Rapport de stage de Master 2 MPRI : Calcul quantique délégué à l'aveugle par un client classique

Léo Colisson, supervisé par Céline Chevalier,
équipe CASCADE (CRYPTO) ENS Ulm

12/03/2018 – 31/07/2018

Le contexte général

La découverte de la physique quantique a révélé un nouveau modèle de calcul (*l'ordinateur quantique*) qui a permis l'apparition d'algorithmes permettant la résolution de problèmes réputés difficiles auparavant (l'exemple le plus emblématique étant la factorisation d'entiers par l'algorithme de Shor [Sho95], qui est un problème central en cryptographie). Bien que l'ordinateur quantique n'est pour l'instant qu'un modèle théorique, les progrès récents de la physique ont permis l'apparition de petits ordinateurs quantiques travaillant sur quelques dizaines de qubits (bits quantiques) que certaines sociétés mettent à disposition du public [IBM].

Le problème étudié

Afin de permettre à de potentiels utilisateurs de ces ordinateurs de ne pas dévoiler leurs calculs, il devient intéressant de se demander s'il est possible de déléguer des algorithmes quantiques sur un serveur quantique distant "à l'aveugle", c'est à dire sans révéler au serveur les calculs effectués (afin de cacher non seulement les entrées, les sorties, et l'algorithme utilisé). Un protocole est déjà connu [BFK08] si le client peut disposer d'un petit ordinateur quantique ainsi que d'un canal quantique entre lui et le serveur quantique, et d'autres variantes on été découvertes, mais jusqu'à récemment toutes nécessitaient des hypothèses relativement fortes, notamment d'avoir deux serveurs quantiques qui ne communiquent pas mais qui partagent un état quantique particulier. Lors de mon précédent stage, j'ai introduit avec Alexandru Cojocaru, Petros Wallden et Elham Kashefi un nouveau protocole *QFactory* [CCKW18] permettant de résoudre ces problèmes. Cependant, la construction nécessitait une fonction cryptographique bien particulière, et la proposition de notre papier originel s'est avérée non fonctionnelle. Le premier travail de mon stage a donc été de trouver un candidat pour cette fonction,

puis dans un second temps d'étudier la preuve de sécurité de notre protocole dans le cas où le serveur pouvait être totalement malhonnête.

La contribution proposée

L'idée de base de QFactory est de simuler un canal quantique entre le client classique et le serveur quantique. Afin de ne pas révéler d'informations au serveur, l'idée est d'utiliser une fonction cryptographique particulière qui est (entre autre) difficile à inverser pour le serveur, mais simple à inverser pour le client. Étant donné que notre fonction doit résister à un adversaire quantique, nous avons choisi de baser notre candidat sur un problème réputé sécurisé y compris contre un adversaire quantique nommé LWE (Learning With Errors), et plus particulièrement sur le crypto-système développé par Micciancio et Peikert [MP11]. Cependant, notre fonction doit également être 2-vers-1 (2-to-1), et pour y arriver nous avons du essayer de rendre ce cryptosystème (presque) homomorphe, en exploitant le fait que LWE est sécurisé même avec un bruit faible.

Les arguments en faveur de sa validité

J'ai explicitement programmé cette fonction pour m'assurer que tout fonctionnait. Nous avons de plus une preuve de sécurité formelle pour les propriétés principales de notre fonction, et notre papier QFactory a été accepté récemment à la conférence en cryptographie quantique *QCrypt*. De plus, quelques semaines après notre première publication, un article indépendant a été publié et il obtient un résultat similaire au notre, avec une méthode relativement proche sur l'idée fondamentale.

Le bilan et les perspectives

La découverte de cette fonction permet de compléter notre protocole, qui est l'un des premiers protocoles permettant de réaliser du calcul à l'aveugle sur un ordinateur quantique. La dernière faiblesse de QFactory concerne sa preuve de sécurité. Pour l'instant, le protocole est montré sécurisé dans un cas très particulier, quand le serveur est honnête mais curieux. Durant la deuxième partie de mon stage, j'ai montré qu'il était équivalent pour notre protocole d'être sécurisé dans le modèle de Composabilité Universelle [Unr09a] et dans un modèle plus simple basé sur une sécurité de jeux. Cependant, la preuve de sécurité de ce jeu n'est pas un problème facile, et nous allons concentrer dans le futur nos efforts sur la résolution de ce problème.

Remerciements

Je souhaiterais commencer par remercier sincèrement ma directrice de stage Céline Chevalier, pour son écoute sans faille, ses fines analyses, et les discussions fructueuses que nous avons eues et qui m'ont été d'une aide si précieuse. Je tenais également à remercier mes collaborateurs : Elham Kashefi, avec son dynamisme à toute épreuve et ses idées toujours percutantes, Petros Wallden, et ses débats parfois animés, mais toujours passionnants, et sans oublier Alexandru Cojocaru, mon inséparable collaborateur avec qui j'ai pu passer des centaines d'heures à discuter, argumenter, débattre... et sans qui ce projet ne serait certainement pas ce qu'il est à l'heure actuelle. Enfin, un grand merci à tous les membres du laboratoire de Cryptographie d'Ulm pour leur accueil chaleureux, leurs discussions captivantes, les sorties et les jeux de sociétés !

Table des matières

1 Introduction et état de l'art

Depuis plus de 80 ans, le modèle de la machine de Turing est le modèle de référence pour la conception d'algorithmes. Cependant, avec la découverte de la physique quantique, un nouveau modèle, plus puissant, à été développé : c'est lui qui a mené à la définition d'*ordinateur quantique*. Ainsi, certains problèmes réputés difficiles pour une machine de Turing deviennent simples à résoudre pour un ordinateur quantique. C'est le cas, par exemple, du problème de factorisation (problème qui a permis la mise au point de nombreuse primitives cryptographiques comme le chiffrement RSA), pour lequel il existe un algorithme quantique polynomial permettant de le résoudre. Pour l'instant, l'ordinateur quantique n'est qu'une hypothétique machine, mais la recherche progresse très rapidement, et certaines sociétés comme IBM ou Google ont réussi à construire des ordinateurs quantiques travaillant sur plusieurs dizaines de qubits, et permettent au grand public de les tester via des services de « cloud »[IBM]. Ainsi, la réalisation d'un ordinateur quantique utilisable à distance par le plus grand nombre devient de plus en plus une réalité.

Si un tel ordinateur venait à voir le jour, de nombreux utilisateurs pourraient vouloir l'utiliser. Afin de leur permettre de garantir la confidentialité de leurs données, ainsi que de cacher au serveur quantique les calculs qu'il effectue, un modèle a été développé : *l'Universal Blind Quantum Computing* (UBQC)[BFK08]. Ce modèle permet d'effectuer n'importe quel algorithme quantique sur un serveur distant en ne révélant rien au serveur. Cependant, l'UBQC a besoin que le client puisse préparer de manière aléatoire un « qubit » (photon polarisé...), et puisse l'envoyer physiquement au serveur. Il faudrait donc que chaque utilisateur ait une connexion quantique qui le relie au serveur, et qu'il puisse produire et envoyer sur demande des qubits. À l'heure actuelle, les chercheurs arrivent difficilement à faire des canaux quantiques qui dépassent la centaine de kilomètres par fibre optique, et le millier de kilomètres par satellite [RXY+17]. Ainsi, développer un « internet quantique » sur toute la surface du globe sera bien évidemment un processus extrêmement coûteux qui mettra des dizaines d'années à se mettre en place. De plus, à notre connaissance, la seule méthode pour envoyer de l'information quantique est de passer par des photons, et il est relativement difficile de faire interagir les technologies les plus prometteuses pour faire un ordinateur quantique (atomes froids...) avec des photons. L'intérêt d'avoir un client purement classique est donc double : permettre la délégation de calcul à l'aveugle sur un matériel ne gérant pas l'interaction matière-photon, et permettre la délégation de calculs sur un réseau internet classique, bien moins coûteux qu'un internet quantique.

Il y a eu un nombre important de propositions qui ont été faites pour essayer de simplifier au maximum les besoins du client. Certaines arrivent à avoir un client purement classique [RUV12], mais en contrepartie elles nécessitent d'avoir deux serveurs quantiques ne pouvant pas communiquer mais

partageant une paire EPR (un état quantique particulier), ce qui est une hypothèse très forte et assez improbable. Une autre approche [MDMF17] arrivait à avoir un client purement classique, mais cette proposition a deux problèmes : premièrement la sécurité est très imprécise car le serveur arrive à extraire certaines informations. Deuxièmement, et plus problématique encore, il n'y a pas de manière connue pour transformer n'importe quel circuit en un circuit exploitable par ce protocole, ce qui le rend non universel. Lorsque j'ai commencé à étudier ce problème il y a un an lors de mon stage précédent, nous avons créé le premier protocole permettant de résoudre ce problème [CCKW18] en simulant de manière classique le canal de communication entre le client et le serveur. Cependant, la construction nécessitait une fonction cryptographique particulière, et la première proposition que nous avons faite s'est avérée non fonctionnelle. Avant que nous publions notre article avec une première preuve de sécurité, Urmila Mahadev a proposé un autre protocole [Mah17] réalisant cette fonctionnalité en se basant sur une idée centrale similaire à la notre, mais en l'utilisant de manière très différente. Ce nouveau protocole a l'avantage d'être non interactif, mais nécessite d'utiliser un chiffrement homomorphe classique pour mettre à jour la clé de chiffrement, et telle quelle, cette approche ne cache pas le calcul effectué, et nécessiterait d'utiliser une machine universelle, pour cacher également le calcul effectué. De plus, il ne permet pas de vérifier si le serveur a été honnête et s'il a bien renvoyé le bon résultat, contrairement à notre approche. Deux autres articles ont par la suite été publiés, [BCM+18, Mah18] et proposent une autre approche pour arriver à la vérification, en forçant le serveur à mesurer le résultat de manière honnête au lieu de simuler un canal quantique.

Ces nouvelles approchent ont également besoin d'une fonction cryptographique très particulière, mais n'ayant pas exactement les mêmes besoins qu'elles, nous ne pouvons pas réutiliser ces fonctions pour notre protocole QFactory. J'ai donc, pendant la première partie de mon stage cherché à trouver un candidat pour la fonction que notre protocole QFactory requiert. Ensuite, la première version de notre papier prouve que le protocole est sécurisé quand le serveur est honnête mais curieux, c'est à dire quand le serveur suit le protocole et essaye d'en extraire de l'information. Ainsi pendant la deuxième partie de mon stage j'ai donc essayé d'étudier la sécurité de notre protocole dans le cas où le serveur est totalement malhonnête. Cette étape n'est pas encore terminée et est très complexe, mais j'ai réduit la preuve dans le modèle le plus complet de Composabilité Universelle Quantique [Unr09a] à un modèle plus simple basé sur les « jeux » (game-based proof).

2 Préliminaires

2.1 Rôle de la fonction cryptographique dans QFactory

Dans ce rapport, je vais me focaliser sur l'élaboration de la fonction cryptographique dont notre protocole QFactory a besoin, car ce protocole n'a pas vraiment été modifié depuis mon stage précédent, et son explication nécessiterait l'introduction d'un certain nombre de concepts que la suite du rapport n'utilisera pas (bases de l'informatique quantique entre autre). Je vais donc me contenter de donner une idée approximative de l'utilisation de cette fonction pour déléguer du calcul quantique à l'aveugle (c'est à dire sans révéler ni le calcul à effectuer, ni les entrées, ni les sorties) à partir d'un client classique, et j'invite le lecteur plus curieux à aller lire notre article [CCKW18]. *Veuillez également noter qu'il n'est pas nécessaire de comprendre totalement cette section pour comprendre le reste du rapport.*

La première brique pour comprendre notre protocole est de savoir comment on peut déléguer du calcul quantique à l'aveugle *à partir d'un client disposant d'un petit ordinateur quantique et d'un canal quantique le reliant au serveur* :

Théorème 2.1 (Informel, UBQC [BFK08])**.** *Il existe un protocole nommé Universal Blind Quantum Computing (UBQC) qui permet de déléguer du calcul quantique à l'aveugle à partir d'un client qui peut seulement envoyer au serveur de manière aléatoire des qubits $|+_\theta\rangle$ avec θ choisit aléatoirement dans $E := \{0, \frac{\pi}{4} \dots \frac{7\pi}{4}\}$.*

Vous trouverez Annexe A une description simplifiée du protocole UBQC.

La deuxième brique a été apportée par notre protocole QFactory (pour lequel nous avons pour l'instant une démonstration dans le cas où le serveur est honnête mais curieux), qui, combiné à UBQC, permet de réaliser notre calcul délégué :

Théorème 2.2 (Informel, QFactory [CCKW18])**.** *Il existe un protocole (nommé QFactory) entre un client purement classique et un serveur quantique, qui permet de simuler l'envoie de qubits $|+_\theta\rangle$ (avec θ choisit aléatoirement dans E) entre le client et le serveur, de manière à ce que seul le client connaisse θ, sous réserve de l'existence d'une fonction cryptographique f_k qui soit (entre autre) difficile à inverser pour le serveur, simple à inverser pour le client, et qui possède exactement deux antécédents pour chaque image (on dit que la fonction est 2-vers-1).*

Démonstration. L'idée principale consiste à demander au serveur de créer (de manière quantique) une grande superposition sur tous les couples antécédents/images $\sum_x |x\rangle|f_k(x)\rangle$, puis de mesurer le deuxième registre. Comme la fonction est 2-vers-1, le serveur va alors obtenir un état $(|x\rangle + |x'\rangle) \otimes |y\rangle$,

avec $f_k(x) = f_k(x') = y$, $x \neq x'$. Intuitivement, le serveur ne peut pas obtenir x et x' à partir du y car la fonction est difficile à inverser pour lui, mais en envoyant le y au client, le client va pouvoir recalculer de son côté le x et le x' et ainsi savoir quel est l'état quantique obtenu par le serveur. Une ultime étape permet de transformer l'état $|x\rangle + |x'\rangle$ en un état $|+_\theta\rangle$, avec θ qui dépends de x et x', de manière à cacher au serveur l'angle θ. La preuve de sécurité dans le cas honnête mais curieux se base sur une adaptation du théorème de Goldreich-Levin [GL89] et sur l'utilisation du théorème de Vazirani-Vazirani [VV85]. □

On notera également que QFactory peut remplacer à priori n'importe quel canal quantique (on peut toujours le coupler à UBQC pour envoyer n'importe quel qubit), et est donc une fonctionnalité très intéressante en elle même, qui pourra être adaptée à de nombreux protocoles, y compris des protocoles permettant d'effectuer des calculs multi-partites [PCW+12, KMW17], quitte à éventuellement rajouter une preuve *Zero-Knowledge* pour attester de l'honnêteté du client, et ainsi booster la sécurité de ces protocoles.

2.2 Notations et définitions cryptographiques

Dans la suite du rapport, nous allons principalement nous focaliser sur l'implémentation de cette fonction, et nous allons donc commencer par voir les notations et définitions de sécurité élémentaires en cryptographie.

Nous noterons par $\mathsf{negl}(n)$ toute fonction f dite *négligeable*, c'est à dire telle que pour toute constante c, $f(n) = o(1/n^c)$. Étant donné que nous souhaitons une sécurité contre des adversaires quantiques, les adversaires seront ainsi vus comme des adversaires appartenant à la classe *Quantum Polynomial Time algorithms* (QPT), qui est la version quantique de l'usuelle classe des algorithmes probabilistes s'exécutant en temps polynomial sur un ordinateur classique : *Probabilistic Polynomial Time algorithms* (PPT). Ainsi, toutes les définitions suivantes reprennent des définitions usuelles en cryptographie, en les adaptant à des adversaires quantiques. Dans la suite, les fonctions seront souvent paramétrées par un indice $k \in K$, et on supposera que l'on dispose toujours d'un algorithme PPT $\mathrm{Gen}(1^\lambda) \in K$ pouvant générer aléatoirement des indices de fonction en fonction d'un paramètre de sécurité λ. On dira également d'une fonction qu'elle est **2-vers-1** si la fonction est déterministe et que pour chaque image il existe exactement deux antécédents, et δ-**2-vers-1** lorsque la fonction est 2-vers-1 avec une probabilité (sur l'indice k de la fonction et sur le $y \in \mathrm{Im}(f_k)$) supérieure à δ.

Définition 2.1 (Fonction à sens unique)**.** *Une famille de fonctions* $\{f_k : D \to R\}_{k \in K}$ *est dite* ***à sens unique*** *(ou one-way) si :*

- *pour tout indice de fonction k et pour toute entrée $x \in D$, il existe un algorithme PPT pouvant calculer $f_k(x)$*

— *il n'existe pas d'adversaire QPT $\mathcal{A}$ pouvant inverser f_k avec une probabilité non négligeable :*

$$\Pr_{\substack{k\leftarrow \mathrm{Gen}(1^\lambda)\\ x\leftarrow D}}[f(\mathcal{A}(k, f_k(x)) = f(x)] \leq \mathsf{negl}(n)$$

Définition 2.2 (Résistance au deuxième antécédent)**.** *Une famille de fonctions $\{f_k : D \to R\}_{k\in K}$ est **résistante au deuxième antécédent** (ou second preimage resistant) si :*

— *pour tout indice de fonction k et pour toute entrée $x \in D$, il existe un algorithme PPT pouvant calculer $f_k(x)$*

— *il n'existe pas d'algorithme QPT $\mathcal{A}$, qui, étant donné une entrée x, peut trouver une autre entrée x' telle que $f_k(x) = f_k(x')$ avec une probabilité non négligeable :*

$$\Pr_{\substack{k\leftarrow \mathrm{Gen}(1^n)\\ x\leftarrow D}}[\mathcal{A}(k, x) = x' \text{ tel que } x \neq x' \text{ et } f_k(x) = f_k(x')] \leq \mathsf{negl}(n)$$

Définition 2.3 (Résistance aux collisions)**.** *Une famille de fonctions $\{f_k : D \to R\}_{k\in K}$ est dite **résistante aux collisions** si :*

— *pour tout indice de fonction k et pour toute entrée $x \in D$, il existe un algorithme PPT pouvant calculer $f_k(x)$*

— *il n'existe pas d'algorithme QPT $\mathcal{A}$ pouvant trouver deux entrées $x \neq x'$ telles que $f_k(x) = f_k(x')$ avec une probabilité non négligeable :*

$$\Pr_{\substack{k\leftarrow Gen(1^n)\\ rc\leftarrow\{0,1\}^*}}[\mathcal{A}(k) = (x, x') \text{ tel que } x \neq x' \text{ et } f_k(x) = f_k(x')] \leq \mathsf{negl}(n)$$

Théorème 2.3. *[KL14] Toute fonction résistante aux collisions est également résistante au deuxième antécédent.*

Définition 2.4 (Fonction à porte dérobée)**.** *Une famille de fonctions $\{f_k : D \to R\}_{k\in K}$ est une **fonction à porte dérobée** (ou trapdoor function) si :*

— *il existe un algorithme PPT* Gen *qui, sur l'entrée 1^λ (avec λ un paramètre de sécurité) donne un couple (k, t_k), où $k \in K$ représente l'indice de la fonction, et t_k représente la porte dérobée*

— *$\{f_k : D \to R\}_{k\in K}$ est une famille à sens unique*

— *il existe un algorithme PPT* `Inv`*, qui, étant donné une porte dérobée t_k et $y = f_k(x)$, peut donner tous les antécédents de y par f_k*

Enfin, nous aurons besoin par la suite de fonctions homomorphes :

Définition 2.5 (Fonction homomorphe)**.** *On dit d'une fonction f qu'elle est homomorphe lorsqu'il existe deux lois de groupes $\circ$ et $\bullet$ tels que $\forall x, x'$, $f(x \circ x') = f(x) \bullet f(x')$. Ainsi, il existe un élément neutre noté 1 pour la relation $\circ$ et chaque élément x possède un inverse noté x^{-1} de manière à ce que $\forall x, x \circ x^{-1} = x^{-1} \circ x = 1$.*

Le Théorème 2.2 (informel) peut maintenant être précisé. Nous avons ainsi besoin de trouver une famille de fonctions 2-vers-1 qui soit à porte dérobée et résistante aux collisions. [1]

2.3 Réseaux euclidiens et apprentissage avec des erreurs (LWE)

Pour trouver une telle fonction, nous allons devoir nous appuyer sur des problèmes cryptographiques réputés sécurisés face à un adversaire quantique. Nous ne pouvons donc pas utiliser les problèmes basés sur la factorisation d'entiers (comme RSA [RSA78]) largement utilisés à l'heure actuelle, ou sur le logarithme discret/courbes elliptiques car ces problèmes ne sont pas sécurisés face à un ordinateur quantique. Dans la liste des cryptosystèmes réputés sécurisés face à un ordinateur quantique, les méthodes les plus éprouvées à nos jours se basent sur des problèmes sur les réseaux euclidiens (*lattices* en anglais), ou sur sa variante appelée Apprentissage Avec des Erreurs (abrégée par la suite LWE). Nous avons également essayé d'exploiter d'autres cryptosystèmes, notamment basés sur les codes correcteurs d'erreurs [CFS01], mais l'efficacité de ces systèmes s'est révélée être désastreuse dans notre cas, nous nous cantonnerons donc à étudier les réseaux euclidiens et le problème LWE.

Les réseau euclidiens sont des structures élémentaires assez simples à appréhender et très visuelles, comme illustré Figure 1 :

Définition 2.6 (Réseau euclidien [Pei15])**.** *Un réseau euclidien $\mathcal{L}$, de dimension n, est un sous-groupe additif discret de $\mathbb{R}^n$.*

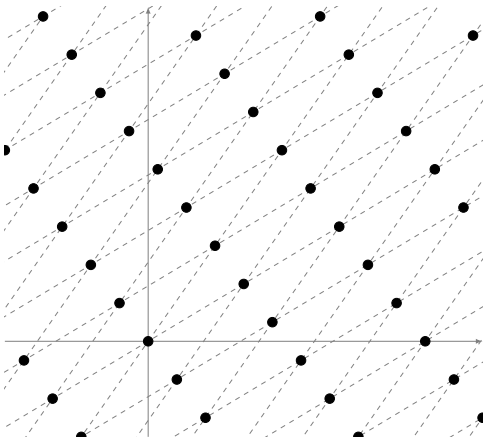


FIGURE 1 – Réseau euclidien de dimension 2

1. Notez que dans la preuve actuelle de sécurité dans le cas honnête mais curieux, nous avons seulement besoin de la propriété de résistance au deuxième antécédent, mais cette propriété n'est pas suffisante lorsque l'on désire éviter les attaques actives.

L'idée centrale autour de la sécurité des réseaux euclidiens repose sur le fait que l'on peut associer à chaque réseau une « bonne base » (une base étant comme dans les espaces vectoriels des vecteurs générateurs du réseau) et une « mauvaise base ». Ainsi, un utilisateur possédant une bonne base pourra résoudre efficacement la plupart des problèmes liés à ce réseau (par exemple trouver un plus court vecteur, ou trouver le point du réseau le plus proche d'un point arbitraire...). Cependant, si on ne possède qu'une « mauvaise base », tous ces problèmes sont difficiles à résoudre [2] [Pei15].

Le problème qui nous intéressera le plus par la suite sera la problème de décodage à distance borné (ou bounded-distance decoding (BDD) problem) représenté sur la Figure 2, qui consiste à devoir trouver le point du réseau le plus proche d'une cible garantie d'être suffisamment proche du réseau. Plus formellement :

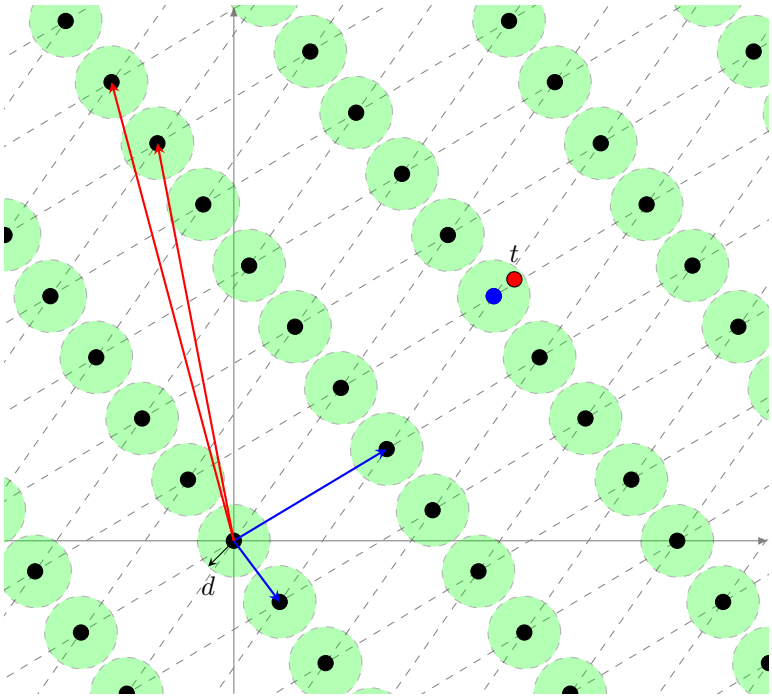


FIGURE 2 – Représentation du problème BDD_γ. Les cercles verts délimitent la distance maximale de la cible, la cible t est représentée en rouge, et le point le plus proche de t sur le réseau est représenté en bleu. Une bonne base a été représentée par les flèches en bleu, tandis qu'une mauvaise base (reconnaissable à sa grande longueur et au fait qu'elle est loin d'être orthogonale) a été représentée en rouge.

2. Il est important de noter que ces problèmes sont difficiles lorsque la dimension n devient grande. Ainsi, pour de petites dimensions, il existe des algorithmes polynomiaux pouvant résoudre ces problèmes.

Définition 2.7 (BDD_γ, [Pei15]). *Étant donné une base* $\mathbf{B}$ *d'un réseau euclidien* $\mathcal{L}$ *et un point cible* $\mathbf{t} \in \mathbb{R}^n$ *avec la garantie que la distance entre* $\mathbf{t}$ *et le point le plus proche dans le réseau est inférieure à une certaine distance* $d := \lambda_1(\mathcal{L})/(2\gamma)$ *(γ est un facteur d'approximation, et $\lambda_i(\mathcal{L})$ est défini de manière récursive de manière à ce que λ_i soit le plus petit vecteur de $\mathcal{L}$ qui soit linéairement indépendant avec $\lambda_1 \dots \lambda_{i-1}$). Le but est alors de retrouver l'unique vecteur* $\mathbf{v} \in \mathcal{L}$ *tel que* $||\mathbf{t} - \mathbf{v}|| < d$.

Parmi les autres problèmes liés aux réseaux euclidiens, on trouve également le problème du plus court ensemble approximé de vecteurs indépendants (ou Approximate Shortest Independent Vectors Problem, SIVP_γ), qui nous sera utile surtout pour correctement régler nos paramètres de sécurité :

Définition 2.8 (SIVP_γ, [Pei15]). *Étant donné une base* $\mathbf{B}$ *d'un réseau euclidien* $\mathcal{L}$*, le but est de trouver un ensemble* $\mathbf{S} = \{\mathbf{s}_i\} \subset \mathcal{L}$ *de* n *vecteurs linéairement indépendants, avec pour tout* i, $||\mathbf{s}_i|| \leq \gamma(n)\lambda_n(\mathcal{L})$. *($\lambda_i(\mathcal{L})$ est défini de manière récursive, afin que λ_i soit le plus petit vecteur de $\mathcal{L}$ qui soit linéairement indépendant avec $\lambda_1 \dots \lambda_{i-1}$.)*

Dans le cas général (sans hypothèse particulière sur la base $\mathbf{B}$ choisie) le meilleur algorithme pour résoudre ces problèmes pour un facteur γ polynomial (en n) nécessite un temps exponentiel [Pei15].

Un autre problème, fortement lié aux problèmes de réseaux euclidiens, a également été développé et joue un rôle très important dans l'élaboration de cryptosystèmes : le problème d'apprentissage avec des erreurs (ou Learning With Errors, abrégé par la suite LWE) :

Définition 2.9 ($\mathrm{LWE}_{q,\chi}$, [Reg05]). *Soient* $n \geq 1$ *et* $q = \mathsf{poly}(n)$ *des entiers. Soient* $\mathbf{s} \in \mathbb{Z}_q^n$ *et* $\mathbf{a}_i$ *une liste de* m *vecteurs dans* $\mathbb{Z}_q^n$*, choisis aléatoirement de manière uniforme. Soit* χ *une distribution d'erreurs sur* $\mathbb{Z}_q$ *donnée en paramètre, et* e_i *une liste de* m *erreurs échantillonnées selon* χ*. Le problème d'apprentissage avec des erreurs* $\mathrm{LWE}_{q,\chi}$ *est de retrouver le vecteur* $\mathbf{s}$ *à partir de la liste de valeurs* $\langle \mathbf{s}, \mathbf{a}_i \rangle + e_i$.

En mettant la liste $\mathbf{a}_i$ *sous forme matricielle* $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ *et* $e_i \in \mathbb{Z}_q^m$ *sous forme de vecteur* $\mathbf{e}$*, le problème est alors de retrouver* $\mathbf{s}$ *à partir de* $\mathbf{As} + \mathbf{e}$.

La difficulté de ce problème réside beaucoup dans le choix de la distribution χ. Ainsi, en définissant Ψ_α comme la distribution Gaussienne sur $\mathbb{R}/[0,1]$, centrée en 0 et de paramètre α, et $\bar{\Psi}_\alpha$ comme étant sa version discrète sur $\mathbb{Z}_p$ comme illustré Figure 3, ce problème est au moins aussi difficile que SIVP_γ :

Théorème 2.4 ([Reg05]). *Soit* n, q *deux entiers, et* $\alpha \in]0,1[$ *tel que* $\alpha p > 2\sqrt{n}$*. S'il existe un algorithme efficace pour résoudre* $\mathrm{LWE}_{q,\bar{\Psi}_\alpha}$ *alors il existe un algorithme (quantique) efficace pour résoudre* SIVP_γ *avec* $\gamma = \tilde{O}(n/\alpha)$ *dans le pire des cas.*

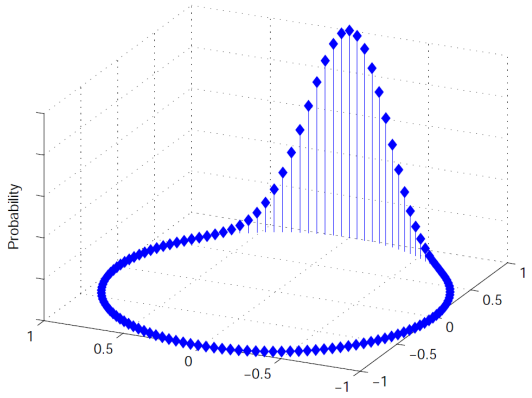


FIGURE 3 – $\bar{\Psi}_\alpha$ avec $\alpha = 0.1$ et $q = 127$, les éléments de $\mathbb{Z}_q$ étant arrangés suivant un cercle.

Ce problème offre ainsi l'avantage d'être difficile *en moyenne*, alors que la plupart des problèmes utilisés en cryptographie ne disposent de preuves de sécurité que dans le pire des cas. Nous allons maintenant voir comment ces problèmes sont utilisés en cryptographie, et plus particulièrement dans le cryptosystème de Micciancio et Peikert que nous avons utilisés pour définir notre fonction.

2.4 Introduction au cryptosystème de Micciancio et Peikert

Comme vu précédemment, il est crucial pour arriver à exploiter les réseaux euclidiens en cryptographie d'avoir un moyen pour générer des bonnes bases et de mauvaises bases pour les réseaux euclidiens. Le papier [MP11] fournit non seulement une méthode très efficace pour générer de telles bases, mais ils fournissent également un algorithme optimisé qui exploite une propriété particulière sur la famille de réseaux euclidiens qu'ils utilisent. Notre fonction utilisant de manière intensive cette construction, nous l'explicitons rapidement dans cette section. Notez que la description suivante est un cas particulier d'instanciation, et que [MP11] offre de nombreuses variantes que nous n'étudierons pas pour des raisons de simplicité.

Ce cryptosystème permet de définir une fonction injective $g_{\mathbf{A}}$ à porte dérobée, et utilise plusieurs paramètres : n la dimension du secret dans le problème LWE, $q := 2^k$ le module du problème LWE, α qui va servir à paramétrer la Gaussienne utilisée pour les erreurs, $\bar{m} = 2n$ et $\omega = nk$ seront des paramètres internes, et $m := \bar{m} + \omega$ sera le nombre d'échantillons. Ce système utilise également une matrice publique $\mathbf{G} \in \mathbb{Z}_q^{n \times nk}$ qui a la bonne propriété d'être très simple à inverser, et qui est définie ainsi : soit $\mathbf{g}^t :=$

$\begin{pmatrix} 1 & 2 & 4 & \cdots & 2^{k-1} \end{pmatrix} \in \mathbb{Z}_q^{1\times k}$, on pose alors

$$\mathbf{G} := \begin{pmatrix} \cdots \mathbf{g}^t \cdots & & & & \\ & \cdots \mathbf{g}^t \cdots & & & \\ & & & \ddots & \\ & & & & \cdots \mathbf{g}^t \cdots \end{pmatrix}$$

Le but est ensuite de générer une matrice $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ (qui sera rendue publique et correspond à la matrice du problème LWE) et une porte dérobée $\mathbf{R} \in \mathbb{Z}_q^{\bar{m}\times\omega}$, de manière à ce que $\mathbf{A}$ semble totalement aléatoire (nécessaire pour la sécurité de LWE). Pour générer les matrices $\mathbf{A}$ et $\mathbf{R}$, on commence par choisir uniformément au hasard une matrice $\bar{\mathbf{A}} \in \mathbb{Z}_q^{n\times\bar{m}}$, on prend ensuite la matrice $\mathbf{R} \in \mathbb{Z}_q^{\bar{m}\times\omega}$ en choisissant chaque élément de la matrice suivant la distribution Gaussienne discrète $\bar{\Psi}_{\alpha q}$, et on pose enfin $\mathbf{A} = [\bar{\mathbf{A}}|\mathbf{G} - \bar{\mathbf{A}}\mathbf{R}] \in \mathbb{Z}_q^{n\times m}$.

La fonction injective à porte dérobée est alors définie par $g_{\mathbf{A}}(\mathbf{s}, \mathbf{e}) = \mathbf{A}^t\mathbf{s} + \mathbf{e}$, avec $\mathbf{s} \in \mathbb{Z}_q^n$, et $\mathbf{e} \in \mathbb{Z}_q^m$ de manière à avoir $||\mathbf{e}|| \leq L$ (L sera explicité plus loin), et $\mathbf{R}$ permet de retrouver $\mathbf{s}$ et $\mathbf{e}$ à partir de $\mathbf{A}^t\mathbf{s} + \mathbf{e}$.

Je ne vais pas rentrer dans les détails de l'inversion de la fonction $g_{\mathbf{A}}$, dont il existe un algorithme très efficace qui exploite la structure spéciale de $\mathbf{G}$, mais la chose qu'il est important de retenir est que l'algorithme fonctionne à partir du moment où $L \leq \frac{q}{4\sqrt{s_1(\mathbf{R})^2+1}}$, avec $s_1(\mathbf{R})$ la plus grande valeur propre de $\mathbf{R}$. De plus [MP11, Lemme 2.9] nous permet de borner cette valeur :

Théorème 2.5 ([MP11, Lemme 2.9])**.** *Soit $\mathbf{R} \in \mathbb{R}^{n\times m}$ une matrice dont chaque élément est tiré de manière aléatoire suivant la distribution Gaussienne discrète $\bar{\Psi}_{\alpha q}$. Alors il existe une constante C (empiriquement proche de $1/\sqrt{2\pi}$) telle que pour tout $t \geq 0$, avec une probabilité d'au moins $1 - 2\exp(-\pi t^2)$ nous avons $s_1(R) \leq C \cdot \alpha q \cdot (\sqrt{n} + \sqrt{m} + t)$.*

Maintenant que nous avons vu les briques élémentaires dont nous avons besoin, nous allons nous intéresser à l'élaboration de notre fonction.

3 Première contribution : élaboration de la fonction nécessaire à QFactory

Pour rappel, nous souhaitons construire une famille de fonctions 2-vers-1, à porte dérobée, et résistante aux collisions. Nous allons pour cela étudier dans un premier temps deux constructions générales qui nous permettent de réduire notre recherche à des fonctions plus naturelles. Cependant, ces nouvelles fonctions s'avèrent être également difficiles à construire, et nous présenterons dans une seconde partie une famille de fonctions qui s'en approche suffisamment pour être utilisable en pratique.

3.1 Constructions générales

3.1.1 À partir de fonctions bijectives à porte dérobée

La première méthode générale pour obtenir notre famille de fonctions est assez naturelle, et suppose l'existence d'une famille de fonctions bijectives à porte dérobée $\{f_k : X \to Y\}_{k \in K}$. Notez que nous n'allons pas trop nous concentrer sur cette construction car ce n'est pas celle que nous allons utiliser par la suite.

Théorème 3.1. *Si $\mathcal{G} = \{g_k : X \to Y\}_{k \in K}$ est une famille de fonctions bijectives à porte dérobée, alors il existe une famille de fonctions $\mathcal{F}$ 2-vers-1 à porte dérobée résistante au deuxième antécédent.*

L'idée de la construction est la suivante : pour générer les clés, on choisit au hasard deux indices $k_0, k_1 \in K$, avec leur porte dérobée associée t_{k_0}, t_{k_1}, puis lors de l'évaluation, la fonction prend en entrée $x \in X$ et un bit $c \in \{0, 1\}$, et calcule $g_{k_c}(x)$. Pour inverser, comme les fonctions sont bijectives, on sait que pour tout $y \in Y$ il existe exactement un $x_0 \in X$ et un $x_1 \in X$ tel que $f_{k_0}(x_0) = f_{k_1}(x_1) = y$, on a donc exactement deux antécédents $(x_0, 0)$ et $(x_1, 1)$ et les portes dérobées t_{k_0} et t_{k_1} permettent de retrouver ces deux antécédents, ce qui est impossible à faire sans les portes dérobées.

Cependant, bien que l'on connaisse des familles de fonctions bijective à porte dérobée résistantes face à un ordinateur classique (RSA et les problèmes de logarithme discret type El-Gamal permettent tous deux ce genre de constructions), on ne connaît pas à l'heure actuelle de telle construction qui soit *résistante face à un ordinateur quantique.*

3.1.2 À partir de fonctions injectives et homomorphes à porte dérobée

La seconde méthode générale pour obtenir notre fonction se base sur une famille de fonctions aux propriétés d'homomorphies :

Théorème 3.2. *Si $\mathcal{G}$ est une famille de fonctions à porte dérobée injective et homomorphe alors il existe une famille de fonctions $\mathcal{F}$ qui est 2-vers-1, à porte dérobée, et résistante aux collisions.*

Pour construire $\mathcal{G}$ à partir de $\mathcal{F} = \{f_k : X \to Y\}_{k \in K}$, on procède ainsi : pour générer la porte dérobée, on prend une porte dérobée t_k et l'indice de fonction $k \in K$ associé de manière aléatoire, et on choisit également au hasard un élément $x_0 \in X$. La porte dérobée de la nouvelle fonction est alors $t'_k := (t_k, x_0)$ et l'indice de fonction devient $k' := (k, y_0 := f_k(x_0))$. Pour évaluer la fonction $g_{k'} : X \times \{0, 1\} \to Y$, on prend en entrée un $x \in X$, un bit $c \in \{0, 1\}$, et si $c = 0$, alors $g_{k'}(x, 0) := f_k(x)$, et si $c = 1$, alors $g_{k'}(x, 1) := f_k(x) \bullet f_k(x_0)$. Étant donné que la fonction f_k est homomorphe, on a également $g_{k'}(x, 1) = f_k(x \circ x_0)$. Ainsi, pour inverser notre fonction, on

commence par calculer le premier antécédent $x = f_k^{-1}(y)$ en utilisant la porte dérobée t_k, puis pour obtenir le deuxième antécédent, on calcule $x' = x \circ x_0^{-1}$. Ainsi, on a bien

$$\begin{aligned} g_{k'}(x', 1) &= f_k(x') \bullet f_k(x_0) \\ &= f_k(x \circ x_0^{-1}) \bullet f_k(x_0) \\ &= f_k(x \circ x_0^{-1} \circ x_0) \\ &= f_k(x) = y = g_{k'}(x, 0) \end{aligned}$$

La preuve de sécurité est disponible en Annexe B. Nous avons ainsi réduit notre problème à la recherche d'une famille de fonctions injectives, homomorphes, et à porte dérobée, qui ressemble ainsi fortement à un schéma de chiffrement. Encore une fois, ce type de fonctions serait simple à trouver (toujours en utilisant RSA et les problèmes de logarithme discret) si nous n'avions pas la contrainte d'être également sécurisé face à un ordinateur quantique. Cependant, le problème s'avère être plus difficile si on souhaite être sécurisé également contre les ordinateurs quantiques. Il reste cependant un espoir : les problèmes basés sur les réseaux (et les codes correcteurs d'erreurs) ont une structure naturellement homomorphe, étant donné que l'évaluation est souvent limitée à des opérations linéaires (addition et multiplication matricielle). Cependant, il semble impossible d'obtenir une fonction purement homomorphe, comme nous allons le voir.

3.2 Les difficultés rencontrées

Le principal problème qui nous empêche d'avoir des fonctions parfaitement homomorphes est qu'il y a dans la quasi totalité des problèmes de cryptographie post-quantique une notion de bruit qui doit rester faible, et que lorsque l'on ajoute deux bruits, même faibles, le bruit total n'est plus aussi faible qu'au début, et n'appartient donc plus à l'ensemble de départ : ainsi avec une grande probabilité notre fonction n'aura qu'un seul antécédent, comme illustré Figure 4. Notez qu'en petite dimension, la probabilité d'avoir deux antécédents peut sembler non négligeable, malheureusement lorsque la dimension augmente, la probabilité d'avoir deux antécédents décroît de manière exponentielle.

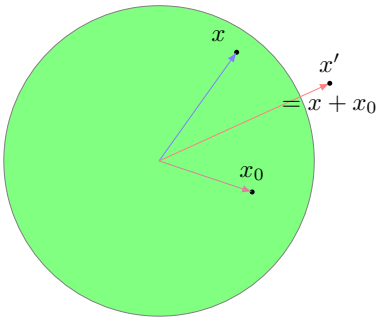


FIGURE 4 – Exemple des problèmes rencontrés : le cercle vert représente l'ensemble des entrées où est définie notre fonction, x_0 représente la porte dérobée, et x, x' sont deux potentiels antécédents. Cependant, même si x et x_0 sont bien dans l'ensemble d'entrée, x' n'appartient pas à l'ensemble d'entrée, et n'est donc pas un antécédent valide.

Ainsi, lors de nos premiers essais, nous avons essayé de partir des « fonctions à pertes » (lossy functions) [PW07]. L'idée est que le message était encodé par un vecteur de bit $x \in \{0,1\}^n$, puis on le multipliait par une matrice pour calculer la fonction. Le problème est que pour deux messages $x, x' \in \{0,1\}^n$, la différence des deux messages $x - x'$ n'est pas forcément dans l'ensemble d'entrée $\{0,1\}^n$, et peut à priori appartenir à $\{-1,0,1\}^n$. Changer l'ensemble de départ de x pour le mettre dans l'ensemble « maximal » $\mathbb{Z}_q^n$ résout ce problème, mais en créé un autre : la fonction est alors fortement non injective...

La deuxième idée consistait à essayer de rester dans $\{0,1\}^n$, mais de limiter le nombre maximal de bits autorisés. Le problème alors est que la fonction n'est plus forcément sécurisée, et il est important de bien choisir les paramètres pour garder à la fois une sécurité élevée, et une probabilité non-négligeable d'avoir deux antécédents. Cette idée avait déjà été exploitée par [CFS01] pour implémenter un système de signatures, cependant pour arriver à avoir une sécurité intéressante les auteurs du papier ont du choisir des paramètres qui rendaient la probabilité de réussite très faible, et ils avaient ainsi besoin de répéter l'algorithme un nombre très important de fois pour signer un seul document, ce qui aurait mené à une complexité désastreuse une fois adaptée à notre protocole QFactory.

3.3 Implémentation à base de LWE

La méthode à laquelle nous sommes finalement arrivés ne force plus l'entrée à être dans $\{0,1\}^n$, mais maintenant un élément de l'espace d'entrée est composé de deux vecteurs dans $\mathbb{Z}_q$: $x := (s, e)$. s est un élément tiré aléatoirement dans $\mathbb{Z}_q^n$ et représente le vecteur secret de la construction de Micciancio et Peikert présentée plus haut [MP11]. $e \in \mathbb{Z}_q^m$ représente le vecteur d'erreur de cette même construction. Suivant le rôle du x, e va être choisi de manière différente :

- la fonction $g_{k'}$ sera définie sur tous les $x := (s, e) \in \mathbb{Z}_q^n \times \mathbb{Z}_q^m$ tels que $||e||_\infty \leq \mu$, avec μ judicieusement choisi.
- en revanche pour choisir la porte dérobée $x_0 = (s_0, e_0)$, les composants du vecteur d'erreur vont être tirés suivant une Gaussienne de paramètre αq, avec α "petit" par rapport à μ. En pratique, il faut que αq soit au moins $\sqrt{m}$ fois plus petit que μ, comme nous le verrons par la suite.

Ceci revient ainsi à forcer la différence entre les deux antécédents à être suffisamment petite pour que si un antécédent est dans l'espace d'entrée, alors l'autre antécédent l'est également avec une grande probabilité.

On voit ainsi apparaître un compromis entre la sécurité et la probabilité de succès. Plus αq va être petite, plus la probabilité de succès sera grande, mais plus la sécurité sera faible. La magie réside dans le fait qu'il est possible de trouver un ensemble de paramètres qui permettent à la fois d'avoir une

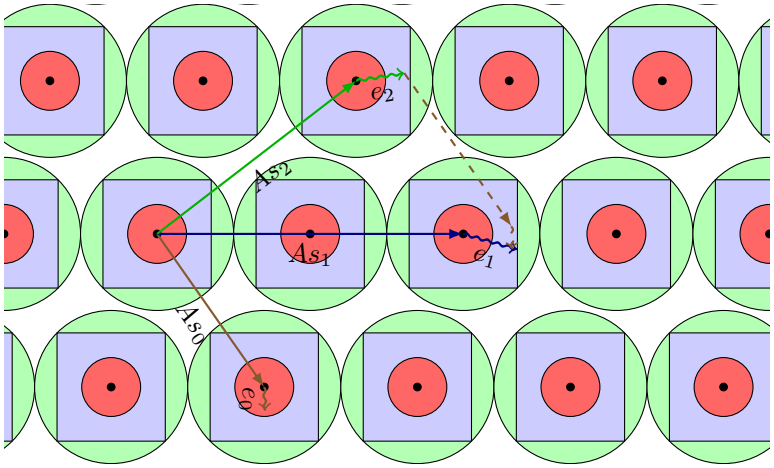


FIGURE 5 – Ce dessin permet à la fois de représenter l'entrée et la sortie de $f_{k'}$. Le point à la fin de la double flèche représente la sortie, tandis que l'entrée est caractérisée par les deux vecteurs intermédiaires représentant le vecteur s (en réalité As) et le vecteur d'erreur e. La porte dérobée est représentée en marron, et une collision a été représentée : le premier vecteur est en bleu et le deuxième en vert. La zone verte représente la partie que l'on peut déchiffrer, la zone bleue représente la partie sur laquelle la fonction est définie, et l'erreur de la porte dérobée est choisie suivant une Gaussienne représentée en rouge.

sécurité qui s'accroît de manière exponentielle avec le paramètre de sécurité, tout en gardant une probabilité de succès non négligeable.

La Figure 5 illustre ce choix de paramètre dans la fonction 2-vers-1, définie en mixant les deux constructions définies plus haut par $f_{k'}((s,e),c) = As + e + c \times y_0$.

Nous présentons maintenant les deux théorèmes principaux : d'abord nous explicitons des conditions suffisantes sur les paramètres pour avoir une fonction sécurisée avec une probabilité de succès non négligeable, puis nous donnons un ensemble de paramètres satisfaisant ces conditions :

Lemme 3.3 (Conditions sur les paramètres)**.** *Pour tous $n, q, \mu \in \mathbb{Z}, \mu' \in \mathbb{R}$, on définit :*

$$k := \lceil log(q) \rceil \qquad \bar{m} = 2n \qquad \omega = nk$$
$$m := \bar{m} + \omega = 2n + nk \qquad \alpha' = \frac{\mu'}{\sqrt{mq}} \qquad \alpha = m\alpha'$$

ainsi que C la constante définie dans [MP11, Lemme 2.9] (de l'ordre de $\frac{1}{\sqrt{2\pi}}$) et $B = 2$ si q est une puissance de 2 ou $B = \sqrt{5}$ sinon. Maintenant, si pour tout paramètre de sécurité n (dimension du réseau euclidien), il existe q (le module de LWE*) et μ (l'amplitude maximale des composants de l'erreur) tels que :*

1. *m est tel que $n = o(m)$ (requis pour l'injectivité de la fonction [Vai])*

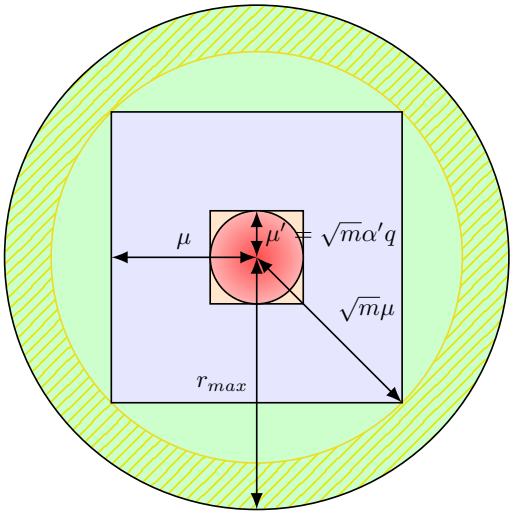


FIGURE 6 – Le cercle rouge représente le domaine de définition des erreurs pour la porte dérobée, choisit suivant une distribution Gaussienne. Le carré orange est une approximation du domaine, qui doit être bien plus petit que la longueur du carré bleu, utilisé comme domaine d'entrée des erreurs de la fonction, lui même placé dans la zone où la fonction peut-être inversée (cercle vert). La partie hachurée sert à s'assurer qu'en cas de collision (x_1, x_2), on ait $x_1 = x_2 \pm x_0$.

2. $0 < \alpha < 1$
3. $\mu' = O(\mu/m)$ *(requis pour avoir une probabilité non négligeable d'avoir deux antécédents*[3]*)*
4. $\alpha' q \geq 2\sqrt{n}$ *(requis pour la réduction de* SIVP *vers* LWE*)*
5. $\frac{n}{\alpha'}$ *est* $\mathsf{poly}(n)$ *(représentant, à un facteur multiplicatif près, le facteur d'approximation* γ *dans le problème* SIVP_γ*)*
6. $$\sqrt{m}\mu < \underbrace{\frac{q}{2B\sqrt{\left(C \cdot (\alpha \cdot q) \cdot (\sqrt{2n} + \sqrt{kn} + \sqrt{n})\right)^2 + 1}}}_{r_{max}} - \mu'\sqrt{m}$$

   (requis pour la correction de l'algorithme d'inversion - r_{max} *représente la longueur maximale du vecteur d'erreur que l'on peut corriger avec [MP11]*[4]*, et le dernier terme est requis pour la preuve de résistance*

3. en précisant la preuve on doit pouvoir avoir même $\mu' = O(\mu/\sqrt{m})$
4. Nous avons choisi d'utiliser la définition calculatoire de [MP11], mais ce théorème

à la collision pour s'assurer de l'injectivité même lorsque l'on ajoute le bruit de la porte dérobée comme illustré Figure 6)

alors il existe une famille de fonctions (définie Définition C.1) qui est δ-2-vers-1 (avec δ au moins aussi grand qu'une constante), à porte dérobée et résistante aux collisions, sous l'hypothèse qu'il n'y a pas d'algorithme quantique qui peut résoudre efficacement $SIVP_\gamma$, avec $\gamma = \mathsf{poly}(n)$.

Lemme 3.4 (Existence des paramètres)**.** *Les paramètres suivants respectent les conditions du Lemme 3.3 :*

$n = \lambda$	$k = 5\lceil log(n)\rceil + 21$	$q = 2^k$
$\bar{m} = 2n$	$\omega = nk$	$m = \bar{m} + \omega$
$\mu = \lceil 2mn\sqrt{2+k} \rceil$	$\mu' = \mu/m$	$B = 2$

avec α, α', C définies comme dans le Lemme 3.3.

La preuve est donnée en Annexe D. Nous avons ainsi montré qu'il est possible de construire une fonction qui a des propriétés suffisantes pour pouvoir implémenter notre protocole QFactory. Veuillez notez cependant que nous n'avons pas essayé de trouver des paramètres optimaux, et qu'il existe un compromis entre sécurité/efficacité et probabilité de succès.

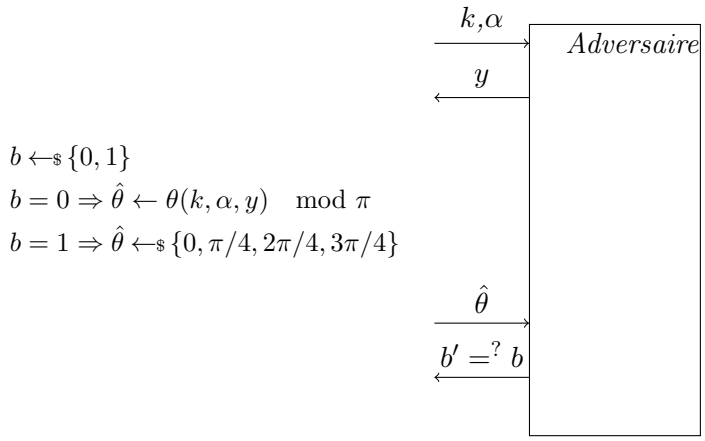
4 Deuxième contribution : discussions sur la sécurité dans le cadre de la Composabilité Universelle

Durant la deuxième partie de mon stage, je me suis appliqué à essayer de prouver la sécurité du protocole dans un modèle beaucoup plus réaliste, dans lequel le serveur est totalement malicieux (la première preuve nécessitait que le y soit aléatoirement choisit). Le modèle choisit est celui de *Composabilité Universelle Quantique* (abrégé UC, introduit au départ dans sa version classique [Can00] puis étendu au cas quantique par la suite [Unr09b]), qui est l'un des modèles d'adversaires les plus puissants (il permet notamment de garantir la sécurité même en cas de composition en parallèle).

Nous avons ainsi montré que pour prouver la sécurité de notre protocole dans ce modèle, il suffisait de montrer qu'il n'existait pas d'adversaire pouvant gagner un jeu donné :

Théorème 4.1 (UC vers preuve à base de jeux)**.** *Le protocole QFactory est sécurisé dans le modèle UC si il n'existe pas d'adversaire quantique polynomial pouvant résoudre le jeu suivant (une variante donne le α après le y) :*

peut facilement être étendu à d'autres définitions du même papier, ou même à d'autres constructions de base courte pour un réseau euclidien)



Nous avons essayé de prouver la sécurité de ce jeu, cependant le fait que l'adversaire est quantique et qu'il peut posséder une copie $|+_\theta\rangle$ qu'il peut utiliser pour extraire de l'information, couplé au fait que la méthode dite du « rewinding » est très délicate à utiliser avec un adversaire quantique rend le problème très difficile. Nous avons essayé de modifier QFactory en rajoutant avec le squeezing une fonction de hash 2-universelle, et nous avons une preuve informelle qui dit que comme la première fonction est résistante aux collisions, il y a au moins une sortie qu'il ne connaît pas parfaitement. Donc après application de la fonction de hash, l'une des deux images est totalement inconnue de l'adversaire, et donc le xor des deux valeurs est inconnu de l'adversaire : ainsi le θ est caché au serveur. Malheureusement, cet argument informel n'est pas simple à formaliser, et nécessitera des efforts supplémentaires pour arriver à une preuve totalement satisfaisante...

5 Conclusion et perspectives

Pour résumer, nous avons ainsi trouvé une fonction candidate pour notre protocole QFactory, qui est 2-vers-1 avec une probabilité au moins constante, à porte dérobée, et résistante contre un adversaire quantique. Pour cela nous avons défini deux réductions générales, puis utilisé l'une de ces réductions avec une fonction dérivée de [MP11]. Pour satisfaire toutes nos conditions, nous avons dû exploiter le fait que la fonction reste sécurisée même lorsque l'on utilise des paramètres inhabituels, en trouvant un ensemble de paramètres satisfaisant toutes nos exigences.

Dans un second temps, nous avons essayé de regarder si la fonction restait sécurisée lorsque l'on avait un modèle d'adversaire plus fort, basé sur le modèle de Composabilité Universelle. Nous avons réduit la preuve dans ce modèle à un modèle plus simple, et étendu le protocole QFactory pour faciliter la preuve dans ce nouveau modèle. Cependant, la preuve s'avère être non triviale, et sera un de nos axes de recherche principaux par la suite.

Références

[BCM+18] Z. Brakerski, P. Christiano, U. Mahadev, U. Vazirani, and T. Vidick. Certifiable Randomness from a Single Quantum Device. *ArXiv e-prints*, April 2018.

[BFK08] A. Broadbent, J. Fitzsimons, and E. Kashefi. Universal blind quantum computation. *ArXiv e-prints*, July 2008.

[Can00] Ran Canetti. Universally composable security : A new paradigm for cryptographic protocols. Cryptology ePrint Archive, Report 2000/067, 2000. https://eprint.iacr.org/2000/067.

[CCKW18] A. Cojocaru, L. Colisson, E. Kashefi, and P. Wallden. On the possibility of classical client blind quantum computing. *ArXiv e-prints*, February 2018.

[CFS01] Nicolas T. Courtois, Matthieu Finiasz, and Nicolas Sendrier. How to achieve a mceliece-based digital signature scheme. In Colin Boyd, editor, *Advances in Cryptology — ASIACRYPT 2001*, pages 157–174, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg.

[GL89] O. Goldreich and L. A. Levin. A hard-core predicate for all one-way functions. In *Proceedings of the Twenty-first Annual ACM Symposium on Theory of Computing*, STOC '89, pages 25–32, New York, NY, USA, 1989. ACM.

[IBM] IBM. IBM Q Experience. www.research.ibm.com/ibm-q. [Accès en ligne le 24/07/2018].

[KL14] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography, Second Edition*. Chapman & Hall/CRC, 2nd edition, 2014.

[KMW17] E. Kashefi, L. Music, and P. Wallden. The Quantum Cut-and-Choose Technique and Quantum Two-Party Computation. *ArXiv e-prints*, March 2017.

[Mah17] U. Mahadev. Classical Homomorphic Encryption for Quantum Circuits. *ArXiv e-prints*, August 2017.

[Mah18] U. Mahadev. Classical Verification of Quantum Computations. *ArXiv e-prints*, April 2018.

[MDMF17] A. Mantri, T. F. Demarie, N. C. Menicucci, and J. F. Fitzsimons. Flow Ambiguity : A Path Towards Classically Driven Blind Quantum Computation. *Physical Review X*, 7(3) :031004, July 2017.

[MP11] Daniele Micciancio and Chris Peikert. Trapdoors for lattices : Simpler, tighter, faster, smaller. Cryptology ePrint Archive, Report 2011/501, 2011. https://eprint.iacr.org/2011/501.

[PCW+12] A. Pappa, A. Chailloux, S. Wehner, E. Diamanti, and I. Kerenidis. Multipartite Entanglement Verification Resistant against Dishonest Parties. *Physical Review Letters*, 108(26) :260502, June 2012.

[Pei15] Chris Peikert. A decade of lattice cryptography. Cryptology ePrint Archive, Report 2015/939, 2015. https://eprint.iacr.org/2015/939.

[PW07] Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. Cryptology ePrint Archive, Report 2007/279, 2007. https://eprint.iacr.org/2007/279.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proceedings of the Thirty-seventh Annual ACM Symposium on Theory of Computing*, STOC '05, pages 84–93, New York, NY, USA, 2005. ACM.

[RSA78] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2) :120–126, February 1978.

[RUV12] B. W. Reichardt, F. Unger, and U. Vazirani. A classical leash for a quantum system : Command of quantum systems via rigidity of CHSH games. *ArXiv e-prints*, September 2012.

[RXY+17] J.-G. Ren, P. Xu, H.-L. Yong, L. Zhang, S.-K. Liao, J. Yin, W.-Y. Liu, W.-Q. Cai, M. Yang, L. Li, K.-X. Yang, X. Han, Y.-Q. Yao, J. Li, H.-Y. Wu, S. Wan, L. Liu, D.-Q. Liu, Y.-W. Kuang, Z.-P. He, P. Shang, C. Guo, R.-H. Zheng, K. Tian, Z.-C. Zhu, N.-L. Liu, C.-Y. Lu, R. Shu, Y.-A. Chen, C.-Z. Peng, J.-Y. Wang, and J.-W. Pan. Ground-to-satellite quantum teleportation. 549 :70–73, September 2017.

[Sho95] P. W. Shor. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. *eprint arXiv :quant-ph/9508027*, August 1995.

[Unr09a] D. Unruh. Universally Composable Quantum Multi-Party Computation. *ArXiv e-prints*, October 2009.

[Unr09b] D. Unruh. Universally Composable Quantum Multi-Party Computation. *ArXiv e-prints*, October 2009.

[Vai] Vinod Vaikuntanathan. Advanced topics in cryptography : Lattices. https://people.csail.mit.edu/vinodv/6876-Fall2015/L13.pdf.

[VV85] Umesh V. Vazirani and Vijay V. Vazirani. Efficient and secure pseudo-random number generation (extended abstract). In George Robert Blakley and David Chaum, editors, *Advances in Cryptology*, pages 193–202, Berlin, Heidelberg, 1985. Springer Berlin Heidelberg.

Appendices

A Idée de base d'UBQC (Théorème 2.1)

Démonstration. L'idée de base est que chaque circuit quantique (un circuit quantique est un circuit ressemblant à un circuit booléen, mais avec des portes quantiques) peut être transformé en un autre circuit avec en entrée des qubits de la forme $|+_\theta\rangle$ avec θ un angle choisit aléatoirement (et uniformément) dans $E := \{0, \frac{\pi}{4} \dots \frac{7\pi}{4}\}$, des portes CTRL-Z, et des opérations de mesure avec un angle appartenant au même ensemble E. Les états quantiques $|+_\theta\rangle$ sont ensuite préparés par le client et envoyés au serveur (grâce aux lois de la physique quantique, le serveur ne peut pas déterminer l'angle θ avec certitude à partir de l'état quantique $|+_\theta\rangle$, le client est donc le seul à connaître l'angle θ). Le serveur va ensuite appliquer les portes CTRL-Z, choisies de manière à ne pas révéler le calcul effectué, puis mesurer avec les angles que le client lui donne (qui dépendent du calcul à effectuer, de la valeur des angles θ, des résultats des mesures précédentes, et d'un bit aléatoire r), en renvoyant au client les résultats des mesures au fur et à mesure. □

B Preuve du Théorème 3.2

Démonstration. Nous avons décrit juste après le Théorème 3.2 comment construire notre famille $\mathcal{F}$ à partir d'une famille $\mathcal{G}$. Il reste maintenant à montrer que cette famille $\mathcal{F}$ est bien 2-vers-1, à porte dérobée, et résistante à la collision. Pour la propriété *2-vers-1*, on a montré juste au dessus qu'il y a toujours au moins deux antécédents. Il reste à montrer qu'il n'y en a pas plus, et pour ça il suffit d'utiliser la propriété d'injectivité : comme f_k est injective, les deux fonctions partielles $g_{k'}(\cdot, 0)$ et $g_{k'}(\cdot, 1)$ sont également injectives, donc on a au plus deux antécédents.

Ensuite, la fonction est bien *résistante aux collisions*. En effet, on remarque que si on arrive à avoir une collision entre deux entrées $(x, 0)$ et $(x', 1)$, alors on a $x' = x \circ x_0^{-1}$, c'est-à-dire $x_0 = (x^{-1} \circ x')^{-1}$: on a donc réussi à retrouver la porte dérobée x_0. À partir de cette remarque, il est facile de construire un adversaire $\mathcal{A}$ qui casse la propriété de sens unique de la famille $\mathcal{G}$ ce qui est absurde ($\mathcal{G}$ est une fonction à porte dérobée, donc à sens unique, cf notre papier [CCKW18] pour les détails de la réduction).

Enfin, cette fonction est bien à *porte dérobée*, car on a montré plus haut qu'avec la trapdoor, on pouvait retrouver les deux antécédents, et le lemme suivant (Lemme B.1) nous permet de montrer que si notre fonction est 2-vers-1 et résistance à la collision, alors elle est également à sens unique (nous réutiliserons une version un peu plus générale de ce lemme plus tard, donc nous allons énoncer directement la version générale), ce qui termine la preuve. □

Lemme B.1 (Résistance à la collision vers fonction à sens unique)**.** *Soit $f : X \to Y \cup \perp$ (tel que $\perp \notin Y$), avec A un ensemble fini pouvant être échantillonné aléatoirement et uniformément de manière efficace, et soit C l'ensemble de tous les $y \in Y$ qui admettent deux antécédents. Si la restriction de f à l'ensemble $f^{-1}(Y)$ est une fonction résistante à la collision et si $\frac{|f^{-1}(C)|}{|X|}$ est non-négligeable, alors f restreinte à $f^{-1}(C)$ est une fonction à sens unique.*

Démonstration. Pour prouver ce lemme, on procède par l'absurde : supposons que f restreinte à $f^{-1}(C)$ n'est pas à sens unique, c'est-à-dire qu'avec une probabilité non négligeable on peut trouver un antécédent à $y := f(x)$, avec x choisi uniformément dans $f^{-1}(C)$, et montrons que l'on peut arriver à trouver une collision. L'idée est de tirer au hasard un $x \in X$, et ensuite de calculer $y := f(x)$. Ensuite, étant donné que $\frac{|f^{-1}(C)|}{|A|}$ est non négligeable, on sait qu'avec une probabilité non négligeable y aura deux antécédent. Dans ce cas, toujours avec une probabilité non négligeable, cette fonction sera simple à inverser, et on obtiendra alors un x'. Étant donné que l'on a tiré x de manière uniforme, alors on a la même probabilité de tirer un antécédent ou l'autre, tandis que le y donné à l'adversaire est le même dans les deux cas. Ainsi, avec une probabilité de $1/2$, $x' \neq x$: on a trouvé une collision. Si l'une des étapes a échoué, on recommence à partir de zéro : étant donné que chaque étape a une probabilité non négligeable de réussir, alors en un temps polynomial on trouve une collision dans $f^{-1}(Y)$, ce qui est absurde car la fonction f est censée être résistante à la collision. □

C Preuve de Lemme 3.3

On définit ainsi notre fonction :

Définition C.1. *Pour un ensemble de paramètres $\mathcal{P}$ choisis comme dans le Lemme 3.3, nous définissons les fonctions suivantes, similaires à la construction définie Théorème 3.2, mis à part que la génération des clés nécessite une erreur choisie aléatoirement dans un ensemble plus petit :*

$\texttt{REG2.Gen}_{\mathcal{P}}(1^n)$

1 : $\mathbf{A}, \mathbf{R} \leftarrow \texttt{MP11.Gen}_{\mathcal{G}}(1^n)$
2 : $\mathbf{s}_0 \leftarrow \mathbb{Z}_q^{n,1}$
3 : $\mathbf{e}_0 \leftarrow \mathcal{D}_{\alpha' q}^{m,1}$
4 : $\mathbf{b}_0 := \mathbf{A}\mathbf{s}_0 + \mathbf{e}_0$
5 : $k := (\mathbf{A}, \mathbf{b}_0)$
6 : $t_k := (\mathbf{R}, (\mathbf{s}_0, \mathbf{e}_0))$
7 : **return** (k, t_k)

$\texttt{REG2.Eval}_{\mathcal{P}}((\mathbf{A}, \mathbf{b}_0), (\mathbf{s}, \mathbf{e}, c))$

1 : // $\mathbf{s}$ *est un élément aléatoire dans* $\mathbb{Z}_q^{n,1}$, $c \in \{0, 1\}$
2 : // $\mathbf{e}$ *uniformément choisit tel que chaque*
3 : // *composant soit plus petit que* μ
4 : **return** $\mathbf{A}\mathbf{s} + \mathbf{e} + c \times \mathbf{b}_0$

$\texttt{REG2.Inv}_{\mathcal{P}}((\mathbf{A}, \mathbf{R}, (\mathbf{s}_0, \mathbf{e}_0)), b)$

1 : $(\mathbf{s}_1, \mathbf{e}_1) := \texttt{LWE.Inv}(\mathbf{R}, \mathbf{b})$
2 : **if** $||\mathbf{e}_1 - \mathbf{e}_0||_\infty \leq \mu$ **then return** $\perp$
3 : **return** $((\mathbf{s}_1, \mathbf{e}_1, 0), (\mathbf{s}_1 - \mathbf{s}_0, \mathbf{e}_1 - \mathbf{e}_0, 1))$

Par la suite, nous dénoterons par $f(\mathbf{s}, \mathbf{e}, c)$, la fonction $\texttt{REG2.Eval}_{\mathcal{P}}(k, (\mathbf{s}, \mathbf{e}, c))$ avec k un indice de fonction obtenu par $\texttt{REG2.Gen}_{\mathcal{P}}(1^n)$, et par $\mathbf{s}_0, \mathbf{e}_0$ la partie de la porte dérobée associée à cette fonction.

Nous allons maintenant prouver séparément le fait que cette fonction est δ-2-vers-1, résistante aux collisions et à porte dérobée.

C.1 δ-2-vers-1

Ici nous décrivons comment obtenir la propriété δ-2-vers-1. Ceci revient à prouver que les deux entrées de la fonction $(\mathbf{s}, \mathbf{e})$ and $(\mathbf{s} - \mathbf{s}_0, \mathbf{e} - \mathbf{e}_0)$ appartiennent tous deux au domaine d'entrée de la fonction. Comme le premier élément du couple doit seulement appartenir à Z_q^n, et comme Z_q est fermé par soustraction modulo q, pour tout $\mathbf{s}, \mathbf{s}_0 \in \mathbb{Z}_q^n$, on a $s - s_0 \in Z_q^n$. Ensuite, le second élément du couple doit appartenir à Z_q^m tel que chaque composant du vecteur soit borné en valeur absolue par une valeur μ. Dans ce cas, il n'y a aucune garantie pour qu'ajouter ou soustraire deux éléments résulte en un élément toujours dans le même domaine, et nous voulons nous assurer pour qu'avec une probabilité (suivant le choix de (s, e) and (s_0, e_0)) au moins constante (par rapport à m), le résultat $(s - s_0, e - e_0)$ reste dans le domaine de définition.

Il n'est pas difficile de montrer que si (s_0, e_0) est choisit dans le même ensemble de définition que la fonction, alors $(s - s_0, e - e_0)$ reste dans le domaine de définition de la fonction avec une probabilité exponentiellement petite. C'est pour cette raison que nous avons besoin de restreindre e_0 à être dans un sous ensemble du domaine de définition. Par un choix judicieux de ce sous domaine, nous pouvons obtenir une probabilité de succès (d'avoir deux antécédents) – vue comme une fonction de m – au moins aussi grande qu'une fonction constante.

Premièrement, nous remarquons que la probabilité exacte de succès peut-être calculée explicitement. En effet, si la porte dérobée e_0 est choisie à partir

d'une Gaussienne de dimension m et d'écart type σ, et si le bruit e_1 est choisit uniformément dans un hypercube C de longueur 2μ (les deux distributions étant centrées en 0) alors la probabilité que $e_0 + e_1$ reste dans C est :

$$\left(\operatorname{erf}\left(\frac{\sqrt{2}\mu}{\sigma}\right) - \frac{\sigma}{\sqrt{2\pi}\mu}\left(1 - \exp\left(-2\left(\frac{\mu}{\sigma}\right)^2\right)\right)\right)^m$$

Cependant, pour des raisons de simplicité, et étant donné que nous n'avons pas pour but de trouver des paramètres optimaux, nous allons utiliser une borne inférieure plus simple de cette probabilité (qui sera moins efficace par un facteur de $\sqrt{m}$). Pour cela, nous remarquons que en utilisant [Reg05, Lemme 2.5], nous savons que si $e_0 \in \mathbb{Z}^m$, tel que chaque composant de e_0 soit choisit suivant une Gaussienne d'écart type $\alpha' q$, alors chaque composant du vecteur e_0 est plus petit que $\mu' := \alpha' q\sqrt{m}$ avec une écrasante probabilité quand m augmente. Ainsi, à un terme négligeable près, la distribution Gaussienne d'écart type $\alpha' q$ est "plus proche de 0" que la distribution uniforme sur $[-\alpha' q\sqrt{m}; \alpha' q\sqrt{m}]$ pour m suffisamment large (c'est à dire, pour tout x, l'intégrale entre $-x$ et x de la distribution Gaussienne est plus grande, à un terme négligeable près, que l'intégrale de la distribution uniforme). Ainsi, pour obtenir un borne inférieure sur la probabilité d'avoir deux antécédents, on peut considérer que e_0 est choisit suivant un distribution uniforme sur un hypercube de longueur $2\alpha' q\sqrt{m}$ plutôt que suivant une distribution Gaussienne d'écart type $\alpha' q$. Ceci simplifie notre analyse, et nous permet de trouver un sous ensemble dans lequel e_0 doit se situer, comme indiqué dans le lemme suivant. Remarquez que si l'on ne fait aucune hypothèse sur la distribution d'entrée, excepté le fait que sa norme infinie est plus petite que μ', le même Lemme s'applique avec la constante 4 remplacée par 2.

Lemme C.1 (Addition des domaines)**.** *Soit $V = \mathbb{R}^m$ un espace vectoriel de dimension m, et soit $\mathcal{D}_{m,\mu}$ la distribution uniforme à l'intérieur de l'hypercube de dimension m et de longueur 2μ centré en 0. Alors pour tout $\mu' < \mu$, nous avons :*

$$P_{m,\mu,\mu'} := \Pr\left[||e_0 + e_1||_\infty \leq \mu \mid e_0 \leftarrow \mathcal{D}_{m,\mu'}, e_1 \leftarrow \mathcal{D}_{m,\mu}\right] = \left(1 - \frac{\mu'}{4\mu}\right)^m$$

De plus, si $\mu' = O(\frac{\mu}{m})$ alors la probabilité $P_{m,\mu,\mu'}$ devient minorée par une constante positive.

Démonstration. Comme $||e_0 + e_1||_\infty$ doit être inférieur à μ, on doit donc avoir chaque composant du vecteur résultant inférieur à μ. De plus, comme chaque composant est tiré indépendamment des autres, nous pouvons simplifier notre analyse en considérant que e_0 and e_1 sont des vecteurs dans $\mathbb{R}$, permettant de déterminer $P_{1,\mu,\mu'}$ et ensuite on peut calculer $P_{m,\mu,\mu'} = {P_{1,\mu,\mu'}}^m$. Ensuite, soit E_1 la variable aléatoire uniforme sur $[-\mu, \mu]$, E_0 la variable

aléatoire uniforme sur $[-\mu', \mu']$ et E la variable aléatoire $E = E_1 + E_0$. Par conséquent, $P_{1,\mu,\mu'} = Pr[-\mu \leq E \leq \mu]$.
Maintenant, nous pouvons calculer la fonction de densité de E en utilisant des convolutions :

$$f_E(e) = \int_{-\infty}^{\infty} f_{E_1}(e_1) \cdot f_{E_0}(e - e_1)\, de_1$$

où f_{E_1} et f_{E_0} sont les densités de probabilité de E_1 et E_0 ($f_{E_1}(e_1) = \frac{1}{2\mu}$ quand $e_1 \in [-\mu, \mu]$ et 0 sinon, et $f_{E_0}(e_0) = \frac{1}{2\mu'}$ quand $e_0 \in [-\mu', \mu']$ et 0 sinon).
Ainsi, nous nous intéressons seulement aux cas où les deux valeurs de $f_{E_1}(e_1)$ et $f_{E_0}(e - e_1)$ sont non nulles et pour cela nous avons besoin de considérer 3 cas pour e, suivant sont appartenance aux intervalles suivants : $e \in [-\mu - \mu', \mu' - \mu] \bigcup [\mu' - \mu, \mu - \mu'] \bigcup [\mu - \mu', \mu + \mu']$. Nous avons ainsi :

$$f_E(e) = \begin{cases} \int_{-\mu}^{e+\mu'} \frac{1}{4\mu\mu'}\, de_1 \ , \ e \in [-\mu - \mu', \mu' - \mu] \\ \int_{e-\mu'}^{e+\mu'} \frac{1}{4\mu\mu'}\, de_1 \ , \ e \in [\mu' - \mu, \mu - \mu'] \\ \int_{e-\mu'}^{\mu} \frac{1}{4\mu\mu'}\, de_1 \ , \ e \in [\mu - \mu', \mu + \mu'] \end{cases}$$

Pour conclure, nous rassemblons ces trois intervalles : $Pr[-\mu \leq E \leq \mu] = \int_{-\mu}^{\mu} f_E(e)\, de = \int_{-\mu}^{-\mu+\mu'} f_E(e)\, de + \int_{-\mu+\mu'}^{\mu-\mu'} f_E(e)\, de + \int_{\mu-\mu'}^{\mu} f_E(e)\, de = \int_{-\mu}^{-\mu+\mu'} \frac{e+\mu'+\mu}{4\mu\mu'}\, de + \int_{-\mu+\mu'}^{\mu-\mu'} \frac{1}{2\mu}\, de + \int_{\mu-\mu'}^{\mu} \frac{\mu+\mu'-e}{4\mu\mu'}\, de = 1 - \frac{\mu'}{4\mu}$.

Ce qui nous donne $P_{m,\mu,\mu'} = (1 - \frac{\mu'}{4\mu})^m$.

Maintenant, étant donné que μ est une fonction dépendant de m ($\mu = \mu(m)$), nous souhaitons déterminer les valeurs de μ' telles que cette probabilité (vue comme fonction de m) soit au moins aussi grande qu'une constante positive.

— Si $\lim_{m\to\infty} \dfrac{\mu'}{\mu} = 0$, alors :

$$\lim_{m\to\infty} \left(1 - \frac{\mu'}{4\mu}\right)^m = \lim_{m\to\infty} \left(1 - \frac{\mu'}{4\mu}\right)^{\frac{4\mu}{\mu'}\frac{\mu' m}{4\mu}} = \left(\frac{1}{e}\right)^{\lim_{m\to\infty} \frac{\mu' m}{4\mu}}$$

Ainsi, nous avons besoin d'avoir $\lim_{m\to\infty} \dfrac{\mu' m}{4\mu} = c \geq 0$, avec c une constante, puisque ainsi, la probabilité de succès est au moins une constante supérieure à $\left(\frac{1}{e}\right)^c$.

— Si $\lim_{m\to\infty} \dfrac{\mu'}{\mu} > 0$ (et inférieure à 1, comme $0 < \mu' < \mu$), alors :

$$\lim_{m\to\infty} \left(1 - \frac{\mu'}{4\mu}\right)^m = 0$$

Ainsi, il est clair que pour pouvoir minorer la probabilité par une constante strictement positive, on doit avoir :

$$\mu' = c \cdot \frac{4\mu}{m}, \quad c \geq 0$$

□

Par conséquent, dans notre cas, si e_1 est choisit uniformément dans un hypercube de longueur 2μ et si e_0 est choisit selon une Gaussienne d'écart type $\alpha' q$, en remplaçant la valeur actuelle de $\mu = \alpha q\sqrt{m}$ et $\mu' := \alpha' q\sqrt{m}$, nous avons besoin d'avoir :

$$\alpha' = c \cdot \frac{4\alpha}{m}, \quad c \geq 0$$

C.2 Résistance à la collision

Nous commençons par remarquer qu'en utilisant les paramètres définis dans la Définition C.1, aucun adversaire QPT ne peut déterminer l'information de la porte dérobée (s_0, e_0), étant donné que déterminer s_0 à partir de $k = (A, b_0)$ serait équivalent à résoudre le problème $\mathrm{LWE}_{q,\bar{\Psi}_{\alpha' q}}$:

Corollaire C.2 ([Reg05, Theorem 1.1])**.** *Sous l'hypothèse que* SIVP_γ *est dur (avec* $\gamma = \mathsf{poly}(n)$*), aucun adversaire QPT ne peut retrouver les informations de la porte dérobée* (s_0, e_0)*.*

Lemme C.3 (Résistance à la collision)**.** *La fonction* f *définie à la Définition C.1 est résistante à la collision si les paramètres sont choisis suivant le Lemme 3.3, sous l'hypothèse que* SIVP_γ *est dur.*

Démonstration. Par l'absurde, supposons que cette fonction n'est pas résistante à la collision. Alors il existe deux couples (s_1, e_1), (s_2, e_2) tels que $f(s_1, e_1, 0) = y = f(s_2, e_2, 1)$. On remarque que le dernier bit est nécessairement différent puisque les deux fonctions partielles qui fixent le dernier bit sont injectives quand l'erreur est plus petite que r_{max} (d'après [MP11, Theorem 5.4]). Par définition de f, $||e_1||_\infty \leq \mu$ et $||e_2||_\infty \leq \mu$, c'est-à-dire e_1 et e_2 ont tous deux une norme Euclidienne inférieure à $\sqrt{m}\mu$. Ainsi, par définition, $y = f(s_2, e_2, 1) = f(s_2, e_2, 0) + f(s_0, e_0, 0) = A(s_2 + s_0) + (e_2 + e_0)$. Maintenant, nous remarquons qu'avec une probabilité écrasante (sur le choix de la porte dérobée), $||e_0||_2 \leq \mu'\sqrt{m}$ comme indiqué dans [Reg05, Lemme 2.5], donc dans ce cas, $||e_2 + e_0||_2 \leq \sqrt{m}(\mu + \mu') \leq r_{max}$ (dernière hypothèse du Lemme 3.3). Alors, d'après [MP11, Theorem 5.4], il y a exactement un élément (s, e) avec e ayant une longueur inférieure à r_{max} tel que $As + e = y$. Puisque (s_1, e_1) est une solution, alors nous avons : $s_2 + s_0 = s_1$ et $e_2 + e_0 = e_1$, c'est-à-dire $e_0 = e_1 - e_2$ et $s_0 = s_1 - s_2$. Par conséquent, il est possible de déduire s_0 et e_0 à partir d'une collision, ce qui est impossible d'après Corollaire C.2. □

C.3 Porte dérobée

Pour avoir la propriété de porte dérobée, il faut déjà vérifier que la fonction `REG2.Eval` est simple à inverser lorsque l'on possède la porte dérobée (qui consiste en (s_0, e_0) et t_k). Premièrement, on remarque que pour trouver tous les antécédents, il faut simplement lancer la fonction inverse de [MP11] sur b et sur $b - b_0$, et de ne prendre que les inverses qui qui appartiennent au domaine de définition de la fonction, c'est-à-dire quand la norme infinie de l'erreur est bornée par μ : $||e||_\infty \leq \mu$. Puisque cette fonction est injective, ce sont les deux seules antécédents possibles. Cependant, étant donné que nous sommes intéressés seulement par le cas où il y a exactement deux antécédents, et que le terme $-\mu'\sqrt{m}$ nous permet de nous assurer que lorsqu'il y a deux antécédents, alors on a $e_2 = e_1 - e_0$, ceci est équivalent à lancer en premier l'algorithme d'inversion sur b et d'obtenir (s_1, e_1). Ensuite, l'inversion est complétée en retournant $(s_1, e_1, 0)$ et $(s_1 - s_0, e_1 - e_0, 1)$, qui sont deux antécédents valides si et seulement si la fonction a deux antécédents (c.f. le Lemme C.3 pour plus de détails).

Ensuite, pour prouver la propriété de sens unique de la fonction définie dans la Définition C.1, on pourrait imaginer que cette propriété est impliquée par la propriété de sens unique de la fonction dans [MP11]. Cependant, nous devons faire attention ici car dans notre construction le terme d'erreur n'est pas choisit suivant une distribution Gaussienne (contrairement au terme d'erreur e_0). Nous pouvons cependant utiliser le fait que nous avons déjà prouvé que notre fonction est résistante à la collision et utiliser le Lemme B.1 :

Corollaire C.4 (Sens unique à partir du Lemme C.3 et du Lemme B.1)**.** *Quand les paramètres sont choisis suivant le Lemme 3.3, la fonction définie dans Définition C.1 est à sens unique pour tous les y qui admettent deux antécédents, sous l'hypothèse que* SIVP_γ *est difficile.*

D Preuve du Lemme 3.4

Démonstration. En utilisant les valeurs suivantes pour les paramètres de la fonction injective à porte dérobée de Micciancio et Peikert [MP11]

$$\begin{aligned} n &= \lambda \\ k &= 5\lceil \log(n) \rceil + 21 \\ q &= 2^k \\ \bar{m} &= 2n \\ \omega &= nk \\ m &= \bar{m} + \omega \\ \mu &= \left\lceil 2mn\sqrt{2+k} \right\rceil \end{aligned}$$

$$\mu' = \mu/m$$
$$B = 2$$

et α, α', C sont définis comme dans Lemme 3.3, nous allons prouver qu'ils respectent tous les critères du Lemme 3.3 :

— Les trois premiers critères sont satisfaits de manière triviale.

— Dans le quatrième critère, la seule difficulté est de montrer que $\alpha < 1$. Par définition,

$$\begin{aligned}
\alpha = \frac{m\mu}{\sqrt{m}mq} &= \frac{\mu}{\sqrt{m}q} &&= \frac{\lceil 2mn\sqrt{2+k} \rceil}{\sqrt{m}q} &&\leq \frac{4mn\sqrt{2+k}}{\sqrt{m}q} \leq \frac{8mn\sqrt{k}}{\sqrt{m}q} \\
&\leq \frac{8\sqrt{m}nk}{q} \leq \frac{8\sqrt{2n+nk}nk}{2^{21}n^5} &&\leq \frac{8\sqrt{2nk}nk}{2^{21}n^5} &&\leq \frac{16(nk)^{3/2}}{2^{21}n^5} \\
&\leq \frac{16(n(5(\log(n)+1)+21))^{3/2}}{2^{21}n^5} \leq \frac{16(5\times 21n^2)^{3/2}}{2^{21}n^5} \leq \frac{16\times 1076n^3}{2^{21}n^5} &&< \frac{1}{n^2} \leq 1
\end{aligned}$$

— Maintenant, montrons que le cinquième critère est respecté, c'est-à-dire que $\alpha' q \geq 2\sqrt{n}$. Premièrement, remarquons que $\alpha' q := \frac{\mu}{\sqrt{m}m} \geq 2\sqrt{n} \Leftrightarrow \mu \geq 2\sqrt{n}m\sqrt{m} = 2mn\sqrt{2+k}$. Ensuite, par définition $\mu = \lceil 2mn\sqrt{2+k} \rceil$, la condition est donc satisfaite.

— Pour le sixième critères, c'est-à-dire que $\frac{n}{\alpha'}$ est $\mathsf{poly}(n)$, on doit juste remarquer que $1/\alpha' = \frac{m^{3/2}q}{\mu} < m^{3/2}q$, et que m et q sont tous deux $\mathsf{poly}(n)$.

— Finallement, pour montrer que la dernière condition est satisfaite, on remarque que :

$$\sqrt{m}\mu < \frac{q}{2B\sqrt{\left(C\cdot(\alpha\cdot q)\cdot(\sqrt{2n}+\sqrt{kn}+\sqrt{n})\right)^2+1}} - \mu'\sqrt{m} \quad (1)$$
$$= \frac{q}{4\sqrt{\left(C\cdot\frac{\mu}{\sqrt{m}}\cdot(\sqrt{2n}+\sqrt{kn}+\sqrt{n})\right)^2+1}} - \frac{\mu}{\sqrt{m}} \quad (2)$$

si et seulement si

$$A := 4\left(\sqrt{m}+\frac{1}{\sqrt{m}}\right)\mu\sqrt{\left(C\cdot\frac{\mu}{\sqrt{m}}\cdot(\sqrt{2n}+\sqrt{kn}+\sqrt{n})\right)^2+1} \leq q$$

Maintenant, supposons que $k := u\lceil\log(n)\rceil + v$ avec $u \leq 5$ et $v \geq 19$, et essayons de trouver u, v tels que $A \leq 2^k$. Remarquons que nous allons inclure v dans certaines constantes, et nous chercherons à la fin quelle valeur donner à v : Premièrement remarquons que :

$$\sqrt{m}+\frac{1}{\sqrt{m}} = \sqrt{m}(1+\frac{1}{m}) \quad (3)$$

$$= \sqrt{m}(1 + \frac{1}{n(2+k)}) \tag{4}$$

$$\leq \sqrt{m}(1 + \frac{1}{2+k}) \tag{5}$$

$$\leq \sqrt{m}\underbrace{(1 + \frac{1}{2+v})}_{\gamma_0} = \gamma_0\sqrt{m} \tag{6}$$

Donc maintenant :

$$A \leq 4C\gamma_0\mu^2\sqrt{kn}\sqrt{\left(1 + \sqrt{\frac{2}{k}} + \frac{1}{\sqrt{k}}\right)^2 + \frac{1}{kn\left(C \cdot \frac{\mu}{\sqrt{m}}\right)^2}}$$

$$= 4C\gamma_0\left\lceil 2mn\sqrt{2+k}\right\rceil^2\sqrt{kn}\sqrt{\left(1 + \sqrt{\frac{2}{k}} + \frac{1}{\sqrt{k}}\right)^2 + \frac{1}{kn\left(C \cdot \frac{\lceil 2mn\sqrt{2+k}\rceil}{\sqrt{m}}\right)^2}}$$

$$\leq 4C\gamma_0\left\lceil 2mn\sqrt{2+k}\right\rceil^2\sqrt{kn}\underbrace{\sqrt{\left(1 + \sqrt{\frac{2}{v}} + \frac{1}{\sqrt{v}}\right)^2 + \frac{1}{v\left(2C\sqrt{2+v}\right)^2}}}_{\gamma_1}$$

$$\leq 4C\gamma_0\gamma_1\left(2mn\sqrt{2+k} + 1\right)^2\sqrt{kn}$$

$$= 4C\gamma_0\gamma_1\left(2n^2(2+k)^{3/2} + 1\right)^2\sqrt{kn}$$

$$= 16C\gamma_0\gamma_1 n^4(2+k)^3\left(1 + \frac{1}{2n^2(2+k)^{3/2}}\right)^2\sqrt{kn}$$

$$\leq 16C\gamma_0\gamma_1 n^4(2+k)^3\underbrace{\left(1 + \frac{1}{2(2+v)^{3/2}}\right)^2}_{\gamma_2}\sqrt{kn}$$

$$\leq 16C\gamma_0\gamma_1\gamma_2 n^4\left(k\left(1 + \frac{2}{k}\right)\right)^3\sqrt{kn}$$

$$\leq 16C\gamma_0\gamma_1\gamma_2 n^4k^3\underbrace{\left(1 + \frac{2}{v}\right)^3}_{\gamma_3}\sqrt{kn}$$

$$\leq 16C\gamma_0\gamma_1\gamma_2\gamma_3 n^{9/2}\left(u\lceil\log(n)\rceil + v\right)^{7/2}$$

$$= 16C\gamma_0\gamma_1\gamma_2\gamma_3 n^{9/2}v^{7/2}\left(1 + \frac{u\lceil\log(n)\rceil}{v}\right)^{7/2}$$

$$\leq 16C\gamma_0\gamma_1\gamma_2\gamma_3 n^{9/2}v^{7/2}\left(1 + \frac{5\lceil\log(n)\rceil}{19}\right)^{7/2}$$

$$\leq 16C\gamma_0\gamma_1\gamma_2\gamma_3 n^{9/2}v^{7/2}3n^{1/2}$$

$$\leq 48C\gamma_0\gamma_1\gamma_2\gamma_3 v^{7/2} n^5$$

Finalement, nous observons que si $v = 21$ et $u = 5$, nous avons $A \leq 2^{v+u\lceil \log(n) \rceil} = 2^k$, ce qui conclut la preuve.

□