NTRU et ses variantes, sécurité et applications

Abderrahmane Nitaj
Laboratoire de Mathématiques Nicolas Oresme
Université de Caen, France
nitaj@math.unicaen.fr
http://www.math.unicaen.fr/~nitaj

Résumé

Depuis son invention entre 1996 et 1998 par Jeffrey Hoffstein, Jill Pipher et Joseph H. Silverman, le cryptosystème NTRU a connu plusieurs améliorations et développements. Ce cryptosystème se distingue par la rapidité des processus de chiffrement et de déchiffrement et par sa résistance à la cryptanalyse quantique, au contraire des cryptosystèmes classiques qui sont RSA, ECC et EL-Gamal. Le but de cet article est de présenter les derniers développements du cryptosystème NTRU, discuter les problèmes difficiles qui garantissent sa sécurité et introduire certaines de ses variantes.

1 Introduction

La cryptographie a été inventée en même temps que l'écriture. Dans l'ancienne Egypte par exemple, les scriptes utilisaient des symboles hieroglyphique que eux seuls pouvaient déchiffrer. Du temps des romains, la cryptographie était basées sur une rotation de l'alphabet. Cette méthode est connue sous le nom de *chiffrement de César*. Ce chiffrement fût cassé par les mathématiciens arabes du temps d'Al Kindi (801-873). Pour chiffrer et déchiffrer avec ce type de cryptographie, il faut connaître la clé secrète. Ce procédé cryptographique est appelé *cryptographie à clé secrète* ou encore la *cryptographie symétrique*. Les cryptosystèmes les plus connus dans la cryptographie symétrique sont Rijndael de AES (J. Daemen et V. Rijmen, 2002), IDEA (X. Lai et J. Massey, 1990) et 3DES (W. Tuchman, 2005). Ce type de cryptographie peut présenter évidement un danger car il faudrait échanger les clés. En 1976, une cryptographie différente a été inventée par W. Diffie et M. Hellman [5]. Ce fût le début de la cryptographie moderne, connu sous le nom de *cryptographie à clé publique* ou encore *cryptographie asymétrique*. Les cryptosystèmes les plus connus dans la cryptographie asymétrique sont :

- Le cryptosystème RSA, inventé en 1977 par R. Rivest, A. Shamir et L. Adleman [18]. La sécurité de RSA est basée sur la difficulté de factoriser les grands nombres entiers et la difficulté d'extraire la racine n-ième d'un entier modulo un grand nombre entier dont la factorisation est inconnue.
- Le cryptosystème El Gamal, inventé par T. El Gamal [6] en 1985. La sécurité de ce cryptosystème est basée sur le problème du logarithme discret : *Etant donné deux nombres entiers g et y et un nombre premier p, déterminer un entier x tel que $g^x \equiv y \pmod{p}$.*
- Le cryptosystème ECC, the Elliptic Curve Cryptography, inventé indépendement en 1985 par N. Koblitz [10] et V.S. Miller [14]. La sécurité de ce cryptosystème est basée sur le problème du logarithme discret elliptique : *Etant donné deux points P et Q d'une courbe elliptique E, déterminer un entier n tel que $nP = Q$.*
- Le cryptosystème NTRU, inventé entre 1996 et 1998 par J.H. Silverman, J. Hoffstein et J. Pipher [8]. La sécurité de NTRU est basée sur le problème du plus court vecteur non nul d'un réseau (SVP).

Pour une comparaison des performances de ces cryptosystèmes, on peut consulter page de comparaison de NTRU Inc. [17].

Notons qu'il est possible aussi d'utiliser les deux types de cryptographie. Ce troisième type de cryptographie s'appelle *la cryptographie hybride* : Dans un système à base de cryptographie à clé secrète, on peut échanger la clé secrète commune à l'aide d'un système à base de cryptographie à clé publique et ensuite utiliser le système à base de cryptographie à clé secrète.

Le reste de cet article est organisé comme ce qui suit. Dans la partie 2, on présente le cryptosystème NTRU et son fonctionnement. Dans la partie 3, on présente les base de la sécurité de NTRU, notamment les problèmes SVP et CVP ainsi que quelques attaques sur NTRU. Dans la partie 4, on présente quelques variantes et généralisations de NTRU. Enfin, dans la partie 5, on présente quelques avantages de NTRU ainsi que quelques unes de ses applications.

2 Le cryptosystème NTRU

Le cryptosystème NTRU à été présenté en 1996 dans la rump session de Crypto 96 et publié en 1998 [8]. Le domaine de calculs de NTRU est l'anneau des polynômes $\mathbb{Z}[X]/(X^N - 1)$ et utilise des réductions modulo deux nombres (ou polynômes) premiers entre eux p et q.

NTRU se compose de deux protocoles. Le protocole de chiffrement-déchiffrement NTRUEncrypt et le protocole de signature NTRUSign. NTRU est aujourd'hui considéré

comme fiable par le standard IEEE P1363.1 [9].

2.1 Définitions

Le cryptosystème NTRU utilise plusieurs sortes de paramètres, en particulier les paramètres N, p et q.

Soit N un nombre entier. Les opérations de NTRU ont pour domaine l'anneau des polynômes suivant :

$$\mathcal{P} = \mathbb{Z}[X]/\left(X^N - 1\right).$$

Ainsi, les éléments f de $\mathcal{P}$ peuvent être représentés sous la forme

$$f = (f_0, f_1, \cdots, f_{N-1}) = \sum_{i=0}^{N-1} f_i X^i.$$

L'addition de deux polynômes $f, g \in \mathcal{P}$ se fait de façon naturelle terme à terme de même degrés, alors que la multiplication se fait par un produit de convolution noté ici $*$. Si $f * g = h$ avec $f = \sum_{i=0}^{N-1} f_i X^i$ et $g = \sum_{i=0}^{N-1} g_i X^i$, alors $h = \sum_{i=0}^{N-1} h_i X^i$ avec pour tout $0 \leq k \leq N-1$,

$$h_k = \sum_{i+j \equiv k \mod N} f_i g_j = \sum_{i=0}^{k} f_i g_{k-i} + \sum_{i=k+1}^{N-1} f_i g_{N+k-i}.$$

La table ci-dessous illustre le produit $h = f * g$ avec $N = 7$, $f = \sum_{i=0}^{6} f_i X^i$ et $g = \sum_{i=0}^{6} g_i X^i$.

	1	X	X^2	X^3	X^4	X^5	X^6
	f_0g_0	f_0g_1	f_0g_2	f_0g_3	f_0g_4	f_0g_5	f_0g_6
$+$	f_1g_6	f_1g_0	f_1g_1	f_1g_2	f_1g_3	f_1g_4	f_1g_5
$+$	f_2g_5	f_2g_6	f_2g_0	f_2g_1	f_2g_2	f_2g_3	f_2g_4
$+$	f_3g_4	f_3g_5	f_3g_6	f_3g_0	f_3g_1	f_3g_2	f_3g_3
$+$	f_4g_3	f_4g_4	f_4g_5	f_4g_6	f_4g_0	f_4g_1	f_4g_2
$+$	f_5g_2	f_5g_3	f_5g_4	f_5g_5	f_5g_6	f_5g_0	f_5g_1
$+$	f_6g_1	f_6g_2	f_6g_3	f_6g_4	f_6g_5	f_6g_6	f_6g_0
$h =$	h_0	h_1	h_2	h_3	h_4	h_5	h_6

FIGURE 1 – Calcul de $h = f * g$

Soient $p, q \in \mathcal{P}$ deux polynômes premiers entre eux. Généralement, q est de la forme $q = 2^l$ avec $l = \lfloor \log_2 N \rfloor$ et $p = 2$ ou $p = 3$ ou dans certaines versions

$p = 2 + X$. On note $\mathbb{Z}_q$ l'anneau des entiers modulo q, c'est à dire $\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$. Dans NTRU, $\mathbb{Z}_q$ est représenté par l'intervalle $\left[-\frac{q}{2}, \frac{q}{2}\right[$. En réduisant $\mathcal{P}$ modulo q, on obtient l'anneau

$$\mathcal{P}_q = \mathbb{Z}_q[X]/(X^N - 1).$$

La correspondance entre $\mathcal{P}$ et $\mathcal{P}_q$ se fait par l'homomorphisme $\pi_q : \mathcal{P} \to \mathcal{P}_q$. Ainsi un polynôme $f = (f_0, f_1, \cdots, f_{N-1}) \in \mathcal{P}$ sera tout simplement représenté par le polynôme

$$\pi_q(f) = (f_0 \pmod q, f_1 \pmod q, \cdots, f_{N-1} \pmod q) \in \mathcal{P}_q.$$

Un polynôme $f \in \mathcal{P}_q$ est dit inversible modulo q s'il existe un polynôme f_q dans $\mathcal{P}_q$ tel que $f * f_q = f_q * f = 1$ dans $\mathcal{P}_q$. Si $\gcd(f, X^N - 1) = 1$, alors il existe deux polynômes u et v de $\mathcal{P}$ tels que dans

$$u * f + v * \left(X^N - 1\right) = 1.$$

Alors $f_q \equiv u \pmod q$. De façon similaire, pour un polynôme $p \in \mathcal{P}$, on définit l'anneau $\mathcal{P}_p$ par

$$\mathcal{P}_p = \mathbb{Z}[X]/\left(X^N - 1, p\right),$$

obtenu en réduisant $\mathcal{P}$ modulo p. Si p est un entier, alors $\mathcal{P}_p = \mathbb{Z}_p[X]/\left(X^N - 1\right)$.

2.2 Paramètres de NTRU

NTRU est basé sur la combinaison de plusieurs paramètres qui ont évolué suivant les différentes attaques qui ont été proposées contre NTRUEncrypt et plus spécialement contre les premières versions de signatures basées sur NTRU.

Soit d un entier positif. On pose

$$\mathcal{B}(d) = \left\{ f \in \mathbb{Z}_2[X]/\left(X^N - 1\right) \mid f = \sum_{i=1}^{d} X^{n_i},\ 0 \leq n_1 < \cdots < n_d \leq N \right\}.$$

Ceci signifie que les éléments de $\mathcal{B}(d)$ ont exactement d coefficients égaux à 1 et le reste des coefficients est nul. De même, soient d_1 et d_2 deux entiers positifs. On pose

$$\mathcal{T}(d_1, d_2) = \left\{ f \in \mathbb{Z}_3[X]/\left(X^N - 1\right) \mid f = \sum_{i=1}^{d_1} X^{n_i} - \sum_{j=1}^{d_2} X^{m_j},\ n_i \neq m_j \right\}.$$

Autrement dit, $\mathcal{T}(d_1, d_2)$ est l'ensemble des polynômes ayant exactement d_1 coefficients égaux à 1, d_2 coefficients égaux à -1 et le reste des coefficients est nul.

Les paramètres de NTRU sont alors les suivants.

- Un nombre entier N. Ce nombre doit être premier et assez grand.
- Un nombre entier q, généralement de la forme $q = 2^l$ avec $l = \lfloor \log_2(N) \rfloor$.
- Un paramètre p qui est soit un nombre entier p premier avec q soit un polynôme $p = \alpha + \beta X \in \mathbb{Z}[X]$, inversible modulo q. Généralement $p = 2 + X$ ou $p = 2$.
- Un ensemble $\mathcal{L}_f \subset \mathcal{P}$ de polynômes f.
- Un ensemble $\mathcal{L}_g \subset \mathcal{P}$ de polynômes g.
- Un ensemble $\mathcal{L}_m \subset \mathcal{P}$ des messages secrets.
- Un ensemble $\mathcal{L}_r \subset \mathcal{P}$ des polynômes auxiliaires secrets.

Ainsi, suivant les versions de NTRU, les paramètres ci-dessus ont évolués en moyenne tous les trois ans.

Version	p	q	$\mathcal{L}_f$	$\mathcal{L}_g$	$\mathcal{L}_r$	$\mathcal{L}_m$
1998	3	2^k	$\mathcal{T}(d_f, d_f - 1)$	$\mathcal{T}(d_g, d_g)$	$\mathcal{T}(d_r, d_r)$	$\mathcal{T}(d_m, d_m)$
2001	$2 + X$	2^k	$1 + p * F$	$\mathcal{B}(d_g)$	$\mathcal{B}(d_r)$	$\mathcal{B}(d_m)$
2005	2	2^k	$1 + p * F$	$\mathcal{B}(d_g)$	$\mathcal{B}(d_r)$	$\mathcal{B}(d_m)$

Les nombres entiers d_f, d_g, d_r et d_m sont fixés pour chaque version. Actuellement, ces paramètres sont les suivants (voir [8]) :

Sécurité	N	p	q	d_f	d_g	d_r
Moyenne	251	2	128	72	71	72
Haute	347	2	128	64	173	64
Très haute	503	2	256	420	251	170

2.3 Utilisation de NTRU

L'utilisation du cryptosystème se déroule de la façon suivante. Supposons qu'une personne B souhaite envoyer un message M à une personne A. Alors le processus se fait en trois étapes :

1. A doit générer une clé publique h et des clés privées f et f_p.
2. B doit transformer le message clair M en un message chiffré e et envoyer le message chiffré e à A.
3. A doit déchiffrer le message reçu e et retrouver le message clair M.

2.3.1 Générations de clés

L'étape de génération des clés doit être réalisée par la personne A qui choisit le niveau de sécurité souhaitée par le choix de N, p et q. La procédure est alors la suivante.

Génération des clés :

1. Choisir aléatoirement un polynôme $f \in \mathcal{L}_f$.
2. Calculer l'inverse f_q de f dans $\mathcal{P}_q$, c'est à dire $f * f_q \equiv 1 \pmod{q}$.
3. Calculer l'inverse f_p de f dans $\mathcal{P}_p$, c'est à dire $f * f_p \equiv 1 \pmod{p}$.
4. Choisir aléatoirement un polynôme $g \in \mathcal{L}_g$.
5. Calculer $h \equiv p * g * f_q \pmod{q}$ dans $\mathcal{P}_q$.

La clé publique est alors h et la clé secrète est (f, f_p). Pour s'assurer de l'existence de l'inverse f_p de f dans $\mathcal{P}_p$, NTRU préconise de prendre f vérifiant $f \equiv 1 \mod p$, c'est à dire de prendre f de la forme $f = 1 + pf_1$.

2.3.2 Chiffrement

Pour chiffrer un message M, la personne B doit le transformer en un polynôme $m \in \mathcal{L}_m$ et ensuite le transformer en un polynôme chiffré e avec la procédure suivante.

Chiffrement :

1. Choisir aléatoirement un polynôme $r \in \mathcal{L}_r$.
2. Calculer $e \equiv r * h + m \pmod{q}$ dans $\mathcal{P}_q$.

Le message chiffré est alors e. Il faut remarquer que, puisque r est aléatoire, si on chiffre deux fois le même message M, le message chiffré peut être différent.

2.3.3 Déchiffrement

Pour déchiffrer un message $e \in \mathcal{P}_q$, la personne A doit utiliser les clés secrètes f et f_p dans la procédure suivante.

Déchiffrement :

1. Calculer $a \equiv f * e \pmod{q}$ dans $\mathcal{P}_q$ avec des coefficients dans l'intervalle $\left[-\frac{q}{2}, \frac{q}{2}\right[$.
2. Calculer $m \equiv f_p * a \pmod{p}$ dans $\mathcal{P}_p$.

2.3.4 Exactitude du déchiffrement

Pour s'assurer de l'exactitude du déchiffrement, on commence par calculer $a \equiv f * e \pmod{q}$ dans $\mathcal{P}_q$ sachant que $e \equiv r * h + m \pmod{q}$ et que $h \equiv p * g * f_q$

mod q. On a

$$\begin{aligned} a &\equiv f * e \pmod{q} \\ a &\equiv f * (r * h + m) \pmod{q} \\ a &\equiv f * r * (p * g * f_q) + f * m \pmod{q} \\ a &\equiv p * r * g * f * f_q + f * m \pmod{q} \\ a &\equiv p * r * g + f * m \pmod{q}. \end{aligned}$$

Maintenant, soit $a' = p * r * g + f * m$ défini par un calcul exacte dans $\mathcal{P}$ et non dans $\mathcal{P}_q$, c'est à dire sans modulo q. Si les coefficients de a' sont dans un intervalle $[A, A+q[$, alors en ramenant les coefficients de a dans le même intervalle $[A, A+q[$, on obtient $a = a'$ et en réduisant a modulo p, on obtient $a \equiv f * m \pmod{p}$. Alors

$$f_p * a \equiv f_p * f * m \equiv m \pmod{p}.$$

Si les coefficients de a' ne sont pas dans un intervalle de la forme $[A, A+q[$, alors $a' \neq a$ et le déchiffrement peut ne pas être conforme. En utilisant la largeur $||a'||_\infty$ de a' définie par

$$||a'||_\infty = \max_{0 \leq i < N} a'_i - \min_{0 \leq i < N} a'_i,$$

deux cas sont alors possible.

- Si $||a'||_\infty \geq q$, il n'y a pas moyen de passer de a à a' avec une translation des coefficients. Dans ce cas, le déchiffrement n'est pas exacte, et m ne peut pas être retrouvé.
- Si $||a'||_\infty < q$, alors on peut trouver un intervalle $[A, A+q[$ dans lequel on peut ramener les coefficients de a par une translation. Dans ce cas, on obtient le bon message après déchiffrement.

3 La sécurité de NTRU

Dans cette section, on décrit les problèmes sur lesquelles repose la sécurité de NTRU, à savoir SVP et CVP, ainsi que quelques attaques sur ce cryptosystème.

3.1 Les problèmes SVP et CVP

Les problèmes SVP et CVP sont des problèmes issus de le théorie des réseaux et de leur réduction. Soit m un entier positif. On considère l'ensemble $\mathbb{R}^m$ des vecteurs $u = (u_1, \cdots, u_m)$, muni du produit scalaire et de la norme euclidienne.

Définition 3.1. *Soient $u = (u_1, \cdots, u_m)$ et $v = (v_1, \cdots, v_m)$ deux vecteurs de $\mathbb{R}^m$. Le produit scalaire de u et v est*

$$\langle u, v \rangle = \sum_{i=1}^{m} u_i v_i.$$

Définition 3.2. *Soit $u = (u_1, \cdots, u_m)$ un vecteur de $\mathbb{R}^m$. La norme euclidienne de u est*

$$||u|| = \sqrt{\langle u, u \rangle} = \sqrt{\sum_{i=1}^{m} u_i^2}$$

Soit n un entier positif avec $n \leq m$. Soient $(b_1, \cdots, b_n)$ des vecteurs linéairement indépendants de $\mathbb{R}^m$. Soit B la matrice dont les colonnes sont composées des coordonnées des vecteurs $b_1, \cdots, b_n$.

Définition 3.3. *Le réseau engendré par la famille $(b_1, \cdots, b_n)$ (ou par B) est l'ensemble*

$$L = \sum_{i=1}^{n} \mathbb{Z} b_i = \left\{ \sum_{i=1}^{n} x_i b_i \mid x_i \in \mathbb{Z} \right\}.$$

L'entier n est la dimension de L.

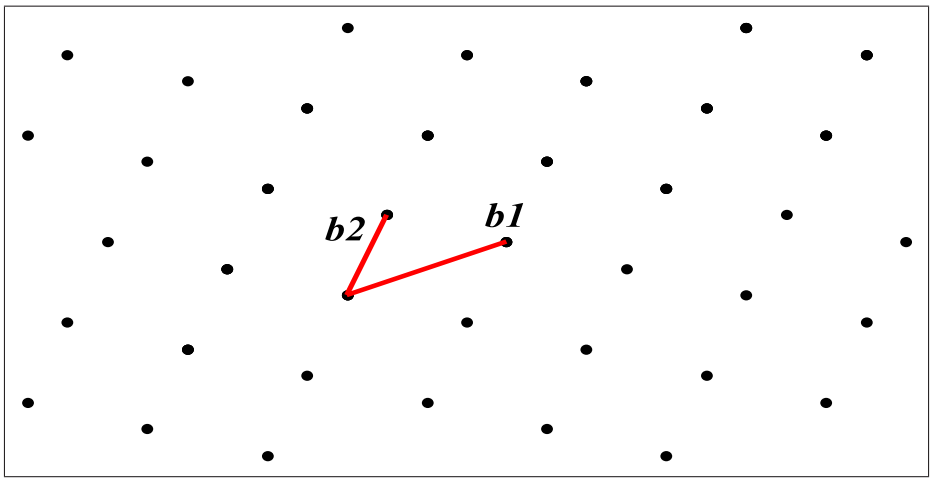

FIGURE 2 – Un réseau de base (b_1, b_2)

Pour la rapidité des calculs dans un réseau, il est plus pratique d'utiliser des bases orthogonales. Le procédé suivant permet de produire une base orthogonale à partir d'une base $(b_1 \cdots, b_n)$.

Théorème 3.4 (Orthogonalisation de Gram-Schmidt)**.** *Soit* $(b_1 \cdots, b_n)$ *une base d'un réseau* $L \subset \mathbb{R}^m$*. On considère la famille de vecteurs* $(b_1^* \cdots, b_n^*)$ *définie par*

$$b_1^* = b_1, \quad b_i^* = b_i - \sum_{j=1}^{i-1} \mu_{i,j} b_j^*,$$

avec pour $j < i$

$$\mu_{i,j} = \frac{\langle b_i, b_j^* \rangle}{\langle b_j^*, b_j^* \rangle}.$$

Alors $(b_1^* \cdots, b_n^*)$ *est une base orthogonale de* $\mathbb{R}^m$*.*

Le déterminant du réseau L est alors défini par

$$\det(L) = \prod_{i=1}^{n} ||b_i^*||.$$

Un réseau L admet plusieurs bases. Le passage entre deux bases est une matrice carrée à coefficients dans $\mathbb{Z}$, de déterminant ± 1. Parmi toutes les bases possibles, certaines ont de meilleures propriétés que d'autres. La recherche d'une bonne base est un problème NP-complet. En 1982, Lenstra, Lenstra et Lovasz [12] ont proposé l'algorithme LLL qui détermine une base avec de très bonnes propriétés. Cette algorithme utilise la notion de base réduite au sens de LLL.

Définition 3.5. *Une base* $(b_1 \cdots, b_n)$ *est LLL-réduite si, la base* $(b_1^* \cdots, b_n^*)$ *produite par la la méthode d'orthogonalisation de Gram-Schmidt vérifie*

$$\begin{aligned} |\mu_{i,j}| &\leq \frac{1}{2}, \quad \textit{pour} \quad 1 \leq j < i \leq n, \\ \frac{3}{4}\|b_{i-1}^*\|^2 &\leq \|b_i^* + \mu_{i,i-1} b_{i-1}^*\|^2, \quad \textit{pour} \quad 1 < i \leq n. \end{aligned}$$

L'algorithme LLL transforme une base $(v_1 \cdots, v_n)$ de L en une base LLL-réduite $(b_1 \cdots, b_n)$, avec entre autre les propriété du théorème suivant (voir [3]).

Théorème 3.6. *Soit* $(b_1 \cdots, b_n)$ *une base LLL-réduite et* $(b_1^*, \cdots, b_n^*)$ *la base orthogonale associée par la méthode de Gram-Schmidt. Alors*

1. $\|b_j^*\|^2 \leq 2^{i-j}\|b_i^*\|^2$ *pour* $1 \leq j \leq i \leq n$.
2. $\det(L) \leq \prod_{i=1}^{n} \|b_i\| \leq 2^{\frac{n(n-1)}{4}} \det(L)$.
3. $\|b_j\| \leq 2^{\frac{i-1}{2}}\|b_i^*\|$ *pour* $1 \leq j \leq i \leq n$.
4. $\|b_1\| \leq 2^{\frac{n-1}{4}} (\det(L))^{\frac{1}{n}}$.
5. *Pour tout vecteur non nul* $v \in L$, $\|b_1\| \leq 2^{\frac{n-1}{2}}\|v\|$.

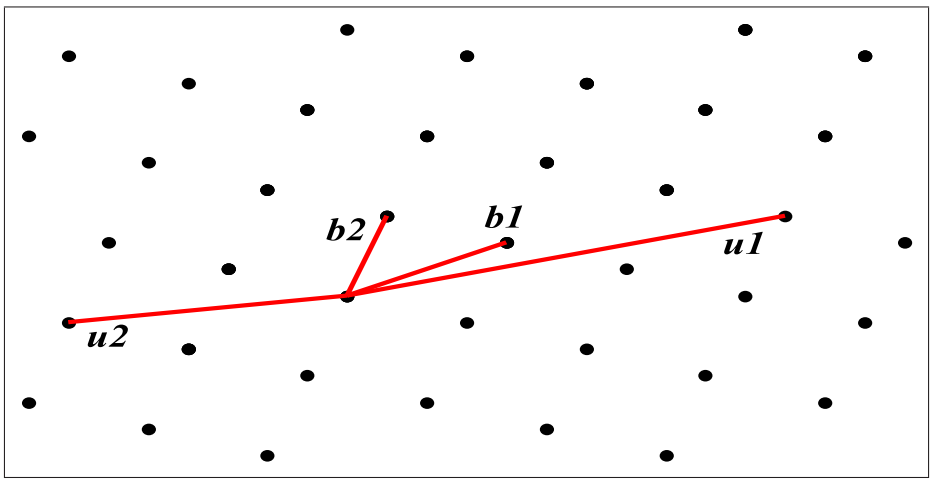

FIGURE 3 – Un réseau de base (u_1, u_2) et la base réduite $(b_1, b_2) = (u_1+u_2, 2u_1+3u_2)$

L'algorithme LLL est très efficace puisque sa complexité est polynômiale en $\mathcal{O}\left(n^6(\ln b)^3\right)$ où $b = \max_{1\leq i\leq n} \|b_i\|^2$. On remarque en particulier que l'algorithme LLL produit des bases avec des vecteurs assez courts, ce qui peut apporter une réponse partiellement satisfaisante aux deux problèmes (NP-durs) suivants.

Problème SVP, Shortest Vector Problem (problème du plus court vecteur) : *Etant donné une base B d'un réseau L, trouver un vecteur non nul de L le plus court possible pour la norme euclidienne.*

Problème CVP, Closest Vector Problem (problème du plus proche vecteur) : *Etant donné une base B d'un réseau L et un vecteur $v \in L$, trouver un vecteur de L le plus proche possible de v pour la norme euclidienne.*

3.2 L'attaque de Coppersmith et Shamir

Juste après la publication de NTRU, Coppersmith et Shamir [4] ont proposé une attaque contre la clé publique pour les petites valeurs du paramètre N. Dans NTRU, la clé publique $h \equiv f_q * g \pmod q$ vérifie $f * h \equiv g \pmod q$. Alors, il existe un polynôme $u \in \mathcal{P} = \mathbb{Z}/\left(X^N - 1\right)$ tel que

$$f * h - q * u = g.$$

On considère alors l'ensemble

$$\mathcal{E} = \left\{(f, g) \in \mathcal{P}^2 \mid \exists u \in \mathcal{P},\ f * h - q * u = g\right\}.$$

On vérifie facilement que $\mathcal{E}$ est un réseau et sous forme matricielle, ceci s'écrit sous la forme :

$$(f, -u) * \begin{bmatrix} 1 & h \\ 0 & q \end{bmatrix} = (f, g).$$

Avec les coordonnées de f, g et h, l'égalité ci-dessus prend la forme

$$\left[\begin{array}{c} f_0 \\ f_1 \\ \vdots \\ f_{N-1} \\ \hline g_0 \\ g_1 \\ \vdots \\ g_{N-1} \end{array}\right] = \left[\begin{array}{c} f_0 \\ f_1 \\ \vdots \\ f_{N-1} \\ \hline -u_1 \\ -u_2 \\ \vdots \\ -u_{N-1} \end{array}\right] * \left[\begin{array}{cccc||cccc} 1 & 0 & \cdots & 0 & h_0 & h_1 & \cdots & h_{N-1} \\ 0 & 1 & \cdots & 0 & h_{N-1} & h_0 & \cdots & h_{N-2} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & h_1 & h_2 & \cdots & h_0 \\ \hline\hline 0 & 0 & \cdots & 0 & q & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & q & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & q \end{array}\right]$$

Dans la clé publique h, les polynômes f et g ont des petits coefficients et $(f, g) \in \mathcal{E}$. Alors en appliquant l'algorithme LLL au réseau $\mathcal{E}$, on obtient une base réduite dans laquelle les premiers vecteurs sont assez courts, et on peut ainsi déterminer f et g. Si (f, g) est le plus court vecteur du réseau $\mathcal{E}$, cela revient à résoudre le problème SVP. Ceci illustre que la sécurité de NTRU est basée sur ce problème difficile.

3.3 Attaques exhaustives

Dans NTRU, les clés secrètes f et g sont des polynômes avec peu de coefficients et qui sont tous égaux à ± 1. La clé publique est de la forme $h \equiv p * g * f_q \pmod{q}$ où $f_q * f \equiv 1 \pmod{q}$. On a ainsi

$$f * h \equiv p * g \pmod{q} \quad \text{et} \quad p * h^{-1} * g \equiv f \pmod{q} \quad .$$

Une première attaque exhaustive consiste à tester des valeurs de $f \in \mathcal{L}_f$ en calculant $f * h \pmod{q}$ et tester si le polynôme obtenu est de la forme $p * g$ avec $g \in \mathcal{L}_g$. Le nombre de possibilités pour le choix de f est

$$|\mathcal{L}_f| = \binom{N}{d_f} = \frac{N!}{d_f!(N - d_f)!}.$$

Par exemple, avec $N = 251$ et $d_f = 72$, on obtient $|\mathcal{L}_f| \approx 1.187 \times 10^{64}$.

Une autre attaque exhaustive consiste à tester des valeurs de $g \in \mathcal{L}_g$ en calculant $p * h^{-1} * g \pmod{q}$ et tester si le polynôme obtenu est un polynôme f avec $f \in \mathcal{L}_f$. Puisque les polynômes de $\mathcal{L}_g$ sont de la forme $g = \sum_{i=1}^{d_g} X^{n_i},\ 0 \leq n_1 < \cdots <$

$n_d \leq N$, alors cette méthode exhaustive porte sur un nombre total de possibilités de la forme

$$|\mathcal{L}_g| = \binom{N}{d_g} = \frac{N!}{d_g!(N-d_g)!}.$$

Une troisième attaque exhaustive, appelé attaque du milieu, concerne aussi la clé publique h. En écrivant $f = f_1 + f_2$, on obtient

$$f_1 * h + f_2 * h \equiv p * g \pmod{q}.$$

L'attaque du milieu consiste à tester des polynômes f_1 et f_2 dont le nombre total de coefficients non nuls égaux à 1 est d_f en calculant $f_1 * h + f_2 * h \pmod{q}$, et en testant si le polynôme obtenu est de la forme $p * g$ avec $g \in \mathcal{L}_g$.

Enfin, une quatrième attaque exhaustive porte sur le message chiffré e qui est de la forme $e \equiv p * r * h + m \pmod{q}$. En écrivant $e - p * r * h \equiv m \pmod{q}$, cette attaque consiste à tester les polynômes $r \in \mathcal{L}_r$ en calculant $e - p * r * h \pmod{q}$ et en testant si le polynôme obtenu est dans $\mathcal{L}_m$. Le nombre total de possibilités est de la forme

$$|\mathcal{L}_r| = \binom{N}{d_r} = \frac{N!}{d_r!(N-d_r)!}.$$

Toutes ces attaques exhaustives ne peuvent fonctionner que si les clés privés sont très mal choisies, ce qui est fortement improbables puisque f et g sont définis de façon aléatoire.

4 Variantes de NTRU

Le cryptosystème NTRU possède plusieurs versions à cause de quelques attaques sur les premières versions. Ceci a permis l'évolution de NTRU vers des versions plus sûres. Une particularité des attaques basées sur la réduction des réseaux, c'est que ces attaques ne sont pas applicables si l'anneau $\mathbb{Z}$ est remplacé par un autre anneau, non commutatif par exemple. Ainsi, plusieurs variantes ont été proposées.

4.1 Généralisation par Banks et Shparlinski, 2002

En 2002, Banks et Shparlinski [1] ont présenté une variante de NTRU. Cette variante diffère essentiellement de NTRU dans l'étape de génération des clés.

Génération des clés :

1. Choisir aléatoirement un polynôme $f \in \mathcal{L}_f$.
2. Calculer l'inverse f_p de f dans $\mathcal{P}_p$. Ainsi $f * f_p \equiv 1 \pmod{p}$.
3. Choisir aléatoirement un polynôme $g \in \mathcal{L}_g$.
4. Choisir aléatoirement un polynôme $G \in \mathcal{P}$.
5. Calculer l'inverse G_q de G dans $\mathcal{P}_q$. Ainsi $G * G_q \equiv 1 \pmod{p}$.
6. Calculer $h \equiv G_q * g \pmod{q}$ dans $\mathcal{P}_q$.
7. Calculer $H \equiv G_q * f \pmod{q}$ dans $\mathcal{P}_q$.

La clé publique est alors constituée par le couple (h, H) et la clé secrète est (f, f_p, G). Pour chiffrer un message $m \in \mathcal{L}_m$, la procédure est la suivante.

Chiffrement :

1. Choisir aléatoirement un polynôme $r \in \mathcal{L}_r$.
2. Calculer $e \equiv p * r * h + H * m \pmod{q}$ dans $\mathcal{P}_q$.

Le message chiffré e est alors transmis. Pour déchiffrer e et en déduire m, la procédure est la suivante.

Déchiffrement :

1. Calculer $a \equiv G * e \pmod{q}$ dans $\mathcal{P}_q$ avec des coefficients dans l'intervalle $\left[-\frac{q}{2}, \frac{q}{2}\right[$.
2. Calculer $m \equiv f_p * a \pmod{p}$ dans $\mathcal{P}_p$.

En raison de la similitude avec NTRU, cette variante se trouve confrontée aux mêmes problèmes de déchiffrement que NTRU. En effet le déchiffrement est similaire :

$$\begin{aligned}
a &\equiv G * e \pmod{q} \\
a &\equiv G * (p * r * h + H * m) \pmod{q} \\
a &\equiv p * r * G * h + G * H * m \pmod{q} \\
a &\equiv p * r * G * (G_q * g) + G * (G_q * f) * m \pmod{q} \\
a &\equiv p * r * g + f * m \pmod{q},
\end{aligned}$$

et aboutit donc au même polynôme $p * r * g + f * m$ que dans NTRU. La suite est alors identique.

Cette variante est désavantagée par rapport à NTRU car le temps d'exécution est pratiquement le double à cause de la présence de deux polynômes h et H. Le principal avantage est que cette variante évite les attaques classiques sur NTRU,

en particulier les attaques exhaustives sur la clé publique et l'attaque basée sur la réduction des réseaux.

4.2 MaTRU, Coglianese et Goi, 2005

En 2005, Coglianese et Goi [2] ont proposé une variante de NTRU en prenant pour domaine de calculs l'anneau $\mathbb{M}$ des matrices carrées $k \times k$ à termes dans $\mathcal{P} = \mathbb{Z}[X]/\left(X^N - 1\right)$:

$$\mathbb{M} = \{M \mid M = [f_{ij}]_{1 \leq i,j \leq k},\ f_{ij} \in \mathcal{P}\}.$$

La matrice unité sera notée I. En plus de N et k, cette variante définit des nombres premiers entre eux p et q en plus des sous ensembles $\mathcal{L}_f$, $\mathcal{L}_\phi$, $\mathcal{L}_A$, $\mathcal{L}_w$ et $\mathcal{L}_m$ de $\mathbb{M}$.

- $\mathcal{L}_A$ est l'ensemble des matrices $C \in \mathbb{M}$ pour lesquelles la famille $C^0, C^1, \cdots, C^{k-1}$ est libre modulo q.
- $\mathcal{L}_f$ et $\mathcal{L}_\Phi$ sont des ensembles de matrices $D \in \mathbb{M}$ vérifiant

$$D = \sum_{i=0}^{k-1} c_i C^i, \quad C \in \mathcal{L}_A, \quad c_0, c_1, \cdots, c_{k-1} \in \mathcal{P}.$$

- $\mathcal{L}_m$ est l'ensemble des matrices de $\mathbb{M}$ dont les termes sont des polynômes ayant des coefficients dans l'intervalle $\left\lceil -\frac{p-1}{2}, \frac{p-1}{2} \right\rceil$.

Le fonctionnement de MaTRU repose aussi sur trois étapes. La première étape est la génération des clés.

Génération des clés :

1. Choisir aléatoirement deux matrices $A, B \in \mathcal{L}_A$.
2. Choisir aléatoirement k polynômes $\alpha_0, \alpha_1, \cdots, \alpha_{k-1} \in \mathcal{P}$.
3. Choisir aléatoirement k polynômes $\beta_0, \beta_1, \cdots, \beta_{k-1} \in \mathcal{P}$.
4. Calculer $f = \sum_{i=0}^{k-1} \alpha_i A^i \in \mathcal{L}_f$.
5. Calculer $g = \sum_{i=0}^{k-1} \beta_i B^i \in \mathcal{L}_f$.
6. Calculer F_q tel que $F_q * f \equiv I \pmod q$ et F_p tel que $F_p * f \equiv I \pmod p$.
7. Calculer G_q tel que $G_q * g \equiv I \pmod q$ et G_p tel que $G_p * g \equiv I \pmod p$.
8. Choisir aléatoirement une matrice $w \in \mathcal{L}_w$.
9. Calculer $h \equiv F_q * w * G_q \pmod q$ dans $\mathcal{P}_q$.

La clé publique est alors le triplet de matrices (h, A, B). La clé secrète est (f, g, F_p, G_p).

La deuxième étape est le chiffrement du message $m \in \mathcal{L}_m$.

Chiffrement :

1. Choisir aléatoirement k polynômes $\phi_0, \phi_1, \cdots, \phi_{k-1} \in \mathcal{P}$.
2. Choisir aléatoirement k polynômes $\psi_0, \psi_1, \cdots, \psi_{k-1} \in \mathcal{P}$.
3. Calculer $\Phi = \sum_{i=0}^{k-1} \phi_i A^i \in \mathcal{L}_\Phi$.
4. Calculer $\Psi = \sum_{i=0}^{k-1} \psi_i B^i \in \mathcal{L}_\Phi$.
5. Calculer $e \equiv p * \Phi * h * \Psi + m \pmod{q}$.

Le message chiffré à envoyer est alors e. La dernière étape sera le déchiffrement du message e.

Déchiffrement :

1. Calculer $a \equiv f * e * g \pmod{q}$ en mettant les coefficients des polynômes dans l'intervalle $\left[-\frac{q}{2}, \frac{q}{2}\right[$.
2. Calculer $m \equiv F_p * a * G_p \pmod{p}$.

L'analyse du déchiffrement donne lieu aux calculs suivants.

$$\begin{aligned} a &\equiv f * e * g \pmod{q} \\ &\equiv f * (p * \Phi * h * \Psi + m) * g \pmod{q} \\ &\equiv p * f * \Phi * h * \Psi * g + f * m * g \pmod{q} \\ &\equiv p * f * \Phi * F_q * w * G_q * \Psi * g + f * m * g \pmod{q}. \end{aligned}$$

Puisque f et Φ sont combinaisons linéaires des A^i, alors $f * \Phi = \Phi * f$. Il en est de même pour Ψ et g. Alors, sachant que $F_q * f \equiv I \pmod{q}$ et $G_q * g \equiv I \pmod{q}$, on obtient

$$\begin{aligned} a &\equiv p * f * \Phi * F_q * w * G_q * \Psi * g + f * m * g \pmod{q} \\ &\equiv p * \Phi * f * F_q * w * G_q * g * \Psi + f * m * g \pmod{q} \\ &\equiv p * \Phi * w * \Psi + f * m * g \pmod{q} \end{aligned}$$

Le processus de déchiffrement peut rencontrer le même problème que NTRU. En effet, si le calcul exact de $p * \Phi * w * \Psi + f * m * g$ a des polynômes dont les coefficients ne sont pas dans $\mathbb{Z}_q$ après translation des coefficients, alors le déchiffrement ne sera pas conforme. Dans le cas contraire, on a $a \equiv f * m * g \pmod{p}$ et donc

$$m \equiv F_p * a * G_p \pmod{p}.$$

Le chiffrement et le déchiffrement de MaTRU sont $\frac{k}{2}$ fois plus rapides que NTRU mais la clé publique de MaTRU a une longueur plus grande que celle qui provient

de NTRU. D'autre part, MaTRU présente des niveaux de sécurité comparables à ceux de NTRU.

4.3 CTRU, Gaborit, Ohler et Solé, 2006

CTRU a été proposé en 2006 par Gaborit, Ohler et Solé [7] en replaçant le rôle de l'anneau des entiers $\mathbb{Z}$ de NTRU par l'anneau

$$\mathbb{A} = \mathbb{F}_2[T].$$

Les opérations de CTRU ont pour domaine l'anneau

$$\mathcal{R} = \mathbb{A}[X]\left(X^N - 1\right).$$

Si F est un polynôme de $\mathcal{R}$, on note $\deg_T(F)$ le degrés de F en tant que polynôme en T.

Soit N un nombre entier et P, Q deux polynômes de $\mathbb{A}$, avec $\deg_T(P) = s$, $\deg_T(Q) = m$, $\gcd(s, m) = 1$, $2 \leq s \leq m$.

Soient d_f, d_g et d_ϕ des entiers vérifiant $d_f = m - s - 1$ et $d_g = d_\phi = \left\lfloor \frac{1}{2}(m - s - 1) \right\rfloor$. On considère les ensembles

$$\begin{aligned}
\mathcal{L}_m &= \{M \in \mathcal{R} \mid \deg_T M < s\} \\
\mathcal{L}_f &= \{f \in \mathcal{R} \mid \deg_T f < 1 + d_f\} \\
\mathcal{L}_g &= \{g \in \mathcal{R} \mid \deg_T g < 1 + d_g\} \\
\mathcal{L}_\phi &= \{\phi \in \mathcal{R} \mid \deg_T \phi < 1 + d_\phi\},
\end{aligned}$$

où $\deg_T f$ est le degré de f en tant que polynôme en T.

Pour utiliser CTRU, on procède de façon similaire que NTRU avec les trois procédure : génération des clés, chiffrement et déchiffrement.

Génération des clés :

1. Choisir aléatoirement $f \in \mathcal{L}_f$.
2. Calculer l'inverse f_Q de f modulo $\left(X^N - 1, Q\right)$, c'est à dire que $f * f_Q = 1$ dans l'anneau quotient $\mathcal{R}/(Q)$.
3. Calculer l'inverse f_P de f modulo $\left(X^N - 1, P\right)$, c'est à dire que $f * f_P = 1$ dans l'anneau quotient $\mathcal{R}/(P)$.
4. Choisir aléatoirement $g \in \mathcal{L}_g$.
5. Calculer $h \equiv g * f_Q$ dans l'anneau quotient $\mathcal{R}/(Q)$.

La clé publique est alors h et la clé secrète est (f, f_P). Pour envoyer un message $M \in \mathcal{L}_m$, on procède comme ci-dessous.

Chiffrement :

1. Choisir aléatoirement $\phi \in \mathcal{L}_\phi$.
2. Calculer $e \equiv P * \phi * h + M \pmod{Q}$ dans l'anneau quotient $\mathcal{R}/(Q)$.

Le message chiffré à transmettre est alors e. Pour déchiffrer e et retrouver le message originale M, on procède selon la procédure de déchiffrement.

Déchiffrement :

1. Calculer $a \equiv e * f$ dans l'anneau quotient $\mathcal{R}/(Q)$.
2. Calculer $M \equiv a * f_Q$ dans l'anneau quotient $\mathcal{R}/(P)$.

Le déchiffrement se démontre comme dans NTRU. En effet, dans l'anneau quotient $\mathcal{R}/(Q)$, on a

$$e \equiv P * \phi * h + M \equiv P * \phi * g * f_Q + M,$$

et donc

$$a \equiv e * f \equiv P * \phi * g * f_Q * f + f * M \equiv P * \phi * g + f * M.$$

On a alors

$$\begin{aligned} \deg_T(P * \phi * g + f * M) &\leq \max\left(\deg_T(P * \phi * g), \deg_T(f * M)\right) \\ &\leq \max\left(s + d_\phi + d_g, s + d_f\right). \end{aligned}$$

Ainsi si $\max\left(s + d_\phi + d_g, s + d_f\right) < m$, le calcul de $P * \phi * g + f * M$ est exact dans $\mathcal{R}$ et donc a peut être considéré modulo P et par la suite $M \equiv a * f_Q \pmod{P}$ sera conforme. Ces conditions sont vérifiées avec le choix de départ des nombres entiers d_f, d_g et d_ϕ.

Certaines attaques sur NTRU peuvent être transformées en attaques sur CTRU, notamment les attaques exhaustives sur la clé publique h ainsi que l'attaque par le milieu et l'attaque contre les transmissions multiples. Par contre, l'attaque de Coppersmith et Shamir, basée sur la réduction des réseaux ne s'adapte pas à CTRU. Par contre, une attaque basée sur les formes normales de Popov des matrices à coefficients polynômiaux peut être envisagée mais n'est pas efficace sur CTRU quand les paramètres sont bien choisis. Malgré toutes ces précautions, CTRU fût par la suite cryptanalysé par Kouzmenko [11] en 2006 et par Vats[20] en 2008.

4.4 Autres généralisations

En 2006, Kouzmenko [11] a présenté une variante de NTRU en remplaçant l'anneau $\mathcal{P} = \mathbb{Z}/\left(X^N - 1\right)$ par l'anneau $\mathcal{P}' = (\mathbb{Z} + \mathbb{Z}i)[X]/\left(X^N - 1\right)$ où

$$\mathbb{Z} + \mathbb{Z}i = \{a + ib \mid (a, b) \in \mathbb{Z}\},$$

est l'anneau des entiers de Gauss.

Une autre variante de NTRU fût présentée par Nevins, KarimianPour et Miri [15] en 2009. Cette variante a pour domaine de calculs l'ensemble $(\mathbb{Z}[\zeta_3])/\left(X^N - 1\right)$ où $\mathbb{Z}[\zeta_3]$ est l'ensemble des entiers d'Eisenstein :

$$\mathbb{Z}[\zeta_3] = \{a + b\omega \mid (a, b) \in \mathbb{Z}\} \quad \text{avec} \quad \omega = \frac{1}{2}\left(-1 + i\sqrt{3}\right) = e^{2\pi i/3}.$$

Notons enfin une autre variante de NTRU, appelée QTRU, présentée par Malekian, Zakerolhosseini et Mashatan [13] en 2009. Cette version remplace l'ensemble $\mathbb{Z}$ par l'anneau des quaternions $\mathbb{H}$:

$$\mathbb{H} = \left\{a + bi + cj + dk, \quad i^2 = j^2 = k^2 = ijk = -1\right\}.$$

L'avantage commun de ces variantes de NTRU est leur résistance aux attaques basées sur la réduction des réseau.

5 Applications et avantages de NTRU

5.1 Avantages de NTRU

Malgré le fait que le cryptosystème à clé publique le plus utilisé soit RSA, le principal avantage de NTRU par rapport aux autres cryptosystèmes à clé publique est la rapidité des processus de chiffrement et de déchiffrement. En effet, le produit de convolution $*$ opérant sur deux polynômes a un coût en $\mathfrak{O}\left(N^2\right)$ opérations. En plus, les polynômes ont des coefficients assez petits (0, 1 et -1), ce qui simplifie et accélère les calculs. Un autre avantage concernant le futur de la cryptographie à clé publique est que NTRU résiste encore contre les attaques des ordinateurs quantiques contrairement à RSA, EL Gamal et ECC qui seront cassés (voir Shor [19]). En effet, les problèmes sur lesquelles repose la sécurité de NTRU son les problèmes SVP et CVP et il n' existe pas pour l'instant, pour ces deux problèmes, de solutions basés sur la théorie quantique.

5.2 Applications de NTRU

Dans les dernières années, l'échange de l'information a atteint de niveaux très importants. Les procédés cryptographiques ont donc été sollicités de façon intense. Dans un grand nombre d'outils technologiques, les capacités de calculs sont très réduites. Dans ce cadre, NTRU se présente comme un moyen cryptographique efficace. Parmi les utilisations notables de NTRU, on peut citer les éléments suivants :

- Authentification et accès sécurisé.
- RFID, Radio Frequency Identification.
- Protection des données.
- E-commerce, le commerce electronique.
- Protection de l'identité.
- La signature électronique.
- Secure Wireless, la télécommunication sans fil et les connexions wifi sécurisées.
- L'authentication biométrique.
- Vehicular Communications, la communication directe vers les véhicules.
- Anti-Counterfeiting, la protection des produits industriels et pharmaceutiques contre les contrefaçons.

Références

[1] W. D. Banks and I. Shparlinski, A variant of NTRU with non-invertible polynomials. INDOCRYPT, 2002

[2] M. Coglianese, B.-M. Goi, MaTRU : A new NTRU-based cryptosystem, ACISP Progress in Cryptology - INDOCRYPT 2005 : 6th International Conference on Cryptology in India 2005, Bangalore, India, Springer-Verlag, 2005.

[3] H. Cohen, A Course in Computational Number Theory, Graduate Texts in Mathematics, Springer, 1993.

[4] D. Coppersmith and A. Shamir, Lattice attacks on NTRU. In Advances in cryptology—EUROCRYPT '97, volume 1233 of Lecture Notes in Comput. Sci., pages 52–61. Springer, Berlin, 1997.

[5] W. Diffie, E. Hellman, New Directions in Cryptography, IEEE Trans- actions on Information Theory, 22, 5 (1976), pp. 644–654.

[6] T. El Gamal, A public key cryptosystem and signature scheme based on discrete logarithms. IEEE Transactions on Information Theory IT-31, 496-473,1976.

[7] P. Gaborit, J. Ohler, P. Solé, CTRU, a polynomial analogue of NTRU, Rapport de Recherche, INRIA, 2002, no RR-4621 `http://www.inria.fr/rrrt/rr-4621.html`

[8] J. Hoffstein, J. Pipher, and J. H. Silverman, NTRU : A Ring Based Public Key Cryptosystem in Algorithmic Number Theory. Lecture Notes in Computer Science 1423, Springer-Verlag, pages 267–288, 1998.

[9] IEEE P1363.1 Public-Key Cryptographic Techniques Based on Hard Problems over Lattices, June 2003. IEEE.

[10] N. Koblitz, Elliptic curve cryptosystems, Mathematics of Computation 48, 1987, pp. 203–209

[11] R. Kouzmenko, Generalizations of the NTRU cryptosystem, Diploma Project, Winter Semester 2005-2006

[12] A.K. Lenstra, H.W. Lenstra and L. Lovasz, Factoring polynomials with rational coefficients, Mathematische Annalen, Vol. 261, 513—534, 1982.

[13] E. Malekian, A. Zakerolhosseini and A. Mashatan, Qtru : A Lattice Attack Resistant Version of Ntru,`http://eprint.iacr.org/2009/386`.

[14] V.S. Miller, Use of elliptic curves in cryptography, CRYPTO 85, 1985.

[15] M. Nevins, C. KarimianPour and A. Miri, NTRU over rings beyond $\mathbb{Z}$, Designs, Codes and Cryptography, August 2009. DOI : 10.1007/s10623-009-9342-7

[16] NTRU Inc. `http://www.ntru.com/`

[17] NTRU Inc. `http://www.ntru.com/cryptolab/faqs.htm#six`

[18] R. Rivest, A. Shamir, L. Adleman, A Method for Obtaining Digital Signatures and Public-Key Cryptosystems, Communications of the ACM, Vol. 21 (2), 120—126 (1978)

[19] P.W. Shor, Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, SIAM J. Computing 26, pp. 1484-1509 (1997).

[20] N. Vats, Algebraic Cryptanalysis of CTRU Cryptosystem, COCOON 2008, Dalian, China, Springer, 2008.