

2 Notation

We let $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^n + 1)$. For a distribution Ψ, we write $x \leftarrow\!\!\$\, \Psi$ to denote that x is sampled from the distribution Ψ. If x is a vector or a polynomial of dimension n, we write $x \leftarrow\!\!\$\, \Psi^n$ to say that each component of x is sampled independently from Ψ. For some vector or polynomial x, x_i is the i-th coefficient and $(x)_i$ is the subset composed of the i-th first coefficients of x. For some set $\mathcal{X}$, $x \leftarrow\!\!\$\, \mathcal{X}$ means that x is sampled uniformly at random from $\mathcal{X}$. For $x \in \mathbb{Z}_q$, we write $x' = \langle x \rangle_q$ for the unique integer $x' \in (-\lfloor \frac{q}{2} \rfloor, \lfloor \frac{q}{2} \rfloor]$ s.t. $x' \equiv x \pmod q$.

Table 1: KR-PCA on NIST round 2 post-quantum cryptosystems. For each attack, we report the number of unknowns in the key, the number of oracle calls to recover the private key and the expected number of oracle calls, respectively. Values are rounded to the closest power of 2. The results obtained in this paper are highlighted.

Schemes	Unknowns	max. #queries	$\mathbb{E}$[#queries]
CRYSTALS-Kyber-512	2^{10}	2^{11}	2^{10}
Frodo-640 [3]	2^{12}	2^{16}	-
HQC-128 (see App. C)	2^{15}	2^{16}	2^{16}
LAC-128	2^{9}	2^{11}	2^{11}
NewHope1024 [27]	2^{10}	-	2^{20}
Round5 (HILA5) [5]	2^{10}	-	2^{13}
RQC-I	2^{13}	2^{67}	$\leq 2^{38}$
SABER (LightSaber)	2^{9}	2^{11}	2^{11}

We denote by $\lceil x \rfloor$ rounding x to the nearest integer, with ties rounded up. If f is a function defined on a component of a vector (or polynomial) v, we write $f(v)$ to denote the function being applied to each component of v. Finally, we denote $[n]$ the set $\{0, 1, \ldots, n-1\}$.

3 Plaintext-Checking Attack

We first recall the definition of a Public-Key Cryptosystem (PKC).

Definition 1 (Public-Key Cryptosystem). *A Public-Key Cryptosystem (*PKC*) is a tuple of four algorithms (*setup, gen, enc, dec*) defined as follows.*

- $\mathsf{pp} \leftarrow_{\$} \mathsf{setup}(1^\lambda)$*: The setup algorithm outputs the public parameters* pp*.*
- $(\mathsf{pk}, \mathsf{sk}) \leftarrow_{\$} \mathsf{gen}(\mathsf{pp})$*: The key generation algorithm takes the public parameters as inputs and outputs the public key* pk *and the secret key* sk*.*
- $\mathsf{ct} \leftarrow_{\$} \mathsf{enc}(\mathsf{pp}, \mathsf{pk}, \mathsf{pt})$*: The encryption procedure takes the public parameters* pp*, the public key* pk *and a plaintext* pt *as inputs and outputs a ciphertext* ct*.*
- $\mathsf{pt}' \leftarrow \mathsf{dec}(\mathsf{pp}, \mathsf{sk}, \mathsf{ct})$*: The decryption function takes the public parameters* pp*, the secret key* sk *and the ciphertext* ct *as inputs and outputs a plaintext* pt'*.*

A PKC *is correct if for any plaintext* pt*, after running the four procedures we have*

$$\Pr[\mathsf{pt} \neq \mathsf{pt}'] = \mathsf{negl}(\lambda).$$

The first three algorithms are randomized but can be considered as deterministic algorithms using random coins. In the following sections, we omit the public parameters in the inputs for the sake of simplicity.

The real-life scenario where a malicious user can detect whether or not a ciphertext decrypts to some plaintext was formally captured in [3]. In this work,

KR-PCA($\mathcal{A}$)

$\mathsf{pp} \leftarrow\!\!\$\ \mathsf{setup}(1^\lambda)$
$(\mathsf{pk}, \mathsf{sk}) \leftarrow\!\!\$\ \mathsf{gen}(\mathsf{pp})$
$\mathsf{sk}' \leftarrow \mathcal{A}^{\mathcal{O}^{\mathsf{PCO}}}(\mathsf{pp}, \mathsf{pk})$
return $1_{\mathsf{sk}'=\mathsf{sk}}$

Oracle $\mathcal{O}^{\mathsf{PCO}}(\mathsf{pt}, \mathsf{ct})$

1 : $\mathsf{pt}' \leftarrow \mathsf{dec}(\mathsf{pp}, \mathsf{sk}, \mathsf{ct})$
2 : **return** $1_{\mathsf{pt}'=\mathsf{pt}}$

Fig. 1: KR-PCA game.

LEARN$_{\Psi,\rho,\|\cdot\|}(\mathcal{A})$

$\delta \leftarrow\!\!\$\ \Psi$
$\delta' \leftarrow \mathcal{A}^{\mathcal{O}^{\mathsf{learn}}}$
return $1_{\delta'=\delta}$

Oracle $\mathcal{O}^{\mathsf{learn}}(x)$

return $1_{\|\delta+x\|\leq\rho}$

Fig. 2: LEARN game.

the authors define the notion of Key-Recovery under Plaintext-Checking Attack (KR-PCA), where an adversary has access to a plaintext-checking oracle and aims at recovering the secret key. This notion is defined by the game given in Figure 1.

In the same work, the authors define the notion of learning game. In this game, an adversary tries to learn a secret value given access to an oracle that returns whether or not the distance between the secret and the given value exceeds some threshold. We give this game in Figure 2. The game is parametrized by the threshold ρ, the secret value distribution Ψ and the norm $\|\cdot\|$. The adversary has access to the public parameters and to the oracle $\mathcal{O}^{\mathsf{learn}}$ and tries to guess the secret δ.

The authors then showed that for most of the lattice-based schemes of the NIST competition, the KR-PCA game reduces to the LEARN game. In addition, for most common norms (e.g. Hamming, L_1 in $\mathbb{Z}_q$, ...) the learning game can be solved in a logarithmic number of queries in the size of the secret domain (i.e. $O(\log_2(|D|))$ for $\delta \in D$). This led to the design of several efficient KR-PCA attacks.

4 LAC

4.1 LAC-CPA

In LAC [23], the elements are in $\mathcal{R}_q$. For $v \in \mathcal{R}_q, x \in \mathbb{Z}_q$, let $h(v, x) := |\{i : v_i = x, i \in [n]\}|$ be the function that counts the number of coefficients set to x in v. Then, we define $S_w = \{v : v \in \mathcal{R}_q, h(v, -1) = h(v, 1) = \frac{w}{2}\}$ for w even, as the set of polynomials in $\mathcal{R}_q$ that contains exactly $\frac{w}{2}$ 1s and -1s. In addition, we consider a centered binary distribution ψ_σ on $\{-1, 0, 1\}$ with variance σ and a BCH code of error-correcting capacity t. The scheme works as follows.

- gen: Sample $(\mathsf{sk}, d) \leftarrow\!\!\$\ S_w^2$ and $A \leftarrow\!\!\$\ \mathcal{R}_q$. Set $\mathsf{pk} = (A, B = A \times \mathsf{sk} + d)$.
- $\mathsf{enc}(\mathsf{pk}, \mathsf{pt} \in \{0,1\}^k)$: Sample $(t, e, f) \leftarrow\!\!\$\ S_w^2 \times \Psi_\sigma^{\ell_v}$ and output

$$(U, V) \leftarrow \left(t \times A + e, (t \times B)_{\ell_v} + f + \left\lceil \frac{q}{2} \right\rfloor \times \mathsf{encode}_{\mathrm{BCH}}(\mathsf{pt})\right).$$

- $\mathsf{dec}(\mathsf{sk}, U, V)$: Compute $W \leftarrow V - (U \times \mathsf{sk})_{\ell_v}$ and output $\mathsf{decode}(W)$.

The $\mathsf{decode}(W)$ function first computes

$$W'_i = \begin{cases} 1, & \text{if } \lceil \frac{q}{4} \rfloor \leq W_i < \lceil \frac{3q}{4} \rfloor \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

then outputs $\mathsf{decode}_{\mathrm{BCH}}(W')$.

4.2 KR-PCA

Consider w.l.o.g. that the KR-PCA attack uses $\mathsf{pt} = 0^k$. Hence, we have

$$\mathsf{encode}_{\mathrm{BCH}}(\mathsf{pt}) = 0^{\ell_v} \in \mathbb{Z}_q^{\ell_v}.$$

Then, since the BCH code can correct up to t errors, the decryption of some ciphertexts (U, V) will be incorrect (i.e. $\mathcal{O}^{\mathsf{PCO}}(\mathsf{pt}, (U, V)) = 0$) iff for at least t of the components of W we have $W_i \in [\lceil \frac{q}{4} \rfloor, \lceil \frac{3q}{4} \rfloor)$ by Eq. (1). Therefore, we can consider the following plaintext-checking attack (see Figure 3 in Appendix A for detailed pseudocode).

- Set $U = -(\lceil \frac{q}{4} \rfloor - 1) \in \mathcal{R}_q$ (i.e. a constant polynomial).
- We observe that

$$1 + (-U \times \mathsf{sk})_i \notin \left[-\lceil \frac{q}{4} \rfloor, \lceil \frac{q}{4} \rfloor \right) \Leftrightarrow \mathsf{sk}_i = 1 \tag{2}$$

$$-2 + (-U \times \mathsf{sk})_i \notin \left[-\lceil \frac{q}{4} \rfloor, \lceil \frac{q}{4} \rfloor \right) \Leftrightarrow \mathsf{sk}_i = -1. \tag{3}$$

  Then, let $V = \mathbf{1} \in \mathbb{Z}_q^{\ell_v}$ be the vector with 1 in every component. By Eq. (2), if there are more than t ones in sk, $V - (U \times \mathsf{sk})_{\ell_v}$ will decode incorrectly and $\mathcal{O}^{\mathsf{PCO}}(\mathsf{pt}, (U, V))$ will return a failure. Then, by iteratively cutting the number of 1s in V by half and querying the oracle, one can perform a binary search to find $\tilde{V} = (\tilde{V}_0, \ldots, \tilde{V}_{\ell_v}), \tilde{V}_i \in \{0, 1\}$ s.t. $\tilde{V} - (U \times \mathsf{sk})_{\ell_v}$ contains exactly t errors. Finally, given this vector $\tilde{V}$, one can perform the following algorithm.

  1. Let $V = \tilde{V}$ and $\mathcal{J} = \{i : \tilde{V}_i \neq 1\}$ be the subset of indices i for which $\tilde{V}_i$ $(= V_i)$ is not 1. Then, let's pick some $i \in \mathcal{J}$ and set $V_i = 1$. If the oracle returns an error, it means that $t + 1$ errors have been detected and thus the decoding of the ith component failed. In turn, that implies that condition in Eq. (2) is fulfilled. Hence, we know that $\mathsf{sk}_i = 1$. If the oracle returns no error, we set $V_i = -2$ and query again. If an error is returned it means $\mathsf{sk}_i = -1$ by Eq. (3), otherwise $\mathsf{sk} = 0$. One can iterate for every $i \in \mathcal{J}$. Thus, at the end of this step, we recovered all sk_i s.t. $i \in \mathcal{J}$.
  2. To get the other components of sk, we set $V = \tilde{V}$ as in the beginning of step 1 but we add an extra error such that $V - (U \times \mathsf{sk})_{\ell_v}$ contains $t + 1$ errors (we can do it easily since we know some values sk_i). Then, for each i s.t. $V_i = 1$ (i.e. $i \notin \mathcal{J}$), we proceed as follows. We set $V_i = 0$

and query the oracle. If the oracle does not return an error, it means the ith component was part of the $t+1$ errors (i.e. Eq. (2) was fulfilled) and therefore $\mathsf{sk}_i = 1$. Otherwise, if the oracle returns an error, we thus know $\mathsf{sk}_i \in \{-1, 0\}$. Let $\mathcal{I}$ be the indices of such components.

3. Set $V = \tilde{V}$ (i.e. $V - (U \times \mathsf{sk})_{\ell_v}$ contains t errors). For each $i \in \mathcal{I}$, set $V_i = -2$. If the oracle returns an error, it means that Eq. (3) is fulfilled and thus $\mathsf{sk}_i = -1$, otherwise $\mathsf{sk}_i = 0$. Hence, we recovered each components sk_i for $i \in \{1, \ldots, \ell_v\}$.

4.3 Remarks and results

Note that we assumed that $(\mathsf{sk})_{\ell_v}$ contained more than t ones for the binary search to succeed in finding $\tilde{V}$. If this is not the case, we can still perform the attack by first looking for $\tilde{V}, \tilde{V}_i \in \{-1, 0\}$ s.t. the decryption contains t errors and modify the signs in the attack. Note that for the parameters considered by LAC authors, it is very unlikely that sk contains less than t 1s (same for -1s). For example, for LAC128 ($n = 512, w = 256, \ell_v = 400, t = 16, \sigma = 1$), the probability to have less than t ones and minus ones in $(\mathsf{sk})_{\ell_v}$ if we assume each component i.i.d. with $\Pr[\mathsf{sk}_i = 0] = \Pr[\mathsf{sk}_i \in \{-1, 1\}] = \frac{1}{2}$ is

$$\Pr\left[|\{i : \mathsf{sk}_i = 0, \mathsf{sk}_i \in (\mathsf{sk})_{\ell_v}\}| > \ell_v - t\right] = \sum_{i=\ell_v - t + 1}^{\ell_v} \frac{1}{2^{\ell_v}} \binom{\ell_v}{i} \approx 2^{-311}.$$

In the worst case, we performed the binary search and queried 2 times for each component, thus the total number of queries is $\log_2(\ell_v) + 2 \times \ell_v$ Hence, since $\ell_v = 400$, we can recover 400 unknowns of sk in at most $\log_2(400) + 2 \times 400 \approx 2^{10}$ queries. Actually, if we denote $\mathsf{sk} = (\mathsf{sk}_1, \ldots, \mathsf{sk}_n)$, we will recover the ℓ_v leftmost coefficients. We can recover the $n - \ell_v$ remaining coefficients by applying the same attack using $U = (\lceil \frac{q}{4} \rceil - 1) \times X^{n-\ell_v}$. This will shift the $n - \ell_v$ coefficients to the leftmost positions (note that $-X^n = 1$ in $\mathcal{R}_q$). Hence, we need to apply at most two times the attack, resulting in a total number of queries smaller than 2^{11}. In the round 2 specifications [23], each component of V has its 4 least significant bits dropped after encryption. At decryption, each component is thus multiplied by 2^4. This does not impact our attack as Eq. (2)-(3) still hold with $\pm 2^4$ instead of $1, -2$. Finally, we note that in a recent independent work, D'Anvers et al. [10] exploits similar properties to perform a timing attack against LAC.

5 CRYSTALS-Kyber

5.1 Kyber-CPA

In CRYSTALS-Kyber [30], the elements are in $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^n + 1)$. Elements are sampled from a distribution Ψ_η which is defined as

$$\{(a_i, b_i)\}_{i \in [\eta]} \leftarrow\!\!\$\, \{0,1\}^{2\times\eta};\ \text{return}\ \sum_{i=1}^{\eta} (a_i - b_i)$$

with $\eta = 2$. Thus, Ψ_η returns a value in $\{-2, -1, 0, 1, 2\}$. For a polynomial $P \in \mathcal{R}_q$, we write $P \leftarrow_\$ \Psi_\eta$ to denote that each component of P is sampled independently from Ψ_η. Moreover, we define

$$\mathsf{compress}(x, d) = \left\lceil \frac{2^d}{q} \times x \right\rfloor \bmod 2^d$$
$$\mathsf{decompress}(x, d) = \left\lceil \frac{q}{2^d} \times x \right\rfloor .$$

Such functions guarantee that for any $x \in \mathbb{Z}_q$, we have

$$\left| \langle x - \mathsf{decompress}(\mathsf{compress}(x, d), d) \rangle_q \right| \leq \left\lceil \frac{q}{2^{d+1}} \right\rfloor .$$

When we apply these functions to vectors or polynomials in $\mathcal{R}_q$, we assume they are applied to each coefficient. Then, CRYSTALS-Kyber-CPA works as follows.

- gen: Sample $A \leftarrow_\$ \mathcal{R}_q^{k \times k}$ and $(\mathsf{sk}, d) \leftarrow_\$ (\Psi_\eta^k)^2$. Set $\mathsf{pk} \leftarrow (A, B) = (A, A \times \mathsf{sk} + d)$.
- $\mathsf{enc}(\mathsf{pk}, \mathsf{pt} \in \{0,1\}^n)$: Sample $(t, e, f) \leftarrow_\$ (\Psi_\eta^k)^2 \times \Psi_\eta$. Compute $(U, V) \leftarrow (t \times A + e, t \times B + f + \lceil \frac{q}{2} \rfloor \times \mathsf{pt}) \in \mathcal{R}_q^k \times \mathcal{R}_q$. Output $(\mathsf{compress}(U, d_U), \mathsf{compress}(V, d_V))$.
- $\mathsf{dec}(\mathsf{sk}, U', V')$: Compute $(U, V) \leftarrow (\mathsf{decompress}(U', d_U), \mathsf{decompress}(V', d_V))$. Return $\mathsf{compress}(V - U \times \mathsf{sk}, 1)$.
  We note that with the parameters proposed by the authors, we have

$$\mathsf{compress}(x, 1) = \begin{cases} 0, & \text{if } -\lceil \frac{q}{4} \rfloor \leq \langle x \rangle_q \leq \lceil \frac{q}{4} \rfloor \\ 1, & \text{otherwise} \end{cases} . \tag{4}$$

  Finally, we define δ as $V - U \times \mathsf{sk} = \delta + \mathsf{encode}(\mathsf{pt})$.

5.2 KR-PCA

From now on, we consider the parameters proposed by the authors for Kyber512, namely $n = 256, q = 3\,329, \eta = 2, d_U = 10$ and $d_V = 3$. In addition, we assume $k = 1$ for now. In the plaintext-checking attack, we consider the message with all components set to 0 (i.e. $\mathsf{pt} = 0 \in \mathcal{R}_q$) for the sake of simplicity, although some minor changes would allow the attack to work for any pt. Let $\rho = \lceil \frac{q}{4} \rfloor$. Then, by the definition of dec and Eq. (4), we know the plaintext-checking oracle (PCO) will return 1 (i.e. success) iff $|\langle \delta_i \rangle_q| \leq \rho, \forall i \in [n]$. First, we state the following lemma.

Lemma 1. *Let $U = -\lceil \frac{q}{4} \rfloor / 2 = -\rho/2$ be a constant polynomial and $U' = \mathsf{compress}(U, d_U)$. Given $k_i \in \{-3, \ldots, 4\}$, $i \in [n]$, let $V' = (0, \ldots, k_i, \ldots, 0)$ be the polynomial with k_i in the i-th coefficient and 0 elsewhere. Then, for $\mathsf{pt} = 0$ and the parameters of Kyber512, we have*

$$\mathcal{O}^{\mathsf{PCO}}(\mathsf{pt}, (U', V')) = 1 \Leftrightarrow \left| \left\langle \mathsf{sk}_i \times \frac{\rho}{2} + k_i \times \frac{\rho}{2} \right\rangle_q \right| \leq \rho.$$

Proof. First, we observe that for the given parameters, $\mathsf{decompress}(U', d_U) = U$.

Then, for $V' = (0, \ldots, k_i, \ldots, 0)$, $k_i \in \{-3, \ldots, 4\}$ we have $V = \mathsf{decompress}(V', d_V) = (0, \ldots, k_i \times \frac{\rho}{2}, \ldots, 0)$ because

$$\mathsf{decompress}(k_i, d_V) = \left\lceil \frac{q}{8} \times k_i \right\rfloor \overset{*}{=} k_i \times \left\lceil \frac{q}{4} \right\rfloor / 2 = k_i \times \frac{\rho}{2} \tag{5}$$

where the $*$ equality holds with the parameters $q = 3\,329$ and $k_i \in \{-3, \ldots, 4\}$.

Let $\delta = V - U \times \mathsf{sk}$. Then, for all $j \in [n], j \neq i$

$$\delta_j = 0 - \mathsf{sk}_j \times U = \mathsf{sk}_j \times \frac{\rho}{2} \in [-\rho, \rho]$$

since $\mathsf{sk}_j \in \{-2, \ldots, 2\}$ and $U = -\rho/2$ is a constant polynomial. For $j = i$ we have $\delta_i = k_i \times \frac{\rho}{2} + \mathsf{sk}_i \times \frac{\rho}{2}$. Now, since $\delta_j \in [-\rho, \rho]$ for all $j \neq i$, an error in the decoding can only happen in the i-th component. Hence, querying $\mathcal{O}^{\mathsf{PCO}}(\mathsf{pt}, (U', V'))$ is equivalent to querying some oracle $\mathcal{O}^{\mathsf{learn}}(k_i) = \mathbf{1}_{\left|\langle \alpha_i + k_i \times \frac{\rho}{2} \rangle_q\right| \leq \rho}$, where $\alpha_i = \mathsf{sk}_i \times \frac{\rho}{2} \in [-\rho, \rho]$. □

Note that the oracle $\mathcal{O}^{\mathsf{learn}}(k_i)$ in the proof above is similar to the one in the learning game defined in Figure 2. Now we set $k_i = -(k_i' + 2) \times \frac{\rho}{2}$ for some $k_i' \in \{-2, \ldots, 1\}$, $\alpha_i = \mathsf{sk}_i \times \frac{\rho}{2}$ and (U', V') as in Lemma 1. Then, if the condition

$$|\alpha_i + k_i| = \left|\alpha_i - \rho - k_i' \times \frac{\rho}{2}\right| \leq \lceil q/2 \rfloor \tag{6}$$

holds, then

$$\begin{aligned} \mathcal{O}^{\mathsf{PCO}}(\mathsf{pt}, (U', V')) = 1 \Leftrightarrow |\langle \alpha_i - \rho - k_i' \times \frac{\rho}{2} \rangle_q| \leq \rho \overset{(6)}{\Leftrightarrow} \\ |\alpha_i - \rho - k_i' \times \frac{\rho}{2}| \leq \rho \Leftrightarrow -\rho \leq \alpha_i - \rho - k_i' \times \frac{\rho}{2} \leq \rho \Leftrightarrow \\ k_i' \times \frac{\rho}{2} \leq \alpha_i \leq 2\rho + k_i' \times \frac{\rho}{2} \Leftrightarrow k_i' \times \frac{\rho}{2} \leq \alpha_i \Leftrightarrow k_i' \leq \mathsf{sk}_i \end{aligned}$$

where the first equivalence follows from Lemma 1, the second to last equivalence follows from $\alpha_i \leq \rho$ and $k_i' \times \frac{\rho}{2} \leq \rho$ (hence the second inequality always holds) and the last because $\alpha_i = -\mathsf{sk}_i \times U = \mathsf{sk}_i \times \frac{\rho}{2}$. Hence, by setting $k_i = -(k_i' + 2)$ and (U', V') as in Lemma 1, one can perform a binary search and recover sk_i by querying $\mathcal{O}^{\mathsf{PCO}}(0, (U', V'))$ and varying k_i'. In order for condition (6) to hold, we start with $k_i' = 0$. Then, in the further iterations the condition holds for any $\alpha_i, k_i' \times \rho/2 \in [-\rho, 0]$ or $\alpha_i, k_i' \times \rho/2 \in [0, \rho]$.

The last difficulty is in the case where the final interval is $[1, 2]$ (i.e. we know $\mathsf{sk}_i \in \{1, 2\}$ after some iterations). In this case, we would need to pick $k_i' = 2$ and set $V_i' = -(k_i' + 2) = -4$. However, in this case the $*$ equality in Equation (5) of the proof of Lemma 1 does not hold. A solution is to set $V_i' = -1$ and $U' = \mathsf{compress}(\frac{\rho}{2}, d_U)$ before querying $\mathcal{O}^{\mathsf{PCO}}(0^n, (U', V'))$. Then, for $\mathsf{sk}_i \in \{1, 2\}$ we have

$$\left|-\frac{\rho}{2} - \mathsf{sk}_i \times \frac{\rho}{2}\right| \leq \rho \Leftrightarrow \mathsf{sk}_i = 1.$$

Hence, if the query returns a success we can set $\mathsf{sk}_i \leftarrow 1$, otherwise $\mathsf{sk}_i \leftarrow 2$.

We give the KR-PCA algorithm in Figure 4 in Appendix A.

5.3 Efficiency and implementation

First, we note that the value of k (remember we work in $\mathcal{R}_q^k$) does not impact the attack but simply increases the number of coefficients we need to recover. Since we do 1 binary search with at most 3 queries and the total number of unknowns is $n \times k = 256 \times 2 = 512$, one can recover sk in at most $3 \times 512 = 1536$ queries. In addition, the number of queries in the binary search is only 2 when $\mathsf{sk}_i \in \{-2, -1, 0\}$. The probability that happens given $\mathsf{sk}_i \leftarrow\!\!\$\, \Psi_\eta$ is $\Pr[\mathsf{sk}_i \in \{-2, 1, 0\}] = \frac{11}{16}$. Hence, $\mathbb{E}[\#\text{queries}] = 512 \times \left(\frac{11}{16} \times 2 + \frac{5}{16} \times 3\right) = 1\,184$. We implemented a proof of concept of the attack in Sage for $k = 1$. Our code is based on a code[1] implemented for a paper by Albrecht et al. [1]. Finally, we note that the only differences between Kyber512 and the more secure versions are the parameter k and the compression factors d_U, d_V. For the higher security levels, the compression is less aggressive thus does not impact our attack and the number of queries required increases linearly with k.

6 SABER

6.1 SABER-CPA

SABER [9] works with vectors and matrices where components are polynomials in $\mathcal{R}_q$ for some integer q, as in Kyber. Components of the secret key are sampled from a centered binomial distribution Ψ_η, where the sampled elements are in the range $[-\eta/2, \eta/2]$. The security of SABER is based on the Module Learning With Rounding (M-LWR) problem. We apply our attack to the weaker version of SABER, namely LightSaber. In this version, the parameters are $e_q = 13, e_p = 10, e_T = 3, q = 2^{e_q}, p = 2^{e_p}, T = 2^{e_T}, \eta = 10, n = 256$ and $k = 2$. We also define the polynomial $h \in \mathcal{R}_p$ with all coefficients equal to $2^{e_p-2} + 2^{e_p-e_T-1} + 2^{e_q-e_p-1} = 196$ and the polynomial $h' \in \mathcal{R}_p$ with all coefficients set to $2^{e_q-e_p-1} = 4$. The $\times$ operation is the standard vector/matrix multiplication with component-wise polynomial multiplication (most elements are matrices or vectors of polynomials). The scheme works as follows.

- gen: Sample $\mathsf{sk} \leftarrow\!\!\$\, (\Psi_\eta^n)^k \in \mathcal{R}_q^k$, $A \leftarrow\!\!\$\, \mathcal{R}_q^{k \times k}$ and set $d \in \mathcal{R}_q^k$ as the vector with each coefficient set to h'. Then, compute $B \leftarrow (A \times \mathsf{sk} + d) \gg (e_q - e_p) \in \mathcal{R}_p^k$ where $\gg$ is the component-wise bitshift operation. Then, set $\mathsf{pk} = (A, B)$.
- $\mathsf{enc}(\mathsf{pk}, m \in \{0,1\}^n)$: Sample $t \leftarrow\!\!\$\, (\Psi_\eta^n)^k$, set $e \in \mathcal{R}_q^k$ as the vector with each coefficient set to h' and compute $U \leftarrow (A \times t + e) \gg (e_q - e_p) \in \mathcal{R}_p^k$. Set $V \leftarrow (B^T \times t + h - 2^{e_p-1} m) \gg (e_p - e_T) \in \mathcal{R}_T$ and output (U, V).
- $\mathsf{dec}(\mathsf{sk}, U, V)$: Output $(U^T \times \mathsf{sk} - 2^{e_p-e_T} V + h) \gg (e_p - 1) \in \mathcal{R}_2$.

[1] Available on https://github.com/fvirdia/lwe-on-rsa-copro

Let $W_i = (U \times \mathsf{sk})_i - 128 \times V_i + 196$. Then, a decrypted component can be written as

$$\mathsf{dec}(\mathsf{sk}, U, V)_i = \begin{cases} 0, & \text{if } W_i < 2^{e_p - 1} = 2^9 \\ 1, & \text{if } W_i \geq 2^{e_p - 1} = 2^9 \end{cases}.$$

6.2 KR-PCA

The idea of the Plaintext-Checking attack is similar to the one used in the previous section. However, here we have to deal with the addition of the polynomial $h = 196 + \ldots + 196 \cdot X^{n-1}$. Moreover, the domain of the components of the secret key is $\{-5, \ldots, 5\}$, which is much larger than in Kyber.

First, we consider $k = 1$, $\mathsf{pt} = 0^n$ and $V = 0 \in \mathcal{R}_T$. Then, for any constant polynomial $U \in [-\lfloor \frac{196}{5} \rfloor, \lfloor \frac{196}{5} \rfloor]$ and $\mathsf{sk}_i \in \{-5, \ldots, 5\}$, we have

$$W_i = (U \times \mathsf{sk})_i + 196 < 2^9 \ \forall i \in [n] \iff \mathcal{O}^{\mathsf{PCO}}(\mathsf{pt}, (U, V)) = 1.$$

This means that if we set $V = v_i \cdot X^i$ (i.e. only the i-th term is non-zero), we have the following equivalence

$$\mathcal{O}^{\mathsf{PCO}}(\mathsf{pt}, (U, V)) = 0 \iff (U \times \mathsf{sk})_i - 2^{e_p - e_T} v_i + 196 \geq 2^9.$$

In other words, an error can occur only in the i-th component. Let $v_i = 2$, then $-2^{e_p - e_T} v_i + 196 \pmod p = 964$. Now for $c \in \{2, 3, 4, 5\}$, we have

$$\mathcal{O}^{\mathsf{PCO}}\left(\mathsf{pt}, \left(\frac{60}{c}, 2X^i\right)\right) = 1 \iff 964 + \mathsf{sk}_i \times \frac{60}{c} \pmod p < 512 \iff \mathsf{sk}_i \geq c.$$

similarly, for $c \in \{-5, \ldots, -2\}$

$$\mathcal{O}^{\mathsf{PCO}}\left(\mathsf{pt}, \left(\frac{60}{c}, 2X^i\right)\right) = 1 \iff 964 + \mathsf{sk}_i \times \frac{60}{c} \mod p < 512 \iff \mathsf{sk}_i \leq c.$$

Hence, by querying $\mathcal{O}^{\mathsf{PCO}}(\mathsf{pt}, (U, v_i \cdot X^i))$ with $U = \frac{60}{c}$ one can perform a binary search to find all sk_i s.t. $\mathsf{sk}_i \in \{-5, \ldots, -2, 2, \ldots, 5\}$. Let $\mathcal{I}$ be the set of indices of such components.

In a second step, we want to find all $\mathsf{sk}_i \in \{-1, 0, 1\}$. As in the previous step, we can set $U = \pm\frac{60}{1}, V = 2X^i$. The problem is that in this case $U \notin [-\lfloor \frac{196}{5} \rfloor, \lfloor \frac{196}{5} \rfloor]$ and therefore it is not guaranteed that an error will occur only in the i-th component. However, since we know every $\mathsf{sk}_j, j \in \mathcal{I}$, we can find two vectors $\tilde{V}^{\pm} = \sum_{j \in \mathcal{I}} v_j^{\pm} \cdot X^j$ s.t. $\mathcal{O}^{\mathsf{PCO}}(\mathsf{pt}, (\pm 60, \tilde{V}^{\pm})) = 1$. Hence, by setting $U = \pm 60$ and $V = \tilde{V}^{\pm} + 2X^i$, one can find the remaining $\mathsf{sk}_i \in \{-1, 0, 1\}$. Finally, for $k > 1$, we can simply shift the polynomial U in an k-length vector and apply the same algorithm k times. The full algorithm is given in Figure 5 in Appendix A.

6.3 Efficiency and Implementation

The binary search for one secret component takes at most $\lceil \log(\eta) \rceil$ queries and there are $k \times n$ components. For LightSaber, it means that one can recover sk in at most $4 \times 512 = 2^{11}$ queries. The higher security levels for SABER require a less aggressive compression (as in Kyber) and a smaller domain for the components of the secret key. It means that a similar attack can be applied. For Saber and FireSaber, $3 \times 768 \approx 2^{11}$ and $3 \times 1024 = 3072$ queries would be needed, respectively. Interestingly, the maximal number of queries required for Saber would be roughly the same as for LightSaber. As a proof of concept, we implemented the attack against LightSaber using the reference implementation in C.

Finally, we leave as a future improvement the optimization of the way the value c is picked in the binary search. Following the results presented in [3], it should be feasible to design a binary search algorithm with an expected number of queries close to $H(\mathsf{sk}_i)$, where $H(\cdot)$ is the Shannon entropy. For instance, in LightSaber we have $H(\mathsf{sk}_i) \approx 2.7$.

7 RQC

7.1 Rank-based cryptography

The RQC cryptosystem [25] is similar to HQC [26] but uses the rank metric instead of the Hamming distance. Let q be a prime and consider the finite field $\mathbb{F}_{q^m}$. Let $g \in \mathbb{F}_q[X]$ be an irreducible polynomial of degree m. Then, we have $\mathbb{F}_{q^m} \simeq \mathbb{F}_q[X]/\langle g \rangle \simeq \mathbb{F}_q^m$. Now, let $\mathbb{F}_{q^m}^n$ be the vector space over the finite field $\mathbb{F}_{q^m}$. Each element of this vector space can be seen as a polynomial in $\mathbb{F}_{q^m}[X]/\langle f \rangle$ where $f \in \mathbb{F}_q[X]$ is an irreducible polynomial of degree n, using the trivial isomorphism

$$\phi : \ v \in \mathbb{F}_{q^m}^n \mapsto \sum_{i=0}^{n-1} v_i X^i \pmod{f}.$$

For elements in $\mathbb{F}_{q^m}^n$, the multiplication $\times$ is defined as the polynomial multiplication in $\mathbb{F}_{q^m}[X]/\langle f \rangle$. More formally, for any $a, b \in \mathbb{F}_{q^m}^n$

$$a \times b := \phi^{-1}(\phi(a) \cdot \phi(b)).$$

Similarly, the multiplication in $\mathbb{F}_{q^m}$ is defined as the polynomial multiplication in $\mathbb{F}_q[X]/\langle g \rangle$. In RQC-I, as $m = 97$ and $n = 67$, the two polynomials are $f = X^{67} + X^5 + X^2 + X + 1$ and $g = X^{97} + X^6 + 1$.

Rank metric and support. Let $v = (v_0, v_1, \ldots, v_{n-1}) \in \mathbb{F}_{q^m}^n$ and $\{\beta_i\}_{i \in [m]}$ be a basis of $\mathbb{F}_{q^m}$ over $\mathbb{F}_q$. Then, each component $v_i \in \mathbb{F}_{q^m}$ can be written as a vector in $\mathbb{F}_q^m$ using the basis representation. Hence, v can be represented as a $m \times n$

matrix with elements in $\mathbb{F}_q$. We denote this matrix by $\mathcal{M}(v)$, which is of the form

$$\mathcal{M}(v) = \begin{pmatrix} v_{0,0} & \cdots & v_{n-1,0} \\ \vdots & \ddots & \vdots \\ v_{0,m-1} & \cdots & v_{n-1,m-1} \end{pmatrix}$$

with $v_{i,j} \in \mathbb{F}_q$ s.t. $v_i = \sum_{j \in [m]} v_{i,j}\beta_j$. While not important, the choice of basis of $\mathbb{F}_{q^m}$ impacts the matrix representation. In what follows, we consider the canonical basis. That is, we consider $v \in \mathbb{F}_{q^m}$ as a polynomial in $\mathbb{F}_q[X]/\langle g \rangle$ and take the trivial representation of this polynomial as a vector in $\mathbb{F}_q^m$.

Definition 2 (Rank in $\mathbb{F}_{q^m}^n$). *Let $v \in \mathbb{F}_{q^m}^n$ be a vector and $\mathcal{M}(v) \in \mathbb{F}_q^{m \times n}$ be its matrix representation as defined above. Then, we define the rank of v as*

$$\|v\| := \mathsf{rank}(\mathcal{M}(v))$$

that is, the rank of the matrix representation of v. Then, the distance between $v, w \in \mathbb{F}_{q^m}^n$ is defined as

$$\|v - w\| = \mathsf{rank}(\mathcal{M}(v) - \mathcal{M}(w)).$$

For an arbitrary matrix A, let $\mathsf{span}(A)$ be the vector space spanned by the columns of A. Then, the support of a vector is defined as follows.

Definition 3 (Support in $\mathbb{F}_{q^m}^n$). *Let $v \in \mathbb{F}_{q^m}^n$. Then, the support is*

$$\mathsf{supp}(v) := \mathsf{span}(\mathcal{M}(v))$$

i.e. the vector space spanned by the columns of $\mathcal{M}(v)$. Similarly, we write $\mathsf{supp}(v^T)$ for the vector space spanned by the rows of $\mathcal{M}(v)$. Finally, by the definition of the rank of a matrix, we have $\mathsf{dim}(\mathsf{supp}(v)) = \mathsf{dim}(\mathsf{supp}(v^T)) = \|v\|$.

A useful tool when dealing with vector subspaces is the q-binary coefficient (also called Gaussian coefficient), which counts the number of subspaces of dimension r in a vector space of dimension n over a field of cardinality q. It is defined as

$$\begin{bmatrix} n \\ r \end{bmatrix}_q = \prod_{i=0}^{r-1} \frac{q^n - q^i}{q^r - q^i}.$$

7.2 RQC scheme

Let $S_w^n = \{v \in \mathbb{F}_{q^m}^n : \|v\| = w\}$ and $S_{1,w}^n = \{v \in \mathbb{F}_{q^m}^n : \|v\| = w,\ 1 \in \mathsf{supp}(v)\}$. In addition, let $w, w' \in \mathbb{Z}$ be parameters. RQC uses a random Gabidulin code [16] defined by a generating matrix $G \in \mathbb{F}_{q^m}^{k \times n}$ and with decoding capacity $\rho = \lfloor \frac{n-k}{2} \rfloor$. We denote the corresponding decoding algorithm by $\mathsf{decode}_{\mathrm{gab}}$. Then, RQC-CPA works as follows.

- gen: Sample $(\mathsf{sk}, d) \leftarrow_{\$} S_{1,w}^{2n}$ and $A \leftarrow_{\$} \mathbb{F}_{q^m}^n$. Set $B \leftarrow A \times \mathsf{sk} + d$. Pick a random generating matrix $G \in \mathbb{F}_{q^m}^{k \times n}$ for some Gabidulin code. Output $(\mathsf{pk} = (A, B, G), \mathsf{sk})$.
- $\mathsf{enc}(\mathsf{pk}, m \in \{0,1\}^k)$: Sample $(t, e, f) \leftarrow_{\$} S_{w'}^{3n}$. Compute $U \leftarrow A \times t + e$ and $V \leftarrow B \times t + mG + f$. Output (U, V).
- $\mathsf{dec}(\mathsf{sk}, U, V)$: Output $\mathsf{decode}_{\mathrm{gab}}(V - U \times \mathsf{sk})$.

Correctness. Let $\delta = t \times d + f - e \times \mathsf{sk}$. Then, for any legit ciphertext (U, V) (i.e. $(U, V) = \mathsf{enc}(\mathsf{pk}, m)$ for some pk, m) we have $V - U \times \mathsf{sk} = mG + \delta$. Since the decoding capacity of the code is ρ, we assume $\mathsf{dec}(\mathsf{sk}, U, V) = m \iff \|\delta\| \leq \rho$ thus, $\mathcal{O}^{\mathsf{PCO}}(\mathsf{pt}, U, V) = 1 \iff \|\delta\| \leq \rho$.

7.3 KR-PCA

We give a Key-Recovery under Plaintext-Checking attack that works with an expectation of $O(wq^{\min\{m,n\}-\rho+1})$ queries. As $q = 2, w = 5, m = 97, n = 67$ and $\rho = 31$ for RQC-I, we obtain a complexity of $O(2^{39})$. First, we state a useful theorem and two lemmas.

Theorem 1 (Lemma 1, [8] or Theorem 11, [24]). *Let $X, Y \in \mathbb{F}^{m \times n}$ be two $m \times n$ matrices over an arbitrary field $\mathbb{F}$. Then,*

$$\mathsf{rank}(Y + X) = \mathsf{rank}(Y) + \mathsf{rank}(X)$$

iff

$$\mathsf{span}(Y) \cap \mathsf{span}(X) = \{0\} \text{ and } \mathsf{span}(Y^T) \cap \mathsf{span}(X^T) = \{0\}.$$

In other words, for two matrices over a field, the rank of their sum is equal to the sum of their rank iff their column space (resp. their row space) trivially intersect.

Lemma 2. *We consider the RQC* PKC*. Let $B = A \times \mathsf{sk} + d$, $\mathsf{sk}, d \in \mathbb{F}_{q^m}^n$, $\mathsf{supp}(\mathsf{sk}) = \mathsf{supp}(d)$ and $\|\mathsf{sk}\| = \|d\| = w$. Then, finding a subspace $F \subset \mathbb{F}_{q^m}$ s.t. $z = \mathsf{dim}(F) \leq \frac{m}{2}$ and $\mathsf{supp}(\mathsf{sk}) = \mathsf{supp}(d) \subseteq F$ is sufficient to recover sk and d. Similarly, let $z = \mathsf{dim}(F)$, $z' = \mathsf{dim}(F')$, then finding $F, F' \subset \mathbb{F}_q^n$ s.t. $z + z' \leq n$, $\mathsf{supp}(\mathsf{sk}^T) \subseteq F$ and $\mathsf{supp}(d^T) \subseteq F'$ is sufficient to recover sk and d.*

Proof sketch. We give here an informal argument. A complete discussion can be found in [2]. If one can find a subspace F s.t. the support of sk (and d) is contained in it, one can compute a basis $\{\beta_i\}_{i \in [z]}$ for the subspace F. Then, one can write $\mathsf{sk}_i = \sum_{j=0}^{z-1} a_{i,j}\beta_j$ and $d_i = \sum_{j=0}^{z-1} b_{i,j}\beta_j$, where the $2nz$ coefficients $a_{i,j}, b_{i,j}$ are unknown. Then, $B = (A, 1) \cdot (\mathsf{sk}, d)^T \in \mathbb{F}_{q^m}^n$ can be seen as a system of nm linear equations in $\mathbb{F}_q$ with $2nz$ unknown coefficients. Hence, as long as $nm \geq 2nz \iff z \leq \frac{m}{2}$, one can solve the system of equations to recover sk, d.

Similarly, if one can find a basis for a subspace containing the row space of $\mathcal{M}(\mathsf{sk})$ and another for the row space of $\mathcal{M}(d)$, one can write the system of mn equations in $\mathbb{F}_q$ given by B as a system with $m(z + z')$ unknown coefficients. In this case, the system is solvable for $z + z' \leq n$. □

Lemma 3. *Let $p_{k,w}^n$ the probability that some random subspace of dimension k non-trivially intersects a given subspace of dimension w in $\mathbb{F}_q^n$, with $k + w \leq n$. Then,*

$$p_{k,w}^n \leq (q^k - 1)\frac{(q^w - 1)}{(q^n - 1)} \leq q^{w+k-n}.$$

Proof. See Appendix B.1.

The attack. Let $V = x$ for some $x \in \mathbb{F}_{q^m}^n$ and $U = -1 \in \mathbb{F}_{q^m}^n$. Then,

$$\mathcal{O}^{\mathsf{PCO}}(0, (U, V)) = 1 \iff \|\mathsf{sk} + x\| \leq \rho.$$

Let's pick $x \in \mathbb{F}_{q^m}^n$ at random s.t. $\|x\| = \rho - w$. Then, by Theorem 1, we have $\|\mathsf{sk} + x\| = \rho$ iff the column spaces (resp. the row spaces) of sk and x do not intersect (i.e. trivially intersect). By Lemma 3, the probability an intersection occurs in the column space or in the row space is upper bounded by $p^m_{\rho-w,w} + p^n_{\rho-w,w} \leq q^{\rho-m} + q^{\rho-n}$. Since $m \geq n$ and $\rho < \frac{n}{2}$ in RQC, this can be further bounded by $O(q^{-n/2})$, which is negligible in n. Hence, we assume this does not occur and $\|\mathsf{sk} + x\| = \rho$. In this case, $\mathsf{supp}(\mathsf{sk}) \subset \mathsf{supp}(\mathsf{sk} + x)$ and $\mathsf{supp}(\mathsf{sk}^T) \subset \mathsf{supp}((\mathsf{sk} + x)^T)$. Indeed, each vector in $\mathsf{supp}(\mathsf{sk} + x)$ can be written as a linear combination of vectors in the union of the basis of sk and x. Clearly, the union of the two basis is then a basis for $\mathsf{supp}(\mathsf{sk}+x)$ since $\|\mathsf{sk}+x\| = w+(\rho-w)$. The same argument works for the row space. Hence, the attack consists of finding a basis of $\mathsf{supp}(\mathsf{sk} + x)$ or $\mathsf{supp}((\mathsf{sk} + x)^T)$ and then finding sk by Lemma 2. We focus on finding the first one.

Let $u = \mathsf{sk} + x$ with $\|u\| = \rho$ and $y = (\alpha, 0, \ldots, 0) \in \mathbb{F}_{q^m}^n$. Then,

$$\mathcal{M}(y) = \begin{pmatrix} \alpha_0 & 0 \cdots 0 \\ \alpha_1 & 0 \cdots 0 \\ \vdots & \vdots \ddots \vdots \\ \alpha_{m-1} & 0 \cdots 0 \end{pmatrix}.$$

We observe that $\mathsf{supp}(u) \cap \mathsf{supp}(y) \neq \{0\} \iff \alpha \in \mathsf{supp}(u)$ for $\|y\| = 1$. Therefore, by Theorem 1, $\|u+y\| = \rho$ iff $y \in \mathsf{supp}(u)$ or $(1, 0, \ldots, 0) \in \mathsf{supp}(u^T) \subset \mathbb{F}_q^n$. Now, if we consider $\mathsf{supp}(u^T)$ as a random subspace of dimension ρ in $\mathbb{F}_q^n$, the probability that $(1, 0, \ldots 0) \in \mathsf{supp}(u^T)$ can be upper bounded by $q^{\rho+1-n} \leq q^{-n/2+1}$ by Lemma 3, which is negligible. Hence, one can iterate over all $\alpha \in \mathbb{F}_{q^m}$ and mark α whenever $\|u + y\| \leq \rho$. At the end, all marked α's form the vector space $\mathsf{supp}(u)$. Then, one can find a basis for this subspace and recover the secret key sk by Lemma 2, since $\rho < \frac{n}{2} < \frac{m}{2}$. In this case, the total number of queries needed is $O(q^m)$. Note that the strategy of querying y with only one non-null component is similar to a recent timing attack against RQC [6].

Improved attack. Now, instead of marking all α's in the vector space $\mathsf{supp}(u)$, one can mark α s.t. α is not in the subspace spanned by the already marked α's. More formally, in the i-th step, if we know that $\alpha^{(1)}, \ldots, \alpha^{(i-1)} \in \mathsf{supp}(u)$, we do not mark $\alpha^{(i)}$ s.t. $\alpha^{(i)} \in \langle \alpha^{(1)}, \ldots, \alpha^{(i-1)} \rangle$. In that way, the expected number of queries needed is lowered since we recover only a basis of $\mathsf{supp}(u)$ and not the whole subspace. Note that we could check for linear independence of $\alpha^{(i)}$ before querying it, sparing a few queries but increasing the amount of offline work.

The expected number of queries needed can be approximated as follows. Let X_i be the number of queries needed to find a new basis vector in $\mathsf{supp}(u)$, knowing we already found $\alpha^{(1)}, \ldots, \alpha^{(i)} \in \mathsf{supp}(u)$. We refer to the vectors which are not a new basis vector as *bad*. In each step, we assume we did not query any

bad vectors. Thus, the number of potential basis vectors is $q^\rho - q^i$ and the total number of vectors left to query is $q^m - i$. The expected number of draws before getting a *good* vector (i.e. a new basis vector) is therefore $\mathbb{E}[X_i] = \frac{q^m - i + 1}{q^\rho - q^{i-1} + 1}$. At the beginning, we already know that the basis of x is a set of $\rho - w$ linearly independent elements of $\mathsf{supp}(u)$. Therefore, we set $\alpha^{(1)}, \dots, \alpha^{(\rho-w)}$ as the basis of x and only w basis vectors need to be found. Hence, the expected total number of queries before getting the ρ basis vectors is approximately

$$\sum_{i=\rho-w}^{\rho-1} \frac{q^m - i + 1}{q^\rho - q^i + 1} \leq \sum_{i=\rho-w}^{\rho-1} \frac{q^m}{q^\rho - q^i} \leq wq^{m-\rho+1}.$$

Note that this is actually an upper bound on the real expectation, since we made an assumption that worsens the actual performance (i.e. we forget we already queried some *bad* vectors). The full attack is given in Figure 6 in Appendix A. Hence, the expected total number of queries is $O(wq^{m-\rho+1})$. The success probability of the algorithm is at least $1 - O(q^{-n/2+1})$. Finally, observe that in RQC, sk, d are picked uniformly at random from $\mathbb{F}_{q^m}^n$ s.t. $\|\mathsf{sk}\| = \|d\| = w$, $\mathsf{supp}(\mathsf{sk}) = \mathsf{supp}(d)$ and $1 \in \mathsf{supp}(\mathsf{sk}) = \mathsf{supp}(d)$. The fact that we know one vector of the subspace spanned by sk does not impact the attack but merely decreases the randomness of sk.

Row support recovery. The attack that recovers a vector subspace $\mathsf{supp}(u^T)$ which contains the row space of sk is nearly identical to the one above. The only difference is that we iterate over all $\alpha \in \mathbb{F}_q^n$ by setting $y \in \mathbb{F}_{q^m}^n$ s.t. $y = (\alpha_0 X, \alpha_1 X, \dots, \alpha_{n-1} X)$. We do not set $y = (\alpha_0, \alpha_1, \dots, \alpha_{n-1})$, otherwise $1 \in \mathsf{supp}(y)$ and thus $\|u + y\| \leq \rho$ for all α. Now, the row space of the secret key $\mathsf{supp}(\mathsf{sk}^T)$ is not necessarily equal to the row space of d. However, one can recover a subspace containing the latter in the exact same way. Indeed, the only difference is that we set $U = A, V = B + x$ for any $x \in \mathbb{F}_{q^m}^n$ and then $\mathcal{O}^{\mathsf{PCO}}(\mathsf{pt}, (U, V)) = 1 \iff \|V - U \times \mathsf{sk}\| = \|d + x\| \leq \rho$. Note that Lemma 2 still applies since $\rho < \frac{n}{2}$. The expected number of queries is upper bounded by $wq^{n-\rho+1}$.

Total cost. Hence, the total number of queries needed to recover the key is upper bounded by $wq^{\min\{m,n\}-\rho+1}$. For the CPA version of RQC-I (which targets 128-bit security), this amounts to roughly 2^{39} queries.

7.4 Hardness of learning in the rank metric

As the KR-PCA attack against RQC given above has an exponential complexity, one could wonder whether a polynomial attack would be possible. While not proving the hardness of the KR-PCA game in the RQC setting, we show here that the learning game is hard for small errors.

First, we state useful theorems and lemmas.

Theorem 2 (Corollary 8.1, [24]). *Let $X, Y \in \mathbb{F}^{m\times n}$ be two $m \times n$ matrices over a field $\mathbb{F}$, $c = \mathsf{dim}(\mathsf{span}(X) \cap \mathsf{span}(Y))$ and $d = \mathsf{dim}(\mathsf{span}(X^T) \cap \mathsf{span}(Y^T))$. Then,*

$$\mathsf{rank}(X) + \mathsf{rank}(Y) - c - d \leq \mathsf{rank}(X + Y) \leq \mathsf{rank}(X) + \mathsf{rank}(Y) - \max(c, d).$$

Theorem 2 directly implies the following corollary.

Corollary 1. *Let $x, y \in \mathbb{F}_{q^m}^n$ s.t. $\|x\| = w$, $\|y\| = z$ and $z \geq w$. Let $c = \mathsf{dim}(\mathsf{supp}(x) \cap \mathsf{supp}(y))$, $d = \mathsf{dim}(\mathsf{supp}(x^T) \cap \mathsf{supp}(y^T))$ and ρ be some positive integer. Then, if $z > \rho + w$*

$$\|x + y\| > \rho.$$

Theorem 3 (Intersection of subspaces). *Let $w, d, n \in \mathbb{N}$ and W be some random secret subspace of $\mathbb{F}_q^n$ of dimension w. We consider the following game. A participant who does not know W tries to find a subspace X of $\mathbb{F}_q^n$ of dimension d s.t. the intersection $W \cap X$ is non-trivial. The game stops when such a subspace is found. Then, the probability $p_{w,d}^{n,t}$ of success in t trials is*

$$p_{w,d}^{n,t} \leq \frac{t}{q^{n-d-w}}.$$

Proof. See Appendix B.2.

Now we can prove the hardness of the learning game in the rank metric setting.

Theorem 4 (Hardness of learning in the rank metric). *Let $q = 2, w, \rho, n, m$ and $d = \rho + w$ be some positive integers s.t. $w + d = \rho + 2w < \min\{m, n\}$. In addition, we consider $S_w^n = \{v \in \mathbb{F}_{q^m}^n : \|v\| = w\}$, Ψ the uniform distribution over S_w^n and $\|\cdot\|$ the rank distance. Then, for any ppt learning adversary $\mathcal{A}_t$ restricted to t number of queries with $t < q^{\min\{m,n\}-w-d}$, we have*

$$\mathsf{Adv}^{\mathrm{learn}}_{\Psi,\rho,\|\cdot\|}(\mathcal{A}_t) = \Pr[LEARN_{\Psi,\rho,\|\cdot\|}(\mathcal{A}_t) \Rightarrow 1] \leq \frac{t}{q^{n-w-d}} + \frac{t}{q^{m-w-d}} + \mathsf{negl}$$

where $\mathsf{negl} = \left(\begin{bmatrix} n \\ w \end{bmatrix}_q \prod_{i=0}^{w-1} (q^m - q^i) \right)^{-1}$.

Proof. See Appendix B.3.

Discussion. While not proving the hardness of KR-PCA attacks, Theorem 4 shows that the learning game in the rank metric is difficult for some parameters. As many reaction attacks are based on the capability to solve an instance of the learning game, this result is still significant. Note that when the error weight w is large, $q^{n-\rho-2w} \leq 1$ and the bound becomes meaningless. However, in most settings, the value w is picked small enough. For example, in RQC-I, we have $w = 5, \rho = 31, m = 97$ and $n = 67$. Therefore, the advantage of a t-bounded adversary is roughly bounded by $\frac{t}{2^{26}}$. This means that a number of queries of

the order of 2^{26} is necessary to win with good probability. While feasible, the cost is still exponential. More generally, if $\rho + 2w$ is smaller but proportional to n (and m), the learning problem requires an exponential number of queries in the rank metric.

From a broader perspective, this result shows that the rank distance leaks less information than other common norms. Indeed, as shown in [3], the learning problem for other distances such as the Hamming distance, the L_∞ norm in $\mathbb{Z}_q$ or some variants can be solved with a polynomial number of queries. One explanation is that the learning problem for other metrics can be solved component-wise. That is, by varying one component of x in the query, one can extract information only about the corresponding component in the secret value. Then, it is sufficient to recover the secret component by component. In the rank metric though, this strategy is not possible as varying one entry in the value x does not necessarily give information about a given component. More generally, this confirms the intuition that the rank leaks less information, as flipping one entry in a vector always changes the Hamming weight but not necessarily the rank.

This result tends to show that the rank metric may be well suited to resist to key misuse and similar attacks.

8 Conclusion

In this work, we have have presented key-reuse attacks against several NIST PQC round 2 candidates, namely Kyber, SABER, LAC, HQC and RQC. We have shown that for all but one of these schemes, a few thousands reuses can lead to the recovery of the secret key. In the model considered, the adversary only knows whether the decryption is a success or not.

As our misuse attack against RQC is borderline practical, we have demonstrated that for RQC-I parameters, similar attacks cannot be efficient. More generally, we proved that the distance between a secret and a given value leaks less information in the rank metric than in other metrics. While interpreting this result with care, this tends to show that practical reaction (or similar) attacks in the rank metric may not be as straightforward as in other metrics. We leave the proof of (im)possibility of efficient KR-PCA attacks against RQC-like schemes as an open problem.

Acknowledgements

Loïs Huguenin-Dumittan is supported by a grant (project Nº 192364) of the Swiss National Science Foundation (SNSF).

References

1. Albrecht, M.R., Hanser, C., Hoeller, A., Pöppelmann, T., Virdia, F., Wallner, A.: Implementing RLWE-based schemes using an RSA co-processor. Cryptology ePrint Archive, Report 2018/425 (2018), https://eprint.iacr.org/2018/425

2. Aragon, N., Gaborit, P., Hauteville, A., Tillich, J.: A new algorithm for solving the rank syndrome decoding problem. In: 2018 IEEE International Symposium on Information Theory (ISIT). pp. 2421–2425 (June 2018). https://doi.org/10.1109/ISIT.2018.8437464
3. Băetu, C., Durak, F.B., Huguenin-Dumittan, L., Talayhan, A., Vaudenay, S.: Misuse attacks on post-quantum cryptosystems. In: Ishai, Y., Rijmen, V. (eds.) Advances in Cryptology – EUROCRYPT 2019. pp. 747–776. Springer International Publishing, Cham (2019)
4. Bauer, A., Gilbert, H., Renault, G., Rossi, M.: Assessment of the key-reuse resilience of NewHope. In: Matsui, M. (ed.) Topics in Cryptology – CT-RSA 2019. pp. 272–292. Springer International Publishing, Cham (2019)
5. Bernstein, D.J., Groot Bruinderink, L., Lange, T., Panny, L.: HILA5 pindakaas: On the CCA security of lattice-based encryption with error correction. In: Joux, A., Nitaj, A., Rachidi, T. (eds.) Progress in Cryptology – AFRICACRYPT 2018. pp. 203–216. Springer International Publishing, Cham (2018)
6. Bettaieb, S., Bidoux, L., Gaborit, P., Marcatel, E.: Preventing timing attacks against RQC using constant time decoding of Gabidulin codes. In: Ding, J., Steinwandt, R. (eds.) Post-Quantum Cryptography. pp. 371–386. Springer International Publishing, Cham (2019)
7. Bleichenbacher, D.: Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS #1. In: Krawczyk, H. (ed.) Advances in Cryptology — CRYPTO '98. pp. 1–12. Springer Berlin Heidelberg, Berlin, Heidelberg (1998)
8. Carlson, D.: Matrix decompositions involving the Schur complement. SIAM Journal on Applied Mathematics **28**(3), 577–587 (1975), http://www.jstor.org/stable/2100380
9. D'Anvers, J.P., Karmakar, A., Roy, S.S., Vercauteren, F., Verbauwhede, I.: SABER: Mod-LWR based KEM. NIST Round 2 Submissions (2019), https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions
10. D'Anvers, J.P., Tiepelt, M., Vercauteren, F., Verbauwhede, I.: Timing attacks on error correcting codes in post-quantum secure schemes. Cryptology ePrint Archive, Report 2019/292 (2019), https://eprint.iacr.org/2019/292
11. Ding, J., Alsayigh, S., RV, S., Fluhrer, S., Lin, X.: Leakage of signal function with reused keys in RLWE key exchange. Cryptology ePrint Archive, Report 2016/1176 (2016), https://eprint.iacr.org/2016/1176
12. Fabsic, T., Hromada, V., Zajac, P.: A reaction attack on LEDApkc. Cryptology ePrint Archive, Report 2018/140 (2018), https://eprint.iacr.org/2018/140
13. Fluhrer, S.: Cryptanalysis of ring-LWE based key exchange with key share reuse. Cryptology ePrint Archive, Report 2016/085 (2016), https://eprint.iacr.org/2016/085
14. Fujisaki, E., Okamoto, T.: Secure integration of asymmetric and symmetric encryption schemes. In: Annual International Cryptology Conference. pp. 537–554. Springer (1999)
15. Fujisaki, E., Okamoto, T.: Secure integration of asymmetric and symmetric encryption schemes. Journal of Cryptology **26**(1), 80–101 (2013), https://doi.org/10.1007/s00145-011-9114-1
16. Gabidulin, E.: Theory of codes with maximum rank distance (translation). Problems of Information Transmission **21**, 1–12 (01 1985)
17. Gadouleau, M., Yan, Z.: Properties of codes with the rank metric. CoRR **abs/cs/0610099** (10 2006). https://doi.org/10.1109/GLOCOM.2006.173

18. Guo, Q., Johansson, T., Stankovski, P.: A key recovery attack on MDPC with CCA security using decoding errors. In: Cheon, J.H., Takagi, T. (eds.) Advances in Cryptology – ASIACRYPT 2016. pp. 789–815. Springer Berlin Heidelberg, Berlin, Heidelberg (2016)
19. Hall, C., Goldberg, I., Schneier, B.: Reaction attacks against several public-key cryptosystem. In: Varadharajan, V., Mu, Y. (eds.) Information and Communication Security. pp. 2–12. Springer Berlin Heidelberg, Berlin, Heidelberg (1999)
20. Hofheinz, D., Hövelmanns, K., Kiltz, E.: A modular analysis of the Fujisaki-Okamoto transformation. In: Theory of Cryptography Conference. pp. 341–371. Springer (2017)
21. Howgrave-Graham, N., Nguyen, P.Q., Pointcheval, D., Proos, J., Silverman, J.H., Singer, A., Whyte, W.: The impact of decryption failures on the security of NTRU encryption. In: Boneh, D. (ed.) Advances in Cryptology - CRYPTO 2003. pp. 226–246. Springer Berlin Heidelberg, Berlin, Heidelberg (2003)
22. Lepoint, T.: Algorithmic of LWE-based submissions to NIST post-quantum standardization effort. Presented at Post-Scryptum Spring School 2018 (2018), https://postscryptum.lip6.fr/tancrede.pdf
23. Lu, X., Liu, Y., Jia, D., Xue, H., He, J., Zhang, Z., Liu, Z., Yang, H., Li, B., Wang, K.: LAC (2019), https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions
24. Marsaglia, G., Styan, G.P.H.: Equalities and inequalities for ranks of matrices. Linear and Multilinear Algebra **2**(3), 269–292 (1974). https://doi.org/10.1080/03081087408817070
25. Melchor, C.A., Aragon, N., Bettaieb, S., Bidoux, L., Blazy, O., Couvreur, A., Deneuville, J.C., Gaborit, P., Hauteville, A., Zémor, G.: Rank quasi-cyclic (RQC). NIST Round 2 Submissions (2019), https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions
26. Melchor, C.A., Aragon, N., Bettaieb, S., Bidoux, L., Blazy, O., Deneuville, J.C., Gaborit, P., Persichetti, E., Zémor, G.: Hamming quasi-cyclic (HQC). NIST Round 2 Submissions (2019), https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions
27. Qin, Y., Cheng, C., Ding, J.: A complete and optimized key mismatch attack on NIST candidate NewHope. Cryptology ePrint Archive, Report 2019/435 (2019), https://eprint.iacr.org/2019/435
28. Qin, Y., Cheng, C., Ding, J.: An efficient key mismatch attack on the NIST second round candidate Kyber. Cryptology ePrint Archive, Report 2019/1343 (2019), https://eprint.iacr.org/2019/1343
29. Samardjiska, S., Santini, P., Persichetti, E., Banegas, G.: A reaction attack against cryptosystems based on LRPC codes. Cryptology ePrint Archive, Report 2019/845 (2019), https://eprint.iacr.org/2019/845
30. Schwabe, P., Avanzi, R., Bos, J., Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schanck, J.M., Seiler, G., Stehl, D.: CRYSTALS-Kyber (2019), https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions
31. Targhi, E.E., Unruh, D.: Post-quantum security of the Fujisaki-Okamoto and OAEP transforms. In: Theory of Cryptography Conference. pp. 192–216. Springer (2016)
32. Yu, Y., Zhang, J.: Lepton: Key encapsulation mechanisms from a variant of learning parity with noise. NIST Round 1 Submissions (2017), https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions

A Algorithms

Algorithm $\mathsf{lac\_kr\_pca}(\mathsf{pp})$
  $(A, B) \leftarrow \mathsf{pk}$;
  $\mathsf{pt} \leftarrow 0^k$
  $U \leftarrow -(\lceil \frac{q}{4} \rfloor - 1) \in \mathcal{R}_q$
  Find $\tilde{V}$ s.t. $\mathsf{decode}(V - U \times \mathsf{sk})$ detects t errors.
  $\mathcal{J} \leftarrow \{i : \tilde{V}_i \neq 1\}$
  for $i \in \mathcal{J}$ **do**
    $V \leftarrow \tilde{V}; V_i \leftarrow 1$
    $r \leftarrow \mathcal{O}^{pco}(\mathsf{pt}, (U, V))$
    if $r = 0$ **then**
      $\mathsf{sk}_i \leftarrow 1$; **continue**
    end if
    $V_i \leftarrow -2; r \leftarrow \mathcal{O}^{pco}(\mathsf{pt}, (U, V))$
    if $r = 0$ **then**
      $\mathsf{sk}_i \leftarrow -1$ **continue**
    end if
    $\mathsf{sk}_i \leftarrow 0$
  end for
  Set $\tilde{V}'$ s.t. $\mathsf{decode}(V - U \times \mathsf{sk})$ detects $t + 1$ errors ▷ E.g. we can set $\tilde{V}' \leftarrow V$ if one oracle call failed in the last iteration of the previous loop.
  $\mathcal{I} = \emptyset$
  for $i \in [\ell_v] \setminus \mathcal{J}$ **do**
    $V \leftarrow \tilde{V}'; V_i \leftarrow 0$
    $r \leftarrow \mathcal{O}^{pco}(\mathsf{pt}, (U, V))$
    if $r = 1$ **then**
      $\mathsf{sk}_i \leftarrow 1$; **continue**
    end if
    $\mathcal{I} \leftarrow \mathcal{I} \cup \{i\}$ ▷ $\mathsf{sk}_i \in \{-1, 0\}$
  end for
  for $i \in \mathcal{I}$ **do**
    $V \leftarrow \tilde{V}; V_i \leftarrow -2$
    $r \leftarrow \mathcal{O}^{pco}(\mathsf{pt}, (U, V))$
    if $r = 0$ **then**
      $\mathsf{sk}_i \leftarrow -1$; **continue**
    end if
    $\mathsf{sk}_i \leftarrow 0$
  end for
  return sk
end Algorithm

Fig. 3: KR-PCA adversary against LAC.

Algorithm $\mathsf{kyber\_kr\_pca}(\mathsf{pp})$
  $(A, B) \leftarrow \mathsf{pk};\ \rho \leftarrow \lceil \frac{q}{4} \rfloor$
  $\mathsf{pt} \leftarrow 0 \in \mathcal{R}_q$
  $U' \leftarrow \mathsf{compress}(-\rho/2, d_U); U'_2 \leftarrow \mathsf{compress}(\rho/2, d_U);$
  for $\ell \in \{1, \ldots, k\}$ **do**
    We write U' (resp. U'_2) for the vector in $\mathcal{R}_q^k$ with polynomial U' (resp. U'_2) at position ℓ and $0 \in \mathcal{R}_q$ elsewhere.
    for $i \in \{1, \ldots, n\}$ **do**
      $V' \leftarrow 0 \in \mathcal{R}_q$
      $a \leftarrow -2;\ b \leftarrow 2$
      while $b > a$ **do** ▷ Binary search to find sk_i
        $c \leftarrow \lceil \frac{b+a}{2} \rceil; V'_i \leftarrow -2 - c$ ▷ After decompression $V = -\rho - c \times \frac{\rho}{2}$
        if $c = 2$ **then** ▷ $c = 2$ special case
          $V'_i \leftarrow -1; r \leftarrow \mathcal{O}^{\mathsf{PCO}}(\mathsf{pt}, (U'_2, V'))$
          if $r = 1$ **then** $a \leftarrow 1$
          else $a \leftarrow 2$
          end if
          continue
        end if
        $r \leftarrow \mathcal{O}^{\mathsf{PCO}}(\mathsf{pt}, (U', V'))$
        if $r = 1$ **then** ▷ $\mathsf{sk}_i \geq c$
          $a \leftarrow c$
        else ▷ $\mathsf{sk}_i < c$
          $b \leftarrow c - 1$
        end if
      end while
      $\mathsf{sk}_i \leftarrow a$
    end for
  end for
  return sk
end Algorithm

Fig. 4: KR-PCA adversary against CRYSTALS-Kyber.

Algorithm saber_kr_pca(pk)
  $(A, B) \leftarrow \mathsf{pk}$; $\rho \leftarrow \lceil \frac{q}{4} \rfloor$
  $\mathsf{pt} \leftarrow 0^{256}$
  for $\ell \in \{1, \ldots, k\}$ **do**
    We write U for the vector in $\mathcal{R}_p^k$ with polynomial U at position ℓ and $0 \in \mathcal{R}_p$ elsewhere.
    $\mathcal{I} \leftarrow \emptyset$
    for $i \in \{1, \ldots, n\}$ **do**
      $V \leftarrow 2 \cdot X^i \in \mathcal{R}_T$
      $a \leftarrow -5$; $b \leftarrow 5$
      if $\mathcal{O}^{\mathsf{PCO}}(\mathsf{pt}, (30, V))$ **then** $a \leftarrow 2$
      else
        if $\mathcal{O}^{\mathsf{PCO}}(\mathsf{pt}, (-30, V))$ **then** $b \leftarrow -2$
        else
          $\mathcal{I} \cup \{i\}$; **continue**
        end if
      end if
      while $b > a$ **do** ▷ Binary search to find $\mathsf{sk}_{\ell,i}$
        $c \leftarrow \mathsf{sgn}(a+b)\lceil |\frac{b+a}{2}| \rceil$; $U \leftarrow \frac{60}{c}$ ▷ For $\eta = 10$, $c|60$ for all $c \in \{-5, \ldots, 5\}$
        if $\mathcal{O}^{\mathsf{PCO}}(\mathsf{pt}, (U, V))$ **then**
          if $c > 0$ **then** $a \leftarrow c$
          else $b \leftarrow c$
          end if
        else
          if $c > 0$ **then** $b \leftarrow c - 1$
          else $a \leftarrow c + 1$
          end if
        end if
      end while
      $\mathsf{sk}_{\ell,i} \leftarrow a$
    end for
    find two vectors $\tilde{V}^{\pm}$ s.t. $\mathcal{O}^{\mathsf{PCO}}(0^{256}, (\pm 60, \tilde{V}^{\pm})) = 1$
    for $i \in \mathcal{I}$ **do**
      $V \leftarrow \tilde{V}^{+} + 2X^i$
      if $\mathcal{O}^{\mathsf{PCO}}(\mathsf{pt}, (60, V))$ **then** $\mathsf{sk}_{\ell,i} \leftarrow 1$; **continue**
      end if
      $V \leftarrow \tilde{V}^{-} + 2X^i$
      if $\mathcal{O}^{\mathsf{PCO}}(\mathsf{pt}, (-60, V))$ **then** $\mathsf{sk}_{\ell,i} \leftarrow -1$; **continue**
      end if
      $\mathsf{sk}_{\ell,i} \leftarrow 0$
    end for
  end for
  return sk
end Algorithm

Fig. 5: KR-PCA adversary against LightSaber-CPA.

Algorithm $\mathsf{rqc\_kr\_pca}(\mathsf{pp})$
  $(A, B) \leftarrow \mathsf{pk}$;
  $\mathsf{pt} \leftarrow 0^k$
  $U \leftarrow -1$
  $x \leftarrow\!\!\$\ S^n_{\rho-w}$
  compute basis $\{\beta_i\}_{i\in[\rho-w]}$ of $\mathsf{span}(x)$
  $\mathcal{W} \leftarrow \{\beta_i\}_{i\in[\rho-w]}$
  for $\alpha \in \mathbb{F}_{q^m}$ **do**
    $y \leftarrow (\alpha, 0, \ldots, 0) \in \mathbb{F}^n_{q^m}$
    $V \leftarrow x + y$
    $r \leftarrow \mathcal{O}^{pco}(\mathsf{pt}, (U, V))$
    if $r = 1$ **then**
      if α not in subspace spanned by the elements of $\mathcal{W}$ **then**
        $\mathcal{W} = \mathcal{W} \cup \{\alpha\}$
      end if
      if $|\mathcal{W}| = \rho$ **then break**
      end if
    end if
  end for
  Set $\mathsf{sk}_i = \sum_{i=0}^{\rho-1} a_{i,j}\gamma_i$ and $d_i = \sum_{i=0}^{\rho-1} b_{i,j}\gamma_i$ with $\gamma_i \in \mathcal{W}$
  Solve $B = (A, 1) \cdot (\mathsf{sk}, d)^T$
  return sk
end Algorithm

Fig. 6: KR-PCA adversary against RQC.

B Proofs

B.1 Proof of Lemma 3

Proof. The proof of the first inequality is a simple union bound. The probability that a random non-zero random vector is in the subspace of dimension w is $\frac{(q^w-1)}{(q^n-1)}$ (i.e. the number of non-zero vectors in the subspace over the number of non-zero vectors in $\mathbb{F}_q^n$). Then, the probability that at least one of the q^k-1 non-zero vectors of the random subspace is in the given subspace is upper bounded by $(q^k-1)\frac{(q^w-1)}{(q^n-1)}$. The second bound is straightforward analysis: one can compute the following equivalence

$$(q^k-1)\frac{(q^w-1)}{(q^n-1)} \leq q^{w+k-n} \iff q^w+q^k-1 \geq \frac{q^{w+k}}{q^n}$$

which holds with $k+w \leq n$. □

B.2 Proof of Theorem 3

Proof. By a union bound, the probability of finding an intersection with a subspace of dimension d in a given trial is upper bounded by the probability of finding an intersection with a subspace of dimension 1 (i.e. a vector) in q^d-1 trials. Therefore, we have

$$p_{w,d}^{n,t} \leq p_{w,1}^{n,(q^d-1)t} \leq p_{w,1}^{n,t'} \tag{7}$$

for $t' = q^d t - 1$ (and $t > 0$). Then, in any of the t' trials, the probability that a given vector is in the secret subspace of dimension w is upper bounded by $\frac{q^w-1}{q^n-t'-1}$ (i.e. there are q^w-1 non-zero vectors in W and at most t' non-zero vectors have already been tried). Hence,

$$p_{w,1}^{n,t'} \leq t'\frac{q^w-1}{q^n-t'-1} \leq \frac{t'}{q^{n-w}} = \frac{t}{q^{n-d-w}} \tag{8}$$

where the first inequality follows from a union bound and the second holds iff $t'+1 \leq q^{n-w} \iff t \leq q^{n-w-d}$. As the theorem clearly holds for $t > q^{n-w-d}$ since $p_{w,d}^{n,t} \leq 1$, combining Eq. (7) and (8) concludes the proof. □

B.3 Proof of Theorem 4

Proof. The idea of the proof is to show that the oracle of the learning game is useful only if the adversary can find a non-trivial intersection with the subspace spanned by the columns or the rows of $\mathcal{M}(\delta)$. We proceed by the game hopping technique.

First, consider the learning game of Figure 2 but we replace the oracle with the oracle $\mathcal{O}^{\mathsf{G}_0}$ of Figure 7. We call this new game G_0. One can see that this

game is the same as the learning game. Indeed, by Corollary 1, the condition in line 2 returns the same result as $1_{\|\delta+x\|\leq\rho}$. Then, if the column (and row) spaces of $\mathcal{M}(x)$ and $\mathcal{M}(y)$ trivially intersect, we have $\|\delta+x\| = \|\delta\|+\|x\|$ by Theorem 1. Hence, line 9 returns the correct result because the condition on lines 5-6 did not hold. Finally, if this condition did hold, the result is obviously the same as in the original oracle. Now, consider the game G_1 which is the same as G_0 except it returns $1_{\|\delta\|+\|x\|\leq\rho}$ when both $\|x\| \leq \rho + w$ and condition on lines 5-6 of $\mathcal{O}^{\mathsf{G}_0}$ hold. Let's call this event *int*. Clearly, G_0 and G_1 are the same except when *int* happens.

We want to compute $\Pr[int]$, that is, the probability that the adversary finds some x s.t. $\|x\| \leq \rho + w$ and a non-trivial intersection with the column or row space of δ in less than t queries. Now, in the learning game, the oracle replies $1_{\|\delta\|+\|x\|\leq\rho}$ (which contains no extra information about δ) as long as *int* does not occur. Therefore, the probability of *int* to occur is upper bounded by the probability to find a non-trivial intersection in the row or column space in t tries with $\|x\| = \rho + w$. Hence, by a union bound and Theorem 3, we have

$$\Pr[int] \leq \frac{t}{q^{n-\rho-2w}} + \frac{t}{q^{m-\rho-2w}}.$$

In G_1, the oracle gives no information to the adversary, as $\|\delta\|$ and $\|x\|$ are known. Therefore, one can remove the oracle and the probability of success of the adversary is simply the probability to guess the correct value δ. The number of vectors in S_w^n is $\begin{bmatrix} n \\ w \end{bmatrix}_q \prod_{i=0}^{w-1}(q^m - q^i)$ (see [17] for example). Therefore,

$$\Pr[\mathsf{G}_1(\mathcal{A}_t) \Rightarrow 1] \leq \left(\begin{bmatrix} n \\ w \end{bmatrix}_q \prod_{i=0}^{w-1}(q^m - q^i) \right)^{-1}.$$

Hence,

$$\begin{aligned} \mathsf{Adv}^{\text{learn}}_{\Psi,\rho,\|\cdot\|}(\mathcal{A}_t) &\leq |\Pr[\mathsf{G}_0(\mathcal{A}_t) \Rightarrow 1] - \Pr[\mathsf{G}_1(\mathcal{A}_t) \Rightarrow 1]| + \Pr[\mathsf{G}_1(\mathcal{A}_t) \Rightarrow 1] \\ &\leq \Pr[int] + \Pr[\mathsf{G}_1(\mathcal{A}_t) \Rightarrow 1] \\ &\leq \frac{t}{q^{n-w-d}} + \frac{t}{q^{m-w-d}} + \left(\begin{bmatrix} n \\ w \end{bmatrix}_q \prod_{i=0}^{w-1}(q^m - q^i) \right)^{-1}. \end{aligned}$$

□

C HQC

The HQC [26] scheme works mainly in $\mathcal{R}_2$ and with the Hamming weight $\|x\| = |\{i : x_i \neq 0\}|$. In addition, let w_{sk}, w_t, w_f be some integers and $S_w = \{v : v \in \mathcal{R}_2, \|v\| = w\}$ be the set of polynomials in $\mathcal{R}_2$ with Hamming weight w. Then, the HQC scheme works as follows.

- gen: Sample $(\mathsf{sk}, d) \leftarrow\!\!\$\, S^2_{w_{sk}}$ and $A \leftarrow\!\!\$\, \mathcal{R}_2$. Set $\mathsf{pk} = (A, B = A \times \mathsf{sk} + d)$.

Oracle $\mathcal{O}^{\mathsf{G}_0}(x)$

$w \leftarrow \|\delta\|$
if $\|x\| > \rho + w$ **then**
  return $1_{\|\delta\|+\|x\|\leq\rho} = 0$
end if
if $\mathsf{supp}(x) \cap \mathsf{supp}(\delta) \neq \{0\}$ **or**
  $\mathsf{supp}(x^T) \cap \mathsf{supp}(\delta^T) \neq \{0\}$ **then**
  return $1_{\|\delta+x\|\leq\rho}$
end if
return $1_{\|\delta\|+\|x\|\leq\rho}$

Oracle $\mathcal{O}^{\mathsf{G}_1}(x)$

return $1_{\|\delta\|+\|x\|\leq\rho}$

Fig. 7: Oracles of games G_0 and G_1.

- $\mathsf{enc}(\mathsf{pk}, m \in \{0,1\}^k)$: Sample $(t, e, f) \leftarrow\!\!\$\, S^2_{w_t} \times S_{w_f}$. Then, the ciphertext is $(U, V) = (t \times A + e, t \times B + f + mG)$ where G is a generator matrix in $\mathbb{Z}_2^{k,n}$ for some linear $[k, n]$-code $\mathcal{C}$.
- $\mathsf{dec}(\mathsf{sk}, U, V)$: output $\mathsf{decode}(V - U \times \mathsf{sk})$ where decode is the decoding function of the code $\mathcal{C}$ generated by G.

Now, we have $V - U \times \mathsf{sk} = mG + \delta$ with $\delta = t \times d + f - e \times \mathsf{sk}$. Thus, the decoding (hence the decryption) is correct iff $\|\delta\| = \|t \times d + f - e \times \mathsf{sk}\| < \rho$ for some ρ. The goal is to recover δ and use the known relation $B = A \times \mathsf{sk} + d$. Then, $(t \times A + e) \times \mathsf{sk} = t \times B + f - \delta$ gives n linear equations in n unknowns in $\mathbb{Z}_q$ and we can solve for sk.

The code used in HQC is a composition of a d-repetitions code and BCH code. Namely,

$$\mathsf{decode} = \mathsf{decode}_{\mathrm{BCH}}\left(\mathsf{decode}_{\mathrm{REP}}(c)\right).$$

This is the same decoding function as the one in Lepton [32] and therefore one can use the same learning algorithm to deduce δ and thus obtain n linear equations in sk. A learning algorithm against Lepton decoding function was described in [3]. For HQC-128, it requires

$$n + \frac{n}{d}\log_2 d + \frac{n}{d} + \log_2 \frac{n}{d} \approx 2^{15}$$

oracle queries to recover sk, with $n = 24\,677$ and $d = 31$. In the revised version of HQC for the round 2, the polynomial V is truncated to fit into n_c coefficients at the end of the encryption, where n_c is the length of the code. Similarly, it is expanded by ℓ coefficients set to 0 before decryption, with $\ell = n - n_c$. As n is picked as the least prime larger than n_c, the value ℓ is typically very small (e.g. $\ell = 1$ for HQC-128). Still, this implies that we can only get n_c equations for n unknowns at the end of the attack. We can run twice the attack to obtain enough equations or use a bruteforce technique (if ℓ is small) to recover the full key sk.