Giant Does NOT Mean Strong: Cryptanalysis of BQTRU

Ali Raya[1], Vikas Kumar[2], Aditi Kar Gangopadhyay[2], and Sugata Gangopadhyay[1]

[1] Department of Computer Science and Engineering
Indian Institute of Technology Roorkee, 247667, Uttarakhand, India.
{ali_r, sugata.gangopadhyay}@cs.iitr.ac.in
[2] Department of Mathematics
Indian Institute of Technology Roorkee, 247667, Uttarakhand, India.
{v_kumar, aditi.gangopadhyay}@ma.iitr.ac.in

Abstract. NTRU-like constructions are among the most studied lattice-based schemes. The freedom of design of NTRU resulted in many variants in literature motivated by faster computations or more resistance against lattice attacks by changing the underlying algebra. To the best of our knowledge, BQTRU (DCC 2017), a noncommutative NTRU-like cryptosystem, is the fastest claimed variant of NTRU built over the quaternion algebra of the bivariate ring of polynomials. The key generation and the encryption of BQTRU are claimed to be 16/7 times faster than standard NTRU for equivalent levels of security. For key recovery attacks, the authors claim that retrieving a decryption key is equivalent to solving the Shortest Vector Problem (SVP) in expanded Euclidean lattices of giant dimensions. This work disproves this claim and proposes practical key and message recovery attacks that break the moderate parameter sets of BQTRU estimated to achieve 2^{92} message security and 2^{166} key security on a standard desktop within less than two core weeks. Furthermore, our analysis shows that the proposed parameter set for the highest security level claiming 2^{212} message security and 2^{396} key security can barely achieve 2^{82} message security and 2^{125} key security. Our work not only provides cryptanalysis for BQTRU but also demonstrates the potential of extending Gentry's attack to other rings beyond the cyclotomic polynomial ring.

Keywords: Post-quantum cryptography · Lattice · NTRU · BQTRU · Quaternion algebra

1 Introduction

NTRU [19] is one of the initial and extensively studied lattice-based post-quantum cryptosystems. It is known for its efficiency, low memory requirements, and long cryptanalytic history. Its design flexibility allows the construction of new schemes with different algebras. NTRU proposals [10, 23] that proceeded to the final round of NIST's post-quantum standardization process are built over commutative rings of polynomials. However, there is also growing interest in utilizing

noncommutative algebras within the NTRU framework motivated by improved performance or resistance against some attacks. Recently, Raya et al. [37, 38] and Kumar et al. [28, 29] designed variants of NTRU over the group rings of the noncommutative groups. The quaternion algebra has also been investigated in the context of designing noncommutative NTRU-like cryptosystems. QTRU [34] is built upon the quaternion algebra of the ring of polynomials. It is claimed to be more resistant to lattice attacks than NTRU, with a performance trade-off of 4 times slower than NTRU for equivalent security levels. Ling and Mendelsohn [32] theoretically introduced an IND-CPA secure variant of NTRU using quaternion algebra of bounded discriminant.

BQTRU [5] is a noncommutative variant of NTRU built upon the quaternion algebra of the bivariate ring of polynomials. The scheme's security is considered based on hard problems in hybrid lattices rather than the usual Euclidean lattices. Hybrid lattices are algebraic objects that provide a way to combine two mainstream post-quantum families, lattice-based and code-based cryptography. Hence, they can be used to build cryptosystems whose security relies upon hard problems in lattices and codes.

BQTRU is a timely advancement of a series of works beginning with GB-NTRU [9] that uses a *hidden ideal* of a bivariate ring of polynomials for key construction and decryption. Although deciphering is more costly, GB-NTRU claimed to improve the encryption costs compared to NTRU. Boschini et al. [8] showed that GB-NTRU can be interpreted as a cryptosystem over hybrid lattices combining Euclidean and Hamming distances. GB-NTRU was shown to be vulnerable to some algebraic attacks on messages [8]. In the same work [8], NTWO was proposed as a modification to address the vulnerabilities of GB-NTRU. Similarly to GB-NTRU, the decryption process of NTWO is costly as it involves solving a hard lattice problem. Consequently, NTWO failed to provide a practical alternative to NTRU but introduced an interesting application of hybrid lattices as a proof-of-concept. The advertised attractive feature of NTWO is its enhanced key security, which makes lattice-based key attacks almost infeasible. According to the authors, the secret key is the shortest vector in the hybrid lattice and, hence, cannot be recovered by usual lattice reduction algorithms. One can map the secret to the short vector of purely Euclidean lattice but with such an extended dimension where the complexity of lattice reduction algorithms is too high to be practical.

Motivation behind cryptanalysis. BQTRU is the blend of NTWO and QTRU. The authors claim that it inherits the amplified key security due to its hybrid lattice structure, and the quaternion algebra makes the decryption possible, which was a bottleneck in the previous designs. It is claimed to be the fastest variant of NTRU in the literature, with key generation and encryption being 16/7 times faster compared to NTRU. Additionally, the noncommutativity of BQTRU could be an extra advantage from a security standpoint, as some attacks could exploit the commutative structure. For instance, Kim and Lee [25] and Bai et al. [6] demonstrated attacks on the NTRU Learning Problem [36, Sec-

tion 4.4.4] by leveraging the commutativity to recover the secret key, as discussed and motivated in [29, 37].

Despite these promising features and theoretical soundness, BQTRU has a few shortcomings. The lack of robust security analysis and implementation raises concerns about its practical suitability, prompting us to delve into its security and implementation aspects. Our study identified technical weaknesses that render the cryptosystem susceptible to attacks. We successfully compromised the keys and messages for the proposed moderate-level security parameters and proved that the highest-level security parameters do not provide the claimed security. Interestingly, we discovered that the specific structure of the chosen quaternion algebra, intended to improve performance, actually weakens the security of the cryptosystem. Therefore, our work highlights the significance of addressing new security vulnerabilities arising from changing the underlying algebra for fast multiplications. In brief, this work provides a perspective to generalize Gentry's dimension reduction attack [17] on NTRU-composite over the commutative ring of polynomials $R' = \mathbb{Z}[x]/\langle x^N - 1\rangle$ (N composite) to the noncommutative algebra of quaternions. Gentry's approach is based on the Chinese Remainder Theorem (CRT) that factors the ring R' into polynomial rings of smaller degrees. Here, we provide a different frame of view to look for the possibilities of dimension reduction in a particular algebraic structure using the matrix representations and *folding* them to reduce dimensions. We find our method easy to implement, and the concept of dealing with matrices may extend its applicability to other algebras.

1.1 Technical Overview

Simply put, a parameter set that achieves a certain security level λ for a specific construction indicates that an adversary requires at least 2^λ operations to break the scheme using the best-known attacks. Lattice attacks are the most successful attacks against NTRU-like constructions. In the first proposal of NTRU introduced over the truncated polynomial ring $\mathbb{Z}[x]/\langle x^N - 1\rangle$ for prime N and modulus q, the public key is calculated as $h = f^{-1} * g \pmod q$ for ternary polynomials f and g with f being invertible modulo q. Key recovery attack against NTRU is mapped into finding the vector $(\boldsymbol{f}, \boldsymbol{g})$ associated with the private key or its rotations in the lattice Λ_{CS} [13], also referred to as Coppersmith and Shamir lattice, generated by the basis matrix

$$\mathcal{M}_{CS} = \begin{pmatrix} I_N & \mathcal{H} \\ 0_N & qI_N \end{pmatrix} \tag{1}$$

where $\mathcal{H}$ is a circulant matrix constructed from h; the $i-$th row of $\mathcal{H}$ is calculated as $x^i * h \mod (x^N - 1)$, for $i = 0, 1, \ldots, N-1$.

For BQTRU, the public key is calculated as $h = f^{-1} * g + v \,(\mathrm{mod}\, q)$, where f, g, and v are sampled from the quaternion algebra of the bivariate ring of polynomials denoted by $\mathbb{A}$ (as detailed in Section 3) with the following conditions:

- f, g are ternary elements (i.e., small in Euclidean norm), and f is invertible modulo q and a *private ideal* of $\mathbb{A}$.

– v has just a small number of non-zero coefficients with respect to the Lagrange basis (i.e., small in Hamming weight).

Since v does not need to have small coefficients as in f, g, the authors claim that any search attack to retrieve the decryption key (f, g, v) is extremely costly and much harder than that for the original NTRU. Additionally, lattice attacks are not as efficient as NTRU key recovery attacks since the problem of finding the decryption key is mapped into finding short vectors in expanded Euclidean lattices of extremely large dimensions.

This work disproves this claim and shows that getting a possible decryption key in BQTRU is much easier than the original NTRU for equivalent dimensions. Our key recovery attack involves two main steps: ① guessing step, ② lattice reduction step. We show that according to the procedure of the proposed key generation in BQTRU, guessing **the positions** of the non-zero coefficients of v (with respect to Lagrange basis) is enough to correctly retrieve v. After retrieving v correctly, one can compute $s = h - v \,(\mathrm{mod}\, q)$ and proceed using lattice reduction attacks against Euclidean lattices to find a short vector $(\boldsymbol{f}, \boldsymbol{g})$ as in the case of the original NTRU. It may look like finding a decryption key is at least as hard as that for the original NTRU in the same dimension, as the attacker needs to guess the positions of nonzero elements before proceeding with the lattice reduction. However, the lattice $\mathcal{L}_{CS}$ associated with BQTRU, is generated from the basis

$$\mathcal{B}_{CS} = \begin{pmatrix} I_N & \mathcal{S} \\ 0_N & qI_N \end{pmatrix} \tag{2}$$

where $N = 4n^2$ for quaternion algebras of the bivariate ring of polynomials, and $\mathcal{S}$ is the matrix corresponding to s that has a special structure:

$$\mathcal{S} = \begin{pmatrix} S_0 & S_1 & S_2 & S_3 \\ S_1 & S_0 & S_3 & S_2 \\ S_2 & -S_3 & S_0 & -S_1 \\ -S_3 & S_2 & -S_1 & S_0 \end{pmatrix}. \tag{3}$$

Our dimension reduction method, which we call *folding*, exploits the structure of $\mathcal{S}$, and define a map from $\mathcal{S}$ to a matrix of half the dimension given by

$$\phi(\mathcal{S}) \longrightarrow \begin{pmatrix} S_0 + S_1 & S_2 + S_3 \\ S_2 - S_3 & S_0 - S_1 \end{pmatrix} \tag{4}$$

that preserves the matrix addition and multiplication. Therefore, instead of looking for the short vector $(\boldsymbol{f}, \boldsymbol{g}) \in \mathcal{L}_{CS}$, we look for its image in the lattice $\mathcal{L}_{CS,\phi}$ generated by the basis

$$\mathcal{B}_{CS,\phi} = \begin{pmatrix} I_{N/2} & \phi(S) \\ 0_{N/2} & qI_{N/2} \end{pmatrix}. \tag{5}$$

Finding this image in $\mathcal{L}_{CS,\phi}$ is much easier for a lattice reduction algorithm like BKZ [39] as the dimension of the lattice is reduced by a factor of 2, while the norm of the target vector (on average) does not increase. Overall, combining the costs of the guessing step and the lattice reduction step is more beneficial for attacking the cryptosystem and retrieving the decryption key. Furthermore, the

idea of key *folding* attack is extended to the message recovery attack against the lattice generated from BQTRU's ciphertext. This work proves the efficiency of our folding attack theoretically and cross-validates the results experimentally.

1.2 Our work

Implementation of BQTRU. We provide an implementation of BQTRU for a better understanding of the practical aspects of the cryptosystem. Our findings identify some issues related to the key generation process proposed in the original work, as well as some other problems related to the decryption failure and possible alternative keys.

Efficient key and message recovery attacks. We propose an efficient key and message recovery attack against BQTRU. For a key recovery attack, instead of considering only the authors' search cost or the proposed lattice reduction in expanded lattices, we consider a combined approach that searches for some values of the key before converting the key retrieval attack into reducing a structured lattice of lower dimension that is further susceptible to dimension reduction attacks. For message recovery attacks, similarly, we show that one can benefit from the structure of the underlying ring to launch an attack in a lattice of smaller dimensions compared to the original NTRU. Using our approach, we estimate that the proposed parameter sets of BQTRU achieve much lower security levels than claimed, as shown in Figure 1 and Table 4. Further, this work experimentally breaks the moderate parameter set of claimed key security 2^{166} and message security 2^{92} just in less than 12 core days (on average) for key and message attacks on a standard desktop.

How NOT to fix BQTRU. We provide our attempts to fix the BQTRU cryptosystem against the proposed attacks, especially the key recovery attack. These attempts originated from the trials to change the key generation process to make the guessing part harder for an attacker. However, we managed to show that an extension of the proposed attack can be applied even against the new proposal. We consider these attempts unsuccessful and advise against using them in future efforts to fix BQTRU.

Our artifacts for the experiment with a detailed documentation can be accessed at `https://github.com/Mr-PQ-Crypto/BQTRU_cryptanalysis`.

1.3 Road map

Section 2 introduces the preliminaries and notations and briefly discusses the lattice attacks on NTRU-like constructions. Section 3 describes BQTRU and its relation to hybrid lattices. Section 4 is about the security claims by authors of BQTRU regarding keys and messages. In Section 5, we propose our key and message attacks, while Section 6 experimentally verifies the correctness of our attacks along with a cost analysis. Section 7 gives a few of our unsuccessful attempts to fix BQTRU. We conclude our work in Section 8.

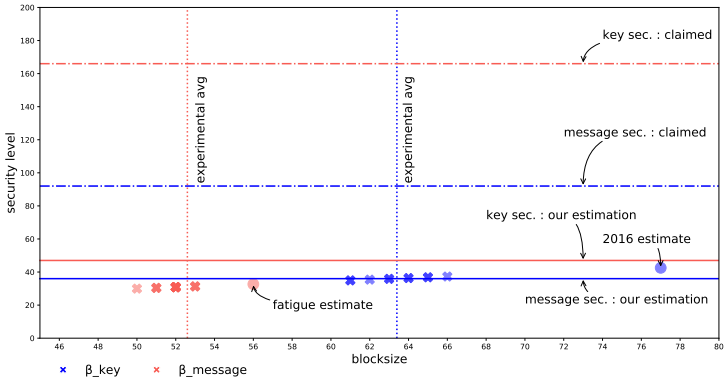

Fig. 1: The claimed vs. our estimation of the security level for the moderate parameter set of BQTRU ($n = 7$ and $q = 113$). The original proposal of BQTRU estimates the key security to be greater than $2^{166}-$bit and the message security to be $2^{92}-$bit. On the other hand, our estimation shows that the key security and the message security provide no more than 2^{47} and $2^{37}-$bit security level and hence can be broken on a standard desktop within 2 core weeks, approximately. Although the experimentally obtained blocksizes for the key attack are lower than for the message attack, our *worst-case* estimate for the key security is higher than the message as the key attack involves guessing and lattice reduction, whereas the message is straightforward based on lattice reduction.

2 Preliminaries

2.1 Notations

- $\mathbb{Z}, \mathbb{R}$ denote the set of integers and real numbers, respectively.
- For a positive integer q, $\mathbb{Z}_q$ is the ring of integers modulo q, and $\mathbb{Z}_q^\star$ is the group of units, i.e., group of invertible elements in $\mathbb{Z}_q$.
- For a set A, $|A|$ is the cardinality of A, and $a \xleftarrow{\$} A$ denotes sampling an element a uniformly at random from A.
- For any ring R and a positive integer m, $R^m = \{(a_1, a_2, \ldots, a_m) : a_i \in R\}$, and $M_m(R)$ denotes the ring of $m \times m$ matrices with entries from R.
- The symbols $*$ and $\star$, respectively, denote the multiplication of elements of the underlying algebraic structure and the multiplication of their associated vectors. The underlying algebras should be clear from the context. Further, $\star$ is also used to denote matrix multiplication.
- The symbol $\otimes$ denotes the Kronecker product of matrices.
- For a vector $\boldsymbol{v} = (v_1, v_2, \ldots, v_m) \in \mathbb{R}^m$, the Euclidean norm of $\boldsymbol{v}$ is defined as $||\boldsymbol{v}|| = \sqrt{\sum_{i=1}^{m} v_i^2}$, and for a vector $\boldsymbol{v} = (v_1, v_2, \ldots, v_m) \in \mathbb{Z}_q^m$, the Hamming norm of $\boldsymbol{v}$ is defined as $||\boldsymbol{v}||_H = |\{i : v_i \neq 0\}|$.
- $A \cong B$ denotes that two algebraic structures A and B are isomorphic to each other.

2.2 Lattices: definitions and reductions

Definition 1 (Lattice). *Let B be a $d \times m$ matrix with d linearly independent rows $\{\boldsymbol{b}_0, \boldsymbol{b}_1, \ldots, \boldsymbol{b}_{d-1}\} \subset \mathbb{R}^m$. The lattice generated by B is defined as*

$$\mathcal{L}_B = \mathbb{Z}^d B = \left\{ \sum_{i=0}^{d-1} \gamma_i \boldsymbol{b_i} : \gamma_i \in \mathbb{Z} \right\}. \tag{6}$$

The matrix B is called the basis matrix for the lattice $\mathcal{L}_B$. Here, d, i.e., the number of linearly independent rows in the basis matrix, is called the *rank*, and m is called the *dimension* of $\mathcal{L}_B$. The lattice is referred to as *full-rank* if $d = m$. If $\boldsymbol{b}_i \in \mathbb{Z}^m$, we call the lattice to be an *integral lattice*. For this work, we consider full-rank integral lattices. The volume of a lattice $\mathcal{L}_B$ defined by a basis matrix B is given by $vol(\mathcal{L}_B) = \sqrt{|det(BB^{Tr})|}$, and it is independent of the choice of basis. For $i \in \{0, 1, \ldots, d-1\}$, define π_i to be the projection on the space orthogonal to the span of $\{\boldsymbol{b}_0, \ldots \boldsymbol{b}_{i-1}\}$, and denote Gram-Schmidt basis as $\{\boldsymbol{b}_0^\star, \boldsymbol{b}_1^\star, \ldots \boldsymbol{b}_{d-1}^\star\}$, where $\boldsymbol{b}_i^\star = \pi_i(\boldsymbol{b}_i)$. We refer to the lattice generated from $\{\pi_\ell(\boldsymbol{b}_\ell), \ldots, \pi_\ell(\boldsymbol{b}_{r-1})\}$ as the projected sublattice and denote it by $\mathcal{L}_{B[\ell,r)}$. We refer to the lengths of $||\boldsymbol{b}_i^\star||$ for $i \in \{0, 1, \ldots, d-1\}$ as the profile of the basis B.

Definition 2 ($q-$ary lattice). *A lattice of dimension d is called $q-$ary lattice if $q\mathbb{Z}^d \subset \mathcal{L}_B$ for some $q > 0$.*

Definition 3 (Minimum length). *The minimum length $\lambda_1(\mathcal{L})$ of a lattice $\mathcal{L}$ is defined as the length of its shortest nonzero vector, i.e., $\lambda_1(\mathcal{L}) = \min_{\boldsymbol{v} \in \mathcal{L} - \{0\}} ||\boldsymbol{v}||$.*

Definition 4 (Gaussian heuristics). *Given a random $d-$dimensional lattice $\mathcal{L}_B$ defined by basis B, Gaussian heuristic estimates that the expected length of the shortest nonzero vector in $\mathcal{L}_B$ is*

$$\lambda_1(\mathcal{L}_B) \approx \sqrt{\frac{d}{2\pi e}} vol(\mathcal{L}_B)^{\frac{1}{d}}. \tag{7}$$

Definition 5 (Hard lattice problems). *Let $\mathcal{L}_B \subset \mathbb{R}^d$ be a full-rank lattice defined by the basis B.*

1. *Shortest Vector Problem (SVP): Find a nonzero vector $\boldsymbol{v} \in \mathcal{L}_B$ such that $||\boldsymbol{v}|| = \lambda_1(\mathcal{L}_B)$.*
2. *Closest Vector Problem (CVP): Find a vector $\boldsymbol{v} \in \mathcal{L}_B$ closest to the given target vector $\boldsymbol{t} \in \mathbb{R}^d$, i.e., $||\boldsymbol{v} - \boldsymbol{t}|| \leq ||\boldsymbol{w} - \boldsymbol{t}||$ for all $\boldsymbol{w} \in \mathcal{L}_B$. Further, when $||\boldsymbol{v} - \boldsymbol{t}|| < \alpha\lambda_1(\mathcal{L}_B)$ for some $\alpha < 1$, the problem is referred to as the Bounded Distance Decoding (BDD) problem.*

The Kannan embedding technique [24] transforms the problem of solving the Closest Vector Problem (CVP) in a $d-$dimensional lattice into solving the Shortest Vector Problem (SVP) in a $(d+1)-$dimensional lattice. For instance, finding the closest vector in the lattice $\mathcal{L}_B$ (generated by basis B) to the target vector $\boldsymbol{t} \in \mathbb{R}^d$ can be converted into solving the SVP in the lattice generated by the basis

$$B' = \begin{pmatrix} B & 0 \\ \boldsymbol{t} & u \end{pmatrix} \tag{8}$$

where $u \in \mathbb{R}$ is the embedding factor (usually 1).

2.3 Lattice reduction

There are infinitely many bases to define a lattice of dimension ≥ 2. From the attacker's perspective, some bases are more friendly to launch lattice attacks against the public key and, therefore, described as *'good'* basis. Compared to *'bad'* basis, good ones are defined with a set of reasonably short and almost orthogonal vectors. Given a publicly available bad basis, a lattice reduction algorithm tries to find a good basis that defines the same lattice. LLL [31] is a famous example of a polynomial-time basis reduction algorithm that produces a reasonably reduced good basis for low dimensions. Although LLL runs in polynomial time, the quality of the reduced basis degrades as the dimension of the lattice increases. BKZ [39] can be thought of as a generalization of LLL that considers an additional parameter: the blocksize or β.

Definition 6 (BKZ). *A basis $B = \{\boldsymbol{b}_0, \boldsymbol{b}_1, \ldots, \boldsymbol{b}_{d-1}\}$ is called BKZ$-\beta$ reduced if*

$$\left\|\boldsymbol{b}_\kappa^\star\right\| = \lambda_1(\mathcal{L}_{B[\kappa:min(\kappa+\beta,d))}) \quad \textit{for all} \quad \kappa = 0, \ldots, d-1.$$

For each $\kappa \in \{0, 1, \ldots, d-1\}$, the BKZ algorithm calls internally an SVP oracle to find the shortest vector in the projected sublattice $\mathcal{L}_{B[\kappa:min(\kappa+\beta,d))}$. Repeating this process for all the indices is called a BKZ tour, and the algorithm keeps on applying tours until the condition is satisfied for all the positions. The most expensive part of generating the BKZ$-\beta$ reduced basis is due to calling the SVP oracle while the number of the tours is polynomially bounded. Enumeration [16, 35] and Sieving [7, 18] are commonly used techniques in the oracle. Although the memory requirements for enumeration are polynomial, the running time is super-exponential in the blocksize β. On the other hand, memory and time requirements for sieving are both exponential in β.

Several improvements have been introduced to BKZ, resulting in BKZ2.0 [12] and progressive BKZ [4]. Progressive BKZ reduces the running time in practice while instead of running many tours for a blocksize β, the algorithm applies a few tours for increasing blocksizes up to β. Generally, the quality of a BKZ$-\beta$ reduced basis is measured by a quantity called the root Hermite factor.

Definition 7 (Root Hermite factor). *For a $d-$dimensional lattice $\mathcal{L}_B$, the root Hermite factor is defined as*

$$\delta_\beta = \left(\|b_1\| / vol(\mathcal{L}_B)^{1/d}\right)^{1/d}. \tag{9}$$

For a small blocksize β, the root Hermite factor can be computed experimentally, while Chen [11] showed that for reasonably large $\beta > 50$, δ_β can be estimated as

$$\delta_\beta = \left(\frac{\beta}{2\pi e}(\pi\beta)^{\frac{1}{\beta}}\right)^{\frac{1}{2(\beta-1)}}. \tag{10}$$

This leads to the Geometry Series Assumption(GSA) that heuristically estimates the profile for a BKZ$-\beta$ reduced basis.

Definition 8 (Geometry Series Assumption). *Let $B = \{\boldsymbol{b}_0, \boldsymbol{b}_1, \ldots, \boldsymbol{b}_{d-1}\}$ be a BKZ$-\beta$ reduced basis, then the GSA estimates that $\left\|\boldsymbol{b}_i^\star\right\| \approx {\delta_\beta}^{-2}\left\|\boldsymbol{b}_{i-1}^\star\right\|$.*

The accuracy of the GSA is observed for sufficiently large blocksizes ($\beta > 50$ and $\beta \ll d$) when BKZ is applied to random lattices.

2.4 Lattice attack against NTRU-like constructions

Since its introduction in 1998, several versions of NTRU have emerged in the literature. NTRU is now recognized as a hard problem in cryptography rather than a unique cryptosystem that can be extended to different algebraic structures. The NTRU design and the problem can be outlined as:

Definition 9 (NTRU). *Let N be a prime, q be a positive integer, and $f, g \in \mathbb{Z}[x]/\langle x^N - 1\rangle$ be two polynomials with small coefficients (mostly ternary) such that f is invertible modulo q. The pair (f, g) forms the secret key and $h = f^{-1} * g \,(\mathrm{mod}\, q) \in \mathbb{Z}_q[x]/\langle x^N - 1\rangle$ is the public key. The NTRU problem asks to find the private key or its rotations $(x^i * f, x^i * g)$.*

As discussed earlier, the most renowned technique to attack the NTRU problem is to solve SVP in the $2N-$dimensional lattice Λ_{CS} generated by the basis

$$\mathcal{M}_{CS} = \begin{pmatrix} I_N & \mathcal{H} \\ 0_N & qI_N \end{pmatrix}, \tag{11}$$

since the vector $(\boldsymbol{f}, \boldsymbol{g})$ associated with the private key (f, g) or its rotations are the shortest vectors in the lattice Λ_{CS} with high probability.

Gentry attack. The selection of N as prime is crucial to NTRU construction over the ring $\mathbb{Z}_q[x]/\langle x^N - 1\rangle$. For example, Silverman [40] proposed a variant of NTRU where N was selected to be a power of 2 to enable Fast Fourier Transformations (FFTs) for fast polynomial multiplications. However, Gentry [17] used the Chinese Remainder Theorem (CRT) to demonstrate that for composite values of N, the ring $\mathbb{Z}_q[x]/\langle x^N - 1\rangle$ can be factored into polynomial rings with smaller degree such that the coefficients of the polynomials under this factoring map do not grow much. In particular, for even N, we have the following isomorphism:

$$\begin{gathered} \frac{\mathbb{Z}_q[x]}{\langle x^N - 1\rangle} \to \frac{\mathbb{Z}_q[x]}{\langle x^{N/2} - 1\rangle} \times \frac{\mathbb{Z}_q[x]}{\langle x^{N/2} + 1\rangle} \\ u \to (u_0 + u_1, u_0 - u_1) \end{gathered} \tag{12}$$

for every $u = (u_0, u_1) \in \mathbb{Z}_q[x]/\langle x^N - 1\rangle$. Consequently, the secret vector $(\boldsymbol{f}, \boldsymbol{g}) \in \Lambda_{CS}$ is mapped to the short vectors $(\boldsymbol{f}_0 + \boldsymbol{f}_1, \boldsymbol{g}_0 + \boldsymbol{g}_1) \in \Lambda^+_{CS}$ and $(\boldsymbol{f}_0 - \boldsymbol{f}_1, \boldsymbol{g}_0 - \boldsymbol{g}_1) \in \Lambda^-_{CS}$, where Λ^+_{CS} and Λ^-_{CS} are the $N-$dimensional lattices generated by the matrices

$$\mathcal{M}^+_{CS} = \begin{pmatrix} I_{N/2} & \mathcal{H}^+ \\ 0_{N/2} & qI_{N/2} \end{pmatrix} \quad \text{and} \quad \mathcal{M}^-_{CS} = \begin{pmatrix} I_{N/2} & \mathcal{H}^- \\ 0_{N/2} & qI_{N/2} \end{pmatrix}. \tag{13}$$

Here, $\mathcal{H}^+$ is the matrix representation of the image of the public key $h_0 + h_1 \in \mathbb{Z}_q[x]/\langle x^{N/2} - 1\rangle$ whose $i-$th row is defined by $x^i * (h_0 + h_1) \bmod (x^{N/2} - 1)$ and $\mathcal{H}^-$ is the matrix representation of $h_0 - h_1 \in \mathbb{Z}_q[x]/\langle x^{N/2} + 1\rangle$ whose $i-$th row is defined by $x^i * (h_0 - h_1) \bmod (x^{N/2} + 1)$. This way, Gentry exploited the special structure of the underlying algebra to reduce the dimension of the lattice

to be attacked by half.

Different perspective. We look at Gentry's dimension reduction from the perspective of matrices. For even N, the matrix of the public key h is of a particular form

$$\mathcal{H} = \begin{pmatrix} H_0 & H_1 \\ H_1 & H_0 \end{pmatrix} \in M_N(\mathbb{Z}_q). \tag{14}$$

The effect of the Chinese Remainder Theorem in Gentry's method can be captured in the matrix ring homomorphisms $\mathcal{H} \to \mathcal{H}^+ = H_0 + H_1$ and $\mathcal{H} \to \mathcal{H}^- = H_0 - H_1 \in M_{N/2}(\mathbb{Z}_q)$. These homomorphisms allow mapping the public key equation $\boldsymbol{f} \star \mathcal{H} = \boldsymbol{g} (\mathrm{mod}\, q)$ to $(\boldsymbol{f}_0 + \boldsymbol{f}_1) \star \mathcal{H}^+ = \boldsymbol{g}_0 + \boldsymbol{g}_1 (\mathrm{mod}\, q)$ and $(\boldsymbol{f}_0 - \boldsymbol{f}_1) \star \mathcal{H}^- = \boldsymbol{g}_0 - \boldsymbol{g}_1 (\mathrm{mod}\ q)$. As a result, the vectors $(\boldsymbol{f}_0 + \boldsymbol{f}_1, \boldsymbol{g}_0 + \boldsymbol{g}_1) \in \Lambda^+_{CS}$ and $(\boldsymbol{f}_0 - \boldsymbol{f}_1, \boldsymbol{g}_0 - \boldsymbol{g}_1) \in \Lambda^-_{CS}$, which is the same scenario as for Gentry.

We find our description of Gentry's attack in terms of matrices suitable to those algebras whose matrix representations possess special structures and can be reduced homomorphically; however, it is difficult to define algebra linked to the matrices of reduced dimensions. We believe that our approach extends the possible applicability of Gentry's attack to different rings. In this work, we demonstrate an application of our dimension reduction technique based on matrices on BQTRU, which is built over the quaternion algebra of a bivariate ring of polynomials.

3 BQTRU

In this section, we give a detailed description of BQTRU and its relation to the hybrid lattices. For more insights, we refer the readers to [5].

Definition 10 (Quaternion algebra). *[5, Definition 1] The quaternion algebra $\mathbb{A}$ over a field $\mathbb{F}$ by two nonzero elements $a, b \in \mathbb{F}$ is a 4-dimensional vector space generated with basis $1, i, j,$ and k, defined as*

$$\left(\frac{a, b}{\mathbb{F}}\right) = \{f_0 + f_1 i + f_2 j + f_3 k : f_i \in \mathbb{F}\}. \tag{15}$$

The bilinear multiplication is defined by the conditions that 1 is the unity element and

$$i^2 = a, j^2 = b, ij = -ji = k. \tag{16}$$

Consequently,

$$k^2 = -ab, jk = -kj = -ib, ki = -ik = -ja. \tag{17}$$

For an element $f = f_0 + f_1 i + f_2 j + f_3 k \in \mathbb{A}$, the conjugate of f is given by $\bar{f} = f_0 - f_1 i - f_2 j - f_3 k$, and norm of f is $N(f) = f * \bar{f} = \bar{f} * f = f_0^2 - a f_1^2 - b f_2^2 + ab f_3^2$. A quaternion f is invertible if and only if $N(f)$ is nonzero, and in that case, the inverse of f is given by $f^{-1} = N(f)^{-1} * \bar{f}$. A quaternion with norm 1 is called a unit quaternion. It is known that for any field $\mathbb{F}$, the quaternion algebra $\left(\frac{1,1}{\mathbb{F}}\right) \cong M_2(\mathbb{F})$ [33], where the isomorphism $\psi : \left(\frac{1,1}{\mathbb{F}}\right) \to M_2(\mathbb{F})$ is given by

$$\psi(f_0 + f_1 i + f_2 j + f_3 k) = \begin{pmatrix} f_0 + f_1 & f_2 + f_3 \\ f_2 - f_3 & f_0 - f_1 \end{pmatrix}. \tag{18}$$

In general, the quaternion algebra $\mathbb{A}$ can be defined over any commutative ring with unity R, i.e., $\mathbb{A} = \left(\frac{a,b}{R}\right)$ where a, b are nonzero elements of the ring R. For the rings R, considered in this paper, we have $\left(\frac{a,b}{R}\right) \cong M_2(R)$ with the same isomorphism as defined in (18).

Setup. The parameters (n, p, q) are chosen such that n, p, and q are primes with $p \ll q$, and let $d_f = d_g = d_r = d_m = \lfloor n^2/7 \rfloor$, where $\lfloor \cdot \rfloor$ is the greatest integer function. Let $\mathcal{T}(d_1, d_2)$ be a subset of R consisting of ternary elements with d_1 coefficients equal to 1, d_2 coefficients equal to -1, other coefficients equal to 0. Define $L_f = L_g = L_r = L_m = \mathcal{T}(\lfloor n^2/7 \rfloor, \lfloor n^2/7 \rfloor)$. BQTRU operates on the quaternion algebras of the bivariate ring of polynomials

$$\mathbb{A} = \left(\frac{1,1}{R}\right), \ \mathbb{A}_p = \left(\frac{1,1}{R_p}\right), \text{ and } \mathbb{A}_q = \left(\frac{1,1}{R_q}\right),$$

where,

$$R = \frac{\mathbb{Z}[x,y]}{\langle x^n - 1, y^n - 1\rangle}, \ R_p = \frac{\mathbb{Z}_p[x,y]}{\langle x^n - 1, y^n - 1\rangle}, \text{ and } R_q = \frac{\mathbb{Z}_q[x,y]}{\langle x^n - 1, y^n - 1\rangle}.$$

Every element $v(x,y) = v_0 + v_1 x + \ldots + v_{n-1}x^{n-1} + v_n y + v_{n+1}yx + \ldots + v_{2n-1}yx^{n-1} + \ldots + v_{n^2-n}y^{n-1} + v_{n^2-n+1}y^{n-1}x + \ldots + v_{n^2-1}y^{n-1}x^{n-1} \in R$ can be uniquely mapped to its coefficient vector

$$\boldsymbol{v} = (v_0, v_1, \ldots, v_{n^2-1}) \in \mathbb{Z}^{n^2}.$$

Therefore, considering the monomial basis, R is isomorphic to $\mathbb{Z}^{n^2}$ as an additive module over $\mathbb{Z}$. Similarly, $\mathbb{A} \cong \mathbb{Z}^{4n^2}$ as every quaternion $f = f_0 + f_1 i + f_2 j + f_3 k \in \mathbb{A}$ can be mapped uniquely to its coefficient vector

$$\boldsymbol{f} = (\boldsymbol{f}_0, \boldsymbol{f}_1, \boldsymbol{f}_2, \boldsymbol{f}_3) \in \mathbb{Z}^{4n^2},$$

where $\boldsymbol{f}_i \in \mathbb{Z}^{n^2}$ is the coefficient vector of f_i for $i = 0, 1, 2, 3$. For two quaternions $f, g \in \mathbb{A}$, the coefficient vectors of $f + g$ and $f * g$ are denoted by $\boldsymbol{f} + \boldsymbol{g}$ and $\boldsymbol{f} \star \boldsymbol{g}$, respectively.

In addition, n is chosen such that $n|q-1$ so that $\mathbb{Z}_q$ contains the nth roots of unity. Let $E = \{(a,b) \in \mathbb{Z}_q \times \mathbb{Z}_q : a^n = b^n = 1\}$, then $|E| = n^2$. The ring R_q is an n^2-dimensional vector space over the field $\mathbb{Z}_q$ with Lagrange basis $\{\lambda_{a,b}(x,y) : (a,b) \in E\}$, where $\lambda_{a,b}(x,y)$ are the Lagrange interpolants given by

$$\lambda_{a,b}(x,y) = \frac{ab(x^n - 1)(y^n - 1)}{n^2(x-a)(y-b)}. \tag{19}$$

Every polynomial $f(x,y) \in R_q$ can be expressed uniquely as the linear combination of Lagrange basis as

$$f(x,y) = \sum_{(a,b)\in E} f(a,b)\lambda_{a,b}(x,y). \tag{20}$$

Let T be a non-empty subset of E and Q_q be the ideal of R_q consisting of all polynomials vanishing outside T. The ideal Q_q is also a vector subspace of R_q

generated by the basis elements corresponding to $T \cap E$. The set T is chosen to be small for cryptographic purposes, particularly to facilitate correct decryption. Without loss of generality, assume that T consists of the first $|T|$ elements of E. As an ideal, Q_q can be generated by a polynomial

$$\sigma(x, y) = \sum_{i=1}^{|T|} q_i \lambda_{a_i, b_i}(x, y), \tag{21}$$

where q_i are randomly chosen nonzero elements in $\mathbb{Z}_q$. Let $Q = \langle q, \sigma(x, y) \rangle_R$ be the ideal of R and $J = Q + Qi + Qj + Qk = \langle q, \sigma(x, y) \rangle_{\mathbb{A}}$ be the ideal of $\mathbb{A}$. The ideal J is called the *private ideal* and is used for key generation and decryption. Since Q is an ideal of R and hence an additive subgroup. Therefore, Q can be viewed as an n^2-dimensional lattice $\mathcal{L}_Q$ in $\mathbb{Z}^{n^2}$. Similarly, $\mathbb{A} \cong \mathbb{Z}^{n^2} + \mathbb{Z}^{n^2} i + \mathbb{Z}^{n^2} j + \mathbb{Z}^{n^2} k \cong \mathbb{Z}^{4n^2}$ and the ideal J can be viewed as a $4n^2-$dimensional lattice in $\mathbb{Z}^{4n^2}$. This lattice is called *private lattice* and is denoted by $\mathcal{L}_{private}$.

We discuss the method given in [5] to construct the generator matrix D' of the lattice $\mathcal{L}_Q$. First, consider a matrix L whose rows are the coefficients vectors of the Lagrange interpolants $\{\lambda_{a_i, b_i}(x, y) : i = 1, 2, \ldots, |T|\}$. Since the row rank of a matrix is equal to the column rank, suppose that $i_1, i_2, \ldots, i_{|T|}$ columns in L are linearly independent. Let $\{j_1, j_2, \ldots, j_{n^2 - |T|}\} = \{1, 2, \ldots, n^2\} \backslash \{i_1, i_2, \ldots, i_{|T|}\}$. Then the matrix D' whose rows are the coefficient vectors of Lagrange interpolants $\{\lambda_{a_i, b_i}(x, y)\}_{i=1}^{|T|}$ and $qe_{j_1}, qe_{j_2}, \ldots, e_{j_{n^2 - |T|}}$ forms a basis of $\mathcal{L}_Q$ with a high probability. Further, since $J \cong Q^4$, the private lattice $\mathcal{L}_{private}$ is generated by the basis matrix

$$\mathcal{B}_{private} = \begin{pmatrix} D' & 0 & 0 & 0 \\ 0 & D' & 0 & 0 \\ 0 & 0 & D' & 0 \\ 0 & 0 & 0 & D' \end{pmatrix}. \tag{22}$$

Key generation. Two quaternions $f = f_0 + f_1 i + f_2 j + f_3 k$ and $g = g_0 + g_1 i + g_2 j + g_3 k$ are randomly chosen such that $f_0 \in \mathcal{T}(d_f + 1, d_f)$, $f_i \in L_f$ for $i = 1, 2, 3$, and $g_i \in L_g$, for $i = 0, 1, 2, 3$. Since f_i, g_i are ternary polynomials, therefore, the Euclidean norm of coefficients vectors $\boldsymbol{f} = (\boldsymbol{f}_0, \boldsymbol{f}_1, \boldsymbol{f}_2, \boldsymbol{f}_3)$ and $\boldsymbol{g} = (\boldsymbol{g}_0, \boldsymbol{g}_1, \boldsymbol{g}_2, \boldsymbol{g}_3)$ is small. Further, f and g are chosen to be invertible in $\mathbb{A}$ modulo the private ideal J, i.e., there are elements $f^{-1}, g^{-1} \in \mathbb{A}$ such that

$$f * f^{-1} = f^{-1} * f = 1 \,(\mathrm{mod}\, J)$$
$$g * g^{-1} = g^{-1} * g = 1 \,(\mathrm{mod}\, J).$$

Additionally, f must also be invertible in $\mathbb{A}_p$, i.e., there is an element $f_p^{-1} \in \mathbb{A}_p$ such that $f * f_p^{-1} = f_p^{-1} * f = 1 (\mathrm{mod}\ p)$. For f and g to be invertible in $\mathbb{A}/J$, their norms $N(f), N(g)$ must be invertible in R/Q. Similarly, for f to be invertible in $\mathbb{A}_p$, $N(f)$ must be invertible in R_p. Here, we describe the technique given in [5] to generate such f and g. First, randomly choose $g = g_0 + g_1 i + g_2 j + g_3 k$ where $g_i \in L_g$ and define the set

$$T = \cap_{i=0}^{3} \{(a, b) \in E : g_i(a, b) = 0\}.$$

If T is empty, choose another g. Then, randomly choose f such that $f_{set} = \{(a,b) \in E : N(f)(a,b) = 0\} \subseteq T$. This gives the required f and g since $N(f), N(g)$ are invertible in R/Q if and only if the roots of $N(f), N(g)$ in E are also contained in T. However, this method has some issues, which we will discuss later. Finally, to construct the public key, a quaternion $w \in \mathbb{L}_q = \left(\frac{1,1}{\mathbb{Z}_q}\right)$ is chosen such that w is invertible in $\mathbb{L}_q$, and the public key is computed as

$$h = f^{-1} * g + v \,(\mathrm{mod}\, q) \tag{23}$$

where $v = w * \sigma \pmod q$ is kept private. The above-discussed key generation process (as given on [5, Page 11]) is compiled in Algorithm 1.

Algorithm 1: Key generation

for $i \leftarrow 0$ *to* 3 **do** $g_i \xleftarrow{\$} L_g$
$g \leftarrow g_0 + g_1 i + g_2 j + g_3 k$
$T \leftarrow \cap_{i=0}^{3} \{(a,b) \in E : g_i(a,b) = 0\}$
if T *is empty* **then** `go to step 2`
for $(a_i, b_i) \in T$ **do** $q_i \xleftarrow{\$} \mathbb{Z}_q^\star$
$\sigma \leftarrow \sum_{(a_i,b_i)\in T} q_i \lambda_{a_i,b_i}(x,y)$
$w \xleftarrow{\$} \mathbb{L}_q^\star$ `/* ` $\mathbb{L}_q^\star :=$ `set of invertible elements in` $\mathbb{L}_q$ `*/`
$v \leftarrow w * \sigma \,(\mathrm{mod}\, q)$
$f_0 \xleftarrow{\$} \mathcal{T}(d_f + 1, d_f)$
for $i \leftarrow 1$ *to* 3 **do** $f_i \xleftarrow{\$} L_f$
$f \leftarrow f_0 + f_1 i + f_2 j + f_3 k$
$f_{set} \leftarrow \{(a,b) \in E : N(f)(a,b) = 0\}$
if $f_{set} \subseteq T$ **then**
  return `Public Key:` $h = f^{-1} * g + v \,(\mathrm{mod}\, q)$
  `Private key:` f, g, v
else `go to step 9`

Encryption. To encrypt a message $m = m_0 + m_1 i + m_2 j + m_3 k$ where $m_i \in L_m$, a random quaternion $r = r_0 + r_1 i + r_2 j + r_3 k \in \mathbb{A}$ is chosen such that $r_i \in L_r$. Then, the ciphertext is given by $c = ph * r + m \,(\mathrm{mod}\, q)$.

Decryption. In order to decrypt c, first compute $a = f * c \,(\mathrm{mod}\, q)$. Then, find the closest vector to $\boldsymbol{a}$ in the private lattice $\mathcal{L}_{private}$, call it $\boldsymbol{b}$, and let $v = a - b$. The receiver then recovers the message by $m = f_p^{-1} * v \,(\mathrm{mod}\, p)$.

Correctness of decryption. We have $f * f^{-1} = f^{-1} * f = 1 + \alpha * \sigma \,(\mathrm{mod}\, q)$, for some $\alpha \in \mathbb{A}$. Therefore,

$$\begin{aligned} f * h &= (f * f^{-1}) * g + f * v \,(\mathrm{mod}\, q) \\ &= g + \alpha * \sigma * g + f * v \,(\mathrm{mod}\, q) \\ &= g + \gamma \,(\mathrm{mod}\, q) \end{aligned}$$

where $\gamma = \alpha * \sigma * g + f * v \,(\mathrm{mod}\, q)$. Receiver on computing $a = f * c \,(\mathrm{mod}\, q)$ gets

$$\begin{aligned} f * c &= f * (ph * r + m) \,(\mathrm{mod}\, q) \\ &= pg * r + p\gamma * r + f * m \,(\mathrm{mod}\, q) \\ &= pg * r + f * m + (p\gamma * r + \varepsilon q) \\ &= pg * r + f * m + b \end{aligned}$$

where $\varepsilon \in \mathbb{A}$ and $b = (p\alpha * g * r + pf * w * r) * \sigma + \varepsilon q \in J$. The element b is unknown to the receiver and must be found to decrypt the ciphertext. Since f, g, r, and m are ternary polynomials, therefore, the norm of vector $p\boldsymbol{g}\star\boldsymbol{r}+\boldsymbol{f}\star\boldsymbol{m}$ is small compared to norm of $\boldsymbol{b} \in \mathcal{L}_{private}$. Hence, the vector $\boldsymbol{b} \in \mathcal{L}_{private}$ is closest to $\boldsymbol{f}\star\boldsymbol{c}$ that the receiver recovers by solving 4 instances of CVP in n^2-dimensional lattice $\mathcal{L}_Q$. Subtracting $\boldsymbol{b}$ from $\boldsymbol{f}\star\boldsymbol{c}$ gives $p\boldsymbol{g}\star\boldsymbol{r}+\boldsymbol{f}\star\boldsymbol{m}$. Therefore, for the correct choice of parameters, $\boldsymbol{m}$ can be recovered similarly to NTRU as $\boldsymbol{m} = \boldsymbol{f}_p^{-1} \star (p\boldsymbol{g}\star\boldsymbol{r}+\boldsymbol{f}\star\boldsymbol{m}) \,(\mathrm{mod}\, p)$. For more details, refer to [5].

4 Claimed Security

4.1 Key security

Lemma 1. *[5, Lemma 1] Let $\rho : \mathbb{A}_q \to (\mathbb{L}_q)^{n^2}$ be a map defined as*

$$\rho(f) = (f(a_1, b_1), f(a_2, b_2), \ldots, f(a_{n^2}, b_{n^2})), \tag{24}$$

where $(a_i, b_i) \in E$ and $\mathbb{L}_q = \left(\frac{1,1}{\mathbb{Z}_q}\right)$ is the quaternion algebra over $\mathbb{Z}_q$. Then, the following properties hold

$$\rho(f + g) = \rho(f) + \rho(g) \text{ and } \rho(f * g) = \rho(f) * \rho(g), \tag{25}$$

where

$$\rho(f) * \rho(g) = (f(a_1, b_1) * g(a_1, b_1), \ldots, f(a_{n^2}, b_{n^2}) * g(a_{n^2}, b_{n^2})).$$

For an element $f = f_0+f_1i+f_2j+f_3k \in \mathbb{A}_q$, $\rho(f) = \rho(f_0)+\rho(f_1)i+\rho(f_2)j+\rho(f_3)k$ and the associated vector is given by

$$\boldsymbol{\rho}(\boldsymbol{f}) = (\rho(f_0), \rho(f_1), \rho(f_2), \rho(f_3)) \in \mathbb{Z}_q^{4n^2}.$$

Further, every quaternion $f = f_0 + f_1 i + f_2 j + f_3 k \in \mathbb{A}$ is associated with its unique matrix representations

$$\mathcal{F} = \begin{pmatrix} F_0 & F_1 & F_2 & F_3 \\ F_1 & F_0 & F_3 & F_2 \\ F_2 & -F_3 & F_0 & -F_1 \\ -F_3 & F_2 & -F_1 & F_0 \end{pmatrix}, \; \tilde{\mathcal{F}} = \begin{pmatrix} F_0 & F_1 & F_2 & F_3 \\ F_1 & F_0 & -F_3 & -F_2 \\ F_2 & F_3 & F_0 & F_1 \\ -F_3 & -F_2 & F_1 & F_0 \end{pmatrix} \in M_{4n^2}(\mathbb{Z}), \tag{26}$$

where $F_i \in M_{n^2}(\mathbb{Z})$ is the matrix representation of $f_i \in R$, such that for every $g \in \mathbb{A}$,

$$\boldsymbol{f}\star\boldsymbol{g} = \boldsymbol{f}\star\mathcal{G} = \boldsymbol{g}\star\tilde{\mathcal{F}}, \tag{27}$$

where $\mathcal{G}$ is the matrix representation of quaternion g. We refer the readers to Appendices A, B for more details on the matrix representations of elements in the ring R and the quaternion algebra $\mathbb{A}$.

Theorem 1. *[5, Proposition 1] Suppose that f, g, γ, and h be the private and public BQTRU keys with a quaternion $u \in \mathbb{A}$ such that*

$$f * h = g + \gamma + qu. \tag{28}$$

Then, the vector $(\boldsymbol{g}, \boldsymbol{f}, -\boldsymbol{\rho}(\boldsymbol{\gamma}))$ belongs to the BQTRU lattice $\mathcal{L}_{BQTRU}$ generated by the basis matrix

$$\mathcal{B}_{BQTRU} = \begin{pmatrix} qI_{4n^2} & 0 & 0 \\ \mathcal{H} & I_{4n^2} & 0 \\ \mathcal{D} & 0 & I_{4n^2} \end{pmatrix} \tag{29}$$

where, $\mathcal{H}$ is the matrix representation of h, and

$$\mathcal{D} = \begin{pmatrix} D & 0 & 0 & 0 \\ 0 & D & 0 & 0 \\ 0 & 0 & D & 0 \\ 0 & 0 & 0 & D \end{pmatrix} \tag{30}$$

with $D \in M_{n^2}(\mathbb{Z})$ is a matrix whose rows are the coefficient vectors of the Lagrange interpolants $\lambda_{a,b}(x, y)$ for $(a, b) \in E$.

Since $\boldsymbol{g}, \boldsymbol{f}$ are ternary vectors, they have small Euclidean norms. And, γ is the linear combination of $\lambda_{a,b}$'s for $(a, b) \in T$ with $|T|$ small. Thus, the Hamming norm of $\boldsymbol{\rho}(\boldsymbol{\gamma})$ is small. Consequently, the hybrid norm of the vector $(\boldsymbol{g}, \boldsymbol{f}, -\boldsymbol{\rho}(\boldsymbol{\gamma})) \in \mathbb{Z}^{4n^2} \times \mathbb{Z}^{4n^2} \times \mathbb{Z}_q^{4n^2}$ defined as

$$||(\boldsymbol{g}, \boldsymbol{f}, -\boldsymbol{\rho}(\boldsymbol{\gamma}))||_{Hyb} = ||(\boldsymbol{g}, \boldsymbol{f})|| + ||\boldsymbol{\rho}(\boldsymbol{\gamma})||_H \tag{31}$$

is also small. In fact, $(\boldsymbol{g}, \boldsymbol{f}, -\boldsymbol{\rho}(\boldsymbol{\gamma}))$ is most likely one of the shortest vectors in the hybrid lattice $\mathcal{L}_{BQTRU}$ [5, Theorem 4]. Therefore, the usual lattice reduction algorithms cannot find this short vector in hybrid metric. However, authors in [5] show that the security of the key can be related to finding short vectors in higher dimensional Euclidean lattices by expanding the hybrid lattice.

An attacker can select a subset $\{b_0, b_1, \ldots, b_\ell\} \subseteq \mathbb{Z}_q$ such that every element $a \in \mathbb{Z}_q$ can be expressed as $a = \sum_{i=0}^{\ell} a_i b_i$ where $(a_0, a_1, \ldots, a_\ell)$ is a vector with small Euclidean norm. In particular, one can choose $\{b_0, b_1, \ldots, b_\ell\} = \{1, 2, \ldots, 2^\ell\}$ where $2^\ell \leq q < 2^{\ell+1}$ then for every $a \in \mathbb{Z}_q$, $(a_0, a_1, \ldots, a_\ell)$ is a binary vector. Then, expand the BQTRU lattice $\mathcal{L}_{BQTRU}$ to the Euclidean lattice $\mathcal{L}_{exp}$ generated by the rows of the matrix

$$\mathcal{B}_{exp} = \begin{pmatrix} qI_{4n^2} & 0 & 0 \\ \mathcal{H} & I_{4n^2} & 0 \\ \mathcal{D}_{exp} & 0 & I_{4(\ell+1)n^2} \end{pmatrix} \tag{32}$$

where

$$\mathcal{D}_{exp} = \begin{pmatrix} \boldsymbol{b} \otimes D & 0 & 0 & 0 \\ 0 & \boldsymbol{b} \otimes D & 0 & 0 \\ 0 & 0 & \boldsymbol{b} \otimes D & 0 \\ 0 & 0 & 0 & \boldsymbol{b} \otimes D \end{pmatrix}, \ \boldsymbol{b} = (b_0, b_1, \ldots, b_\ell)^{Tr}. \tag{33}$$

It can be shown that the image of the private vector $(\boldsymbol{g}, \boldsymbol{f}, -\boldsymbol{\rho(\gamma)})$ is one of the shortest vectors in the expanded lattice with high probability [5, Proposition 3]. Therefore, the problem of finding the private key is equivalent to solving SVP in a lattice of dimension $(4\ell + 12)n^2$, which is very large. E.g. for $n = 7$, the dimension of the expanded lattice is approximately 2036. Therefore, the authors claim that the hybrid structure of BQTRU thwarts lattice attacks on key for the proposed parameters.

Alternatively, a brute-force search for the key can be performed. As per the BQTRU authors, an attacker first needs to guess the polynomial $\sigma(x, y) = \sum_{i=1}^{|T|} q_i \lambda_{a_i,b_i}(x, y)$ that is a generator for the ideal $Q = \langle q, \sigma \rangle_R$. Since $|T| \leq n$ is unknown to the attacker, the worst-case cost of searching for σ is

$$\sum_{i=1}^{n} (q-1)^i \binom{n^2}{i}. \tag{34}$$

Then, for each choice of σ, the attacker searches for all the quaternions $f = f_0 + f_1 i + f_2 j + f_3 k \in \mathbb{A}$ where $f_i \in L_f$ such that $f * h \,(\mathrm{mod}\, J)$ is small ternary quaternion. The possible number of such f is roughly

$$|L_f| = \binom{n^2}{d_f}^4 \binom{n^2 - d_f}{d_f}^4. \tag{35}$$

In fact, a meet-in-the-middle search [20] is possible on the ternary vector. Further, since $\rho(\gamma) = \rho(f * v)$ where $v = w * \sigma$, the attacker needs to search for $w \in \mathbb{A}_q^\star$ (space of invertible quaternions in $\mathbb{A}_q$) in order to find $\rho(\gamma)$. Hence, the size of the search space is approximately

$$\binom{key}{security} = |\mathbb{A}_q^\star| \binom{n^2}{d_f}^2 \binom{n^2 - d_f}{d_f}^2 \sum_{i=1}^{n} (q-1)^i \binom{n^2}{i}, \tag{36}$$

that amounts to 2^{166} and 2^{396} for the moderate and highest level security parameters, respectively (see Table 4).

4.2 Message security

The authors discuss the search attack on messages without any mention of the possibility of lattice attacks. As per them, the message can be deciphered by randomly searching for a ternary $r = r_0 + r_1 i + r_2 j + r_3 k \in \mathbb{A}$ where $r_i \in L_r$ such that $c - ph * r \,(\mathrm{mod}\, q)$ is ternary quaternion. The size of the search space following the meet-in-the-middle strategy is

$$\binom{message}{security} = \binom{n^2}{d_r}^2 \binom{n^2 - d_r}{d_r}^2 = \frac{(n^2!)^2}{(d_r!)^4 (n^2 - 2d_r)!^2}. \tag{37}$$

This results in 2^{92} and 2^{212} message security achieved for the proposed moderate and highest level security parameters, respectively (see Table 4).

5 Our attack

5.1 On the proposed key generation

As discussed above, the authors estimated the key security based on the combinatorial search solely since, according to them, the lattice attacks on the keys are not feasible. However, they neglected the possibility of combining both kinds of attacks. We propose a hybrid attack that involves searching for the low Hamming weight part of the key and then applying lattice reduction techniques on the remaining Euclidean part.

For a quick revision, the private keys are ternary vectors f, g, both are invertible modulo $J = \langle q, \sigma \rangle_{\mathbb{A}}$. The set $T = \cap_{i=0}^{3}\{(a,b) \in E : g_i(a,b) = 0\}$ is non-empty, and $f_{set} = \{(a,b) \in E : N(f)(a,b) = 0\} \subseteq T$ to ensure the invertibility of f modulo J. We have

$$f * f^{-1} = f^{-1} * f = 1 + \alpha * \sigma \,(\mathrm{mod}\, q) \tag{38}$$

for some $\alpha \in \mathbb{A}$. An attacker first tries to guess the indices of elements of T in E. In simple words, if $E = \{(a_i, b_i) : i = 1, 2, \ldots, n^2\}$ then guess the set $I = \{i : (a_i, b_i) \in T\} \subset \{1, 2, \ldots, n^2\}$ where $|I| = |T| \leq n$. For the correct guess of I and using the relation

$$h = f^{-1} * g + v \,(\mathrm{mod}\, q),$$

one can compute

$$h(a_i, b_i) = v(a_i, b_i) \text{ for } i \in I, \tag{39}$$

as $g(a,b) = 0$ for $(a,b) \in T$. Since $v = w * \sigma \,(\mathrm{mod}\, q) \in J \,(\mathrm{mod}\, q) = \langle \sigma \rangle_{\mathbb{A}}$, therefore v is completely determined by its evaluation over the set T. Let

$$s = h - v \,(\mathrm{mod}\, q) = f^{-1} * g \,(\mathrm{mod}\, q)$$

and $\mathcal{S}$ be the matrix representation of s. Then,

$$\begin{aligned} f * s &= (f * f^{-1}) * g \,(\mathrm{mod}\, q) \\ &= (1 + \alpha * \sigma) * g \,(\mathrm{mod}\, q) = g \,(\mathrm{mod}\, q) \\ &= g + qu \quad \text{for some } u \in \mathbb{A} \end{aligned}$$

since $\sigma * g = 0 \,(\mathrm{mod}\, q)$ as $\sigma \in J \pmod q$ and g vanishes on T. Equivalently, $\boldsymbol{f} \star \mathcal{S} = \boldsymbol{g} + q\boldsymbol{u}$. Therefore, the private key $(\boldsymbol{f}, \boldsymbol{g})$ can be recovered by solving SVP in $8n^2-$dimensional Euclidean lattice $\mathcal{L}_{CS}$ generated by the matrix

$$\mathcal{B}_{CS} = \begin{pmatrix} I_{4n^2} & \mathcal{S} \\ 0 & qI_{4n^2} \end{pmatrix}. \tag{40}$$

It is worth noting that the matrix $\mathcal{S}$ has a specific structure, as shown in Equation (26). We will now discuss how this structure can be exploited to reduce the dimension of the lattice based on the idea introduced by Gentry [17]. We demonstrate that the matrices associated with the elements of quaternion algebra can be *homomorphically folded*, which will reduce the dimension of the lattice to be attacked by half. This thereby disproves the conjecture that BQTRU

is safe against Gentry's attack.

Using $\boldsymbol{f} \star \mathcal{S} = \boldsymbol{g} + q\boldsymbol{u}$, we get the following set of equations:

$$\left.\begin{aligned} \boldsymbol{f}_0 \star S_0 + \boldsymbol{f}_1 \star S_1 + \boldsymbol{f}_2 \star S_2 - \boldsymbol{f}_3 \star S_3 - q\boldsymbol{u}_0 = \boldsymbol{g}_0 \\ \boldsymbol{f}_0 \star S_1 + \boldsymbol{f}_1 \star S_0 - \boldsymbol{f}_2 \star S_3 + \boldsymbol{f}_3 \star S_2 - q\boldsymbol{u}_1 = \boldsymbol{g}_1 \\ \boldsymbol{f}_0 \star S_2 + \boldsymbol{f}_1 \star S_3 + \boldsymbol{f}_2 \star S_0 - \boldsymbol{f}_3 \star S_1 - q\boldsymbol{u}_2 = \boldsymbol{g}_2 \\ \boldsymbol{f}_0 \star S_3 + \boldsymbol{f}_1 \star S_2 - \boldsymbol{f}_2 \star S_1 + \boldsymbol{f}_3 \star S_0 - q\boldsymbol{u}_3 = \boldsymbol{g}_3 \end{aligned}\right\} \tag{41}$$

Lemma 2 (Key folding). *The map* $\phi : \{\mathcal{F} : f \in \mathbb{A}\} \to M_{2n^2}(\mathbb{Z})$ *defined as*

$$\phi(\mathcal{F}) = \begin{pmatrix} F_0 + F_1 & F_2 + F_3 \\ F_2 - F_3 & F_0 - F_1 \end{pmatrix} \tag{42}$$

is a matrix ring homomorphism, i.e., $\phi(\mathcal{F} + \mathcal{G}) = \phi(\mathcal{F}) + \phi(\mathcal{G})$ *and* $\phi(\mathcal{F} \star \mathcal{G}) = \phi(\mathcal{F}) \star \phi(\mathcal{G})$ *for all* $f, g \in \mathbb{A}$.

Proof. See Appendix C. □

To simplify notations, for every quaternion $f = f_0 + f_1 i + f_2 j + f_3 k \in \mathbb{A}$, we denote by

$$\phi_1(\boldsymbol{f}) = (\boldsymbol{f}_0 + \boldsymbol{f}_1, \boldsymbol{f}_2 + \boldsymbol{f}_3) \quad \text{and} \quad \phi_2(\boldsymbol{f}) = (\boldsymbol{f}_2 - \boldsymbol{f}_3, \boldsymbol{f}_0 - \boldsymbol{f}_1). \tag{43}$$

Then, Lemma 2 and equation set (41) give

$$\begin{aligned} (\phi_1(\boldsymbol{f}), \phi_1(\boldsymbol{u})) \star \mathcal{B}_{CS,\phi} &= (\phi_1(\boldsymbol{f}), \phi_1(\boldsymbol{g})), \\ (\phi_2(\boldsymbol{f}), \phi_2(\boldsymbol{u})) \star \mathcal{B}_{CS,\phi} &= (\phi_2(\boldsymbol{f}), \phi_2(\boldsymbol{g})) \end{aligned}$$

where

$$\mathcal{B}_{CS,\phi} = \begin{pmatrix} I_{2n^2} & \phi(\mathcal{S}) \\ 0 & qI_{2n^2} \end{pmatrix} \tag{44}$$

generates a $4n^2-$dimensional lattice that we call $\mathcal{L}_{CS,\phi}$. The Gaussian heuristic estimates the expected length of the shortest vector in the lattice $\mathcal{L}_{CS,\phi}$ to be

$$gh(\mathcal{L}_{CS,\phi}) = \sqrt{\frac{4n^2}{2\pi e}}\ |det(\mathcal{B}_{CS,\phi})|^{\frac{1}{4n^2}} = \sqrt{\frac{2qn^2}{\pi e}}. \tag{45}$$

Whereas the images of the private key belonging to the lattice $\mathcal{L}_{CS,\phi}$ have norms

$$||(\phi_1(\boldsymbol{f}), \phi_1(\boldsymbol{g}))|| \approx ||(\phi_2(\boldsymbol{f}), \phi_2(\boldsymbol{g}))|| \leq \sqrt{2}||(\boldsymbol{f}, \boldsymbol{g})|| = \sqrt{\frac{32n^2}{7}}. \tag{46}$$

Since q is taken to be approximately $24n^2/7$, therefore, for the recommended parameters of BQTRU, the vectors $(\phi_1(\boldsymbol{f}), \phi_1(\boldsymbol{g}))$ and $(\phi_2(\boldsymbol{f}), \phi_2(\boldsymbol{g}))$ are $O(1/n)$ shorter than estimated by the Gaussian heuristics. Thus, we expect them to be the shortest vectors in the lattice $\mathcal{L}_{CS,\phi}$ with high probability. This proves that attacking the key is equivalent to searching for the indices of elements of set T in set E times the cost of solving SVP in a $4n^2-$dimensional lattice. We discuss the concrete cost analysis of our attack in Section 6.

Alternative keys. In case the lattice reduction algorithms do not return the exact images of the private keys in the lattice $\mathcal{L}_{CS,\phi}$, we provide an alternative way to handle such situations.

Theorem 2 (Lift-back). *Let the vectors* $(\boldsymbol{x}, \boldsymbol{y}) = (\boldsymbol{x}_0, \boldsymbol{x}_1, \boldsymbol{y}_0, \boldsymbol{y}_1)$ *and* $(\boldsymbol{w}, \boldsymbol{z}) = (\boldsymbol{w}_0, \boldsymbol{w}_1, \boldsymbol{z}_0, \boldsymbol{z}_1)$ *belonging to the lattice* $\mathcal{L}_{CS,\phi}$*. Then the vector*

$$lift(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{w}, \boldsymbol{z}) = (\boldsymbol{x}_0 + \boldsymbol{w}_1, \boldsymbol{x}_0 - \boldsymbol{w}_1, \boldsymbol{x}_1 + \boldsymbol{w}_0, \boldsymbol{x}_1 - \boldsymbol{w}_0, \boldsymbol{y}_0 + \boldsymbol{z}_1, \boldsymbol{y}_0 - \boldsymbol{z}_1, \boldsymbol{y}_1 + \boldsymbol{z}_0, \boldsymbol{y}_1 - \boldsymbol{z}_0)$$

belongs to the lattice $\mathcal{L}_{CS}$*.*

Proof. Let $\boldsymbol{a} = (\boldsymbol{a}_0, \boldsymbol{a}_1), \boldsymbol{b} = (\boldsymbol{b}_0, \boldsymbol{b}_1) \in \mathbb{Z}^{2n^2}$ be such that

$$(\boldsymbol{x}, \boldsymbol{a}) \star \mathcal{B}_{CS,\phi} = (\boldsymbol{x}, \boldsymbol{y}) \quad \text{and} \quad (\boldsymbol{w}, \boldsymbol{b}) \star \mathcal{B}_{CS,\phi} = (\boldsymbol{w}, \boldsymbol{z}). \tag{47}$$

From (47), we get

$$\left.\begin{aligned} \boldsymbol{x}_0 \star (H_0 + H_1) + \boldsymbol{x}_1 \star (H_2 - H_3) + q\boldsymbol{a}_0 &= \boldsymbol{y}_0 \\ \boldsymbol{x}_0 \star (H_2 + H_3) + \boldsymbol{x}_1 \star (H_0 - H_1) + q\boldsymbol{a}_1 &= \boldsymbol{y}_1 \\ \boldsymbol{w}_0 \star (H_0 + H_1) + \boldsymbol{w}_1 \star (H_2 - H_3) + q\boldsymbol{b}_0 &= \boldsymbol{z}_0 \\ \boldsymbol{w}_0 \star (H_2 + H_3) + \boldsymbol{w}_1 \star (H_0 - H_1) + q\boldsymbol{b}_1 &= \boldsymbol{z}_1 \end{aligned}\right\} \tag{48}$$

Using (48), one can show that

$$lift(\boldsymbol{x}, \boldsymbol{a}, \boldsymbol{w}, \boldsymbol{b}) \star \mathcal{B}_{CS} = lift(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{w}, \boldsymbol{z}).$$

This proves our claim. □

Moreover,

$$|| \, lift(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{w}, \boldsymbol{z})|| \leq 2\sqrt{||(\boldsymbol{x}, \boldsymbol{y})||^2 + ||(\boldsymbol{w}, \boldsymbol{z})||^2}. \tag{49}$$

Therefore, if one is able to find two short enough vectors in the lattice $\mathcal{L}_{CS,\phi}$, then their *lift-back* can serve as a possible decryption key with high probability. The same is reflected in our experiments.

Similar to NTRU, all the rotations of the private key of BQTRU are also potential decryption keys.

Definition 11 (Rotations). *For an element* $f = f_0 + f_1 i + f_2 j + f_3 k \in \mathbb{A}$ *where* $f_i \in \frac{\mathbb{Z}[x,y]}{\langle x^n - 1, y^n - 1\rangle}$*, the* $4n^2$ *rotations of* f *are given by*

$$x^a y^b * f * \delta = (x^a y^b * f_0)\delta + (x^a y^b * f_1)(i\delta) + (x^a y^b * f_2)(j\delta) + (x^a y^b * f_3)(k\delta)$$

where $\delta \in \{1, i, j, k\}$ *and* $a, b \in \{0, 1, \ldots, n-1\}$*.*

It is clear that if f, g, and $h \in \mathbb{A}$ are such that $f * h = g \pmod q$. Then, $(x^a y^b * f * \delta) * h = (x^a y^b * g * \delta) \pmod q$, for all $\delta \in \{1, i, j, k\}$ and $a, b \in \{0, 1, \ldots, n-1\}$. Therefore, all the rotations of the private key belong to the lattice $\mathcal{L}_{CS}$ and, by definition, have the same norm as the private key. Hence, their images in the lattice $\mathcal{L}_{CS,\phi}$ are also short vectors with high probability. This increases the chances of an attacker to find suitable short vectors in lower dimensional lattices and lift them back to the original lattice to check for their eligibility as a decryption key.

5.2 On messages

The ciphertext $c = c_0 + c_1 i + c_2 j + c_3 k$ for a message $m = m_0 + m_1 i + m_2 j + m_3 k$ is computed as

$$c = ph * r + m \,(\mathrm{mod}\, q) \tag{50}$$

where $r = r_0 + r_1 i + r_2 j + r_3 k \in \mathbb{A}$ such that each r_i is chosen randomly from L_r. Using Equation (26), we get

$$\boldsymbol{c} = p\boldsymbol{r} \star \tilde{\mathcal{H}} + \boldsymbol{m} + q\boldsymbol{u} \quad \text{for some } u \in \mathbb{A}. \tag{51}$$

This gives us the following set of equations:

$$\left.\begin{aligned} p(\boldsymbol{r}_0 \star H_0 + \boldsymbol{r}_1 \star H_1 + \boldsymbol{r}_2 \star H_2 - \boldsymbol{r}_3 \star H_3) + \boldsymbol{m}_0 + q\boldsymbol{u}_0 &= \boldsymbol{c}_0 \\ p(\boldsymbol{r}_0 \star H_1 + \boldsymbol{r}_1 \star H_0 + \boldsymbol{r}_2 \star H_3 - \boldsymbol{r}_3 \star H_2) + \boldsymbol{m}_1 + q\boldsymbol{u}_1 &= \boldsymbol{c}_1 \\ p(\boldsymbol{r}_0 \star H_2 - \boldsymbol{r}_1 \star H_3 + \boldsymbol{r}_2 \star H_0 + \boldsymbol{r}_3 \star H_1) + \boldsymbol{m}_2 + q\boldsymbol{u}_2 &= \boldsymbol{c}_2 \\ p(\boldsymbol{r}_0 \star H_3 - \boldsymbol{r}_1 \star H_2 + \boldsymbol{r}_2 \star H_1 + \boldsymbol{r}_3 \star H_0) + \boldsymbol{m}_3 + q\boldsymbol{u}_3 &= \boldsymbol{c}_3 \end{aligned}\right\} \tag{52}$$

Lemma 3 (Message folding). *The map* $\tilde{\phi} : \{\tilde{\mathcal{F}} : f \in \mathbb{A}\} \to M_{2n^2}(\mathbb{Z})$ *defined as*

$$\tilde{\phi}(\tilde{\mathcal{F}}) = \begin{pmatrix} F_0 + F_1 & F_2 - F_3 \\ F_2 + F_3 & F_0 - F_1 \end{pmatrix} \tag{53}$$

is a matrix ring homomorphism.

Proof. Similar to the proof of Lemma 2. □

Further, for every quaternion $f = f_0 + f_1 i + f_2 j + f_3 k \in \mathbb{A}$, we denote by

$$\tilde{\phi}_1(\boldsymbol{f}) = (\boldsymbol{f}_0 + \boldsymbol{f}_1, \boldsymbol{f}_2 - \boldsymbol{f}_3) \quad \text{and} \quad \tilde{\phi}_2(\boldsymbol{f}) = (\boldsymbol{f}_2 + \boldsymbol{f}_3, \boldsymbol{f}_0 - \boldsymbol{f}_1). \tag{54}$$

Then, Lemma 3 and equation set (52) give

$$\begin{aligned} (0, \tilde{\phi}_1(\boldsymbol{c})) &= (\tilde{\phi}_1(\boldsymbol{r}), \tilde{\phi}_1(\boldsymbol{u})) \star \mathcal{B}_{CS,\tilde{\phi}} + (-\tilde{\phi}_1(\boldsymbol{r}), \tilde{\phi}_1(\boldsymbol{m})), \\ (0, \tilde{\phi}_2(\boldsymbol{c})) &= (\tilde{\phi}_2(\boldsymbol{r}), \tilde{\phi}_2(\boldsymbol{u})) \star \mathcal{B}_{CS,\tilde{\phi}} + (-\tilde{\phi}_2(\boldsymbol{r}), \tilde{\phi}_2(\boldsymbol{m})) \end{aligned}$$

where

$$\mathcal{B}_{CS,\tilde{\phi}} = \begin{pmatrix} I_{2n^2} & \tilde{\phi}(\tilde{\mathcal{H}}) \\ 0 & qI_{2n^2} \end{pmatrix} \tag{55}$$

generates a $4n^2-$dimensional lattice that we call $\mathcal{L}_{CS,\tilde{\phi}}$. Since $\boldsymbol{r}, \boldsymbol{m}$ are ternary vectors with many zeros, therefore, $(-\tilde{\phi}_1(\boldsymbol{r}), \tilde{\phi}_1(\boldsymbol{m}))$, $(-\tilde{\phi}_2(\boldsymbol{r}), \tilde{\phi}_2(\boldsymbol{m}))$ take values from the set $\{0, \pm 1, \pm 2\}$ with majority of 0s, ± 1s and a few ± 2s. Therefore, we expect with a high probability that the vectors

$$(\tilde{\phi}_1(\boldsymbol{r}), \tilde{\phi}_1(\boldsymbol{u})) \star \mathcal{B}_{CS,\tilde{\phi}} \quad \text{and} \quad (\tilde{\phi}_2(\boldsymbol{r}), \tilde{\phi}_2(\boldsymbol{u})) \star \mathcal{B}_{CS,\tilde{\phi}}$$

in the lattice $\mathcal{L}_{CS,\tilde{\phi}}$ are closest to the targets $(0, \tilde{\phi}_1(\boldsymbol{c}))$ and $(0, \tilde{\phi}_2(\boldsymbol{c}))$, respectively. Thus, the message can be recovered by solving CVP in $4n^2-$dimensional lattice $\mathcal{L}_{CS,\tilde{\phi}}$.

5.3 Other related issues

Small values of q. Since $d_f = d_r = d_g = d_m \approx n^2/7$ and p is fixed to be 3, the value of q required to avoid decryption failure is $q \geq 72n^2/7$. One can choose smaller values of q, allowing negligible decryption failure to only an extent that does not pose any security threat [21]. For instance, a cryptosystem achieving security level 2^λ should not have the probability of decryption failure higher than $2^{-\lambda}$. However, the proposed parameters $(n, q, p) = (7, 113, 3), (11, 199, 3)$ claiming to achieve moderate and highest security levels allow approximately 2^{-10} and 2^{-22} decryption failure rate, respectively, which are far away from the requirements.

Low cardinality of T. The greater the size of set T, the better security against combinatorial search. However, our experiments indicate that for the suggested parameters, the size of set T is quite small when g is randomly selected. In almost all the cases, we found that $|T| = 1$ or 2, which benefits the attacker.

Weak instances. In experiments, we encountered some instances where for a wrong guess of T, say $T' \neq T$, and corresponding v', the lattice generated by the matrix

$$\mathcal{B}'_{CS} = \begin{pmatrix} I_{4n^2} & \mathcal{H}' \\ 0 & qI_{4n^2} \end{pmatrix}$$

where $\mathcal{H}'$ is matrix of $h' = h - v' \pmod q$, contains short vector (f', g'). This gives a potential decryption key as $f' * (h - v') = g' \pmod q$. Therefore, for a ciphertext $c = ph * r + m \pmod q$, $f' * c = pg' * m + f' * m + pf' * r * v' + \epsilon' q$, for some $\epsilon' \in \mathbb{A}$. Now, decrypt in a similar way as for BQTRU but with the private lattice corresponding to T'. Therefore, these weak instances are beneficial to an adversary when $|T'| < |T|$.

6 Cost analysis and experimental verification

Besides the proposed parameter sets in the original paper, we consider another set of parameters that provides no decryption failure in the same dimension and a toy parameter set with decryption failure for reference comparison in Table 1.

Table 1: Original and decryption-free BQTRU parameter sets (n, q, d_f, d_g, d_r) in equivalent dimensions.

Security tag	No Decryption failure	Decryption failure	
Toy	(5, 241, 3, 3, 3)	(5, 71, 3, 3, 3)	
Moderate	(7, 547, 7, 6, 6)	(7, 113, 7, 6, 6)	} original parameters
Highest	(11, 1277, 17, 17, 13)	(11, 199, 17, 17, 13)	

As per the claimed security estimation for key and message attack (equations (36), (37)), the decryption-free parameter sets have higher key security and equivalent message security compared to the original BQTRU parameter sets in the same dimension. Later, our analysis shows that they are easier to attack than the original parameter sets.

Key recovery attack. We first guess the positions of the low Hamming weight vector $\boldsymbol{v}$ with respect to the Lagrange basis. Then, for each guess, we attempt to find a short vector in the lattice $\mathcal{L}_{CS}$ constructed based on the guessed $\boldsymbol{v}$. If the positions are guessed correctly, the vector $(\boldsymbol{f}, \boldsymbol{g})$ or its rotations (as in Definition 11) lie in the lattice $\mathcal{L}_{CS}$.

The cost of guessing is related to the cardinality of the set T. In BQTRU's paper, the authors state that $|T| \leq n$ is needed for successful decryption. Therefore, in the worst case, the attacker has to try $\sum_{i=1}^{n} \binom{n^2}{i}$ different guesses for the nonzero positions of $\boldsymbol{v}$. Since the lattice $\mathcal{L}_{CS}$ is vulnerable to the folding attack introduced in subsection 5.1, the attacker constructs the folded lattice and runs a lattice reduction algorithm, such as the *progressive* BKZ, in a specific range where β is expected to successfully retrieve the secret key $(\boldsymbol{f}, \boldsymbol{g})$. The lattice reduction cost is estimated based on the blocksize needed to find the two short vectors $(\phi_1(\boldsymbol{f}), \phi_1(\boldsymbol{g}))$ and $(\phi_2(\boldsymbol{f}), \phi_2(\boldsymbol{g}))$ as summarized in Figure 2. Consequently, the total cost of the key attack is calculated as *the guessing cost* $\times$ *lattice reduction cost*.

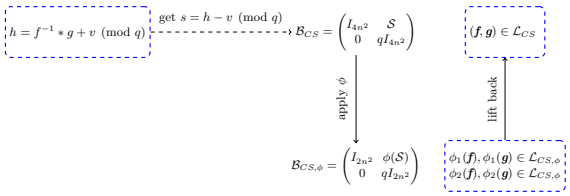

Fig. 2: Key recovery attack against BQTRU lattice; instead of reducing a lattice of dimension $8n^2$ to find a short vector $(\boldsymbol{f}, \boldsymbol{g})$, we map the problem into reducing a lattice of dimension $4n^2$ to find two short vectors $(\phi_1(\boldsymbol{f}), \phi_1(\boldsymbol{g})), (\phi_2(\boldsymbol{f}), \phi_2(\boldsymbol{g}))$ of the same norm (on average) as $(\boldsymbol{f}, \boldsymbol{g})$.

Message recovery attack. The message recovery attack, as discussed in subsection 5.2, is relatively straightforward. The complexity of recovering the message is determined by the effort required to solve the Closest Vector Problem (CVP) for the lattice $\mathcal{L}_{CS,\tilde{\phi}}$ with respect to two distinct target vectors, namely $(0, \tilde{\phi_1}(\boldsymbol{c}))$ and $(0, \tilde{\phi_2}(\boldsymbol{c}))$. To assess the difficulty of solving the CVP involved in the message attack, we employ the embedding technique (Equation 5) to transform the CVP into the Shortest Vector Problem (SVP). Consequently, the attack cost depends on the blocksize required to identify the shortest vectors in the two lattices generated from the basis $\mathcal{B}_{CS,\phi_1}$ and $\mathcal{B}_{CS,\phi_2}$, as summarized in Figure 3.

Estimating $\boldsymbol{\beta}$. The literature has introduced several works [3,26] that estimate the blocksize needed for BKZ to recover an unusually short vector (that serves as a decryption key) in lattices. The *2016-estimate* is a well-regarded estimate that suggests that, under the GSA, the BKZ algorithm can find a short vector $\boldsymbol{v}$ in a random $d-$dimensional lattice $\mathcal{L}_B$ if the following condition is met:

$$\sqrt{\beta/d}\ ||\boldsymbol{v}|| < {\delta_\beta}^{2\beta-d-1} \cdot det(\mathcal{L}_B)^{1/d}, \tag{56}$$

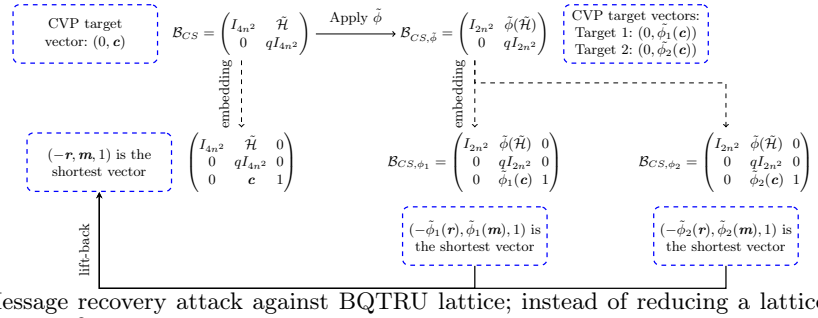

Fig. 3: Message recovery attack against BQTRU lattice; instead of reducing a lattice of dimension $8n^2$ to find a short vector $(-\boldsymbol{r}, \boldsymbol{m}, 1)$, we map the problem into reducing two lattices of dimension $4n^2 + 1$ to find two short vectors $(-\tilde{\phi}_1(\boldsymbol{r}), \tilde{\phi}_1(\boldsymbol{m}), 1)$ and $(-\tilde{\phi}_2(\boldsymbol{r}), \tilde{\phi}_2(\boldsymbol{m}), 1)$ of the same norm (on average) as $(-\boldsymbol{r}, \boldsymbol{m}, 1)$.

where δ represents the root Hermite factor. Having the inequality satisfied (for some β) indicates that the BKZ algorithm has potentially identified the projection of the unusually short vector $\boldsymbol{v}$ in the projected sublattice $\mathcal{L}_{B[d-\beta:d)}$ that can be lifted to the vector $\boldsymbol{v} \in \mathcal{L}_B$ and serve as a decryption key. A simple analysis shows that higher values of β are needed to satisfy the inequality (56) when the dimension of the lattice d or the *lattice gap*, defined as $\frac{||\boldsymbol{v}||}{gh(\mathcal{L})}$, increases.

The NTRU *Fatigue-estimate* [15] is the state-of-art estimate for NTRU-like lattices. The estimate itself is a refinement of the *2016-estimator* and the probabilistic estimate [14] that allows for describing the profile of a reduced basis even using a few tours of progressive BKZ. The *Fatigue-estimator* incorporates the ZGSA (instead of GSA) as a more accurate description of the profile as well.

Definition 12 (ZGSA). *[15, Heuristic 2.8] Let B be a basis of a $2d-$dimensional $q-$ary lattice $\mathcal{L}_B$ with d $q-$vectors. After BKZ$-\beta$ reduction, the profile of the reduced basis (for $t = \frac{1}{2} + \frac{ln(q)}{ln(\delta_\beta)}$) has the following shape:*

$$||\boldsymbol{b}_i^\star|| = \begin{cases} q, & \text{if } i \leq d - t \\ \sqrt{q} \cdot \delta_\beta^{2d-1-2i}, & \text{if } d - t < i < d + t \\ 1, & \text{if } i \geq d + t \end{cases} \tag{57}$$

The *Fatigue-estimate* gives a better prediction for the blocksize in both the lattices $\mathcal{L}_{CS}$ and $\mathcal{L}_{CS,\phi}$ as the two lattices are NTRU-like lattices.

Although our attack reduces the dimension of the lattices by half, the blocksize and, hence, the reduction cost depends not only on the dimension but also on the lattice gap. Therefore, to prove the effectiveness of our attack, we need to demonstrate that the blocksize required to find $(\phi_1(\boldsymbol{f}), \phi_1(\boldsymbol{g})), (\phi_2(\boldsymbol{f}), \phi_2\boldsymbol{g})) \in \mathcal{L}_{CS,\phi}$ is much smaller than that needed for finding $(\boldsymbol{f}, \boldsymbol{g}) \in \mathcal{L}_{CS}$.

The secret key $(\boldsymbol{f}, \boldsymbol{g}) \in \mathbb{Z}^{8n^2}$ consists of ternary coefficients, with approximately $8n^2/7$ of the coefficients equal to 1 and -1 each, and the rest being 0. When estimating β using the *Fatigue-estimator*, we consider the coefficients of $(\boldsymbol{f}, \boldsymbol{g})$ to follow a discrete Gaussian distribution with mean $\mu = 0$ and variance $\sigma^2 = 2/7$. Using the fact that if X and Y are two independent and identically

distributed discrete Gaussian random variables with mean μ and variance σ^2, then $X + Y$ is a discrete Gaussian random variable with mean 2μ and variance $2\sigma^2$, we get that the coefficients of $(\phi_1(\boldsymbol{f}), \phi_1(\boldsymbol{g}))$ and $(\phi_2(\boldsymbol{f}), \phi_2(\boldsymbol{g}))$ take values from the set $\{-2, -1, 0, 1, 2\}$ following a discrete Gaussian distribution with mean $\mu' = 0$ and variance $\sigma'^2 = 2\sigma^2 = 4/7$. This gives

$$||(\phi_1(\boldsymbol{f}), \phi_1(\boldsymbol{g}))|| \approx ||(\phi_2(\boldsymbol{f}), \phi_2(\boldsymbol{g}))|| \approx \sqrt{4n^2\,\sigma'^2} = 2n\sigma' = ||(\boldsymbol{f}, \boldsymbol{g})||. \tag{58}$$

We experimentally also verified that (58) holds with minor deviations while equation (46) gives an upper bound on the norm.

One can check that any β that satisfies the inequality (56) for the lattice $\mathcal{L}_{CS}$ of dimension $8n^2$ also satisfies it for the lattice $\mathcal{L}_{CS,\phi}$ of dimension $4n^2$ for the same norm $||\boldsymbol{v}|| = 2n\sigma'$. Therefore, *2016-estimator* results in smaller values of β for $\mathcal{L}_{CS,\phi}$ compared to $\mathcal{L}_{CS}$. A more precise description of the blocksize estimation for the concrete parameters (with and without decryption failure) based on the *Fatigue* and *2016-estimator* is given in Table 2. The noticeable difference in the blocksizes before and after dimension reduction clearly shows the benefit of our folding.

Table 2: Blocksize estimation using *2016-estimate* and *Fatigue-estimate* for retrieving a short vector that represents a key/message in BQTRU lattices of dimension $8n^2$ without folding versus $4n^2$ with folding attack; the estimators predicts lower blocksizes when our folding reduction is applied.

	Parameters	**No folding**		**Folding**	
	(n, q, d_f, d_g, d_ϕ)	$\beta_{Fatigue}$	β_{2016}	$\beta_{Fatigue}$	β_{2016}
Decryption Failure	$(5, 71, 3, 3, 3)$	52	72	3	–
	$(7, 113, 7, 6, 6)$	145	166	56	77
	$(11, 199, 17, 17, 13)$	421	456	204	224
	$(13, 677, 24, 24, 24)$	529	562	250	268
	$(17, 919, 41, 41, 41)$	960	1014	469	493
No Decryption Failure	$(5, 241, 3, 3, 3)$	20	39	2	–
	$(7, 547, 7, 6, 6)$	101	115	18	–
	$(11, 1277, 17, 17, 13)$	320	340	137	152
	$(13, 1847, 24, 24, 24)$	465	490	210	225
	$(17, 3061, 41, 41, 41)$	840	879	395	414

Experimental verification. For experimental verification, we apply our key and message attack against the parameter sets of BQTRU. We run progressive BKZ and identify the smallest blocksize needed to retrieve the decryption key and the message with enumeration as an SVP oracle. We depend on FPyLLL [43] as a Python wrapper to FPLLL [42]. Timed results[‡] have been measured on a system running Linux (Ubuntu 22.04.2 LTS) with 13th Gen Intel(R) Core(TM) i7-13700 equipped with 16 physical cores @ 800 MHZ (min) and 32 GB RAM; each core can run up to 2 threads on parallel.

[‡]Other devices have also been used to run different experiments, as detailed on the GitHub link. We are reporting only the blocksize β for the other results; the time required is an orthogonal question.

Trivial short vectors. As in NTRU lattice, BQTRU key recovery lattice $\mathcal{L}_{CS}$ contains trivial short vectors like $(\mathbf{1}^{4n^2}, \mathbf{0}^{4n^2})$ that does not help in the decryption process. After applying our folding reduction, the images of these vectors also lie in the lattice $\mathcal{L}_{CS,\phi}$. To avoid these useless vectors, one can project against using what so-called *short vector hints* [14]. However, the cost of retrieving the decryption key may increase as the volume of the lattice to be reduced decreases [27, Theorem 5.6]. In our experiment, we ignore these vectors automatically, considering only the invertible short vectors as possible decryption keys.

Effect of rotations. Similarly to NTRU, the BQTRU key-recovery lattice does not contain only the decryption key but also other 'rotations' with respect to the underlying structure. While in NTRU, these rotations are cyclic rotations, for BQTRU, these rotations are slightly different (given by Definition 11) but maintain the same norm as the decryption key. Additionally, the number of these rotations is similar to the NTRU lattice and equal $d/2$ where d is the lattice dimension. The effect of these rotations is reflected as lower values of β that are needed to recover the secret key. *Fatigue-estimate* takes into consideration the probabilities of finding one of these rotations, and hence the expected β is smaller compared to the *2016-estimate* that considers only the lattice volume and vector projection. Experimentally, one can also notice that the blocksize required for retrieving the key is smaller than that needed to retrieve messages for roughly the same dimension and lattice volume. See Table 3.

Table 3: Blocksize required to retrieve the key/message verified experimentally against BQTRU parameter sets. The blocksizes are averaged over at least 50 trials except for $\beta > 60$ where only 10 trials have been executed.

	Parameters	**No folding**		**Folding**	
	(n, q, d_f, d_g, d_ϕ)	β_{Key}	$\beta_{Message}$	β_{Key}	$\beta_{Message}$
Decryption Failure	$(5, 71, 3, 3, 3)$	27	56	2.5	7.3
	$(7, 113, 7, 6, 6)$	–	–	52.6	63.4
No Decryption Failure	$(5, 241, 3, 3, 3)$	4.4	17.9	2	2.3
	$(7, 547, 7, 6, 6)$	–	–	7	18.1

From Table 3, we can see the efficiency of our folding attack. When a decryption failure is allowed, for $n = 5$, the blocksize required to retrieve the key and message drop from 27 and 56 (when no folding is applied) to just 2.5 and 7.3 on average (when folding is used). Furthermore, for $n = 7$, the estimated blocksize to retrieve the key and the message with no folding is greater than 100, which is higher than the record β that ever has been reached experimentally for NTRU-like lattice [27]. However, with our folding attack, we can retrieve the key and the message with average blocksize 52.6 and 63.4, respectively, for the parameters with decryption failure and just 7 and 18.1 for the no decryption failure parameter sets.

Revised security estimation. Following our experimental findings and discussion on the estimated blocksize to find the decryption key or the message, we find that the *Fatigue estimate* serves as a good estimator for the blocksize

required to retrieve the decryption key, and the *2016-estimate* serves as a conservative estimator for the blocksize needed to retrieve the message (for larger blocksize $\beta > 50$).

As stated earlier, the cost of the lattice reduction using an algorithm like BKZ is heavily determined by the SVP oracles: Enumeration and sieving. Recent advances [2, 27] suggest that sieving can outperform enumeration starting from $\beta > 65$. The literature has introduced different models to estimate the security based on the blocksize β. The proposed models have different elementary operations of measurements. In enumeration, the unit is the number of nodes visited during enumeration, which costs approximately 100 CPU cycles, while in sieving, the unit is an operation on a word-sized integer that costs about 1 CPU cycle. One can refer to [1] for a detailed discussion of these models. By analyzing the different models, it is observed that the security level increases almost by $1-$bit every $2-4$ blocksizes (a relation that does not need to be linear).

For the parameter sets with $n > 7$, the estimated blocksize to get the decryption key or the message is greater than 65; therefore, we rely on the sieving as an SVP oracle for the lattice reduction. The conservative cost of BKZ with the sieving model, which is widely used to estimate security in many schemes, is given by $2^{0.292\beta+o(\beta)}$ (classically) [7] and $2^{0.265\beta+o(\beta)}$ (quantumly) [30]. However, in our security estimation, we rely on the sieving model that gives the highest classical security estimation of $2^{0.368\beta}$ (according to survey [1]) in favor of BQTRU construction. For $n = 7$, we report our experimental findings as an accurate description of the security level. For key lattice, the average β to retrieve the vector is approximately 52.6 compared to 63.4 for the message recovery attack; however, guessing v also contributes to the total cost of the key recovery attack. Similarly to sieving, if we consider the enumeration model that gives the highest security estimation that is $2^{0.000784\beta^2+0.366\beta-0.9+\log_2(8d)}$ (classically), then the lattice reduction part of the key and the message contributes 31 and $36-$bit, respectively. The guessing part for v in the key attack contributes at maximum by $26-$bit security when the set $|T|$ cardinality is maximal. Overall, the key security is estimated to be $47-$bit versus $36-$bit for the message security. Experimentally, the key attack is much more successful than estimated as the cardinality of T is coming to be smaller than the maximum possible value n for most of the generated keys as in Algorithm 1. On our system, it takes almost 12 core days (on average) to retrieve the key and the message for the moderate parameter sets of BQTRU. Consequently, we can summarize our security estimation (in favor of BQTRU construction) for the message and key recovery attack against the parameter sets proposed in the original work of BQTRU as in Table 4.

7 Failed attempts to fix BQTRU

7.1 Modifying the key generation

Our attack on the key works for the following reasons. Guessing the set $I = \{i : (a_i, b_i) \in T\}$ is enough to completely determine v. Further, since $\sigma * g = 0 \,(\mathrm{mod}\, q)$ gives $f * (h - v) = g \,(\mathrm{mod}\, q)$. Therefore, for the correct v, one can

Table 4: Revised security levels for proposed BQTRU parameter sets. The key security estimation is calculated as the cost of guessing v (that is $\sum_{i=1}^{n}\binom{n^2}{i}$) times the lattice reduction cost, while the message security cost is due to the lattice reduction only.

		key sec.		**message sec.**	
claimed security tag	**parameters**	claimed	our estimate	claimed	our estimate
Moderate	$(7, 113, 7, 6, 6)$	> 166	**47**	92	**36**
Highest	$(11, 199, 17, 17, 13)$	> 396	**125**	212	**82**

overcome the hybrid nature of the associated lattice by searching for the private key as a short vector in an $8n^2-$dimensional $q-$ary lattice $\mathcal{L}_{CS}$ that can be further subjected to a dimension reduction by half. Moreover, the low cardinality of T also favors the adversary.

To avoid the above-discussed scenarios, we modified the key generation as follows:

Algorithm 2: Modified key generation

$f_0 \xleftarrow{\$} \mathcal{T}(d_f + 1, d_f)$
for $i \leftarrow 1$ *to* 3 **do** $f_i \xleftarrow{\$} L_f$
$f \leftarrow f_0 + f_1 i + f_2 j + f_3 k$
$f_{set} \leftarrow \{(a, b) \in E : N(f)(a, b) = 0\}$
if $|f_{set}| \geq n$ *or* f_{set} *is empty* **then** `go to step 1`
$T \leftarrow f_{set}$
while $|T| < n$ **do**
    $(a, b) \xleftarrow{\$} E \backslash T$
    $T = T \cup \{(a, b)\}$
for $(a_i, b_i) \in T$ **do** $q_i \xleftarrow{\$} \mathbb{Z}_q^\star$
$\sigma \leftarrow \sum_{(a_i,b_i)\in T} q_i \lambda_{a_i,b_i}(x, y)$
for $i \leftarrow 0$ *to* 3 **do** $g_i \xleftarrow{\$} L_g$
$g \leftarrow g_0 + g_1 i + g_2 j + g_3 k$
if $\sigma * g = 0 \,(\mathrm{mod}\, q)$ *or* $g_{set} \leftarrow \{(a, b) \in E : g(a, b) = 0\} \subseteq T$ **then**
    `go to step 12`
$w \xleftarrow{\$} \mathbb{L}_q^\star$
$v \leftarrow w * \sigma \,(\mathrm{mod}\, q)$
return `Public Key:` $h = f^{-1} * g + v \,(\mathrm{mod}\, q)$, `Private key:` f, g, v

Since f_{set} is a proper subset of T and $g_{set} \nsubseteq T$, the equation $h(a, b) \neq v(a, b) \,(\mathrm{mod}\ q)$ for at least some of the $(a, b) \in T$. Therefore, Algorithm 2 returns keys such that guessing the set I is not enough to determine v. Moreover, since f_{set} is non-empty, $f * f^{-1} = 1 + \alpha * \sigma \,(\mathrm{mod}\, q)$ for some nonzero $\alpha \in \mathbb{A}$, and $\sigma * g \neq 0 \,(\mathrm{mod}\, q)$. Therefore, even for the correct v, $f * (h - v) \neq g \,(\mathrm{mod}\, q)$. Hence, (f, g) cannot be recovered as a short vector in the $q-$ary lattice $\mathcal{L}_{CS}$.

Intuitively, this approach seems to thwart our key attack. However, we demonstrate that a similar hybrid attack is feasible on the modified key generation. The only change is that we now need to solve SVP in a lattice with a smaller deter-

minant, making the problem slightly harder. Suppose the attacker can correctly guess the set I, then, he can construct the $4n^2 \times 4n^2$ matrix

$$\mathcal{B}_{private} = \begin{pmatrix} D' & 0 & 0 & 0 \\ 0 & D' & 0 & 0 \\ 0 & 0 & D' & 0 \\ 0 & 0 & 0 & D' \end{pmatrix}$$

generating the private lattice $\mathcal{L}_{private}$ corresponding to the ideal $J \cong Q^4 \subseteq \mathbb{A}$ where D' is $n^2 \times n^2$ generator matrix for the lattice $\mathcal{L}_Q$ corresponding to the ideal $Q \subseteq R$. We have,

$$\begin{aligned} f * h &= (f * f^{-1}) * g + v \,(\mathrm{mod}\, q) \\ &= (1 + \alpha * \sigma) * g + f * v \,(\mathrm{mod}\, q) \\ &= g + (\alpha * g + f * w) * \sigma \,(\mathrm{mod}\, q) \\ &= g + (\alpha * g + f * w) * \sigma + qu \quad \text{for some } u \in \mathbb{A} \\ &= g + \gamma \end{aligned}$$

where, $\gamma = (\alpha * g + f * w) * \sigma + qu \in J$. Therefore, there exists some $\boldsymbol{u} \in \mathbb{Z}^{4n^2}$ such that $\boldsymbol{\gamma} = \boldsymbol{u} \star \mathcal{B}_{private}$. As a result,

$$(\boldsymbol{f}, -\boldsymbol{u}) \star \mathcal{B}_{CS}^{new} = (\boldsymbol{f}, \boldsymbol{g}), \tag{59}$$

where,

$$\mathcal{B}_{CS}^{new} = \begin{pmatrix} I_{4n^2} & \mathcal{H} \\ 0 & \mathcal{B}_{private} \end{pmatrix}. \tag{60}$$

Consequently, the private key can be recovered by solving SVP in $8n^2-$dimensional lattice $\mathcal{L}_{CS}^{new}$ generated by the matrix $\mathcal{B}_{CS}^{new}$. Further, a similar dimension reduction as for our attack on the original key generation is possible for the lattice $\mathcal{L}_{CS}^{new}$. Recovering the private key for the modified key generation is also equivalent to solving SVP in a $4n^2-$dimensional lattice $\mathcal{L}_{CS,\phi}^{new}$ generated by the matrix

$$\mathcal{B}_{CS,\phi}^{new} = \begin{pmatrix} I_{2n^2} & \phi(\mathcal{H}) \\ 0 & \mathcal{B}_{private,\phi} \end{pmatrix}, \tag{61}$$

where $\mathcal{B}_{private,\phi} = \begin{pmatrix} D' & 0 \\ 0 & D' \end{pmatrix}$. However, since

$$det(\mathcal{B}_{CS,\phi}^{new}) = det(D')^2 = q^{2(n^2-|T|)} < q^{2n^2} = det(\mathcal{B}_{CS,\phi}).$$

Therefore, solving SVP in the lattice $\mathcal{L}_{CS,\phi}^{new}$ is costlier than $\mathcal{L}_{CS,\phi}$ as reflected in our experiments. For instance, for the parameter set with $n = 7$ and $q = 547$, the average blocksize to find a decryption key increases from 7 (for the old key generation process, Algorithm 1) to 9 (for the modified key generation, Algorithm 2). Similarly for $n = 7$ and $q = 113$, the average blocksize increases from 52.6 to 56. This increment makes the key attack slightly higher but does not thwart the practicality of our folding technique.

7.2 Updating parameters

The other straightforward way to avoid potential attacks is by increasing the parameter size. For example, one could allow for a larger set T to increase the search cost. However, as discussed in [5], and also experimentally observed, that $|T| \leq n$ is essential for successful decryption. Another possible attempt could be to increase the value of n, which will increase both the search and the lattice reduction cost. We searched for the BQTRU parameter set achieving the first level, i.e., 128-bit security, as suggested by NIST, that allows for no more than 2^{-128} decryption failure probability. Considering our key and message attack, $n = 17$ with $q = 919$ is the smallest prime that reaches the desired requirements. However, the issue with BQTRU is that the attacker needs to solve CVP in an n^2-dimensional ($n^2 = 289$, for $n = 17$) lattice to decrypt. Even with the best algorithms for solving CVP in this dimension, the cost of decryption is prohibitively high, making it impractical as a cryptosystem.

8 Concluding remarks

The proposal of building a cryptosystem based on the hardness of two prominent post-quantum families (lattice and code-based) is undoubtedly interesting. BQTRU has been introduced as a possibly practical scheme after a few attempts to build such construction based on the hardness of solving the SVP in hybrid lattices. In principle, choosing quaternion algebra was a smart choice to make the decryption practical by mapping the problem of solving CVP in a $4n^2-$dimensional private lattice to 4 instances of CVP in an n^2-dimensional lattice. Further, fast multiplication methods [41] that enable faster multiplication in quaternion algebra defined over $\left(\frac{1,1}{R}\right)$ was an additional reason for the authors of BQTRU to consider their construction. Our analysis indicates that the same reason that allowed for faster multiplication and feasible decryption in the chosen structure made the construction susceptible to a dimension folding attack.

This work demonstrates the effectiveness of the proposed attack both theoretically and experimentally. Consequently, we were able to compromise the moderate BQTRU parameter set and show that higher parameter sets offer much lower security levels than claimed. Further, our few attempts to fix BQTRU in its current form show the possibility of extending our folding attack to the modified key generation or that the scheme is yielding an impractical construction for secure parameter sets.

As a result, creating a secure yet practical cryptosystem based on the hardness of the SVP in a hybrid lattice remains an open research problem. One future direction is to explore different structures that are not vulnerable to folding attacks or to build a trapdoor that enables solving the CVP easily in the decryption process for the party possessing the trapdoor information.

A Group rings

For a ring R and a finite group $G = \{g_i : i = 1, 2, \ldots, n\}$ of order n, the group ring of G over R is the set of formal sums

$$RG = \left\{ a = \sum_{i=1}^{n} \alpha_{g_i} g_i : \alpha_{g_i} \in R \text{ for } i = 1, 2, \ldots, n \right\}, \tag{62}$$

that forms a ring under the following operations: let $a = \sum_{i=1}^{n} \alpha_{g_i} g_i$ and $b = \sum_{i=1}^{n} \beta_{g_i} g_i$ in RG then the sum of a and b is defined as:

$$a + b = \sum_{i=1}^{n} \alpha_{g_i} g_i + \sum_{i=1}^{n} \beta_{g_i} g_i = \sum_{i=1}^{n} (\alpha_{g_i} + \beta_{g_i}) g_i \,, \tag{63}$$

and the product of a and b as:

$$a * b = \left(\sum_{i=1}^{n} \alpha_{g_i} g_i \right) * \left(\sum_{i=1}^{n} \beta_{g_i} g_i \right) = \sum_{i=1}^{n} \gamma_{g_i} g_i \,, \tag{64}$$

where

$$\gamma_{g_k} = \sum_{g_i g_j = g_k} \alpha_{g_i} \beta_{g_j} = \sum_{i=1}^{n} \alpha_{g_i} \beta_{{g_i}^{-1} g_k} = \sum_{i=1}^{n} \alpha_{g_k {g_i}^{-1}} \beta_{g_i}. \tag{65}$$

For each element $a = \sum_{i=1}^{n} \alpha_{g_i} g_i \in RG$, we associate a unique coefficient vector $\boldsymbol{a} = (\alpha_{g_1}, \alpha_{g_2}, \ldots, \alpha_{g_n})$. We use a and $\boldsymbol{a}$ interchangeably to refer to an element of group ring RG. In vector notation

$$\boldsymbol{a} + \boldsymbol{b} = (\alpha_{g_1} + \beta_{g_1}, \alpha_{g_2} + \beta_{g_2}, \ldots, \alpha_{g_n} + \beta_{g_n}), \;\; \boldsymbol{a} \star \boldsymbol{b} = (\gamma_{g_1}, \gamma_{g_2}, \ldots, \gamma_{g_n})$$

where γ_{g_i}, for $i = 1, 2, \ldots, n$, are given by (65), denote coordinatewise addition and the convolutional product of two vectors $\boldsymbol{a}, \boldsymbol{b} \in RG$, respectively. Using Equation (65), we have

$$\boldsymbol{a} \star \boldsymbol{b} = (\alpha_{g_1}, \alpha_{g_2}, \ldots, \alpha_{g_n}) \star \begin{pmatrix} \beta_{g_1^{-1} g_1} & \beta_{g_1^{-1} g_2} & \ldots\ldots & \beta_{g_1^{-1} g_n} \\ \beta_{g_2^{-1} g_1} & \beta_{g_2^{-1} g_2} & \ldots\ldots & \beta_{g_2^{-1} g_n} \\ \vdots & \vdots & \ddots & \vdots \\ \beta_{g_n^{-1} g_1} & \beta_{g_n^{-1} g_2} & \ldots\ldots & \beta_{g_n^{-1} g_n} \end{pmatrix} \tag{66}$$

$$(\boldsymbol{a} \star \boldsymbol{b})^{Tr} = \begin{pmatrix} \alpha_{g_1 g_1^{-1}} & \alpha_{g_1 g_2^{-1}} & \ldots\ldots & \alpha_{g_1 g_n^{-1}} \\ \alpha_{g_2 g_1^{-1}} & \alpha_{g_2 g_2^{-1}} & \ldots\ldots & \alpha_{g_2 g_n^{-1}} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{g_n g_1^{-1}} & \alpha_{g_n g_2^{-1}} & \ldots\ldots & \alpha_{g_n g_n^{-1}} \end{pmatrix} \star \begin{pmatrix} \beta_{g_1} \\ \beta_{g_2} \\ \vdots \\ \beta_{g_n} \end{pmatrix}. \tag{67}$$

Definition 13 (RG-matrices). *[22] For an element $\boldsymbol{a} = (\alpha_{g_1}, \alpha_{g_2}, \ldots, \alpha_{g_n}) \in RG$, define the $RG-$matrices of $\boldsymbol{a}$ in $M_n(R)$ as follows:*

$$A = \begin{pmatrix} \alpha_{g_1^{-1} g_1} & \alpha_{g_1^{-1} g_2} & \ldots\ldots & \alpha_{g_1^{-1} g_n} \\ \alpha_{g_2^{-1} g_1} & \alpha_{g_2^{-1} g_2} & \ldots\ldots & \alpha_{g_2^{-1} g_n} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{g_n^{-1} g_1} & \alpha_{g_n^{-1} g_2} & \ldots\ldots & \alpha_{g_n^{-1} g_n} \end{pmatrix}, \; A' = \begin{pmatrix} \alpha_{g_1 g_1^{-1}} & \alpha_{g_1 g_2^{-1}} & \ldots\ldots & \alpha_{g_1 g_n^{-1}} \\ \alpha_{g_2 g_1^{-1}} & \alpha_{g_2 g_2^{-1}} & \ldots\ldots & \alpha_{g_2 g_n^{-1}} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{g_n g_1^{-1}} & \alpha_{g_n g_2^{-1}} & \ldots\ldots & \alpha_{g_n g_n^{-1}} \end{pmatrix}.$$

Lemma 4. *For* $\boldsymbol{a} = (\alpha_{g_1}, \alpha_{g_2}, \ldots, \alpha_{g_n}), \boldsymbol{b} = (\beta_{g_1}, \beta_{g_2}, \ldots, \beta_{g_n}) \in RG$*, the following hold:*

$$\boldsymbol{a} \star \boldsymbol{b} = \boldsymbol{a} \star B \quad \textit{and} \quad (\boldsymbol{a} \star \boldsymbol{b})^{Tr} = A' \star \boldsymbol{b}^{Tr}. \tag{68}$$

Further, if G *is abelian group then* $\boldsymbol{a} \star \boldsymbol{b} = \boldsymbol{b} \star A$.

Proof. The first part of the proof immediately follows from Equations (66) and (67). The other part follows from the observation that if G is an abelian group, then $A = (A')^{Tr}$. □

Theorem 3. *[22, Thereom 1] The mapping* $\tau : RG \to M_n(R)$ *defined as* $\tau(\boldsymbol{a}) = A$ *is a ring homomorphism, i.e.,* $\tau(\boldsymbol{a} + \boldsymbol{b}) = A + B$ *and* $\tau(\boldsymbol{a} \star \boldsymbol{b}) = A \star B$*, where* $+, \star$ *denote the usual matrix addition and multiplication, respectively.*

Example 1. Suppose $v(x) = v_0 + v_1 x + \ldots + v_{n-1}x^{n-1} \in \mathbb{Z}[x]/\langle x^n - 1\rangle$. Let $G = \langle x : x^n = 1\rangle$ be a cyclic group of order n, then $\mathbb{Z}[x]/\langle x^n - 1\rangle \cong \mathbb{Z}G$. The matrix representation of the vector $\boldsymbol{v} = (v_0, v_1, \ldots, v_{n-1})$ associated with the element $v(x)$ is a circulant matrix whose first row is the vector $\boldsymbol{v}$, i.e.,

$$V = \begin{pmatrix} v_0 & v_1 & \ldots & v_{n-1} \\ v_{n-1} & v_0 & \ldots & v_{n-2} \\ \vdots & \vdots & \ddots & \vdots \\ v_1 & v_2 & \ldots & v_0 \end{pmatrix} \in M_n(\mathbb{Z}). \tag{69}$$

Example 2. Let $G = \langle x, y : x^n = 1, y^n = 1, xy = yx\rangle$ be a group of order n^2, then $R = \mathbb{Z}[x, y]/\langle x^n - 1, y^n - 1\rangle \cong \mathbb{Z}G$. We can express every element of ring R as

$$v(x, y) = v_0(x) + yv_1(x) + y^2v_2(x) + \ldots + y^{n-1}v_{n-1}(x),$$

where each $v_i(x) \in \mathbb{Z}[x]/\langle x^n - 1\rangle$. Then, the coefficient vector of $v(x, y)$ is $\boldsymbol{v} = (\boldsymbol{v}_0, \boldsymbol{v}_1, \ldots, \boldsymbol{v}_{n-1}) \in \mathbb{Z}^{n^2}$, where $\boldsymbol{v}_i \in \mathbb{Z}^n$ is the coefficient vector of $v_i(x)$, and the matrix representation of $\boldsymbol{v}$ has the form

$$V = \begin{pmatrix} V_0 & V_1 & \ldots & V_{n-1} \\ V_{n-1} & V_0 & \ldots & V_{n-2} \\ \vdots & \vdots & \ddots & \vdots \\ V_1 & V_2 & \ldots & V_0 \end{pmatrix} \in M_{n^2}(\mathbb{Z}), \tag{70}$$

where $V_i \in M_n(\mathbb{Z})$ is the matrix representation of $\boldsymbol{v}_i$ and $V' = V^{Tr}$.

B Multiplication in $\mathbb{A} = \left(\frac{\mathbf{1,1}}{\boldsymbol{R}}\right)$

For two quaternions, $f = f_0 + f_1 i + f_2 j + f_3 k, g = g_0 + g_1 i + g_2 j + g_3 k \in \mathbb{A}$, consider the product

$$\begin{aligned} f * g =& (f_0g_0 + f_1g_1 + f_2g_2 - f_3g_3) + (f_1g_0 + f_0g_1 + f_3g_2 - f_2g_3)i + \\ & (f_2g_0 - f_3g_1 + f_0g_2 + f_1g_3)j + (f_3g_0 - f_2g_1 + f_1g_2 + f_0g_3)k \end{aligned} \tag{71}$$

Using Lemma 4, the coefficient vector of the product $f * g$ is given by

$$\boldsymbol{f} \star \boldsymbol{g} = (\boldsymbol{f}_0, \boldsymbol{f}_1, \boldsymbol{f}_2, \boldsymbol{f}_3) \star \begin{pmatrix} G_0 & G_1 & G_2 & G_3 \\ G_1 & G_0 & G_3 & G_2 \\ G_2 & -G_3 & G_0 & -G_1 \\ -G_3 & G_2 & -G_1 & G_0 \end{pmatrix}, \tag{72}$$

$$(\boldsymbol{f} \star \boldsymbol{g})^{Tr} = \begin{pmatrix} F'_0 & F'_1 & F'_2 & -F'_3 \\ F'_1 & F'_0 & F'_3 & -F'_2 \\ F'_2 & -F'_3 & F'_0 & F'_1 \\ F'_3 & -F'_2 & F'_1 & F_0 \end{pmatrix} \star \begin{pmatrix} \boldsymbol{g}_0^{Tr} \\ \boldsymbol{g}_1^{Tr} \\ \boldsymbol{g}_2^{Tr} \\ \boldsymbol{g}_3^{Tr} \end{pmatrix}. \tag{73}$$

where $G_i, F'_i \in M_{n^2}(\mathbb{Z})$ are the matrix representations of g_i, f_i as defined in Equation (70), respectively.

Definition 14 (Quaternion matrices). *For a quaternion $f = f_0 + f_1 i + f_2 j + f_3 k \in \mathbb{A}$, define the matrix representations of f in $M_{4n^2}(\mathbb{Z})$ as follows:*

$$\mathcal{F} = \begin{pmatrix} F_0 & F_1 & F_2 & F_3 \\ F_1 & F_0 & F_3 & F_2 \\ F_2 & -F_3 & F_0 & -F_1 \\ -F_3 & F_2 & -F_1 & F_0 \end{pmatrix}, \ \mathcal{F}' = \begin{pmatrix} F'_0 & F'_1 & F'_2 & -F'_3 \\ F'_1 & F'_0 & F'_3 & -F'_2 \\ F'_2 & -F'_3 & F'_0 & F'_1 \\ F'_3 & -F'_2 & F'_1 & F_0 \end{pmatrix}. \tag{74}$$

where $F_i, F'_i \in M_{n^2}(\mathbb{Z})$ are the matrix representations of f_i as defined in Equation (70).

Lemma 5. *For two quaternions $f = f_0 + f_1 i + f_2 j + f_3 k, g = g_0 + g_1 i + g_2 j + g_3 k \in \mathbb{A}$, the following hold:*

$$\boldsymbol{f} \star \boldsymbol{g} = \boldsymbol{f} \star \mathcal{G} \quad \textit{and} \quad (\boldsymbol{f} \star \boldsymbol{g})^{Tr} = \mathcal{F}' \star \boldsymbol{b}^{Tr}. \tag{75}$$

Further, if $\tilde{\mathcal{F}} = (\mathcal{F}')^{Tr}$ then $\boldsymbol{f} \star \boldsymbol{g} = \boldsymbol{g} \star \tilde{\mathcal{F}}$.

Proof. The proof immediately follows from Equations (72) and (73). □

Note: We would like to point out that in [5], the matrix representation of the quaternions in $\mathbb{A}$ is incorrect due to the wrong multiplication in [5, Equation 16].

C Proof of Lemma 2

Let $\mathcal{F}$ and $\mathcal{G}$ be matrices of elements f and $g \in \mathbb{A}$, respectively. Then,

$$\begin{aligned} \phi(\mathcal{F} + \mathcal{G}) &= \phi \begin{pmatrix} F_0 + G_0 & F_1 + G_1 & F_2 + G_2 & F_3 + G_3 \\ F_1 + G_1 & F_0 + G_0 & F_3 + G_3 & F_2 + G_2 \\ F_2 + G_2 & -F_3 - G_3 & F_0 + G_0 & -F_1 - G_1 \\ -F_3 - G_3 & F_2 + G_2 & -F_1 - G_1 & F_0 + G_0 \end{pmatrix} \\ &= \begin{pmatrix} F_0 + G_0 + F_1 + G_1 & F_2 + G_2 + F_3 + G_3 \\ F_2 + G_2 - F_3 - G_3 & F_0 + G_0 - F_1 - G_1 \end{pmatrix} \\ &= \begin{pmatrix} F_0 + F_1 & F_2 + F_3 \\ F_2 - F_3 & F_0 - F_1 \end{pmatrix} + \begin{pmatrix} G_0 + G_1 & G_2 + G_3 \\ G_2 - G_3 & G_0 - G_1 \end{pmatrix} \\ &= \phi(\mathcal{F}) + \phi(\mathcal{G}). \end{aligned}$$

Similarly, one can verify that $\phi(\mathcal{F} \star \mathcal{G}) = \phi(\mathcal{F}) \star \phi(\mathcal{G})$. Therefore, ϕ is a matrix ring homomorphism.

References

1. Albrecht, M.R., Curtis, B.R., Deo, A., Davidson, A., Player, R., Postlethwaite, E.W., Virdia, F., Wunderer, T.: Estimate all the {LWE, NTRU} schemes! In: Security and Cryptography for Networks. pp. 351–367. Springer International Publishing, Cham (2018), https://doi.org/10.1007/978-3-319-98113-0_19
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: Ishai, Y., Rijmen, V. (eds.) Advances in Cryptology – EUROCRYPT 2019. pp. 717–746. Springer International Publishing, Cham (2019), https://doi.org/10.1007/978-3-030-17656-3_25
3. Alkim, E., Ducas, L., Pöppelmann, T., Schwabe, P.: Post-quantum key {Exchange—A} new hope. In: 25th USENIX Security Symposium (USENIX Security 16). pp. 327–343 (2016)
4. Aono, Y., Wang, Y., Hayashi, T., Takagi, T.: Improved progressive bkz algorithms and their precise cost estimation by sharp simulator. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 789–819. Springer (2016). https://doi.org/10.1007/978-3-662-49890-3_30
5. Bagheri, K., Sadeghi, M.R., Panario, D.: A non-commutative cryptosystem based on quaternion algebras. Designs, Codes and Cryptography **86** (10 2018). https://doi.org/10.1007/s10623-017-0451-4
6. Bai, S., Jangir, H., Ngo, T., Youmans, W.: An algebraic algorithm for breaking ntru with multiple keys. Designs, Codes and Cryptography pp. 1–24 (2024), https://doi.org/10.1007/s10623-024-01473-z
7. Becker, A., Ducas, L., Gama, N., Laarhoven, T.: New directions in nearest neighbor searching with applications to lattice sieving. In: Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms. pp. 10–24. SIAM (2016). https://doi.org/10.1137/1.9781611974331.ch2
8. Boschini, C., Orsini, E., Traverso, C.: Between codes and lattices: Hybrid lattices and the NTWO cryptosystem. Proc. Effective Methods Algebr. Geometry (2015), http://people.cs.bris.ac.uk/~cseao/papr/MEGA2015.pdf
9. Caboara, M., Caruso, F., Traverso, C.: Gröbner bases for public key cryptography. In: Proceedings of the International Symposium on Symbolic and Algebraic Computation, ISSAC. pp. 315–324 (2008). https://doi.org/10.1145/1390768.1390811
10. Chen, C., Danba, O., Hoffstein, J., Hülsing, A., Rijneveld, J., Schanck, J.M., Saito, T., Schwade, P.S., Whyte, W.W., Xagawa, K.X., Yamakawa, T., Zhang, Z.: PQC round-3 candidate: NTRU. technical report. Tech. rep., NTRU Cryptosystems Technical Report No.11, Version 2, March 2001. Report (2019), https://ntru.org/f/ntru-20190330.pdf
11. Chen, Y.: Réduction de réseau et sécurité concrète du chiffrement complètement homomorphe. Ph.D. thesis, l'Université Paris Diderot (2013), http://www.theses.fr/2013PA077242
12. Chen, Y., Nguyen, P.Q.: BKZ 2.0: Better lattice security estimates. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 1–20. Springer (2011). https://doi.org/10.1007/978-3-642-25385-0_1

13. Coppersmith, D., Shamir, A.: Lattice Attacks on NTRU. In: Advances in Cryptology — EUROCRYPT '97. pp. 52–61. Springer Berlin Heidelberg, Berlin, Heidelberg (1997). https://doi.org/10.1007/3-540-69053-0_5
14. Dachman-Soled, D., Ducas, L., Gong, H., Rossi, M.: Lwe with side information: Attacks and concrete security estimation. In: Micciancio, D., Ristenpart, T. (eds.) Advances in Cryptology – CRYPTO 2020. pp. 329–358. Springer International Publishing, Cham (2020). https://doi.org/10.1007/978-3-030-56880-1_12
15. Ducas, L., van Woerden, W.: NTRU Fatigue: How stretched is overstretched? Cryptology ePrint Archive, Paper 2021/999 (2021), https://eprint.iacr.org/2021/999
16. Fincke, U., Pohst, M.: Improved methods for calculating vectors of short length in a lattice, including a complexity analysis. Mathematics of computation **44**(170), 463–471 (1985)
17. Gentry, C.: Key recovery and message attacks on NTRU-composite. In: Pfitzmann, B. (ed.) Advances in Cryptology — EUROCRYPT 2001. pp. 182–194. Springer Berlin Heidelberg, Berlin, Heidelberg (2001), https://doi.org/10.1007/3-540-44987-6_12
18. Herold, G., Kirshanova, E., Laarhoven, T.: Speed-ups and Time–Memory Trade-Offs for Tuple Lattice Sieving. In: Public-Key Cryptography – PKC 2018. pp. 407–436. Springer International Publishing, Cham (2018). https://doi.org/10.1007/978-3-319-76578-5_14
19. Hoffstein, J., Pipher, J., Silverman, J.H.: NTRU: A ring-based public key cryptosystem. In: International algorithmic number theory symposium. pp. 267–288. Springer, Berlin, Heidelberg (1998). https://doi.org/10.1007/BFb0054868
20. Hoffstein, J., Silverman, J.H., Whyte, W.: Meet-in-the-middle attack on an NTRU private key. Tech. rep., Technical report, NTRU Cryptosystems, July 2006. Report (2006)
21. Howgrave-Graham, N., Nguyen, P.Q., Pointcheval, D., Proos, J., Silverman, J.H., Singer, A., Whyte, W.: The impact of decryption failures on the security of ntru encryption. In: Boneh, D. (ed.) Advances in Cryptology - CRYPTO 2003. pp. 226–246. Springer Berlin Heidelberg, Berlin, Heidelberg (2003). https://doi.org/10.1007/978-3-540-45146-4_14
22. Hurley, T.: Group rings and rings of matrices. International Journal of Pure and Applied Mathematics **31**, 319–335 (01 2006), https://www.researchgate.net/publication/228928727_Group_rings_and_rings_of_matrices
23. Hülsing, A., Rijneveld, J., Schanck, J., Schwabe, P.: High-speed key encapsulation from NTRU. In: International Conference on Cryptographic Hardware and Embedded Systems, CHES 2017. pp. 232–252 (2017). https://doi.org/10.1007/978-3-319-66787-4_12
24. Kannan, R.: Improved algorithms for integer programming and related lattice problems. In: Proceedings of the fifteenth annual ACM symposium on Theory of computing. pp. 193–206 (1983), https://doi.org/10.1145/800061.808749
25. Kim, J., Lee, C.: A polynomial time algorithm for breaking NTRU encryption with multiple keys. Designs, Codes and Cryptography pp. 1–11 (2023)
26. Kirchner, P., Fouque, P.A.: Revisiting lattice attacks on overstretched NTRU parameters. In: Advances in Cryptology – EUROCRYPT 2017. pp. 3–26. Springer International Publishing (2017). https://doi.org/10.1007/978-3-319-56620-7_1
27. Kirshanova, E., May, A., Nowakowski, J.: New NTRU records with improved lattice bases. In: Johansson, T., Smith-Tone, D. (eds.) Post-Quantum Cryptography. pp. 167–195. Springer Nature Switzerland, Cham (2023)

28. Kumar, V., Das, R., Gangopadhyay, A.K.: GR-NTRU: Dihedral group over ring of Eisenstein integers. Journal of Information Security and Applications **83**, 103795 (2024). https://doi.org/https://doi.org/10.1016/j.jisa.2024.103795
29. Kumar, V., Raya, A., Gangopadhyay, A.K., Gangopadhyay, S., Hussain, M.T.: An efficient noncommutative NTRU from semidirect product. Cryptology ePrint Archive, Paper 2024/1721 (2024), https://eprint.iacr.org/2024/1721
30. Laarhoven, T.: Search problems in cryptography: from fingerprinting to lattice sieving. Phd thesis, Eindhoven University of Technology (2015), available at https://research.tue.nl/en/publications/search-problems-in-cryptography-from-fingerprinting-to-lattice-si
31. Lenstra, A.K., Lenstra, H.W., Lovász, L.: Factoring polynomials with rational coefficients. Mathematische annalen **261**(ARTICLE), 515–534 (1982). https://doi.org/10.1007/BF01457454
32. Ling, C., Mendelsohn, A.: NTRU in quaternion algebras of bounded discriminant. In: Post-Quantum Cryptography. pp. 256–290. Springer Nature Switzerland (2023). https://doi.org/10.1007/978-3-031-40003-2_10
33. Maclachlan, C., Reid, A.W.: Arithmetic Hyperbolic 3-Manifolds and Orbifolds, pp. 275–304. Springer New York (2003), https://doi.org/10.1007/978-1-4757-6720-9_10
34. Malekian, E., Zakerolhosseini, A., Mashatan, A.: QTRU : a lattice attack resistant version of NTRU PKCS based on quaternion algebra. IACR Cryptology ePrint Archive **386** (2009), https://eprint.iacr.org/2009/386
35. Micciancio, D., Walter, M.: Fast lattice point enumeration with minimal overhead. In: Proceedings of the twenty-sixth annual ACM-SIAM symposium on Discrete algorithms. pp. 276–294. SIAM (2014)
36. Peikert, C.: A decade of lattice cryptography. Foundations and trends® in theoretical computer science **10**(4), 283–424 (2016)
37. Raya, A., Kumar, V., Gangopadhyay, S.: DiTRU: A Resurrection of NTRU over Dihedral Group. In: Progress in Cryptology - AFRICACRYPT 2024. pp. 349–375. Springer Nature Switzerland (2024). https://doi.org/10.1007/978-3-031-64381-1_16
38. Raya, A., Kumar, V., Gangopadhyay, S., Gangopadhyay, A.K.: Results on the key space of group-ring NTRU: The case of the dihedral group. In: Security, Privacy, and Applied Cryptography Engineering. pp. 1–19 (2024). https://doi.org/10.1007/978-3-031-51583-5_1
39. Schnorr, C.P.: A hierarchy of polynomial time lattice basis reduction algorithms. Theoretical computer science **53**(2-3), 201–224 (1987). https://doi.org/10.1016/0304-3975(87)90064-8
40. Silverman, J.: Wraps, gaps, and lattice constants. Tech. rep., NTRU Cryptosystems Technical Report No.11, Version 2, March 2001. Report (2001), https://ntru.org/f/tr/tr011v2.pdf
41. Strassen, V.: Gaussian elimination is not optimal. Numerische mathematik **13**(4), 354–356 (1969), https://doi.org/10.1007/BF02165411
42. development team, T.F.: fplll, a lattice reduction library, Version: 5.4.4 (2023), available at https://github.com/fplll/fplll
43. development team, T.F.: fpylll, a Python wraper for the fplll lattice reduction library, Version: 0.5.9 (2023), available at https://github.com/fplll/fpylll