La cryptographie post-quantique

Lilian Pattier (patl21039604)

12 décembre 2018

Introduction

Les méthodes cryptographiques actuellement en place pour protéger les systèmes d'informations remplissent parfaitement leur tâche aujourd'hui. Que ce soit en cryptographie symétrique ou asymétrique, avec des algorithmes tels que l'AES, le 3DES ou encore le RSA, les échanges peuvent se faire de manière relativement sécurisée (à condition d'implémenter correctement les protocoles) en garantissant le respect des concepts fondamentaux de la cryptographie : authenticité, intégrité, confidentialité, non-répudiation.

Ces algorithmes sont souvent basés sur la création de clés simples à générer mais mathématiquement et informatiquement difficiles à inverser. Par exemple dans le cas de RSA, la création de la clé repose sur le choix de deux grands nombres premiers p et q dont on va calculer le produit n. Or, l'opération inverse est difficilement réalisable, connaissant uniquement n. Ainsi, les ordinateurs « classiques » ne sont pas en mesure de casser ce genre de chiffrements aujourd'hui.

Cependant, avec l'amélioration des technologies de l'information et l'apparition de l'informatique quantique, certains algorithmes pourraient potentiellement être menacés dans le futur.

On se propose d'étudier une branche de la cryptographie qui pourrait s'avérer pertinente face aux problèmes cités précédemment : la cryptographie post-quantique. Il s'agit du domaine visant à sécuriser l'information face à un attaquant qui disposerait d'une force de calcul quantique. On étudiera globalement les enjeux de cet aspect, puis on détaillera les grandes classes d'algorithmes répondant à cette problématique.

1 Motivations de la cryptographie post-quantique

1.1 État de l'art

De nos jours, la cryptographie à clé publique est utilisée dans de nombreux systèmes d'échanges (Commerce électronique, échange d'informations privées). Elle tire son avantage de la non-nécessité d'utiliser un canal secret pour pratiquer l'échange de la clé publique (contrairement à la cryptographie symétrique). Le récepteur doit uniquement s'assurer de l'authenticité de cette dernière. Elle permet aussi de réduire considérablement le nombre de clés en jeu lors d'un échange : Il suffit de $2n$ clés pour que n acteurs puissent communiquer entre eux par cryptographie asymétrique, contre $\sum_{k=1}^{n-1} k = \frac{n(n-1)}{2}$ clés secrètes dans le cadre d'un cryptosystème symétrique.

En contre-partie de ces propriétés, la cryptographie à clé publique est globalement moins performante que la cryptographie symétrique et requiert des clés plus longue pour jouir du même niveau de sécurité. Pour toutes ces raisons, un système sécurisé classique va la plupart du temps, avoir recours à ces deux modes de chiffrement : la cryptographie asymétrique sera utilisée pour l'authentification de données (signatures) ainsi que pour la distribution d'une clé de cryptage symétrique, clé qui sera utilisée par la suite pour sécuriser des échanges par chiffrement symétrique.

La cryptographie à clé publique repose sur des problèmes mathématiquement difficiles à résoudre. Par exemple, le chiffrement RSA repose sur la difficulté de décomposer des grands nombres en produit de facteurs premiers, tandis que le protocole d'échange de Diffie-Hellman repose sur la difficulté de déterminer un logarithme discret dans certains corps fini.

Les deux exemples de protocoles cryptographiques précédemment cités ne sont qu'un échantillon d'un grand ensemble de méthodes reposant sur les mêmes propriétés mathématiques (chiffrements de ElGamal, DSA, Blum - Goldwasser...) Cela induit un problème de taille : si l'on parvient à remettre en cause la difficulté de ces problèmes mathématiques d'une quelconque manière (innovation technologique, avancée mathématique...), cela rend désuet tout type de communications utilisant ces protocoles.

Et c'est justement la menace qu'encourent ces protocoles avec l'arrivée des ordinateurs quantiques. À ce stade, il est encore tôt pour craindre un quelconque risque (immédiat) dans la sécurisation des échanges. Nous n'en sommes qu'aux prémices de l'informatique quantique, les technologies actuellement en place ne permettent pas de casser les cryptosystèmes actuels de manière efficace et pratique. Mais sur le long terme, et en prenant en considération les avancées et recherches dans ce domaine, il n'est pas impossible d'envisager la nécessité d'un basculement des protocoles pour pallier à cet hypothétique problème. Il est à noté qu'en 2015, la NSA postait un communiqué recommandant de préparer la transition de la cryptographie à clé publique vers la cryptographie post-quantique, probable signe avant-coureur d'une révolution du monde de la cryptographie dans les années à venir.

1.2 Les ordinateurs quantiques

Détaillons un petit peu les enjeux d'une puissance de calcul quantique dans le problème de la cryptographie asymétrique. Les ordinateurs quantiques sont des calculateurs qui utilisent les propriétés quantiques de la matière (principe de superposition, intrication quantique) dans le but d'effectuer des opérations sur ces données. Ainsi, alors qu'un ordinateur classique dont l'architecture primaire est basée sur des transistors et où les données sont représentées par des bits pouvant prendre 2 valeurs (0 ou 1), un calculateur quantique travaille sur des *qubits* dont l'état quantique peut posséder une infinité de valeurs. Détaillons un petit peu ce dernier point :

Un système quantique est mathématiquement représenté par la somme des systèmes potentiellement mesurables pondérés d'amplitudes de probabilité. Un qubit est composé d'une superposition de deux états nommés conventionnellement $|0\rangle$ et $|1\rangle$ et que l'on peut décrire par une combinaison linéaire

$$\alpha\,|0\rangle + \beta\,|1\rangle$$

tels que $|\alpha|^2 + |\beta|^2 = 1$ avec α et β des nombres complexes. On peut alors en théorie transmettre une infinité d'informations en les stockant dans l'angle de polarisation du qubit, mais il est impossible de récupérer l'information directement lors de la lecture. (En physique quantique, la mesure définit l'état). Ainsi après une mesure, le qubit est projeté dans *l'état mesuré*.

Là où l'ordinateur quantique offre de grandes possibilité, c'est par la combinaison de qubits : l'état de plusieurs qubits n'est pas simplement une combinaison des états respectifs de chaque qubit (comme ce serait le cas en informatique classique). Deux qubits réunis sont dans une superposition d'états :

$$\alpha\,|00\rangle + \beta\,|01\rangle + \gamma\,|10\rangle + \delta\,|11\rangle$$

avec $|\alpha|^2+|\beta|^2+|\gamma|^2+|\delta|^2 = 1$. En d'autres termes, il s'agit d'une superposition des 4 états élémentaires des qubits. C'est la seconde grande propriété de la physique quantique : *l'intrication*. En conséquence, la puissance de calcul théorique d'un ordinateur quantique double à chaque fois que l'on ajoute un qubit.

L'enjeu est alors de concevoir des algorithmes tels que toutes les propriétés quantiques soient utilisées pour le calcul, les qubits devant à la fin de l'exécution se trouver dans un état donnant le résultat du calcul sans risque d'obtenir un résultat aléatoire.

C'est justement dans cette optique que Shor a proposé un algorithme capable de résoudre les problèmes de décomposition en facteurs premiers et de logarithmes discrets en temps polynomial sur une machine quantique.

1.3 Algorithme de Shor

L'algorithme de Shor pose un problème de taille dans le monde de la cryptographie. La conséquence d'un algorithme de résolution des problèmes mathématiques précédemment cités en temps polynomial est sans appel : d'un point

de vue quantique, RSA et DIFFIE-HELMAN ne sont théoriquement pas plus sécurisés qu'un simple chiffrement de CÉSAR. Détaillons cet algorithme :

Tout d'abord, par nature de l'ordinateur quantique (induite par la structure des qubits), l'algorithme de SHOR est probabiliste : il donnera la réponse correcte avec une forte probabilité. L'algorithme est composé de deux parties. La première partie, classique, transforme le problème de factorisation en un problème de recherche de période d'une fonction. La seconde partie, quantique, détermine cette période en utilisant la transformée de FOURIER quantique.

Pour calculer la période, la fonction est évaluée en tous ses points simultanément, en utilisant justement les propriétés de superpositions du calculateur quantique. Il est à noté que l'on ne peut pas accéder à toute l'information directement. Comme mentionné précédemment, une mesure déterminera une valeur possible. Par conséquent, l'enjeu est de passer de la superposition en un état qui retournera la réponse correcte avec une haute probabilité. C'est justement ce qui est accompli par la transformée de FOURIER quantique.

Détaillons maintenant un deuxième algorithme quantique qui pourrait compromettre la sécurité des systèmes cryptographiques : l'algorithme de GROVER.

1.4 L'algorithme de Grover

L'algorithme de GROVER est une menace aux cryptosystèmes symétriques : il permet d'effectuer une recherche d'un élément dans une structure de données de n éléments avec une complexité temporelle en $\mathcal{O}(n^{\frac{1}{2}})$ (accélération quadratique de la recherche exhaustive par rapport à un algorithme classique). Par exemple, une attaque force-brute sur un chiffrement AES avec une taille de clé de 128 bits nécessite d'énumérer dans le pire des cas 2^{128} combinaisons pour un algorithme classique. Dans le cas d'un calculateur quantique implémentant l'algorithme de GROVER, il ne faudrait que 2^{64} énumérations. Bien que l'impact paraisse moins critique, il n'en reste pas moins conséquent. La cryptographie post-quantique se concentrant principalement sur la recherche de solutions aux problèmes de chiffrement à clé publique, nous ne détaillerons pas ici le fonctionnement de l'algorithme de GROVER.

Pour conclure cette partie, nous devons garder à l'esprit que même s'il n'existe pas à ce jour de calculateur quantique permettant d'implémenter ces algorithmes de manière pratique, ces derniers permettent néanmoins de résoudre de manière conceptuelle les problèmes mathématiques sur lesquels reposent la sécurité de nos cryptosystèmes. La menace reste encore faible, mais suffisamment percutante pour que de nombreux chercheurs se penchent sur l'implémentation d'algorithmes de chiffrement résistants aux calculateurs quantiques : les algorithmes post-quantiques.

2 Cryptographie multivariée

2.1 Problèmes NP-Complets

En théorie de la complexité, on regroupe les problèmes de décision dans différentes classes en fonction de leur « difficulté » :

— La classe NP (*nondeterministic polynomial*) : ensemble des problèmes dont une solution peut être vérifiée en temps polynomial
— La classe P (*polynomial*) : ensemble des problèmes résolubles en temps polynomial
— La classe BQP (*bounded error quantum polynomial time*) : ensemble des problèmes résolubles en temps polynomial par un calculateur quantique

Par exemple le problème de factorisation d'un entier n est supposé dans BQP mais en dehors de P.

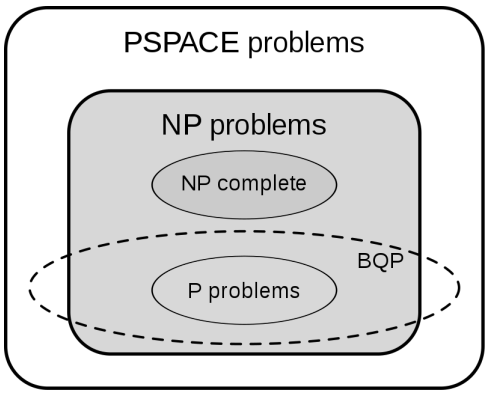

FIGURE 1 – Représentation des classes de complexité

On définit également les problèmes :

— NP-difficile : au moins aussi difficile que n'importe quel autre problème de NP (Tout problème de NP peut se ramener à un problème NP-difficile)
— NP-complet : problème à la fois dans NP et NP-difficile.

Le but de la cryptographie post-quantique est d'élaborer des algorithmes dont la sécurité repose sur des problèmes NP-complets (ou au moins NP-difficile). Sous l'hypothèse $P \neq NP$, un calculateur quantique n'aura aucun avantage sur un ordinateur classique pour résoudre ces problèmes.

2.2 Principe général de la cryptographie multivariée

Le but de la cryptographie multivariée est de construire des cryptosystèmes qui reposent sur la difficulté du problème mathématique consistant à trouver un zéro commun d'un ensemble de polynômes non-linéaires.

Ce problème étant NP-difficile, il n'est pas remis en cause par l'émergence des calculateurs quantiques.

Le but est de se placer dans un corps fini $\mathbb{F}$. La clé publique est donnée par un systèmes de polynômes :

$$P(x_1, ..., x_n) = (P_1(x_1, ..., x_n), ..., P_m(x_1, ..., x_n))$$

avec $P_i \in \mathbb{F}[x_1, ..., x_n], \forall i \in \{1, ..., m\}$

Un message est représenté par un n-uplet $M = (m_1, ..., m_n) \in \mathbb{F}^n$. Le message chiffré est simplement l'évaluation de chaque polynôme de la clé publique en M :

$$C = P(M) = (P_1(M), ..., P_m(M))$$

L'intérêt de cette famille est la rapidité du chiffrement : l'évaluation polynomiale est très efficace d'un point de vue de la complexité.

Le déchiffrement repose sur l'inversion du polynôme P. La clé secrète est alors P^{-1}. En général, la façon d'inverser le polynôme est propre au cryptosystème utilisé.

2.3 Cryptosystèmes d'Imai-Matsumoto

Détaillons un exemple de cryptosystème multivarié. Dans ce cadre, nous allons nous placer dans un corps fini à $q = p^2$ éléments, avec p premier. Ce corps peut se construire par extension quadratique du corps $\mathbb{F}_p$.

Considérons ici le corps fini $\mathbb{F}_2 = \mathbb{Z}/2\mathbb{Z}$ (il s'agit bien d'un corps car 2 est premier). Le polynôme $P(X) = X^2 + X + 1$ est irréductible dans $\mathbb{F}_2[X]$ (l'ensemble des polynômes à coefficient dans $\mathbb{F}_2$). On réalise alors l'extension quadratique de $\mathbb{F}_2$ en quotientant $\mathbb{F}_2[X]$ par l'idéal engendré par P, donnant ainsi le corps :

$$\mathbb{F}_4 = \mathbb{F}_2[X]/(P)$$

On dit que $\mathbb{F}_4$ est un corps de rupture (et même ici le corps de décomposition) de P.

Ce corps dispose de 4 éléments : $0, 1, \alpha$ et $1 + \alpha$ où α est la classe de X modulo l'idéal (P) et ses lois sont les suivantes :

$+$	0	1	α	$1+\alpha$
0	0	1	α	$1+\alpha$
1	1	0	$1+\alpha$	α
α	α	$1+\alpha$	0	1
$1+\alpha$	$1+\alpha$	α	1	0

$\times$	0	1	α	$1+\alpha$
0	0	0	0	0
1	0	1	α	$1+\alpha$
α	0	α	$1+\alpha$	1
$1+\alpha$	0	$1+\alpha$	1	α

Nous avons spécifié dans quel corps nous travaillons, et les lois qu'il induit.

Détaillons maintenant quelques outils mathématiques nécessaires au chiffrement de Imai-Matsumoto. Posons n le nombre de caractères que nous voulons chiffrer. Considérons également $g(X)$ un polynôme irréductible de $\mathbb{F}_4[X]$ et de degré n. De la même manière que précédemment on peut construire un corps fini K par extension : $K = \mathbb{F}_4[X]/(g)$. Définissons maintenant ϕ, l'isomorphisme de K dans $\mathbb{F}_4^n$ donné par :

$$\phi(a_0 + a_1 x + ... + a_{n-1}x^{n-1}) = (a_0, a_1, ..., a_{n-1})$$

où x désigne la classe de X modulo (g).

Choisissons θ tel que $0 < \theta < n$ et $q^\theta + 1$ est premier avec $q^n - 1$ (ici $q = 4$) et t tel que $t \times (1 + q^\theta) \equiv 1 \mod (q^n - 1)$. On définit alors l'application $\tilde{F}$ sur K par :

$$\tilde{F}(X) = X^{1+q^\theta}$$

On a :

$$\tilde{F}^{-1}(X) = X^t$$

Finalement, choisissons deux transformations affines inversibles L_1 et L_2 sur $\mathbb{F}_4^n$.

On note :

$$F(x_1, ..., x_n) = L_1 \circ \phi \circ \tilde{F} \circ \phi^{-1} \circ L_2 = (f_1, ..., f_n)$$

Les éléments étant définis, spécifions ce qui constituent les clé publique et privée du cryptosystème.

La clé publique se compose :
— du corps choisi initialement ainsi que de ses lois (ici il s'agit de $\mathbb{F}_4$).
— des polynômes $f_1, ..., f_n$.
La clé privée est constituée de L_1 et L_2 et $\tilde{F}$.

Détaillons maintenant le chiffrement et le déchiffrement d'un message :

En considérant un message $(m_1, ..., m_n)$, le message chiffré correspondant est $(c_1, ..., c_n)$ où :

$$c_i = f_i(m_1, ..., m_n)$$

pour $i = 1, ..., n$.

Pour le déchiffrement, en partant d'un message $(c_1, ..., c_n)$, on calcule :

$$F^{-1}(c_1, ..., c_n) = L_2^{-1} \circ \phi \circ \tilde{F}^{-1} \circ \phi^{-1} \circ L_1^{-1}(c_1, ..., c_n)$$

2.3.1 Application :

On se place dans le corps $\mathbb{F}_4$ défini précédemment. Pour simplifier les notations on notera $x = \alpha$ et $1 + x = \beta$. Ainsi $\mathbb{F}_4 = \{0, 1, \alpha, \beta\}$ muni des lois définies précédemment. Choisissons $n = 3$ et $g(X) = X^3 + X + 1$ qui est bien irréductible sur $\mathbb{F}_4[X]$.

Choisissons $\theta = 2$. ($4^2 + 1 = 17$ est premier avec $4^3 - 1 = 63$). Choisissons $t = 26$ (On a bien $26 \times 17 \equiv 1 \mod 63$)

L'application $\tilde{F}$ est donnée par :

$$\tilde{F}(X) = X^{4^2+1} = X^{17} \quad \text{donc} \quad \tilde{F}^{-1}(X) = X^{26}$$

Définissons les transformations L_1 et L_2 :

$$L_1(x_1, x_2, x_3) = \begin{pmatrix} \beta & \alpha & \alpha \\ \alpha & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \\ \alpha \end{pmatrix}$$

et

$$L_2(x_1, x_2, x_3) = \begin{pmatrix} 1 & 0 & \alpha \\ 0 & 1 & \alpha \\ 1 & \alpha & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} \alpha \\ \beta \\ \beta \end{pmatrix}$$

Calculons $F(x_1, ..., x_2)$:

$$\phi^{-1} \circ L_2(x_1, x_2, x_3) = (\alpha + x_1 + \alpha x_3) + (\beta + x_2 + \alpha x_3)x + (\beta + x_1 + \alpha x_2)x^2$$

Calculons $\tilde{F}(X) = X^{17}$, comme nous travaillons dans le groupe fini, il n'y a pas de puissance supérieure à 2.

Ainsi :

$\tilde{F} \circ \phi^{-1} \circ L_2(x_1, x_2, x_3) = 1 + \beta x_1 + \alpha x_2 + x_3 + x_1x_2 + \alpha x_1x_3 + \beta x_2x_3 + (\alpha + \alpha x_1 + x_2 + \beta x_3 + x_1^2 + \beta x_1x_2 + x_2^2 + x_2x_3)x + (\beta + \beta x_1 + \alpha x_2 + \alpha x_3 + x_1^2 + x_1x_2 + \alpha x_1x_3 + \beta x_2^2 + \alpha x_2x_3 + \beta x_3^2)x^2$

Calculons finalement $L_1 \circ \phi(X)$ pour déterminer les polynômes :

$$f_1(x_1, x_2, x_3) = 1 + x_3 + \alpha x_1x_3 + \beta x_2^2 + \beta x_2x_3 + x_3^2$$
$$f_2(x_1, x_2, x_3) = 1 + \beta x_1 + \alpha x_2 + x_3 + x_1^2 + x_1x_2 + \beta x_1x_3 + x_2^2$$
$$f_3(x_1, x_2, x_3) = \beta x_3 + x_1^2 + \beta x_2^2 + x_2x_3 + \beta x_3^2$$

Nous pouvons maintenant chiffrer un texte.

Prenons $(m_1, m_2, m_3) = (1, \alpha, \beta)$

On a alors :

$$c_1 = f_1(1, \alpha, \beta) = 0$$

$$c_2 = f_2(1, \alpha, \beta) = 0$$
$$c_3 = f_3(1, \alpha, \beta) = 1$$

Le texte chiffré est $(c_1, c_2, c_3) = (0, 0, 1)$.

Maintenant, supposons qu'une personne disposant de la clé privée veuille déchiffrer le texte $(0, 0, 1)$. Cette personne dispose de $L_1^{-1}, \tilde{F}^{-1}$ et L_2^{-1}.

$$L_1^{-1}(c_1, c_2, c_3) = \begin{pmatrix} \beta & 1 & 1 \\ 1 & \beta & \alpha \\ \beta & 1 & 0 \end{pmatrix} \begin{pmatrix} c_1 - 0 \\ c_2 - 1 \\ c_3 - \alpha \end{pmatrix}$$

et donc :

$$L_1^{-1}(0, 0, 1) = \begin{pmatrix} 1 + \beta \\ \beta + \alpha(1 + \alpha) \\ 1 \end{pmatrix} = \begin{pmatrix} \alpha \\ \alpha \\ 1 \end{pmatrix}$$

il s'ensuit :

$$X = \phi^{-1} \circ L_1^{-1}(0, 0, 1) = \alpha + \alpha x + x^2$$

puis :

$$\tilde{F}^{-1}(X) = X^{26} = \alpha + x^2$$

Ensuite :

$$\phi \circ \tilde{F}^{-1}(X) = (\alpha, 0, 1)$$

Finalement :

$$L_2^{-1}(y_1, y_2, y_3) = \begin{pmatrix} \beta & \beta & \alpha \\ \alpha & \alpha & \alpha \\ 1 & \alpha & 1 \end{pmatrix} \begin{pmatrix} y_1 - \alpha \\ y_2 - \beta \\ y_3 - \beta \end{pmatrix}$$

d'où :

$$L_2^{-1}(\alpha, 0, 1) = \begin{pmatrix} \beta^2 + \alpha^2 \\ \alpha\beta + \alpha^2 \\ \alpha\beta + \alpha \end{pmatrix} = \begin{pmatrix} 1 \\ \alpha \\ \beta \end{pmatrix}$$

On obtient bien le message envoyé initialement.

Cela conclut notre présentation du chiffrement d'Imai-Matsumoto. En théorie, la cryptanalyse a montré qu'il existe des attaques très efficaces contre ce cryptosystème. Il ne serait donc probablement pas utilisé en remplacement des algorithmes actuels mais d'autres variantes reposant sur les mêmes grandes lignes offrent de bonnes performances et une sécurité efficace. On peut notamment citer les variantes HFE.

Enfin, un point qui semble important de souligner ici : le corps de base choisi dans notre exemple est un corps à 4 éléments $(0, 1, \alpha, \beta)$ ce qui n'a pas beaucoup de pertinence au vue du format actuel des données codées sur des bits $(0, 1)$. Nous aurions donc très bien pu travailler sur le corps $\mathbb{Z}/2\mathbb{Z}$ mais nous avons explicité le fonctionnement sur un corps plus « étendu » pour justifier le fonctionnement de l'algorithme.

3 Cryptographie par codes correcteurs d'erreurs

La cryptographie basée sur les codes correcteurs d'erreurs est une autre solution considérée comme robuste aux calculateurs quantiques. Elle est d'ailleurs très prometteuse car les cryptosystème proposés dans ce domaine sont relativement anciens et ont résisté à toutes les tentatives de cryptanalyse à ce jour.

Détaillons le fonctionnement général de cette branche de la cryptographie post-quantique : tout d'abord, un code correcteur d'erreurs permet de corriger une information potentiellement altérée lors de sa transmission. On utilise pour cela la redondance : ajout d'information à la fin du mot transmis. À la réception, la redondance est utilisée pour corriger les potentielles erreurs induites lors de la transmission et retrouver le mot réellement transmis.

Le but de la cryptographie basée sur les codes et de volontairement ajouter des erreurs dans le message à transmettre tout en offrant la possibilité au destinataire de les corriger.

De cette façon, un tiers ne disposant pas de la méthode permettant de corriger les erreurs ne pourra pas lire le message volontairement altéré. La méthode d'encodage peut néanmoins être laissée publique si elle ne révèle pas d'information sur la méthode de décodage.

Le cryptosystème le plus populaire à ce jour est le cryptosystème de McEliece que nous détaillons dans la partie suivante. Il repose sur une famille de codes pouvant être facilement camouflés et devenir indistinguables d'un code aléatoire. La sécurité repose alors sur la difficulté de décoder un code linéaire aléatoire (NP-difficile). La personne qui connait la structure peut cependant utiliser l'algorithme de décodage adéquat et retrouver le message en clair.

3.1 Définitions mathématiques

Soit p un nombre premier, d une puissance de p, n un entier strictement positif et k un entier inférieur à n. Soit $\mathbb{F}_d$ un corps fini à d élements. Un code linéaire C de dimension k et de longueur n est un sous-espace vectoriel de $\mathbb{F}_d^n$ de dimension k.

On muni notre code de la distance de Hamming : pour deux mots $x = x_0...x_{n-1}$ et $y = y_0...y_{n-1}$ de $\mathbb{F}_d^n$, la distance de Hamming entre x et y est définie par :

$$d(x, y) = \text{Card}\{i, x_i \neq y_i\}$$

(Il s'agit grossièrement du nombre d'endroits ou x et y diffèrent)

On note δ la distance minimum d'un code C donné. δ correspond à la plus courte distance de Hamming possible entre deux mots du code distincts.

Un code linéaire de dimension k et de longueur n est appelé un (n, k)-code. Si la distance δ est spécifiée, on parle de (n, k, δ)-code.

Le nombre maximum d'erreurs corrigibles t d'un code C est le plus grand entier strictement inférieur à $\delta/2$.

Un code C dispose d'une application dite application d'encodage. Il s'agit d'une application linéaire injective φ de $\mathbb{F}_d^k$ dans $\mathbb{F}_d^n$ qui permet de munir la redondance au message souhaité.

On appelle matrice génératrice du code C la matrice G vérifiant :

$$\forall x \in \mathbb{F}_d^k,\ x.G = \varphi(x) \in C$$

On appelle matrice de contrôle de C la matrice H vérifiant :

$$\forall x \in \mathbb{F}_d^k,\ H.{}^t x = 0 \Leftrightarrow x \in C$$

Dans toute la suite, l'alphabet considéré est le corps fini $\mathbb{F}_2 = \mathbb{Z}/2\mathbb{Z} = \{0, 1\}$ des entiers modulos 2. On parle alors de code binaire.

3.2 Cryptosystème de McEliece

Le cryptosystème de McEliece dispose de trois algorithmes : un algorithme qui génère une clé secrète et une clé publique, un algorithme de chiffrement et un algorithme de déchiffrement. Seul le dernier algorithme est déterministe. Les paramètres de sécurité n, k et t sont connus de tous.

3.2.1 Génération des clés

1. On choisit un (n, k)-code linéaire C capable de corriger t erreurs et disposant d'un algorithme de décodage.
2. On génère une matrice génératrice G (de dimension $k \times n$) pour ce code.
3. On génère une matrice binaire aléatoire S inversible de dimension $k \times k$. (matrice de brouillage)
4. On génère une matrice de permutation aléatoire P de dimension $n \times n$. (matrice avec un et un seul 1 par ligne, un et un seul 1 par colonne et le reste à 0).
5. On calcule la matrice $\hat{G} = SGP$ de taille $k \times n$.

La clé publique est $(\hat{G}, t)$ et la clé privée est (S, G, P).

3.2.2 Chiffrement

On cherche à chiffrer un message binaire m de longueur k.

1. on calcule le vecteur $c' = m\hat{G}$.
2. on génère un vecteur d'erreurs e de poids t : ie. un mot aléatoire de longueur n contenant t 1 et $n - t$ 0.
3. on calcule le message chiffré $c = c' + e$

3.2.3 Déchiffrement

1. on calcule $\hat{c} = cP^{-1}$
2. on utilise l'algorithme de décodage de C pour décoder $\hat{c}$ en $\hat{m}$.
3. On calcule $m = \hat{m}S^{-1}$

On peut aisément s'assurer du fonctionnement du cryptosystème. On a :

$$cP^{-1} = (c' + e)P^{-1} = (mSGP + e)P^{-1} = mSG + eP^{-1}$$

mSG est un mot du code C choisis et eP^{-1} est une erreur de poids t (une permutation conserve le poids) donc en appliquant la fonction de déchiffrement du code C, on obtient mS. Il suffit alors de multiplier par S^{-1} pour avoir le message.

Généralement les cryptosystèmes de McEliece sont implémentés avec une famille spécifique de codes dit "codes de Goppa". Ce choix est justifié par leur capacité à corriger beaucoup d'erreurs (jusqu'à $\binom{n^k}{\ln(n)}$) et leurs algorithmes efficaces de décodage.

3.2.4 Exemple concret

Nous allons détailler un exemple de la procédure de McEliece pour le chiffrement/déchiffrement d'un message. Nous n'utiliserons pas ici le code de Goppa, dont nous n'avons pas présenté le fonctionnement, mais bien le code de Hamming déja introduit en cours.

On travaille sur le $(7, 4)$-code de Hamming. On rappelle que pour ce code, $\delta = 3$.

Les primitives sont ainsi : $n = 7, k = 4, t = 1$ (1 erreur simple corrigée).

On rappelle également la matrice génératrice G associée au code :

$$G = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

On génère une matrice binaire aléatoire S inversible de dimension 4×4 :

$$S = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \end{pmatrix}$$

On génère une matrice de permutation aléatoire P de dimension 7×7 :

$$P = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

On calcule $\hat{G}$:

$$\hat{G} = SGP = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 \end{pmatrix}$$

Alice veut chiffrer le message $m = 1011$. Elle calcule $c' = m\hat{G}$.

$$c' = \begin{pmatrix} 1011 \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 0011101 \end{pmatrix}$$

Puis, elle génère un vecteur d'erreurs e de poids $t = 1$ qu'elle somme avec c' :

$$c = \begin{pmatrix} 0011101 \end{pmatrix} + \begin{pmatrix} 0000100 \end{pmatrix} = \begin{pmatrix} 0011001 \end{pmatrix}$$

Maintenant supposons que Bob veuille déchiffrer le message. Il calcule $\hat{c} = cP^{-1}$. (On note au passage que $P^{-1} = {}^tP$, P étant une matrice de permutation : elle appartient au groupe orthogonal $O(7, \mathbb{R})$)

$$\hat{c} = cP^{-1} = \begin{pmatrix} 0011001 \end{pmatrix} \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 0110100 \end{pmatrix}$$

Bob utilise alors l'algorithme de décodage de Hamming : Il génère H, la matrice de contrôle de C. On remarque que G est de la forme $\begin{pmatrix} I_4 & M \end{pmatrix}$. H est alors de la forme $\begin{pmatrix} {}^tM & I_3 \end{pmatrix}$. Il calcule donc $\hat{c}\,{}^tH$:

$$\hat{c}\,{}^tH = \begin{pmatrix} 0110100 \end{pmatrix} \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 010 \end{pmatrix}$$

L'erreur se trouve en position 6 : le code corrigé est $\tilde{c} = \big(0110110\big)$.

Bob peut alors décoder $\hat{m} = \tilde{c}R$ avec $R = {}^t\big(I_4 \quad O\big)$

$$\hat{m} = \big(0110110\big)\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} = \big(0110\big)$$

Finalement il calcule $m = \hat{m}S^{-1}$:

$$m = \big(0110\big)\begin{pmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix} = \big(1011\big)$$

On retrouve bien le message envoyé par Alice.

4 Cryptographie fondée sur les réseaux euclidiens

La cryptographie fondée sur les réseaux euclidiens repose sur la difficulté mathématique de trouver une solution à plusieurs problèmes concernant la-dite structure mathématique. Les algorithmes de chiffrement (voire de signature) sont le plus souvent basés sur les problèmes de recherche SVP et CVP (respectivement *Shortest Vector Problem* et *Closest vector problem*), à savoir la recherche du vecteur le plus cours possible pour la norme euclidienne dans un réseau donné, et la recherche du vecteur le plus proche d'une cible donnée dans un réseau. Ces problèmes issus de la théorie des réseaux ont permis l'élaboration d'algorithmes populaires comme celui que nous allons détailler : NTRUEncrypt.

Le lien entre les problèmes SVP et CVP et l'algorithme NTRUEncrypt n'est pas forcément trivial, nous détaillerons ce point en dernière partie.

4.1 Outils mathématiques

Soit N un entier positif. On se place dans l'anneau quotient des polynômes : $\mathcal{P} = \mathbb{Z}[X]/(X^N - 1)$. Tout élement f de $\mathcal{P}$ s'écrit donc sous la forme :

$$f = \sum_{i=0}^{N-1} f_i X^i$$

La loi additive de $\mathcal{P}$ est l'addition usuelle des polynômes, la loi multiplicative est le produit de convolution $*$:

Soient $f, g, h \in \mathcal{P}$ tels que $f * g = h$. Alors $h = \sum_{i=0}^{N-1} h_k X^k$ avec pour tout $k \in \{0, ..., N-1\}$:

$$h_k = \sum_{i+j=k \mod N} f_i g_j$$

Pour un entier positif q, on note $\mathcal{P}_q$ l'anneau $\mathcal{P}$ réduit modulo q :

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

où $\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$.

La correspondance entre $\mathcal{P}$ et $\mathcal{P}_q$ se fait assez instinctivement : pour un polynôme f dans $\mathcal{P}$, le polynôme de $\mathcal{P}_q$ correspondant est le polynôme de coefficients $f_i \mod q$.

Un polynôme $f \in \mathcal{P}$ est dit inversible modulo q s'il existe un polynôme f_q tel que $f * f_q \equiv 1 \mod q$.

Soit d un entier positif on note $\mathcal{B}(d)$ l'ensemble :

$$\mathcal{B}(d) = \left\{ f = \sum_{i=0}^{N-1} f_i X^i \in \mathcal{P}_2, \ \sum_{i=0}^{N-1} f_i = d \right\}$$

C'est l'ensemble des polynômes de $\mathbb{Z}_2[X]/(X^N - 1)$ qui ont exactement d coefficients égaux à 1 et le reste des coefficients nuls.

Considérons maintenant les entiers p, q, d_f, d_g, d_r et les espaces $\mathcal{L}_f, \mathcal{L}_g, \mathcal{L}_m, \mathcal{L}_r$ tels que :

- $p \wedge q = 1$ et $p << q$.
- $\mathcal{L}_f = \mathcal{B}(d_f)$ et $\mathcal{L}_g = \mathcal{B}(d_g)$ sont des espaces de polynômes pour les clés privées.
- $\mathcal{L}_m = \mathbb{Z}_p[X]/(X^N - 1)$ est l'ensemble des messages en clair.
- $\mathcal{L}_r = \mathcal{B}(d_r)$ est un ensemble de polynôme auxiliaires.

4.2 Implémentation de l'algorithme

4.2.1 Génération des clés

1. Choisir aléatoirement un polynôme $f \in \mathcal{L}_f$ inversible dans $\mathcal{P}$ modulo p et modulo q.
2. Calculer l'inverse f_q de f modulo q (ie. tel que $f * f_q \equiv 1 \mod q$)
3. Calculer l'inverse f_p de f modulo p (ie. tel que $f * f_p \equiv 1 \mod p$)
4. Choisir aléatoirement un polynôme $g \in \mathcal{L}_g$.
5. Calculer $h \equiv g * f_q \mod q$

La clé publique est alors h. On partage également les paramètres N, p, q et les 4 espaces. La clé privée est (f, f_p).

4.2.2 Chiffrement

1. Choisir un message à chiffrer, le représenter comme un polynôme $m \in \mathcal{L}_m$
2. Choisir un polynôme aléatoire $r \in \mathcal{L}_r$
3. Utiliser la clé publique h pour chiffrer le message comme suit :
   $e \equiv p * r * h + m \mod q$.

4.2.3 Déchiffrement

1. Le récepteur dispose de la clé privée (f, f_p). Calculer $a \equiv f * e \mod q$.
2. Transformer a en un polynôme avec coefficients dans l'intervalle $[-q/2, q/2[$.
3. Déchiffrer le message en calculant $m \equiv f_p * a \mod p$

4.2.4 Preuve de fonctionnement

On a :

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

Si l'on translate les coefficients de $a \in \mathcal{P}_q$ dans l'intervalle $[-q/2, q/2[$ et que $a \in \mathcal{P}$ est également à coefficient dans le même intervalle, on a alors : $a \in \mathcal{P}_q = a \in \mathcal{P}$. Dans ce cas le chiffrement fonctionne :

$$a * f_p \equiv (p * r * g * f_p) + (f * f_p) * m \equiv m \mod p$$

L'égalité $(a \in \mathcal{P}_q = a \in \mathcal{P})$ est assurée par la condition $p << q$

4.2.5 Application

On considère les paramètres simplifiés suivants :

$N = 7$, $p = 2$, $q = 41$, $\mathcal{L}_f = \mathcal{L}_g = \mathcal{L}_r = \mathcal{B}(3)$:

Supposons que Alice veuille envoyer le message 1100101 à Bob. Elle le convertit en un message dans $\mathcal{L}_m = \mathbb{Z}_2[X]/(X^7 - 1)$. Le message est alors codé par le polynôme

$$m = X^6 + X^5 + X^2 + 1$$

Les coefficients à 1 représentent les bits à 1 dans le message.

Bob va alors générer les clés publiques et privées. Il commence par générer un $f \in \mathcal{L}_f$ aléatoire et inversible modulo p et q.

$$f = X^4 + X^3 + 1$$

(Le critère d'inversibilité est assuré par $\text{PGCD}(f, X^7 - 1) = 1$).

Puis il va générer les inverses f_q modulo q et f_p modulo p de f :

$$f_q = 13X^6 + 14X^5 + 14X^4 + 14X^3 + 14X^2 + 13X + 14$$
$$f_p = X^5 + X^4 + X^3 + X^2 + 1$$

Ces inverses peuvent être calculés en pratique à l'aide de l'algorithme d'Euclide étendu.

Bob génère un second polynôme aléatoire $g \in \mathcal{L}_g$:

$$g = X^6 + X^4 + X$$

Pour finir il calcule $h \equiv g * f_q \mod q$:

$$\begin{aligned} h &\equiv (X^6 + X^4 + X)(13X^6 + 14X^5 + 14X^4 + 14X^3 + 14X^2 + 13X + 14) \mod q \\ &\equiv 42X^6 + 40X^5 + 42X^4 + 41X^3 + 41X^2 + 42X + 40 \mod q \\ &\equiv X^6 - X^5 + X^4 + X - 1 \mod q \end{aligned}$$

Il transmet alors la clé publique h à Alice et garde la clé privée (f, f_p).

Alice va maintenant chiffrer son message m pour l'envoyer à Bob. Elle génère un polynôme aléatoire $r \in \mathcal{L}_r$:

$$r = X^5 + X^2 + 1$$

Puis elle calcule $e \equiv p * r * h + m \mod q$:

$$\begin{aligned} e &\equiv 2(X^5 + X^2 + 1)(X^6 - X^5 + X^4 + X - 1) + X^6 + X^5 + X^2 + 1 \mod q \\ &\equiv 7X^6 - 3X^5 + 4X^4 + X^2 + 4X - 3 \mod q \end{aligned}$$

Elle envoie alors le message chiffré e à Bob.

Bob reçoit e. Il utilise sa clé privée (f, f_p) pour déchiffrer le message. Il calcule d'abord $a \equiv f * e \mod q$:

$$\begin{aligned} a &\equiv (X^4 + X^3 + 1)(7X^6 - 3X^5 + 4X^4 + X^2 + 4X - 3) \mod q \\ &\equiv 8X^6 + 2X^5 + 5X^4 + 4X^3 + 5X^2 + 5X + 1 \mod q \end{aligned}$$

Les coefficients de a sont bien dans l'intervalle [-20,20[

Il calcule enfin $m \equiv f_p * a \mod p$:

$$\begin{aligned} m &\equiv (X^5 + X^4 + X^3 + X^2 + 1)(8X^6 + 2X^5 + 5X^4 + 4X^3 + 5X^2 + 5X + 1) \mod p \\ &\equiv 27X^6 + 17X^5 + 24X^4 + 20X^3 + 21X^2 + 24X + 17 \mod p \\ &\equiv X^6 + X^5 + X^2 + 1 \mod p \end{aligned}$$

Il retrouve bien le message original d'Alice $m = X^6 + X^5 + X^2 + 1$ qu'il peut traduire en binaire : 1100101

4.3 Lien avec les réseaux euclidiens

La sécurité de NTRUEncrypt repose sur le problème mathématique CVP (recherche du vecteur du réseau le plus proche d'un vecteur donné). Il s'agit d'un problème NP-difficile.

Posons au préalable quelques définitions. Soit une famille de vecteurs linéairement indépendants $u_1, ..., u_k$ de $\mathbb{R}^n$. On note L le réseau engendré par ces vecteurs :

$$L = \left\{ \sum_{i=1}^{k} a_i u_i,\ a_i \in \mathbb{Z} \right\}$$

On dit que $(u_1, ..., u_k)$ est une base de L. Un réseau peut admettre plusieurs bases et on passe d'une base à une autre par une matrice de passage à coefficients entiers.

On considère la norme euclidienne $||\,.\,||$ sur $\mathbb{R}^n$ définie par :

$$||x|| = \sqrt{\sum_{i=1}^{n} x_i^2}$$

Le problème du CVP est le suivant : étant donné une base B d'un réseau L et un vecteur $v \notin L$, trouver le vecteur $u \in L$ le plus proche de v au sens de la norme euclidienne (ie. tel que $||u - v||$ soit minimale).

NTRUEncrypt est basé sur le réseau :

$$M_{h,q} = \{(u, v) \in \mathcal{P},\ v \equiv u * h \mod q\}$$

La génération des clés consiste donc à créer ce réseau à partir de h et tel que le vecteur (f, g) soit choisi aléatoirement dans le réseau. Comme $h \equiv g * f_q$, on a : $h * f \equiv g * (f_q * f) \mod q$ d'où

$$g \equiv f * h \mod q$$

donc $(f, g) \in M_{h,q}$.

Le chiffrement revient à créer un point qui n'appartient pas au réseau mais qui est suffisamment proche d'un point du réseau. On choisit alors un point aléatoire du réseau, le point $(2 * r,\ 2 * r * h \mod q)$ qu'on somme ensuite avec $(0, m)$ ce qui nous donne le point $(2 * r, e)$ avec $e \equiv 2 * r * h + m \mod q$.

Déchiffrer un message revient alors à résoudre un problème CVP.

5 Conclusion

Ainsi s'achève notre tour d'horizon non-exhaustif des branches de la cryptographie post-quantique sur lesquelles de nombreux chercheurs se penchent aujourd'hui. D'autres approches non mentionnées ici existent comme par exemple la cryptographie basée sur les isogénies de courbes supersingulières, ou encore la cryptographie basée sur les fonctions de hachage.

Malgré l'aspect un peu déconcertant des calculateurs quantiques sur la résolution de problèmes, les cryptosystèmes de nos jours restent néanmoins sécuritaires. À moins d'une avancée technologique ou mathématique cruciale, il est encore trop tôt pour craindre un renversement des algorithmes de chiffrement actuels par la puissance de calcul quantique.

Cependant, les méthodes « post-quantiques » offrent parfois de bonnes performances et certaines sont même déja utilisées dans le chiffrements de systèmes (Par exemple, NTRU dans l'industrie des services financiers)