Malaysian Journal of Computing and Applied Mathematics 2020, Vol 3(1): 1-10
©Universiti Sultan Zainal Abidin
(Online)

MALAYSIAN JOURNAL OF COMPUTING AND APPLIED MATHEMATICS

# Two new Quantum Attack Algorithms against NTRU_pke # KA_NTRU # & # PA_NTRU #

***Laaji El Hassane[a], Azizi Abdelmalek[b] and Azzouak Siham[c]***

[a,b]ACSA Laboratory, Mohammed First University, Oujda, Morocco
[c]ACSA Laboratory, Sidi Mahammed Ben Abdellah University, Fes, Morocco

[*]Corresponding author: e.laaji@ump.ac.ma

*Received: 18/12/2019, Accepted: 22/04/2020*
http://dx.doi.org/10.37231/myjcam.2020.3.1.37

## Abstract

The NTRUencrypt authors submitted four versions to National Institute of Standardization (NIST) competition since 2016 and it is still in process for the second round. The NTRU_pke release is one of them, it is defined in the ring $\mathbf{R_q} = Z_q[X] / (X^n-1)$; and the private keys and the plaintext are codified in trinary polynomial, that means all their coefficients are in {-1,0,1}. Our two quantum attacks algorithms KA_NTRU and PA_NTRU on NTRU_pke implementation, inspired from Grover's Algorithm, targeted respectively to find Private Keys and Plaintext. For testing the implementation attacks, we create a NTRU_pke test version named NTRU Attacks. In the general case, the quantum algorithms can break a system of dimension n in $2^{n/2}$ time.

**Keywords**: NTRU, NewHope, Lattice- Based-Cryptography, Post Quantum cryptography, Grover's Algorithm.

## INTRODUCTION

The big concern of the cryptographic community now, is to build new cryptosystem Post-Quantum cryptosystems able to resist against these attacks, because the near future the cryptosystems used right now like RSA, ECDH, El Gamal will be easily broken the quantum computer using quantum algorithms like Grover's algorithm, Shor's algorithm or others "Quantum computer using a quantum algorithm can solve a problem of dimension n in $2^{n/2}$ time " (Vredendaal, 2014).

This is why the National Institute of Standards and Technology (NIST) launch a competition since 2016 and it is still in process for choosing one or more post-quantum cryptosystems (Chen et al., 2016). The authors of NTRUencrypt submitted four schemes NTRU PKE (Public Key Encryption) NTRU KEM (Key Exchange Mechanism) ss-NTRU_pke and ss-NTRU_kem. Our focus in this work on NTRU_pke as described by the authors, it is based on the original NTRU with new parameters (Chen et al., 2011) that achieves CCA-2 security via NAEP transformation (Vredendaal, 2016). It is amongst the most important Lattice-Based submissions to the NIST competition (Chen et al., 2011). It uses the hardest problems on points lattices in $R^n$ like SVP (Short Vector Problem) and CVP (closet Vector Problem). the NTRU assumption is defined by:" Having $h = g/f$ it is hard to find $f$ and $g$. The NTRU assumption can be reduced to the *uSVP* for the NTRU lattices". The NTRU has survived for 20 years of cryptanalysis, and judged to be able to resist against quantum attacks, and deemed able to take over and take the place of classical cryptography.

### A. Related Works

The most attacks used to check the robustness of Lattice-Based Cryptosystems are the Reduction algorithms attack and Meet-In-the-Middle (MIM) attack developed by (Oldgz) (Vredendaal, 2016), or the hybrid attacks that use the mixture of the reduction attack and MIM attack (Hoffstein et al., 2017). Others types attacks exist like Cold Boot attack developed by Martin Albrecht et.al. (2018) and another aspect of security problems named Side-channel attack developed by Kocher (Liu et al., 2017).

### B. Our Work

In this paper, we present two attacks algorithms on NTRU_pke implementation level, named KA_NTRU and PA_NTRU targeted respectively to find the private keys and plaintext. The domain of NTRU_pke is the ring $\mathbf{R_q} = Zq[X]/(X^n - 1)$, and the private keys are trinary polynomials generated according to Uniform Distribution, that means that all their coefficients are in {-1,0,1} and the plaintext also codified in a trinary polynomial. For testing our attacks, we re-implemented new release NTRU Attacks which include the attacks implementation functions. For example, we tested the attacks with small polynomial dimensions, we obtained the bellow result: when the polynomial dimension is equal to 17 the system was broken in about 1,5 minutes for KA_NTRU and in about 30 seconds for PA _NTRU. All our test is performed in computer PC Toshiba i7. And we note that our work is inspired from Grover's algorithm and Scott Fluhrer work (Fluhrer, 2015). The NTRU_pke scheme chooses all the trinary polynomials according to a Uniform Distribution (UFD), indeed, an attacker can use the Quantum Computer and our attacks to break the system by sampling simultaneously the coefficients of the trinary polynomials to find respectively the private keys and plaintext as we are going to describe in section 5.

### C. Outline

The remainder of our work is organized as follows: Section 1: this introduction; Section 2: preliminaries: We start by noting and defining the terminology used in this paper. Then we recall the necessary knowledge of the Lattice-Based-Cryptography, a brief description of sampling methods Uniform distribution (UFD) and we also give a brief description of quantum algorithms. Section 3: We recall the related work's description of the principal attacks on NTRU cryptosystem. Section 4: We give an overview of the original NTRU scheme and NTRU_pke release submitted to NIST competition; Section 5: We present our two attacks algorithms KA NTRU and PA_NTRU against NTRU_pke to recover respectively the private keys and the plaintext. Section 6: In this section, we present the principal specification of our attacks, implementations and our new release implementation NTRU Attacks. Section 7: We draw conclusions and some topics that may provide interesting research area.

## PRELIMINARIES

### *A. Notations*

In the remainder of this paper, we use the following notations: LBC for Lattice-Based-Cryptography (Hoffstein et al., 1998); $L_{(B)}$ Lattice of $\mathbf{R}^n$ with base $\mathbf{B} = (\vec{e}_1,... \vec{e}, _n)$ with the form $\mathbf{L}_{(B)} = \{ \vec{v} \in R^n, (a_1,...,a_n) \in Z^n$ et $\vec{v} = a_1\vec{e}_1 + ... + a_n \vec{e}_n\}$ (Chen et al., 2011); $\mathbf{R} = \mathbf{Z}[X]/(X^n - 1)$ the ring polynomial with coefficient in $Z^n$; and $R_q$ the quotient ring polynomial with coefficients in [0 , q]; $||\vec{v}||$ the norm of vector $\vec{v}$ ; $||\vec{v}|| = max|v_i|$ the low-norm ; (a,b,..) uppercase the elements of R; ( a,b,...) lowercase the coefficients of elements of **R**; the $< \vec{v}. \vec{u} >$ inert product of $\vec{v}$ and $\vec{u}$ ; we refer to sampUFD(seed) the polynomial sampled according to Uniform Distribution . We refer to *Trinary Polynomials*
$T(d_1,d_2) = a(x) \in R$ with the polynomial $a(x)$ has $d_1$ coefficients equal to *1*; $d_2$ coefficients equal to *−1* and the rest of coefficients equal to *0*.

We refer to KA_NTRU the Private Key Attack algorithm, we refer to PA_NTRU the Plaintext Attack Algorithm and the we refer to NTRU_Attacks Our test implementation release of the original cryptosystem NTRU_pke which takes into consideration our idea and keeps all the parameters (Hoffstein et al., 2017).

### *B. Lattices Problems*

LBC like all cryptosystems it is based on mathematical concepts and theories to encrypt and decrypt as well as to demonstrate the complexity and the difficulty of breaking those cryptographic systems by posing problems that are difficult to solve. The principal Lattice problems are SVP and CVP and their approximate versions. *The Shortest Vector Problem (SVP)* (Hoffstein et al., 1998): Finding (SVP) in Lattice $L_{(B)}$ is finding a nonzero vector that minimizes the Euclidean norm. Formally the problem SVP is to find a non-zero vector:

$$\tilde{\mathbf{v}} \in \mathbf{L}_{(\mathbf{B})} forall \tilde{\mathbf{x}} \in \mathbf{L}_{(\mathbf{B})} we have \|\tilde{\mathbf{v}}\| \leq \|\tilde{\mathbf{x}}\| \qquad (1)$$

*The Closest Vector Problem (CVP)*: Given the Lattice $L_{(B)}$ , and a vector $\tilde{\mathbf{w}} \in \mathbf{R^m}$ to find a vector $\tilde{\mathbf{v}} \in \mathbf{L}_{(\mathbf{B})}$ "Closest" to $\vec{w}$ , is to find a vector$\vec{v} \in L_{(B)}$ that minimizes the Euclidean norm $\|\vec{w} - \vec{v}\|$ where:

$$\|\vec{w} - \vec{v}\| = \mathbf{min}\{\|\vec{w} - \vec{v}\| / \tilde{\mathbf{v}} \in \ \mathbf{L}_{(\mathbf{B})}\} \qquad (2)$$

### *C. LBC Cryptosystems*

There are different cryptosystems based on LBC the first construction was created by AJTAI-DWORK[10] in 1996 but in 2001 Nguyen and Stern broke this cryptosystem by solving the SVP and CVP. Also, Goldreich Goldwasser and Halevi (GGH) in 1997 are created, their cryptosystem, but also in 2001 Nguyen and Stern broke this cryptosystem by solving the SVP and CVP (Hartmann, 2015) and others cryptosystems exist like Learning With Errors (LWE) created by Oded Regev in 2005 but it is still of research stage right now. Only NTRU( standardized by IEEE P1363.1 on April 2011) is judged able to resist against all attacks conjectured (Micciancio & Regev, 2009).

### *D. Description of Uniform Distribution law (UFD) (Fleury & Dupont, 1986).*

*Definition*: Let X be a random variable on ($\omega$,$P$($\omega$),$P$) on image space: $X(\omega) = \{x_1,x_2,...,x_n\}$ of law $p$. We say that X is uniform variable or X follows a Uniform law:

$$if \quad \forall\, i \in \{1, 2, ..., n\} \quad p(x_i) = \frac{1}{n} \qquad (3)$$

*Properity*: $\mathbf{X}$ being a uniform variable of image space $X(\omega) = \{x_1, x_2, ..., x_n\}$ we have Esperance $E(X) = \frac{1}{n}\sum_{i=1}^{n} x_i$ and the variance $Var(X) = \frac{1}{n}\sum_{i=1}^{n} x_i^2 - \frac{1}{n^2}(\sum_{i=1}^{n} x_i)^2$

### *E. Quantum Algorithm*

*Quantum algorithm Definition*: The information in a quantum computer is in a superposition of states, therefore, the quantum algorithm performs simultaneously multiple treatments. The majority of quantum algorithms (like Shor's and Grover's algorithms) is defined by three steps consecutive:

1) The creation of a configuration in which the amplitude of the system is in any of the $2^n$ basic states of the system are equals;
2) The Walsh-Hadamard transformation operations (is an example of a generalized class of a Fourier transform); transform *N* qubits initialized with ($|0>$) into a superposition of all $2^n$ orthogonal states expressed in the base ($|0> |1>$) with equal weighting (Williams, 2011).
3) The selective rotation of different states. When a classical algorithm finds an element in a list of *N* randomly selected elements with a probability of N/2, the quantum algorithm of Grover does so with a probability of $O(N^{1/2})$.

### *F. Grover's Algorithm Principle*

1) Problem: Search some solutions in an unstructured database ;
2) Classical: Essential problem N entries —¿ in average *N/2* tests;
3) Quantum Grover' algorithm: $O(N^{1/2})$ In general can speed up all classical algorithms using a search heuristic.
4) Formulation of the problem: *N* elements indexed from *0* to *N-1*,$N = 2^n|U>x$ search register elements repertories via their index;

The search problem admits M solutions (For more detail the reader can see (Williams, 2011)).

## RELATED WORKS

Many cryptanalysis works are performed, their principal goal was to check the robustness of the Lattices cryptosystems. Others works exist for cryptosystems analysis like cold boot attack and side-channel attack as we are going to describe them in this section.
The best tools used to prove the security and the efficiency of an LBC is Hybrid Attack combined with Lattice reduction attack with LLL(Lenstra, Lenstra, Lovasz) and BKZ2 (Blockwise-Korkine-Zolotarev) and Meet-in-The-Middle attack(MIM) developed by (Odlgzko) (Vredendaal, 2016).
In the NIST-2018 benchmarking realized by Albrecht et al. the authors claim that NTRUencrypt (Albrecht et al., 2018) warrant 198-bit security level against quantum attack and more than 256 bits of classical security.

### *A. Lattice Reduction*

The reduction of Lattice Basis is very important, and the SVP or CVP will be easier to solve with a reduced Basis. The most reduction algorithms used to solve those Lattices problems are the LLL algorithm, BKZ.
*LLL : Lestro-Lenstra-Lovasz Algorithm:*The role of the LLL algorithm is to transform a Bad Basis $B = (\vec{e}_1,..., \vec{e}_n)$ to a Good Basis $G = (\vec{u}_1,..., \vec{u}_n)$ of a Lattice $\mathbf{L_{(B)}} \subset \mathbf{R^n}$ . It allows approximating the smallest vector in polynomial times. This can provide a partially satisfactory answer to the problems Lattices SVP and CVP on which the security of the Lattices cryptosystems is based on (Peikert, 2017).
Definition: The basis of a Lattice is said to be reduced by the LLL algorithm if it is obtained by the Gram-Schmidt algorithm and satisfies the following two properties:

$$\forall 1 \leq i \leq n, j < i \quad |\lambda_{i,j}| \leq \frac{1}{2} \quad |\lambda_{i,j}| = \frac{|\vec{e_i}.\vec{u_j}|}{\|\vec{u_j}\|^2} \qquad (4)$$

$$\forall 1 \leq i \leq n \quad \delta.\|\vec{u_i}\|^2 \leq \|\lambda_{i+1j}\vec{u_i} + \vec{u_{i+1}}\|^2 \quad \delta \in [\frac{1}{4}, 1] \quad (5)$$

*BKZ Algorithm:* The main goal of BKZ is to output a BKZreduced basis $G_{bkz} = (\vec{u}_1,..., \vec{u}_n)$ with block size $B_{[j,min(j+\beta-1,n)]}$ , with $\beta \geq 2$ and factor $\varepsilon > 0$ from reduced LLL-reduced basis. $B_{[j,min(j+\beta-1,n)]}$ is obtained by iteration reduction for $j = 1$ *to* $n$ , until finding the block with the SVP in the projected lattice. For more detail of this algorithm, see the Yuanmi Chen et.al work (Chen & Nguyen, 2011).

### *B. Meet-in-the-Middle Attack:*

The goal of a general Meet-In-the-Middle attack, is to find specific elements *x,x'* in a search space of which it is known that $f_1(x) = f_2(x')$ search *x, x'* values ; the unique solution is called the golden collision; Having the private key decomposed into:
$f = f + f_2$ with $f_1(deg = \frac{n}{2} - 1)$ and $f^2(deg \in [\frac{n}{2}, n-1])$

$$h = (f_1 + f_2)^{-1} * g \Rightarrow f_1 * h = g - f_2 * h \quad (6)$$

All rotations of the keys f and g will be a solution to this equation. Search with (almost) equal number of ones; For more description of this attack see the paper Choosing Parameters for NTRUencrypt (Hoffstein, 2017).

### *C. Cold Boot Attack:*

This attack is firstly described by Albrecht (2018). This attack consists of the fact that bits in RAM save their value for some time after power is cut, to explore this and retain a *p0 = 1%* bit-flip, the memory must have a cold temperature at about *(-50 C°)* to retain bits for *10mn* after computer power down. Halderman et al. also noted that " bit-flip rates as low as p0 = 0.17% are possible when liquid nitrogen is used for cooling".

### *D. Side-channel Attack:*

The work realized by Zhe Liu et al. (2017) and the work realized by El Mrabet (2010), explain the side-channel attack allows the attacker to find the private keys, with the measurement of the times performing and energy consumption, or current intensity by intercepting their variations. To prevent this attack, it is possible to add fictive operations for redressing (correlation of power) consumption. The main idea is to find witch fictive operation to add in each cryptographic function

## NTRUENCRYPT

### *A. Overview of NTRU*

It was created in 1996 by the three mathematicians J. Hofstein J. Pipher and J. H. Silverman and published in 1998. It is the first cryptosystem that is completely LBC. Its domain of computation is the ring of the polynomials *mathbf* $R_q = Zq[X]/(X^N - 1)$ where N is a prime number.

The major advantages of using a ring structure are a relatively smaller key size and faster speed that can be achieved. NTRU consists of two protocols the NTRUEncrypt Protocol and the NTRUSign Signature Protocol. NTRU was considered reliable, by the IEEE P1363.1 standard and in April 2011 NTRUEncrypt was accepted in the X9.98 standard. In terms of security, NTRU was resisted for 20 years of cryptanalysis. Its low memory consumption allows us to use it for applications such as mobile devices or smart cards.

### *B. NTRU schemes submitted to the NIST competition*

The NTRUencrypt team (Chen et al., 2011) submitted four schemes:

1) NTRU_pke and NTRU kem with the sequence parameters is {$N = 443, q = 2048, p = 3, d = 143$} and {$N = 743, q = 2048, p = 3, d = 247$}, for those schemes, all the polynomials are sampled according to a Uniform Distribution (UFD) with the polynomial coefficients are in {-1,0,1}.
2) ss-NTRU_pke and ss-NTRU _kem releases with the sequence parameter {$N = 1024, q = 1073750017, p = 2, \sigma = 724$} where $\sigma$ is the standard deviation, and for those schemes, all the polynomials are sampled according to Discrete Gaussian distribution (DGD) or Deterministic Discrete Gaussian Distribution (DDGD).

### *C. NTRU_pke Description*

Our work focused and concern only the NTRU_pke schemes with the parameters set warrant quantum security level 84 bits and 159 bits claimed by NIST. To explain more clearly our purpose and presenting easily our work, we are going to describe the principal cryptographic functions of NTRU_pke cryptosystem. For more detail, the reader can see the authors¨ original document available in the NIST competition web site.

*a. Parameters:* : As described in Chen et al. (2011) The domain of NTRU_pke is the quotient Ring with: $\mathbf{R} = Z[X]/(X^N - 1)$ and $\mathbf{R_q} = Zq[X]/(X^N - 1)$. These schemes require a function that generates the trinary polynomials according to a Uniform Distribution (UFD) with *seed* sampled with Salsa20 based pseudo-random number, the authors state that Salsa20 is 5 times faster than AES. We note that NTRU_pke takes the private key $F$ in the form $F = p * f + 1$ [16]. This form allows us to avoid to calculate the inverse of $f \bmod p$ because $F = p * f + 1 \pmod p = 1$. For the actual parameters the reader can see Hoffstein et al., (2017).

### *D. NTRU_pke Algorithms*

*a)* ***Algorithm 1: Keys Generation****:* .

**Input :** the sequence parameters $\{N, q, p, d\}$ and *seed*.

1) $f, g \leftarrow sampUFD(seed)$; as Trinary Polynomials;
2) $F \leftarrow p*f+1)$;
3) $F_q \leftarrow inverse(F) \bmod q$;
4) $h \leftarrow g * F_q \pmod q$;

**Output:** the public key $h$ and the private key $F$ and *seed*.

*b)* ***Algorithm 2: Encryption*** *:* .

**Input:** The public key $h$, the message with its length *msg,len*, and the *seed*.

1) $M \leftarrow choosing \in \{-1, 0, 1\}$ ;
2) $r \leftarrow sampUD(rseed)$ ;
3) $e \leftarrow sampUD(eseed)$ ;
4) $c \leftarrow p * r * h + m \pmod q$ ;

**Output:** The ciphertext c.

*c)* ***Algorithm 3: Decryption*** *:* .

**Input:** The private key $F$ and the ciphertext $c$.

1) $M \leftarrow F * c \pmod q$ ;
2) $M \leftarrow lift\ M \in \{\frac{-q}{2}, \frac{q}{2}\}$ ;
3) $M \leftarrow M \pmod p$

**Output:** the message $M$.

## OUR CONTRIBUTION

As we described before in the abstract and the introduction sections, our quantum attacks algorithms KA_NTRU and KA_NTRU aims to break the NTRU_pke cryptosystem by finding respectively, the private key and the plaintext. An attacker with a quantum computer can use our algorithms by sampling simultaneously the polynomial coefficients respectively, of *f* and of *r* until breaking the system. We describe both algorithms as follow:

### *A. Our Attacks Algorithms description*

*a)* ***Algorithm 4: KA_NTRU*** *: .*
Input: the public key *h*, the modulus *q*, and *n*

1) *Repeat :*
2) $f' \leftarrow sampUD(seed)$ ;
3) $F' \leftarrow p*f'+1)$;
4) $g' \leftarrow F' * h \pmod q$;
5) $if \quad g'_i = q-1 \rightarrow g'_i = -1$;
6) $Until : g' \in \{-1, 0, 1\}$;

**Output :** $f = f' \quad and \quad g = g'$.

**Comment**: For the algorithm.4 KA _NTRU the attacker samples simultaneously the polynomial coefficients of *f'* (line 2) and computes *g' = F* ∗ *h* (mod *q*) (line 4) until the coefficients of *g'* are in {-1,0,1} (line 6) then he finds the private keys $f = f^0$ and *g = g''* . We note that if a coefficient polynomial equal to *q − 1* we must replace it by (-1) as in line 5.

*b)* ***Algorithm 5: PA_NTRU*** *: .*
**Input**: the public key *c*, the modulus *q*, and *n*.

1) *Repeat :*
2) $r \leftarrow sampUFD(seed)$ ;
3) $c' \leftarrow p * r * h \pmod q$;
4) $M' \leftarrow c - c' \pmod q$;
5) $if \quad M'_i = q-1 \rightarrow M'_i = -1$;
6) $Until : M' \in \{-1, 0, 1\}$

**Output :** The plaintext $M = M'$.

**Comment**: the same for the algorithm.5 PA NTRU, the attacker samples simultaneously the polynomial coefficients of *r*(line 2) and computes a polynomial *c'= p.r* ∗ *h* (mod *q*) as in line 4 and computes a *M'= c − c'*(mod *q*) (line 5) until the coefficients of polynomials *M'* are in {-1,0,1} as in line 6, then he finds the plaintext *M = M'*. We note that if a coefficient polynomial equal to *q − 1* we must replace it by (-1) as in line 5.

### *B. Result*

In this subsection, we present the result obtained by using our NTRU_attacks release implementation. Unfortunately, we haven't a quantum computer, then we are going to try our algorithms in a classical computer with small parameters just for having an idea of the speed performance of both attack algorithms. The algorithms generate trinary polynomials with parameters *d* the number of coefficients in trinary polynomial equal to (1) and equal to (-1) chosen respectively in {2,3,4,5,6} , the dimension *n* chosen respectively in {7,11,13,17,19} and we chose also small modulus *q = 127*. In the table1 we give the cost of the KA_NTRU attack to find the private keys and in the table2 we give the cost of PA_NTRU attack to find the plaintext.

TABLE 1. Cost of KA _NTRU attack against NTRU _PKE

| Attacks | n:7 | n:11 | n:13 | n:17 | n:19 |
|---|---|---|---|---|---|
| Times(ms) | 5 | $1{,}5\ .10^3$ | $3.10^3$ | $29.10^3$ | $51.10^3$ |
| Operations | 54 | $35.10^3$ | $65.10^3$ | $662.10^3$ | $10^6$ |

TABLE 2. Cost of PA _NTRU attack against NTRU _PKE

| Attacks | n:7 | n:11 | n:13 | n:17 | n:19 |
|---|---|---|---|---|---|
| Times(ms) | 46 | 69 | $4{,}7.10^3$ | $98.10^3$ | $3. .10^6$ |
| Operations | 346 | 690 | $95.10^3$ | $2{,}3.10^6$ | $5{,}6.10^6$ |

## IMPLEMENTATIONS

Our implementation of KA_NTRU algorithm and PA_NTRU algorithm and ours NTRU Attacks were implemented on C++ and performed in PC-TOSHIBA –Satellite, Processor Intel, Core™i7 -2630QM CPU, 2GHz, RAM 8 GO, under environment Windows 7-32 bits and
Dev-C++ 4.9.9.2.
For NTRU_Attacks we keep all the parameters and the reference implementation functions of NTRU_pke, with few modifications. , specially we re-used the reference implementation of " *void trinary poly gen(rr,d,n);* " function for sampUFD as described in the algorithms.

For the polynomials multiplication we not used the **Karashuba** algorithm as in NTRU_pke release submitted to NIST, we used our own polynomials multiplication *XKwarizm* algorithm in the ring $\mathbf{R_q} = Zq[X]/(X^N - 1)$, in our paper under title New Multiplication Algorithm +600% faster than NTT algorithm when applied to NTRU1024 submitted to Journal of Cryptographic Engineering on mars-2019 "unpublished" (Azizi et al, (2019), the reader can see the *XKwarizm* algorithm in [Appendix A]. For testing the KA_NTRU and PA_NTRU attacks algorithms, we integrate their implementation functions in NTRU Attacks as follow:

- The program performs the keys generation function and encryption function and checks the decryption function;
- The *KA NTRU(.)* function receives the public key *h* and returns the private keys *f, g*;
- The *PA NTRU(.)* function receives the ciphertext *c* and returns the plain text *m*.
-

Note: All source implementations of those attacks are available into the website at : Sources Code or the reader can contact us at: *e.laaji@ump.ac.ma*.

## CONCLUSION

The evolution in quantum computer science is very fast, whereas the choice of PQcryptosystem for standardization is in the NIST process, to prepare for a smooth migration of classical cryptosystems to post-quantum cryptosystems. We are not sure that some cryptosystems submitted will be resistant against quantum attacks. It is possible to modeling our algorithm attacks and apply them to some others Lattice-based submissions to NIST inspired from original NTRU schemes like NTRUprime (Bernstein et al, 2016) or R-LWE (Ring Learning With Error) schemes like NewHope (Alkim, 2015). For example, to apply **KA_NTRU** Attack to NTRUprime, the attacker can generate simultaneously a polynomial $f$ and computing $3 * h * f = g \pmod q$ until the coefficients of $g$ are in {-1,0,1}.

The same for New Hope implementation, the hardness assumption is defined by: " Having $b = a * s + e$, it is hard to find $s$". Therefore an attacker can sample a polynomial $s$ and computes $b - as = e \pmod q$ until the polynomial coefficients of $e$ are in{0,1} or in{0,1,2,3 } according to the choice of error area and then return the private key $s$. A lot of researchers in this cryptographic domain said that " only Quantum Cryptography will resist against Quantum Attacks ", but we continue our work on the cryptanalysis and improvement of the post-quantum cryptosystems and we hope to contribute with all cryptographic community to build the strong cryptosystems to save the private life of the person by learning from the best actual practices and innovate new methods.

## References

Albrecht M R et al. (2018). Cold boot attacks on ring and module LWE Keys under the NTT, Royal Holloway, University of London.

Albrecht M R et al. (2018). Estimate All the {LWE, NTRU} Schemes!. In: Catalano D., De Prisco R. (eds) Security and Cryptography for Networks. SCN 2018. Lecture Notes in Computer Science, vol 11035. Springer, Cham

Alkim E. (2015). Post-quantum key exchange-New Hope. Department of Mathematics, Ege University, Turkey-LeoDucas.

Azizi A et al. (2019). New multiplication algorithm +600 % faster than NTT algorithm when applied to NTRU1024–Xkhwarizm. Mohammed First University Morocco.

Bernstein D J et al. (2016). NTRU Prime, Department of Computer Science- University of Illinois at Chicago, Chicago, USA.

Chen L et al. (2016). NISTIR 8105 - Report on post-quantum cryptography, Tone.

Chen C et al. (2011). NIST PQ submission: NTRUencrypt a lattice-based encryption algorithm", Brown University and Onboard security Wilmington USA.

Chen Y, Nguyen P Q. (2011) BKZ 2.0: Better Lattice Security Estimates. In: Lee D.H., Wang X. (eds) Advances in Cryptology – ASIACRYPT 2011. ASIACRYPT 2011. Lecture Notes in Computer Science, vol 7073. Springer, Berlin, Heidelberg

El Mrabet N. (2010). Attaques par canaux caches, Universite de Caen, France. Available at https://www.emse.fr/~nadia.el-mrabet/Presentation/Cours5_SCA.pdf

Liu Z et al. (2017). FourQ on embedded devices with strong countermeasures against side-channel attacks. University of Waterloo, Canada.

Fleury J –P, Dupont R. (1986). Probabilités, statistiques, programmation linéaire: exercices avec solutions. Paris: Vuibert

Fluhrer S. (2015). Quantum cryptanalysis of NTRU. Cisco systems, Available at https://eprint.iacr.org/2015/676.pdf.

Hartmann M. (2015). Ajtai-Dwork cryptosystem and other cryptosystems based on lattices. Universite de Zurich.

Hoffstein J et al. (1998). An Introduction to Mathematical and Cryptography. Springer.

Hoffstein J et al. (2017). Choosing Parameters for NTRUEncrypt. In: Handschuh H. (eds) Topics in Cryptology – CT-RSA 2017. CT-RSA 2017. Lecture Notes in Computer Science, vol 10159. Springer, Cham.

Micciancio D, Regev O. (2009) Lattice-based Cryptography. In: Bernstein D.J., Buchmann J., Dahmen E. (eds) Post-Quantum Cryptography. Springer, Berlin, Heidelberg

Peikert C. (2014). Lattice cryptography for the Internet. Available at https://web.eecs.umich.edu/~cpeikert/pubs/suite.pdf

Vredendaal C V. (2014). Available at http:photon.physnet.unihamburg.defileadminuser uploadILP.

Vredendaal C V. (2016). Reduced memory meet-in-the-middle attack against the NTRU private key. Available at https://eprint.iacr.org/2016/177.pdf

Wiliams P W. (2011). Explorations in Quantum Computing, 2nd edition, Springer.

## Appendix A

Our own polynomial multiplication *XKwarizm* algorithm in the ring $\mathbf{R_q} = Zq[X]/(X^N - 1)$. It allows us to multiply two polynomials *f and g* in $\mathbf{R_q}$ = and then return a polynomial *h* directly reduced in $\mathbf{R_q}$ =.

***a) Algorithm : Xkhwarizm :*** **.**
**Input**: Polynomials *f and g* , with their degrees less than
*(n)*, and *modulus q*; Output: the polynomial *h*.
1. Function *Xkhawarism (f,g) :*
2. . For : int $i = 0$ to $n - 1$ do :
3. ... If $f_i ! = 0$ then 4. ..... For : int $j = 0$ to $n - 1$ do :
5. ..... If $g_j ! = 0$ then :
6. ..... *If*$((i + j) < n)$ then : $h_{i+j} \leftarrow h_{i+j} + f_i.g_j$;
7. ..... else : $h_{i+j-n} \leftarrow h_{i+j-n} + f_i.g_j$;
8.....endif(line6)
9.....endif(line5)
10.....endfor(line4)
11.... endif(line3)
12. .endfor (line2)
13. For : integer $i = 0$ to n do: $h_i \leftarrow h_i \pmod q$;
14. endfunction.
**Output**: The result polynomial of product *h*;

***b) Algorithm description****::* In line2, the algorithm opens the first loops, it performs all instructions from index $i = 0$ *to* $i = n-1$.Then in line3, It checks, if the polynomial coefficient $f_i$ equal to zero then return to the line2, else we perform the line3, it opens the second loops, also in line5, if the polynomial coefficient $g_j$ is equal to zero then return to the line4. In line6, if the sum of degrees $(i+j) < n$, the algorithm computes the polynomial coefficient$h_{i+j}$ by adding its value to the product of $f_i$ and $g_j$, else *if*$(i+j) > n$, it computes $h_{i+j-n}$, in line7, by adding its value to the product of$f_i$and $g_j$, and finally the algorithm reduces all coefficients of the polynomial h mod q, and out put it.