

MESSAGE RECOVERY ATTACK IN NTRU VIA KNAPSACK

EIRINI POIMENIDOU AND K. A. DRAZIOTIS

ABSTRACT. In the present paper, we introduce a message-recovery attack based on the Modular Knapsack Problem, applicable to all variants of the NTRU-HPS cryptosystem. Assuming that a fraction ϵ of the coefficients of the message $\mathbf{m} \in \{-1, 0, 1\}^N$ and of the nonce vector $\mathbf{r} \in \{-1, 0, 1\}^N$ are known in advance at random positions, we reduce message decryption to finding a short vector in a lattice that encodes an instance of a modular knapsack system. This allows us to address a key question: how much information about $\mathbf{m}$, or about the pair $(\mathbf{m}, \mathbf{r})$, is required before recovery becomes feasible? A FLATTER reduction successfully recovers the message, in practice when $\epsilon \approx 0.45$. Our implementation finds $\mathbf{m}$ within a few minutes on a commodity desktop.

1. INTRODUCTION

In 1996, Hoffstein, Pipher, and Silverman developed the NTRU cryptosystem, aiming to create robust encryption and signature systems, as detailed in [18]. Its security is based on the difficulty of solving a system of linear equations over polynomial rings, a problem that is expected to remain hard even with quantum computers. NTRU cryptosystem has withstood over 25 years of cryptanalysis, and variants of it NTRU have been shown to be closely related to the Ring Learning With Errors (R-LWE) problem, whose hardness is supported by worst-case reductions on ideal lattices. NTRU is known for its exceptional performance and moderate key-size, making it a popular choice for embedded cryptography. It has been standardized by IEEE, X9.98, and PQCrypto, and was a finalist in the NIST post-quantum cryptography standardization effort.

In the present work, we outline a message recovery attack on NTRU based on the Shortest Vector Problem (SVP). Our method assumes partial knowledge of the message $\mathbf{m} \in \{-1, 0, 1\}^N$ and/or of the nonce vector $\mathbf{r} \in \{-1, 0, 1\}^N$. Such leakage assumptions are standard in the literature: for example, several works on DSA [2, 3, 20] recover secret keys by exploiting partial information about ephemeral keys. In [30], the authors use "hints" from an oracle to recover the secret key in Kyber. Also, Coppersmith type attacks assume some knowledge of the one prime in order to compute the remaining part.

1.1. Previous Work. The NTRU cryptosystem was first subjected to a lattice-based attack in 1997 by Coppersmith and Shamir [9]. Later, Gama and Nguyen [14] exploited decryption failures to recover the secret key, under the assumption of access to a decryption oracle. Subsequently, in 2001, Gentry [16] proposed an attack that is particularly effective when the parameter N is composite.

2020 *Mathematics Subject Classification.* 94A60.

Key words and phrases. Public Key Cryptography; NTRU Cryptosystem; Lattices; Lattice Reduction; Modular Knapsack Problem; Shortest Vector Problem;

A number of other approaches have been developed to attack NTRU. One notable line of work reformulates the NTRU problem as a system of multivariate quadratic equations over the binary field, utilizing Witt vectors [7, 40]. Odlyzko [21] proposed a meet-in-the-middle strategy that partitions the search space into two halves, reducing time complexity at the cost of substantial memory usage. Building on this idea, Howgrave-Graham introduced a hybrid attack [19] that combines lattice reduction with a meet-in-the-middle approach. This hybrid technique has since become a standard method for assessing the security of lattice-based encryption schemes.

For NTRU variants operating under larger moduli than those used in the NTRU-Encrypt standard, Albrecht, Bai, and Ducas [5], as well as Cheon, Jeong, and Lee [11], independently extended and refined these hybrid and meet-in-the-middle techniques to remain effective in more demanding parameter regimes. Nguyen [33] later enhanced these methods by clarifying their structure and further improving their efficiency. While the subfield attack proposed in the previous work surpasses several earlier strategies, it still falls short of the most advanced hybrid attacks in terms of performance.

More recently, in 2023, May and Nowakowski [26] introduced a powerful new attack on the latest NTRU encryption scheme. Their approach employs a carefully designed lattice, leveraging the BKZ algorithm in conjunction with the sieving methods from the G6K library. Transitioning away from the traditional Coppersmith-Shamir lattice, they construct a lattice based on the cyclotomic ring, achieving significant performance gains.

Finally, in 2025, two message recovery attacks [4, 35] based on Babai nearest plane algorithm are presented. The authors of [34] also implement a message recovery attack, after reducing the NTRU-lattice to a Voronoi First Kind (VFK) lattice and then use a polynomial exact CVP-algorithm to recover the message.

1.2. Our contribution. In this paper we aim to recover the unknown message. We consider an adversary who observes ciphertexts of the NTRU-HPS encryption scheme and who is assumed to know k coefficients of the ternary plaintext polynomial $m(x) \in \{-1, 0, 1\}^N$ or k_1 coefficients of $m(x)$ and k_2 coefficients of the unknown nonce $r(x) \in \{-1, 0, 1\}^N$. More precisely, knowledge of approximately 45% of the coefficients of the pair of unknown polynomials $(m(x), r(x))$, suffices to reconstruct $r(x)$ and, consequently, recover the plaintext $m(x)$. Such partial information may be leaked via a side-channel, arise from protocol redundancy, or be exposed by format markers.

Our method differs from previous approaches [4, 34, 35], as it relies on the *Shortest Vector Problem (SVP)* rather than the *Closest Vector Problem (CVP)*. The SVP is better understood, both theoretically and algorithmically, providing a more solid foundation for our analysis. Our assumptions in the present work are clearer, as it involves only the message polynomial $m(x)$ and the nonce $r(x)$. Moreover, the underlying problem we attack is different.

In our attack we solve a modular knapsack with coefficients in $\{-1, 0, 1\}$ (instead of $\{0, 1\}$). In more details our method reduces to the following problem: Let a matrix $A \in \mathbb{Z}_q^{n \times m}$ and a vector $B \in \mathbb{Z}_q^{n \times 1}$, we seek a vector X with entries in $\{-1, 0, 1\}$, such that $AX = B$ in $\mathbb{Z}_q$. We call this *modular knapsack problem*, also

known as *Inhomogeneous Short Integer Solution problem* ($ISIS_q$ [13]). Furthermore, the present paper contributes to the study of the previous problem both in theory and in practice.

Finally, our attack can also be applied to the NTRU-HRSS scheme, where only its parameters would need to change. For NTRU-Prime our attack could also be effective, however more research is required since it employs a non cyclic linear system which is different from the system in our case.

1.3. Plausibility of Partial-Plaintext Knowledge. In a Key Encapsulation Mechanism (KEM) the message $\mathbf{m}$ is ideally sampled uniformly at random, however if an implementation instead derives $\mathbf{m}$ (or the Key Derivation Function (KDF) preimage) from a structured seed, large portions of its content may be predictable. Additionally, if a weak pseudorandom number generator (PRNG) is used, then an adversary who collects sufficiently many $(\mathbf{m}_i, \mathbf{c}_i)$ pairs can detect statistical biases in certain coefficients of m_i and exploit them to recover partial information.

In [22] the authors present a side-channel attack that exploits leakage from *schoolbook* (product-scanning) polynomial multiplication when one operand is small with coefficients in $\{-1, 0, 1\}$.

In [23] they propose a power-based side-channel attack targeting the random generation of polynomials in NTRU. By combining a chosen-plaintext message attack with the collection of numerous $(\mathbf{m}, \mathbf{c})$ pairs, one can detect statistical biases in the distribution of the nonce $r(x)$.

1.4. Roadmap. In Section 2 we provide information on lattices, on the NTRU cryptosystem, we introduce modular knapsack problem and FLATTER reduction algorithm. Next, in Section 3 we provide some preliminaries about our attack, such as the construction of a system. In Section 4 we describe our attack, the experiments we conducted and the results we yielded. Finally, in Section 5 we provide a conclusion. Our work's corresponding implementation can be found on Github¹.

2. BACKGROUND

2.1. Lattices. In this section we recall some well-known facts about lattices. Let $\mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_n$ 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 $\mathcal{B} = \{\mathbf{b}_1, \dots, \mathbf{b}_n\}$ 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*. Let M be the $n \times m$ matrix, having as rows the vectors $\mathbf{b}_1, \dots, \mathbf{b}_n$. 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 $\text{vol}(\mathcal{L})$ 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 $\mathbf{t} \in \text{span}(\mathbf{b}_1, \dots, \mathbf{b}_n)$, then by $\text{dist}(\mathcal{L}, \mathbf{t})$, we denote $\min\{\|\mathbf{v} - \mathbf{t}\| : \mathbf{v} \in \mathcal{L}\}$.

¹<https://github.com/poimenidou/knapsack-message-recovery-attack>

There are two main fundamental problems on lattices the Shortest Vector Problem (SVP) and the Closest Vector Problem (CVP).

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

The Closest Vector Problem (CVP): Given a lattice $\mathcal{L}$ and a vector $\mathbf{t} \in \text{span}(\mathbf{b}_1, \dots, \mathbf{b}_n)$ that is not in $\mathcal{L}$, find a vector $\mathbf{b} \in \mathcal{L}$ that minimizes 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 the function $f(n)$ gives a different approximation of the Shortest Vector Problem.

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

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

2.2. Lattice Basis Reduction. The well known LLL algorithm [29], 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. We provide the definition of LLL reduced basis of a lattice $\mathcal{L}$.

Definition 2.1. A basis $\mathcal{B} = \{\mathbf{b}_1, \dots, \mathbf{b}_n\}$ 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 2.2. Let $\mathcal{L}$ be a lattice of rank n . For every LLL-reduced basis $\mathcal{B} = \{\mathbf{b}_1, \dots, \mathbf{b}_n\}$ of a lattice $\mathcal{L}$ we get,

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

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

For details on the algorithm you can refer to [29, Proposition 1.11]. Finally, we need the following Lemma.

Lemma 2.3. Let $\mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_n$ be an LLL-reduced basis of the lattice $\mathcal{L} \subseteq \mathbb{R}^m$, and let $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_t\} \subseteq \mathcal{L}$ be linearly independent vectors in $\mathbb{R}^m$. Then, for all $1 \leq j \leq t$, we have:

$$\|\mathbf{b}_j\|^2 \leq 2^{n-1} \max\{\|\mathbf{x}_1\|^2, \|\mathbf{x}_2\|^2, \dots, \|\mathbf{x}_t\|^2\}.$$

For a proof see [29, Proposition 1.12].

2.2.1. FLATTER Reduction. FLATTER is a fast lattice reduction algorithm for integer lattice bases created by Keegan Ryan and Nadia Heninger [36] in 2023. It enhances the classical LLL-style reduction through a recursive QR decomposition combined with precision compression at each recursion level. The algorithm provides guarantees similar to traditional LLL but with significantly better performance in practice.

The authors define a new notion of reduced-ness, called α -lattice-reduced, based on a metric they call the drop of a lattice basis. They define the drop of B as the total amount of downward steps in the lattice profile ℓ_i with $\ell_i = \log \|\mathbf{b}_i^*\|$, where $(\mathbf{b}_i^*)$ is the i -th Gram-Schmidt vector) for $i \in \{1, \dots, n\}$:

$$\text{drop}(B) = \text{vol} \left(\bigcup_{1 \leq i \leq n-1, \ell_{i+1} < \ell_i} [\ell_{i+1}, \ell_i] \right) = \sum_{i: \ell_{i+1} < \ell_i} (\ell_i - \ell_{i+1}).$$

A basis B of rank n is α -lattice-reduced if it is *size-reduced*, meaning that the upper-triangular Gram-Schmidt matrix has bounded coefficients, and if the drop of B is smaller or equal to αn , i.e. $\text{drop}(B) \leq \alpha n$. According to Theorem 2 of [36], if B is α -lattice-reduced, then it satisfies analogous bounds to LLL:

$$\begin{aligned} \|\mathbf{b}_1\| &\leq 2^{\alpha n} (\det B)^{1/n} \\ \|\mathbf{b}_n^*\| &\geq 2^{-\alpha n} (\det B)^{1/n} \\ \|\mathbf{b}_i\| &\leq 2^{\alpha n + O(n)} \lambda_i(B), \quad \text{for all } i \in \{1, \dots, n\}, \\ \prod_{i=1}^n \|\mathbf{b}_i\| &\leq 2^{\alpha n^2 + O(n^2)} \det B, \end{aligned}$$

where $B = [\mathbf{b}_1, \dots, \mathbf{b}_n]$ and $\lambda_i(B)$ is the i -th successive minimum of the lattice generated by B . Hence, α plays the same mathematical role as δ —constant in the Lovász inequality that governs how orthogonal and how computationally expensive the reduced basis will be. Formally, a Flatter-reduced basis is not LLL-reduced because it replaces the Lovász condition with a new global drop condition, but the resulting basis satisfies the same geometric and approximation guarantees as an LLL-reduced basis, effectively making it an equivalent form of lattice reduction.

The FLATTER algorithm has an asymptotic heuristic running time of

$$O(n^\omega (C + n)^{1+\varepsilon}),$$

where $\omega \in (2, 3]$ is the matrix multiplication exponent, $C = \log(\|B\| \|B^{-1}\|)$ (where $\|\cdot\|$ denotes the spectral norm) bounds the condition number of the input basis and $\varepsilon > 0$ is an arbitrarily small constant accounting for the subpolynomial overhead of fast arithmetic operations on $O(C + n)$ -bit numbers. This cost arises from recursive compression and sublattice-reduction steps, each dominated by size reduction and QR factorization at precision $O(C + n)$. The resulting complexity matches that of prior heuristic recursive methods [25, 28, 31] but is obtained under significantly weaker assumptions, allowing FLATTER to achieve LLL-equivalent reduction quality with practical, near-matrix-multiplication speed.

The implementation of this new algorithm was benchmarked extensively by the authors, against fpLLL² (the current gold-standard implementation) and previous recursive methods. Across a range of lattices, FLATTER outperformed existing tools and could successfully reduce even lattices of dimension 8192 in 6.4 core years. FLATTER’s implementation is available on Github³. All the experiments in this paper failed to yield a correct result with the usage of fpLLL’s reduction algorithms (we used a combination of LLL and BKZ with different blocksizes) while FLATTER’s algorithm successfully reduced our input bases.

²<https://github.com/fplll/fplll>

³<https://github.com/keeganryan/flatter>

2.3. NTRU-HPS. In this section, we discuss about the NTRU-HPS. Let the polynomial ring $\mathcal{R} = \mathbb{Z}[x]/\langle x^N - 1 \rangle$ and we write $\star$ for the multiplication in the ring. If

$$a(x) = a_{N-1}x^{N-1} + \cdots + a_0 \quad \text{and} \quad b(x) = b_{N-1}x^{N-1} + \cdots + b_0,$$

then $c(x) = a(x) \star b(x)$, is given by

$$c_k = \sum_{i+j \equiv k \pmod{N}} a_i b_j, \quad 0 \leq k \leq N-1.$$

Alice selects public parameters (N, q, d) , with N being prime number and $\gcd(q, N) = \gcd(3, q) = 1$. Usually N and q are large, and q is a power of 2. With $\mathcal{T}_a$ we denote the set of ternary polynomials⁴ of $\mathcal{R}$ with degree at most a and $\mathcal{T}_a(d_1, d_2) \subset \mathcal{T}_a$ consists from elements of $\mathcal{T}_a$ with d_1 coefficients equal to 1 and d_2 equal to -1 .

For her private key, Alice randomly selects $(f(x), g(x))$ such that $f(x) \in \mathcal{M}_f = \mathcal{T}_{N-2}$ and $g(x) \in \mathcal{M}_g = \mathcal{T}_{N-2}(\frac{q}{16} - 1, \frac{q}{16} - 1)$. It is important that $f(x)$ is invertible in both $\mathcal{R}/q$ and $\mathcal{R}/3$. The inverses in $\mathcal{R}/3$ and $\mathcal{R}/q$ can be efficiently computed using the Euclidean algorithm and Hensel's Lemma, see [18, Proposition 6.45]. Let $F_q(x)$ and $F_3(x)$ represent the inverses of $f(x)$ in $\mathcal{R}/q$ and $\mathcal{R}/3$, respectively.

Alice next computes

$$(2.1) \quad h(x) = F_q(x) \star g(x) \pmod{q}.$$

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

Bob's plaintext is a polynomial $m(x) \in \mathcal{M}_m$, where $\mathcal{M}_m = \mathcal{T}_{N-2}(\frac{q}{16} - 1, \frac{q}{16} - 1)$. Then he chooses a random ephemeral key $r(x) \in \mathcal{M}_r = \mathcal{T}_{N-2}$ and computes the ciphertext,

$$(2.2) \quad c(x) = 3r(x) \star h(x) + m(x) \pmod{q}.$$

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

To decrypt, Alice computes

$$v(x) = f(x) \star c(x) \pmod{q}.$$

Then, she centerlifts $v(x)$ to an element of $\mathcal{R}$, say $v'(x)$, and she finally computes,

$$b(x) = F_3(x) \star v'(x) \pmod{3}.$$

Therefore, $b(x)$ is equal to the plaintext $m(x)$ (this is true when a simple inequality between d, q , and N is satisfied). For the exact values of N, q we will use the ones proposed by NIST [10].

2.4. Knapsack Problem. Here we discuss the knapsack problem in cryptography.

Definition 2.4 (Knapsack Problem). *Given $\mathbf{a} = (a_1, \dots, a_m)$, $a_i \in \mathbb{N}$ and $s \in \mathbb{Z}$, find $\mathbf{x} = (x_1, \dots, x_m) \in \{0, 1\}^m$ if it exists, such that*

$$\sum_{i=1}^m a_i x_i = s.$$

This problem is NP-complete [15]. A variation of the knapsack problem is the modular knapsack problem, which is the following:

⁴A ternary polynomial is one that has as coefficients only the integers $-1, 0, 1$.

Definition 2.5 (Modular Knapsack Problem). *Given a modulus q , vector $\mathbf{a} = (a_1, \dots, a_m)$, $a_i \in \mathbb{N}$ and integer $s \in \mathbb{Z}$, find $\mathbf{x} = (x_1, \dots, x_m) \in \{0, 1\}^m$ if it exists, such that*

$$\sum_{i=1}^m a_i x_i \equiv s \pmod{q}.$$

We shall provide more general definitions using an abstract Abelian (additive) group G .

Definition 2.6. *$((G, m, \mathcal{B})$ -knapsack). For an Abelian group G (written additively), integer m , and $\mathcal{B} \subset \mathbb{Z}$ small with $0 \in \mathcal{B}$, the $(G, m, \mathcal{B})$ -knapsack problem over G , asks: given elements $g_1, \dots, g_m \in G$ and a target $s \in G$, find coefficients $x_i \in \mathcal{B}$ such that*

$$s = \sum_{i=1}^m x_i g_i.$$

For instance, if $G = \mathbb{Z}_q^n$ for some positive integer q , and say $A \in G^m$ i.e. A is a $n \times m$ matrix with columns in G we are asking for solutions $\mathbf{x} \in \mathcal{B}^m$ such that $A\mathbf{x}^T = \mathbf{s}^T \pmod{q}$. It's worth noting that the knapsack problem over $\mathbb{Z}_q$ is equivalent to the Inhomogeneous Short Integer Solution problem *ISIS* _{q} [13].

There are three type of attacks in knapsack problems. Meet-in-the-middle, branch-and-bound and lattice-based. In meet-in-the-middle attacks [19] the set of variables is split into two halves, all partial sums for each half are computed and then a collision between the two halves is searched for, that reconstructs the target sum. In branch-and-bound attacks [12, 27] depth-first search strategies are used to explore the solution space of integer combinations systematically, pruning subtrees that cannot lead to valid solutions and at the same time bounding the partial sum, norm of the solution or residual target distance. Finally, with lattice-based attacks one can get a solution to the knapsack problem by reducing the problem to the CVP or the SVP in certain lattices or by using reduction algorithms to find short vectors in a lattice that corresponds to valid knapsack solutions. The latter is what the authors in [1] did in their lattice-based attack to the problem. We will call this method AHL and we describe it below.

2.4.1. AHL Knapsack Algorithm. Let A be an integer $N \times k$ matrix, with $k \leq N$ and $\mathbf{s}$ an integer column k -vector. Aardal, Hurkens and A. Lenstra [1] developed an algorithm to solve a system of Diophantine equations $AX = \mathbf{s}$, with lower and upper bounds $0 \leq X \leq \mathbf{u}$, where $\mathbf{u}$ is an integer N -vector. This is an NP-complete problem. In the absence of bound constraints it can be solved in polynomial time, for instance using Smith Normal Form (SNF)⁵.

The authors first create the matrix B :

$$B = \left[\begin{array}{c|c|c} I_N & \mathbf{0}_{N \times 1} & N_2 A_{N \times k} \\ \hline \mathbf{0}_{1 \times N} & N_1 & -N_2 \mathbf{s}_{1 \times k} \end{array} \right],$$

where N_1, N_2 are two positive integer numbers with $N_1 < N_2$. Then they use the LLL reduction algorithm to get the reduced form of the basis formed by the columns of B , which they denote $\hat{B}$. For suitably chosen $N_1, N_2 \in \mathbb{Z}$, the vector $\mathbf{x}$ is given by the first N entries of the $(N - k + 1)$ -th column of $\hat{B}$. We shall use the previous

⁵See Appendix A for details about SNF.

ideas to implemented a mod q variant and we shall use it for our attack. In this paper, we work with row representations; that is, when a reduction is applied to a matrix, it refers to a row reduction.

3. CONSTRUCTION OF OUR LATTICE

Starting from the NTRU-HPS encryption equation (2.2) we get:

$$(3.1) \quad 3^{-1}(c(x) - m(x)) \equiv h(x) \star r(x) \pmod{q}.$$

Our goal is to recover $r(x)$ (knowing $r(x)$ is equivalent to knowing $m(x)$). The idea is the following: using the information about m_i 's ($i = 1, 2, \dots, k$) to construct a linear system in $\mathbb{Z}_q$ with k -equations and N -unknowns. We already know that it has a small solution, namely the nonce $\mathbf{r}$. We shall apply lattice base methods, described earlier, to find this small solution.

The left-side of the previous equation (3.1) can be written as:

$$\sum_{j=0}^{N-1} 3^{-1}(c_j - m_j)x^j \in \mathcal{R}/q,$$

and the right-side of (3.1) as:

$$h(x) \star r(x) = \sum_{\ell=0}^{N-1} a_\ell x^\ell \text{ and } a_\ell = \sum_{i+j \equiv \ell \pmod{N}} h_j r_i.$$

Furthermore, we define the vectors $\mathbf{a}_\ell \in \mathbb{Z}^N$,

$$(3.2) \quad \mathbf{a}_\ell = (h_{(\ell \bmod N)}, h_{(\ell-1 \bmod N)}, \dots, h_{(\ell-(N-1) \bmod N)}), \quad (0 \leq \ell \leq N-1).$$

Now, if we set $\mathbf{r} = (r_0, \dots, r_{N-1})$ we define $a_\ell = \mathbf{a}_\ell \cdot \mathbf{r}$. From the encryption equation (2.2), if we know k coefficients of the message, say $\{m_{i_0}, m_{i_1}, \dots, m_{i_{k-1}}\}$, where

$$\mathbf{m} = (m_0, m_1, \dots, m_{k-1}, \dots, m_{N-1}),$$

then we uniquely determine the integers $\{a_{i_0}, a_{i_1}, \dots, a_{i_{k-1}}\}$, defined earlier.

Without loss of generality we assume that we know the first k coefficients of $m(x)$. We form the $k \times N$ matrix A with the vectors $\mathbf{a}_0, \mathbf{a}_1, \dots, \mathbf{a}_{k-1}$, as rows:

$$(3.3) \quad A = \begin{bmatrix} - & \mathbf{a}_0 & - \\ - & \mathbf{a}_1 & - \\ & \dots & \\ - & \mathbf{a}_{k-1} & - \end{bmatrix} = \begin{bmatrix} h_0 & h_{N-1} & \dots & h_1 \\ h_1 & h_0 & \dots & h_2 \\ \vdots & \vdots & \ddots & \vdots \\ h_{k-1} & h_{k-2} & \dots & h_{(k-N) \bmod N} \end{bmatrix}.$$

Thus, we have $A\mathbf{r}^T = \mathbf{T}_k$, where

$$(3.4) \quad \mathbf{T}_k = (a_0, \dots, a_{k-1})^T.$$

Our system (over $\mathbb{Z}_k$),

$$(3.5) \quad A\mathbf{X} = \mathbf{T}_k,$$

will have k equations and N unknowns, $\mathbf{X}$ is a $N \times 1$ column vector that represents the unknown polynomial $r(x)$ and $\mathbf{T}_k$ is the $k \times 1$ column vector with the known entries $a_0, \dots, a_{k-1}$.

We construct the matrix B_k of dimension $(N + k + 1) \times (N + k + 1)$,

$$(3.6) \quad B_k = \left[\begin{array}{c|c|c} I_N & \mathbf{0}_{N \times 1} & N_2 A^T \\ \hline \mathbf{0}_{1 \times N} & N_1 & -N_2 \mathbf{T}_k^T \\ \hline \mathbf{0}_{k \times N} & \mathbf{0}_{k \times 1} & N_2 q I_k \end{array} \right] \quad (A^T \text{ is } N \times k),$$

where A is defined in (3.3), N_1, N_2 are positive integers which we shall determine later, and $\mathcal{L} \subset \mathbb{Z}^{N+k+1}$ is the lattice generated by the rows of B_k . This lattice is q -ary of full rank of dimension $N + k + 1$ and volume $N_1(N_2q)^k$.

The basis vectors are:

$$\begin{aligned} \mathbf{b}_0 &= (\underbrace{1, 0, \dots, 0}_{N \text{ entries}}, 0, N_2 h_0, N_2 h_1, \dots, N_2 h_{k-1}) \\ \mathbf{b}_1 &= (0, 1, \dots, 0, 0, N_2 h_{N-1}, N_2 h_0, \dots, N_2 h_{k-2}) \\ &\vdots \\ \mathbf{b}_{N-1} &= (\underbrace{0, \dots, 0, 1}_{N \text{ entries}}, 0, N_2 h_1, N_2 h_2, \dots, N_2 h_{(N-k) \bmod N}) \\ \mathbf{b}_N &= (\underbrace{0, 0, \dots, 0}_{N \text{ entries}}, N_1, -N_2 a_0, -N_2 a_1, \dots, -N_2 a_{k-1}) \\ \mathbf{b}_{N+1} &= (\underbrace{0, 0, \dots, 0}_{N \text{ entries}}, 0, \underbrace{N_2 q, 0, \dots, 0}_{k \text{ entries}}) \\ &\vdots \\ \mathbf{b}_{N+k} &= (\underbrace{0, 0, \dots, 0}_{(N+k) \text{ entries}}, N_2 q). \end{aligned}$$

The lattice points are of the form, $(\lambda_0, \dots, \lambda_{N-1}, N_1 \lambda_N, N_2 \beta_0, \dots, N_2 \beta_{k-1})$, where λ_j, β_j are integers (with $q|\beta_j$). In more details, let $(\lambda_0, \lambda_1, \dots, \lambda_{N+k})$ integer vector, and set $\mathbf{\Lambda}_N = (\lambda_0, \lambda_1, \dots, \lambda_{N-1})$. Then the lattice points are of the form:

$$(3.7) \quad \sum_{j=0}^{N+k} \lambda_j \mathbf{b}_j = \left(\lambda_0, \dots, \lambda_{N-1}, N_1 \lambda_N, \right. \\ \left. N_2 \underbrace{(\mathbf{\Lambda}_N \cdot \mathbf{a}_0 - \lambda_N a_0)}_{\text{1st equation of (3.5)}} + N_2 q \lambda_{N+1}, \right. \\ \left. N_2 \underbrace{(\mathbf{\Lambda}_N \cdot \mathbf{a}_1 - \lambda_N a_1)}_{\text{2nd equation}} + N_2 q \lambda_{N+2}, \right. \\ \vdots \\ \left. N_2 \underbrace{(\mathbf{\Lambda}_N \cdot \mathbf{a}_{k-1} - \lambda_N a_{k-1})}_{k\text{th equation}} + N_2 q \lambda_{N+k} \right),$$

or equivalently,

$$\begin{aligned} \sum_{j=0}^{N+k} \lambda_j \mathbf{b}_j &= \left(\mathbf{\Lambda}_N, N_1 \lambda_N, \right. \\ &\quad N_2(\mathbf{\Lambda}_N \cdot \mathbf{a}_0 - \lambda_N a_0 + \lambda_{N+1} q), \\ &\quad N_2(\mathbf{\Lambda}_N \cdot \mathbf{a}_1 - \lambda_N a_1 + \lambda_{N+2} q), \\ &\quad \vdots \\ &\quad \left. N_2(\mathbf{\Lambda}_N \cdot \mathbf{a}_{k-1} - \lambda_N a_{k-1} + \lambda_{N+k} q) \right). \end{aligned}$$

If we can find a vector of the form:

$$(3.8) \quad (\mathbf{\Lambda}_N, N_1, 0, \dots, 0) \in \mathbb{Z}_q \quad (\text{i.e., } \lambda_N = 1)$$

for some suitable integers $(\lambda_0, \dots, \lambda_{N-1})$, then $\mathbf{\Lambda}_N$ constitutes a solution to the system

$$A\mathbf{X} = \mathbf{T}_k \quad \text{over } \mathbb{Z}_q.$$

Also the inverse is true. If there are $\lambda_0, \lambda_1, \dots, \lambda_{N-1}$ such that $A\mathbf{\Lambda}_N^T = \mathbf{T}_k$, then the vector $\mathbf{z} = (\mathbf{\Lambda}_N, N_1, qN_2\rho_{N+1}, \dots, qN_2\rho_{N+k})$ belongs to $\mathcal{L}$. Indeed, $\mathbf{z}$ is written as the integer linear combination: $\sum_{j=0}^{N-1} \lambda_j \mathbf{b}_j + N_1 \mathbf{b}_{N+1} + \sum_{j=N+1}^{N+k} \rho_j \mathbf{b}_j$. We proved the following.

Lemma 3.1. *The integer vector $\mathbf{x} = (x_0, \dots, x_{N-1})$ is a solution of $A\mathbf{X} = \mathbf{T}_k$ in $\mathbb{Z}_q$ if and only if $(\mathbf{x}, N_1, qN_2\rho_{N+1}, \dots, qN_2\rho_{N+k})$ is a point of the lattice $\mathcal{L}$.*

Remark 3.1. *Let $\mathbf{r} = (r_0, \dots, r_{N-1})$ be a nonce of NTRU. Then $\mathbf{Ar}^T = \mathbf{T}_k \pmod{q}$, so the vector*

$$\mathbf{v} = (\pm \mathbf{r}, \pm N_1, \mathbf{0}_k) \in \mathcal{L},$$

where the signs are taken as $(+, +), (-, -)$. Indeed, by definition of $\mathbf{r}$ we have $\mathbf{Ar}^T = \mathbf{T}_k \pmod{q}$, so there is some vector $\mathbf{R} = (\rho_1, \dots, \rho_k)$ such that $\mathbf{Ar}^T = \mathbf{T}_k - q\mathbf{R}$. Therefore, we choose $\lambda_N = \pm 1$, $\lambda_j = \lambda_N r_j$ ($0 \leq j \leq N-1$), and $\lambda_{N+j} = \lambda_N \rho_j$ ($1 \leq j \leq k$). Then, using (3.7) we get $\mathbf{v} = \sum_{j=0}^{N+k} \lambda_j \mathbf{b}_j$. Furthermore, the norm satisfies,

$$\|\mathbf{v}\|^2 = \|\mathbf{r}\|^2 + N_1^2 < N^2 + N_1^2.$$

Thus, $\mathbf{v}$ is a short vector of $\mathcal{L}$. In general, small solutions of the system $A\mathbf{X} = \mathbf{T}_k \pmod{q}$ provide small non-zero vectors of $\mathcal{L}$, and the inverse.

We shall prove that for suitable N_1, N_2 (and under some plausible conditions) the LLL-reduced matrix $\hat{B}_k$ is of the form

$$(3.9) \quad \hat{B}_k = \begin{pmatrix} \hat{b}_{0,0} & \cdots & \hat{b}_{0,N-1} & \varepsilon_0 & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & & \ddots & \\ \hat{b}_{N-1,0} & \cdots & \hat{b}_{N-1,N-1} & \varepsilon_{N-1} & 0 & \cdots & 0 \\ \hat{b}_{N,0} & \cdots & \hat{b}_{N,N-1} & \varepsilon_N & * & \cdots & * \\ \hat{b}_{N+1,0} & \cdots & \hat{b}_{N+1,N-1} & \varepsilon_{N+1} & & & \\ \vdots & & \vdots & \vdots & & \mathbf{C}_k & \\ \hat{b}_{N+k,0} & \cdots & \hat{b}_{N+k,N-1} & \varepsilon_{N+k} & & & \end{pmatrix},$$

where $\varepsilon_i \equiv 0 \pmod{N_1}$ for some indexes i and $\varepsilon_i = 0$ for the remaining indexes, and $\mathbf{C}_k$ is a $k \times k$ matrix. We remark that, at least one of the ε_i will be N_1 or $-N_1$. To see this we first remark that LLL-reduction makes the following two operations to the rows of B_k ,

$$\begin{aligned} \text{row}_j &\leftrightarrow \text{row}_i \\ \text{row}_j &\leftarrow \text{row}_j - \lambda \text{row}_i \quad (\lambda \in \mathbb{Z}), \text{ for } j > i. \end{aligned}$$

So, the $(N+1)$ -column, i.e. the vector $(0, \dots, 0, N_1, 0, \dots, 0)$, after the LLL-reduction shall contain only 0 and some multiples of N_1 . Since the gcd of the $(N+1)$ -column remains the same after the LLL reduction we get,

$$(3.10) \quad \gcd(\varepsilon_0, \dots, \varepsilon_{N+k}) = \gcd(0, \dots, 0, N_1, 0, \dots, 0) = N_1.$$

So, indeed $\varepsilon_i \equiv 0 \pmod{N_1}$.

Let a $k \times N$ ($k < N$) matrix A with $\text{rank}(A) = k$. For the following Theorem we need the Smith Normal Form of A , which is given by:

$$D = PAQ, \quad P \in GL_k(\mathbb{Z}), \quad Q \in GL_N(\mathbb{Z}), \quad D = \text{diag}(d_1, \dots, d_k, 0, \dots, 0) \in \mathbb{Z}^{k \times N}$$

with $d_i \mid d_{i+1}$ and the last $N - k$ diagonal entries of D are zero. Let,

$$Q = [\mathbf{q}_1 \mid \dots \mid \mathbf{q}_N].$$

Since, $\text{rank}(A) = k$, then for each $j \in \{k+1, \dots, N\}$ the columns $\mathbf{q}_j$ generate $\text{Ker}_{\mathbb{Z}}(A)$. In Appendix A, we provide the proof of the following Theorem.

Theorem 3.2. *If $V = \text{span}(\mathbf{q}_{k+1}, \dots, \mathbf{q}_N)$, $W = \text{span}(\mathbf{e}_1, \dots, \mathbf{e}_k)$ subspaces of $\mathbb{R}^N$ and $V \cap W = \{\mathbf{0}\}$, then there are N_1, N_2 such that, the LLL-reduced matrix of B_k is of the form $\hat{B}_k$. In fact we prove that $2^{N+k} N_1^2 < c(N, k) < N_2^2$ for some constant $c(N, k)$.*

Say $\hat{\mathbf{b}}_{i_0}$ a row of the LLL reduced matrix $\hat{B}_k$, that has εN_1 ($\varepsilon \in \{-1, 1\}$) in $N+1$ entry ($N+1$ entry is the element $\hat{b}_{i_0, N}$ since we started counting from 0). Then, the vector $\mathbf{x}_\varepsilon = (\varepsilon \hat{b}_{i_0, 0}, \varepsilon \hat{b}_{i_0, 1}, \dots, \varepsilon \hat{b}_{i_0, N-1})$ is a solution of the system $AX = \mathbf{T}_k$ in $\mathbb{Z}_q$ if $i_0 \leq N-1$. This is immediate from Lemma (3.1). We shall use the previous idea to the attack presented below.

4. THE ATTACK

The procedure for the attack consists of two main stages. In the first stage, we construct the matrix B_k (3.6) and apply the reduction routine FLATTER. In the second stage, we extract from the $(n+1)$ -th column of the reduced matrix (cf. (3.9)), the first N entries, which we denote by $(\varepsilon_0, \dots, \varepsilon_{N-1})$. For every ε_i we check whether $N_1 \mid \varepsilon_i$ and we define the quotient as $\text{quotient} = \varepsilon_i / N_1$. We also extract the first N entries from the row indexed by ε_i ($\text{row} = \hat{B}[\text{index}(\varepsilon_i)][N]$). If the quotient is an integer, then we form the possible solution as,

$$\mathbf{r}' = (\text{row}[0]/\text{quotient}, \dots, \text{row}[N-1]/\text{quotient}).$$

If the Euclidean norm $\|\mathbf{r}'\|$ is smaller than a prescribed threshold `app_value`⁶, we return $\mathbf{r}'$, otherwise the algorithm reports failure. Note that the algorithm does not always recover the nonce in every instance.

⁶The threshold is chosen from the expected norm of a random ternary vector. If $r_i \in \{-1, 0, 1\}$ are i.i.d., then $S = \sum_{i=0}^{N-1} r_i^2 \sim \text{Bin}(N, 2/3)$ so $E[S] = 2N/3$. For large N we have $E[\sqrt{S}] \approx \sqrt{E[S]}$, which yields the empirical thresholds `app_value` ≈ 19 for `ntruhs2048509` ($N = 509$), ≈ 21 for `ntruhs2048677` ($N = 677$) and ≈ 24 for `ntruhs4096821` ($N = 811$).

In practice, we have observed that the solution vector can be found in any of the $[0, N - 1]$ positions in the $(n + 1)$ -th column of the reduced matrix, contrary to what was stated in [1], where they get the solution vector specifically from the $N + 1$ row of the reduced matrix. In Algorithm 1 we present the pseudocode of our attack and in Table 1 the results of our experiments.

It is worth emphasizing that the reduction was carried out using FLATTER [36]. In contrast, the fpLLL implementation were unable to reproduce successful recoveries under identical parameters.

Algorithm 1 Message Recovery Attack

```

1: Input:  $N, N_1, N_2, \mathbf{c}, \{m_0, m_1, \dots, m_{k-1}\}, app\_value, \mathbf{T}_k, q$ 
2: Output: the message  $\mathbf{m}$  of the NTRU-HPS system or null
3:  $\mathbf{a} \leftarrow [3^{-1}(c_i - m_i) \bmod q \mid i \in [0, \dots, k - 1]]$ 
4:  $\mathbf{a}_0 \leftarrow [\mathbf{H}_{k,i} \mid i \in [0, \dots, k - 1]]$  ▷ see subsection 4
5:  $B \leftarrow create\_basis(N, N_1, N_2, \mathbf{a}, \mathbf{a}_0)$  ▷ see relation (3.6)
6:  $\hat{B} \leftarrow FLATTER(B)$ 
7:  $column \leftarrow \hat{B}^T[N]$  ▷ get the  $(N + 1)$ -th column of  $\hat{B}$ 
8: for  $i \in \{0, \dots, N - 1\}$  do
9:    $quotient \leftarrow column[i]/N_1$ 
10:  if  $quotient \neq 0$  then
11:     $row \leftarrow \hat{B}[i][0 \text{ to } N - 1]$  ▷ first  $N$  elements of the  $i$ -th row
12:    if  $gcd(row) == |quotient|$  then
13:       $\mathbf{r}' \leftarrow row/quotient$ 
14:      if  $\|\mathbf{r}'\| \leq app\_value$  and  $\mathbf{r}' \in \{-1, 0, 1\}^N$  and  $A\mathbf{r}' \equiv \mathbf{T}_k \pmod{q}$ 
        then
15:         $m'(x) \leftarrow (c(x) - 3h(x) \star \mathbf{r}'(x)) \bmod q$ 
16:        return centerlift( $\mathbf{m}'$ )
17: return null

```

(N, q)	N_1	$x : N_2 = \lceil q^x \rceil$	k	%	runtime	rate
(509, 2048)	9	8	425	83%	5m	100%
(677, 2048)	1	15	600	89%	12m	90%
(821, 4096)	7	21	750	91%	17m	50%

TABLE 1. Message recovery attack for $N = 509$, $N = 677$, and $N = 821$.

For all of our experiments, we choose N_1 to be much smaller than N_2 , guided also by Theorem 3.2. We ensure that the lattice vector $\mathbf{v} = (\mathbf{r}', \pm N_1, \mathbf{0}_k)$ remains short, thereby improving the chances of recovering $\mathbf{r}$ efficiently via lattice reduction. The choice of N_1 , N_2 is crucial to the success of our algorithm. In Table 1 we see the results of the message recovery attack where we present the different values of N_1 , N_2 that we used for each dimension N , as well as the number of known coefficients of $m(x)$, notated as k . This value of k is the optimal for each dimension N , i.e. for smaller values of k we did not get any solution. The percentage column is calculated with $\frac{k}{N} * 100$ and to calculate the success rate, we have run 10 experiments per row. The runtime column is the total (wall) time of the algorithm in an AMD

Ryzen 7 3700X (16 cores) with 16 GB of RAM, Ubuntu machine. The code for the message recovery attack can be found on Github⁷.

We presented an algorithm capable of finding the full message m when approximately 80% of its entries are known for $N = 509$. This percentage can seem large enough but we can seemingly improve this. In the next part, we will discuss an alternative message recovery attack where the percentage of known elements of m will be reduced by incorporating information of the auxiliary vector r .

4.1. Alternative attack. In the present attack, we make the assumption that k_1 coefficients of the message $m(x)$ and k_2 coefficients of the nonce $r(x)$ are known by the attacker, and we show that we can recover the full nonce $r(x)$, and thus the full message. In the previous attack we had $k_2 = 0$.

The set Z_0 holds the positions of the zero-elements, the set Z_1 has the positions of the one-elements and Z_{-1} has the positions of the minus-one-elements of the known part of $r(x)$. All these sets have k_2 elements in total.

We remove from the matrix A the columns indexed by Z_0 , Z_1 , and Z_{-1} . Instead of solving the same system as in the previous attack $AX = \mathbf{T}_{k_1}$ (A has dimension $k_1 \times (N - k_1)$) we will now solve $A_z X = \mathbf{T}_z$, where $\mathbf{T}_z = \mathbf{T}_{k_1} - \mathbf{S}$ where,

$$\mathbf{S} = \sum_{i:r_i=1} \text{col}_i(A) - \sum_{i:r_i=-1} \text{col}_i(A).$$

Matrix A_z is obtained from A by removing the columns indexed by Z_0 , Z_1 , and Z_{-1} . Now, the dimension of A_z will be $k_1 \times (N - k_2)$ and of B_z will be $(N + k_1 - k_2 + 1) \times (N + k_1 - k_2 + 1)$, where B_z is the matrix:

$$(4.1) \quad B_z = \left[\begin{array}{c|c|c} I_{N-k_2} & \mathbf{0}_{(N-k_2) \times 1} & N_2 A_z^T \\ \hline \mathbf{0}_{1 \times (N-k_2)} & N_1 & -N_2 \mathbf{T}_z^T \\ \hline \mathbf{0}_{k_1 \times (N-k_2)} & \mathbf{0}_{k_1 \times 1} & N_2 q I_{k_1} \end{array} \right] (A_z^T \text{ is } (N - k_2) \times k_1).$$

So, we first construct the matrix B_z and then, we use the FLATTER algorithm to get the reduced basis $\hat{B}_z$ and find the solution vector just like we did in the previous attack. After the reduction, we reconstruct $r(x)$ (by adding the known k_2 elements) and we check if it is valid i.e. its norm satisfies the upper bound given by the *app_value*.

⁷<https://github.com/poimenidou/knapsack-message-recovery-attack/blob/main/attack.ipynb>

Algorithm 2 Alternative Message Recovery Attack

```

1: Input:  $N, N_1, N_2, \mathbf{c}, \{m_0, m_1, \dots, m_{k_1-1}\}, \{r_0, r_1, \dots, r_{k_2-1}\}, \text{app\_value}, \mathbf{T}_{k_1}, q$ 
2: Output: the message  $\mathbf{m}$  of the NTRU-HPS system or null
3:  $\mathbf{a} \leftarrow [3^{-1}(c_i - m_i) \bmod q \mid i \in [0, \dots, k_1 - 1]]$ 
4:  $\mathbf{a}_0 \leftarrow [\mathbf{H}_{k_1, i} \mid i \in [0, \dots, k_1 - 1]]$  ▷ see subsection 4
5:  $B_z \leftarrow \text{create\_basis}(N, N_1, N_2, \mathbf{a}, \mathbf{a}_0)$  ▷ see relation (3.6)
6:  $B'_z \leftarrow \text{delete\_columns}(B_z, N, N_1, N_2, k_1, \{\mathbf{r}_i : i \in [0, \dots, k_2 - 1]\})$ 
7:  $N' \leftarrow N - k_2$ 
8:  $\hat{B}_z \leftarrow \text{FLATTER}(B'_z)$ 
9:  $\text{column} \leftarrow (\hat{B}_z^T)[N]$  ▷ the  $(N + 1)$ -th column of  $\hat{B}_z$ 
10: for  $i \in \{0, \dots, N - 1\}$  do
11:   if  $\text{column}[i] == N_1$  or  $\text{column}[i] == -N_1$  then
12:      $\text{row} \leftarrow \hat{B}_z[i][0 \text{ to } N' - 1]$  ▷ first  $N'$  elements of the  $i$ -th row
13:      $\mathbf{r}' \leftarrow \text{set\_known\_positions}(\text{row}, \{\mathbf{r}_i : i \in [0, \dots, k_2 - 1]\})$ 
14:     if  $\|\mathbf{r}'\| \leq \text{app\_value}$  and  $\mathbf{r}' \in \{-1, 0, 1\}^N$  and  $\mathbf{A}\mathbf{r}' \equiv \mathbf{T}_{k_1} \pmod{q}$  then
15:        $m'(x) \leftarrow (c(x) - 3h(x) \star r'(x)) \bmod q$ 
16:       return  $\text{centerlift}(\mathbf{m}')$ 
17: return null

```

For all the experiments we choose N_1 and N_2 as in the previous attack. In Table 2 we present the results of the alternative message recovery attack, with the different values of N_1 , N_2 that we used for each dimension N , as well as the number of known elements of $m(x)$, notated as k_1 and the the number of known elements of $r(x)$, notated as k_2 . The percentage column is calculated with $\frac{k_1+k_2}{2N} * 100$ and to calculate the success rate we have run 10 experiments per row. The runtime refers to the total (wall) time our algorithm takes in an AMD Ryzen 7 3700X (16 cores) with 16 GB of RAM, Ubuntu machine. The highlighted rows are the ones with $\min(k_1 + k_2)$ and the highest success rate of that specific dimension N . The code for the message recovery attack can be found on Github⁸.

(N, q)	N_1	$x : N_2 = \lceil q^x \rceil$	k_1	k_2	$k_1 + k_2$	%	runtime	rate
(509, 2048)	9	8	300	125	425	42%	3m	10%
(509, 2048)	9	8	250	185	435	43%	3m	0%
(509, 2048)	9	8	300	135	435	43%	3m	100%
(509, 2048)	9	8	230	215	445	44%	2m	50%
(509, 2048)	9	8	250	195	445	44%	3m	100%
(509, 2048)	9	8	350	100	450	44%	3m	100%
(509, 2048)	9	8	230	225	455	45%	2m	100%
(677, 2048)	1	15	500	100	600	44%	10m	10%
(677, 2048)	1	15	400	210	610	45%	5m	0%
(677, 2048)	1	15	500	110	610	45%	10m	90%
(677, 2048)	1	15	400	215	615	45%	5m	70%
(677, 2048)	1	15	400	220	620	46%	5m	10%
(677, 2048)	1	15	315	310	625	46%	2m	40%
(677, 2048)	1	15	315	315	630	47%	2m	60%

⁸https://github.com/poimenidou/knapsack-message-recovery-attack/blob/main/attack_-101.ipynb

(821, 4096)	7	21	700	90	790	48%	14m	90%
(821, 4096)	7	21	410	400	810	49%	2m	0%
(821, 4096)	7	21	500	310	810	49%	3m	0%
(821, 4096)	7	21	600	210	810	49%	5m	10%
(821, 4096)	7	21	410	410	820	50%	2m	100%
(821, 4096)	7	21	500	320	820	50%	3m	100%
(821, 4096)	7	21	600	220	820	50%	5m	100%

Table 2: Alternative attack for $N = 509$, $N = 677$, and $N = 821$

From Table 2 we observe that the most efficient attacks time-wise, require the value $k_1 - k_2$ to be as minimal as possible, while preserving $k_1 \geq k_2$. The most efficient attacks are the ones with $k_1 = k_2$, however they don't necessarily have the highest success probability when the objective is to minimize the sum $k_1 + k_2$. In the end, the best combination of k_1 , k_2 values depend on the information the attacker has, while always keeping in mind the previous observations.

This alternative attack cannot be compared directly to the previous attack since the initial assumptions differ. However, it stands as an improvement to the previous one since it cuts in half the total percentage of the known elements in the assumption about revealed entries (40% – 45% as opposed to 80% – 90%). It also offers greater flexibility since it permits asymmetric leakage between the plaintext m and the nonce r , meaning that the attacker may know different numbers of entries of m and r provided that their combined amount meets the required threshold for the given dimension N . The alternative attack also decreases by a large percentage the total wall time of the reduction time and therefore of the algorithm.

To mitigate our attack, implementers should ensure that the plaintext m is sampled uniformly at random rather than derived from structured or predictable protocol fields. Side-channel protections are essential, particularly for polynomial multiplication, which should be implemented in constant time.

5. CONCLUSION

In the present paper, we propose a practical *SVP*-based message-recovery attack on NTRU-HPS under partial plaintext leakage. Assuming that a subset of the coefficients of both the message and the nonce vector are known, we construct a linear system of modular equations whose underlying structure corresponds to a modular knapsack problem. We then reduce this problem to an instance of the *Shortest Vector Problem (SVP)* on a suitably defined lattice. By applying the FLATTER lattice reduction algorithm, we are able to solve the modular knapsack instance and successfully recover the unknown polynomial $r(x)$, and consequently, the message $m(x)$.

Knowing the 42–48% of the message and the nonce, we can attack the three variants of NTRU-HPS and recover the message in minutes on commodity hardware. For future work, we could address extensions of our method to other lattice-based schemes, such as Kyber and Saber.

Acknowledgment. The second author was co funded by SECUR-EU. The SECUR-EU project funded under Grant Agreement 101128029 is supported by the European Cybersecurity Competence Centre.

REFERENCES

- [1] K. Aardal, Cor A. J. Hurkens and A. K. Lenstra, Solving a System of Linear Diophantine Equations with Lower and Upper Bounds on the Variables, Integer Programming and Combinatorial Optimization. IPCO 1998, LNCS vol. **1412**, Springer 2000.
- [2] M. Adamoudis, K. A. Draziotis and D. Poulakis, Attacking (EC)DSA scheme with ephemeral keys sharing specific bits. Theoretical Computer Science, Vol. **1001**, June 2024, Elsevier, 2024.
- [3] M. Adamoudis, K. A. Draziotis and D. Poulakis, Enhancing a DSA attack, CAI 2019, p. 13-25. LNCS **11545**, Springer 2019.
- [4] M. Adamoudis and K. A. Draziotis, Message recovery attack on NTRU using a lattice independent from the public key, Advances in Mathematics of Communications Volume 19(1), 2025, doi:<http://dx.doi.org/10.3934/amc.2023040>, arXiv:<https://arxiv.org/abs/2203.09620>
- [5] M. Albrecht, S. Bai, and L. Ducas, A subfield lattice attack on overstretched NTRU assumptions. CRYPTO 2016. LNCS **9814**, Springer 2016.
- [6] D. J. Bernstein, B. B. Brumley, M. S. Chen, C. Chuengsatiansup, T. Lange, A. Marotzke, B. Y. Peng, N. Tuveri, C. van Vredendaal, B. Y. Yang, NTRU - Prime Algorithm Specifications And Supporting Documentation, 2020.
- [7] 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.
- [8] G. H. Bradley, Algorithms for Hermite and Smith normal matrices and linear Diophantine equations, Math. Comput. **25**, American Mathematical Society (1971) p. 897–907.
- [9] D. Coppersmith and A. Shamir, Lattice Attacks on NTRU. In Proc. Eurocrypt 1997, LNCS **1223**, Springer, 1997.
- [10] C. Chen, O. Danba, J. Hoffstein, A. Hülsing, J. Rijneveld, J. M. Schanck, T. Saito, P. Schwabe, W. Whyte, K. Xagawa, T. Yamakawa, and Z. Zhang, NTRU - Algorithm Specifications And Supporting Documentation, 2020.
- [11] 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.
- [12] K. A. Draziotis and A. Papadopoulou, Improved attacks on knapsack problem with their variants and a knapsack type ID-scheme, Advances in Mathematics of Communications, Volume 12, Issue 1, 2018, doi:<https://doi.org/10.3934/amc.2018026>.
- [13] L. Ducas, T. Espitau and E. W. Postlethwaite, Finding Short Integer Solutions When the Modulus Is Small, Crypto 2023, LNCS **14083**, Springer, <https://eprint.iacr.org/2023/1125.pdf>
- [14] N. Gama and P. Q. Nguyen, New Chosen-Ciphertext Attacks on NTRU. Public Key Cryptography, PKC 2007, LNCS **4450**, Springer 2007.
- [15] M. R. Garey and D. S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, Freeman, 1979.
- [16] C. Gentry, Key recovery and message attacks on NTRU-composite, EUROCRYPT 2001, LNCS **2045**, Springer 2001.
- [17] J. von zur Gathen and M. Sieveking, Weitere zum Erfüllungsproblem polynomial äquivalente kombinatorische Aufgaben, Komplexität von Entscheidungsproblemen, pp:49-71, 1976.
- [18] 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.
- [19] N. Howgrave-Graham, A Hybrid Lattice-Reduction and Meet-in-the-Middle Attack Against NTRU. CRYPTO 2007, LNCS **4622**, Springer 2007.
- [20] N. A. Howgrave-Graham and N. P. Smart, Lattice Attacks on Digital Signature Schemes, Des. Codes Cryptogr. **23**, p. 283–290, 2001.
- [21] 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>.

- [22] W.L. Huang, J.P. Chen and B.Y. Yang, Power Analysis on NTRU Prime. IACR Transactions on Cryptographic Hardware and Embedded Systems, Vol. **2020(1)**, DOI: <https://doi.org/10.13154/tches.v2020.i1.123-151>
- [23] E. Karabulut, E. Alkim and A. Aysu, Single-Trace Side-Channel Attacks on ω -Small Polynomial Sampling: With Applications to NTRU, NTRU Prime, and CRYSTALS-DILITHIUM, 2021 IEEE International Symposium on Hardware Oriented Security and Trust (HOST), Tysons Corner, VA, USA, 2021, pp. 35–45, doi: <https://doi.org/10.1109/HOST49136.2021.9702284>.
- [24] P. Kirchner and P. A. Fouque, Revisiting Lattice Attacks on Overstretched NTRU Parameters. Eurocrypt 2017, LNCS **10210**, Springer 2017.
- [25] P. Kirchner and T. Espitau and P. A. Fouque, An Improved LLL Algorithm, Advances in Cryptology – ASIACRYPT 2019, Springer, 2019
- [26] E. Kirshanova, A. May, and J. Nowakowski, New NTRU Records with Improved Lattice Bases. PQCrypto 2023, LNCS **14154**, 2023, doi : https://doi.org/10.1007/978-3-031-40003-2_7
- [27] R. M. Kolpakov and M. A. Posypkin, Upper and lower bounds for the complexity of the branch and bound method for the knapsack problem, Discrete Mathematics and Applications, 2010.
- [28] H. Koy and C. P. Schnorr, Segment Lattice Reduction, Proceedings of the Workshop on the Theory and Application of Cryptographic Techniques, EUROCRYPT '89, Springer, 1990
- [29] A. K. Lenstra, H. W. Lenstra, L. Lovász, Factoring polynomials with rational coefficients, Math. Ann. **261**, 515–534 (1982). <https://doi.org/10.1007/BF01457454>
- [30] A. May and J. Nowakowski, Too Many Hints – When LLL Breaks LWE, 2024, <https://eprint.iacr.org/2023/777.pdf>
- [31] A. Neumaier and D. Stehlé, Floating-Point LLL Revisited, Advances in Cryptology – EUROCRYPT 2016, Springer, 2016
- [32] M. Newman, The Smith normal form. Proceedings of the Fifth Conference of the International Linear Algebra Society, Linear Algebra Appl. **254**, p. 367–381, Elsevier 1997.
- [33] P. Q. Nguyen, Boosting the Hybrid Attack on NTRU: Torus LSH, Permuted HNF and Boxed Sphere, Third PQC Standardization Conference, 2021.
- [34] E. Poimenidou, M. Adamoudis, K. A. Draziotis, and K. Tsihclas, Message Recovery Attack in NTRU through VFK Lattices, preprint, <https://doi.org/10.48550/arXiv.2311.17022>
- [35] E. Poimenidou, M. Adamoudis, K. A. Draziotis, Towards message recovery in NTRU Encryption with auxiliary data, NuTMiC 2024, LNCS **14966**, Springer.
- [36] K. Ryan and N. Heninger, Fast Practical Lattice Reduction Through Iterated Compression, Advances in Cryptology – CRYPTO 2023, LNCS **14083**, Springer.
- [37] H. J. S. Smith, On systems of linear indeterminate equations and congruences. Phil. Trans. Roy. Soc. London **151**, p. 293–326, 1861.
- [38] A. Storjohann. Computing hermite and smith normal forms of triangular integer matrices. Linear Algebra and its Applications **282** p.25–45, Elsevier, 1998.
- [39] P. 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.
- [40] J. 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.

APPENDIX A. PROOF OF THE THEOREM 3.2

We start with the following Theorem.

Theorem A.1. (*Smith, 1861 [37]*). *Let $A \in \mathcal{M}_{m \times n}(\mathbb{Z})$ of rank r . Then there is a diagonal integer matrix $D = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_r, 0, \dots, 0)$ ($m \times n$) with*

$$\lambda_1 |\lambda_2| \cdots |\lambda_r|$$

and unimodular matrices $U \in GL_m(\mathbb{Z})$ and $V \in GL_n(\mathbb{Z})$ such that

$$A = UDV.$$

The non zero diagonal elements $\lambda_1, \dots, \lambda_r$ of D are called elementary divisors of A and are defined up to sign. D is the Smith Normal Form (SNF) of A .

The first algorithms for computing the Hermite Normal Form (HNF) and Smith Normal Form (SNF) appeared in 1971 [8]. Later, von zur Gathen and Sieveking [17] presented improved algorithms, in 1976 that achieved polynomial-time complexity. For $A \in \mathcal{M}_{m \times n}(\mathbb{Z})$ there are unimodular matrices P, Q such that

$$PAQ = \begin{bmatrix} \text{diag}(\lambda_1, \dots, \lambda_r) & \mathbf{0}_{r \times (n-r)} \\ \mathbf{0}_{(m-r) \times r} & \mathbf{0}_{(m-r) \times (n-r)} \end{bmatrix},$$

where r is the rank of A and $\lambda_i \in \mathbb{Z}_{>0}$, $\lambda_i | \lambda_{i+1}$. Storjohann [38, Theorem 12] provided a deterministic algorithm for computing SNF with complexity,

$$O(nmr^2 \log^2(r\|A\|) + r^4 \log^3(\|A\|)) \text{ bit operations,}$$

where $\|A\| = \max\{|a_{ij}|\}$.

If $\text{SNF}(A) = PAQ$, then from [32], it is proved that the last $n - r$ columns of Q is a basis for the integer lattice $AX = \mathbf{0}$.

Below we include the proof of Theorem 3.2. Let $D = PAQ$ be the SNF of A , $k \times N$ ($k < N$) matrix with $\text{rank}(A) = k$ (A is given in (3.3)). Also,

$$Q = [\mathbf{q}_1 \mid \dots \mid \mathbf{q}_N].$$

From properties of SNF, the $N - k$ vectors $\mathbf{q}_{k+1}, \dots, \mathbf{q}_N$ is a basis of $\ker_{\mathbb{Z}}(A)$. Let also the k vectors $\mathbf{y}_j = q\mathbf{e}_j$, $1 \leq j \leq k$, where $\mathbf{e}_j$ the j th vector of the standard basis of $\mathbb{R}^N$. We define $V = \text{span}(\mathbf{q}_{k+1}, \dots, \mathbf{q}_N)$ and $W = \text{span}(\mathbf{e}_1, \dots, \mathbf{e}_k)$.

Theorem. *If $V \cap W = \{\mathbf{0}\}$, then there are N_1, N_2 such that, the LLL-reduced matrix of B_k is of the form $\hat{B}_k$. In fact we prove that $2^{N+k}N_1^2 < c(N, k) < N_2^2$ for some constant $c(N, k)$.*

Proof. Let the matrix $\hat{B}_k = (\hat{b}_{i,j})$ be the LLL reduced matrix (row-wise) of B . We shall prove that it has the form of (3.9). It is enough to prove that $\hat{b}_{i,j} = 0$, for all $i \in \{0, \dots, N - 1\}$ and $j \in \{N + 1, \dots, N + k\}$.

We set $\mathbf{q}'_j = (\mathbf{q}_j, \mathbf{0}_{k+1})$ and $\mathbf{y}'_j = (\mathbf{y}_j, \mathbf{0}_{k+1})$ which belong to lattice $\mathcal{L} = L(B_k)$. Indeed, $A\mathbf{q}_j = \mathbf{0}$ and so $A\mathbf{q}'_j = \mathbf{0} \pmod{q}$ and also $A\mathbf{y}_j = \mathbf{0} \pmod{q}$. From the hypothesis $V \cap W = \{\mathbf{0}\}$, thus,

$$\{\mathbf{q}_{k+1}, \dots, \mathbf{q}_N\} \cup \{\mathbf{y}_1, \dots, \mathbf{y}_k\}$$

is a basis of $\mathbb{R}^N$. Moreover,

$$\{\mathbf{q}'_{k+1}, \dots, \mathbf{q}'_N\} \cup \{\mathbf{y}'_1, \dots, \mathbf{y}'_k\} \subset \mathcal{L}$$

is linear independent. We set,

$$c(N, k) = 2^{N+k} \max\{\|\mathbf{r}'\|^2, \|\mathbf{q}'_{k+1}\|^2, \dots, \|\mathbf{q}'_N\|^2, \|\mathbf{y}'_1\|^2, \dots, \|\mathbf{y}'_{k-1}\|^2\},$$

where $\mathbf{r}' = (\mathbf{r}, N_1, \mathbf{0}_k) \in \mathcal{L}$ for some $\mathbf{r}$ such that, $A\mathbf{r}^T = \mathbf{T}_k \pmod{q}$ ⁹.

The set $\{\mathbf{r}, \mathbf{q}_{k+1}, \dots, \mathbf{q}_N, q\mathbf{e}_1, \dots, q\mathbf{e}_{k-1}\}$ (N vectors) is linear independent in $\mathbb{R}^N$. Indeed, it is enough to prove that

$$\mathbf{r} \notin \text{span}(\mathbf{q}_{k+1}, \dots, \mathbf{q}_N, q\mathbf{e}_1, \dots, q\mathbf{e}_{k-1}).$$

⁹Since our system is of NTRU type there is always such a solution $\mathbf{r}$.

If $\mathbf{r}$ is a linear combination of the previous set, then by applying A we get $\mathbf{T}_k = \mathbf{0} \pmod{q}$, which is a contradiction since $\mathbf{T}_k \neq \mathbf{0} \pmod{q}$. Therefore, from Lemma 2.3 we get

$$(A.1) \quad \|\hat{\mathbf{b}}_j\|^2 \leq c(N, k), (0 \leq j \leq N-1).$$

We choose,

$$(A.2) \quad N_2^2 > c(N, k).$$

Suppose that $\hat{b}_{i_0, j_0} \neq 0$, for some $i_0 \in \{0, \dots, N-1\}$ and $j_0 \in \{N+1, \dots, N+k\}$. By the structure of B_k we have $(B_k)_{i_0, j} \equiv 0 \pmod{N_2}$ for all $j \in \{N+1, \dots, N+k+1\}$. Since LLL performs only unimodular integer row operations, this congruence is preserved; hence $(\hat{B}_k)_{i_0, j} \equiv 0 \pmod{N_2}$ for the same indices j . So, $N_2 | \hat{b}_{i_0, j_0}$, but $\hat{b}_{i_0, j_0} \neq 0$, thus we get $|\hat{b}_{i_0, j_0}| \geq N_2$. Therefore,

$$\|\hat{\mathbf{b}}_{i_0}\|^2 \geq (\hat{b}_{i_0, j_0})^2 \geq N_2^2 > c(N, k),$$

which contradicts (A.1) if we set $j = i_0$. We conclude that $\hat{b}_{i_0, j_0} = 0$, thus $\hat{b}_{i, j} = 0$ for all $i \in \{0, \dots, N-1\}$ and $j \in \{N+1, \dots, N+k\}$.

Finally, $c(N, k) \geq 2^{N+k} \|\mathbf{r}'\|^2 = 2^{N+k} (N_1^2 + \|\mathbf{r}\|^2) > 2^{N+k} N_1^2$. This shows that $2^{N+k} N_1^2 < c(N, k) < N_2^2$.

□

We have assumed that $A \in \mathbb{Z}^{k \times N}$ has full row rank. For a random (full-row-rank) integer matrix A , the condition $V \cap W = \{\mathbf{0}\}$ holds generically (i.e., for almost every choice of entries). Otherwise, the homogeneous system $AX = \mathbf{0}$ over $\mathbb{R}$ would have a nonzero solution of the form $(a_1, \dots, a_k, 0, \dots, 0)$.

E. POIMENIDOU, SCHOOL OF INFORMATICS, ARISTOTLE UNIVERSITY OF THESSALONIKI, GREECE
Email address: epoimeni@csd.auth.gr

K. A. DRAZIOTIS, SCHOOL OF INFORMATICS, ARISTOTLE UNIVERSITY OF THESSALONIKI, GREECE
Email address: drazioti@csd.auth.gr