On the impact of decryption failures on the security of LWE/LWR based schemes

Jan-Pieter D'Anvers, Frederik Vercauteren, and Ingrid Verbauwhede

imec-COSIC, KU Leuven
Kasteelpark Arenberg 10, Bus 2452, B-3001 Leuven-Heverlee, Belgium
firstname.lastname@esat.kuleuven.be

Abstract In this paper we investigate the impact of decryption failures on the chosen-ciphertext security of (Ring/Module)-Learning With Errors and (Ring/Module)-Learning with Rounding based primitives. Our analysis is split in three parts: First, we use a technique to increase the failure rate of these schemes called failure boosting. Based on this technique we investigate the minimal effort for an adversary to obtain a failure in 3 cases: when he has access to a quantum computer, when he mounts a multi-target attack and when he can only perform a limited number of oracle queries. Secondly, we examine the amount of information that an adversary can derive from failing ciphertexts. Finally, these techniques are combined in an attack on (Ring/Module)-LWE and (Ring/Module)-LWR based schemes with decryption failures. We provide both a theoretical analysis as well as an implementation to calculate the security impact and show that an attacker can significantly reduce the security of (Ring/Module)-LWE/LWR based schemes that have a relatively high failure rate. However, for the candidates of the NIST post-quantum standardization process that we assessed, the number of required oracle queries is above practical limits due to their conservative parameter choices.

1 Introduction

The recent developments in quantum computing have led to increased research into post-quantum cryptography and motivated NIST to organize a post-quantum cryptography standardization process, with the goal of standardizing one or more quantum-resistant public-key cryptographic primitives. Submissions originate from various fields within post-quantum cryptography, such as lattice-based, code-based and multivariate cryptography.

A lot of research has been done on the security of post-quantum cryptography, such as provable post-quantum secure transformations from IND-CPA to IND-CCA secure schemes [14,30,24,17], security estimates of post-quantum primitives [2,3] and provable reductions for various hard problems underlying the schemes [23,22,20,6].

A striking observation is that numerous proposed Key Encapsulation Mechanisms (KEM's) have a small failure probability during decryption, in which the

involved parties fail to derive a shared secret key. This is the case for the majority of schemes based on lattices, codes or Mersenne primes. The probability of such failure varies from 2^{-64} in Ramstake [29] to 2^{-216} in New Hope [26], with most of the failure probabilities lying around 2^{-128}. As this failure is dependent on the secret key, it might leak secret information to an adversary. However, as suggested by the wide range of failure probabilities in the NIST submissions, the implications of failures are still not well understood.

The first works on cryptanalysis of lattice-based schemes using decryption failures has primarily focused on chosen ciphertext attacks against schemes that are not IND-CCA secure. Jaulmes and Joux [16] presented such an attack on NTRU, which has been extended in [15] and [12]. Later, Fuhrer [10] described an attack against Ring-Learning with Errors (RLWE) schemes.

However, these attacks can be thwarted using an appropriate transformation to a chosen ciphertext secure scheme, such as the Fujisaki-Okamoto transformation [11]. Hofheinz et al. [14] and later Jiang et al. [17] proved a bound on the impact of the failure rate on an IND-CCA secure KEM that is constructed using this transformation, but their bounds are squared in the failure probability in the quantum oracle setting, which seems a very conservative result.

In the absence of a clear evaluation technique for the impact of the failure rate, most NIST submissions have chosen a bound on the decryption failure probability around 2^{-128} based on educated guessing. As far as we know, only NIST-submission Kyber [25] provides an intuitive reasoning for its security against decryption failure attacks, but this approximation is not tight. They introduced a methodology that uses Grover's search algorithm to find ciphertexts that have a relatively high probability of triggering a decryption failure.

1.1 Our contributions

In this paper we investigate the requirements for KEM's to resist decryption failure cryptanalysis. Having better security estimates can benefit the parameter selection process, resulting in improved security and efficiency. We focus on IND-CCA secure KEM's based on the (Ring/Module-)Learning with Errors and (Ring/Module-)Learning with Rounding paradigms. Nonetheless, the general method can also be applied to investigate the impact of failures on other schemes.

The exploitation of decryption failures of an IND-CCA secure cryptographic scheme proceeds in two main steps: obtaining ciphertexts that result in a decryption failure and estimating the secret based on these ciphertexts. In the first step, an adversary can use failure boosting to find 'weak' input vectors that artificially enlarge the failure rate of the scheme. In Section 3, we examine how an adversary can generate these 'weak' ciphertexts that increase the failure probability. We provide a theoretical framework and a Python implementation [1] to

[1] The software is available at `https://github.com/KULeuven-COSIC/PQCRYPTO-decryption-failures`

calculate an estimate of the minimum effort required for an adversary to obtain one failing ciphertext.

Once ciphertexts that trigger a decryption failure are collected, they can be used to estimate the secret. In Section 4, we study how much information is leaked by the collected failures. We develop a statistical model to estimate the secret from the failures and determine the residual entropy of the secret after a certain number of failures is collected. The estimate of the secret can be used to construct an easier problem that can be solved faster.

Section 5 combines failure boosting and the secret estimation technique from Section 4 to estimate the security of schemes based on (Ring/Module)-Learning with Errors and (Ring/Module)-Learning with Rounding under an attack exploiting decryption failures. We show that an attacker can significantly reduce the security of these schemes if he is able to perform sufficient decryption queries.

2 Preliminaries

2.1 Notation

Let $\mathbb{Z}_q$ be the ring of integers modulo q represented in $(-q/2, q/2]$, let R_q denote the ring $\mathbb{Z}_q[X]/(X^n+1)$ and let $R_q^{k_1 \times k_2}$ denote the ring of $k_1 \times k_2$ matrices over R_q. Matrices will be represented with bold uppercase letters, while vectors are represented in bold lowercase. Let $\boldsymbol{A}_{ij}$ denote the element on the i^{th} row and j^{th} column of matrix $\boldsymbol{A}$, and let $\boldsymbol{A}_{ijk}$ denote the k^{th} coefficient of this element. Denote with $\boldsymbol{A}_{:j}$ the j^{th} column of $\boldsymbol{A}$.

The rounding operation $\lfloor x \rceil_{q \to p}$ is defined as $\lfloor p/q \cdot x \rceil \in \mathbb{Z}_p$ for an element $x \in \mathbb{Z}_q$, while $\texttt{abs}(\cdot)$ takes the absolute value of the input. These operations are extended coefficient-wise for elements of R_q and $R_q^{k_1 \times k_2}$. The two-norm of a polynomial $a \in R_q$ is written as $\|a\|_2$ and defined as $\sqrt{\sum_i a_i^2}$, which is extended to vectors as $\|\boldsymbol{a}\|_2 = \sqrt{\sum_i \|\boldsymbol{a}_i\|_2^2}$. The notation $a \leftarrow \chi(R_q)$ will be used to represent the sampling of $a \in R_q$ according to the distribution χ. This can be extended coefficient-wise for $\boldsymbol{A} \in R_q^{k_1 \times k_2}$ and is denoted as $\boldsymbol{A} \leftarrow \chi(R_q^{k_1 \times k_2})$. Let $\mathcal{U}$ be the uniform distribution. Denote with $\chi_1 * \chi_2$ the convolution of the two distributions χ_1 and χ_2, and denote with $\chi^{*n} = \underbrace{\chi * \chi * \chi * \cdots * \chi * \chi}_{n}$ the n^{th} convolutional power of χ.

2.2 Cryptographic definitions

A Public Key Encryption (PKE) is defined as a triple of functions $\text{PKE} = (\texttt{KeyGen}, \texttt{Enc}, \texttt{Dec})$, where the key generation $\texttt{KeyGen}$ returns a secret key sk and a public key pk, where the encryption $\texttt{Enc}$ produces a ciphertext c from the public key pk and the message $m \in \mathcal{M}$, and where the decryption $\texttt{Dec}$ returns the message m' given the secret key sk and the ciphertext c.

A Key Encapsulation Mechanism (KEM) consists of a triple of functions $\text{KEM} = (\texttt{KeyGen}, \texttt{Encaps}, \texttt{Decaps})$, where $\texttt{KeyGen}$ generates the secret and public

keys sk and pk respectively, where `Encaps` generates a key $k \in \mathcal{K}$ and a ciphertext c from a public key pk, and where `Decaps` requires the secret key sk, the public key pk and the ciphertext c to return a key k or the decryption failure symbol $\perp$. The security of a KEM can be defined under the notion of indistinguishability under chosen ciphertext attacks (IND-CCA),

$$\mathrm{Adv}^{\text{ind-cca}}_{\text{KEM}}(A) = \left| P\left(b' = b : \begin{array}{c} (pk, sk) \leftarrow \texttt{KeyGen}(),\ b \leftarrow \mathcal{U}(\{0,1\}), \\ (c, d, k_0) \leftarrow \texttt{Encaps}(pk), \\ k_1 \leftarrow \mathcal{K},\ b' \leftarrow A^{\texttt{Decaps}}(pk, c, d, k_b), \end{array} \right) - \frac{1}{2} \right| .$$

2.3 LWE/LWR problems

The decisional Learning with Errors problem (LWE) [23] consists of distinguishing a uniform sample $(\boldsymbol{A}, \boldsymbol{U}) \leftarrow \mathcal{U}(\mathbb{Z}_q^{k_1 \times k_2} \times \mathbb{Z}_q^{k_1 \times m})$ from an LWE-sample $(\boldsymbol{A}, \boldsymbol{B} = \boldsymbol{AS} + \boldsymbol{E})$, were $\boldsymbol{A} \leftarrow \mathcal{U}(\mathbb{Z}_q^{k_1 \times k_2})$ and where the secret vectors $\boldsymbol{S}$ and $\boldsymbol{E}$ are generated from the small distributions $\chi_s(\mathbb{Z}_q^{k_2 \times m})$ and $\chi_e(\mathbb{Z}_q^{k_1 \times m})$ respectively. The search LWE problem states that it is hard to recover the secret $\boldsymbol{S}$ from the LWE sample.

This definition can be extended to Ring- or Module-LWE [20,18] by using vectors of polynomials. In this case, the problem is to distinguish the uniform sample $(\boldsymbol{A}, \boldsymbol{U}) \leftarrow \mathcal{U}(R_q^{k_1 \times k_2} \times R_q^{k_1 \times m})$ from a generalized LWE sample $(\boldsymbol{A}, \boldsymbol{B} = \boldsymbol{AS} + \boldsymbol{E})$ in which $\boldsymbol{A} \leftarrow \mathcal{U}(R_q^{k_1 \times k_2})$ and where the secret vectors $\boldsymbol{S}$ and $\boldsymbol{E}$ are generated from the small distribution $\chi_s(R_q^{k_2 \times m})$ and $\chi_e(R_q^{k_1 \times m})$ respectively. Analogous to the LWE case, the search problem is to recover the secret $\boldsymbol{S}$ from a generalized LWE sample.

The decisional generalized Learning with Rounding (LWR) problem [6] is defined as distinguishing the uniform sample $(\boldsymbol{A}, \lfloor \boldsymbol{U} \rceil_{q \to p})$, with $\boldsymbol{A} \leftarrow \mathcal{U}(R_q^{k_1 \times k_2})$ and $\boldsymbol{U} \leftarrow \mathcal{U}(R_q^{k_1 \times m})$ from the generalized LWR sample $(\boldsymbol{A}, \boldsymbol{B} = \lfloor \boldsymbol{AS} \rceil_{q \to p})$ with $\boldsymbol{A} \leftarrow \mathcal{U}(R_q^{k_1 \times k_2})$ and $\boldsymbol{S} \leftarrow \chi_s(R_q^{k_2 \times m})$. In the analogous search problem, one has to find $\boldsymbol{S}$ from a generalized LWR sample.

2.4 (Ring/Module-)LWE based encryption

Let `gen` be a pseudorandom generator that expands $\text{seed}_{\boldsymbol{A}}$ into a uniformly random distributed matrix $\boldsymbol{A} \in R_q^{k \times k}$. Define `enc` as an encoding function that transforms a message m into a polynomial representation, and `dec` as the inverse decoding function. A general (Ring/Module-)LWE based PKE, consisting of a key generation, an encryption and a decryption phase, can then be constructed as described in Algorithms 1 to 3 respectively. The randomness required for the generation of the secrets $\boldsymbol{S}'_B$, $\boldsymbol{E}'_B$ and $\boldsymbol{E}''_B$ during the encryption, is generated pseudorandomly from the uniformly distributed seed r that is given as an input.

Using this general framework, specific schemes can be described with appropriate parameter choices. When the ring R_q is chosen as $\mathbb{Z}_q$, the encryption is LWE-based as can be seen in FrodoKEM [21] and Emblem [27]. A value of

Algorithm 1: $\texttt{PKE.KeyGen}()$

$seed_{\boldsymbol{A}} \leftarrow \mathcal{U}(\{0,1\}^{256})$
$\boldsymbol{A} = \texttt{gen}(\text{seed}_{\boldsymbol{A}}) \in R_q^{l \times l}$
$\boldsymbol{S}_A \leftarrow \chi_s(R_q^{l \times m}), \boldsymbol{E}_A \leftarrow \chi_e(R_q^{l \times m})$
$\boldsymbol{B} = \lfloor \boldsymbol{A}\boldsymbol{S}_A + \boldsymbol{E}_A \rceil_{q \to p}$
return $(pk := (\boldsymbol{B}, seed_{\boldsymbol{A}}), sk := \boldsymbol{S}_A)$

Algorithm 2: $\texttt{PKE.Enc}(pk = (\boldsymbol{B}, seed_{\boldsymbol{A}}), m, r)$

$\boldsymbol{A} = \texttt{gen}(\text{seed}_{\boldsymbol{A}}) \in R_q^{l \times l}$
$\boldsymbol{S}'_B \leftarrow \chi_s(R_q^{l \times m}), \boldsymbol{E}'_B \leftarrow \chi_e(R_q^{l \times m})$
$\boldsymbol{E}''_B \leftarrow \chi_e(R_q^{m \times m})$
$\boldsymbol{B}_r = \lceil \boldsymbol{B} \rfloor_{p \to q}$
$\boldsymbol{B}' = \lfloor \boldsymbol{A}^T \boldsymbol{S}'_B + \boldsymbol{E}'_B \rceil_{q \to p}$
$\boldsymbol{V}' = \lfloor \boldsymbol{B}_r^T \boldsymbol{S}'_B + \boldsymbol{E}''_B + \texttt{enc}(m) \rceil_{q \to t}$
return $c = (\boldsymbol{V}', \boldsymbol{B}')$

$l = 1$ indicates a Ring-LWE based scheme including New Hope [4], LAC [19], LIMA [28] or R.Emblem [27]. If $l \neq 1$ and $R_q \neq \mathbb{Z}_q$, the scheme is based on the Module-LWE hard problem such as Kyber [7]. The special case that $\chi_e = 0$ corresponds to (Module/Ring)-LWR-based schemes such as Round2 [5] and Saber [9]. In Lizard [8], a combination of an LWE and LWR problem is proposed. In most (Ring/Module-)LWE based schemes, $q = p$ and no rounding is performed in the calculation of $\boldsymbol{B}$ and $\boldsymbol{B}'$, while t is in most schemes much smaller than q leading to a drastic rounding of $\boldsymbol{V}'$.

We define $\boldsymbol{U}_A, \boldsymbol{U}'_B$ en $\boldsymbol{U}''_B$ as the errors introduced by the rounding operations, which is formalized as follows:

$$\boldsymbol{U}_A = \boldsymbol{A}\boldsymbol{S}_A + \boldsymbol{E}_A - \boldsymbol{B}_r \tag{1}$$
$$\boldsymbol{U}'_B = \boldsymbol{A}^T \boldsymbol{S}'_B + \boldsymbol{E}'_B - \boldsymbol{B}'_r \tag{2}$$
$$\boldsymbol{U}''_B = \boldsymbol{B}_r^T \boldsymbol{S}'_B + \boldsymbol{E}''_B + \texttt{enc}(m) - \boldsymbol{V}'_r \tag{3}$$

Let $\boldsymbol{S}$ be the vector constructed as the concatenation of the vectors $-\boldsymbol{S}_A$ and $\boldsymbol{E}_A + \boldsymbol{U}_A$, let $\boldsymbol{C}$ be the concatenation of $\boldsymbol{E}'_B + \boldsymbol{U}'_B$ and $\boldsymbol{S}'_B$, and let $\boldsymbol{G} = \boldsymbol{E}''_B + \boldsymbol{U}''_B$. An attacker that generates ciphertexts can compute $\boldsymbol{C}$ and $\boldsymbol{G}$ and tries to obtain information about $\boldsymbol{S}$. These variables are summarized below:

$$\boldsymbol{S} = \begin{pmatrix} -\boldsymbol{S}_A \\ \boldsymbol{E}_A + \boldsymbol{U}_A \end{pmatrix} \quad \boldsymbol{C} = \begin{pmatrix} \boldsymbol{E}'_B + \boldsymbol{U}'_B \\ \boldsymbol{S}'_B \end{pmatrix} \quad \boldsymbol{G} = \boldsymbol{E}''_B + \boldsymbol{U}''_B \tag{4}$$

After the execution of this protocol, the two parties will arrive at the same key if the decoding $\texttt{dec}(\boldsymbol{V}'_r - \boldsymbol{V})$ equals m. The term $\boldsymbol{V}'_r - \boldsymbol{V}$ can be rewritten as

Algorithm 3: $\texttt{PKE.Dec}(sk = \boldsymbol{S}_A, c = (\boldsymbol{V'}, \boldsymbol{B'}))$

$\boldsymbol{B}'_r = \lfloor \boldsymbol{B'} \rceil_{p \to q}$
$\boldsymbol{V}'_r = \lfloor \boldsymbol{V'} \rceil_{t \to q}$
$\boldsymbol{V} = \boldsymbol{B}'^T_r \boldsymbol{S}_A$
$m' = \texttt{dec}(\boldsymbol{V}'_r - \boldsymbol{V})$
return m'

Algorithm 4: $\texttt{KEM.Encaps}(pk)$

$m \leftarrow \mathcal{U}(\{0,1\}^{256})$
$r = \mathcal{G}(m)$
$c = \texttt{PKE.Enc}(pk, m, r)$
$K = \mathcal{H}(r)$
return (c, K)

$(\boldsymbol{E}_A + \boldsymbol{U}_A)^T \boldsymbol{S}'_B - \boldsymbol{S}_A^T(\boldsymbol{E}'_B + \boldsymbol{U}'_B) + (\boldsymbol{E}'' + \boldsymbol{U}''_B) + \texttt{enc}(m) = \boldsymbol{S}^T\boldsymbol{C} + \boldsymbol{G} + \texttt{enc}(m)$. The message can be recovered if and only if $\texttt{abs}(\boldsymbol{S}^T\boldsymbol{C} + \boldsymbol{G}) < q_t$ for a certain threshold q_t that is scheme dependent.

We will say that a (decryption) failure occurred if the parties do not arrive at a common key due to a coefficient of $\texttt{abs}(\boldsymbol{S}^T\boldsymbol{C}+\boldsymbol{G})$ that is larger than q_t, and will define $F(\boldsymbol{C}, \boldsymbol{G})$ as the probability of a decryption failure given $\boldsymbol{C}$ and $\boldsymbol{G}$ averaged over all $\boldsymbol{S}$, which can be expressed as $\sum_{\boldsymbol{S}} P(\texttt{abs}(\boldsymbol{S}^T\boldsymbol{C} + \boldsymbol{G}) > q_t \mid \boldsymbol{S})P(\boldsymbol{S})$.

2.5 Fujisaki-Okamoto transformation

Using the Fujisaki-Okamoto transform [11,14], one can transform a chosen plaintext secure PKE to an IND-CCA secure KEM. On top of the encryption from the PKE, the KEM defines an encapsulation and decapsulation function as described in Algorithms 4 and 5, using hash functions $\mathcal{H}$ and $\mathcal{G}$.

3 Failure boosting

In this section, we will develop a method to estimate the minimum amount of work to obtain one ciphertext that triggers a decryption failure. In contrast to an honest party that generates ciphertexts randomly, an attacker can search for ciphertexts that have a higher failure probability than average, which will be called 'weak'. As $\boldsymbol{C}$ and $\boldsymbol{G}$ are the only terms with which an attacker can influence decryption failures, the search for weak ciphertexts boils down to the search for weak $(\boldsymbol{C}, \boldsymbol{G})$. However, the pair $(\boldsymbol{C}, \boldsymbol{G})$ is generated through a hash $H()$ with random seed r, and during decryption it is checked whether the generator of the ciphertext knew the preimage r of $(\boldsymbol{C}, \boldsymbol{G})$. Therefore an attacker is forced to resort to a brute force search, which can be sped up at most quadratically using Grover's algorithm [13].

Algorithm 5: $\texttt{KEM.Decaps}(sk, pk, c)$

$m' = \texttt{PKE.Dec}(sk, c)$
$r' = \mathcal{G}(m')$
$c' = \texttt{PKE.Enc}(pk, m', r')$
if $c = c'$ **then**
  return $K = \mathcal{H}(r)$
else
  return $K = \perp$

To find a criterion for our search, we sort all possible $(\boldsymbol{C}, \boldsymbol{G})$ according to an increasing failure probability $F(\boldsymbol{C}, \boldsymbol{G})$. This list can then be divided into two sets using a threshold failure probability f_t: weak vectors with a failure probability higher or equal than f_t, and strong vectors with lower failure probability. Let $f()$ be the deterministic function that generates $\boldsymbol{C}$ and $\mathbf{G}$ from the random seed r. For a certain f_t, we can calculate the probability of generating a weak pair: $\alpha = P(F(\boldsymbol{C}, \boldsymbol{G}) > f_t \,|\, r \leftarrow \mathcal{U}, (\boldsymbol{C}, \boldsymbol{G}) = f(H(r)))$, and the probability of a decryption failure when a weak pair is used: $\beta = P(\texttt{abs}(\boldsymbol{S}^T\boldsymbol{C} + \boldsymbol{G}) > q_t \,|\, r \leftarrow \mathcal{U}, (\boldsymbol{C}, \boldsymbol{G}) = f(H(r)), F(\boldsymbol{C}, \boldsymbol{G}) > f_t)$.

The amount of work for an adversary to find a weak pair $(\boldsymbol{C}, \boldsymbol{G})$ is proportional to α^{-1}, but can be sped up quadratically using Grover's algorithm on a quantum computer, resulting in an expected workload of $\sqrt{\alpha^{-1}}$. On the other hand, the probability of a decryption failure given a weak pair cannot be improved using quantum computation assuming that the adversary has no quantum access to the decryption oracle. This assumption is in agreement with the premise in the NIST Post-Quantum Standardization Call for Proposals [1]. The expected work required to find a decryption failure given f_t is therefore the expected number of queries using weak ciphertexts times the expected amount of work to find a weak ciphertext, or $(\alpha \cdot \beta)^{-1}$ with a classical computer and $(\sqrt{\alpha} \cdot \beta)^{-1}$ with a quantum computer. An optimization over f_t gives the minimal effort to find one decryption failure.

3.1 Practical calculation

For most schemes, the full sorted list $(\boldsymbol{C}, \boldsymbol{G})$ is not practically computable, but using some observations and assumptions, an estimate can be found. The next three steps aim at calculating the distribution of the failure probability $F(\boldsymbol{C}, \boldsymbol{G})$, i.e. what is the probability of finding a $(\boldsymbol{C}, \boldsymbol{G})$ pair with a certain failure probability f. This distribution gives enough information to calculate α and β for a certain f_t.

First, we can remove the hash $H(.)$ in the probability expression by assuming the output of $f(H(.))$ given random input r to behave as the probability distributions (χ_C, χ_G), resulting in: $\alpha = P(F(\boldsymbol{C}, \boldsymbol{G}) > f_t \,|\, (\boldsymbol{C}, \boldsymbol{G}) \leftarrow (\chi_C, \chi_G))$ and $\beta = P(\texttt{abs}(\boldsymbol{S}^T\boldsymbol{C} + \boldsymbol{G}) > q_t \,|\, (\boldsymbol{C}, \boldsymbol{G}) \leftarrow (\chi_C, \chi_G), F(\boldsymbol{C}, \boldsymbol{G}) > f_t)$.

Secondly, we assume that the coefficients of $\boldsymbol{S}^T\boldsymbol{C}$ are normally distributed, which is reasonable as the coefficients are a sum of $2(l \cdot n)$ distributions that are somewhat close to a Gaussian. The coefficients of the polynomial $(\boldsymbol{S}^T\boldsymbol{C})_{ij}$ will be distributed with mean $\mu = 0$ because of symmetry around 0, while the variance can be calculated as follows, after defining χ_{e+u} as the distribution of the coefficients of $\boldsymbol{E}_A + \boldsymbol{U}_A$:

$$\begin{aligned}
\texttt{var}((\boldsymbol{S}^T\boldsymbol{C})_{ijk}) =& \texttt{var}(\sum_{i=0}^{l-1}\sum_{k=0}^{n-1}\boldsymbol{C}_{ijk}s_{ijk} + \sum_{i=l}^{2l-1}\sum_{k=0}^{n-1}\boldsymbol{C}_{ijk}e_{ijk}) && (5)\\
& \text{where: } s_{ijk} \leftarrow \chi_s \text{ and } e_{ijk} \leftarrow \chi_{e+u} && (6)\\
=& \sum_{i=0}^{l-1}\sum_{k=0}^{n-1}\boldsymbol{C}_{ijk}^2\texttt{var}(\chi_s) + \sum_{i=l}^{2l-1}\sum_{k=0}^{n-1}\boldsymbol{C}_{ijk}^2\texttt{var}(\chi_{e+u}) && (7)\\
=& \|(\boldsymbol{E}'_B + \boldsymbol{U}'_B)_{:j}\|_2^2\texttt{var}(\chi_s) + \|(\boldsymbol{S}'_B)_{:j}\|_2^2\texttt{var}(\chi_{e+u}) && (8)
\end{aligned}$$

Therefore, the variance of the coefficients of $\boldsymbol{S}^T\boldsymbol{C}$ for a given $\boldsymbol{C}$ is the same for all coefficients in the same column. This variance will be denoted as σ_j^2 for coefficients in the j^{th} column of $\boldsymbol{S}^T\boldsymbol{C}$. Furthermore, following the Gaussian assumption, the failure probability given σ_j^2 is the same as the failure probability given the j^{th} column of $\boldsymbol{C}$.

In the third step we gradually calculate the distribution of the failure probability. We start from the distribution of the failure probability of the coefficient at the ijk^{th} position given σ_j, denoted with $\chi_{coef\,|\,\sigma}$. This distribution expresses the probability of finding a $\boldsymbol{G}$ so that the failure probability is equal to f_{ijk} given a certain value of $\boldsymbol{C}$ (or equivalently σ_j^2) and can be expressed as follows:

$$\begin{aligned}
& P(f_{ijk}\,|\,\boldsymbol{G} \leftarrow \chi_G, \boldsymbol{C}) && (9)\\
& && (10)
\end{aligned}$$

where:

$$\begin{aligned}
f_{ijk} &= P(\texttt{abs}(\boldsymbol{S}^T\boldsymbol{C} + \boldsymbol{G})_{ijk} > q_t\,|\,\boldsymbol{G}, \boldsymbol{C}) && (11)\\
&\approx P(\texttt{abs}(x + \boldsymbol{G}_{ijk}) > q_t\,|\,\boldsymbol{G}, x \leftarrow \mathcal{N}(0, \sigma_j^2), \sigma_j^2) && (12)
\end{aligned}$$

The distribution $\chi_{col\,|\,\sigma}$, which models the probability of a failure in the j^{th} column of $\texttt{abs}(\boldsymbol{S}^T\boldsymbol{C} + \boldsymbol{G})$ given σ_j^2, can be calculated using the convolution of the distributions of the mn individual coefficient failures $\chi_{coef\,|\,\sigma}$ as follows:

$$\chi_{col\,|\,\sigma} = \chi_{coef\,|\,\sigma}^{*nm} \tag{13}$$

The conditioning on σ_j^2 is necessary to counter the dependency between the coefficients of the columns of $\texttt{abs}(\boldsymbol{S}^T\boldsymbol{C} + \boldsymbol{G})$, which are dependent as a result of sharing the same variance σ_j^2.

The distribution of failure probabilities in the j^{th} column of $\boldsymbol{S}^T\boldsymbol{C}$, denoted as χ_{col}, can then be calculated using a weighted average over the possible values of σ_j^2 as follows:

$$\chi_{col} = \sum_{l_c} P(f \mid f \leftarrow \chi_{col,\sigma}^{*nm}) P(\sigma_j^2 = l_c) \tag{14}$$

Finally we can calculate the full failure distribution χ_{FAIL} as the convolution of the m probability distributions corresponding to the failure distributions of the different columns. This convolution does not have the dependency on σ_j^2 as failures of different columns are independent conditioned on $\boldsymbol{C}$ and $\boldsymbol{G}$.

$$\chi_{\text{FAIL}} = \chi_{col}^{*m} \tag{15}$$

From this failure distribution, we can calculate α and β for an arbitrary value of f_t:

$$\alpha = P(f > f_t \mid f \leftarrow \chi_{\text{FAIL}}) \tag{16}$$

$$\beta = \frac{\sum_{f > f_t} f \cdot P(f \mid f \leftarrow \chi_{\text{FAIL}})}{\alpha} \tag{17}$$

We want to stress that this calculation is not exact, mainly due to the Gaussian assumption in the second step. More accurate estimates could be obtained with a more accurate approximation in step 2, tailored for a specific scheme. In this case, the assumptions and calculations of step 1 and step 3 remain valid. For the estimations of LAC [19] in subsequent paragraphs, we followed their approach for the calculation of the effect of the error correcting code. Note that this is not an exact formula as the inputs of the error correcting code are correlated through their polynomial structure.

In Figure 1 we compare the values of α and β calculated using the technique described above, with exhaustively tested values on a variant of LAC128 without error correction. For step 2 of the practical calculation, we use both a Gaussian approximation as well as a binomial approximation that is more tailored for LAC. We can observe that our estimation of the effect of failure boosting is relatively close to reality.

3.2 Applications of failure boosting

Failure boosting is a useful technique in at least three scenarios: first, if there is no multi-target protection, second, if the adversary can only perform a limited number of queries to the decryption oracle and third, if the adversary has access to a quantum computer.

In some (Ring/Module-)LWE/LWR schemes, the seed r is the only input to the pseudorandom generator that generates $\boldsymbol{C}$ and $\boldsymbol{G}$. This paves the way to

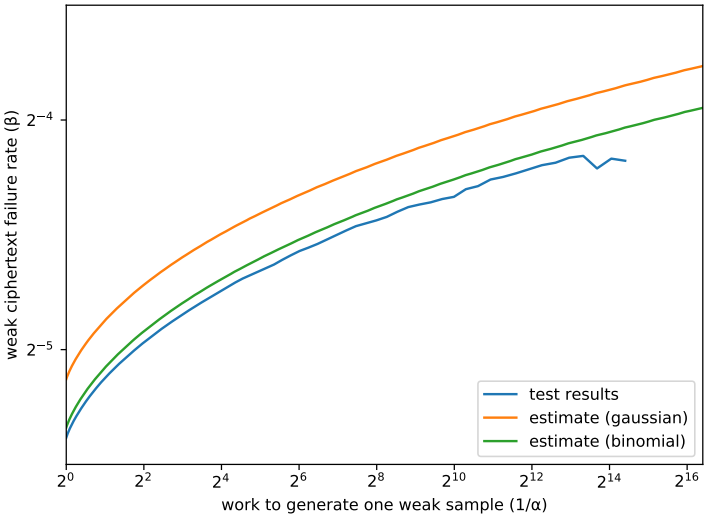

Figure 1: The failure rate of one weak ciphertext (β) as a function of the work required to generate one weak ciphertext (α) on a classical computer for LAC128 without error correction.

a multi-target attack where precomputed weak values of r can be used against multiple targets: after choosing the parameter f_t, the adversary can generate weak ciphertexts in approximately α^{-1} time ($\sqrt{\alpha^{-1}}$ if he has access to a quantum computer). Each precomputed sample has then a failure probability of β against every target. Figure 2 shows the failure probability of one weak ciphertext versus the amount of work to generate that ciphertext on a classical computer. Multi-target protection, for example by including the public key into the generation of $\boldsymbol{C}$ en $\boldsymbol{G}$ as proposed in Kyber [7] and Saber [9] is a relatively cheap option to resolve this issue.

If the adversary can only perform a limited number of decryption queries, for example 2^{64} in the NIST Post-Quantum Standardization Call for Proposals [1], the adversary can use failure boosting to reduce the number of required decryption queries. To this end, he chooses the parameter f_t so that the inverse of the failure probability β^{-1} equals the decryption query limit n_d, which results in a probability of finding a decryption failure of approximately $(1-e^{-1}) \approx 0.63$. To find i failures with similar probability, the failure probability should be brought up so that $\beta^{-1} = n_d/i$. Since the amount of work to generate one input of the decryption query is approximately α^{-1} ($\sqrt{\alpha^{-1}}$ quantumly), the total amount of work expected is $\alpha^{-1}\beta^{-1}$, ($\sqrt{\alpha^{-1}}\beta^{-1}$ quantumly). Figure 3 shows the expected total amount of work to find one decryption failure with a classical computer, versus the failure rate of one weak ciphertext.

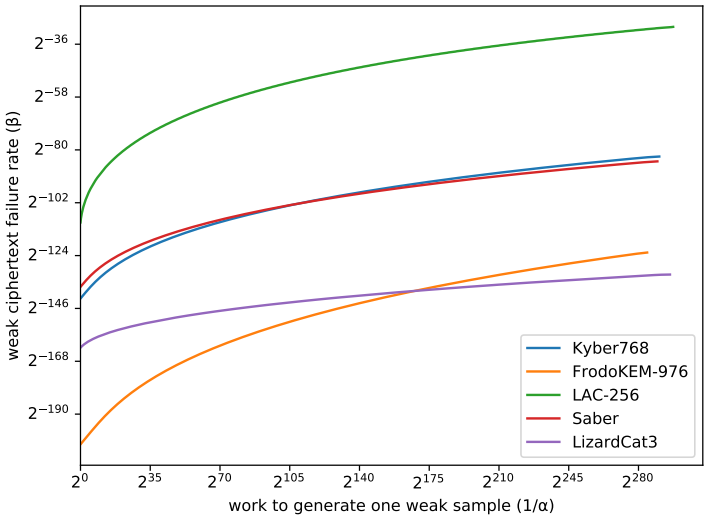

Figure 2: The failure rate of one weak ciphertext (β) as a function of the work required to generate one weak ciphertext (α) on a classical computer.

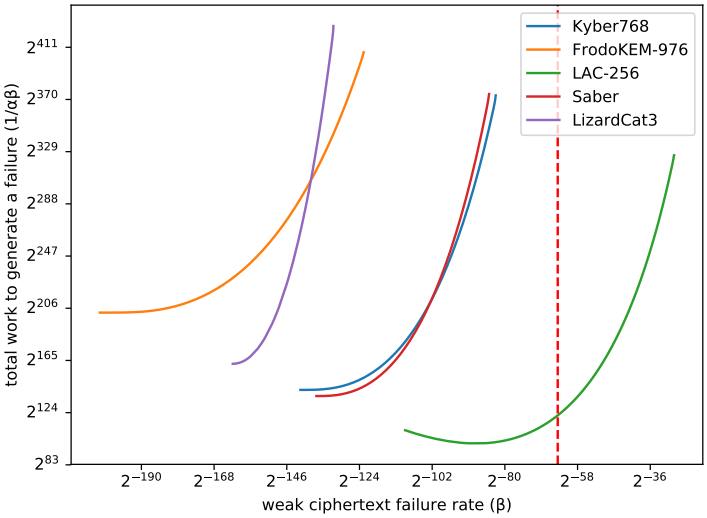

Figure 3: The expected amount of work ($\alpha^{-1}\beta^{-1}$) on a classical computer, as a function of the failure rate of one weak ciphertext (β). The red dotted line indicates a failure rate of 2^{-64}.

An adversary with a quantum computer always benefits from failure boosting, as the search for weak ciphertexts can be sped up using Grover's algorithm. However, this speedup is not quadratic if the adversary has no quantum access to the decryption oracle. Figure 4 shows the total amount of expected work to find one decryption failure, versus the amount of work to find one weak ciphertext on a quantum computer $\sqrt{\alpha^{-1}}$.

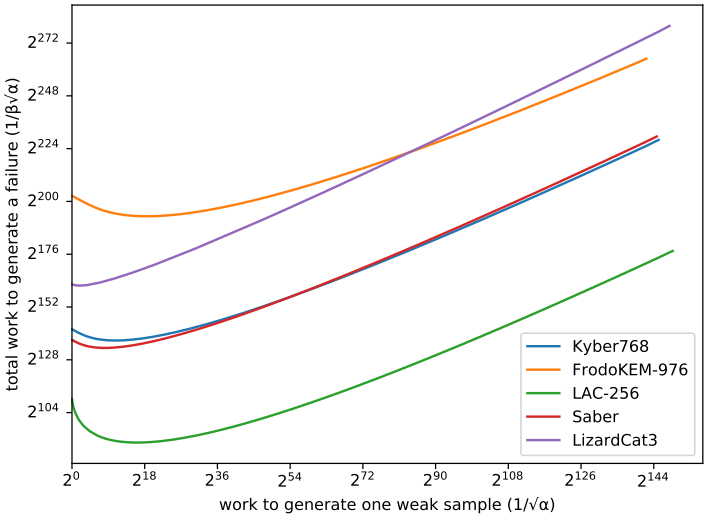

Figure 4: The expected amount of work ($\sqrt{\alpha^{-1}}\beta^{-1}$) as a function of the work required to generate one weak ciphertext ($\sqrt{\alpha^{-1}}$) on a classical computer.

4 Estimation of the secret

Finding a decryption failure does not immediately break the security of the KEM, but it does provide extra information to an adversary. In this section we will investigate how much this information leaks about the secret. An adversary that has obtained ciphertexts that produce decryption failures can use them to make an estimation of the secret $\boldsymbol{S}$.

When a failure occurs, we know that at least one coefficient of $\texttt{abs}(\boldsymbol{S}^T\boldsymbol{C}+\boldsymbol{G})$ is larger than the threshold q_t. This leads to a classification of the coefficients in the set of fail coefficients v_f and no-fail coefficients v_s. To each coefficient at position (i, j, k), a vector of integers $\boldsymbol{s}$ can be associated by taking the coefficients of $\boldsymbol{S}_{:i}$. Similarly, the coefficient can be linked to a vector of integers $\boldsymbol{c}$ calculated as a function of $\boldsymbol{C}_{:j}$ and k, so that the multiplication $\boldsymbol{sc}$ equals that coefficient.

No-fail vectors will contain negligible information about the secret $\boldsymbol{s}$, but failure vectors do carry clues, as the threshold exceeding value of the coefficients

of $\boldsymbol{S}^T\boldsymbol{C}+\boldsymbol{G}$ implies a correlation between the corresponding $\boldsymbol{c}$ and $\boldsymbol{s}$. This correlation can be positive, in case of a large positive value of the coefficient, or negative, in case of a large negative value of the coefficient. Consequently, the fail coefficients can be further divided into the sets of positive v_{fp} and negative v_{fn} fail coefficients respectively. Moreover, negative fail vectors can be transformed into positive fail vectors by multiplication with -1. Note that failure coefficients at position (i,j,k) will only contain information about the j^{th} column of $\boldsymbol{S}$, which is why the estimation of the columns of $\boldsymbol{S}$ can be performed independently.

4.1 One positive failure vector

We will first examine the case where we know one positive fail vector $\boldsymbol{c}$ and associated coefficient $\boldsymbol{G}_{i,j,k}$, which we will denote with g. This corresponds to the case where one failing ciphertext and the location and sign of the error is known. The question is how much the knowledge about $\boldsymbol{c}$ and g can improve our estimate of the associated secret $\boldsymbol{s}$. Applying Bayes' theorem and assuming independence between the coefficients of $\boldsymbol{c}$ and $\boldsymbol{s}$ that are on different positions, we can write:

$$P(\boldsymbol{s}_i \,|\, \boldsymbol{c}, g, \boldsymbol{sc} > q_t - g) \approx P(\boldsymbol{s}_i \,|\, \boldsymbol{c}_i, g, \boldsymbol{sc} > q_t - g) \tag{18}$$

$$= \frac{P(\boldsymbol{sc} > q_t - g \,|\, \boldsymbol{s}_i, \boldsymbol{c}_i, g) P(\boldsymbol{s}_i \,|\, \boldsymbol{c}_i, g)}{P(\boldsymbol{sc} > q_t - g \,|\, \boldsymbol{c}_i, g)} \tag{19}$$

$$= \frac{P(\sum_j^{j\neq i} \boldsymbol{s}_j\boldsymbol{c}_j > q_t - g - \boldsymbol{s}_i\boldsymbol{c}_i \,|\, \boldsymbol{s}_i, \boldsymbol{c}_i, g) P(\boldsymbol{s}_i)}{P(\boldsymbol{sc} > q_t - g \,|\, \boldsymbol{c}_i, g)} \tag{20}$$

The improved estimates for the coefficients of $\boldsymbol{s}$ can in turn be used to get an estimate $\boldsymbol{s}_{est}$ that minimizes its variance $E[(\boldsymbol{s}_{est} - \boldsymbol{s})^2]$ as follows:

$$0 = \frac{d}{d\boldsymbol{s}_{est,i}} E((\boldsymbol{s}_{est,i} - \boldsymbol{s}_i)^2) \tag{21}$$

$$= 2 \sum_{\boldsymbol{s}_i} (\boldsymbol{s}_{est,i} - \boldsymbol{s}_i) P(\boldsymbol{s}_i) \tag{22}$$

$$\text{or:} \quad \boldsymbol{s}_{est,i} = \sum_{\boldsymbol{s}_i} \boldsymbol{s}_i \cdot P(\boldsymbol{s}_i) \tag{23}$$

The estimate of $\boldsymbol{s}$ gives the estimate of the j^{th} column of $\boldsymbol{S}$, which can be divided trivially in an approximation $\boldsymbol{S}_{A,est}$ of $(\boldsymbol{S}_A)_{:j}$ and $\boldsymbol{E}_{A,est}$ of $(\boldsymbol{E}_A + \boldsymbol{U}_A)_{:j}$. These vectors can be used to transform the original (Ring/Module-)LWE/LWR sample $(\boldsymbol{A}, \boldsymbol{A}(\boldsymbol{S}_A)_{:j} + (\boldsymbol{E}_A + \boldsymbol{U}_A)_{:j})$ into a (Ring/Module-)LWE alike problem with a smaller secret variance by subtracting $\boldsymbol{A}\boldsymbol{S}_{A,est} + \boldsymbol{E}_{A,est}$. This results in the sample $(\boldsymbol{A}, \boldsymbol{A}((\boldsymbol{S}_A)_{:j} - \boldsymbol{S}_{A,est}) + (\boldsymbol{E}_A + \boldsymbol{U}_A)_{:j} - \boldsymbol{E}_{A,est})$, which is a problem with smaller secret $(\boldsymbol{S}_A)_{:j} - \boldsymbol{S}_{A,est}$ and noise $(\boldsymbol{E}_A + \boldsymbol{U}_A)_{:j} - \boldsymbol{E}_{A,est}$. We will call this new problem the simplified problem.

4.2 Multiple fail vectors

Having access to m positive fail vectors $\boldsymbol{c}^{(1)} \dots \boldsymbol{c}^{(m)}$ from the same column, with corresponding values of G as $g^{(1)} \dots g^{(m)}$, an adversary can improve his estimate of $P(\boldsymbol{s})$ and therefore obtain a better estimate $\boldsymbol{s}_{est}$, assuming that the failure vectors $\boldsymbol{c}_i$ are independent conditioned on $\boldsymbol{s}$. This corresponds to knowing m failing ciphertexts and the location and sign of their errors.

$$P(\boldsymbol{s}_i \,|\, \boldsymbol{c}^{(1)} \dots \boldsymbol{c}^{(m)}, g^{(1)} \dots g^{(m)}) \approx P(\boldsymbol{s}_i \,|\, \boldsymbol{c}_i^{(1)} \dots \boldsymbol{c}_i^{(m)}, g^{(1)} \dots g^{(m)}) \tag{24}$$

$$= \frac{P(\boldsymbol{c}_i^{(1)} \dots \boldsymbol{c}_i^{(m)} \,|\, \boldsymbol{s}_i, g^{(1)} \dots g^{(m)}) P(\boldsymbol{s}_i \,|\, g^{(1)} \dots g^{(m)})}{P(\boldsymbol{c}_i^{(1)} \dots \boldsymbol{c}_i^{(m)} \,|\, g^{(1)} \dots g^{(m)})} \tag{25}$$

$$= \frac{P(\boldsymbol{s}_i) \prod_{k=1}^{m} P(\boldsymbol{c}_i^{(k)} \,|\, \boldsymbol{s}_i, g^{(k)})}{\prod_{k=1}^{m} P(\boldsymbol{c}_i^{(k)} \,|\, g^{(k)})} \tag{26}$$

Similar to Equation 20, $P(\boldsymbol{c}_i \,|\, \boldsymbol{s}_i, g^{(k)})$ can be calculated as:

$$P(\boldsymbol{c}_i \,|\, \boldsymbol{s}_i, g, \boldsymbol{sc} > q_t - g) = \frac{P(\boldsymbol{sc} > q_t - g \,|\, \boldsymbol{s}_i, \boldsymbol{c}_i, g) P(\boldsymbol{c}_i \,|\, \boldsymbol{s}_i, g)}{P(\boldsymbol{sc} > q_t - g \,|\, \boldsymbol{s}_i, g)} \tag{27}$$

$$= \frac{P(\sum_{j}^{j \neq i} \boldsymbol{s}_j \boldsymbol{c}_j > q_t - g - \boldsymbol{s}_i \boldsymbol{c}_i \,|\, \boldsymbol{s}_i, \boldsymbol{c}_i, g) P(\boldsymbol{c}_i)}{P(\boldsymbol{sc} > q_t - g \,|\, \boldsymbol{s}_i, g)} \tag{28}$$

In subsequent calculations, each value of the coefficient of g is taken as the maximum possible value.

4.3 Classification of vectors

The above approach assumes a prior knowledge of the exact position and sign of the errors. This information is needed to link coefficients of $\boldsymbol{C}$ with their corresponding coefficient of $\boldsymbol{S}$. However, this is not always a trivial problem. For most schemes there are three sources of extra information that will allow to perform this classification with a high probability using only a few decryption failures.

Firstly, a large coefficient of $\boldsymbol{G}$ would induce a higher failure probability for the corresponding coefficient of the error term $\boldsymbol{S}^T\boldsymbol{C} + \boldsymbol{G}$. Thus, failures are more likely to happen at positions linked to that coefficient of $\boldsymbol{G}$. Moreover, a positive value of the coefficient suggests a positive error so that $\boldsymbol{c} \in v_{fp}$, while a negative value hints at a negative error, or $\boldsymbol{c} \in v_{fn}$.

Secondly, as $|\boldsymbol{sc} + g| > q_t$ for $\boldsymbol{c} \in v_f$, the vectors $\boldsymbol{c}$ and $\boldsymbol{s}$ are correlated. Therefore, vectors $\boldsymbol{c} \in v_f$ they are also correlated with each other. Moreover, a high positive correlation suggests that the vectors share the same class v_{fp} or v_{fn}, while a high negative correlation indicates that the vectors have a different classes. This allows for a clustering of the fail vectors using the higher than

average correlation, under the condition that the correlation difference is high enough. This correlation difference is related to the failure rate: a low failure rate implies a higher correlation because only ciphertexts that are highly correlated with the secret lead to a failure rate in this case. For example, Figure 5 shows an estimate of the correlations between vectors of the classes v_{fp} (pos), v_{fn} (neg) and v_s (nofail) in Kyber768. This approach does not work for schemes with strong error correcting codes (ECC) such as LAC, as the bit error rate before correction is relatively high for these types of algorithms, leading to a relatively low correlation between failure vectors.

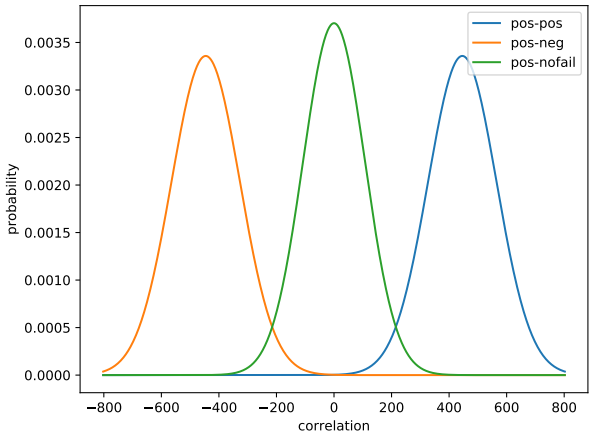

Figure 5: The probability of a certain value of the correlation between different classes of vectors in Kyber768.

In case of a ring/module structure of the coefficients of $\boldsymbol{S}$, an additional structure arises leading to an artifact in which some pairs of no-fail coefficients within the same polynomial also have high correlation of their corresponding vectors. Imagine a pair of failure coefficients at positions (i, j, k_1) and (i, j, k_2) from different decryption failures a, b, with corresponding matrices $\boldsymbol{C}^{(a)}$ and $\boldsymbol{C}^{(b)}$. The correlation of the vectors $\boldsymbol{c}^{(a)}$ and $\boldsymbol{c}^{(b)}$ can be written as $X^{k_1}\boldsymbol{C}_{:,j}^{(a)T} X^{k_2}\boldsymbol{C}_{:,j}^{(b)} = X^{k_1+k_2}\boldsymbol{C}_{:,j}^{(a)T}\boldsymbol{C}_{:,j}^{(b)}$, from which is clear that the vectors from $\boldsymbol{C}^{(a)}$ and $\boldsymbol{C}^{(b)}$, with respective positions $(i, j, k_1 - t)$ and $(i, j, k_2 + t)$ have the same correlation. The clustering will thus result in n classes, with one class containing the failure vectors. Combining this information with the information of the first method gives an adversary the failure vectors with high probability. Otherwise, an adversary can estimate the secret n times and check the validity of the result using the (Ring/Module-)LWE/LWR problem.

Finally, for schemes that use error correcting codes to reduce their failure probability, side channel leakage during the error correction might reveal information on the presence or position of failure coefficients. Note that if this is the

case, it might not even be necessary to obtain a decryption failure since failing coefficients could also be collected on successful decryptions where there is at least one failing coefficient.

4.4 Implications

Figure 6 depicts the relative variance reduction of the secret as a function of the number of positive failure vectors for various schemes. For schemes that have a very low failure probability for individual coefficients of $\boldsymbol{S}^T\boldsymbol{C} + \boldsymbol{G}$, such as Kyber, Saber and FrodoKEM, the variance of the secret drastically reduces upon knowing only a few failing ciphertexts. Assuming that the simplified problem, that takes into account the estimate of the secret, has the same complexity as a regular (Ring/Module-)LWE problem with similar secret variance, we can calculate the remaining hardness of the simplified problem as a function of the number of positive failure vectors as shown in Figure 7 using the toolbox provided by Albrecht et al. [3] using the Q-core sieve estimate.

The effectiveness of the attack declines as the failure probability of the individual coefficients increases, since the correlation between the secret and the failing ciphertext is lower in this case. This can be seen in the case of LAC, where the individual coefficients have relatively high failure rates due to a strong error correcting code. On the other hand, a failing ciphertext will contain multiple errors, making it easier to collect multiple failure vectors.

Note that once one or more failures are found, they can be used to estimate the secret. This estimate in turn can be used to improve the search for weak ciphertexts by considering $F(\boldsymbol{C},\boldsymbol{G})$ as $\sum_{\boldsymbol{S}} P(\text{FAIL}(\boldsymbol{C},\boldsymbol{G}),\boldsymbol{S})$, where $\boldsymbol{S}$ is not sampled from $\chi_{\boldsymbol{S}}$, but from the new probability distribution $\chi_{\boldsymbol{S}_{est}}$. Therefore, the search for ciphertexts that trigger a decryption failure could become easier the more failures have been found. However, we do not take this effect into account in this paper.

5 Decryption failure attack

Using the failure boosting technique from Section 3 and the secret estimation method from Section 4, we can lower the security of a (Ring/Module-)LWE/LWR scheme on the condition that its failure rate is high enough. To this end, we first collect i decryption failures using the failure boosting technique, which would cost approximately $i\sqrt{\alpha^{-1}\beta^{-1}}$ work. Then, the exact error position and failure type should be determined for all of the failure vectors using the techniques of Subsection 4.3. Based on this information, the secret can be estimated, which in turn can be used to simplify the (Ring/Module-)LWE/LWR problem. These last two operations require a negligible amount of work compared to finding decryption failures. Finally, we need to solve the simplified problem, with has a complexity $S_{\text{simplified}}(i)$ as estimated in Section 4. The total amount of work is therefore $\mathcal{O}(S_{\text{simplified}}(i)+i\sqrt{\alpha^{-1}\beta^{-1}})$, which is depicted in Figure 8 as a function of the number of failures i. Note that the practical security of Kyber relies on

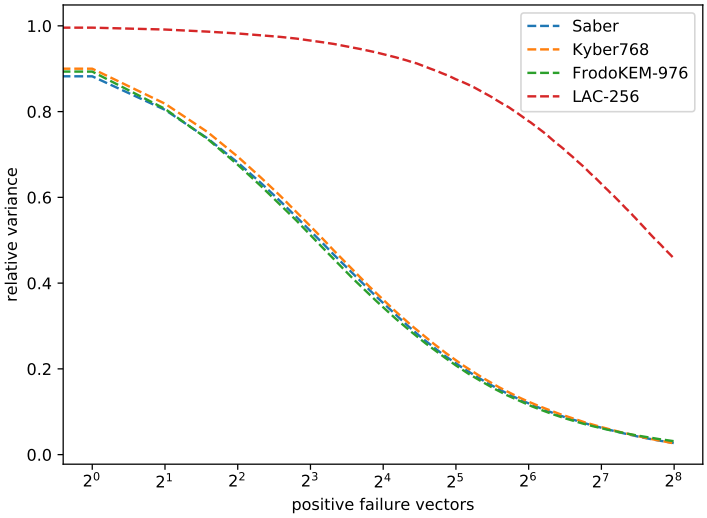

Figure 6: The relative reduction in entropy as a function of the number of positive failure vectors

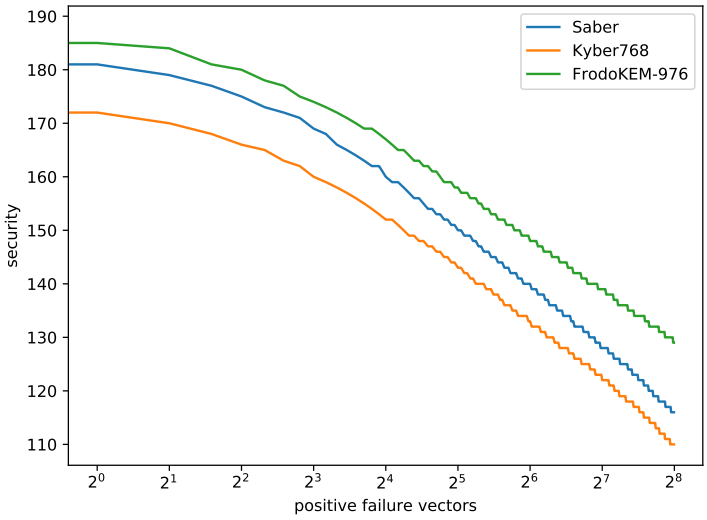

Figure 7: The hardness of the simplified problem as a function of the number of positive failure vectors

an error term $\boldsymbol{E}_A$ as well as a rounding term $\boldsymbol{U}_A$. Both are taken into account in the security calculation.

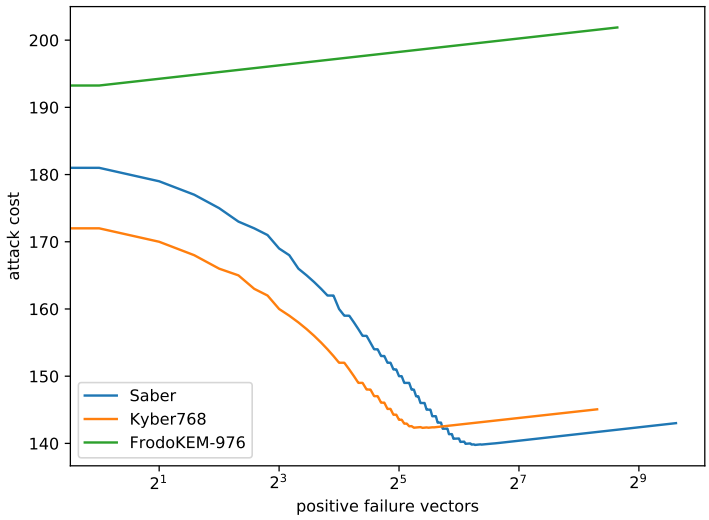

Figure 8: The full amount of work to break the scheme as a function of the number of collected decryption failures

Table 1 gives an overview of the original hardness of the scheme before decryption failure usage S, and the attack cost $S_{\text{simplified}}(i) + i\sqrt{\alpha^{-1}}\beta^{-1}$ using decryption failures for ideal values of i and f_t, which are calculated through a brute force sweep. The number of collected decryption failures i and the expected number of decryption queries $i\beta^{-1}$ is also included. These values are calculated assuming that the adversary can perform an unlimited number of decryption queries. From this table we can see that the security of Kyber and Saber is considerably reduced. This is due to the fact that finding a failure is easier than breaking the security of the scheme S. For the case of FrodoKEM976, the security is not affected as the work to obtain a failure is considerably larger than breaking the security S.

In other situations such as a multi-target attack or having only a limited number of decryption queries, other values of f_t and i will obtain optimal results. For example in a multi-target attack scenario one would select a higher threshold f_t to be able to efficiently re-use the precomputation work α^{-1} for weak ciphertexts and therefore reduce the overall work. A limit on the number of decryptions n_d could make it necessary to increase the amount of precomputational work α^{-1} in order to reduce the failure rate $\beta^{-1} < n_d/i$. This would make the attack more expensive or might even invalidate it. For example, the NIST Post-Quantum Standardization Process decryption limit is set to 2^{64}, which rules

out a decryption failure attack on schemes with a low enough failure rate such as Saber and Kyber, which can be deduced from Figure 3. As such, the security of this schemes is not affected within the NIST framework.

	original security	attack cost	reduction factor	decryption failures	decryption queries
Saber	2^{184}	2^{139}	2^{45}	77	2^{131}
FireSaber	2^{257}	2^{170}	2^{87}	233	2^{161}
Kyber768	2^{175}	2^{142}	2^{33}	42	2^{131}
Kyber1024	2^{239}	2^{169}	2^{70}	159	2^{158}
LAC256	2^{293}	$2^{97\dagger}$	2^{196}	$106 \cdot 56$	2^{80}
FrodoKEM976	2^{188}	2^{188}	2^{0}	0	0

† Note that it seems not straightforward for LAC256 to obtain the exact position and type of the errors, which is required to obtain this result

Table 1: The security of different schemes with and without decryption failures

6 Conclusion

In this paper we investigated failure boosting, a method to increase the decryption failure rate of a scheme based on the search for 'weak' ciphertexts. This method benefits an adversary in at least three scenarios: if he has access to a quantum computer, if he can only perform a limited number of decryption queries or if he wants to stage a multi-target attack on schemes that do not have the appropriate protection. We explicitly calculated the effect of failure boosting in these scenarios for various (Ring/Module-)LWE/LWR schemes. We also proposed a method to estimate the secret key given ciphertexts that lead to decryption failures. The remaining security after a certain number of decryption failures was calculated, given the exact location of the error. We suggested three methods to obtain the exact location of errors in failing ciphertexts. Finally, we estimated the security of several schemes under an attack that optimally uses these decryption failures and show that for some schemes the security could be reduced drastically if an attacker could perform sufficient decryption queries. However, for most NIST post-quantum standardization candidates, the expected number of required decryption queries is too high for a practical attack. We also identify the changes to this attack under a multi-target scenario or when an attacker has only access to a limited number of decryption queries.

7 Acknowledgements

The authors would like to thank Andreas Hülsing for interesting discussions. This work was supported in part by the Research Council KU Leuven: C16/15/058,

by the European Commission through the Horizon 2020 research and innovation programme Cathedral ERC Advanced Grant 695305 and by the Research Council KU Leuven grants C14/18/067 and STG/17/019.

References

1. Submission requirements and evaluation criteria for the post-quantum cryptography standardization process, 2016. https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/call-for-proposals-final-dec-2016.pdf.
2. M. Albrecht, R. Player, and S. Scott. On the concrete hardness of learning with errors. Journal of Mathematical Cryptology, 10 2015.
3. M. R. Albrecht, B. R. Curtis, A. Deo, A. Davidson, R. Player, E. W. Postlethwaite, F. Virdia, and T. Wunderer. Estimate all the LWE, NTRU schemes! Cryptology ePrint Archive, Report 2018/331, 2018. https://eprint.iacr.org/2018/331.
4. E. Alkim, L. Ducas, T. Pöppelmann, and P. Schwabe. Post-quantum key exchange – a new hope. In *USENIX Security 2016*, 2016.
5. H. Baan, S. Bhattacharya, O. Garcia-Morchon, R. Rietman, L. Tolhuizen, J.-L. Torre-Arce, and Z. Zhang. Round2: Kem and pke based on glwr. Cryptology ePrint Archive, Report 2017/1183, 2017. https://eprint.iacr.org/2017/1183.
6. A. Banerjee, C. Peikert, and A. Rosen. *Pseudorandom Functions and Lattices*, pages 719–737. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.
7. J. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J. M. Schanck, P. Schwabe, and D. Stehlé. Crystals – kyber: a cca-secure module-lattice-based kem. Cryptology ePrint Archive, Report 2017/634, 2017. http://eprint.iacr.org/2017/634.
8. J. H. Cheon, D. Kim, J. Lee, and Y. Song. Lizard: Cut off the tail! practical post-quantum public-key encryption from lwe and lwr. Cryptology ePrint Archive, Report 2016/1126, 2016. http://eprint.iacr.org/2016/1126.
9. J. 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*, pages 282–305, 2018.
10. S. Fluhrer. Cryptanalysis of ring-lwe based key exchange with key share reuse. Cryptology ePrint Archive, Report 2016/085, 2016. https://eprint.iacr.org/2016/085.
11. E. Fujisaki and T. Okamoto. Secure integration of asymmetric and symmetric encryption schemes. *Journal of Cryptology*, 26(1):80–101, Jan 2013.
12. N. Gama and P. Q. Nguyen. New chosen-ciphertext attacks on ntru. In T. Okamoto and X. Wang, editors, *Public Key Cryptography – PKC 2007*, pages 89–106, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.
13. L. K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing*, STOC '96, pages 212–219, New York, NY, USA, 1996. ACM.
14. D. Hofheinz, K. Hövelmanns, and E. Kiltz. A modular analysis of the fujisaki-okamoto transformation. Cryptology ePrint Archive, Report 2017/604, 2017. http://eprint.iacr.org/2017/604.
15. N. Howgrave-Graham, P. Q. Nguyen, D. Pointcheval, J. Proos, J. H. Silverman, A. Singer, and W. Whyte. The impact of decryption failures on the security of ntru encryption. In D. Boneh, editor, *Advances in Cryptology - CRYPTO 2003*, pages 226–246, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.

16. É. Jaulmes and A. Joux. A chosen-ciphertext attack against ntru. In M. Bellare, editor, *Advances in Cryptology — CRYPTO 2000*, pages 20–35, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg.
17. H. Jiang, Z. Zhang, L. Chen, H. Wang, and Z. Ma. Post-quantum ind-cca-secure kem without additional hash. Cryptology ePrint Archive, Report 2017/1096, 2017. `https://eprint.iacr.org/2017/1096`.
18. A. Langlois and D. Stehlé. Worst-case to average-case reductions for module lattices. *Designs, Codes and Cryptography*, 75(3):565–599, Jun 2015.
19. X. Lu, Y. Liu, D. Jia, H. Xue, J. He, and Z. Zhang. Lac. Technical report, National Institute of Standards and Technology, 2017. available at `https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions`.
20. V. Lyubashevsky, C. Peikert, and O. Regev. *On Ideal Lattices and Learning with Errors over Rings*, pages 1–23. Springer Berlin Heidelberg, 2010.
21. M. Naehrig, E. Alkim, J. Bos, L. Ducas, K. Easterbrook, B. LaMacchia, P. Longa, I. Mironov, V. Nikolaenko, C. Peikert, A. Raghunathan, and D. Stebila. Frodokem. Technical report, National Institute of Standards and Technology, 2017. available at `https://frodokem.org/files/FrodoKEM-specification-20171130.pdf`.
22. C. Peikert. Public-key cryptosystems from the worst-case shortest vector problem: Extended abstract. In *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing*, STOC '09, pages 333–342, New York, NY, USA, 2009. ACM.
23. O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In *STOC '05*, pages 84–93. ACM, 2005.
24. T. Saito, K. Xagawa, and T. Yamakawa. Tightly-secure key-encapsulation mechanism in the quantum random oracle model. Cryptology ePrint Archive, Report 2017/1005, 2017. `https://eprint.iacr.org/2017/1005`.
25. P. Schwabe, R. Avanzi, J. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J. M. Schanck, G. Seiler, and D. Stehle. Crystals-kyber. Technical report, National Institute of Standards and Technology, 2017. available at `https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions`.
26. P. Schwabe, R. Avanzi, J. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J. M. Schanck, G. Seiler, and D. Stehle. Newhope. Technical report, National Institute of Standards and Technology, 2017. available at `https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions`.
27. M. Seo, J. H. Park, D. H. Lee, S. Kim, and S.-J. Lee. Emblem and r.emblem. Technical report, National Institute of Standards and Technology, 2017. available at `https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions`.
28. N. P. Smart, M. R. Albrecht, Y. Lindell, E. Orsini, V. Osheter, K. Paterson, and G. Peer. Lima. Technical report, National Institute of Standards and Technology, 2017. available at `https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions`.
29. A. Szepieniec. Ramstake. Technical report, National Institute of Standards and Technology, 2017. available at `https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions`.
30. E. E. Targhi and D. Unruh. *Post-Quantum Security of the Fujisaki-Okamoto and OAEP Transforms*, pages 192–216. Springer Berlin Heidelberg, 2016.