Lattice Basis Reduction Attack on Matrix NTRU

Thiago do Rêgo Sousa[1], **Tertuliano Souza Neto**[1]

[1]CEPESC, Brasília - DF

Abstract. *NTRU is one of the most important post-quantum cryptosystems nowadays and since its introduction several variants have been proposed in the literature. In particular, the Matrix NTRU is a variant which replaces the NTRU polynomials by integer matrices. In this work, we develop a lattice-based reduction attack on the Matrix NTRU cryptosystem that allows us to recover the plaintext. We also show that this system is completely vulnerable to the proposed attack for parameters that could be used in practice. In addition, we give sufficient conditions to avoid decryption failure for the Matrix NTRU.*

1. Introduction

Quantum computers are now a reality and if large-scale ones are built, they could break most of the public key cryptosystems used currently. That is one of the reasons that research on post-quantum systems (cryptographic systems underpinned on mathematical problems that are intractable by both quantum and conventional computers) has taken off in the last decades.

In 2016, the National Institute of Standards and Technology (NIST) started a public process for the standardization of post quantum algorithms that could be used to interoperate with existing communications protocols and networks currently used. One of the main mathematical problem underpinning such system is the shortest vector problem (SVP), which aims at finding a shortest vector in a structure called lattice (see [Silverman et al. 2008]).Two lattice-based systems have progressed significantly in the NIST Post-Quantum Cryptography standardization process [NIST]: the NTRU [Chen et al. 2020] and the NTRU Prime systems [Daniel J. Bernstein et al. 2024], both based on the original NTRU system from [Hoffstein et al. 1998].

The NTRU system was presented in 1996 to the cryptographic community during the CRYPTO 96 rump session. Since its introduction, the NTRU system has been extensively analyzed and extended in many different ways. Interesting review articles containing a wealth of examples and references include [Singh and Padhye 2016, Salleh and Kamarulhaili 2020] and [Mittal and Ramkumar 2022]. NTRU is based on modular algebra of polynomials in specific rings and such generalizations range from changing the polynomial coefficients to using matrix structures.

The first NTRU generalization using matrices was introduced in [Coglianese and Goi 2005], where key generation, encryption and decryption operate over matrices whose entries are polynomials. In 2008, [Nayak et al. 2008] proposed a matrix-based system as a variant of NTRU using only matrices with integer entries, replacing the arithmetic of truncated polynomials with a simple modular arithmetic on matrices. This new system, called matrix NTRU, was subject to some further analysis and even suggested for real data applications. Indeed, [Kumar et al. 2013] presented a framework for deploying matrix NTRU in real data applications.

Since its introduction the matrix NTRU has attracted some attention. A comparative study regarding speed and key sizes was performed in [Nayak et al. 2012] showing that matrix NTRU was faster than the classical NTRU for short dimensions. In addition, [Nayak et al. 2012] argue that matrix NTRU was more secure than the classical NTRU since matrices are noncommutative. However, we will show that the matrix NTRU system is seriously affected by the lattice attack developed in Section 5. This demonstrates that the matrix NTRU is far weaker than the classical NTRU system for comparable key sizes, contrary to what was accredited in the literature (see [Nayak et al. 2012] and [Kumar et al. 2013]).

Subsequent works on the matrix NTRU focused on expanding the key space and on eliminating restrictions on the matrices representing the public key by using Gram-Schmidt orthogonalization and companion matrices ([Luo and Lin 2011, Tripathi et al. 2016, Mamdikar et al. 2018]).

With regards to the system's security, [Nayak et al. 2011] applied a reaction type attack to matrix NTRU, by adapting the results of the original attack presented in [Hall et al. 1999] for the NTRU system. In addition, a meet in the middle attack was recently presented in [Wijayanti et al. 2023]. However, the simulation studies presented in these works only show the performance of the aforementioned attacks for very low dimensions of the matrix NTRU, which cannot be used in practice.

Our main contribution to matrix NTRU is in developing a lattice-based attack and analyse its resistance in dimensions much higher than the ones suggested for practical applications in [Kumar et al. 2013]. We will show in Section 5 how this attack can recover the private key (up to a permutation). In addition, this attack is further used to construct a message recovery attack that works for a dimension of the system of up to 110. According to [Nayak et al. 2012], a matrix NTRU system such as this is equivalent to an NTRU system using polynomials of degree 110^2, which is not even close to being broken by lattice-based techniques ([Chen et al. 2020]).

In addition to the lattice attack, we also analyse the decryption failure rate of the matrix NTRU. The results of Proposition 2 give sufficient conditions that depend only on the system's public parameters to avoid decryption failure.

The rest of the paper is organized as follows. In Section 2, we describe the NTRU cryptosystem in details and a brief description of Matrix NTRU is given in Section 3. Sufficient conditions to avoid decryption failure for the Matrix NTRU are also presented in this section. Then, in Section 4, we construct the lattice structure associated with the matrix NTRU and report the results of an attack to recover the private key. This attack is further refined in Section 5 to construct a message recovery type attack that works for a dimension of up to $n^2 = 110^2$ on a personal computer. The results of the previous section are summarized in Section 6 where we explain why the Matrix NTRU is far weaker than the NTRU cryptosystem concerning the lattice attack.

2. The NTRU Cryptosystem

Let n and p be prime numbers. Let $q \geq 1$ be an integer such that $(n, q) = (p, q) = 1$. The main arithmetic operations in the NTRU cryptosystem are calculations over polynomials defined over the rings R, R_p e R_q defined as:

$$R = \frac{\mathbb{Z}[x]}{(x^n - 1)}, R_p = \frac{(\mathbb{Z}/p\mathbb{Z})[x]}{(x^n - 1)}, R_q = \frac{(\mathbb{Z}/q\mathbb{Z})[x]}{(x^n - 1)}.$$

We can notice that the ring R is related to the other two. In fact, for every $a(x)$ in R, we can identify it to an element in R_p or R_q by applying reduction modulo p or q, respectively, over the coefficients of $a(x)$. We say that $a(x)$ is a ternary polynomial if its coefficients lie in the set $\{-1, 0, 1\}$.

The NTRU cryptosystem works as follows.

1. Key Generation: Alice chooses two ternary polynomials at random, namely, $f(x)$ and $g(x)$. Next, Alice tries to compute the inverse of f over the rings R_q, R_p, i.e, $f_q(x) = f(x)^{-1} \in R_q$ and $f_p(x) = f(x)^{-1} \in R_p$ until she succeeds. Finally, she computes the polynomial
$$h(x) = f_q(x) * g(x) \in R_q,$$
where $*$ denotes polynomial multiplication in R_q, i.e., a cyclic convolution product as defined in [Hoffstein et al. 1998, Section 1.1] The polynomial $h(x)$ is Alice's public key and the pair $(f(x), f_p(x))$ is her private key.
2. Encryption: Suppose Bob wants to send an encrypted message to Alice and let $m(x) \in R_p$ be Bob's plaintext. Next, Bob chooses, at random, a ternary polynomial $r(x)$ and computes
$$e(x) \equiv ph(x) * r(x) + m(x) \mod q.$$
Notice that Bob's ciphertext $e(x)$ is an element of the ring R_q.
3. Decryption: Once Alice receives Bob's ciphertext $e(x)$, she starts the decryption process by computing
$$a(x) \equiv f(x) * e(x) \mod q.$$
Then the reduction modulo p gives the desired plaintext
$$b(x) \equiv f_p(x) * a(x) \mod p.$$

Because of the randomness of the polynomial $r(x)$, NTRU is a probabilistic cryptosystem, since a message $m(x)$ can be encrypted to several ciphertexts $ph(x) * r(x) + m(x)$, depending on each instance of $r(x)$. However, in doing so, we can introduce a vulnerability into the NTRU cryptosystem, since some ciphertext may not decrypt correctly to the original message, a phenomenon known as a decryption failure. Some attacks in the literature take advantage of these decryption failures [Howgrave-Graham et al. 2003, Gama and Nguyen 2007, Jaulmes and Joux 2000] and therefore we should choose the parameters carefully.

3. The Matrix NTRU Cryptosystem

Let p be a prime number and $q >> p$ an integer such that $(p, q) = 1$. Let

$$A = \begin{pmatrix} a_{11} & \dots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \dots & a_{nn} \end{pmatrix}$$

be an $n \times n$ matrix whose entries are integer numbers, that is, $A \in M_n(\mathbb{Z})$. In a similar way we have done before with polynomials, we say that A is a ternary matrix if all of its entries lie in the set $\{-1, 0-, 1\}$.

We say that A is reduced modulo p, denoted as $A \mod p$, if every entry of A is reduced modulo p. Therefore

$$A \mod p = \begin{pmatrix} a_{11} \mod p & \dots & a_{1n} \mod p \\ \vdots & \ddots & \vdots \\ a_{n1} \mod p & \dots & a_{nn} \mod p \end{pmatrix}.$$

Notice that, in the latter case, we can view matrix A as an element of the ring $M_n(\mathbb{F}_p)$. In this work, we are going to consider the center lift operation modulo p and q in the decryption process. In that case, it can be useful to consider a noncanonical representation of the elements on the rings. Therefore, when we are dealing with a prime modulus, we have

$$\mathbb{F}_p = \left\{-\frac{p-1}{2}, \dots, \frac{p-1}{2}\right\}$$

and for a composite number, we have

$$\mathbb{Z}_q = \left\{-\frac{q}{2}+1, \dots, \frac{q}{2}\right\}.$$

Modular arithmetic over the rings $M_n(\mathbb{F}_p)$ and $M_n(\mathbb{Z}_q)$ is what underpins the Matrix NTRU system. In such rings we can add, subtract and multiply matrices in the same way we have done with matrices over the field of real numbers. However, in order to invert a matrix in $M_n(\mathbb{F}_p)$ we have to be sure that p and the determinant of A are relatively prime [Jacques-García et al. 2022, Wijayanti et al. 2023]. If that is the case, then there exists a (unique) matrix B such that $AB = BA = I$ in $M_n(\mathbb{F}_p)$, where I is the identity matrix. The matrix B is called the inverse of A modulo p and denoted $A^{-1} \mod p$.

The matrix NTRU system works as follows.

1. Key generation: In the Matrix NTRU cryptosystem, the private and public keys are $n \times n$ matrices. First, we choose a pair of ternary matrices F, G such that $F \mod p$ and $F \mod q$ are invertible in $M_n(\mathbb{F}_p)$ and $M_n(\mathbb{Z}_q)$, respectively. Then we compute the matrices F_p and F_q, where $F_p = F^{-1} \mod p$ and $F_q = F^{-1} \mod q$. The pair F, F_p is the private key and the parameters F, G, F_p, F_q should be kept in secret. Now, we can compute the public key by performing the calculation
$$H = pF_qG \mod q.$$
2. Encryption:To encrypt a message, we encode it as a matrix $M \in M_n(\mathbb{F}_p)$ and choose a ternary matrix $R \in M_n(\mathbb{Z})$, at random. Now, we compute the ciphertext as
$$E = HR + M \mod q.$$
3. Decryption: To decrypt, we compute
$$A = FE \mod q,$$

where F is the private key we saw previously. Next, we reduce A modulo p and compute

$$B = F_pA \mod p.$$

If everything went well, the matrix B will be precisely the message M we started with before the application of the encryption function.

However, depending on the parameters selection, we can have errors when operating on Matrix NTRU, just as we have when dealing with NTRU cryptosystem. Sometimes, the matrices B and M can be different and we say there is a decryption failure if that is the case. We notice that the authors [Nayak et al. 2008] did not address that question. However, it turns out that this is an important issue concerning any probabilistic cryptosystem. To fill such a lack in their proposed algorithm, we prove the following result, which shows that there are parameter selections to prevent failure decryption on Matrix NTRU.

Proposition 1. *Suppose that p, q and n are fixed parameters for the Matrix NTRU defined above. If $n(3p-1) < q$, then any ciphertext E resulting from the encryption of a message M with private key F, decrypts correctly to M.*

Proof. During decryption, one computes

$$\begin{aligned} A &= FE \mod q \\ &= F(HR + M) \mod q \\ &= FHR + FM \mod q \\ &= pFF_qGR + FM \mod q \\ &= pGR + FM \mod q. \end{aligned} \tag{1}$$

Following the same arguments as in the proof of Proposition 6.48 in [Silverman et al. 2008], we need to bound the largest coefficient (in modulus) of each entry of the matrix $pGR + FM$ computed exactly (without module q). For decryption to work, the above matrix should have entries whose absolute value does not exceed $q/2$. Since every entry of the matrices G and R lie in $\{-1, 0, 1\}$, the largest possible entry of the product GR is upper bounded by n. On the other hand, the matrix M has coefficients between $-(p-1)/2$ and $(p-1)/2$ and F has coefficients lying in $\{-1, 0, 1\}$. Therefore, all coefficients of the product FM can be bounded by $n(p-1)/2$. Finally, the entries of the matrix $pGR + FM$ are all upper bounded in module by $np + n(p-1)/2 = n(3p-1)/2$. Under the assumption of the proposition, this can be further bounded by $q/2$ which will ensure correct decryption of the ciphertext. $\square$

We implemented the Matrix NTRU in [Team 2024] and assessed the decryption failure rate for a set of different parameters. Table 1 shows the results:

4. Lattice attack on the private key

In the following section we show how the problem of finding the private key F of the matrix NTRU system is connected to solving a well-known problem in lattices, namely finding the shortest vector in a lattice. A lattice can be defined in the following way.

Table 1. Decryption failure estimated probability for the Matrix NTRU system for fixed $p = 3$ and varying n and q. For each parameter setting a new key and message were generated and encrypted. Results present the proportion of messages decrypted correctly over 10000 repetitions.

	q										
n	32	64	79	128	256	307	512	701	1024	2048	4096
5	0.008	0	0	0	0	0	0	0	0	0	0
10	0.775	0	0	0	0	0	0	0	0	0	0
20	1	0.210	0.008	0	0	0	0	0	0	0	0
30	1	0.989	0.397	0	0	0	0	0	0	0	0
40	1	1	0.989	0.002	0	0	0	0	0	0	0
50	1	1	1	0.039	0	0	0	0	0	0	0
60	1	1	1	0.257	0	0	0	0	0	0	0
70	1	1	1	0.734	0	0	0	0	0	0	0
80	1	1	1	0.984	0	0	0	0	0	0	0
90	1	1	1	1	0	0	0	0	0	0	0
100	1	1	1	1	0	0	0	0	0	0	0
110	1	1	1	1	0	0	0	0	0	0	0

Definition 1. ([Silverman et al. 2008, Section 6.4]) Let $f_1, \ldots, f_n \in \mathbb{R}^n$ be a set of linearly independent vectors. The lattice L generated by $f_1, \ldots, f_n$ is the set of linear combinations of $f_1, \ldots, f_n$ with coefficients in $\mathbb{Z}$,

$$L = \{\alpha_1 f_1 + \cdots + \alpha_n f_n : \alpha_1, \ldots, \alpha_n \in \mathbb{Z}\}. \tag{2}$$

The set of vectors $f_1, \ldots, f_n$ is called the lattice basis and it is usual to stack them into a matrix and work with the matrix as being the lattice basis that generates L. A fundamental problem in lattice is finding a shortest nonzero vector in the lattice which minimizes the Euclidean norm $\|f\|$. This is called the shortest vector problem (SVP). It is important to notice that the SVP problem asks for a shortest vector and not the shortest vector since e.g. f and $-f$ have the same Euclidean norm. According to [Silverman et al. 2008] Finding a solution for the SVP problem can be used to break various cryptosystems, in particular the NTRU system from Section 2 (for certain parameters) and as we will show next how it can be used to finding the private key F in the matrix NTRU system. It is worth noticing that the current version of NTRU submitted to NISTs post quantum competition [Chen et al. 2020] have parameters that avoid private key attacks with current computational resources and it is practical. On the other hand, the matrix NTRU variant has a serious vulnerability in its construction, that makes it breakable for quite high values of parameters that could be used in practice.

Proposition 2. *Let F, G be the private keys and let H be the public key of the matrix NTRU with parameters n, p, q. Let (f_k, g_k) be the k-th line of F and G respectively and p^{-1} be the inverse of p module q. Then, (f_k, g_k) belongs to the lattice generated by the lines of the $2n \times 2n$ block matrix*

$$L = \begin{pmatrix} I_n & p^{-1}H \\ 0_n & qI_n \end{pmatrix}, \tag{3}$$

where I_n is the n dimensional identity matrix and 0_n is the n dimensional zero matrix. In other words, the unknown vector (f_k, g_k) can be written as an integer linear combination of the lines of L (which has only known quantities).

Proof. The public key is defined as $H = pF_qG$, where p and q are coprime, and F and G are the private keys created during the key generation step. Multiplying both sides of the above equation by Fp^{-1} we get $Fp^{-1}H = G \mod q$. This implies that there exists a matrix $A \in M(\mathbb{Z})$ such that

$$G = F(p^{-1}H) + qA, \tag{4}$$

Let $V = (\alpha H)$ and denote by f_k, g_k and a_k the k-th line of the matrices F, G and A. Equation (4) implies that $g_k = f_kV + qa_k$. Therefore, the $1 \times 2n$ vector (f_k, g_k) can be written as

$$\begin{aligned}(f_k, g_k) &= (f_k, f_kV + a_k(qI_n)) \\ &= (f_kI_n + a_k0_n, f_kV + a_k(qI_n)) \\ &= f_k(I_n, V) + a_k(0_n, qI_n) \\ &= f_{k1}(1, 0, \dots, 0, 0, v_{11}, v_{12}, \dots, v_{1n}) \\ &+ f_{k2}(0, 1, \dots, 0, 0, v_{21}, v_{22}, \dots, v_{2n}) \\ &\dots \\ &+ f_{kn}(0, 0, \dots, 0, 1, v_{n1}, v_{n2}, \dots, v_{nn}) \\ &+ a_{k1}(0, 0, \dots, 0, 0, q, 0, \dots, 0) \\ &+ a_{k2}(0, 0, \dots, 0, 0, 0, q, \dots, 0) \\ &\dots \\ &+ a_{kn}(0, 0, \dots, 0, 0, 0, 0, \dots, q).\end{aligned} \tag{5}$$

Since all $1 \times 2n$ dimensional vectors at the rhs of (5) equation are exactly the lines of the matrix L in (3), the proof is completed. □

Recall that the matrices F and G have only coefficients in $\{-1, 0, 1\}$, and therefore any line of the type (f_k, g_k) is a relatively short vector in the lattice L for large q. Indeed, using a Gaussian heuristic, the shortest vector expected from a lattice L depends only on the dimension of L and its determinant. Since L is upper triangular, we have $\det(L) = q^n$. Therefore, the length of the shortest vector expected from L is

$$l = \sqrt{\frac{n}{2\pi e}}(\det(L))^{1/(2n)} = \sqrt{\frac{nq}{2\pi e}}.$$

The target vector (f_k, g_k) has varying norm depending on how the private key is chosen. If every entry is chosen with uniform probability on $\{-1, 0, 1\}$, then, its expected norm is $\alpha = \sqrt{2n/3}$. This means that for higher values of q, the target vector has norm smaller than the expected by the by the Gaussian heuristics, which means that the LLL algorithm has a high probability of find a short vector in L (see [Silverman et al. 2008]).

Another interesting fact is that we do not need to attack the whole private key F which has dimension $2n$. Using a suitable algorithm to solve the SVP in L we can try to find any line of F and hopefully all lines separately reducing the complexity of the attack by a factor of n instead of n^2 (when the attacker is interested in obtaining all entries of F at once).

Solving the SVP problem can be done in many ways, and one commonly done is by using lattice-based reduction algorithms such as the famous LLL algorithm from [Lenstra et al. 1982] and or the BKZ algorithms [Chen and Nguyen 2011]. For more details on the LLL algorithm see e.g. [Bremner 2011, Chapter 4]. These base reduction algorithms try to find a shorter basis for the lattice L, and in doing so, they usually return potential short vectors for the lattice L and we can test if they correspond to any line of the matrix F.

In addition to using the LLL algorithm, there is an improved variant of lattice based reduction algorithms called BKZ as defined in [Chen and Nguyen 2011]. The BKZ algorithm, proceeds with repeated local improvements to the lattice basis. One of its simple implementations is a recurring set of calls to SVP oracle solvers of dimension set by the block size parameter of the BKZ algorithm and LLL calls and will be used in the following for recovering the private key. Although these algorithms can be further refined to lead to better solutions for the SVP problem, the first vectors of the BKZ output are already short enough to give us candidates for lines of the private key matrix F.

In what follows we use the result of Proposition 2 to recover the private key matrix F in the following way.

Algorithm 1: Recovering private key *F*:

Let H the public key of the matrix NTRU system with parameters n, p, q.

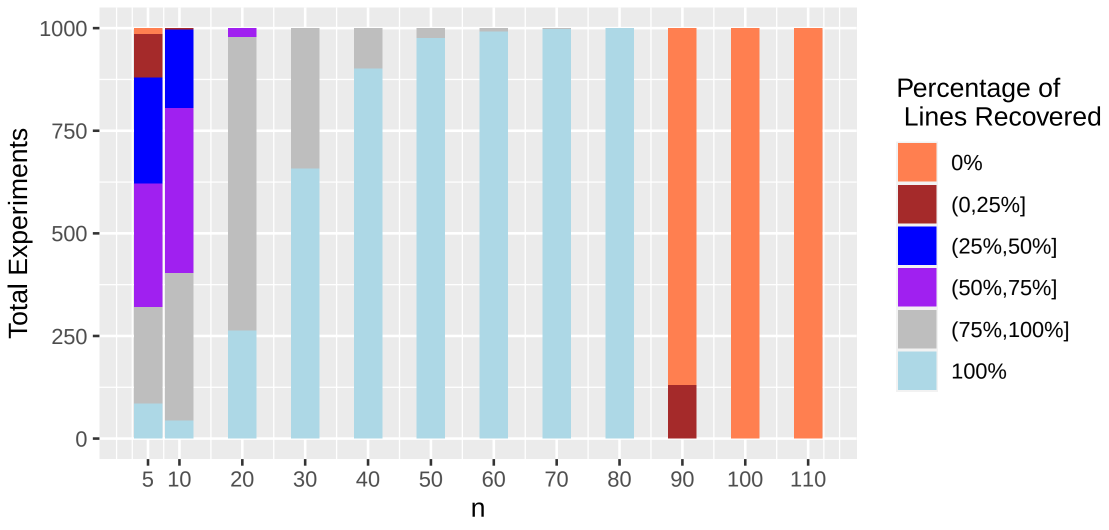
1. Compute $p^{-1} \mod q$and create the matrix L from eq.(3).
2. Run the BKZ algorithm on L and get L_{red}.
3. Use L_{red} to create a submatrix L^*_{red} with entries corresponding to the first n columns of L_{red}.
4. For each line f of the matrix F, check whether or not f is contained in one of the lines of L^*_{red}.

We implemented the above algorithm to assess the performance of the attack for several values of the parameters. Namely, we choose $q \in \{256, 4096\}$ for varying n and recorded what proportion of the lines of F one can recover from the attack using only the public key and public parameters p, q, n.

We see that the attack, in almost all experiments, can recover all lines of the matrix F for $n \in 40, 50, \ldots, 80$ and $q = 256$ and for $n \in 40, 50, \ldots, 110$ and $q = 4096$. For the case $q = 4096$ we also tried $n = 113$ and obtained success for quite some experiments, but in some cases the BKZ algorithm terminates with the message `terminate called recursively` and does not return the reduced basis.. At this stage, the lattice dimension is already $2n = 226$ and it gets more difficult to reduce the basis. For all settings we set the block of the BKZ algorithm to 10 since this allowed quite good results. One could increase the block size at the expense of a bigger running time.

One issue with the attack of algorithm 1 is that even though we can recover some lines of the matrix F, we still do not know how to reorder them to reconstruct the true F and use it for decryption. Nevertheless, we will see in the next Session that any permutation of lines of the private key F can be used to successfully decrypt a message encrypted

Figure 1. Performance of the attack of algorithm 1 for recovering lines of the matrix F for the matrix NTRU with parameters n and fixed $q = 256$ (upper figure) and $q = 4096$ (bellow figure). For each experiment, we generate new keys F and H and apply the attack recording how many experiments were able to recover all lines of F (100%), between 1 and $n-1$ lines of F and none of the lines of F (0%).



with F and this allows us to construct a message recovery attack on the matrix NTRU system.

5. Message recovery attack

In the previous section we showed how to construct the associated lattice for the matrix NTRU system and demonstrated the viability of the attack for recovering, up to a permutation, the whole private key matrix F. In what follows we show that this attack can be used to successfully decrypt a message encrypted with F, even though the correct order of the lines of F is not known by the attacker.

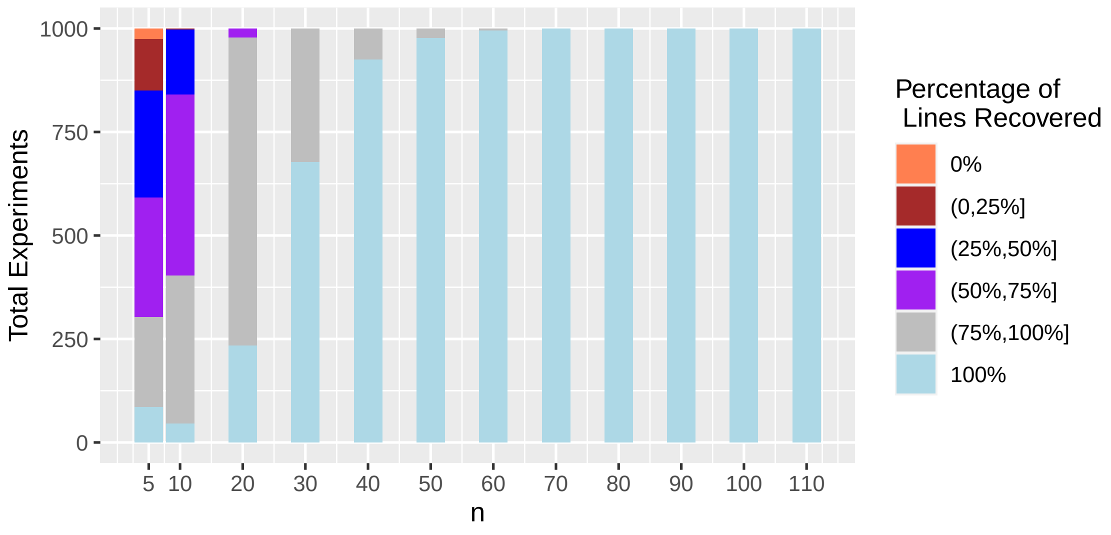
Proposition 3. *Let F, H be a corresponding private and public key pair for the matrix NTRU system with parameters n, p, q and let E be a ciphertext associated with the encryption of a message M such that the decryption process of E using F correctly returns M. Let F^* be a matrix formed by permuting the lines of the matrix F. Then, applying the decryption process to E using F^* returns the message M.*

Proof. By the construction of F^*, there exists an unimodular matrix D_1, such that $F^* = D_1F = FD_2$. Applying the decryption process gives us

$$A_2 = F_2E \mod q = D_1F(HR + M) \mod q.$$

Using the definition of the matrix H and the fact that $FF_q = I$ we get,

Figure 2. Same settings as in Figure 3 but with $q = 4096$.



$$\begin{aligned} A_2 &= D_1F(HR + M) \mod q \\ &= D_1F(pF_qGR + M) \mod q \\ &= D_1pFF_qGR + D_1FM \mod q \\ &= pD_1GR + D_1FM \mod q \end{aligned} \tag{6}$$

Next we are going to make this equations modulo p and notice that since D_1 only change signs and permute lines of GR, we can still make this operation without incurring in the risk of extrapolating the norm of the entries of the matrix D_1GR. Therefore, we have

$$A_2 \mod p = 0 + D_1FM = F^*M.$$

Finally, using the inverse of F_p^* of $F^* \mod p$ we compute

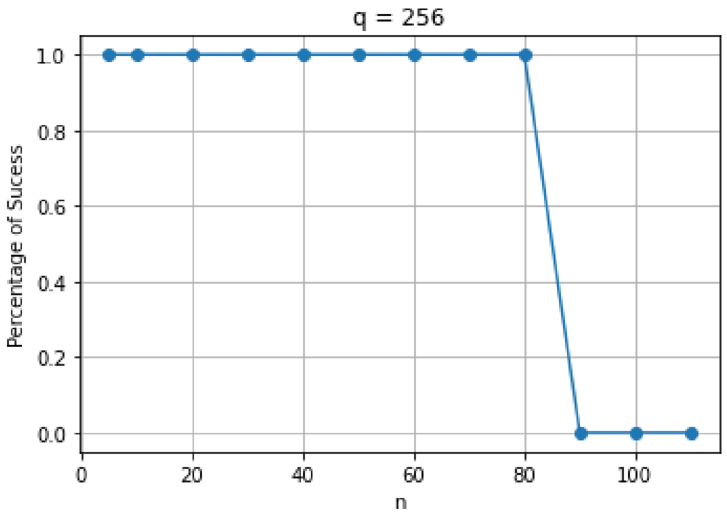
$$C_2 = F_p^*A_2 = F_p^*F^*M \mod p = M.$$

□

We use the result of Proposition 3 in combination with Algorithm 1 to create a practical message recovery attack. This attack uses the upper left part of the reduced lattice basis returned by the BKZ algorithm as a potential key. As we saw in Section 4, this gives us in several cases the whole private key (up to a permutation of lines). The result of Proposition 3 says this potential key can still be used to decrypt a message successfully and this is tested in practice.

For each experiment, we generate new key pair F, H a new message M, encrypt it and try to decrypt with the key F^* returned from the attack of algorithm 1. The experiment was repeated 100 times on an Intel(R) Core(TM) i5-9500T CPU 2.20GHz and the success rate reported in Figures 3 and 4. In the worst case scenario ($n = 110$ and

Figure 3. Performance of the attack of algorithm 1 for recovering lines of the matrix F for the matrix NTRU with parameters n and fixed $q = 256$ (upper figure) and $q = 4096$ (bellow figure). For each experiment, we generate new key pair F, H a new message M, encrypt it and try to decrypt with the key F^* returned from the attack of algorithm 1. The experiment was repeated 100 times and the sucess percentage recorded.

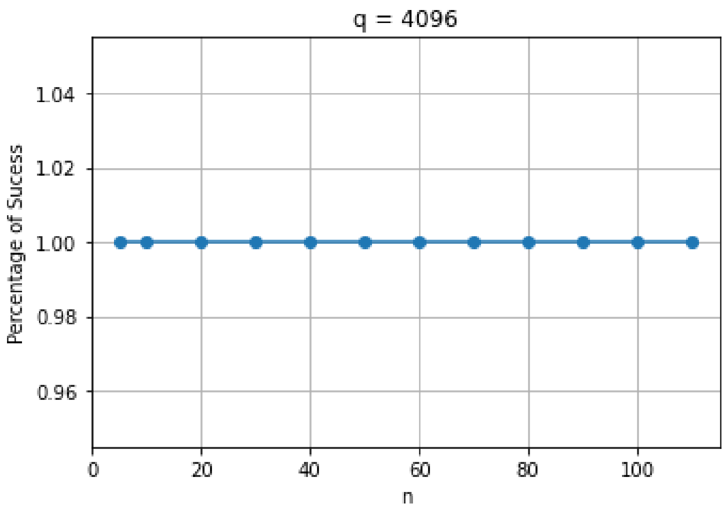


$q = 4096$) we can recover the true message in less than one minute. For $q = 256$ the attack works for dimension until $n = 80$ and for $q = 4096$ it works for dimensions up to 100. Since [Nayak et al. 2012] a matrix NTRU of dimension n with an NTRU of dimension n^2, this clearly shows that matrix NTRU is far weaker than NTRU since an NTRU with dimension $100^2 = 10000$ is far from being broken. In fact, the highest security suggested for NTRU in practical applications has parameters $n = 821$ and $q = 4096$ as shown in [Chen et al. 2020, Section 1.6]. By the results of this section, the matrix NTRU of dimension n^2, should be comparable (in terms of the lattice attack) to at most an NTRU of dimension n. Therefore, even though the Matrix NTRU can encode a similar number of bits in the private key as the NTRU, its internal structure makes it vulnerable to attacks in a much smaller dimension.

6. Conclusion

In [Nayak et al. 2012], the authors compare the speed performance for encryption and decryption of the classical NTRU and the Matrix NTRU arguing that matrix NTRU is faster than NTRU for comparable parameters. They compare a matrix NTRU with parameters n with an NTRU with parameter n^2, since the original idea of the matrix NTRU system is to encode the private key polynomial of degree n^2 into a matrix of dimension $n \times n$. In terms of brute force search for the private key they are comparable. In terms of lattice-based attacks, the post-quantum NIST finalist NTRU encrypt (REF) with parameters $n = 509$ and $q = 2048$ has already moderate security and cannot be broken by current the available techniques using lattice attacks. The equivalent matrix NTRU would have dimension around 23, an integer approximation to $\sqrt{509}$. However, a matrix NTRU with parameters $n = 23$ is completely vulnerable to lattice attacks on a personal computer as showed in Section 4. In fact we can attack such system for even higher dimension.

Figure 4. Same settings as in Figure 3 but with $q = 4096$.



The bottleneck of the proposed attack is in finding a suitable short vector for the matrix NTRU system using lattice basis reduction algorithms. Therefore, any improvement on these techniques such as replacing the BKZ routine by, e.g., the ones described in ([Albrecht and Ducas 2021], [May and Silverman 2001], and [Bi and Han 2021, Zhao and Ye 2023]) would improve the results showed in Figures 1 and 2.

This vulnerability of the matrix NTRU system is very serious since its security drooped from n^2 to n using the refereed lattice attacks developed here. In comparisons with the NTRU the matrix NTRU tries to creates a faster variant of NTRU by separating the private key into slices (lines of a matrices) and using this to create the public key. On the other hand this come at a very high security cost since just one line is used at a time to construct the lines of the public key. That means it runs faster at the cost of reducing the diffusion during the creation of the public key. On the other hand, NTRU creates public keys by using convolution of polynomials, and therefore, every coefficient contributes to create every coffined of the public key.

To conclude, we believe that the proposed attack can be adapted to other NTRU like systems relying on matrices operations such as the one based on the Gaussian Integers.

References

Albrecht, M. and Ducas, L. (2021). Lattice attacks on ntru and lwe: a history of refinements. *Cryptology ePrint Archive*.

Bi, J. and Han, L. (2021). Lattice attacks on ntru revisited. *IEEE Access*, 9:66218–66222.

Bremner, M. (2011). *Lattice basis reduction*. CRC Press New York.

Chen, C., Danba, O., Hoffstein, J., Hulsing, A., Rijneveld, J., Schanck, J. M., Schwabe, P., Whyte, W., and Zhang, Z. (2020). Ntru: algorithm specifications and supporting documentation (2019). *URL: https://csrc.nist.gov/Projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions.*, 1.

Chen, Y. and Nguyen, P. Q. (2011). Bkz 2.0: Better lattice security estimates. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 1–20. Springer.

Coglianese, M. and Goi, B.-M. (2005). Matru: A new ntru-based cryptosystem. In *Progress in Cryptology-INDOCRYPT 2005: 6th International Conference on Cryptology in India, Bangalore, India, December 10-12, 2005. Proceedings 6*, pages 232–243. Springer.

Daniel J. Bernstein, Tanja Lange, Chitchanok Chuengsatiansup, and Peter Schwabe (Accessed: 2024). NTRU Prime. `https://ntruprime.cr.yp.to`. Website.

Gama, N. and Nguyen, P. Q. (2007). New chosen-ciphertext attacks on ntru. In *International Workshop on Public Key Cryptography*, pages 89–106. Springer.

Hall, C., Goldberg, I., and Schneier, B. (1999). Reaction attacks against several public-key cryptosystem. In *Information and Communication Security: Second International Conference, ICICS'99, Sydney, Australia, November 9-11, 1999. Proceedings 2*, pages 2–12. Springer.

Hoffstein, J., Pipher, J., and Silverman, J. H. (1998). Ntru: A ring-based public key cryptosystem. In *International algorithmic number theory symposium*, pages 267–288. Springer.

Howgrave-Graham, N., Nguyen, P. Q., Pointcheval, D., Proos, J., Silverman, J. H., Singer, A., and Whyte, W. (2003). The impact of decryption failures on the security of ntru encryption. In *Annual International Cryptology Conference*, pages 226–246. Springer.

Jacques-García, F. A., Uribe-Mejía, D., Macías-Bobadilla, G., and Chaparro-Sánchez, R. (2022). On modular inverse matrices a computation approach. *South Florida Journal of Development*, 3(3):3100–3111.

Jaulmes, É. and Joux, A. (2000). A chosen-ciphertext attack against ntru. In *Annual international cryptology conference*, pages 20–35. Springer.

Kumar, V., Mamdikar, M. R., and Gosh, D. (2013). Matrix formulation of ntru algorithm using multiple public keys from matrix data bank for high degree polynomials. In *CEEE*, pages 191–198.

Lenstra, A. K., Lenstra, H. W., and Lovász, L. (1982). Factoring polynomials with rational coefficients. *Mathematische annalen*, 261:515–534.

Luo, X.-R. and Lin, C.-H. J. (2011). Discussion on matrix ntru. *IJCSNS International Journal of Computer Science and Network Security*, 11(1):32–35.

Mamdikar, M. R., Kumar, V., and Ghosh, D. (2018). Implementation of automatic invertible matrix mechanism in ntru matrix formulation algorithm.

May, A. and Silverman, J. H. (2001). Dimension reduction methods for convolution modular lattices. In *International Cryptography and Lattices Conference*, pages 110–125. Springer.

Mittal, S. and Ramkumar, K. (2022). A retrospective study on ntru cryptosystem. In *AIP Conference Proceedings*, volume 2451. AIP Publishing.

Nayak, R., Pradhan, J., and Sastry, C. (2011). Reaction attacks in the matrix scheme of ntru cryptosystem. In *International Conference on Advances in Information Technology and Mobile Communication*, pages 27–32. Springer.

Nayak, R., Pradhan, J., and Sastry, C. (2012). Evaluation of performance characteristics of polynomial based and lattice based nrtu cryptosystem.

Nayak, R., Sastry, C., and Pradhan, J. (2008). A matrix formulation for ntru cryptosystem. In *2008 16th IEEE International Conference on Networks*, pages 1–5. IEEE.

NIST. Post-Quantum Cryptography Standardization. https://csrc.nist.gov/Projects/post-quantum-cryptography/post-quantum-cryptography-standardization.

Salleh, N. and Kamarulhaili, H. (2020). Ntru public-key cryptosystem and its variants: An overview. *Int. l J. of Cryptology Research*, 10(1):1–21.

Silverman, J. H., Pipher, J., and Hoffstein, J. (2008). *An introduction to mathematical cryptography*, volume 1. Springer.

Singh, S. and Padhye, S. (2016). Generalisations of ntru cryptosystem. *Security and Communication Networks*, 9(18):6315–6334.

Team, S. D. (2024). *SageMath*. Available from `https://www.sagemath.org`.

Tripathi, B., Thakur, K., Nayak, R., Sastry, C., and Pradhan, J. (2016). Ntru cryptosystem with companion matrix. *A matrix formulation for NTRU cryptosystem*, pages 1–5.

Wijayanti, I. E. et al. (2023). The meet-in-the-middle attack on the matrix ntru cryptosystem. In *2023 IEEE International Conference on Cryptography, Informatics, and Cybersecurity (ICoCICs)*, pages 149–153. IEEE.

Zhao, Z. and Ye, Q. (2023). Revisiting lower dimension lattice attacks on ntru. *Information Processing Letters*, 181:106353.