Cryptographie fondée sur les codes : nouvelles approches pour construction et preuves ; contribution en cryptanalyse

Thomas Debris-Alazard [1]

Thomas Debris-Alazard a soutenu sa thèse [2] opérée au sein d'Inria Paris sous la direction de Jean-Pierre Tillich le 17 décembre 2019 à Sorbonne Université.

Contexte de la thèse

La sécurité de nos données personnelles, de nos communications ou encore de nos échanges bancaires, en bref notre sécurité numérique n'est possible qu'au prix d'une protection : *la cryptographie*. Les menaces contre lesquelles elle nous protège sont légion et ne cessent d'évoluer, que ce soit avec les nouvelles habitudes d'utilisateurs ou l'amélioration des moyens techniques. Il est donc nécessaire d'analyser et d'adapter en permanence la cryptographie.

La cryptographie se scinde en deux grands domaines. Le premier est la cryptographie à *clef secrète*. Les entités souhaitant s'échanger des messages utilisent à la façon d'un mot de passe un même secret partagé préalablement. Mes travaux se sont concentrés sur le second paradigme dit cryptographie *asymétrique ou à clef publique*. Ici, seule l'entité avec laquelle nous souhaitons communiquer possède un secret. Les échanges

1. thomas.debris@inria.fr.

2. https://hal.archives-ouvertes.fr/tel-02926286.

DOI:10.48556/SIF.1024.17.131 **1024** – Bulletin de la SiF, numéro 17, Avril 2021, pp. 131–140

sont possibles avec le détenteur du secret grâce à des données publiques qui lui sont liées. Ce type de cryptographie est particulièrement important car il permet, entre autre, l'échange préalable d'un secret (dit échange de clefs) permettant la cryptographie à clef secrète.

La cryptographie à clef publique repose sur l'utilisation de problèmes « difficiles » de type question/réponse là où la question joue le rôle de donnée publique, tandis que la réponse est le secret. Il doit donc être facile de calculer une question pour une réponse donnée (calculer ses données publiques à partir de son secret), alors qu'il doit être difficile de trouver la réponse d'une question (personne ne peut calculer votre secret à partir de vos données publiques). Ce genre de problème est particulièrement naturel et commun dans nos vies. Prenons par exemple un annuaire : il est facile de trouver le numéro de M. Halliday alors qu'il est « difficile », étant donné un numéro, de retrouver son détenteur. Malheureusement, ce problème ne peut être utilisé en cryptographie. En effet, tout détenteur d'un ordinateur peut parcourir très rapidement l'annuaire pour retrouver une personne à partir de son numéro de téléphone. C'est ici que les mathématiques interviennent en nous offrant des problèmes tels que, même avec toute la puissance de calcul disponible sur terre, aucun ordinateur ne sera en mesure d'offrir une résolution.

C'est en 1978 que naquit le premier système cryptographique à clef publique (chiffrement) avec la proposition de Rivest, Shamir et Adleman [16]. La sécurité de ce cryptosystème repose sur le problème de la factorisation issu de la théorie des nombres : étant donné $n = pq$ où p et q sont premiers, retrouver p et q. Cette même année 1978, une seconde proposition fut faite mais reposant, quant à elle, sur l'utilisation d'objets appelés *codes correcteurs d'erreurs* : le schéma de McEliece [13]. Ce système, bien que jouissant de certains avantages comparativement à RSA, ne fut pas utilisé en pratique. En effet, les données publiques nécessaires à RSA sont bien plus petites. Ce dernier fut jugé à l'époque plus performant. De nos jours, ce système ainsi que le protocole d'échange de clefs de Diffie-Hellman [8] sont toujours les plus utilisés en pratique. Le protocole de [8] est, entre autres, lié au problème dit du *logarithme discret* : étant donné un groupe $\mathbb{G}$ de générateur g et g^x, retrouver x.

La cryptographie à clef publique déployée repose donc uniquement sur deux problèmes. Or, ces-derniers ne sont pas si éloignés, ils sont tout deux une instance du problème dit du sous-groupe caché [11] dans un groupe abélien. Il n'existe pas, à ce jour, d'algorithme fonctionnant sur nos ordinateurs qui résolve efficacement ce problème. En revanche, la situation est bien différente dans un monde où l'ordinateur « quantique » existerait. Depuis la découverte de Shor [17], nous connaissons un algorithme « quantique » résolvant le problème du sous-groupe caché dans un groupe abélien. Les protocoles de sécurité [16] et [8] seront donc caducs une fois que les premiers ordinateurs quantiques de taille suffisante auront été construits. Face à ce danger, il fut, durant plusieurs années, invoqué l'infaisabilité d'un tel ordinateur.

C'était sans compter sur les récents progrès techniques qui aujourd'hui rendent probable l'existence d'un ordinateur quantique efficace dans les prochaines décennies. Google a même récemment prétendu avoir obtenu la suprématie quantique [2], c'est-à-dire être en mesure de résoudre en temps raisonnable avec un ordinateur quantique un problème hors d'atteinte pour des ordinateurs classiques. Ce résultat est cependant contesté par IBM [3]. Quoi qu'il en soit, le temps se fait pressant pour trouver et étudier de nouveaux paradigmes mathématiques pour lesquels la cryptographie à clef publique sera quantiquement sûre. En effet, certaines données ont vocation à être confidentielles pour les cinquante prochaines années.

Fort heureusement, nous connaissons des solutions cryptographiques fonctionnant sur ordinateur classique et qui ont des chances d'être sûres quantiquement. On parle usuellement de *cryptographie post-quantique*. Le cryptosystème de McEliece est, par exemple, revenu sur le devant de la scène après la découverte de Shor. De nos jours, les seules attaques connues contre le système de McEliece requièrent une puissance de calcul (même quantique) exponentielle en sa taille. Il existe actuellement bien d'autres propositions ; nous pouvons citer parmi les plus prometteuses celles reposant sur les réseaux euclidiens, les fonctions de hachage, les systèmes multi-variés ou plus récemment les isogénies. Nous changeons donc d'ère cryptographique ; les problèmes sur lesquels peut reposer la sécurité numérique sont tout aussi bien variés que de natures différentes. Néanmoins, ces domaines cryptographiques ont été bien moins étudiés que ne le furent RSA [16] et le protocole de Diffie et Hellman [8]. Or, il y a aujourd'hui urgence à proposer des systèmes sûrs aussi bien classiquement que quantiquement.

C'est dans ce contexte que le *National Institute of Standard Technology* (NIST) du gouvernement américain lança en 2017 [4] un appel pour la standardisation de systèmes à clef publique sûrs contre un ordinateur quantique. Cet appel se focalise sur deux fonctionnalités cruciales pour le fonctionnement d'internet : les échanges de clefs et les signatures numériques. Les signatures permettent de « signer » des messages. De cette façon, nous sommes sûrs, d'une part de l'émetteur du message, et d'autre part que ce message n'a pas été altéré par une partie tierce. Cette fonctionnalité est, par exemple, primordiale lors de nos mises à jour logiciel : nous voulons nous assurer que la mise à jour vient bien de notre fournisseur et que personne n'y a intégré de virus. Ce dernier nous fournit donc des mises à jour signées.

Ma thèse se déroula dans ce cadre d'étude de la cryptographie post-quantique à clef publique avec comme tâche de fond la standardisation du NIST dont le second tour à démarré peu de temps avant ma soutenance. Je me suis tout particulièrement intéressé à la branche cryptographique née de la proposition de McEliece. La figure 1 donne les paradigmes mathématiques qui étaient encore présents au NIST lors du second tour de standardisation. Nous constatons qu'aucune signature utilisant des

3. https://www.ibm.com/blogs/research/2019/10/on-quantum-supremacy/
4. https://csrc.nist.gov/projects/post-quantum-cryptography/

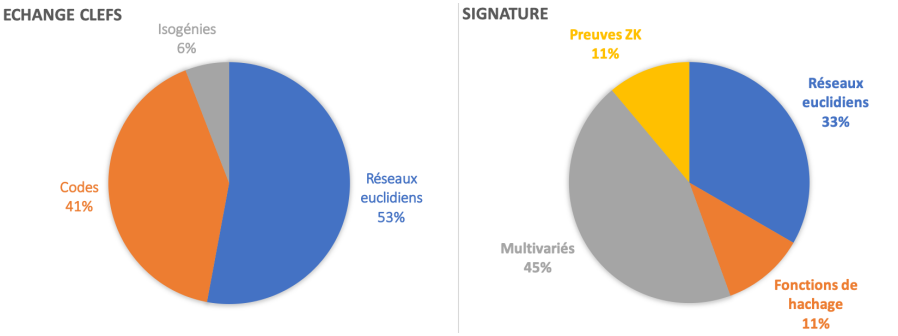


FIGURE 1. Propositions au second tour du NIST

codes correcteurs n'était présente alors que les codes sont l'une des plus vieilles propositions post-quantiques. Il existe en effet d'importants problèmes techniques pour construire des signatures et tout particulièrement avec des codes (problème ouvert par McEliece lui-même dans son article fondateur [13]). Mes travaux se sont entre autres consacrés à la résolution de ce problème ouvert depuis 40 ans.

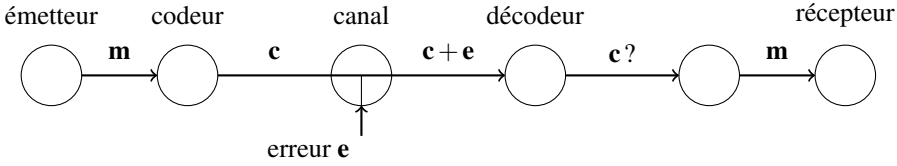
Solutions cryptographiques avec des codes correcteurs

Des télécommunications aux codes correcteurs

La cryptographie à clef publique est, comme nous l'avons vu, en quête de problèmes difficiles. Une source particulièrement prolifique pour cette dernière fut l'un des grands tournants de notre époque : la numérisation de l'information. Revenons un instant sur ce point. La numérisation ouvrit la possibilité de la conservation illimitée de l'information tout comme notre capacité à la transmettre quasi-instantanément. En revanche, ceci ne fut possible qu'au prix de la protection contre les erreurs. En effet, toute donnée enregistrée sur un support (pensons à nos vieux CD-ROM) ou téléchargée d'un serveur à l'autre bout du monde est susceptible d'être altérée. Le principe pour s'en prémunir est alors simple et naturel : adjoindre de la redondance à chaque symbole que l'on souhaite transmettre ou sauvegarder. Une illustration banale est lorsque nous cherchons à épeler notre nom au téléphone : T comme Thierry, I comme Inès, L comme Léo, etc.

Dans un contexte numérique où nous souhaitons envoyer des bits, l'idée est la même : ajouter de la redondance. Partons d'un message que l'on souhaite transmettre, à savoir une suite $\mathbf{m}$ formée de k bits. La première idée est de se donner un sous-espace vectoriel $\mathscr{C}$ de dimension k de $\{0,1\}^n$ avec $n > k$ ($\{0,1\}^n$ représente

l'ensemble des mots de n bits). Un tel espace $\mathscr{C}$ est appelé *code linéaire*. Ce dernier étant de dimension k, nous pouvons le représenter à l'aide d'une base, c'est à dire une matrice $\mathbf{G}$ de dimension $k \times n$. Désormais pour transmettre $\mathbf{m}$, on commence par l'encoder : $\mathbf{c} \triangleq \mathbf{mG}$. L'encodé $\mathbf{c}$ (mot de code) représente $\mathbf{m}$ mais avec $n-k$ bits de redondance. Notons qu'un simple calcul d'algèbre linéaire permet de retrouver $\mathbf{m}$ à partir de $\mathbf{c}$. Le mot de code $\mathbf{c}$ est ensuite transmis à travers un canal pouvant induire des erreurs (des rayures sur notre CD-ROM...). Nous obtenons alors un mot de la forme $\mathbf{c}+\mathbf{e}$. Il s'agit maintenant de retrouver $\mathbf{c}$, et donc $\mathbf{m}$, à partir de $\mathbf{c}+\mathbf{e}$. Cette opération est appelée *décodage*. La figure qui suit résume la situation.



Le seul instant non-maîtrisé ici est l'action du « canal » sur le mot de code. Une première modélisation simple est la suivante : tout bit de $\mathbf{c}$ est indépendamment modifié en son opposé avec une probabilité $0 \leqslant p < 1/2$. Dans ce modèle, une donnée naturelle quantifiant les erreurs possibles est *le poids de Hamming*. Ce dernier est défini comme le nombre de bits non nuls d'un vecteur. Étant donné $\mathbf{x} \in \{0,1\}^n$: $|\mathbf{x}| = \#\{1 \leqslant i \leqslant n : x_i \neq 0\}$. On vérifie facilement qu'une erreur donnée $\mathbf{e}$ de poids de Hamming w (w bits du mot transmis ont été modifiés) a pour probabilité d'apparition $p^w(1-p)^{n-w}$ lors de la transmission. Du fait que $p < 1/2$, cette probabilité est décroissante en w. On en déduit que le mot de code transmis est d'autant plus probable qu'il est proche, *au sens de la métrique de Hamming*, du mot reçu. *Le décodage consiste alors à retrouver le mot de code à la plus petite distance de Hamming du mot reçu*. On parle usuellement de décodage par maximum de vraisemblance. La procédure naïve consistant à énumérer les 2^k mots de code possibles et sélectionner le plus proche est évidemment à écarter. Il nous faut donc ajouter une structure sur le code dont on pourra ensuite tirer parti.

L'ambition de la théorie des codes a alors été de proposer des familles de codes avec une « structure » permettant un algorithme de décodage efficace. Historiquement deux grandes familles furent proposées : (i) les codes munis d'une forte structure algébrique et d'un décodage déterministe tels que les codes de Reed-Solomon [12, chapitre 10] ou les codes de Goppa [12, chapitre 12] et (ii) ceux avec des algorithmes de décodage probabiliste tels que les codes convolutifs [9], les LDPC [10] ou plus récemment les polaires [1] (qui sont d'ailleurs les codes utilisés pour la 5G). Si toutes ces structures ont été introduites, c'est que le problème de décodage d'un code quelconque, c'est à dire sans structure particulière, semble *difficile*, ce que confirment près de soixante années de recherche.

Des codes correcteurs à la cryptographie

McEliece [13] eut alors l'idée d'utiliser ce problème difficile de décodage d'un code quelconque dans un contexte de cryptographie à clef publique.

Supposons que Bob souhaite communiquer à Alice un message $\mathbf{m}$ de k bits. L'idée est alors la suivante : Alice commence par choisir son code préféré $\mathscr{C}$ de dimension k *qu'elle sait décoder*. En d'autres termes, si Alice reçoit $\mathbf{c}+\mathbf{e}$, elle peut retrouver $\mathbf{c} \in \mathscr{C}$ si l'erreur $\mathbf{e}$ est de petit poids de Hamming ($\mathbf{c}$ est le mot de code le plus proche de $\mathbf{c}+\mathbf{e}$). Alice rend alors public son code $\mathscr{C}$, en fournissant à qui le souhaite une base de ce code, c'est à dire une matrice $\mathbf{G} \in \{0,1\}^{k\times n}$. Bob qui souhaite maintenant envoyer $\mathbf{m}$ commence par l'encoder $\mathbf{c} \stackrel{\triangle}{=} \mathbf{mG}$. Bob ajoute ensuite lui-même une erreur $\mathbf{e}$ (de petit poids de Hamming) à ce mot de code puis envoie à Alice le résultat $\mathbf{c}+\mathbf{e}$ qu'elle peut décoder. Alice retrouve donc $\mathbf{c}$ et ainsi le message $\mathbf{m}$. Désormais, si Eve (une personne malveillante...) intercepte les communications entre Alice et Bob elle se retrouve avec $\mathbf{c}+\mathbf{e}$. Eve doit donc savoir décoder le code $\mathscr{C}$ pour retrouver $\mathbf{m}$. C'est ici qu'intervient l'une des idées de McEliece. Alice a choisi son code préféré qu'elle sait décoder mais ce choix doit être précautionneux. *Alice doit être la seule à pouvoir décoder le code*. Pour cela, elle va choisir un code structuré mais elle va cacher cette structure de façon à ce qu'en rendant son code public, ce dernier semble quelconque. Ainsi, même si Eve connaît le code $\mathscr{C}$, elle ne connaît pas la structure permettant le décodage. Elle se retrouve à devoir résoudre le problème de décodage d'un code quelconque, problème difficile. Eve est donc incapable de retrouver le message envoyé par Bob.

Dans le schéma de McEliece, une question cruciale est de savoir quelle structure utiliser, comment bien la cacher et s'en assurer. La théorie des codes est d'une grande richesse. Il y a aujourd'hui pléthore de familles de codes avec un bon algorithme de décodage. Ces codes proviennent des télécommunications. Nous pouvons donc apparemment instancier ce schéma de multiples façons. L'histoire des codes en cryptographie a en revanche démontré à quel point nous devons nous méfier. En effet, l'immense majorité des propositions fut brisée. De nos jours, les familles robustes aux attaques (retrouver la structure cachée) ne sont plus légion. Nous pouvons, par exemple, citer les codes de Goppa (proposition originelle de McEliece) et les codes MDPC [14].

Revenons un instant sur l'introduction des codes MDPC, ces derniers étant symptomatiques d'une idée profonde en théorie des codes et cryptographie. Nous savons, depuis l'introduction des codes LDPC [10], que des mots creux (*i.e* de petit poids de Hamming) dans un code sont d'une grande aide pour le décodage. Les codes LDPC sont aujourd'hui très utilisés en télécommunication et nous pourrions être tentés de les utiliser en cryptographie. Malheureusement, il est trop facile de retrouver des mots creux dans un code, ces mots sont impossibles à cacher. Le rationnel des codes MDPC a alors consisté à augmenter le poids de ces mots. Ceci a pour premier effet

de détériorer l'algorithme de décodage. Il serait donc absurde d'utiliser ces codes dans un contexte de télécommunication. En revanche, le décodage d'un petit nombre d'erreurs est toujours possible et, désormais, il n'y a plus de mots creux dans le code. On peut donc cacher la structure tout en sachant encore décoder.

Contributions de cette thèse

Les travaux de ma thèse se sont inscrits dans ce contexte de cryptographie avec des codes correcteurs, que ce soit à travers des analyses de la difficulté algorithmique du problème de décodage [6, 3], des attaques [7] ou encore la proposition d'un schéma de signature avec des codes Wave [5]. La signature Wave s'est faite en rupture de l'approche classique en cryptographie avec des codes, qui utilise le décodage à petite distance. Nous avons avec Wave introduit une nouvelle notion originale et nouvelle : le décodage à grande distance (*i.e* rechercher le mot de code *le plus éloigné*). Cette idée n'a aucun sens dans un contexte de télécommunication mais en revanche, comme je l'ai montré, ce nouveau paradigme (clef du bon fonctionnement de Wave) a un grand intérêt cryptographique.

Signer avec des codes

Une signature avec des codes repose sur le même principe que le chiffrement de McEliece que nous avons décrit précédemment. Cependant, cette approche soulève des difficultés techniques supplémentaires. Supposons qu'Alice souhaite « signer » une suite de n bits $\mathbf{y}$ à envoyer à Bob. Alice commence par choisir encore une fois son code préféré $\mathscr{C}$ qu'elle sait décoder et qu'elle rend public. Pour signer $\mathbf{y}$, Alice va le décoder pour trouver un mot de code $\mathbf{c} \in \mathscr{C}$ proche de $\mathbf{y}$, disons à distance w. Cette distance w est aussi rendue publique par Alice. La signature de $\mathbf{y}$ est alors $\mathbf{c}$. Bob en recevant le couple message/signature $(\mathbf{y}, \mathbf{c})$ s'assure alors d'une part que $\mathbf{c} \in \mathscr{C}$ et d'autre part que $|\mathbf{y} - \mathbf{c}| = w$. Si Alice est la seule à savoir décoder $\mathscr{C}$ à distance w, Bob peut être sûr qu'Alice a bien envoyé le message $\mathbf{y}$. Il y a cependant ici une grosse difficulté. La théorie des codes offre des structures permettant à partir de $\mathbf{c}+\mathbf{e}$ de retrouver $\mathbf{e}$ de petit poids de Hamming. Or, Alice souhaite ici signer un message $\mathbf{y}$ quelconque et, pour des raisons combinatoires, il est extrêmement peu probable que $\mathbf{y}$ soit égal à un $\mathbf{c}+\mathbf{e}$ si $\mathbf{e}$ est de poids de Hamming trop petit et $\mathbf{c} \in \mathscr{C}$. De ce fait, *décoder* pour signer dans ce paradigme est très contraignant. En effet, du fait que le mot à décoder est quelconque, la condition sur le décodage devient : presque tout mot de l'espace ambiant $\{0,1\}^n$ peut se décoder efficacement. Or, le nombre attendu de mots de code à distance w d'un vecteur $\mathbf{y}$ passe selon w de (i) exponentiellement élevé (ii) à zéro. Plus précisément, c'est dans une fenêtre très étroite autour du poids dit de « Gilbert-Vashamov » w_{GV} que nous nous attendons typiquement à un unique mot de code tandis que, pour des distances inférieures, il n'y aura aucun mot de code. Or, c'est précisément dans ces zones de distance faible qu'ont été développés

les algorithmes de décodage. Il semble donc difficile de suivre une approche à la McEliece avec des codes correcteurs pour fabriquer une signature.

Une des contributions de la première signature avec des codes [4] a été de remarquer qu'en contraignant suffisamment les paramètres des codes de Goppa (les codes proposés par McEliece), et plus précisément en choisissant leur dimension proche de leur longueur (*i.e :* $\mathscr{C} \approx \{0,1\}^n$), ces derniers sont en mesure de décoder à une distance légèrement inférieure à w_{GV}. La densité des mots « décodables » est alors suffisamment proche de 1 pour tenter de décoder $\mathbf{y}$. Malheureusement le système de [4] souffre d'un problème de passage à l'échelle. Du fait que le code $\mathscr{C}$ est de grande dimension, le décodage d'un code quelconque n'est plus un problème si difficile que cela et CFS est une signature impraticable.

Trouver le mot le plus loin

L'idée naturelle pour se défaire d'un tel carcan est de relâcher la contrainte d'un algorithme de décodage *stricto sensu* et donc l'existence d'un unique mot de code à la bonne distance, comme cela fut décrit pour la première fois dans [15]. Le paradigme n'est plus le même, on choisit une distance w telle que pour tout mot de l'espace ambiant il existe un nombre exponentiel de mots de code à cette distance fixée w. On cherche alors à retrouver efficacement pour tout mot de l'espace ambiant *l'un des mots de codes* à distance w. Il ne s'agit plus formellement de décodage mais plutôt de distorsion comme décrit en théorie des codes. Le souci étant maintenant que peu de familles de codes viennent avec un algorithme efficace pour résoudre ce problème. Nous pouvons citer les codes convolutifs ou encore les codes LDGM [18], mais ces-derniers ne résistent pas aux attaques, tout comme la suggestion des codes polaires [15].

Bien que non-sûrs car trop structurés, les codes polaires furent les prémisses de la proposition des codes $(U, U+V)$ généralisés en signature avec Wave. Ces codes correspondent à des codes polaires à « un étage ». Un code polaire étant un code $(U, U+V)$ où U et V sont eux-mêmes des codes $(U, U+V)$ et ainsi de suite... Cette structure récursive permet un algorithme de décodage extrêmement efficace mais elle est impossible à cacher. En revanche, en suivant la même idée que les codes MDPC qui sont une version dégradée des codes LDPC, une structure à un étage peut se cacher tout en continuant à offrir une capacité de décodage.

Les codes $(U, U+V)$ permettent de trouver efficacement $\mathbf{c} \in \mathscr{C}$ à distance w de $\mathbf{y}$, quel que soit le mot $\mathbf{y}$. En revanche, ceci est possible pour des distances w telles qu'il existe typiquement de nombreux mots de code à cette distance. Cela pose alors problème car notre algorithme va devoir choisir un de ces mots de code. Il se peut alors que ce choix soit biaisé et dépende trop fortement de la « géométrie » du code. Ceci est problématique car toute personne malveillante peut collectionner plusieurs signatures, les étudier pour ensuite retrouver la structure cachée du code. On parle alors de *fuite d'information*. Le principe, pour se prémunir de ce grave défaut à moindre coût,

est alors complexe. Il faut filtrer ce que produit l'algorithme pour supprimer le biais. On parle usuellement de *méthode de rejet* (*rejection sampling*). L'objectif étant que notre algorithme produise $\mathbf{c} \in \mathscr{C}$ tel que $\mathbf{y} - \mathbf{c}$ soit uniforme parmi tous les mots de poids de Hamming w, assurant par définition l'indépendance entre les signatures et la structure cachée du code. Nous avons avec la signature Wave réussi à proposer un tel algorithme pour signer. En revanche, et de façon surprenante, cela fut possible en introduisant un nouveau paradigme du décodage. Avec Wave, nous travaillons dans le corps à trois éléments $\{0,1,2\}$ et, pour un mot $\mathbf{y}$ nous le « décodons » en trouvant un mot de code $\mathbf{c}$ *très éloigné*. Ceci fut surprenant et contre-intuitif car ce problème de décodage en grande distance n'a aucun sens en télécommunication. Cependant, dans un contexte cryptographique, ce problème offre bien plus de degrés de liberté, ce qui s'est avéré crucial pour mettre en place la méthode de rejet. Le décodage en grande distance, nouveau paradigme semble alors, comme discuté dans cette thèse, très bien adapté à la cryptographie.

Références

[1] Erdal Arıkan. Channel polarization : a method for constructing capacity-achieving codes for symmetric binary-input memoryless channels. *IEEE Trans. Inform. Theory*, 55(7) :3051–3073, 2009.

[2] Frank Arute, Kunal Arya, Ryan Babbush, Dave Bacon, Joseph C Bardin, Rami Barends, Rupak Biswas, Sergio Boixo, Fernando GSL Brandao, David A Buell, et al. Quantum supremacy using a programmable superconducting processor. *Nature*, 574(7779) :505–510, 2019.

[3] Rémi Bricout, André Chailloux, Thomas Debris-Alazard, and Matthieu Lequesne. Ternary syndrome decoding with large weights. In *Selected Areas in Cryptography - SAC 2019 - 26th International Conference, Waterloo, ON, Canada, August 12-16, 2019, Revised Selected Papers*, pages 437–466, February 2019.

[4] Nicolas Courtois, Matthieu Finiasz, and Nicolas Sendrier. How to achieve a McEliece-based digital signature scheme. In *Advances in Cryptology - ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 157–174, Gold Coast, Australia, 2001. Springer.

[5] Thomas Debris-Alazard, Nicolas Sendrier, and Jean-Pierre Tillich. Wave : A new family of trapdoor one-way preimage sampleable functions based on codes. In *Advances in Cryptology - ASIACRYPT 2019*, LNCS, Kobe, Japan, December 2019.

[6] Thomas Debris-Alazard and Jean-Pierre Tillich. Statistical decoding. preprint, January 2017. arXiv :1701.07416.

[7] Thomas Debris-Alazard and Jean-Pierre Tillich. Two attacks on rank metric code-based schemes : Ranksign and an identity-based-encryption scheme. In *Advances in Cryptology - ASIACRYPT 2018*, volume 11272 of *LNCS*, pages 62–92, Brisbane, Australia, December 2018. Springer.

[8] Whitfield Diffie and Martin Hellman. New directions in cryptography. *IEEE transactions on Information Theory*, 22(6) :644–654, 1976.

[9] Peter Elias. oding for noisy channels. *IRE conv. Rec.*, 3 :37, 1955.

[10] Robert G. Gallager. *Low Density Parity Check Codes*. M.I.T. Press, Cambridge, Massachusetts, 1963.

[11] Richard Jozsa. Quantum factoring, discrete logarithms, and the hidden subgroup problem. *Computing in Science and Engineering*, 3(2) :34–43, 2001.

[12] Florence J. MacWilliams and Neil J. A. Sloane. *The Theory of Error-Correcting Codes*. North–Holland, Amsterdam, fifth edition, 1986.

[13] Robert J. McEliece. *A Public-Key System Based on Algebraic Coding Theory*, pages 114–116. Jet Propulsion Lab, 1978. DSN Progress Report 44.

[14] Rafael Misoczki, Jean-Pierre Tillich, Nicolas Sendrier, and Paulo S. L. M. Barreto. MDPC-McEliece : New McEliece variants from moderate density parity-check codes, 2012.

[15] Ayoub Otmani and Jean-Pierre Tillich. On the Design of Code-Based Signatures. In *Code-based Cryptography Workshop (CBC 2012)*, Lyngby, Denmark, May 2012.

[16] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2) :120–126, 1978.

[17] Peter W. Shor. Algorithms for quantum computation : Discrete logarithms and factoring. In S. Goldwasser, editor, *FOCS*, pages 124–134, 1994.

[18] Martin J. Wainwright, Elitza N. Maneva, and Emin Martinian. Lossy source compression using low-density generator matrix codes : analysis and algorithms. *IEEE Trans. Information Theory*, 56(3) :1351–1368, 2010.