Partial Key Exposure Attacks on BIKE, Rainbow and NTRU

Andre Esser[1], Alexander May[2], Javier Verbel[1], and Weiqiang Wen[3]

[1] Cryptography Research Center, Technology Innovation Institute, UAE
[2] Ruhr University Bochum, Germany
[3] LTCI, Telecom Paris, Institut Polytechnique de Paris, France

{andre.esser, javier.verbel}@tii.ae
alex.may@rub.de, weiqiang.wen@telecom-paris.fr

Abstract. In a so-called partial key exposure attack one obtains some information about the secret key, e.g. via some side-channel leakage. This information might be a certain fraction of the secret key bits (erasure model) or some erroneous version of the secret key (error model). The goal is to recover the secret key from the leaked information.
There is a common belief that, as opposed to e.g. the RSA cryptosystem, most post-quantum cryptosystems are usually resistant against partial key exposure attacks. We strongly question this belief by constructing partial key exposure attacks on code-based, multivariate, and lattice-based schemes (BIKE, Rainbow and NTRU). Our attacks exploit the redundancy that modern PQ cryptosystems inherently use for efficiency reasons. The application and development of techniques from information set decoding plays a crucial role for achieving our results.
On the theoretical side, we show non-trivial information leakage bounds that allow for a polynomial time key recovery attack. As an example, for all schemes the knowledge of a constant fraction of the secret key bits suffices to reconstruct the full key in polynomial time.
Even if we no longer insist on polynomial time attacks, most of our attacks extend well and remain feasible up to large erasure and error rates. In the case of BIKE for example we obtain attack complexities around 60 bits when half of the secret key bits are erased, or a quarter of the secret key bits are faulty.
Our results show that even highly error-prone key leakage of modern PQ cryptosystems may lead to full secret key recoveries.

Keywords: Erasure/Error Model, Asymptotics, Cold Boot Key Recovery.

1 Introduction

Ideally, cryptographic schemes should enjoy robustness against key leakage in the following informal sense. If a scheme uses n-bit keys and $k < n$ bits of information are leaked, then the scheme should offer the security of $(n-k)$-bit keys. The LWE problem is known to provide this *leakage robustness* property [14]. Therefore, it

is widely believed that modern post-quantum schemes provide a good level of resistance against computing the secret key from partial information, i.e., against so-called *partial key exposure attacks*.

Notice that leakage robustness of a cryptographic problem does not imply any resistance against side-channel attacks, see e.g. [12]. It just tells us that the leakage of a certain amount of information does not completely destroy the problem's hardness. Moreover, a cryptographic scheme that relies on a leakage robust problem might not automatically inherit its leakage robustness, since the scheme may introduce weaknesses, e.g. via additional secret key redundancy.

As a main result, our work shows that a certain amount of side-channel information (wherever it may come from) suffices to fully recover the secret key in polynomial time – or from a more practical perspective in reasonable time, say 2^{60} operations – for the post-quantum cryptosystems BIKE, Rainbow, NTRU. We chose these schemes as representative candidates for code-, multivariate- and lattice-based cryptography. NTRU, respectively BIKE, is a NIST 3rd round finalist, respectively alternate, encryption scheme, and Rainbow is a finalist signature scheme.

Partial Key Exposure Attacks on RSA in the Erasure and Error Model. It is well-known that RSA does not enjoy leakage robustness. For instance, by a famous result of Coppersmith [9] an RSA modulus $N = pq$ can be factored in polynomial time given only half of the bits of p. Similar results have been shown for fully recovering the secret key d [7], the RSA plaintext m [9], and also RSA CRT-exponents [6] from partial knowledge in polynomial time.

All of these attacks are in what we call the *erasure model*, i.e., we obtain a certain fraction of the bits, and have to recover the remaining bits. Moreover, the above mentioned attacks usually require that the known as well as the unknown bits are in *consecutive positions*. An erasure model with *known bits in random positions* was first addressed by Heninger and Shacham [17].

The erasure model is from a theoretical perspective convenient, because it usually provides a good starting point for reasoning about algorithms that recover complete secret keys from incomplete information. In practice however, side-channel analysis often gives us full keys with some faults that stem from the side-channel's noisiness. This is what we call the *error model*, which might be further refined by its error type.

In the error model Henecka, May, Meurer [16] showed that an RSA secret key in which bits are randomly flipped with a certain error rate $r \ll \frac{1}{2}$ can be recovered in polynomial time. This was further refined by Paterson, Polychroniadou, Sibborn [23] introducing asymmetric bit-flip rates, where bits may flip from 1 to ground state 0 with large error rate p_1, whereas the inverse direction has only small probability p_0. Such an *asymmetric error* accurately models e.g. the side-channel obtained by *Cold Boot Attacks* from Halderman et al. [15].

Previous Work on PQC Partial Key Exposure Attacks. For post-quantum cryptography, in 2017 a first partial key exposure on NTRU was proposed by Paterson and Villanueva-Polanco [24], followed by two more attacks from Villanueva-

Polanco on BLISS [28] and LUOV[29]. The PhD thesis of Villanueva-Polanco [25] contains a more systematic study of partial key exposure attacks also for Rainbow and McEliece.

However, the results achieved by Paterson and Villanueva-Polanco seem to support the strong belief in *leakage robustness* of post-quantum cryptosystems. For example considering NTRU, even in the asymmetric error setting their attacks experimentally cannot reach with good success probability rather small error probabilities $p_1 = 0.1$ and $p_0 = 0.001$. Similar for Rainbow, the attack in [25] only reaches error probabilities of $p_1 = 0.001$ and $p_0 = 0.001$ even for toy parameters of Rainbow.

These results stand in quite sharp contrast to our attacks, that handle relatively large erasure/error rates, sometimes even in polynomial time. We see two mains reason for our substantial improvement over previous work.

First, we heavily use the key redundancy provided by many modern post-quantum systems to recover secret keys much more efficiently. Key redundancy has already been exploited by Albrecht, Deo, Paterson [1] by using the NTT representation of Kyber keys. However, [1] still only tolerates comparatively small error probabilities around 1%.

Second, we use and further develop more advanced techniques from information set decoding and lattices, thereby building on the work of Horlemann, Puchinger, Renner, Schamberger, Wachter-Zeh [18] for decoding with hints and the work of Dachman-Soled, Ducas, Gong, Rossi [10] for lattices with hints. Especially, our new decoding techniques might be of independent interest.

Our results with Polynomial Time Key Recovery. To the best of our knowledge, our work is the first that achieves polynomial time partial key exposure attacks on post-quantum cryptosystems for non-trivial erasure/error rates. Our results are summarized in Table 1.

Polynomial Attacks	key format	erasure	error
BIKE	standard	0.500	$\frac{\log n}{\sqrt{n}}$
	compact	0.092	$\frac{1}{\log^2 n}$
Rainbow		$\frac{n}{n+1}$	$\frac{\log n}{n}$
NTRU	(un)packed	$n^{-\frac{2}{3}}$	$\frac{\sqrt{\log n}}{n}$
	"consecutive"	0.250	

Table 1: Summary of (asymptotic) bounds on erasure/error probability for polynomial-time key recovery.

Let us briefly discuss our results, what we mean by the different key formats, and how we exploit key redundancies.

BIKE For BIKE keys we achieve erasure rates of $p = \frac{1}{2}$ (standard) and $p = 0.092$ (compact). Let $(\mathbf{s}, \mathbf{e}) \in \mathbb{F}_2^{n/2} \times \mathbb{F}_2^{n/2}$ be a BIKE secret key in standard format, i.e., $(\mathbf{s}, \mathbf{e})$ is given as a bit-vector. Then half of the secret key bits suffice to recover the full secret key in polynomial time.

This might be surprising at first glance, but a BIKE secret key fulfills an LWE/LPN-relation $A\mathbf{s} = \mathbf{e}$. From this alone one can see that $\mathbf{s}$ suffices to efficiently recover $\mathbf{e}$, provided that both vectors are unusually sparse. Our attack now simply shows that any $n/2$ bits suffice. Thus, in comparison to LWE/LPN, BIKE secret keys are redundant, since they store the secret $\mathbf{s}$ and the error $\mathbf{e}$. Both parts are required in BIKE for proper decryption (as opposed to LWE-type schemes).

If $(\mathbf{s}, \mathbf{e})$ is stored compactly, i.e., only the few non-zero positions of both vectors are encoded, then one can only recover the key efficiently given a 0.092-fraction of its bits.

In the error setting, our BIKE results tolerate roughly an $\frac{1}{\sqrt{n}}$-error in the standard case, i.e. roughly $\sqrt{n}$ error positions. However, notice that somewhat surprisingly the compact case allows for a huge number of $n/\log^2 n$ error positions.

We would like to point out that all of our BIKE results also hold identically for HQC [21] secret keys.

Rainbow For Rainbow keys, our partial key exposure attacks are even stronger. This stems from our main observation that a single row of Rainbow's secret key matrix suffices to fully recover the whole matrix. Therefore, a Rainbow secret key inherently already contains a (quadratic) redundancy factor, which is reflected by our results.

In the erasure setting, we can tolerate rates that convergence to 1. In other words, a linear fraction of the secret key is already enough for full key recovery.

However, for error recovery our results are significantly worse. Here, we can only tolerate $\log n$ error positions per key.

NTRU Eventually, for the NTRU cryptosystem we achieve for errors/erasures in random positions the smallest tolerable rates. The NTRU encryption scheme can be considered as a Ring-LWE instance. As explained for BIKE, the consequence is that NTRU secret keys provide less redundancy than BIKE secret key, since NTRU does not store the LWE error. Thus, it is not surprising that we obtain weaker bounds for NTRU.

Yet, if we look into the setting of erasures in consecutive positions of an NTRU secret key, then we can recover the full secret key from only a 3/4-fraction of all positions. This consecutive partial key exposure attack heavily exploits NTRU's ring structure. We are not aware of any cryptanalytic results in the literature that exploit the ring structure with such a significant gain.

Concrete Parameters: The Erasure Case. While from a theoretical perspective it is desirable to understand for which erasure/error rates we can achieve poly-

nomial time attacks, it might be even more interesting to see how our attacks perform on concrete parameter sets. Table 2 shows the results of our attacks on Category-1 parameter sets (equivalent to AES-128 security), when we allow for attack bit complexities of 45, 60 and 80 bit.[4]

Erasures	key format	bit complexity bound		
		45	60	80
BIKE	standard	0.570	0.650	0.730
	compact	0.410	0.425	0.445
Rainbow		0.710	0.810	0.890
NTRU	unpacked	0.219	0.300	0.422
	packed	0.065	0.092	0.138

Table 2: Tolerable erasure rates for different key formats that allow for a key recovery with bit complexity less then 45, 60 and 80 on the Category-1 parameter sets.

Recall from Table 1 that polynomial time attacks on BIKE are feasible for erasure rates of 0.5 (standard key format), respectively roughly 0.1 (compact key format). We observe from Table 2 that the allowed errors in the standard format are significantly larger, namely in the interval $[0.57, 0.73]$, whereas in compact format we even improve to rates beyond 0.4. Hence, BIKE's redundancy leads to efficient key recoveries in practice given roughly half of the secret key bits.

Rainbow with its large key redundancy should tolerate larger rates than BIKE. This is also reflected in the Category-1 erasure rates in Table 2 that allow for key recovery for rates in the interval $[0.71, 0.89]$.

Asymptotically, NTRU has least key redundancy and therefore tolerates the smallest erasure rates. This is in line with the results of Table 2, where we can recover erasure rates in the interval $[0.22, 0.42]$. However, compared to previous results from Paterson and Villanueva-Polanco [24], our results still improve by orders of magnitude. The reason for this improvement comes from using more involved lattice techniques than the simple key enumeration from [24]. Even for the least redundant *packed* key format, we still obtain erasure rates in the interval $[0.06, 0.14]$.

Concrete Parameters: The Error Case. Analogous to Table 2, we denote in Table 3 the error rates that we can recover for Category-1 parameter sets when using bit-complexities of 45, 60, and 80 bits.

Noteworthy, we achieve high error rates in the *asymmetric* error setting. For Rainbow, this again stems from the larger key redundancy. However, for BIKE and NTRU we exploit the partial key information to significantly lower the dimension of the underlying problems. This in turn speeds up our decoding/lattice

[4] The code to rerun our experiments and bitcomplexity calculations is available at https://github.com/Crypto-TII/partial-key-exposure-attacks

Errors	key format	bit complexity					
		45		60		80	
		asym.	sym.	asym.	sym.	asym.	sym.
BIKE	standard	0.050	0.050	0.150	0.120	0.300	0.200
	compact	0.175	0.030	0.240	0.060	0.275	0.080
Rainbow		0.240	0.120	0.375	0.190	0.540	0.270
NTRU	unpacked	0.033	0.002	0.099	0.009	0.273	0.019
	packed	0.009	0.003	0.020	0.008	0.040	0.015

Table 3: Tolerable error rates for key recovery of Category-1 parameter sets with bit complexities bounded by 45, 60 and 80. In the asymmetric setting we fix $p_0 = 0.001$.

reduction algorithms. Most impressive is maybe the error correction of unpacked NTRU keys with error rate $p_1 = 0.27$ within a complexity of 80 bit. Observe that our attack is not yet very effective for 45 bits (with error $p_1 = 0.03$), since it requires a large polynomial overhead.

Conclusion. As opposed to the common belief, current post-quantum schemes allow for effective partial key exposure attacks, both in the erasure and the error setting. We demonstrate this for representative post-quantum candidates from codes, multivariate and lattices. As a rule of thumb, the higher the key redundancy, the more effective are our attacks. But even the least redundant NTRU scheme still allows for quite impressive erasure/error rates.

Organization of the paper. We elaborate on our erasure/error model in Section 2, where we also recap the basics of information set decoding attacks. All our BIKE results, asymptotically and for concrete parameters, are given in Section 3. Our Rainbow results can be found in Section 4, and we conclude with the NTRU results in Section 5.

2 Preliminaries

We denote vectors and matrices as bold lower case and bold capital letters. For a matrix $\mathbf{M} \in \mathbb{F}_q^{k \times n}$ and a set $I \subseteq \{1, \ldots, n\}$ we let $\mathbf{M}_I$ be the matrix obtained by projecting $\mathbf{M}$ onto the columns defined by I. We use the same notation for vectors. We refer by $\mathbf{O}$ to the all-zero matrix. All logarithms are base 2 if not stated otherwise. We use standard Landau notation for complexity statements and call a probability p high if $p = 1 - o(1)$.

2.1 Key Exposure Models

Throughout this work we consider two different key exposure models – the *error* and the *erasure* model. In the error model one obtains knowledge of a faulty

version of the full secret key, whereas in the erasure setting only certain parts of the secret key are known but guaranteed to be correct.

In our theoretical treatment we work with errors and erasures on the field level, while for practical considerations we work with errors and erasures on the bit level. This distinction allows us to keep the theoretical consideration (mostly) independent of the chosen key-representation. In our practical analysis we then analyze the performance of our attacks using specific key formats. Let us more formally define both models, starting with the field level.

Errors and erasures in fields For all schemes that we consider the private keys are either vectors (or matrices) over $\mathbb{F}_q$ or polynomials with coefficients over $\mathbb{F}_q$, which are represented via their coefficient vector. Field erasures then correspond to a set of indices for which the corresponding coordinates or coefficients are unknown, while the rest is known.

Definition 2.1 (Erasures). *Let $n \in \mathbb{N}$, $\mathbf{f} = (f_1, \ldots, f_n) \in \mathbb{F}_q^n$ and $I \subseteq \{1, \ldots, n\}$ denote the erasure positions. For $\mathbf{u}_i$ denoting the i-th unit vector we let*

$$\tilde{\mathbf{f}} := \sum_{i \notin I} f_i \mathbf{u}_i + \sum_{i \in I} y_i \mathbf{u}_i,$$

with $y_i \in \mathbb{F}_q$. In the erasure model I and $\tilde{\mathbf{f}}$ are known, while the y_i are unknown. *Each coordinate of $\mathbf{f}$ gets erased with probability $p := \Pr[i \in I]$, which we denote as* $\mathbb{F}_q$-erasure probability. *We call $\tilde{\mathbf{f}}$ a* partially erased *version of $\mathbf{f}$.*

In a concrete attack scenario of Definition 2.1 $\mathbf{f}$ would be the secret key, and the goal is to recover $\mathbf{f}$ from the partially erased version $\tilde{\mathbf{f}}$. We call such an attack a *(key recovery) attack in the erasure model.*

Let us analogously define the error model.

Definition 2.2 (Errors). *Let $n \in \mathbb{N}$ and $\mathbf{f} = (f_1, \ldots, f_n) \in \mathbb{F}_q^n$. Further let $\mathbf{e} = (e_1, \ldots, e_n) \in \mathbb{F}_q^n$ be an error vector. In the error model one is given*

$$\tilde{\mathbf{f}} := (\tilde{f}_1, \ldots, \tilde{f}_n) = \mathbf{f} + \mathbf{e} = (f_1 + e_1, \ldots, f_n + e_n),$$

while the f_i and e_i are unknown. Every coordinate of the error is different from zero with probability $p := \Pr[e_i \neq 0]$, which we refer to as $\mathbb{F}_q$-error probability. *If $e_i \neq 0$, then e_i is uniformly distributed in $\mathbb{F}_q \backslash \{0\}$, i.e., $Pr[e_i = 0] = (1 - p)$ and $\Pr[e_i = k] = \frac{p}{q-1}$ for all $k \in \mathbb{F}_q \backslash \{0\}$. We call $\tilde{\mathbf{f}}$ an* erroneous *version of $\mathbf{f}$.*

Again, in a partial key exposure scenario $\mathbf{f}$ would be the secret key and $\tilde{\mathbf{f}}$ the known erroneous version of it. The goal is to recover $\mathbf{f}$ from $\tilde{\mathbf{f}}$. We call such an attack a *(key recovery) attack in the error model.*

Errors and erasures of bits For our practical analysis we switch from field to bit errors, i.e., every bit in the binary representation of the secret key is flipped (error model) or erased (erasure model) with a certain probability. One obtains a definition for the erasure and error model considering bit-errors / -erasure by letting $\mathbf{f}$ of Definition 2.1 and 2.2 be the binary representation of the secret key and correspondingly setting $q = 2$. In these cases we might use the term *bit-error* and *bit-erasure* rather than $\mathbb{F}_2$-error and $\mathbb{F}_2$-erasure. Once it is known how the secret key is represented on the bit-level, one can relate the field- and bit-error/-erasure.

Speaking of the error model, in a practical scenario, which is often motivated via cold-boot attacks, we usually find asymmetric error probabilities, i.e., the probability p_0 of a zero flipping to one is different from the probability p_1 of a one becoming a zero. Let the binary representation of the secret key be $\mathbf{f} \in \mathbb{F}_2^n$ and the error be $\mathbf{e} \in \mathbb{F}_2^n$. An attack in the error model then asks to recover $\mathbf{f}$ from $\tilde{\mathbf{f}} := \mathbf{f} + \mathbf{e}$. Therefore

$$p_0 := \Pr\left[e_i = 1 \mid f_i = 0\right] \text{ and } p_1 := \Pr\left[e_i = 1 \mid f_i = 1\right].$$

Usually one of the probabilities is rather small, since bits are more likely to flip to the ground state of the respective memory region to which all bits decay over time. Following the initial work of Halderman et al. [15] we assume in our analysis that all bits decay to the same ground state for simplification. Moreover we consider the ground state to be zero and adopt the choice of $p_0 = 10^{-3}$, experimentally observed in [15].

2.2 Decoding

Some of our attacks make use of information set decoding (ISD) algorithms to decode linear codes. A linear code $\mathcal{C}$ is a k dimensional subspace of $\mathbb{F}_q^n$. We call n the *code length* and k the *code dimension*. Such a code can be represented via the kernel of a matrix $\mathbf{H} \in \mathbb{F}^{(n-k)\times n}$ of rank $n-k$, i.e., $\mathcal{C} := \{\mathbf{c} \in \mathbb{F}_q^n \mid \mathbf{H}\mathbf{c}^\top = \mathbf{0}\}$. Hence, recovering a codeword $\mathbf{c} \in \mathcal{C}$ from a given faulty string $\mathbf{y} = \mathbf{c} + \mathbf{f}$ is equivalent to recover $\mathbf{f}$ from the *syndrome* $\mathbf{s} := \mathbf{H}\mathbf{y} = \mathbf{H}(\mathbf{c} + \mathbf{f}) = \mathbf{H}\mathbf{f}$.

Definition 2.3 (Syndrome Decoding Problem). *Let $n, k \in \mathbb{N}$. Given the parity check matrix $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}$ of a linear code over $\mathbb{F}_q$, a* syndrome $\mathbf{s} \in \mathbb{F}_q^{n-k}$ *and an* error-weight $\delta \in \mathbb{N}$*, the syndrome decoding problem asks to find a vector $\mathbf{f} \in \mathbb{F}_q^n$ of weight $\mathrm{wt}(\mathbf{f}) = \delta$ satisfying $\mathbf{H}\mathbf{f} = \mathbf{s}$. We call $(\mathbf{H}, \mathbf{s}, \delta)$ an* instance *of the syndrome decoding problem and $\mathbf{f}$ the* solution.

The class of ISD algorithms, initiated by Prange in 1962 [26], allows to solve the syndrome decoding problem.

Prange's Information Set Decoding (ISD) For a permutation matrix $\mathbf{P} \in \mathbb{F}_2^{n\times n}$, $\mathbf{P}^{-1}\mathbf{f}$ forms a solution to the permuted syndrome decoding instance $(\mathbf{H}\mathbf{P}, \mathbf{s}, \delta)$. We let the permuted error be $\mathbf{P}^{-1}\mathbf{f} =: (\mathbf{f}_1, \mathbf{f}_2) \in \mathbb{F}_q^{n-k} \times \mathbb{F}_q^k$ and

the permuted parity check matrix $\mathbf{HP} := (\mathbf{H}_1 \mid \mathbf{H}_2)$ with $\mathbf{H}_1 \in \mathbb{F}_q^{(n-k)\times(n-k)}$. From $\mathbf{Hf} = \mathbf{s}$ it follows that

$$(\mathbf{H}_1)^{-1}(\mathbf{HP})(\mathbf{P}^{-1}\mathbf{f}) = (\mathbf{I}_{n-k} \mid \mathbf{H}_2')(\mathbf{f}_1, \mathbf{f}_2) = \mathbf{f}_1 + \mathbf{H}_2'\mathbf{f}_2 = (\mathbf{H}_1)^{-1}\mathbf{s} =: \mathbf{s}',$$

where $\mathbf{H}_2' = (\mathbf{H}_1)^{-1}\mathbf{H}_2$. Let us further assume that the permutation distributes the weight on $(\mathbf{f}_1, \mathbf{f}_2)$ such that $\mathrm{wt}(\mathbf{f}_2) = \gamma$ and, hence, $\mathrm{wt}(\mathbf{f}_1) = \delta - \gamma$. Then finding a solution corresponds to finding an $\mathbf{f}_2$ of weight γ for which the corresponding $\mathbf{f}_1 = \mathbf{s}' - \mathbf{H}_2'\mathbf{f}_2$ has weight $\delta - \gamma$. In the following we call any selection of $n-k$ columns (defined by the permutation) that leads to an $\mathbf{f}_1$ of weight $\delta - \gamma$ an *information set*. Further for $\gamma = 0$ an information set contains the whole error. Prange's algorithm now chooses a random permutation matrix, computes $\mathbf{H}_2'$ and $\mathbf{s}'$ and then enumerates all possible $\mathbf{f}_2$ of weight γ until it finds an $\mathbf{f}_1$ of weight $\delta - \gamma$. This process is repeated with fresh initial permutations until success. The expected complexity of Prange's algorithm (up to polynomial factors) is

$$T_{\text{Prange}} = \underbrace{\frac{\binom{n}{\delta}}{\binom{n-k}{\delta-\gamma}\binom{k}{\gamma}}}_{\text{Permutations}} \underbrace{\binom{k}{\gamma}(q-1)^\gamma}_{\text{Enumeration}} = \frac{(q-1)^\gamma \cdot \binom{n}{\delta}}{\binom{n-k}{\delta-\gamma}} \quad . \tag{1}$$

Obviously, $\gamma = 0$ minimizes the running time. However, for large δ, namely for $\delta > n - k$, choosing γ equal to zero is not possible.

To derive our asymptotic bounds we show the following theorem, specifying a small error regime for which Prange's algorithm has polynomial complexity.

Theorem 2.1. *Let* $\mathbf{H}$ *be the parity-check matrix of a code with length* n *and co-dimension* $n - k = c \cdot n$*, for constant* c*. Then a syndrome decoding instance* $(\mathbf{H}, \mathbf{s}, \delta)$ *with* $\delta = \mathcal{O}(\log n)$ *can be solved in polynomial time.*

Proof. The complexity of Prange's algorithm to solve such an instance is (up to polynomial factors) given by Eq. (1) as

$$\frac{\binom{n}{\delta}}{\binom{n-k}{\delta}} = \frac{\binom{n}{\delta}}{\binom{cn}{\delta}} = \prod_{i=0}^{\delta-1} \frac{n-i}{cn-i} = \prod_{i=0}^{\delta-1} \frac{1}{c - \frac{(c-1)i}{n-i}} = \left(\frac{1}{c - o(1)}\right)^\delta = n^{\mathcal{O}(1)}. \quad \square$$

3 BIKE

For BIKE the private key is the solution to a syndrome decoding instance $(\mathbf{H}, \mathbf{s}, \delta)$ over $\mathbb{F}_2$, of code length n and dimension k, defined via the public key $\mathbf{H}$. Hence, the secret key is a vector $\mathbf{f} \in \mathbb{F}_2^n$. The scheme uses a code rate of $R = \frac{1}{2}$, i.e., $n = 2k$ and $\delta := \mathrm{wt}(\mathbf{f}) = \Theta(\sqrt{n}) = \Theta(\sqrt{k})$.

We consider two different key formats in our analysis. The *standard key format* stores the secret key as bitstring of length n, while the *compact key format* stores only the δ one entries of $\mathbf{f}$.[5] In a nutshell our attacks on both

[5] Both formats can be found in the NIST submission[4] or the implementation accompanying it.

formats follow a similar strategy: Initially we generate likely candidates for the one positions in the secret key based on the given key material. Then to obtain polynomial attacks we upper-bound the total number of candidates dependent on the error/erasure probability. In our practical attacks we consider constant error/erasure probabilities, which lead to a large set of candidates. Our attacks then speed up Prange's ISD procedure by prioritizing the set of candidates.

3.1 Standard Format Keys

We say the secret key is stored in *standard* format or using *standard* representation when it stores the secret value $\mathbf{f} \in \mathbb{F}_2^n$ as a sequence of n bits.

The Erasure Model First let us investigate the erasure model. Since the parity-check matrix $H \in \mathbb{F}_2^{n/2 \times n}$ defines $n/2$ linear equations in $\mathbf{f}$, we can directly recover $\mathbf{f}$ whenever the number of unknowns is at most $n/2$.

Theorem 3.1 (Polynomial Erasure Attack on Standard Format). *Let $\tilde{\mathbf{f}}$ be a given partially erased BIKE secret key in the standard format with $\mathbb{F}_2$-erasure probability $p \leqslant \frac{1}{2}$. Then, the secret key $\mathbf{f} \in \mathbb{F}_2^n$ can be recovered in polynomial time with success probability at least $\frac{1}{2}$.*

Proof. Let X by a random variable for the number of erased coordinates of $\mathbf{f}$. Since X is binomially distributed with parameters n and $p \leqslant \frac{1}{2}$, we have

$$\Pr[X \leqslant n/2] = \sum_{i=0}^{n/2} \binom{n}{i} p^i (1-p)^{n-i} \geqslant 2^{-n} \sum_{i=0}^{n/2} \binom{n}{i} \geqslant \frac{1}{2}.$$

Thus, with probability at least $\frac{1}{2}$ we have at most $n/2$ unknowns. In this case, matrix H with rank $n/2$ provides $n/2$ linearly independent equations in at most $X \leqslant n/2$ unknowns. Therefore, we recover $\mathbf{f}$ via Gaussian elimination in time $\mathcal{O}(n^3)$. □

The Error Model Next let us consider the error model for the standard key format. According to Section 2.1 this corresponds to a given syndrome decoding instance $(\mathbf{H}, \mathbf{s}, \delta)$ together with a faulty solution vector $\tilde{\mathbf{f}} = \mathbf{f} + \mathbf{e}$, where $\Pr[e_i = 1] = p$. Note that we can again easily transform such an instance into a different syndrome decoding instance with weight $\delta' = \mathrm{wt}(\mathbf{e})$. Therefore we just let the new syndrome be $\mathbf{s}' := \mathbf{s} + \mathbf{H}\tilde{\mathbf{f}}$, since

$$\mathbf{s}' := \mathbf{s} + \mathbf{H}\tilde{\mathbf{f}} = \mathbf{H}\mathbf{f} + \mathbf{H}(\mathbf{f} + \mathbf{e}) = \mathbf{H}\mathbf{e}.$$

This gives a first improvement, whenever $\delta' < \delta$. However, intuitively as long as $p \neq \frac{1}{2}$ (which corresponds to a uniform error) the problem should become easier.

To obtain a speedup whenever $p < \frac{1}{2}$, we exploit that the distribution of the weight on the candidate $\tilde{\mathbf{f}}$ in this case carries information about the weight distribution of $\mathbf{f}$. Observe that the expected weight of $\tilde{\mathbf{f}}$ is

$$\delta_{\tilde{\mathbf{f}}} := \mathbb{E}[\mathrm{wt}(\tilde{\mathbf{f}})] = \delta(1-p) + p(n-\delta),$$

where the first addend counts the ones of $\mathbf{f}$ contributing to $\delta_{\tilde{\mathbf{f}}}$, while the second counts the contribution from $\mathbf{e}$. Let γ_1 be the random variable counting the weight of $\mathbf{f}$ restricted to the coordinates where $\tilde{f}_i = 1$, then it follows that $\mathbb{E}[\gamma_1] = \delta(1-p)$. Analogously, the rest of the weight, namely $\gamma_0 := \delta - \gamma_1$, must then distribute over the coordinates where $\tilde{f}_i = 0$.

In the following we adapt the choice of columns selected by Prange's algorithm for finding an information set based on the given key material. Put simple, if a higher fraction of the error-weight is located on the coordinates of $\tilde{\mathbf{f}}$ with $\tilde{f}_i = 1$ than on those with $\tilde{f}_i = 0$, overall more coordinates are taken from the one-associated coordinates and vice versa. More precisely, we introduce a parameter $\rho_1 \leqslant \delta_{\tilde{\mathbf{f}}}$ determining how many of the $n-k$ columns, belonging to the information set, are chosen from the block defined by the one-coordinates of $\tilde{\mathbf{f}}$. Consequently, the remaining $\rho_0 := n-k-\rho_1$ are taken from the block defined by the zeros of $\tilde{\mathbf{f}}$. The probability that $n-k$ columns selected this way form an information set that contains the whole error becomes

$$q := \frac{\binom{\delta_{\tilde{\mathbf{f}}}-\gamma_1}{\rho_1-\gamma_1}}{\binom{\delta_{\tilde{\mathbf{f}}}}{\rho_1}} \cdot \frac{\binom{n-\delta_{\tilde{\mathbf{f}}}-\gamma_0}{\rho_0-\gamma_0}}{\binom{n-\delta_{\tilde{\mathbf{f}}}}{\rho_0}} = \frac{\binom{\rho_1}{\gamma_1}\binom{\rho_0}{\gamma_0}}{\binom{\delta_{\tilde{\mathbf{f}}}}{\gamma_1}\binom{n-\delta_{\tilde{\mathbf{f}}}}{\gamma_0}}. \tag{2}$$

This implies an expected amount of

$$\mathbb{E}[q^{-1}] = \frac{\binom{\rho_1}{(1-p)\delta}\binom{n-k-\rho_1}{p\delta}}{\binom{\delta_{\tilde{\mathbf{f}}}}{(1-p)\delta}\binom{n-\delta_{\tilde{\mathbf{f}}}}{p\delta}} \tag{3}$$

sets to be selected until we find such an information set.

The following theorem states up to which error probability the expected attack complexity, i.e., Eq. (3), stays in the polynomial time regime, using Theorem 2.1.

Theorem 3.2 (Polynomial Error Attack on Standard Format). *Let $\tilde{\mathbf{f}}$ be a given erroneous BIKE secret key in the standard format with $\mathbb{F}_2$-error probability $p = \mathcal{O}\left(\frac{\log n}{\sqrt{n}}\right)$. Then the secret key $\mathbf{f} \in \mathbb{F}_2^n$ can be recovered in polynomial time.*

Proof. Given in Appendix A.1.

Note that the above attack can easily be adapted to asymmetric error probabilities. In this case the expected weight of $\tilde{\mathbf{f}}$ changes to $\delta_{\tilde{\mathbf{f}}} = \delta(1-p_1)+p_0(n-\delta)$. Again the first addend counts the number of ones contributed from $\mathbf{f}$, which lets the fraction of ones in $\tilde{\mathbf{f}}$ also present in $\mathbf{f}$ become $\gamma_1 := \delta(1-p_1)$. Now the adaptation of the ISD algorithm works as before, where Eq. (2) yields the probability of success in each iteration.

3.2 Compact Format Keys

If the secret key is stored via δ integers encoding its one positions, we say the key is stored in *compact* format or representation. Since BIKE's secret key is balanced, i.e., it has weight $\frac{\delta}{2}$ on each half of the coordinates, every integer encoding a position requires $\log k$ bits. We find that the compact compared to standard format allows for improved asymptotic bounds in the error model, while giving a slight disadvantage in the erasure model.

To allow for a direct comparison between the error- and erasure-probabilities for both key formats we stay with bit-errors and -erasures, i.e., we treat the secret key in compact format as a sequence of $\delta \log k$ bits, where any bit might be erroneous or get erased, rather than as δ integers.

The Erasure Model Our strategy for key recovery in the erasure model is again to generate candidates for the one coordinates of the key, based on the given information, and include those candidates in the information set. As long as the amount of candidates stays smaller than the co-dimension, which is $n-k=k$ in the case of BIKE, we can recover the secret key in polynomial time via Gaussian elimination. We get a slightly worse bound than for the standard format because ε bit-erasures in any integer lead to 2^{ε} candidates, i.e., we have an exponential amplification in the amount of candidates.

Theorem 3.3 (Polynomial Erasure Attack on Compact Format). *Let $\tilde{\mathbf{f}}$ be a given partially erased BIKE secret key in the* compact *format with $\mathbb{F}_2$-erasure probability $p \leqslant 0.092$. Then the secret key $\mathbf{f} \in \mathbb{F}_2^{\delta \log k}$ can be recovered in polynomial time with constant success probability.*

Proof. Given in Appendix A.2.

In practice we find that we can handle even higher erasure rates of roughly $p \approx 0.4$ (see Section 3.3), which is because in our analysis we neglect some effects that lie in our favor, as e.g., the possibility of colliding candidates and the gain of ε_i being below its expectation. However, overall this corresponds to a constant factor disadvantage to the standard key format.

The error model Interestingly, if we turn our focus to the error-model, the compact key representation allows for an asymptotic increase in the error probability while staying in the polynomial time regime. This is because the compact representation allows to derive more candidates than the standard format and, hence, exploit the full size of the information set.[6] Therefore, we again generate candidates for each index of the secret key based on the given erroneous indices.

[6] Recall, that for the standard format the most likely candidates are given by the one coordinates of the erroneous secret key, which are only $\delta_{\tilde{\mathbf{f}}} = o(n)$.

Theorem 3.4 (Polynomial Error Attack on Compact Format). *Let $\tilde{\mathbf{f}}$ be a given erroneous BIKE secret key with $\mathbb{F}_2$-error probability $p = \mathcal{O}\left(\frac{1}{\log^{2+\kappa} k}\right)$ for a small constant $\kappa > 0$. Then the secret key $\mathbf{f} \in \mathbb{F}_2^{\delta \log k}$ can be recovered in polynomial time with high success probability.*

Proof. We start similar to the erasure-setting by bounding the number of bit-errors per index to a certain value. So that the total number of candidates remains less than k resulting in a polynomial-time key recovery. Note that if we have no more than $\varepsilon \leqslant \frac{\log k}{2}$ bit-errors per index we find a total amount of candidates of

$$N := \delta \sum_{i=1}^{\varepsilon} \binom{\log k}{i} \leqslant c_\delta \varepsilon \sqrt{k} \binom{\log k}{\varepsilon} \leqslant c_\delta \varepsilon \sqrt{k} \log^{\varepsilon} k,$$

where $\delta \leqslant c_\delta \sqrt{k}$ for some constant c_δ. Consequently, we find $N < k$ if $\varepsilon \leqslant \frac{\log k}{2 \log \log k}(1 - o(1))$. Now, let us consider an error rate of $p = \frac{1}{\log^{2+\kappa} k}$ for some constant $\kappa > 0$. This yields an expected amount of

$$\mathcal{E} := p \cdot \delta \cdot \log k = \mathcal{O}(p \cdot \sqrt{k} \cdot \log k) = \mathcal{O}\left(\frac{\sqrt{k}}{\log^{1+\kappa} k}\right)$$

errors in total. A Chernoff bound gives that we have $\Theta(\mathcal{E})$ errors with high probability. Now to bound the number of bit-errors per index we treat the problem as a balls into bins problem, i.e., we consider the indices as $\sqrt{k}$ bins and throw $\Theta(\mathcal{E})$ balls uniformly at random into those. Note that in contrast to the standard balls-into-bins problem we have a slight bias in the balls' distribution, due to the maximum capacity of $\log k$ per bin. In our setting balls are placed uniformly over all possible free places in those bins, i.e., as soon as a few balls got placed in a bin it becomes less likely that more balls are placed in that same bin. However, since we are interested in bounding the maximum number of errors per bin, this bias lies in our favor. Therefore treating the balls as independent just overestimates the maximum balls per bin, which still gives a valid upper bound for our setting. Note that when throwing m balls into d bins, where $\frac{d}{\text{polylog}(d)} \leqslant m << d \log d$ a result by Raab and Steger [27, Theorem 1] states that with high probability there is no bin with an occupancy higher than

$$\frac{\log d}{\log \frac{d \log d}{m}} \left(1 + \alpha \frac{\log \log \frac{d \log d}{m}}{\log \frac{d \log d}{m}}\right),$$

for any $\alpha > 1$, which gives an upper bound for ε. From here it follows for our choice of $d = \delta$ and $m = \Theta(\mathcal{E})$ that we have $\varepsilon < \frac{\log k}{2 \log \log k}$ as long as κ is a small constant, as desired. We give the necessary technical calculations in Appendix A.3. Eventually, a union bound gives that both events – in total an amount of $\Theta(\mathcal{E})$ errors and small enough ε – happen still with high probability. □

Summarizing, the compact key representation of BIKE allows for a polynomial time key recovery in the error setting up to an error rate of $p = \frac{1}{\log^{2+\kappa} k}$ for a small $\kappa > 0$ in comparison to the less compact variant which only allows for an error-rate of $\frac{\log k}{\sqrt{k}}$.

3.3 Practical Attacks on BIKE

The erasure model Our practical approach for key recovery in the standard format extends the idea from Section 3.1 in the following way. Let I be the set of erased coordinates and $\bar{I} := \{1, \dots, n\} \backslash I$, i.e., all coordinates of $\mathbf{f}_I$ are unknown, while those of $\mathbf{f}_{\bar{I}}$ are known. Now from $\mathbf{Hf} = \mathbf{s}$ it follows that

$$\mathbf{H}_I \mathbf{f}_I = \mathbf{s} + \mathbf{H}_{\bar{I}} \mathbf{f}_{\bar{I}},$$

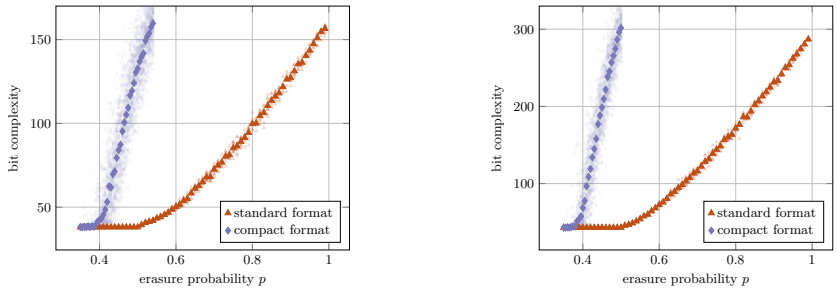
where only $\mathbf{f}_I$ is unknown. Recovering $\mathbf{f}_I$ now corresponds to solving a syndrome decoding instance with code length $n' := |I|$, unchanged co-dimension $n - k$ and error weight $\delta' := |\{i \in I \mid f_i = 1\}|$, which is the number of missing one coordinates of the secret key. Note that for $n' \leqslant n - k$ we can solve for $\mathbf{f}_I$ via Gaussian Elimination again.

For the compact format we also slightly extend our previous approach from Section 3.2. Recall, that the secret key is represented as a vector of integers. We first check if there are erasure-free indices. Any of those decrease the searched error weight and the code length by one. Next, we generate a set of candidates for the remaining one positions of the secret key, i.e., whenever we find ε erasures in an index it contributes with 2^ε candidates. Any coordinate not identified as one in the first step and not appearing among the candidates must be zero, which shortens the code further. Finally this gives a syndrome decoding instance with code length $\tilde{n}$, which is the amount of distinct candidates, unchanged co-dimension $n - k$ and error weight $\tilde{\delta} := \delta - \beta$, where β denotes the number of erasure-free indices identified in the first step.

For our simulation we generate a BIKE secret key in the respective format and simulate the bit-erasures. Next we derive the parameters of the reduced syndrome decoding instances and use the *Syndrome Decoding Estimator* tool by Esser and Bellini [13] to obtain the bit complexity of Stern's algorithm to solve the resulting instance.

In Fig. 1 we illustrate the averaged complexity (solid marks) of several experiments (transparent marks) as a function of the bit-erasure rate. Coherent to our theoretical analysis we find that attacks on the standard format are more efficient in the erasure model than attacks on the compact format. However, we also observe that for practical parameters in the compact format the point where the reduced code length $\tilde{n}$ exceeds the co-dimension (which marks the transition to the non-polynomial regime) is $p \approx 0.38$, rather than the theoretically proven $p = 0.092$.

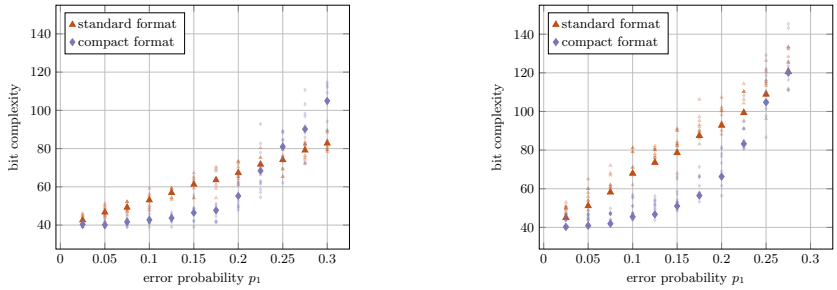
The error model In our practical consideration we consider asymmetric error probabilities, again let those be p_0 and p_1 as defined in Section 2.1.



(a) Category 1: $(k, \delta) = (12323, 142)$ (b) Category 5: $(k, \delta) = (40973, 274)$

Fig. 1: Bit complexity of partial key exposure attack in the *erasure* model on the compact and standard key representations based on experiments.

For the compact format, contrary to what the proof of Theorem 3.4 suggests, we do not take the error-weight as criterion to derive a list of candidate positions, rather we use a maximum-likelihood approach. Therefore let $\tilde{I}$ with $|\tilde{I}| = \delta$ be the given set of erroneous (integer) indices. Then for every $i \in \tilde{I}$ we compute a set S_i containing those $x \in \{1, \ldots, n\}$ for which the probability $p_x := \Pr\left[x \in I \mid i \in \tilde{I}\right]$ is maximized, where I is the set of coordinates representing the true secret key. More precisely, we define a threshold τ and only include those x with $p_x > \tau$. We then compute the union of the S_i as $S = \bigcup_i S_i$. Intuitively, S contains the most likely candidates for true secret key indices. Now, we are in a similar setting to the attack on the standard format, where the set S corresponds to a set of coordinates containing more weight than $\{1, \ldots, n\} \backslash S$.



(a) Category 1: $(k, \delta) = (12323, 142)$ (b) Category 5:$(k, \delta) = (40973, 274)$

Fig. 2: Bit complexity of partial key exposure attacks on the compact and standard key representations based on experiments for $p_0 = 10^{-3}$.

We performed experiments for fixed p_0 and increasing p_1 to estimate the complexity of our attacks on both key formats. We first generate BIKE secret keys in the respective formats and then simulate the error to obtain an erroneous key. For the standard case we now calculate the complexity of an attack on the

generated instance by using Eq. (2). For the compact case we proceed similarly, by first choosing a value for the threshold τ calculating the set S and finally calculating the complexity of the resulting attack via Eq. (2). Then we minimize over the choice of τ.

Fig. 2 illustrates the obtained bit complexities for the Category-1 and -5 parameter sets of BIKE. Each data point is averaged (solid marks) over ten experiments (transparent marks). Coherent to our analysis attacks on the compact key format perform exceptionally well for "small" error rates $p_1 \leqslant 0.225$. The attack on the compact format also scales well when increasing the parameters of the scheme, as shown in Fig. 2, where the break-even point is shifted to $p_1 \approx 0.3$.

We also applied the experiment for symmetric error probabilities. In this case the amount of candidates in the compact format increases drastically, which leads to an overall inferior attack on the compact format, as also reflected in Table 3.

4 Rainbow

Here, we describe the two *layers* Rainbow with parameters (q, v, o_1, o_2). Let $n := v + o_1 + o_2$, $\mathbf{x} := (x_1, \ldots, x_n)$ be a vector of unknowns, and $\mathbb{F}_q[\mathbf{x}]$ be the ring of polynomials with coefficients in $\mathbb{F}_q$.

A *Rainbow central map* is a quadratic map $\mathcal{F} = (f_1, \ldots, f_{o_1+o_2}) \in (\mathbb{F}_q[\mathbf{x}])^{o_1+o_2}$, where the polynomials in $(f_1, \ldots, f_{o_1})$ (resp. $(f_{o_1+1}, \ldots, f_{o_1+o_2})$) are of the form $\sum_{i=1}^{v}\sum_{j=1}^{v+o_1} a_{i,j}x_ix_j$ (resp. $\sum_{i=1}^{v+o_1}\sum_{j=1}^{n} a_{i,j}x_ix_j$). The sequence $(f_1, \ldots, f_{o_1})$ (resp. $(f_{o_1+1}, \ldots, f_{o_1+o_2})$) is called the *first layer* (resp. *second layer*) of $\mathcal{F}$.

Public and secret keys: The public key is a sequence of quadratic polynomials over $\mathbb{F}_q[\mathbf{x}]$ given by $(p_1, \ldots, p_{o_1+o_2}) = \mathcal{S} \circ \mathcal{F} \circ \mathcal{T}$, where $\mathcal{S} : \mathbb{F}_q^{o_1+o_2} \to \mathbb{F}_q^{o_1+o_2}$, $\mathcal{T} : \mathbb{F}_q^n \to \mathbb{F}_q^n$ are linear maps. The secret key is given by $\mathcal{S}^{-1}, \mathcal{T}^{-1}$, and $\mathcal{F}$.

In order to reduce the size of the private key, Ding et al. [11] proposed to use $\mathcal{S}$ and $\mathcal{T}$ such that for all $\mathbf{y} \in \mathbb{F}_q^{o_1+o_2}$ and all $\mathbf{x} \in \mathbb{F}_q^n$ we have

$$\mathcal{S}^{-1}(\mathbf{y}) = \underbrace{\begin{bmatrix} \mathbf{I}_{o_1} & \mathbf{S}' \\ \mathbf{O} & \mathbf{I}_{o_2} \end{bmatrix}}_{:=\mathbf{S}^{-1}} \mathbf{y} \quad \text{and} \quad \mathcal{T}^{-1}(\mathbf{x}) = \underbrace{\begin{bmatrix} \mathbf{I}_v & \mathbf{T}^{(1)} & \mathbf{T}^{(4)} \\ \mathbf{O} & \mathbf{I}_{o_1} & \mathbf{T}^{(3)} \\ \mathbf{O} & \mathbf{O} & \mathbf{I}_{o_2} \end{bmatrix}}_{:=\mathbf{T}^{-1}} \mathbf{x}, \tag{4}$$

where $\mathbf{S}, \mathbf{T}^{(3)} \in \mathbb{F}_q^{o_1 \times o_2}$, $\mathbf{T}^{(4)} \in \mathbb{F}_q^{v \times o_2}$, and $\mathbf{T}^{(1)} \in \mathbb{F}_q^{v \times o_1}$. So far none of the known attacks on Rainbow can benefit from secret matrices $\mathbf{S}$ and $\mathbf{T}$ chosen as in Eq. (4).

Remark 4.1. Suppose that the secret maps $\mathcal{S}$ and $\mathcal{T}$ are homogeneous, and they are represented by the matrices $\mathbf{S}$ and $\mathbf{T}$, respectively. Then, the polynomials of the public key and the central map $\mathcal{F}$ are related by the equation $\sum_{j=1}^{o_1+o_2} s_{i,j}p_j(\mathbf{x}) = f_i(\mathbf{Tx})$, for $i = 1, \ldots, o_1 + o_2$, where $\mathbf{S}^{-1} := [s_{i,j}]$.

We conclude the overview on Rainbow by stating the three suggested parameter sets, which are Rainbow-I ($q = 16, v = 36, o_1 = 32, o_2 = 32$), Rainbow-III ($q = 256, v = 68, o_1 = 32, o_2 = 48$) and Rainbow-V ($q = 256, v = 96, o_1 = 36, o_2 = 64$).

4.1 Attack Strategy

Our partial key exposure attacks exploit the structure of the maps $\mathcal{S}^{-1}$ and $\mathcal{T}^{-1}$ given in Eq. (4). In Proposition 4.1, we derive linear relations between some coefficients of polynomials in $\mathcal{F}$ and coordinates of the matrix $\mathbf{S}'$ (see Eq. (4)). Finally, we use the erroneous/ partially erased private key to find one complete row of $\mathbf{S}'$ by either solving an instance of the syndrome decoding problem (in the error model) or enumerating the minimum amount of information so that we obtain one row by solving a linear system (in the erasure model). We finally observe that this single row of $\mathbf{S}'$ is already sufficient for full key recovery in polynomial time.

Let us introduce the *vinegar part* of a homogeneous quadratic polynomial.

Definition 4.1 (vinegar part). *Let p be a homogeneous quadratic polynomial in $\mathbb{F}_q[x_1, \ldots, x_n]$. The* vinegar part *of p is the homogeneous quadratic polynomial $p^{\mathbf{v}} \in \mathbb{F}_q[x_1, \ldots, x_v]$ such that $p(x_1, \ldots, x_n) - p^{\mathbf{v}}(x_1, \ldots, x_v)$ contains no monomials of the form $x_i x_j$ where $1 \leqslant i, j \leqslant v$.*

We observe the following relation between the vinegar parts of the public polynomials and the polynomials in the hidden central map, which forms the basis of our attacks.

Proposition 4.1. *Let f_i be the i-th polynomial in the first layer of a Rainbow central map, that is, $i \leqslant o_1$. Let $(p_1, \ldots, p_{o_1+o_2})$ be a Rainbow public key, where the corresponding secret maps $\mathcal{S}$ and $\mathcal{T}$ are homogeneous, and their matrix representations are as shown in Equation (4). Then, we have*

$$p_i^{\mathbf{v}}(x_1, \ldots, x_v) + \sum_{j=o_1+1}^{o_1+o_2} s_{i,j} \cdot p_j^{\mathbf{v}}(x_1, \ldots, x_v) = f_i^{\mathbf{v}}(x_1, \ldots, x_v),$$

where $f_i^{\mathbf{v}}(x_1, \ldots, x_v)$ is the vinegar part of f_i, $(s_{i,1}, \ldots, s_{i,o_1+o_2})$ is the i-th row of $\mathbf{S}^{-1}$, and $p_j^{\mathbf{v}}(x_1, \ldots, x_v)$ is the vinegar part of p_j.

Proof. Given in Appendix B.1.

As a second key ingredient for our attacks, we show the following theorem, which allows us to recover the full Rainbow secret key in polynomial time from a single known row of $\mathbf{S}'$.

Theorem 4.1 (Rainbow Full Key Recovery). *Let (q, v, o_1, o_2) be a Rainbow parameter set and $\mathbf{S}'$ as defined in Eq. (4). Then, knowledge of any single row of $\mathbf{S}'$ is sufficient to recover the full Rainbow secret key in polynomial time in the input parameters q, v, o_1, o_2.*

Proof. Given in Appendix B.2.

To the best of our knowledge this was not shown before. Once $\mathbf{S}'$ is fully recovered, strategies for full key recovery in polynomial time are known. However, for completeness we outline the full key recovery procedure in Appendix B.2.

4.2 $\mathbb{F}_q$-errors and -erasures

All over this section we treat o_2 as the major security parameter by assuming that $o_1 = c_{o_1} o_2, v = c_v o_2, q = c_q o_2$, where c_{o_1}, c_v and c_q are considered constant. This allows us to state our results only as a function of o_2.

The erasure model In the erasure model we use Proposition 4.1 together with the given partially erased information to derive linear equations in the unkown coordinates of a single row of $\mathbf{S}'$. Obviously as long as there are more equations than unkowns we can recover the searched row in polynomial time.

Theorem 4.2 (Polynomial Erasure Attack). *Given a partially erased Rainbow secret key with $\mathbb{F}_q$-erasure probability $p = \frac{1}{1+o(1)}$. Then the secret key can be recovered in polynomial time with constant success probability.*

Proof. Let $\mathbf{s}_i$ denote the i-th row of $\mathbf{S}'$ from Eq. (4) and $f_i^{\mathbf{v}}$ be the vinegar part of polynomial f_i in the first layer. Denote by $I_{\mathbf{s}_i}$ the indices of erased coordinates of $\mathbf{s}_i$ and by $I_{f_i^{\mathbf{v}}}$ the indices of unknown coefficients in $f_i^{\mathbf{v}}$. Note that there exists one i with $|I_{\mathbf{s}_i}| \leqslant po_2$ and $|I_{f_i^{\mathbf{v}}}| \leqslant p\frac{v(v-1)}{2}$ with constant probability. Now, by Proposition 4.1, every coefficient of $f_i^{\mathbf{v}}$ is equal to linear combination of the entries of the $\mathbf{s}_i$. Hence, every known coefficient of $f_i^{\mathbf{v}}$ leads to a linear equation in the $|I_{\mathbf{s}_i}| \leqslant p \cdot o_2$ unknown variables of $\mathbf{s}_i$. Thus, we expect to find the remaining unknown entries of $\mathbf{s}_i$ whenever

$$p \cdot o_2 \leqslant (1-p)\frac{v(v-1)}{2} \Leftrightarrow p \leqslant \frac{1}{1+\frac{2}{c_v v}} = \frac{1}{1+o(1)}. \qquad \square$$

Note that if we take the proposed Rainbow parameters, Theorem 4.2 implies that an $\mathbb{F}_q$-erasure probability as high as 94% still leads to a polynomial attack.

Corollary 4.1 (Erasure-Results on Rainbow Parameters). *Given a partially erased Rainbow secret key, following any of the suggested parameter sets, with $\mathbb{F}_q$-erasure probability $p < 0.94$. Then the secret key can be recovered in polynomial time with constant success probability.*

Proof. The proof of Theorem 4.2 requires that $p \leqslant \frac{1}{1+\frac{2}{c_v v}}$ to ensure that we can recover the Rainbow secret key from its partially-erased version in polynomial. Our result follows since $0.94 \leqslant \frac{1}{1+\frac{2}{c_v \cdot v}}$ for all parameters given in the beginning of Section 4. $\square$

When considering the more practical bit-erasures, the tolerable rates become significantly smaller. However, we can still handle bit-erasure rates of roughly 50% (Rainbow-I) and 30% (Rainbow-III and -V).

Remark 4.2 (Relation to Bit-Erasures). Suppose that every field element is represented as a binary sequence of length $\log q$. Then, the $\mathbb{F}_q$-erasure probability p_q and the bit-erasure probability p_b are related by $p_q = 1 - (1-p_b)^{\log q}$. Hence, we expect to recover the full Rainbow-I private key in polynomial time whenever the bit-erasure probability is $p_b \leqslant 0.50$, and the full Rainbow-III and Rainbow-V private keys whenever $p_b \leqslant 0.29$.

The error model For our attack in the error model, we first show a reduction from a partial key exposure attack to the syndrome decoding problem. Then we study the complexity of solving the corresponding syndrome decoding instance. We give the reduction in the following lemma.

Lemma 4.1 (Reduction to Syndrome Decoding). *Recovering the Rainbow secret key from a given erroneous candidate with $\mathbb{F}_q$-error probability p can polynomially be reduced to solving an instance of the syndrome decoding problem $(\mathbf{H}, \mathbf{b}, \delta_p)$, where $\mathbf{H} \in \mathbb{F}_q^{r \times (r+o_2)}$ and $\delta_p := p(r+o_2)$ and $1 \leqslant r \leqslant \frac{v(v-1)}{2}$.*

Proof. We again only focus on recovering one row of the hidden matrix $\mathbf{S}'$, defined in Eq. (4) then using Theorem 4.1 to recover the full key. Without loss of generality, let us assume we want to find the first row $(s_1, \ldots, s_{o_2})$ of $\mathbf{S}'$ and let $f^{\mathbf{v}} = \sum_{1 \leqslant i \leqslant j \leqslant v} f_{i,j} \cdot x_i x_j$ be the vinegar part of the first polynomial in the hidden Rainbow central map. Further, let $(\tilde{s}_1, \ldots, \tilde{s}_{o_2}) := (s_1 + e_1, \ldots, s_{o_2} + e_{o_2})$ and $\tilde{f}^{\mathbf{v}} := f^{\mathbf{v}} + \sum_{1 \leqslant i \leqslant j \leqslant v} \epsilon_{i,j} x_i x_j$ be the given faulty versions of the first row of $\mathbf{S}'$ and $f^{\mathbf{v}}$. Recall that $e_1, \ldots, e_{o_2}, \epsilon_{1,1}, \ldots, \epsilon_{v,v} \in \mathbb{F}_q$ are unknown, but known to be zero with probability $1-p$.

Let us define $\mathbf{x}_v := (x_1, \ldots, x_v)$, and $\tilde{f}_{i,j} := f_{i,j} + \epsilon_{i,j}$ for every pair of integers $1 \leqslant i, j \leqslant v$. Now, by Proposition 4.1 we have

$$p_1^{\mathbf{v}}(\mathbf{x}_v) + \sum_{j=1}^{o_2} (\tilde{s}_j - e_j) \cdot p_{o_1+j}^{\mathbf{v}}(\mathbf{x}_v) = f_1^{\mathbf{v}}(\mathbf{x}_v)$$

$$\epsilon(\mathbf{x}_v) - \sum_{j=1}^{o_2} e_i \cdot p_{o_1+j}^{\mathbf{v}}(\mathbf{x}_v) = -p_1^{\mathbf{v}}(\mathbf{x}_v) - \sum_{j=1}^{o_2} \tilde{s}_j \cdot p_{o_1+j}^{\mathbf{v}}(\mathbf{x}_v) + \tilde{f}^{\mathbf{v}}(\mathbf{x}_v), \quad (5)$$

where $\epsilon(\mathbf{x}_v) := \sum_{1 \leqslant i,j \leqslant v} \epsilon_{i,j} \cdot x_i x_j$. Here, the polynomial on the right side of Eq. (5) is known. Hence, for every monomial $x_i x_j$ on the right-hand side, with coefficient $b_{i,j}$, we have a linear relation of the form $\epsilon_{i,j} + l_{i,j}(e_1, \ldots, e_{o_2}) = b_{i_j}$, where $l_{i,j}$ is a known linear polynomial. Therefore, for every integer $1 \leqslant r \leqslant v(v-1)/2$ and every set of r coefficients of $\epsilon(\mathbf{x}_v)$ there is a matrix $\mathbf{L} \in \mathbb{F}_q^{r \times o_2}$, whose j-th column corresponds to the coefficient-vector of the polynomial $p_j^{\mathbf{v}}$, such that

$$\underbrace{[\mathbf{I}_r | \mathbf{L}]}_{\mathbf{H}} \underbrace{(\epsilon_{i_1,j_1}, \ldots, \epsilon_{i_r,j_r}, e_1, \ldots, e_{o_2})^\top}_{\mathbf{f}} = \mathbf{b}, \quad (6)$$

where $\mathbf{b}$ is the vector containing the corresponding $b_{i,j}$.

Note that the linear system given in Eq. (6) can be seen as an instance of the syndrome decoding problem where the underlying code has rate $R := \frac{r}{r+o_2}$, and the expected weight of the solution $\mathbf{e}$ is $\delta_p := p(r+o_2)$. Finally, solving this instance allows to recover the first row of $\mathbf{S}'$. □

Next in Theorem 4.3, we describe a polynomial-time partial key exposure attack in the error model, that uses the previous reduction.

Theorem 4.3. *Given an erroneous Rainbow secret key with $\mathbb{F}_q$-error probability $p = \mathcal{O}\left(\frac{\log_2 o_2}{o_2}\right)$. Then the secret key can be recovered in polynomial time with constant probability.*

Proof. Lemma 4.1 states that recovering any row of $\mathbf{S}'$ from the given erroneous secret key is equivalent to solving an instance of the syndrome decoding problem $(\mathbf{H}, \mathbf{b}, \delta_p)$, where $\mathbf{H} \in \mathbb{F}_q^{r \times (r+o_2)}$, $\mathbf{b} \in \mathbb{F}_q^r$ and $\delta_p = p(r+o_2)$. In the following we set $r = o_2$, which is a valid choice since according to Lemma 4.1 $r \leqslant \frac{v(v-1)}{2} = \mathcal{O}(o_2^2)$. For this choice we find $\delta_p \leqslant \log(o_2)$ with constant probability.

Note that the resulting syndrome decoding instance possesses a unique solution with high probability, since the expected amount of random solutions is

$$\frac{\binom{r+o_2}{\delta_p}(q-1)^{\delta_p}}{q^{o_2}} = \frac{\binom{2o_2}{\log o_2}(q-1)^{\log o_2}}{q^{o_2}} < \frac{(2o_2)^{\log o_2}}{q^{o_2 - \log o_2}} = o(1).$$

However, the searched row of $\mathbf{S}'$ is a solution by construction. Now, by Theorem 2.1 we can find this unique solution in polynomial time. □

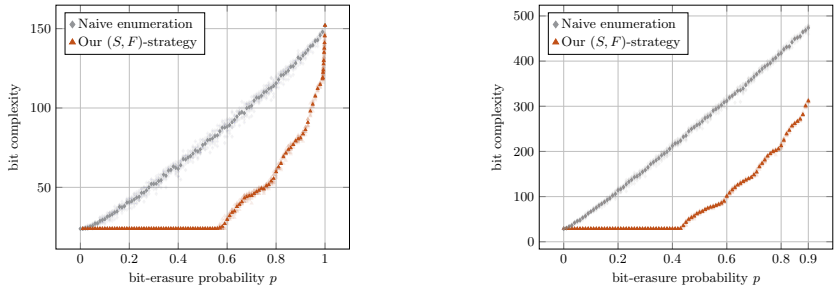
4.3 Practical Attacks on Rainbow

The erasure model Let again $\mathbf{s}_i$ denote the i-th row of $\mathbf{S}'$, and let f_i denote the i-th polynomial in the first layer of the Rainbow central map. Recall, that our key recovery strategy used in Theorem 4.2 requires to know $o_2 - k$ coordinates of the $\mathbb{F}_q$-vector representation of $f_i^{\mathbf{v}}$, where k is the number of known coordinates of the $\mathbb{F}_q$-vector representation of $\mathbf{s}_i$.

Our (S, F)-strategy. Let n_{b_i} for $i = 0, \ldots, o_1$ be the minimum amount of bits we have to enumerate of $(\mathbf{s}_i, f_i^{\mathbf{v}})$ to obtain k coordinates from $\mathbf{s}_i$ and $o_2 - k$ coefficients from $f_i^{\mathbf{v}}$. For each guess, we need to solve a linear system in the $o_2 - k$ unknowns over $\mathbb{F}_q$. Finally, we need to check if the derived solution leads to the correct Rainbow private key, which requires $(v + o_1 + o_2)^3$ field multiplications (see Lemma B.1). Therefore, the amount of field multiplications to recover one row of the secret matrix $\mathbf{S}'$ is $2^{n_b} \cdot \left(o_2^3 + (v + o_1 + o_2)^3\right)$, where $n_b = \min_i\{n_{b_i}\}$.

We compare our (S, F)-strategy to naive enumeration, which enumerates the least amount of erased bits to obtain one row of $\mathbf{S}'$. The resulting bit complexities

are shown in Fig. 3. We assume a field multiplication to cost $\log^2 q$ bit operations. For each p, we computed for ten randomly generated erased private keys the complexity of the key recovery (transparent marks), and the corresponding averaged complexity (solid marks). Note that our (S, F)-strategy outperforms naive enumeration for both parameter sets and all values of p. Also, for both parameter sets, there is range of p where no enumeration in the (S, F)-strategy is necessary, i.e., we found at least one pair $(\mathbf{s}_i, f_i^{\mathbf{v}})$ with enough erasure-free coordinates, such that $n_{b_i} = 0$. This range corresponds to the polynomial regime of Theorem 4.2 and is $0 \leqslant p < 0.57$ in the Rainbow-I case, while $0 \leqslant p < 0.43$ in the Rainbow-V case, both slightly better than estimated in Remark 4.2.



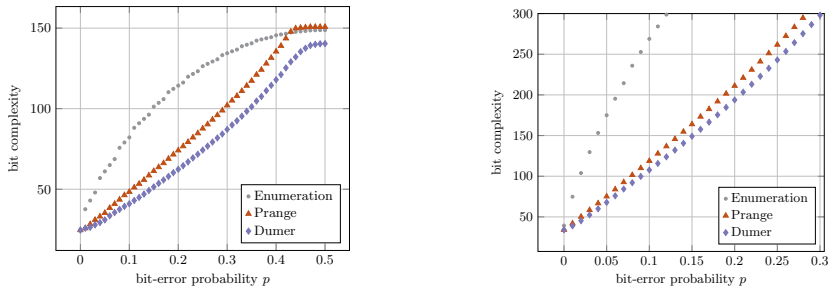
(a) Category-1 (b) Category-5

Fig. 3: Bit complexities of our partial key exposure attack and naive enumeration of a row of $\mathbf{S}'$ in the erasure model against Rainbow-I and Rainbow-V.

The error model Our attack in the error model again uses the reduction to the syndrome decoding problem given in Lemma 4.1, relating field- and bit-errors via Remark 4.2. More precisely, from a given faulty Rainbow secret key with bit-error rate p we derive a syndrome decoding instance $(\mathbf{H}, \mathbf{s}, \delta)$ where $\mathbf{H} \in \mathbb{F}_2^{r \times (r+o_2)}$, $\mathbf{s} \in \mathbb{F}_2^r$, and $\delta := \left(1 - (1-p)^{\log q}\right)(r + o_2)$ with $1 \leqslant r \leqslant \frac{v(v-1)}{2}$ for the respective value of v of the corresponding parameter set.

To solve the resulting syndrome decoding instance we then use an ISD algorithm. For deriving the concrete bit complexity we adapted the *Syndrome Decoding Estimator* by Esser and Bellini [13] to the $\mathbb{F}_q$ case. In this adaptation we assume $\log^2 q$ bit operations per field multiplication. We finally minimize over the choice of r. In contrast to our theoretical analysis the choice of r might result in an instance with multiple solutions. As only a single of these solutions leads to the Rainbow secret key we need to reapply the ISD algorithm for each solution and finally check if it leads to the correct Rainbow private key. This check requires $(o_1 + o_2 + v)^3$ field multiplications (compare to Lemma B.1). Now if there exist E solutions and the cost for finding all of them is T_{ISD}, then the total cost of our partial key exposure attack becomes $T = T_{\text{ISD}} + (o_1 + o_2 + v)^3 \cdot E$.

In Fig. 4 we plot the computed bit complexity of our attack for Rainbow-I and Rainbow-V parameters. For comparison we also give the complexity of a naive enumeration of the error on the initial row of $\mathbf{S}'$.



(a) Category-1 (b) Category-5

Fig. 4: Bit complexity of our partial key exposure attack in the error model on Rainbow.

We observe the impact of a growing value of q on the quality of the attack. The Category-1 parameter set with $q = 16$ yields improved attacks up to $p = 0.5$, while for the Category-5 (and also Category-3) parameter set with $q = 256$, we only achieve improvements up to $p \approx 0.28$ (and $p \approx 0.27$ respectively). This stems form the amplification of the field error for growing q. However, in any case our attacks yield significant speedups over basic enumeration, the previously best known attack to exploit additional information.

5 NTRU

Let n be an integer, $q > 3$ a prime number, and let the ternary field $\mathbb{F}_3$ be represented by the elements $\{-1, 0, 1\}$. We define the ring $R_q := \mathbb{F}_q[x]/\langle x^n - 1\rangle$. During this section, we identify polynomials $v = \sum_{i=0}^{n-1} v_i x^i \in R_q$ with their coefficient vector $\mathbf{v} = (v_{n-1}, v_{n-2}, \ldots, v_0)^{\mathrm{T}}$.

In NTRU, the secret key is given by a polynomial $f \in R_q \cap R_3$ that is invertible in R_q and in R_3. A public $h \in R_q$ associated with f is given by $h = f^{-1} \cdot g \bmod q$, where $g \in R_q \cap R_3$. For efficiency reasons the current NTRU NIST submission [8] stores both f and f_3^{-1} as the private key. Therefore, we refer to (f, f^{-1}) as the NTRU private key, where we drop for convenience the subscript 3 of f_3^{-1}. Note that an NTRU private key fulfills the two equations

$$f \cdot h = g \bmod q \text{ and } f \cdot f^{-1} = 1 \bmod 3.$$

5.1 $\mathbb{F}_q$-errors and -erasures

The results of this section heavily exploit the key redundancy from the second equation $f \cdot f^{-1} = 1 \bmod 3$. We first show how to recover in polynomial time

$\mathcal{O}(\sqrt{n})$ erasures in *random* positions, and second that in the case of *consecutive* positions we can even recover $n/4$ erasures.

For the error setting, an application of Theorem 2.1 shows how to correct $\mathcal{O}(\sqrt{\log n})$ errors.

The Erasure Model Let us start with a polynomial-time attack in the erasure model that exploits the second key equation, i.e., $f \cdot f^{-1} = 1 \mod 3$.

Theorem 5.1. *Given a partially erased NTRU secret key $(\tilde{f}, \tilde{f}^{-1})$ with $\mathbb{F}_q$-erasure probability $p = \frac{1}{\sqrt{2n}}$. Then the secret key $(f, f^{-1}) \in R_q \times R_3$ can be recovered in polynomial time with high success probability.*

Proof. Let $I_f, I_{f^{-1}} \subset \{1, \ldots, n\}$ denote the unknown indices in f, f^{-1}, i.e.,

$$f = \sum_{i \notin I_f} f_i x^i + \sum_{i \in I_f} y_i x^i \text{ and } f^{-1} = \sum_{i \notin I_{f^{-1}}} f_i^{-1} x^i + \sum_{i \in I_{f^{-1}}} z_i x^i.$$

Let us use the secret key identity $f \cdot f^{-1} = 1 \bmod 3$. This gives us n identities, where the k-th identity is

$$\sum_{i+j=k \bmod n} f_i \cdot f_j^{-1} = \delta_{k0}, \text{ where } \delta_{k0} = \begin{cases} 1 & k = 0 \\ 0 & \text{else} \end{cases}.$$

Let X_k be an indicator variable that takes value 1 iff the k-th identity is linear in the variables y, z. Notice that for any (i, j) with $i + j = k \bmod n$ we obtain a quadratic term $y_i z_j$ with probability $p^2 = \frac{1}{2n}$. A union bound shows that we obtain for any (i, j) a quadratic term with probability at most $np^2 = \frac{1}{2}$. Thus, $\mathbf{E}[X_k] \geqslant \frac{1}{2}$. Let $X = X_0 + \ldots, X_{n-1}$ denote the number of linear equations. Then by linearity of expectation $\mathbf{E}[X] \geqslant \frac{n}{2}$.

Any secret key coefficient is unknown with probability p. Thus, we have an expected number of $2np = \sqrt{2n}$ unknowns. An application of Markov's equality shows that the number of linear equations exceeds the number of unknowns with high probability. Therefore, we can solve the resulting system of linear equations in time polynomial in n, thereby recovering all secret key coefficients. □

Remark 5.1. Note that the expected number of linear equations $\mathbf{E}[X] \geqslant n(1 - np^2)$ drops to 0 when $p = \frac{1}{\sqrt{n}}$. This shows that our partial key exposure attack from Theorem 5.1 does not extend to larger error rates, even when we do not restrict to polynomial time.

Consecutive Erasure Attack In the following, instead of having coefficients of f and f^{-1} erased in random positions, we assume that the erasures appear in consecutive positions. Surprisingly, in this case the erasure rate for a polynomial-time attack increases significantly. While Theorem 5.1 allows on expectation $2np = \sqrt{2n}$ erasures, the following theorem can handle $n/4$ erasures.

Theorem 5.2. *Let $(\tilde{f}, \tilde{f}^{-1})$ be a partially erased NTRU secret key having $n/4$ $\mathbb{F}_q$-erasures in (cyclically) consecutive positions of both $\tilde{f}$ and $\tilde{f}^{-1}$. Then the secret key $(f, f^{-1}) \in R_q \times R_3$ can be recovered in polynomial time.*

Proof. We assume without loss of generality that the erasures are in position $0, \ldots, \frac{n}{4}-1$, i.e., we obtain the coefficients of f and f^{-1} in positions $\frac{n}{4}, \ldots, n-1$. By cyclicity, the following argument extends to all other (cyclically) consecutive positions. We have

$$f = \sum_{i=0}^{n/4-1} y_i x^i + \sum_{i=n/4}^{n-1} f_i x^i \text{ and } f^{-1} = \sum_{j=0}^{n/4-1} z_j x^j + \sum_{j=n/4}^{n-1} f_j^{-1} x^j.$$

The key identity $f \cdot f^{-1} = 1 \bmod 3$ gives n equations, where the k-th equation is

$$\sum_{i+j=k \bmod n} f_i f_j^{-1} = \delta_{k0} \text{ with } \delta_{k0} = \begin{cases} 1 & k = 0 \\ 0 & \text{else} \end{cases}.$$

Notice that we obtain quadratic terms $y_i z_j$ only if $0 \leqslant i + j \leqslant 2(\frac{n}{4} - 1)$. Thus, all equations with $2(\frac{n}{4} - 1) < k < n$ are linear. These are $n - (\frac{n}{2} - 2) > n/2$ linear equations. Thus, the number of linear equations exceeds the amount $\frac{n}{2}$ of unknowns y_i, z_j. Solving for the unknowns recovers the secret key in time polynomial in n. □

Theorem 5.2 shows that we can recover 1/4 of the secret key bits, whenever the unknowns are in consecutive positions. The following remark shows that we can recover even up to a 1/3-fraction if certain (unrealistic) conditions are met.

Remark 5.2. Assume that $3|n$, and we obtain erasures y_i, z_j in positions $i, j \in \{0, 3, 6, \ldots, n-3\}$. Then for all $k = i + j$ we have $3|k$. Thus, we obtain quadratic terms only in the k-th equation with $k = 0 \mod 3$. This in turn gives us $n - \frac{n}{3} = \frac{2}{3}n$ linear equations and also $\frac{2}{3}n$ unknowns.

The Error Model Next we give a polynomial-time attack in the error model, again exploiting the second key equation $f \cdot f^{-1} = 1 \bmod 3$.

Theorem 5.3. *Let $(\tilde{f}, \tilde{f}^{-1})$ be an erroneous NTRU secret key with $\mathbb{F}_q$-error probability $p = \mathcal{O}\left(\frac{\sqrt{\log n}}{n}\right)$. Then the secret key (f, f^{-1}) can be recovered in polynomial time with high success probability.*

Proof. Let

$$f = \sum_{i=0}^{n-1} (\tilde{f}_i + e_i) x_i \text{ and } f^{-1} = \sum_{i=0}^{n-1} (\tilde{f}_i^{-1} + e'_i) x_i$$

with $p := \Pr[e_i \neq 0] = \Pr[e'_i \neq 0]$.

Let us rewrite the identity $1 = \tilde{f} \cdot \tilde{f}^{-1}$ as

$$\delta_{k0} = \sum_{i+j=k \bmod n} (\tilde{f}_i + e_i)(\tilde{f}_j^{-1} + e'_j) = \sum_{i+j=k \bmod n} \tilde{f}_i e'_j + \tilde{f}_j^{-1} e_i + e_i e'_j + \tilde{f}_i \tilde{f}_j^{-1}. \quad (7)$$

Note that the terms $\tilde{f}_i \tilde{f}_j^{-1}$ are known. We linearize the unknown quadratic terms $e_i e'_j$ as the error terms

$$\bar{e}_k = \sum_{i+j=k \bmod n} e_i e'_j \bmod 3.$$

Thus, Equation (7) provides us n linear equations in the $3n$ unknowns $(\mathbf{e}, \mathbf{e}', \bar{\mathbf{e}})$. Let $\tilde{\mathbf{F}}$, $\tilde{\mathbf{F}}^{-1} \in \mathbb{F}_3^{n \times n}$ denote the matrices that correspond to multiplication by f and f^{-1} in R_3, respectively. Moreover, let $\mathbf{f} = (f_0, \ldots, f_{n-1})$, where $f_k = \sum_{i+j=k \bmod n} \tilde{f}_i \tilde{f}'_j \bmod 3$ is the constant term of δ_{k0} in Equation (7). Then we can rewrite Equation (7) in matrix-vector form as

$$\left[\tilde{\mathbf{F}}^{-1} | \tilde{\mathbf{F}} | \mathbf{I}_n\right] \cdot (\mathbf{e}, \mathbf{e}', \bar{\mathbf{e}}) = \mathbf{f}.$$

This is an ISD instance over $\mathbb{F}_3^{n \times 3n}$. The unknowns $\mathbf{e}, \mathbf{e}'$ both have expected weight $np = O(\sqrt{\log n})$. It remains to determine the weight of $\bar{\mathbf{e}}$. Since $\Pr[e_i e_j \neq 0] = p^2$, an application of Bernoulli's identity yields

$$\Pr[\bar{e}_k = 0] \geqslant \left(1 - p^2\right)^n \geqslant 1 - np^2.$$

Thus, $\bar{\mathbf{e}}$ has expected weight at most $n^2 p^2 = \mathcal{O}(\log n)$. This in turn implies that the unknown vector $(\mathbf{e}, \mathbf{e}', \bar{\mathbf{e}})$ has in total expected weight at most $\mathcal{O}(\log n)$. Now, Theorem 2.1 allows for polynomial-time recovery of $(\mathbf{e}, \mathbf{e}', \bar{\mathbf{e}})$. □

5.2 Practical Attacks on NTRU

In Section 5.1, we exploited the key equation $f \cdot f^{-1} = 1 \bmod 3$ to recover the secret key in polynomial time. However, Remark 5.1 already indicates limitations for scaling our strategy to larger error rates. Hence, in our practical attacks we devise a new strategy based on the key equation $h \cdot f = g$ instead. First we derive an LWE instance from the given partially erased or erroneous key material, whose complexity is then estimated using the LWE estimator [3] in combination with the asymptotic lattice reduction exponent of 0.3496 obtained in [2]. Second, we give a combinatorial attack that achieves the best complexities for a compact key representation.

Key Formats The NTRU documentation specifies two key formats, a *packed* and an *unpacked* format. The unpacked format stores each coefficient of a ternary polynomial via two bits. Therefore the values 0, 1 and -1 are represented in binary as 00, 01 and 10. This key format is used whenever the secret key is accessed, e.g., during decryption. The packed format is used to store the secret key in the meantime, by packing five ternary coefficients, with total information $5 \log 3 \approx 7.92$ bits, into 8 bits. The conversion algorithm is outlined in Appendix C.1. Let us detail how we translate bit to field errors/erasures for the different key formats.

Unpacked format. For the unpacked format we observe that a single bit-erasure of the form ?1 and 1?, where "?" denotes an erased bit can directly be recovered to 01 and 10 coefficients. For every coefficient the probability of a single bit-erasure is $2p(1-p)$, while the probability for two bit-erasures is p^2. Since the secret key is drawn randomly from all ternary polynomials, the probability that we can directly recover a single-bit affected coefficient is $1/3$.[7] Thus, we can reduce the total amount of expected erasure-affected $\mathbb{F}_3$-coefficients from $(2p-p^2)n$ to

$$(2p-p^2)n - \frac{2p(1-p)n}{3} = \frac{np}{3}(4-p),$$

or put differently obtain a field-erasure probability of $\frac{p}{3}(4-p)$.

For the translation of bit-*errors* to field-*errors* we proceed similar, treating a coefficient as erroneous as long as any bit in its binary representation is error-prone. Hence, a bit-error probability of p results in a field error probability of $1-(1-p)^2 = 2p-p^2$ (analogously to Remark 4.2).

Packed format. For the packed format an erased or erroneous bit in its binary representation might affect multiple coefficients of the polynomial in unpacked form. We determine the field-error and -erasure rates caused by a certain bit-error/-erasure by an exhaustive enumeration of all possibilities. We start with the erasure translation. We enumerate all possible 3^5 values for an 8-bit block in packed representation and all possible positions for i bit-erasures for $i = 1, \ldots, 8$, to derive the proportion of i bit-erasures leading to j field-erasures. Essentially, for every combination of value and i erasure positions, we enumerate all possibilities for these i bits and transform for each guess to unpacked form. This results in five ternary coefficients. Now, those coefficients which are equal among all guesses are known, while those differing among at least two guesses are treated as erased. For completeness we give the derived proportions in Table 4 in Appendix C.1.

For the bit- to field- *error* translation we proceed similar by enumerating all possible value-error pairs and counting the errors caused in the unpacked representation. We give the complete results in Table 5 in Appendix C.1.

Practical Attack in the Erasure Model For our attack in the erasure model we derive a dimension-reduced small-secret LWE instance from the partially erased key material. Let $\tilde{f}$ be a partially erased version of f, where I_f denotes the set of erased coefficients. The key equation gives $h \cdot f = g$ or equivalently $\mathbf{Hf} = \mathbf{g}$, where $\mathbf{H}$ is the multiplication matrix of h and $\mathbf{f}, \mathbf{g}$ are the coefficient vectors of f and g. By denoting the columns of $\mathbf{H}$ as $\mathbf{h}_i$, we obtain

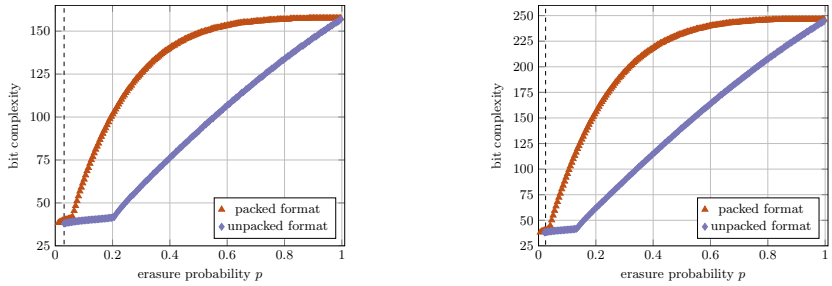
$$\mathbf{Hf} = \mathbf{g} \Leftrightarrow \sum_{i=1}^{n} \mathbf{h}_i f_i = \mathbf{g} \Leftrightarrow \sum_{i \in I_f} \mathbf{h}_i f_i - \mathbf{g} = -\sum_{i \notin I_f} \mathbf{h}_i f_i.$$

[7] The secret key coefficient can take the values 01, 10 and 00 while only the two (out of 6) possible single-bit erasures ?1 and 1? can be recovered directly.

By letting $\widehat{\mathbf{H}}$ denote the matrix containing the columns $\mathbf{H}$ indexed by I_f, and analogously $\widehat{\mathbf{f}}$ denote the vector containing the coordinates of $\mathbf{f}$ indexed by I_f, we obtain $\widehat{\mathbf{H}}\widehat{\mathbf{f}} - \mathbf{g} = -\sum_{i \notin I_f} \mathbf{h}_i f_i$. Note that since the right hand side of the equation is known and $\mathbf{g}$ is small by definition this yields an LWE instance with secret $\widehat{\mathbf{f}}$ of dimension $|I_f|$, which is the number of erased $\mathbb{F}_q$ coefficients.

To determine the bit complexity of the outlined attack for various erasure probabilities p, we proceed as follows. First, we relate the bit-erasure probability p to a field-erasure probability. For the unpacked format, as outlined in Section 5.2, we simply use the field-error probability $\frac{p}{3}(4-p)$. For the packed format Table 4 states how to translate a certain number of bit-errors in an 8-bit block to a certain number of field-errors. Hence, we first calculate the expected number N_i of blocks out of the total $n/5$ affected by i bit-erasures, $i = 1, \ldots, 8$. Then we compute the number of expected field-erasures as $\sum_i N_i E_i$, where E_i is the entry of the last column of Table 4 in row i.

From there we use the LWE estimator to determine the bit complexity for solving the derived LWE instances. Our results for both formats are depicted in Fig. 5. The vertical dashed line represents the erasure probability up to which our polynomial time attacks from Section 5.1 can be applied.



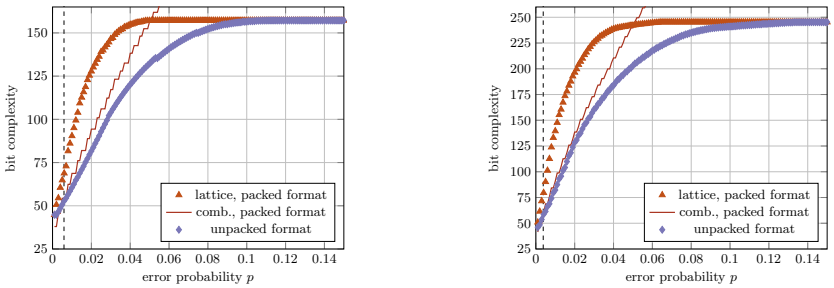
(a) Category-1: $(n, q) = (509, 2048)$ (b) Category-3: $(n, q) = (821, 4096)$

Fig. 5: Bit complexity of our partial key exposure attack in the erasure model on NTRU.

Practical Attack in the Error Model Our practical attack in the error model is quite similar to the attack in erasure model. Let us first outline the attack for a symmetric error, which does not exploit any dimension reduction. In the error setting, we obtain a noisy version $\tilde{f} = f + e$ of f. Similar to before we have

$$\mathbf{H}\tilde{\mathbf{f}} = \mathbf{H}\mathbf{e} + \mathbf{H}\mathbf{f} = \mathbf{H}\mathbf{e} + \mathbf{g}, \tag{8}$$

which defines an LWE instance of dimension n and secret $\mathbf{e}$, where $\mathbf{e}$ has expected $p_q \cdot n$ entries different from zero, where p_q is the field-error probability.



(a) Category-1: $(n, q) = (509, 2048)$ (b) Category-3: $(n, q) = (821, 4096)$

Fig. 6: Bit complexity of key recovery on NTRU in the symmetric error model.

To derive the bit complexity of recovering the secret key for a given bit-error probability p we again first relate p to the field-error probability p_q. Then we use the LWE estimator to estimate the hardness of the above LWE instance.

For the unpacked format we have $p_q = 2p - p^2$ (compare to Section 5.2). For the packed format we again compute the number N_i of expected 8-bit blocks affected by $i = 1, \ldots, 8$ errors. Then we derive the number of field-errors as $\sum_i N_i E_i$, where E_i is the expected number of field-errors caused by i bit-errors, which we obtain from the i-th row of the last column of Table 5.

In Fig. 6 we plot the derived bit-complexity as a function of the error-probability p. Since already a few errors in packed representation often lead to multiple errors in the unpacked form, we see a steep incline for our attack in packed form. Therefore we give in the following another, combinatorial approach inspired by recent results of May [20].

Combinatorial approach Imagine we guess the first $\ell < n$ coordinates of $\mathbf{g}$ in Eq. (8). This gives an equation of the form $\mathbf{H}_\ell(\tilde{\mathbf{f}} - \mathbf{e}) = \mathbf{g}_\ell$, where $\mathbf{H}_\ell$ is the matrix formed by the first ℓ rows of $\mathbf{H}$ and analogously $\mathbf{g}_\ell$ contains the first ℓ coordinates of $\mathbf{g}$.

From here we perform a meet-in-the-middle attack on $\mathbf{e}$. Therefore let $\mathbf{e} = (\mathbf{e}_1, \mathbf{e}_2) \in \mathbb{F}_3^{n/2} \times \mathbb{F}_3^{n/2}$, $\tilde{\mathbf{f}} = (\tilde{\mathbf{f}}_1, \tilde{\mathbf{f}}_2) \in \mathbb{F}_3^{n/2} \times \mathbb{F}_3^{n/2}$ and $\mathbf{H}_\ell = (\mathbf{H}_1 \mid \mathbf{H}_2)$, which gives

$$\mathbf{H}_1(\tilde{\mathbf{f}}_1 - \mathbf{e}_1) = \mathbf{g}_\ell - \mathbf{H}_2(\tilde{\mathbf{f}}_2 - \mathbf{e}_2).$$

Now we enumerate all possible values for $\mathbf{e}_1$ (*resp.* $\mathbf{e}_2$) and store the corresponding value of the right-hand side (*resp.* left-hand side) of the above equation in a list L_1 (*resp.* L_2). Then we search between those lists for equal elements, where by construction one matching pair reveals $\mathbf{e} = (\mathbf{e}_1, \mathbf{e}_2)$.

Note that the expected amount of matching pairs is $L := \frac{|L_1| \cdot |L_2|}{q^\ell}$ and that the matching can be performed in time $\mathcal{O}\big(\max\{|L|, |L_1|, |L_2|\}\big)$. The list construction requires per element (after the first) roughly 2ℓ field multiplications, if a gray-code style enumeration for $\mathbf{e}_1$ and $\mathbf{e}_2$ is chosen. Moreover to check if an element of $\mathbf{x} \in L$ reveals the searched error we need to compute $\mathbf{H}_\ell \mathbf{x}$, which costs ℓn

field multiplications. Thus by accounting for a single field multiplication $\log^2 q$ bit operations, and observing that $|L_1| = |L_2|$, we find a bit complexity of

$$\mathcal{O}\big(\max\{\ell n \cdot |L|, \ell \cdot |L_1|\} \cdot 3^\ell \log^2 q\big).^{8}$$

In the packed form, we enumerate the error first, subtract it from the respective part of the key material and then convert it to unpacked form. Thus, more precisely, the lists contain $\mathbf{H}_1 \cdot \mathrm{unpack}(\tilde{\mathbf{f}}_1 - \mathbf{e}_1)$ and $\mathbf{g}_\ell - \mathbf{H}_2 \cdot \mathrm{unpack}(\tilde{\mathbf{f}}_2 - \mathbf{e}_2)$ respectively. This is possible since every 8-bit block of the packed form can be converted to unpacked form independently.

In the packed form the bit length of $\tilde{\mathbf{f}}$ is roughly $\frac{8n}{5}$, hence the expected Hamming weight of the binary representation of $\mathbf{e}$ is $\frac{8np}{5}$. Thus, in expectation we have $|L_1| = |L_2| = \mathcal{O}\big(\binom{4n/5}{4np/5}\big)$.

This combinatorial attack yields an improved key recovery attack on the packed format, while on the unpacked format lattice enumeration is still preferable. This is because the combinatorial attack benefits from the small bit length of $\tilde{\mathbf{f}}$ in packed representation.

In [20] May gives further advanced combinatorial attacks extending the here presented meet-in-the-middle by a search-tree approach and the representation technique. We leave it as an open research task to determine the gain of those advancements in our settings.

An improved practical attack when facing asymmetric error probabilities is given in Appendix C.2.

References

1. Albrecht, M.R., Deo, A., Paterson, K.G.: Cold boot attacks on ring and module LWE keys under the NTT. Cryptology ePrint Archive (2018)
2. Albrecht, M.R., Ducas, L., Herold, G., Kirshanova, E., Postlethwaite, E.W., Stevens, M.: The general sieve kernel and new records in lattice reduction. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 717–746. Springer (2019)
3. Albrecht, M.R., Player, R., Scott, S.: On the concrete hardness of learning with errors. Journal of Mathematical Cryptology **9**(3), 169–203 (2015)
4. Aragon, N., Barreto, P., Bettaieb, S., Bidoux, L., Blazy, O., Deneuville, J.C., Gaborit, P., Gueron, S., Guneysu, T., Melchor, C.A., et al.: BIKE: bit flipping key encapsulation (2020)
5. Billet, O., Gilbert, H.: Cryptanalysis of Rainbow. In: De Prisco, R., Yung, M. (eds.) Security and Cryptography for Networks. pp. 336–347. Springer Berlin Heidelberg, Berlin, Heidelberg (2006)
6. Blömer, J., May, A.: New partial key exposure attacks on RSA. In: Annual International Cryptology Conference. pp. 27–43. Springer (2003)
7. Boneh, D., Durfee, G., Frankel, Y.: An attack on RSA given a small fraction of the private key bits. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 25–34. Springer (1998)

[8] The factor 3^ℓ can be slightly improved by guessing zero coordinates of $\mathbf{g}$ instead of enumerating the first ℓ coordinates.

8. Chen, C., Danba, O., Hoffstein, J., Hülsing, A., Rijneveld, J., Schanck, J.M., Schwabe, P., Whyte, W., Zhang, Z.: NTRU algorithm specifications and supporting documentation (2019), https://ntru.org/f/ntru-20190330.pdf
9. Coppersmith, D.: Small solutions to polynomial equations, and low exponent RSA vulnerabilities. Journal of cryptology **10**(4), 233–260 (1997)
10. Dachman-Soled, D., Ducas, L., Gong, H., Rossi, M.: LWE with side information: attacks and concrete security estimation. In: Annual International Cryptology Conference. pp. 329–358. Springer (2020)
11. Ding, J., Chen, M.S., Petzoldt, A., Schmidt, D., Yang, B.Y.: Rainbow. NIST CSRC (2020), https://csrc.nist.gov/Projects/post-quantum-cryptography/round-3-submissions
12. Espitau, T., Fouque, P.A., Gérard, B., Tibouchi, M.: Side-channel attacks on BLISS lattice-based signatures: Exploiting branch tracing against strongswan and electromagnetic emanations in microcontrollers. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. pp. 1857–1874 (2017)
13. Esser, A., Bellini, E.: Syndrome decoding estimator. In: International Workshop on Public Key Cryptography. Springer (2022)
14. Goldwasser, S., Kalai, Y.T., Peikert, C., Vaikuntanathan, V.: Robustness of the learning with errors assumption. In: ICS. pp. 230–240. Tsinghua University Press (2010)
15. Halderman, J.A., Schoen, S.D., Heninger, N., Clarkson, W., Paul, W., Calandrino, J.A., Feldman, A.J., Appelbaum, J., Felten, E.W.: Lest we remember: cold-boot attacks on encryption keys. Communications of the ACM **52**(5), 91–98 (2009)
16. Henecka, W., May, A., Meurer, A.: Correcting errors in RSA private keys. In: Annual Cryptology Conference. pp. 351–369. Springer (2010)
17. Heninger, N., Shacham, H.: Reconstructing RSA private keys from random key bits. In: Annual International Cryptology Conference. pp. 1–17. Springer (2009)
18. Horlemann, A., Puchinger, S., Renner, J., Schamberger, T., Wachter-Zeh, A.: Information-set decoding with hints. In: International Workshop on Code-Based Cryptography (CBCrypto) (2021)
19. Lidl, R., Niederreiter, H.: Finite Fields. Encyclopedia of Mathematics and its Applications, Cambridge University Press, 2 edn. (1996)
20. May, A.: How to meet ternary LWE keys. In: Annual International Cryptology Conference. pp. 701–731. Springer (2021)
21. Melchor, C.A., Aragon, N., Bettaieb, S., Bidoux, L., Blazy, O., Deneuville, J.C., Gaborit, P., Persichetti, E., Zémor, G., Bourges, I.: Hamming quasi-cyclic (HQC) (2020)
22. Mitzenmacher, M., Upfal, E.: Probability and computing: Randomization and probabilistic techniques in algorithms and data analysis. Cambridge university press (2017)
23. Paterson, K.G., Polychroniadou, A., Sibborn, D.L.: A coding-theoretic approach to recovering noisy RSA keys. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 386–403. Springer (2012)
24. Paterson, K.G., Villanueva-Polanco, R.: Cold boot attacks on NTRU. In: International Conference on Cryptology in India. pp. 107–125. Springer (2017)
25. Polanco, R.V.: Cold Boot Attacks on Post-Quantum Schemes. Ph.D. thesis, Royal Holloway, University of London (2019)
26. Prange, E.: The use of information sets in decoding cyclic codes. IRE Transactions on Information Theory **8**(5), 5–9 (1962)

27. Raab, M., Steger, A.: "balls into bins"—a simple and tight analysis. In: International Workshop on Randomization and Approximation Techniques in Computer Science. pp. 159–170. Springer (1998)
28. Villanueva-Polanco, R.: Cold boot attacks on BLISS. In: International Conference on Cryptology and Information Security in Latin America. pp. 40–61. Springer (2019)
29. Villanueva-Polanco, R.: Cold boot attacks on LUOV. Applied Sciences **10**(12) (2020). https://doi.org/10.3390/app10124106

A BIKE

A.1 Proof of Theorem 3.2

Before we give the proof of Theorem 3.2 , let us make explicit the following remark on the approximation of binomial coefficients, which we already proved implicitly in the proof of Theorem 2.1.

Remark A.1. Let $n \in \mathbb{N}$ and $0 < c < 1$. For $\delta = o(cn)$ it holds that

$$\frac{\binom{n}{\delta}}{\binom{cn}{\delta}} = \left(\frac{1}{c}\right)^{\delta(1+o(1))}.$$

Now we give the proof of Theorem 3.2, for convenience we restate the theorem first.

Theorem 3.2 (Polynomial Error Attack on Standard Format). *Let $\tilde{\mathbf{f}}$ be a given erroneous BIKE secret key in the standard format with $\mathbb{F}_2$-error probability $p = \mathcal{O}\left(\frac{\log n}{\sqrt{n}}\right)$. Then the secret key $\mathbf{f} \in \mathbb{F}_2^n$ can be recovered in polynomial time.*

Proof. The expected running time to recover the BIKE secret key is (up to polynomial factors) factors given by $\mathbb{E}[q^{-1}]$, as given by Eq. (3). Let us first approximate this expectation via Remark A.1. Note that for the BIKE choice of $\delta := \mathbb{E}[\mathrm{wt}(\mathbf{f})] = \Theta(\sqrt{n})$ and our choice of $p = \mathcal{O}\left(\frac{\log n}{\sqrt{n}}\right)$, we have $(1-p)\delta = o(\delta_{\tilde{\mathbf{f}}})$. Recall that $\delta_{\tilde{\mathbf{f}}} := \mathbb{E}[\mathrm{wt}(\tilde{\mathbf{f}})] = \delta(1-p) + p(n-\delta)$, hence,

$$(1-p)\delta = o(\delta_{\tilde{\mathbf{f}}}) \;\Leftrightarrow\; (1-p)\delta = o(p(n-\delta)) \;\Leftrightarrow\; \delta = o(\sqrt{n\log n}).$$

Also we have $p\delta = o(n - \delta_{\tilde{\mathbf{f}}})$, and since we consider constant rate and $\rho_1 \leqslant \delta_{\tilde{\mathbf{f}}}$ we also have $p\delta = o(n-k-\rho_1)$. In turn this allows us to apply Remark A.1 to approximate the expected time complexity of the attack up to polynomial factors as

$$\mathbb{E}[q^{-1}] = \frac{\binom{\delta_{\tilde{\mathbf{f}}}}{(1-p)\delta}\binom{n-\delta_{\tilde{\mathbf{f}}}}{p\delta}}{\binom{\rho_1}{(1-p)\delta}\binom{n-k-\rho_1}{p\delta}} \overset{n\to\infty}{\to} \left(\frac{\delta_{\tilde{\mathbf{f}}}}{\rho_1}\right)^{(1-p)\delta(1+o(1))} \cdot \left(\frac{n-\delta_{\tilde{\mathbf{f}}}}{n-k-\rho_1}\right)^{p\delta(1+o(1))},$$

as long as $\rho_1 = \omega\big((1-p)\delta\big)$, to ensure the prerequisite of Remark A.1. Note that for small p the first term grows inevitably exponential in δ as soon as ρ_1 is at least a constant factor smaller than $\delta_{\tilde{\mathbf{f}}}$. On the other hand decreasing ρ_1 does not help to decrease the second term (asymptotically), since $\rho_1 \leqslant \delta_{\tilde{\mathbf{f}}} = o(n-k)$. Thus, the choice $\rho_1 = \delta_{\tilde{\mathbf{f}}}$ minimizes the running time as

$$\begin{aligned}\mathbb{E}[q^{-1}] &= \left(\frac{n-\delta_{\tilde{\mathbf{f}}}}{n-k-\delta_{\tilde{\mathbf{f}}}}\right)^{p\delta(1+o(1))} = \left(\frac{1-o(1)}{1-R-o(1)}\right)^{p\delta(1+o(1))} \\ &= \left(\frac{1}{1-R}\right)^{p\delta(1+o(1))} = \left(\frac{1}{1-R}\right)^{\mathcal{O}(\log n)},\end{aligned}$$

which is polynomial in n. □

A.2 Proof of Theorem 3.3

Proof. Considering the concrete format of the private key every index encoding a position where the private key is one is stored using $\log k$ bits. Hence, an amount of ε bit-erasures in any index leads to 2^{ε} candidates for that position. Given the set of erased bits $I \subseteq [\delta \log(k)]$ we denote by $\varepsilon_i := |\{j \in [(i-1)\log(k)+1, i\log(k)] \mid j \in I\}|$ for $i = 1, \ldots, \delta$ the amount of bit-erasures in index i. Again, as long as the amount of total candidates, i.e., the amount of unknowns, is less or equal to k we find the private key in polynomial time via Gaussian elimination, as the parity-check matrix yields k linear equations. Hence, as long as $\sum_{i=1}^{\delta} 2^{\varepsilon_i} \leqslant k$ we recover the private key in polynomial time. Note that since $\delta = \Theta(\sqrt{k})$ the above inequality is especially fulfilled if for all i it holds that $2^{\varepsilon_i} \leqslant \sqrt{k}$, which is equivalent to $\varepsilon_i \leqslant \frac{\log k}{2}$. Hence, it is enough to ensure that each index contributes with at most $\frac{\log k}{2}$ erasures. Using a Chernoff bound [22] yields

$$q := \Pr\left[\varepsilon_i > (1+\gamma)p\log k\right] < \left(\frac{e^{\gamma}}{(1+\gamma)^{(1+\gamma)}}\right)^{p\log k} \leqslant e^{-\frac{\gamma\ln(1+\gamma)p\log k}{2}},$$

for any $\gamma > 0$, where p is the erasure probability. Thus, since $\delta = c\sqrt{k}$, for some constant c, the probability that for all i we have $\varepsilon_i \leqslant (1+\gamma)p\log k$ is

$$\Pr\left[\varepsilon_i \leqslant (1+\gamma)p\log k \ \forall i\right] = (1-q)^{\delta} \geqslant (1-e^{-\frac{\gamma\ln(1+\gamma)p\log k}{2}})^{c\sqrt{k}}.$$

Hence, as long as $(1+\gamma)p \leqslant \frac{1}{2}$ and $q \leqslant \frac{1}{\sqrt{k}}$ there is no block with more than $\sqrt{k}$ erasures with constant probability. Solving the system

$$(1+\gamma)p = \frac{1}{2} \quad \text{and} \quad \frac{\gamma\ln(1+\gamma)p\log(e)}{2} = \frac{1}{2},$$

yields $\gamma = 4.43$ and $p = 0.092$. Hence, with constant probability an erasure rate of $p \leqslant 0.092$ leads to a polynomial time recovery of the secret key. □

A.3 Details on the Proof of Theorem 3.4

Recall that we have

$$\varepsilon \leqslant \frac{\log d}{\log \frac{d \log d}{m}} \left(1 + \alpha \frac{\log \log \frac{d \log d}{m}}{\log \frac{d \log d}{m}}\right),$$

for any $\alpha > 1$ with high probability. Thus for our choice of $d = \delta \leqslant c_\delta \cdot \sqrt{k}$ and $m = \Theta(\mathcal{E}) \geqslant c_m \cdot \frac{\sqrt{k}}{\log^{1+\kappa} k}$, where c_δ, c_m are constants, we have $\frac{d \log d}{m} \leqslant \frac{c_\delta}{2c_m} \sqrt{k} \cdot \log^{2+\kappa}(c_\delta k) = \mathcal{O}\left(\sqrt{k} \cdot \log^{2+\kappa} k\right)$ resulting in

$$\varepsilon \leqslant c_\varepsilon \cdot \frac{\log k}{\log \log^{2+\kappa} k} \left(1 + \alpha \frac{\log \log \log^{2+\kappa} k}{\log \log^{2+\kappa} k}\right)$$

for some small constant c_ε of order $\log c_d + \log c_m$ with high probability. Further analysis shows that this implies that

$$\lim_{k \to \infty} \varepsilon \leqslant \frac{c_\varepsilon}{2 + \kappa} \cdot \frac{\log k}{\log \log k}.$$

Thus, as long as $\kappa > 2(c_\varepsilon - 1)$, we have $\varepsilon < \frac{\log k}{2 \log \log k}$ as desired.

B Rainbow

In this section we give the proof of Proposition 4.1 and outline the procedure for full Rainbow key recovery.

B.1 Proof of Proposition 4.1

Proposition 4.1. *Let f_i be the i-th polynomial in the first layer of a Rainbow central map, that is, $i \leqslant o_1$. Let $(p_1, \ldots, p_{o_1+o_2})$ be a Rainbow public key, where the corresponding secret maps $\mathcal{S}$ and $\mathcal{T}$ are homogeneous, and their matrix representations are as shown in Equation (4). Then, we have*

$$p_i^{\mathbf{v}}(x_1, \ldots, x_v) + \sum_{j=o_1+1}^{o_1+o_2} s_{i,j} \cdot p_j^{\mathbf{v}}(x_1, \ldots, x_v) = f_i^{\mathbf{v}}(x_1, \ldots, x_v),$$

where $f_i^{\mathbf{v}}(x_1, \ldots, x_v)$ is the vinegar part of f_i, $(s_{i,1}, \ldots, s_{i,o_1+o_2})$ is the i-th row of $\mathbf{S}^{-1}$, and $p_j^{\mathbf{v}}(x_1, \ldots, x_v)$ is the vinegar part of p_j.

Proof. Let $(p_1, \ldots, p_{o_1+o_2}) = \mathcal{S} \circ (f_1, \ldots, f_{o_1+o_2}) \circ \mathcal{T}$ be a Rainbow public key, where the linear secret maps $\mathcal{S}$ and $\mathcal{T}$ are homogeneous.

Let $\mathbf{S}$ and $\mathbf{T}$ be the matrices representing $\mathcal{S}$ and $\mathcal{T}$, respectively. By Remark 4.1, we know that $\sum_{j=1}^{o_1+o_2} s_{i,j} p_j(\mathbf{x}) = f_i(\mathbf{Tx})$, where $(s_{i,1}, \ldots, s_{i,o_1+o_2}) \in \mathbb{F}_q^{o_1+o_2}$ is the i-th row of $\mathbf{S}^{-1}$. Hence, we have that

$$f_i^{\mathbf{v}}(\mathbf{Tx}) = \left(\sum_{j=1}^{o_1+o_2} s_{i,j} p_j\right)^{\mathbf{v}} = \sum_{j=1}^{o_1+o_2} s_{i,j} p_j^{\mathbf{v}}(x_1, \ldots, x_v)$$

If $\mathbf{T}$ is as defined in Equation (4), then the variables $x_1, \ldots, x_v$ only appear in the first v coordinates of $\mathbf{Tx}$. Furthermore, the vector formed by the first v coordinates of $\mathbf{Tx}$ is given by $(x_1, \ldots, x_v)^\top + \mathbf{T}^{(1)}_{v\times o_1} \cdot (x_{v+1}, \ldots, x_{v+o_1})^\top + \mathbf{T}^{(3)}_{v\times o_2} \cdot (x_{v+o_1+1}, \ldots, x_{v+o_1+o_2})^\top$.

Hence, the coefficient of the monomial $x_i x_j$ is the same in $f_i(\mathbf{x})$ and in $f_i(\mathbf{Tx})$ for any $1 \leqslant i, j \leqslant v$,. That is, $f_i^{\mathbf{v}}(\mathbf{Tx}) = f_i^{\mathbf{v}}(\mathbf{x})$. Finally, if $\mathbf{S}$ is also as in Equation (4), then $p_i^{\mathbf{v}}(x_1, \ldots, x_v) + \sum_{j=o_1+1}^{o_1+o_2} s_{i,j} \cdot p_j^{\mathbf{v}}(x_1, \ldots, x_v) = f_i^{\mathbf{v}}(x_1, \ldots, x_v)$.

B.2 Full Key Recovery on Rainbow

Let us describe the algorithm that runs in time $\mathrm{Poly}(o_2)$ and recovers the full Rainbow private key from one row of the hidden matrix $\mathbf{S}'$, which represent the hidden map $\mathcal{S}$ (see Equation (4)). Recall that we let $n := o_1 + o_2 + v$.

Without loss of generality let us assume we know the first row $(s_1, \ldots, s_{o_2}) \in \mathbb{F}_q^{o_2}$ of $\mathbf{S}'$. We denote by $p_1, \ldots, p_{o_1+o_2}$ the polynomials in a given Rainbow public key, while by $f_1, \ldots, f_{o_1+o_2}$ the polynomials in the hidden central map. Also, we use $\mathbf{P}_i$ and $\mathbf{F}_k$ to denote, respectively, the symmetric matrices representing the bilinear forms

$$p_i'(\mathbf{x}, \mathbf{y}) := p_i(\mathbf{x} + \mathbf{y}) - p_i(\mathbf{x}) - p_i(\mathbf{y}) + p_i(\mathbf{0}) \text{and}$$
$$f_k'(\mathbf{x}, \mathbf{y}) := f_k(\mathbf{x} + \mathbf{y}) - f_k(\mathbf{x}) - f_k(\mathbf{y}) + f_k(\mathbf{0}).$$

We first establish the following well known result.

Lemma B.1. *Let $(s_1, \ldots, s_{o_2})$ be the i-th row of $\mathbf{S}' \in \mathbb{F}_q^{o_1\times o_2}$ and $n := v+o_1+o_2$. Then,*

$$\mathit{Rank}\left(\mathbf{P}_i + \sum_{j=1}^{o_2} s_j \mathbf{P}_{o_1+j}\right) = n - o_2.$$

Also, it is expected to be the only vector satisfying the above equation. Finally, it is possible to check whether a given vector $\mathbf{s} \in \mathbb{F}_q^{o_2}$ is or is not a row of the hidden matrix $\mathbf{S}'$ in

$$o_2 \cdot (v + o_1 + o_2)^2 + (v + o_1 + o_2)^\omega$$

multiplications over $\mathbb{F}_q$.

Proof. By Remark 4.1 and Equation (4), we know that

$$\underbrace{p_i(\mathbf{x}) + \sum_{j=1}^{o_2} s_j p_j(\mathbf{x})}_{:=p(\mathbf{x})} = f_i\left(\mathbf{Tx}\right).$$

Then, $p(\mathbf{x})$ and $f_i\left(\mathbf{Tx}\right)$ induce the same bilinear form, that is, $p'(\mathbf{x}, \mathbf{y}) = (f_i \circ \mathcal{T})'(\mathbf{x}, \mathbf{y})$. Let $\mathbf{P}$ and $\mathbf{F}_{i,t}$ be the matrices representing p' and $(f_i \circ \mathcal{T})'$, respectively. It holds

$$\mathbf{P}_i + \sum_{j=1}^{o_2} s_j \mathbf{P}_{o_1+j} = \mathbf{P} = \mathbf{F}_{i,t} = \mathbf{T}^\top \mathbf{F}_i \mathbf{T},$$

and consequently

$$\mathsf{Rank}\left(\mathbf{P}_i + \sum_{j=1}^{o_2} s_j \mathbf{P}_{o_1+j}\right) = \mathsf{Rank}\left(\mathbf{T}^\top \mathbf{F}_i \mathbf{T}\right) = n - o_2.$$

The rest of the Lemma directly follows from the dimensions of the matrices.

Recovering the last o_2 columns of $\mathbf{T}^{-1}$ Remember from Equation (4) that

$$\mathbf{T}^{-1} = \begin{bmatrix} \mathbf{I}_v & \mathbf{T}^{(1)} & \mathbf{T}^{(4)} \\ \mathbf{O} & \mathbf{I}_{o_1} & \mathbf{T}^{(3)} \\ \mathbf{O} & \mathbf{O} & \mathbf{I}_{o_2} \end{bmatrix}.$$

The last o_2 columns of $\mathbf{T}^{-1}$ define a submatrix $\mathbf{A} := \begin{bmatrix} \mathbf{T}^{(4)\top} & \mathbf{T}^{(3)\top} & \mathbf{I}_{o_2} \end{bmatrix}^\top \in \mathbb{F}_q^{n \times o_2}$. Suppose for a moment that the following equation holds

$$\left(\mathbf{P}_1 + \sum_{j=1}^{o_2} s_j \mathbf{P}_{o_1+j}\right) \cdot \mathbf{A} = \mathbf{O}. \tag{9}$$

By Lemma B.1

$$\mathsf{Rank}\left(\mathbf{P}_1 + \sum_{j=1}^{o_2} s_j \mathbf{P}_{o_1+j}\right) = n - o_2.$$

Thus, $\mathbf{A}$ is the unique generator matrix of the right kernel of $\mathbf{P}_1 + \sum_{j=1}^{o_2} s_j \mathbf{P}_{o_1+j}$ whose submatrix formed by its last o_2 rows is the identity. Therefore, since we know the first row $(s_1, \ldots, s_{o_2})$ of $\mathbf{S}'$, we can find $\mathbf{A}$ in polynomial time in n.

Now we proof that $\mathbf{A}$ indeed satisfy Equation (9).

Lemma B.2. *Let $(s_{i,1}, \ldots, s_{i,o_2})$ be the i-th row of $\mathbf{S}'$. Then,*

$$\left(\mathbf{P}_i + \sum_{j=1}^{o_2} s_{i,j} \mathbf{P}_{o_1+j}\right) \cdot \mathbf{A} = \mathbf{O}.$$

Proof. First notice that $\mathbf{T} \cdot \mathbf{A} = \begin{bmatrix} \mathbf{O} & \mathbf{O} & \mathbf{I}_{o_2} \end{bmatrix} \in \mathbb{F}_q^{n \times o_2}$. By Remark 4.1, for any column $\mathbf{a}$ of $\mathbf{A}$, it holds

$$\begin{aligned} \left(\mathbf{P}_i + \sum_{j=1}^{o_2} s_{i,j} \mathbf{P}_{o_1+j}\right) \cdot \mathbf{a} &= \left(\mathbf{T}^\top \mathbf{F}_1 \mathbf{T}\right) \cdot \mathbf{a} \\ &= \mathbf{T}^\top \cdot \mathbf{0} \\ &= \mathbf{0}. \end{aligned}$$

Recovering the map $\mathcal{S}$ Now we show how to recover the remaining $o_2 - 1$ rows of the matrix $\mathbf{S}'$.

Remember that for any $\mathbf{y} \in \mathbb{F}_q^{o_1+o_2}$

$$\mathcal{S}^{-1}(\mathbf{y}) = \underbrace{\begin{bmatrix} \mathbf{I}_{o_1} & \mathbf{S}' \\ \mathbf{O} & \mathbf{I}_{o_2} \end{bmatrix}}_{:=\mathbf{S}^{-1}} \mathbf{y}.$$

By Lemma B.2, once we know the matrix $\mathbf{A}$, which is the submatrix formed by the last o_2 columns of $\mathbf{T}^{-1}$, every row of $\mathbf{S}'$ satisfies a linear system of $(v + o_1 + o_2)o_2$ equations. Hence, once $\mathbf{A}$ is recovered, we can obtain each row of $\mathbf{S}$ in polynomial time. We provide an implementation of the recovery of $\mathbf{S}'$ from single row in SageMath.

Finding the matrix $\mathbf{T}^{(1)}$ To recover the submatrix $\mathbf{T}^{(1)}$ we use the strategy suggested by Billet and Gilbert in [5, Sec. 3.3].

Initially, we use the first row $(s_1, \ldots, s_{o_2})$ of $\mathbf{S}'$ to compute the polynomial $p_1(\mathbf{x}) + \sum_{j=1}^{o_2} s_j p_{o_1+j}(\mathbf{x}) = (f_1 \circ \mathcal{T})(\mathbf{x})$. Notice that, every element in

$$\mathcal{O}_1 := \mathcal{T}^{-1}\left(\{\text{First } o_1 \text{ canonical vectors in } \mathbb{F}_q^{v+o_1+o_2}\}\right)$$

is a root of $f_1 \circ \mathcal{T}$. Additionally, $\mathcal{O}_1$ is generated by the columns of the matrix

$$\begin{bmatrix} \mathbf{T}^{(1)} \\ \mathbf{I}_{o_1} \\ \mathbf{O} \end{bmatrix} \in \mathbb{F}_q^{n \times o_1}.$$

Hence, recovering $\mathbf{T}^{(1)}$ reduces to find a basis to the vector space $\mathcal{O}_1$.

We define $p_t(\mathbf{x}) := p_1(\mathbf{x}) + \sum_{j=1}^{o_2} s_j p_{o_1+j}(\mathbf{x})$. To retrieve $\mathbf{T}^{(1)}$, we need to find o_1 linearly independent roots of p_t which have zeros in its last o_2 entries. By [19, Theorem 6.30], there is an algorithm, whose running time is $\mathcal{O}\left((v + o_1 + o_2)^3\right)$, that finds a matrix $\mathbf{B} \in \mathbb{F}_q^{(v+o_1)\times n}$ and $a, b \in \mathbb{F}_q$ such that $p_t(\mathbf{x}) = g(\mathbf{B} \cdot \mathbf{x})$, where $g(z_1, \ldots, z_{v+o_1})$ is given by

$$\begin{cases} z_{v+o_1}^2 + \sum_{k=1}^{v+o_1} z_k z_{k+1} & \text{if } v + o_1 \text{ is odd} \\ b(z_{v+o_1-1}^2 + a \cdot z_{v+o_1}^2) + \sum_{k=1}^{v+o_1} z_k z_{k+1} & \text{otherwise,} \end{cases}$$

We use the knowledge of $\mathbf{B}$ and g to find the basis of $\mathbf{O}_1$. Then, we use it to recover $T^{(1)}$.

Recovering the central map Once $\mathbf{S}$ and $\mathbf{T}$ are recovered, we just need to compute

$$\mathcal{S}^{-1} \circ \mathcal{P} \circ \mathcal{T}^{-1}$$

to recover the hidden central map.

C NTRU

C.1 Packed format

Algorithm 1 describes the procedure of converting the secret polynomial in unpacked format to packed format.

Algorithm 1 Algorithm for storing secret f

Require: The secret polynomial f with coefficients $(f_1, \cdots, f_n)$
Ensure: A byte array $(b_1, b_2, \ldots, b_{8\lceil(n-1)/5\rceil})$ of length $8\lceil(n-1)/5\rceil$
  Prepare $(b_1, b_2, \ldots, b_{8\lceil(n-1)/5\rceil}) = (0, 0, \ldots, 0)$
  for $i = 0 \ldots \lceil(n-1)/5\rceil - 1$ **do**
    Set $(c_1, c_2, \ldots, c_5) \in \{0, 1, 2\}^5$ such that $c_j = f_{5i+j} \bmod 3$
    Set $(b_{8i+1}, b_{8i+2}, \ldots, b_{8i+8})$ such that $\sum_{j=0}^{7} 2^j b_{8i+1+j} = \sum_{j=0}^{4} 3^j c_{1+j}$
  end for

In Table 4 we state the calculated proportion of $i = 1, \ldots, 8$ bit-erasures in packed form leading to $j = 1, \ldots, 5$ field-erasures in unpacked form.

bit \ field	1/5	2/5	3/5	4/5	5/5	E[#field]
1/8	12.55%	23.66%	25.10%	25.77%	12.91%	3.03
2/8	0.00%	6.00%	16.11%	27.15%	50.75%	4.23
3/8	0.00%	0.41%	5.47%	19.25%	74.87%	4.69
4/8	0.00%	0.00%	0.94%	10.35%	88.71%	4.88
5/8	0.00%	0.00%	0.00%	4.00%	96.00%	4.96
6/8	0.00%	0.00%	0.00%	0.94%	99.06%	4.99
7/8	0.00%	0.00%	0.00%	0.00%	100.00%	5
8/8	0.00%	0.00%	0.00%	0.00%	100.00%	5

Table 4: Relation between bit erasures and field erasures for the packed format

Table 5 states the proportion of i bit-*errors* resulting in j field-errors.

C.2 Practical Attack on Asymmetric Errors

Let us assume an asymmetric error probability, where we again have $p_0 = 10^{-3}$, i.e., there occur almost entirely one-to-zero-flips. By brute-forcing the positions of the few zero-to-one-flips, this implies that whenever we find a 1 in the binary representation of $\tilde{\mathbf{f}}$, we can take it for certain.

For the unpacked format this means that all 1 (01) and -1 (10) coefficients of $\tilde{\mathbf{f}}$ are also present in $\mathbf{f}$. Only 0 (00) or 3 (11) coefficients could either relate to 0,1 or -1 entries. Similar to the erasure setting any known coefficient of f allows

bit \ field	1/5	2/5	3/5	4/5	5/5	E[#field]
1/8	12.55%	23.66%	25.10%	25.77%	12.91%	3.03
2/8	7.20%	23.57%	33.50%	27.32%	8.41%	3.06
3/8	3.53%	15.89%	29.70%	35.33%	15.55%	3.43
4/8	3.73%	16.69%	33.03%	33.83%	12.72%	3.35
5/8	2.76%	14.20%	35.32%	33.98%	13.74%	3.42
6/8	3.69%	13.40%	36.68%	32.79%	13.32%	3.38
7/8	3.81%	14.97%	34.52%	32.05%	14.45%	3.37
8/8	4.12%	16.46%	32.92%	32.92%	13.17%	3.33

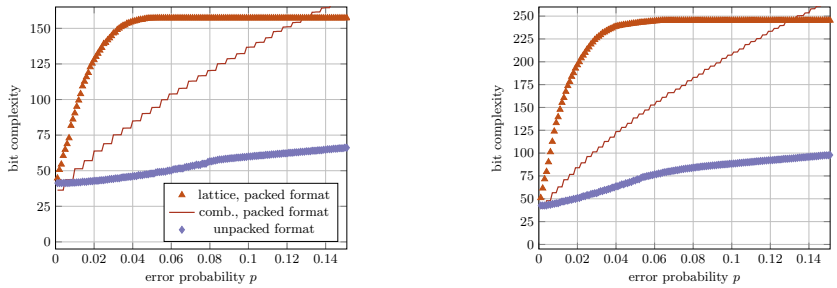
Table 5: Relation between bit errors and field errors for the packed format (symmetric)

to decrease the dimension of the LWE instance by one. As on expectation the secret key contains $\frac{n}{3}$ coefficients equal to 0, 1 and -1 the amount of 1 and -1 entries still present in the erroneous version is $\frac{2n}{3}(1-p_1)$. Or put differently the remaining LWE instance has an expected dimension of $\frac{n}{3}(1+2p_1)$. Note that the secret has weight $\frac{2p_1 \cdot n}{3}$, which are exactly those 1 and -1 entries of the secret key which are not present in the given erroneous key.

For the packed format we can not similarly benefit from the asymmetry as we need to transform to unpacked format first. Thus, for the lattice approach we proceed exactly as in the symmetric setting.

However, For the combinatorial attack we can benefit from the asymmetry. Recall that for this attack we enumerate the possible errors already on the packed form. The binary representation of the packed version of f of length $n_{\mathrm{p}} := \frac{8n}{5}$ contains expected $\frac{n_{\mathrm{p}}}{2}$ entries equal to one, of which on expectation $\frac{(1-p_1)n_{\mathrm{p}}}{2}$ will still be present in the given erroneous version. Thus, we only need to enumerate the error on the $\frac{n_{\mathrm{p}}(1+p_1)}{2}$ positions defined by the zero coordinates of the candidate. Further the errors to enumerate are exactly the expected $\frac{p_1 n_{\mathrm{p}}}{2}$ ones which flipped to zero in the candidate. In turn this leads to a reduced expected list size of the attack of $\mathbb{E}[|L_1|] = \mathcal{O}\big(\binom{(1+p_1)n_{\mathrm{p}}/4}{p_1 n_{\mathrm{p}}/4}\big) = \mathcal{O}\big(\binom{2(1+p_1)n/5}{2p_1 n/5}\big)$.

Fig. 7 illustrates the achieved bit complexities for the packed and unpacked format. We observe that the possible dimension reduction in the unpacked case leads to a significant improvement of the lattice reduction. Note that the complexity converges to the bitsecurity level of the respective parameter set for $p_1 = 1$. For the packed format lattice reduction is entirely outperformed by the combinatorial attack. This is because the lattice attack does not allow for dimension reduction, while the combinatorial attack can achieve slightly more than a square root gain compared to the symmetric case.



(a) Category-1: $(n, q) = (509, 2048)$ (b) Category-3: $(n, q) = (821, 4096)$

Fig. 7: Bit complexity to recover NTRU secret key in the error model with asymmetric error $p_0 = 10^{-3}$.