Message Recovery in NTRU Encrypt based on CVP

Marios Adamoudis[1], Konstantinos A. Draziotis[2] and Eirini Poimenidou[3⋆]

[1] marios.p7@hotmail.com
[2] drazioti@csd.auth.gr
School of Informatics
Aristotle University of Thessaloniki,
[3] epoimeni@csd.auth.gr
School of Informatics
Aristotle University of Thessaloniki,
& CERN

Abstract. In the present paper, we implement a message recovery attack on the NTRU-HPS cryptosystem using its state-of-the-art parameters. We make the assumption that the first and second most significant bits (MSB) of the polynomial $u(x)$, which is a multiple of the ephemeral key $r(x)$, are known and using Babai's nearest plane algorithm we successfully recover the message. Additionally, we discuss a possibility of a side-channel attack method designed to extract the necessary bit information from the cryptographic operations.
MSC 2010: 94A60, 11T71, 11Y16.

Keywords: Public Key Cryptography; NTRU; Closest Vector Problem; LLL algorithm; Babai's Nearest Plane Algorithm.

1 Introduction

The NTRU cryptosystem was developed in 1996 by Hoffstein, Pipher, and Silverman [10]. To encrypt and decrypt data, NTRU makes use of lattice-based cryptography. The two algorithms that make up this system are NTRUSign for digital signatures and NTRUEncrypt for encryption. Notably, NTRU seems immune to quantum attacks, whereas RSA and Diffie-Hellman are vulnerable to Shor's quantum attack [28]. Compared to RSA, NTRU completes private-key operations substantially more quickly. NTRU became a finalist in the 3rd round of the Post-Quantum Cryptography Standardization project but NIST will not

⋆ Corresponding author.
This work was co funded by SECURE-EU. The SECUR-EU project funded under Grant Agreement 101128029 is supported by the European Cybersecurity Competence Centre.
The paper will be presented in nutmic2024

standardize it [21]. In May 2016, Daniel Bernstein, Chitchanok Chuengsatiansup, Tanja Lange, and Christine van Vredendaal introduced NTRU Prime. As of August 2022, starting from version 9.0, OpenSSH employs NTRU in conjunction with the X25519 Elliptic Curve Diffie-Hellman (ECDH) key exchange as its default configuration [31]. GoldBug Messenger [27] holds the distinction of being the pioneer chat and email client to incorporate the NTRU algorithm under an open-source license. This implementation is rooted in the Spot-On Encryption Suite Kernels. Another implementation is in wolfSSL, which supports NTRU cipher [32].

In the present work, we describe an attack on NTRU based on [2, 24]. First, we multiply the encryption equation by a positive integer k and consider the polynomial $u(x) = -kh(x) * r(x)$ where $h(x)$ is the public key and $r(x)$ is the ephemeral key. We construct a lattice-based attack, taking as a hypothesis that we know the binary length of each u_i, where $u(x) = u_{N-1}x^{N-1} + \cdots + u_0$. In the case where this binary length equals to $\text{bits}(q) - 1$, we need to know the second most significant bit of u_i. This kind of assumption is quite common, for example, the authors in [12] attack the DSA signature, assuming that a proportion of the bits of each of the associated ephemeral keys can be recovered. By this assumption, we succeeded in enhancing the similar attack presented in [2], see Remark 2. The lattice we are using is the same type as in [24], but a main difference is that there, the authors selected a small k in order to create a Voronoi First Kind (VFK) type lattice, while here we let k take larger values. Additionally, we explain our choice of k.

1.1 Roadmap

In Section 2, we present the previous work related to NTRU attacks based on lattices as well as some information about side-channel attacks. In Section 3, we provide the fundamental lattice theory that is necessary for understanding our attack. In Section 4, we present the NTRU cryptosystem. In Section 5, we describe in detail our attack. Finally, in Section 6 we summarize our results. Furthermore, in Appendix A, we prove a Theorem that provides the length of the shortest vector in a specific lattice.

Our work's corresponding implementation can be found at
https://github.com/drazioti/ntru_cvp_conf.

2 Previous Work

The NTRU cryptographic system initially became the focus of a lattice attack in 1997, spearheaded by Coppersmith [6]. Subsequently, Gentry devised a potent approach, especially beneficial when the variable N is composite; refer to [8] for details on this method. In [19], May employed a distinct category of lattices known as run-lattices to address analogous challenges and provide solutions. Expanding upon May's concept, Silverman [29] introduced a method that entails

the selection of r coefficients while concurrently diminishing the lattice's dimension to force them to zero. This approach mirrors the strategy adopted by the researchers in [9], where decryption failures were utilized to reveil the secret key, provided that the decryption oracle supported such recovery. Alternative methods have been devised, including the transformation of the NTRU problem into a multivariate quadratic system over a finite field with two elements by employing Witt vectors, as outlined in [5, 30].

Odlyzko [14] introduced a meet-in-the-middle attack, which partitions the search space into two sub-spaces, leading to reduced time complexity. In a complementary approach, Howgrave's hybrid attack [11] integrates lattice reduction with a meet-in-the-middle algorithm. This hybrid method has been extensively tested by researchers to assess the security of lattice-based encryption techniques.

In their latest research, detailed in the publication [15], the authors introduce a novel strategy for addressing the most recent versions of the NTRU encryption scheme. Their approach entails the utilization of a meticulously designed lattice and the application of the BKZ algorithm in tandem with the lattice sieving algorithm from the G6K library. A pivotal aspect of their investigation centers on the substantial benefits gained by deviating from the conventional Coppersmith-Shamir lattice towards a basis grounded in the cyclotomic ring. This adjustment yields notable outcomes, exemplified by their achievement in decrypting the NTRU-HPS-171 instance within 83 core days using the cyclotomic ring basis, as opposed to the 172 core days required with the Coppersmith-Shamir basis. Furthermore, the authors take on an official NTRU challenge, specifically one featuring $N = 181$, posed by Security Innovation, Inc. Their approach proves successful in cracking this challenge within 20 core years.

To compromise the NTRU cryptosystem featuring a modulus higher than that specified in the NTRU-Encrypt standard, comparable techniques were independently suggested by Albrecht, Bai, and Ducas [3], as well as by Cheon, Jeong, and Lee [7]. Kirchner [13] illustrated that the time complexity becomes polynomial when q is set to $2^{\Omega(\sqrt{n \log \log n})}$ in the field $\mathbb{Q}(\zeta_{2^n})$. Finally, Nguyen [22] improved and elucidated the hybrid and meet-in-the-middle attacks. While the subfield attack variation introduced in this study surpasses previous methods, it is not better than the hybrid attack.

2.1 Side-channel attacks

Lattice-based cryptosystems are resistant to post-quantum computers in theory but in practice they are vulnerable to side-channel attacks (SDAs). Side-channel attacks exploit unintended information leakage from a cryptographic system by targeting weaknesses in the physical implementation of the algorithm or its execution environment. These attacks rely on observing measurable physical properties of the cryptographic device or system, such as power consumption [17, 18], electromagnetic radiation, timing information [16], or sound emanations.

In this paper we require information about the (unknown) polynomial $u(x) = -kh(x) * r(x) \mod (q, x^N - 1)$, more specifically we know the binary length ℓ_i of all the coefficients u_i of $u(x)$, which means we know the Most Significant Bit

(MSB) and for all u_i such that $\ell_i = \ell$, where $\ell = \text{bits}(q) - 1$, we also know the second MSB. We have two ideas on how these data can be acquired using side-channel attacks. The first one would be through a cold boot attack [23], where the attacker gets a noisy version of the polynomial $u(x)$ from the system's memory during a power-up/power-down cycle before $u(x)$ gets cleaned or overwritten by the system. Instead of getting the whole key like the authors of [23] do, we should be able to get candidates only for the MSB and the second MSB, for every u_i. Finally, the actual bits can be determined by verifying the correctness of the decryption operation for a known plaintext.

The second method for finding the previous MSBs, is from a scan-based side-channel attack [1], where one can find the corresponding locations of the flip-flops of $u(x)$ [1, Chapter III] in the scan chain and again, instead of recovering the whole key like the authors of the above-mentioned paper do, the attacker should deduce possible candidates for the first and second MSB. The correctness of the bits can be tested, as previous, through trial and error. This second method proves more difficult, since such an attack would require detailed knowledge of the circuit layout, including the design of the scan chains and the assignment of flip-flops to various data paths.

Side-channel attack techniques demand precise instrumentation and controlled environments to capture and analyze the subtle signals indicative of sensitive information and often require specialized equipment, therefore in this paper we will not be conducting the practical implementations of the side-channel attack. However, we encourage fellow researchers to explore and validate the theoretical analysis presented here. For the rest of the paper we assume that we have the information necessary to proceed with our attack.

3 Preliminaries on Lattices

In this section, we recall some well-known facts about lattices. In the field of cryptology, lattices play a central role.

3.1 Basic Definitions

Let $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n$ be linearly independent vectors of $\mathbb{R}^m$. The set

$$\mathcal{L} = \left\{ \sum_{j=1}^{n} \alpha_j \mathbf{b}_j : \alpha_j \in \mathbb{Z}, 1 \leq j \leq n \right\}$$

is called a *lattice* and the finite vector set $\mathcal{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ is called a basis of the lattice $\mathcal{L}$. All the bases of $\mathcal{L}$ have the same number of elements, i.e. in our case n, which is called *dimension* or *rank* of $\mathcal{L}$. If $n = m$, then the lattice $\mathcal{L}$ is said to have *full rank*. We consider M be the $n \times m$ matrix, having as rows the vectors $\mathbf{b}_1, \ldots, \mathbf{b}_n$. If $\mathcal{L}$ has full rank, then the *volume* of the lattice $\mathcal{L}$ is defined to be the positive number $|\det M|$. The volume, as well as the rank, are independent of the basis $\mathcal{B}$. It is denoted by $vol(\mathcal{L})$ or $\det \mathcal{L}$. Let now $\mathbf{v} \in \mathbb{R}^m$,

then $\|\mathbf{v}\|$ denotes the Euclidean norm of $\mathbf{v}$. Additionally, we denote by $\lambda_1(\mathcal{L})$ the least of the lengths of vectors of $\mathcal{L} - \{\mathbf{0}\}$. Finally, if $\mathbf{t} \in \text{span}(\mathbf{b}_1, ..., \mathbf{b}_n)$, then by $dist(\mathcal{L}, \mathbf{t})$, we denote $\min\{\|\mathbf{v} - \mathbf{t}\| : \mathbf{v} \in \mathcal{L}\}$.

3.2 Computation Problems on Lattices

Here we describe the fundamental problems on lattices.

The Shortest Vector Problem (SVP): Given a lattice $\mathcal{L}$ find a non zero vector $\mathbf{b} \in \mathcal{L}$ that minimize the (Euclidean) norm $\|\mathbf{b}\|$.

The Closest Vector Problem (CVP): Given a lattice $\mathcal{L}$ and a vector $\mathbf{t} \in \mathbb{R}^m$ that is not in $\mathcal{L}$, find a vector $\mathbf{b} \in \mathcal{L}$ that minimize the distance $\|\mathbf{b} - \mathbf{t}\|$.

The approximate Shortest Vector Problem (apprSVP): Given a lattice $\mathcal{L}$ and a function $f(n)$, find a non-zero vector $\mathbf{b} \in \mathcal{L}$, such that:

$$\|\mathbf{b}\| \leq f(n)\lambda_1(\mathcal{L}).$$

Each choice of function $f(n)$ gives a different apprSVP.

The approximate Closest Vector Problem (apprCVP): Given a lattice $\mathcal{L}$, a vector $\mathbf{t} \in \text{span}(\mathbf{b}_1, \ldots, \mathbf{b}_n)$ and a function $f(n)$, find a non-zero vector $\mathbf{b} \in \mathcal{L}$ such that,

$$\|\mathbf{b} - \mathbf{t}\| \leq f(n)dist(\mathcal{L}, \mathbf{t}).$$

Each choice of function $f(n)$ gives a different apprCVP.

3.3 Lattice Basis Reduction

The security of various cryptosystems is determined by the difficulty of solving appSVP or appCVP in different kinds of lattices. This section introduces the LLL algorithm, which finds a polynomial-time solution to the given problem. The LLL algorithm solves SVP rather well in small dimensions but performs poorly in large dimensions. The inability of LLL and other lattice reduction algorithms to effectively solve apprSVP and apprCVP determines the security of lattice-based cryptosystems.

Definition 1. *A basis $\mathcal{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ of a lattice $\mathcal{L}$ is called LLL-reduced if it satisfies the following conditions:*

1. $|\mu_{i,j}| = \frac{|\mathbf{b}_i \cdot \mathbf{b}_j^*|}{||\mathbf{b}_j^*||^2} \leq \frac{1}{2}$ *for every i, j with $1 \leq j < i \leq n$,*
2. $||\mathbf{b}_i^*||^2 \geq (\frac{3}{4} - \mu_{i,i-1}^2)||\mathbf{b}_{i-1}^*||^2$ *for every i with $1 < i \leq n$.*

Proposition 1. *Let $\mathcal{L}$ be a lattice of rank n. For every LLL-reduced basis $\mathcal{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ of a lattice $\mathcal{L}$, it is*

$$||\mathbf{b}_1|| \leq 2^{(n-1)/2}\lambda_1(\mathcal{L}).$$

Thus, an LLL-reduced basis solves the approximate SVP to within a factor of $2^{(n-1)/2}$.

The LLL Algorithm:

INPUT: A basis $\mathcal{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ of a lattice $\mathcal{L}$
OUTPUT: A LLL-reduced basis of$\mathcal{L}$

1. compute the G-S basis $\{\mathbf{b}_1^*, \ldots, \mathbf{b}_n^*\}$
2. set $k = 2$
3. $\mathbf{b}_1^* \leftarrow \mathbf{b}_1$
4. **while** $k \leq n$
5. **for** $j = k-1, \ldots, 1$
6. $\mathbf{b}_k \leftarrow \mathbf{b}_k - \lfloor \mu_{k,j} \rceil \mathbf{b}_j$
7. update $\mu_{k,j}$ for every j with $1 \leq j < k$.
8. **if** $||\mathbf{b}_k^*||^2 \geq (\frac{3}{4} - \mu_{k,k-1}^2)||\mathbf{b}_{k-1}^*||^2$
9. $k \leftarrow k+1$
10. **else**
11. swap $\mathbf{b}_{k-1}$ and $\mathbf{b}_k$
12. update $\mathbf{b}_k^*, \mathbf{b}_{k-1}^*, \mu_{k-1,j}, \mu_{k,j}$ for j with $1 \leq j < k$.
13. update $\mu_{i,k}, \mu_{i,k-1}$ for i with $k < i \leq n$.
14. $k \leftarrow \max(2, k-1)$
15. return the LLL-reduced basis $\{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$

The BKZ algorithm, first presented by Schnorr in [26] and its modifications are now the best lattice reduction algorithms in use for high dimensions. BKZ is a generalization of the LLL algorithm. The blocksize is the key BKZ parameter, and it has an impact on both the algorithm's execution time and the quality of the reduced basis.

3.4 Babai's Algorithm

To solve apprCVP, we usually use Babai's algorithm [4] (which has polynomial running time). In fact, combining this algorithm with the LLL algorithm, we can solve apprCVP for some lattice $\mathcal{L} \subset \mathbb{Z}^m$ having $f(n) = 2^{n/2}$ and $n = rank(\mathcal{L})$, in polynomial time. Below, we present the algorithm.

Babai's Nearest plane Algorithm

INPUT: A $n \times m$-matrix M with rows the vectors of a basis $\mathcal{B} = \{\mathbf{b}_i\}_{1 \leq i \leq n} \subset \mathbb{Z}^m$ of the lattice $\mathcal{L}$ and a vector $\mathbf{t} \in \mathbb{R}^m$
OUTPUT: $\mathbf{x} \in \mathcal{L}$ such that $||\mathbf{x} - \mathbf{t}|| \leq 2^{n/2} dist(\mathcal{L}, \mathbf{t})$.
1. $M^* = \{(\mathbf{b}_j^*)_j\} \leftarrow GSO(M)$ # GSO : Gram-Schmidt Orthogonalization
2. $\mathbf{b} \leftarrow \mathbf{t}$
3. **for** $j = n$ to 1
4. $c_j \leftarrow \left\lfloor \frac{\mathbf{b} \cdot \mathbf{b}_j^*}{||\mathbf{b}_j^*||^2} \right\rceil$ $\quad \# \lfloor x \rceil = \lfloor x + 0.5 \rfloor$

5. $\mathbf{b} \leftarrow \mathbf{b} - c_j \mathbf{b}_j$
6. return $\mathbf{t} - \mathbf{b}$.

If the rank of $\mathcal{L}$ is "quite" small, then we can solve the CVP with the deterministic algorithm of Micciancio-Voulgaris [20].

4 NTRU-HPS

Let the polynomial ring $\mathcal{R} = \mathbb{Z}[x]/\langle D(x)\rangle$ for some $D(x) \in \mathbb{Z}[x]$ and $\langle D(x)\rangle$ be the ideal generated by $D(x)$. We write $*$ for the multiplication in the ring $\mathcal{R}$. Also, fix a polynomial $h(x) \in \mathbb{Z}[x]$ of degree $N-1$. We set,

$$B_h = \begin{pmatrix} - \ h(x) \ - \\ - \ x * h(x) \ - \\ \vdots \\ - \ x^{N-1} * h(x) \ - \end{pmatrix}, \tag{1}$$

where with $x^i * h(x)$, we write the vector with coordinates the coefficients of the polynomial $h(x)$, after multiplication in $\mathcal{R}$ with x^i. In expressing the coefficient vector of $h(x) = a_{N-1}x^{N-1} + \cdots + a_0$, we denoted as $\mathbf{h} = (a_0, ..., a_{N-1}) \in \mathbb{Z}^N$. Then, the multiplication $g(x) * h(x)$ in $\mathcal{R}$ can be represented as the multiplication of the row matrix $[\mathbf{g}]$ and matrix B_h, i.e., $[\mathbf{g}]B_h$.

The set

$$\mathcal{L}_h = \{(f(x), g(x)) \in \mathcal{R}^2 : g(x) = f(x) * h(x)\}$$

is a lattice, where $h(x)$ has degree $N-1$. To see this we write,

$$\mathcal{L}_h = \mathbb{Z}^{2N} B'_h,$$

where B'_h is the block matrix,

$$\left[\frac{B_h}{qI_N}\right].$$

If we consider the previous lattice, but taking $\mod q$ (for some positive q), we get a (NTRU type) lattice

$$\mathcal{L}_h^q = \{(f(x), g(x)) \in \mathcal{R}^2 : g(x) = f(x) * h(x) \pmod q\},$$

where we also write it as,

$$\mathcal{L}_h^q = \{(\mathbf{f}, \mathbf{g}) \in \mathbb{Z}^{2N} : [\mathbf{g}] = [\mathbf{f}]M_h\}$$

with

$$M_h = \left[\frac{I_N \mid B_h}{\mathbf{0}_N \mid qI_N}\right].$$

The lattice $\mathcal{L}_h$ has several interesting properties when B_h is a cyclic matrix (see the matrix given by (1)). One example is when we choose $D(x) = x^N - 1$. In this

case, if $(\mathbf{a}, \mathbf{b})$ is a vector in the lattice, then performing a cyclic permutation of $\mathbf{a}$ and $\mathbf{b}$ $k-$times will result in another vector in the lattice. On the other hand, if $D(x) = x^p - x - 1$ (the case of NTRU-Prime), then B_h is not circulant.

Alice selects public parameters (N, p, q), with N and $p = 3$ being prime numbers, and both co-prime to q. Usually N and q are large, and q is a power of 2. We also assume that $D(x) = x^N - 1$.

We set

$-$ $\mathcal{R} = \mathbb{Z}[x]/\langle D(x)\rangle$, $\mathcal{R}/3 = \mathbb{Z}_3[x]/\langle D(x)\rangle$ and $\mathcal{R}/q = \mathbb{Z}_q[x]/\langle D(x)\rangle$.

$-$ $\mathcal{S} = \mathbb{Z}[x]/\langle \Phi_N(x)\rangle$, $\mathcal{S}/3 = \mathbb{Z}_3[x]/\langle \Phi_N(x)\rangle$ and $\mathcal{S}/q = \mathbb{Z}_q[x]/\langle \Phi_N(x)\rangle$, where $\Phi_N(x) = D(x)/\Phi_1(x) = x^{N-1} + x^{N-2} + \cdots + x + 1$.

Moreover, we define the set of ternary polynomials $\mathcal{T}_\alpha$ of degree α, as the set of polynomials with coefficients from the set $\{-1, 0, 1\}$ and degree at most α. With $\mathcal{T}(d_1, d_2) \subset \mathcal{R}$, we denote the polynomials of $\mathcal{R}$ with d_1 entries equal to one, d_2 entries equal to minus one and the remaining entries are zero.

We assume $q \leq 16N/3 + 16$ and we define the following sample spaces:
$-$ $\mathcal{L}_m = \mathcal{L}_g = \mathcal{T}_{N-2}(\frac{q}{16} - 1, \frac{q}{16} - 1)$,
$-$ $\mathcal{L}_f = \mathcal{L}_r = \mathcal{T}_{N-2}$.

Alice, for her private key randomly selects $(f(x), g(x))$ such that $f(x) \in \mathcal{L}_f$ and $g(x) \in \mathcal{L}_g$. It is important that $f(x)$ is invertible in both $\mathcal{S}/q$ and $\mathcal{S}/3$. The inverses in $\mathcal{S}/3$ and $\mathcal{S}/q$ can be efficiently computed using the Euclidean algorithm and Hensel's Lemma, see [10, Proposition 6.45]. Let $F_q(x)$ and $F_3(x)$ represent the inverses of $f(x)$ in $\mathcal{S}/q$ and $\mathcal{S}/3$, respectively.

Alice next computes

$$h(x) = 3F_q(x) \star g(x) \mod q.$$

The polynomial $h(x)$ is Alice's public key.

The problem of distinguishing $h(x)$ from uniform elements in $\mathcal{R}/q$ is called *decision NTRU problem.* While, the problem of finding the private key $(f(x), g(x))$ is referred to as the *search NTRU problem.* Bob's plaintext is a polynomial $m(x) \in \mathcal{R}$, whose coefficients are in the set $\{-1, 0, 1\}$. Thus, the plaintext $m(x)$ is the centerlift of a polynomial in $\mathcal{R}/3$. Bob chooses a random ephemeral key $r(x) \in \mathcal{L}_r$ and computes the ciphertext,

$$c(x) \equiv h(x) \star r(x) + m(x) \mod q. \tag{2}$$

Finally, Bob sends to Alice the ciphertext $c(x) \in \mathcal{R}/q$.

To decrypt, Alice follows the algorithm:

1. $a(x) \leftarrow c(x) * f(x) \mod (q, \Phi_1\Phi_N)$
2. $m(x) \leftarrow a(x) * f_3(x) \mod (3, \Phi_N)$
3. $m'(x) \leftarrow \text{Lift}_3(m(x))$
4. $r(x) \leftarrow (c(x) - m'(x))h_q(x) \mod (q, \Phi_N)$
5. **if** $(r(x), m(x)) \in \mathcal{L}_r \times \mathcal{L}_m$ **then**
6. **return** $(m(x), r(x), 0)$

7. else
8. return $(0, 0, 1)$

5 The attack

5.1 The general idea

We use the encryption equation (2),

$$c(x) = h(x) \star r(x) + m(x) \mod (q, x^N - 1).$$

Let k be a positive integer which we shall choose later. We multiply the previous equation by k and we set $b(x) = kc(x) \mod (q, x^N - 1)$ and $u(x) = -kh(x) * r(x) \mod (q, x^N - 1)$, then

$$km(x) = b(x) + u(x) \mod (q, x^N - 1). \tag{3}$$

Therefore,

$$km(x) = b(x) + u(x) + qv(x), \text{ for some polynomial } v(x).$$

Polynomials $m(x)$ and $u(x)$ are unknown. Let $\mathbf{m} = (m_i), \mathbf{b} = (b_i), \mathbf{u} = (u_i)$, and $\mathbf{v}$ be the vectors corresponding to $m(x), b(x), u(x)$, and $v(x)$, respectively. We set $\mathbf{V}$ to be the unknown vector $(-\mathbf{m}, \mathbf{u})$. We remark that $(-\mathbf{m}, \mathbf{b}+\mathbf{u})$ is in $\mathcal{L}_k$, where $\mathcal{L}_k$ is the lattice generated by the rows of the matrix

$$M_k = \left[\begin{array}{c|c} I_N & -kI_N \\ \hline \mathbf{0}_N & qI_N \end{array}\right]. \tag{4}$$

Indeed, if we consider $(-\mathbf{m}, -\mathbf{v}) \in \mathbb{Z}^{2N}$, then

$$(-\mathbf{m}, -\mathbf{v})M_k = (-\mathbf{m}, -\mathbf{v}) \left[\begin{array}{c|c} I_N & -kI_N \\ \hline \mathbf{0}_N & qI_N \end{array}\right] = (-\mathbf{m}, k\mathbf{m} - q\mathbf{v}) = (-\mathbf{m}, \mathbf{b} + \mathbf{u}).$$

Now, we want a nice approximation of the unknown vector $\mathbf{V}$. Assume that we can find a vector $\mathbf{E}' = (\mathbf{0}_N, \mathbf{E}) = (\mathbf{0}_N, E_0, ..., E_{N-1}) \in \mathbb{Z}^{2N}$ such that,

$$||\mathbf{V} - \mathbf{E}'|| < \frac{1}{2}\lambda_1, \text{ where } \lambda_1 \text{ is the length of a shortest vector in } \mathcal{L}_k. \tag{5}$$

Note that, neither $\mathbf{V}$ nor $\mathbf{E}'$ is in $\mathcal{L}_k$. We choose the target vector $\mathbf{t}$ through $\mathbf{E}$ as follows,

$$\mathbf{t} = (0, ..., 0, b_0 + E_0, ..., b_{N-1} + E_{N-1}) \in \mathbb{Z}^{2N},$$

and set $\mathbf{w} \leftarrow CVP(\mathcal{L}_k, \mathbf{t})$. We shall prove that $\mathbf{w}$ provides the message $\mathbf{m}$. First, we remark that

$$||\mathbf{w} - \mathbf{t}|| \leq ||(-\mathbf{m}, \mathbf{b} + \mathbf{u}) - \mathbf{t}||, \tag{6}$$

For instance if $m(x) = m_{N-1}x^{N-1} + m_{N-2}x^{N-2} + \cdots + m_1 x + m_0$ then $\mathbf{m} = (m_0, m_1, ..., m_{N-2}, m_{N-1})$. In this case $m_i \in \{-1, 0, 1\}$ and $m_{N-1} = 0$ since $\mathcal{L}_m = \mathcal{T}_{N-2}(\frac{q}{16} - 1, \frac{q}{16} - 1)$.

since $(-\mathbf{m}, \mathbf{b}+\mathbf{u}) \in \mathcal{L}_k$. Then,

$$\|(-\mathbf{m}, \mathbf{b}+\mathbf{u}) - \mathbf{t})\| =$$

$$= \|(-m_0, ..., -m_{N-1}, b_0+u_0, ..., b_{N-1}+u_{N-1}) - (0, ..., 0, E_0+b_0, ..., E_{N-1}+b_{N-1})\| =$$

$$= \|(-m_0, ..., -m_{N-1}, u_0 - E_0, ..., u_{N-1} - E_{N-1})\| =$$

$$= \|(-\mathbf{m}, \mathbf{u}) - \mathbf{E}'\| = ||\mathbf{V} - \mathbf{E}'\| < \frac{1}{2}\lambda_1.$$

Finally,

$$\|\mathbf{w} - (-\mathbf{m}, \mathbf{b}+\mathbf{u})\| = \|(\mathbf{w}-\mathbf{t}) + \big(\mathbf{t} - (-\mathbf{m}, \mathbf{b}+\mathbf{u})\big)\|$$

$$\leq \|\mathbf{w}-\mathbf{t}\| + \|\mathbf{t} - (-\mathbf{m}, \mathbf{b}+\mathbf{u})\| \leq 2\|(-\mathbf{m}, \mathbf{b}+\mathbf{u}) - \mathbf{t}\| < \lambda_1.$$

But $\mathbf{w} - (-\mathbf{m}, \mathbf{b}+\mathbf{u}) \in \mathcal{L}_k$, thus $\mathbf{w} = (-\mathbf{m}, \mathbf{b}+\mathbf{u})$. We conclude therefore that the first $N-$coordinates of $\mathbf{w}$ provide the message $\mathbf{m}$.

Remark 1. If $q > (k+1)\sqrt{k+1}$, then $\lambda_1(\mathcal{L}_k) = \sqrt{1+k^2}$. See Appendix A. In general $\lambda_1(\mathcal{L}_k) \geq \sqrt{1+k^2}$ since the first vector of the matrix M_k has Euclidean length $\sqrt{1+k^2}$.

5.2 Choosing E and k

Let $u(x) = u_{N-1}x^{N-1} + \cdots + u_1 x + u_0$ be as previous.
(Assumption – A). We assume that for each coefficient u_i we know ℓ_i such that $u_i \in [2^{\ell_i - 1}, 2^{\ell_i})$. I.e. u_i has binary length ℓ_i.
(Assumption – B). For all u_i such that $\ell_i = \ell$, where $\ell = \text{bits}(q) - 1$, we also know the second most significant bit, i.e. we know the z_i's such that $u_i = 2^{\ell-1} + z_i 2^{\ell-2} + \cdots$, for $i = 0, 1, ..., N-1$.

The previous two assumptions can be provided by an oracle which outputs the length of the coefficients (u_i) and in the case (u_i) has the maximum length, i.e. $\text{bits}(q)-1$, we also know the second most significant bit. We remark here that, in NTRU-HPS and their variants, q is a power of 2. For instance in ntruhps2048509 $q = 2048$, i.e. the minimum number with 11 bits. So taking $\mod q$ to the polynomials we get at most 10 bits numbers, that's why we have set $\ell = \text{bits}(q) - 1$ and not $\text{bits}(q)$. We consider the following two cases.

Case 1. $\ell_i = \text{bits}(u_i) = \ell$, then we set $E_i = 2^{\ell-1} + 2^{\ell-2} + 2^{\ell-3}$ if the second most significant bit is 1, else we set $E_i = 2^{\ell-1} + 2^{\ell-3}$.
Case 2. $\ell_i = \text{bits}(u_i) < \ell$, then we set $E_i = 2^{\ell_i - 1} + 2^{\ell_i - 2}$.

That is, if $u_j = x_j 2^{\ell-1} + y_j x^{\ell-2} + \cdots$, where $x_j, y_j \in \{0, 1\}$, then we set,

$$E_j = \begin{cases} 2^{\ell-1} + 2^{\ell-2} + 2^{\ell-3}, & \text{if } x_j = 1, y_j = 1 \\ 2^{\ell-1} + 2^{\ell-3}, & \text{if } x_j = 1,\ y_j = 0 \\ 2^{\ell_j - 1} + 2^{\ell_j - 2}, & \text{if } x_j = 0 \end{cases}$$

We get the following Lemma.

Lemma 1. *We have* $|u_j - E_j| \leq 2^{\ell-3} - 1$.

Proof. Lets see for instance the case $x_j = y_j = 1$. Then,

$$|u_j - E_j| = |(z_j - 1)2^{\ell-3} + \cdots|.$$

Since $z_j \in \{0, 1\}$ we get $|u_j - E_j| \leq 2^{\ell-4} + 2^{\ell-5} + \cdots + 2 + 1 = 2^{\ell-3} - 1$. Similar for the other two cases, we have

• $x_j = 1, y_j = 0$

$$|u_j - E_j| = |(2^{\ell-1} + z_j 2^{\ell-3} + \cdots) - (2^{\ell-1} + 2^{\ell-3}) = |(z_j - 1)2^{\ell-3} + \cdots| \leq 2^{\ell-3} - 1.$$

• $x_j = 0$. We remind that ℓ_j is the binary length of u_j.

$$|u_j - E_j| = |(2^{\ell_j - 1} + r_j 2^{\ell_j - 2} + \cdots) - (2^{\ell_j - 1} + 2^{\ell_j - 2})| =$$

$$= |(r_j - 1)2^{\ell_j - 2} + \cdots| \leq 2^{\ell_j - 2} - 1,$$

and since $\ell_j < \ell$ i.e. $\ell_j \leq \ell - 1$ we get $|u_j - E_j| \leq 2^{\ell-3} - 1$.

To summarize, our selection of $\mathbf{E}$ is based on an oracle that provides the binary length of the coefficients of $u(x) = -kh(x) * r(x)$ in $\mathcal{R}$.

Let $\mathbf{E} = (E_0, E_1, ..., E_{N-1})$. We apply the following algorithm.

`Input:` The ciphertext $\mathbf{c}$, a positive integer k, and the previous $\mathbf{E}$.
`Output:` The message $\mathbf{m}$ or fail.
1 : Set b_i the coefficients of $kc(x)$. Further, let the target vector $\mathbf{t} = (0, ..., 0, b_0 + E_0, ..., b_{N-1} + E_{N-1})$.
2 : Call Babai algorithm to the pair $(\mathcal{L}_k, \mathbf{t})$ and let $\mathbf{w}$ be its output.
3 : Return the first $N-$coordinates of $\mathbf{w}$.
In step 3 we get the possible message $\mathbf{m}$, in this case the first $N-$coordinates of $\mathbf{w}$ is $-\mathbf{m}$.

We continue with the choice of integer k. The value of k defines the lattice $\mathcal{L}_k$. Babai algorithm is used to approximate the distance $d_1 = d(\mathcal{L}_k, \mathbf{t})$, where $\mathcal{L}_k$ is our lattice with the parameter k. Typically, Babai's algorithm is employed to find an approximation of the closest vector of a lattice given a target vector $\mathbf{t}$. Here, the goal is to choose k such that d_1 is close to an unknown distance $d_2 = d(\mathbf{u}, \mathbf{E})$. The vector $\mathbf{u}$ is unknown, but through experimentation, an estimation of d_2 is obtained. If $d_1 \approx d_2 = \|\mathbf{u} - \mathbf{E}\|$ then the output of Babai, say the vector $\mathbf{w} = (\mathbf{w}_1, \mathbf{w}_2)$ will be such that $\mathbf{w}_1 = -\mathbf{m}$.

We shall try to explain the previous i.e. $\mathbf{w}_1 = -\mathbf{m}$. In general, $||\mathbf{w} - \mathbf{t}|| \leq ||\mathbf{V} - \mathbf{E}'||$ (we proved this by analyzing (6)). In an extreme case we can have

$$\mathbf{V} - \mathbf{E}' = \mathbf{w} - \mathbf{t}, \ \mathbf{w} \leftarrow CVP(\mathcal{L}_k, \mathbf{t}),$$

then by equating the first $N-$coordinates of two parts we get, $\mathbf{w}_1 = -\mathbf{m}$ and since $d_2 \approx ||\mathbf{V} - \mathbf{E}'|| = ||\mathbf{w} - \mathbf{t}|| = d_1$, we build our heuristic : *choose k such that $d_1 \approx d_2$*. It is proved that this heuristic works very well in practice, since it provides the message.

In the provided figure (Fig.1), the parameter q is set to 2048, and k ranges from 1 to 1000. Certain parameters for the NTRU cryptosystem, as well as the message $m(x)$ and the nonce $r(x)$, are fixed. For each value of k, the target vector $\mathbf{t}$ is computed based on the previous selection of $\mathbf{E}$, and the unknown vector $\mathbf{u}$ is also computed. In the $y-$axis we compute the difference $|d_1 - d_2|$, where d_1 is computed using Babai's algorithm. We remark that for k say ≈ 550 the previous difference is minimized. So, using such a k we expect the output of Babai to reveal the message. Similar for $k = 4096$ we pick $k = 1080$.

Now having a way to select both $\mathbf{E}$ and k we can execute our attack. We applied it for the three variants of NTRU-HPS (the code is in the github repository), namely ntruhps2048509, ntruhps2048677 and ntruhps4096821. These are the suggested parameters for the NTRU-HPS when submitted to NIST competition. For all the experiments we revealed the unknown message. The attack time was negligible, approximately 1 second.

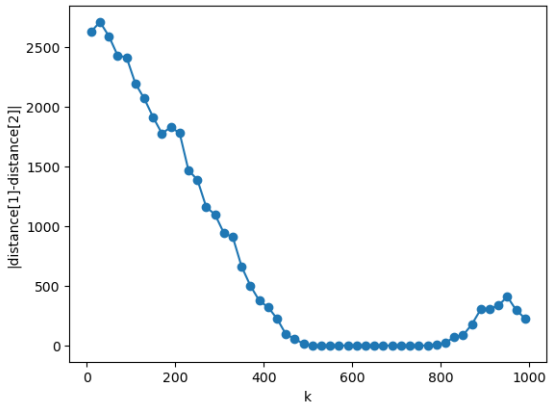

Fig. 1. In this graph we set $q = 2048$. k takes values in the horizontal axis and on the $y-$axis is the $|\text{distance}(\mathbf{u}, \mathbf{E}) - \text{distance}(\mathcal{L}_k, \mathbf{t})|$. We remark that Babai's algorithm provides outputs with distances close to $\text{distance}(\mathbf{u}, \mathbf{E})$ for $k \in [520, 790]$. We finally select k to be 550.

Remark 2. In [2, Example 7] for ntruhps2048509 we get $|u_i - E_i| \leq 36$ and in our attack we get $2^{\ell-3} - 1 = 256$ (here $\ell = \text{bits}(q) - 1 = 11$), which is a significant improvement.

6 Conclusion

Eve having the public keys, a ciphertext and using a simple oracle that outputs the first and in some cases the second Most Significant Bit of the coefficients of an unknown polynomial, reveals the message. We have successfully applied

our attack to NTRU-HPS using state-of-the-art parameters. The attack was easily implemented in sagemath [25] and the results showed great efficiency. We expect that the same attack will be successful for the other two variants of NTRU, namely NTRU-HRSS and NTRU-Prime. Finally, an oracle that satisfies assumptions A and B may be constructed using a side-channel attack, see 2.1.

References

1. K. Abdel and Y. Amr, A Scan-Based Side Channel Attack on the NTRUEncrypt Cryptosystem, 7th International Conference on Availability, Reliability and Security, 2012.
2. M. Adamoudis and K. A. Draziotis, Message recovery attack on NTRU using a lattice independent from the public key, 2023, To appear in Advances in Mathematics of Communications (Amer. Inst. of Math. Sciences), doi:http://dx.doi.org/10.3934/amc.2023040, arXiv:https://arxiv.org/abs/2203.09620
3. M. Albrecht, S. Bai, and L. Ducas, A subfield lattice attack on overstretched NTRU assumptions. CRYPTO 2016. LNCS **9814**, Springer 2016.
4. L. Babai. On Lovász lattice reduction and the nearest lattice point problem. Combinatorica, 6(1):1–13, 1986.
5. G. Bourgeois and J. C. Faugère, Algebraic attack on NTRU using Witt vectors and Gröbner bases, Journal of Mathematical Cryptology **3(3)** p. 205–214, 2009.
6. D. Coppersmith and A. Shamir, Lattice Attacks on NTRU. In Proc. Eurocrypt 1997, LNCS **1223**, Springer.
7. J. H. Cheon, J. Jeong, and C. Lee, An algorithm for NTRU problems and cryptanalysis of the GGH multilinear map without an encoding of zero. Cryptology ePrint Archive, Report 2016/139, 2016.
8. C. Gentry, Key recovery and message attacks on NTRU-composite, EUROCRYPT 2001, LNCS **2045**, Springer 2001.
9. N. Gama and P. Q. Nguyen, New Chosen-Ciphertext Attacks on NTRU. Public Key Cryptography – PKC 2007, LNCS **4450**, Springer 2007.
10. J. Hoffstein, J. Pipher, and J. H. Silverman, NTRU: A ring-based public key cryptosystem, in Proceedings of ANTS '98 (ed. J. Buhler), LNCS **1423**, p. 267–288, 1998.
11. N. Howgrave-Graham. A Hybrid Lattice-Reduction and Meet-in-the-Middle Attack Against NTRU. CRYPTO 2007, LNCS **4622**, Springer 2007.
12. N. A. Howgrave-Graham and N. P. Smart, Lattice Attacks on Digital Signature Schemes, *Des. Codes Cryptogr.* 23, p. 283–290, 2001.
13. P. Kirchner and P. A. Fouque, Revisiting Lattice Attacks on Overstretched NTRU Parameters. Eurocrypt 2017, LNCS **10210**, Springer 2017, doi: 10.1007/978-3-319-66787-4_12
14. N. Howgrave-Graham, J. H. Silverman, and W. Whyte, Meet-in-the-middle Attack on an NTRU private key, Technical report, NTRU Cryptosystems, July 2006. Report 04, available at http://www.ntru.com.
15. E. Kirshanova, A. May and J. Nowakowsk, New NTRU Records with. Improved Lattice Bases. eprint : 2023/582
16. P. C. Kocher, Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems Advances in Cryptology — CRYPTO '96, Berlin, Heidelberg, 2001.

17. P. Kocher, J. Jaffe and B. Jun Differential Power Analysis, Advances in Cryptology — CRYPTO' 99, Lecture Notes in Computer Science, vol 1666. Springer, Berlin, Heidelberg, 1999
18. S. Mangard , E. Oswald and T. Popp, Power Analysis Attacks, Springer New York, NY, 2007.
19. A. May, Cryptanalysis of NTRU (preprint), 1999., http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.41.3484.
20. D. Micciancio and P. Voulgaris. A deterministic single exponential time algorithm for most lattice problems based on Voronoi cell computations. *In Proc. of STOC, ACM*, p. 351-358, 2010.
21. NIST, 3rd round candidate announcement, https://csrc.nist.gov/news/2020/pqc-third-round-candidate-announcement (Accessed 1 January 2022).
22. P. Q. Nguyen, Boosting the Hybrid Attack on NTRU: Torus LSH, Permuted HNF and Boxed Sphere, Third PQC Standardization Conference, 2021.
23. K.G. Paterson and R. Villanueva-Polanco Cold Boot Attacks on NTRU, Progress in Cryptology, INDOCRYPT 2017, Springer, 2017.
24. Eirini Poimenidou, Marios Adamoudis, Konstantinos A. Draziotis, and Kostas Tsichlas, Message Recovery Attack in NTRU through VFK Lattices, preprint, doi:https://doi.org/10.48550/arXiv.2311.17022
25. Sage Mathematics Software, The Sage Development Team, http://www.sagemath.org.
26. C. P. Schnorr. A hierarchy of polynomial time lattice basis reduction algorithms. Theoretical Computer Science, Volume 53, Issues 2-3, p. 201–224, 1987.
27. Scott Edwards, GoldBug Crypto Messenger (2018). https://compendio.github.io/goldbug-manual/
28. Peter W. Shor, Algorithms for quantum computation: Discrete logarithms and factoring. In 35th Annual Symposium on Foundations of Computer Science, Santa Fe, New Mexico, USA, 20-22 November 1994, p. 124–134. IEEE Computer Society, 1994.
29. J. H. Silverman, Dimension-Reduced Lattices, Zero-Forced Lattices, and the NTRU Public Key Cryptosystem. Technical Report 13, Version 1, NTRU Cryptosystems, 1999.
30. H. Silverman, N. P. Smart, and F. Vercauteren, An algebraic approach to NTRU ($q = 2n$) via Witt vectors and overdetermined systems of non linear equations. Security in Communication Networks – SCN 2004, LNCS **3352**, p. 278–298. Springer, 2005.
31. https://www.openssh.com/txt/release-9.0
32. https://www.wolfssl.com/products/wolfssl/

Appendix A

Proposition 2. *Let k, N and q be positive integers with $q \geq (k+1)\sqrt{k^2+1}$. We set*

$$M_k = \left[\begin{array}{c|c} I_N & -kI_N \\ \hline \boldsymbol{0}_N & qI_N \end{array}\right].$$

Let $\mathcal{L}_k$ be the lattice generated by the rows of M_k. Then it is $\lambda_1(\mathcal{L}) = \sqrt{k^2+1}$.

Proof. First we will prove that for all non-zero $\mathbf{v} \in \mathcal{L}_k$ we have $\|\mathbf{v}\| \geq \sqrt{k^2+1}$. Suppose that there is a vector $\mathbf{v} \in \mathcal{L}_k \setminus \{\mathbf{0}\}$ such that

$$\|\mathbf{v}\| < \sqrt{k^2+1}. \tag{7}$$

Let $\mathbf{b}_1, \ldots, \mathbf{b}_{2N}$ be the rows of the matrix M_k. Since $\mathbf{v} \in \mathcal{L}_k$, there are integers $l_1, \ldots, l_{2N}$ such that,

$$\mathbf{v} = l_1\mathbf{b}_1 + \cdots + l_{2N}\mathbf{b}_{2N} =$$
$$(l_1, \ldots, l_N, -l_1k + ql_{N+1}, \ldots, -l_Nk + ql_{2N})$$

From the inequality (7) we get

$$\begin{cases} |l_1|, |l_2|, \ldots, |l_N| < \sqrt{k^2+1} \\ |-l_1k + ql_{N+1}| < \sqrt{k^2+1} \\ \ldots \\ |-l_Nk + ql_{2N}| < \sqrt{k^2+1} \end{cases} \tag{8}$$

So we can easily see that for $i = 1, \ldots, N$ we get

$$|l_ik| < \sqrt{k^2+1}k. \tag{9}$$

Case 1: not all the integers $l_{N+1}, l_{N+2}, \ldots, l_{2N}$ are zero.
Without loss of generality, say l_{N+j} is not zero for some $j \in \{1, \ldots, N\}$. Then from (9) and (8), we get

$$\|\mathbf{v}\| \geq |-l_jk + ql_{N+j}| \geq |l_{N+j}|q - |l_jk| > q - \sqrt{k^2+1}k \geq \sqrt{k^2+1},$$

which contradicts to inequality (7).
Case 2: Let $l_{N+1} = l_{N+2} = \cdots = l_{2N} = 0$.
In this case

$$\mathbf{v} = (l_1, \ldots, l_N, -l_1k, \ldots, -l_Nk).$$

Then,

$$\|\mathbf{v}\| = \sqrt{l_1^2(1+k^2) + l_2^2(1+k^2) + \cdots + l_N^2(1+k^2)} > \sqrt{k^2+1},$$

which contradicts our hypothesis (7).