Cryptographie à clef publique
Cours 8

Julien Lavauzelle

Université Paris 8

Master 1 mathématiques et applications – parcours ACC

18/03/2024

Séance précédente

Vu à la **séance précédente** :

- Protocoles d'identification
- Signature de Schnorr
- Chiffrement basé sur l'identité : Cocks (+ Boneh–Franklin)

Questions ?

Aujourd'hui

1. Cryptographie post-quantique

L'ordinateur quantique : un nouveau modèle de calcul
Une cryptographie post-quantique ?

2. Cryptographie basée sur les réseaux euclidiens

Fondements mathématiques
NTRU : un premier schéma de chiffrement
Chiffrement fondé sur le problème LWE

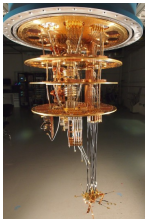
Plan

Plan

Le modèle de calcul quantique

Modèle de calcul **classique** : machine de Turing, opérations effectuées sur des séquences de bits (éléments de $\{0,1\}^n$) avec des opérateurs binaires.

À partir des années 80, introduction d'un nouveau modèle de calcul **quantique**



- $\rightarrow$ opérations sur des états quantiques (**qbits**) modélisés par des éléments $|x\rangle \in \mathbb{C}^{2^n}$,
- $\rightarrow$ avec des **opérateurs logiques quantiques** (au lieu des `xor`, `and`, etc.)
- $\rightarrow$ constructions récentes de processeurs quantiques ($<$ 100 qbits en 2020)
- $\rightarrow$ toujours moins efficace que les machines classiques, mais en progression rapide

année	factorisation de...
2012	21
2016	200 099
2019	1 099 551 473 989

Idée informelle : alors qu'un bit porte l'information d'une **valeur** de $\{0,1\}$, un état quantique porte l'information d'une **distribution** de probabilité sur $\{0,1\}$.

Avantage : on peut effectuer un calcul identique sur une **superposition d'états** en temps constant.
$\rightarrow$ exemple : transformée de Fourier discrète essentiellement en temps $O(1)$.

Le modèle de calcul quantique

D'un point de vue des **classes de complexité**, on a

$$\mathsf{P} \subseteq \mathsf{BPP} \subseteq \mathsf{BQP} \subseteq \mathsf{PSPACE}\,.$$

P : problèmes décidables en temps polynomial de manière déterministe
BPP : problèmes décidables en temps polynomial avec probabilité $\geq 2/3$
BQP : problèmes décidables en temps polynomial quantique avec probabilité $\geq 2/3$
PSPACE : problèmes décidables en espace polynomial

On conjecture également que $\mathsf{BQP} \neq \mathsf{BPP}$, et que $\mathsf{BQP} \neq \mathsf{NP}$.

Remarque : le modèle quantique apporte aussi des contraintes. Par exemple, il n'existe pas de porte quantique qui permette de copier un état (*no-cloning theorem*).

Référence pour (beaucoup) plus de détails : notes de cours de R. de Wolf (CWI, Univ. Amsterdam)

`https://homepages.cwi.nl/~rdewolf/qcnotes.pdf`

Quelques problèmes avec une accélération quantique (1)

Problème de Deutsch-Josza. Soit $N = 2^n$.

Instance. Un vecteur $\boldsymbol{x} \in \{0,1\}^N$ qui est

1. ou bien constant ($\forall i, j,\ x_i = x_j$)
2. ou bien équilibré ($\sum_i x_i = N/2$).

But. Distinguer si $\boldsymbol{x}$ est constant ou équilibré.

Dans un **modèle de calcul classique**, combien d'opérations sont nécessaires pour résoudre le problème de manière déterministe ?

- Il faut lire strictement plus de la moitié des bits de $\boldsymbol{x}$ pour différencier les deux cas (avec la moitié des bits, les deux cas restent possibles).
- Donc, $N/2 + 1$ opérations sont nécessaires.

Dans un modèle de calcul **quantique**, il existe un algorithme avec 1 **seule opération quantique** et $O(n) = O(\log_2(N))$ autres opérations (classiques).

Quelques problèmes avec une accélération quantique (2)

Problème de Simon.

Instance. Une fonction $f : \{0,1\}^n \to \{0,1\}$ pour laquelle il existe une **période**, c'est-à-dire un vecteur $s \in \{0,1\}^n$ tel que

$$\forall x \in \{0,1\}^n,\ f(x \oplus s) = f(x)\,.$$

But. Trouver la période s de f.

Théorème (Simon) :

1. Tout algorithme **classique** (même probabiliste) nécessire $\Omega(\sqrt{2^n})$ opérations.
2. Par ailleurs , il existe un algorithme **quantique** avec $O(n)$ **opérations quantiques** et $O(n^4)$ autres opérations.

Quelques problèmes avec accélération quantique (3)

Un autre problème admet une **accélération quantique moindre**, mais avec beaucoup d'**impact en cryptographie**.

Problème de recherche. Soit $N = 2^n$.

Instance. Un vecteur $\boldsymbol{x} \in \{0,1\}^N$ différent de $\mathbf{0}$.

But. Trouver $i \in \{1, \ldots, N\}$ tel que $x_i = 1$.

Complexité de la résolution.

- Dans le pire cas, un algorithme classique nécessite $\Omega(N)$ opérations.
- L'algorithme de **Grover** (1996) nécessite $O(\sqrt{N})$ opérations quantiques.

Application. Essentiellement, cela réduit de moitié la complexité de l'attaque exhaustive sur les clés de chiffrement. Conséquences pratiques pour les chiffrements symétriques et les fonctions de hachage.
$\rightarrow$ exemple : AES-256 a une sécurité quantique $\leq$ 128 bits.

Plan

Cryptanalyse quantique

En 1994, Shor présente un algorithme qui permet de résoudre les problèmes de la factorisation et du logarithme discret, en **temps polynomial dans le modèle de calcul quantique**.

Algorithms for Quantum Computation : Discrete Logarithms and Factoring. P. Shor. FOCS. **1994**.

Idée très informelle : pour factoriser $n = pq$, on peut chercher un élement $x \in (\mathbb{Z}/n\mathbb{Z})^\times$ d'ordre pair.

$$\begin{aligned} x^r \equiv 1 \mod n &\iff (x^{r/2} - 1)(x^{r/2} + 1) \equiv 0 \mod n \\ &\iff \operatorname{pgcd}(x^{r/2} - 1, n) \neq 1 \text{ ou } \operatorname{pgcd}(x^{r/2} + 1, n) \neq 1 \end{aligned}$$

La fonction $f_x : i \mapsto x^i \mod n$ admet alors une **période** r, que l'on peut chercher en adaptant l'algorithme de Simon.

Conséquence. Tous les cryptosystèmes dont la sécurité repose sur la difficulté de la factorisation (par conséquent, également sur la résiduosité quadratique) ou du logarithme discret sont **vulnérables dans un modèle de calcul quantique**.

On cherche donc de nouveaux systèmes « post-quantiques », c'est-à-dire qui se fondent sur des problèmes **difficiles dans un modèle de calcul quantique**.

L'appel à standardisation du NIST

En 2017, appel à **standardisation de primitives post-quantiques** du NIST (*National Institute of Standards and Technology*).

- → 2 compétitions : (i) chiffrement & encapsulation de clé, (ii) signature
- → + de 60 propositions au premier tour
- → en 2023 :
  - un chiffrement a été **sélectionné** pour une standardisation : CRYSTALS-KYBER (dont le sécurité repose sur un problème de réseaux euclidiens) ;
  - ~~4~~ 3 chiffrements restent à l'étude pour ajouter de la diversité : BIKE (codes), Classic McEliece (codes), HQC (codes), ~~SIKE~~ (isogénies)
  - 3 signatures ont été pré-sélectionnées pour standardisation : CRYSTALS-DILITHIUM (réseaux), Falcon (réseaux), SPHINCS+ (hash-based)
  - la signature ~~Rainbow~~ (multivarié), prometteuse, a été récemment cassée
  - nouvel appel pour les signatures (pas assez de diversité !) en 2023. Liste des candidats : `https://csrc.nist.gov/Projects/pqc-dig-sig/round-1-additional-signatures`

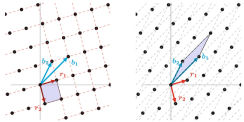
Processus de désignation public, détails ici :

`https://csrc.nist.gov/Projects/post-quantum-cryptography`

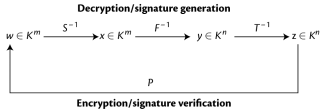
Familles de problèmes

Parmi les **propositions** retenues :

Réseaux euclidiens
(NTRU, CRYSTALS, FALCON, SABER...)

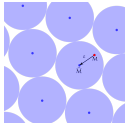


Systèmes polynomiaux multivariés
(Rainbow, ...)



Graphes d'isogénies
(SIKE)

Codes correcteurs
(Classic McEliece, ...)

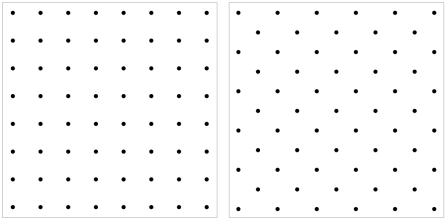


Plan

Plan

Définition. Un **réseau euclidien** (*lattice* en anglais) de dimension n est un **sous-groupe discret** $\mathcal{L}$ de $(\mathbb{R}^n, +)$.

Deux réseaux dans $\mathbb{R}^2$:



Exemples.

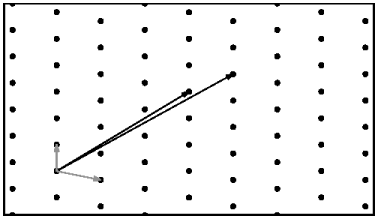
- L'ensemble $\mathbb{Z}^n$.
- Si $\mathcal{L}$ est un réseau, alors pour toute matrice $\boldsymbol{G} \in \mathrm{GL}_n(\mathbb{R})$,

$$\boldsymbol{G}\mathcal{L} \coloneqq \{\boldsymbol{G}\boldsymbol{x} \mid \boldsymbol{x} \in \mathcal{L}\}$$

  est aussi un réseau.

Autres définitions.

- Une **base** d'un réseau est un ensemble de vecteurs libres engendrant le réseau.



deux bases (une bonne et une mauvaise) pour un même réseau

- Le **rang** du réseau $\mathcal{L}$ est le nombre d'éléments dans une base de $\mathcal{L}$. On supposera souvent que le rang vaut n.
- On munit les éléments de $\mathbb{R}^n$ de la **norme** euclidienne : $\|\boldsymbol{x}\|_2 = \sqrt{\sum_{i=1}^{n} x_i^2}$.
- La **distance minimale** d'un réseau est $\lambda_1(\mathcal{L}) := \min_{v \in \mathcal{L} \setminus \{\mathbf{0}\}} \|\boldsymbol{v}\|_2$.
- Si $\mathcal{L} \subseteq \mathbb{Z}^n$, le **réseau orthogonal** à $\mathcal{L}$, noté $\mathcal{L}^\perp$, est l'ensemble des élements $v \in \mathbb{Z}^n$ tels que pour tout $\boldsymbol{x} \in \mathcal{L}$, on a $\sum_{i=1}^{n} x_i v_i = 0$.

Problèmes difficiles : vecteurs courts

On note maintenant, et dans toute la suite, $\mathbb{Z}_q \coloneqq \mathbb{Z}/q\mathbb{Z}$. Il faut généralement penser q de taille modérée (quelques centaines ou milliers), et souvent premier (pas toujours).

En cryptographie à clef publique, on a besoin de **problèmes difficiles**. Les problèmes de recherche suivants sont NP-difficiles.

Problème SVP (*shortest vector problem*), problème du **vecteur le plus court**.

Instance. Une base **quelconque** $\boldsymbol{B}$ d'un réseau $\mathcal{L} \subseteq \mathbb{Z}_q^n$.

But. Trouver $v \in \mathcal{L}$ tel que $\|v\|_2 = \lambda_1(\mathcal{L})$.

Problème SVP_γ (*approximate shortest vector problem*), problème d'**approximation du vecteur le plus court**. Étant donné $\gamma(n) > 1$:

Instance. Une base **quelconque** $\boldsymbol{B}$ d'un réseau $\mathcal{L} \subseteq \mathbb{Z}_q^n$.

But. Trouver $v \in \mathcal{L}$ tel que $\|v\|_2 \leq \gamma(n) \cdot \lambda_1(\mathcal{L})$.

On peut également définir le **problème décisionnel** associé : le problème GapSVP_γ, qui consiste à **distinguer si un réseau possède un vecteur de norme** ≤ 1 **ou** $\geq \gamma(n)$.

$\rightarrow$ utile pour les réductions de sécurité (IND-CPA).

Problème CVP (*closest vector problem*), problème du **vecteur le plus proche**.

Instance. Une base $\boldsymbol{B}$ d'un réseau $\mathcal{L} \subseteq \mathbb{Z}_q^n$, un vecteur $\boldsymbol{x} \in \mathbb{Z}_q^n$.

But. Trouver $\boldsymbol{v} \in \mathcal{L}$ tel que $\|\boldsymbol{v} - \boldsymbol{x}\|_2$ est le plus petit possible.

Problème SIS_β (*short integer solution*), problème de la « solution entière courte » de paramètre β.

Instance. m vecteurs $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m \in \mathbb{Z}_q^n$

But. Trouver $\boldsymbol{z} = (z_1, \ldots, z_m) \in \mathbb{Z}_q^m$ de poids $\|z\|_2 \leq \beta$ tel que

$$z_1 \boldsymbol{a}_1 + \cdots + z_m \boldsymbol{a}_m = \boldsymbol{0}\,.$$

Question (légitime). En quoi SIS est un problème de réseau ?

Soit $\mathcal{L}$ le réseau défini par la matrice

$$\boldsymbol{A} = \Big(\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m\Big)$$

On cherche un vecteur $\boldsymbol{z} \in \mathbb{Z}_q^m$ tel que $\boldsymbol{A}\boldsymbol{z} = \boldsymbol{0}$. Autrement dit, on cherche un **vecteur court** dans le réseau orthogonal $\mathcal{L}^\perp$.

Réduction pire cas – cas moyen

Les problèmes de réseau sont également équipés de certaines **réductions « pire cas – cas moyen »**.

Par exemple :

Théorème de réduction pire cas – cas moyen (Ajtai 1996, expression informelle). Soit $\mathcal{L}$ un réseau quelconque de $\mathbb{Z}_q^n$ et $\beta \ll q$.
Supposons que l'on connaisse un algorithme qui résolve SIS_β avec bonne probabilité sur une instance aléatoire.
Alors on peut résoudre $\mathsf{GapSVP}_{\beta\sqrt{n}}$ dans tout réseau de dimension n.

Conséquence. Cela a un intérêt fort en cryptographie, car les clés sont tirées aléatoirement.

On veut que le problème qui assure la sécurité d'une clé soit « presque aussi difficile » dans le cas moyen (tirage aléatoire) que dans le pire cas (clé de meilleure sécurité).

Pour attaquer les problèmes SVP, SIS, etc., on peut essayer de transformer la base en entrée en une meilleure base.

Algorithme LLL, pour Lenstra, Lenstra, Lovasz, publié en 1982.

Étant donnée une base quelconque $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n)$ d'un réseau $\mathcal{L} \subseteq \mathbb{R}^n$, calcule une autre base (dite **base LLL-réduite**) de $\mathcal{L}$, qui est « **presque orthogonale** » et **assez courte**.

Propriétés.

1. L'algorithme LLL termine en temps polynomial, précisément $O(n^6 \log^3(\max \|\boldsymbol{b}_i\|_2))$.
2. Le vecteur $\widetilde{\boldsymbol{b}}_1$ de la base produite par LLL a pour norme

$$\left\|\widetilde{\boldsymbol{b}}_1\right\|_2 \leq 2^{(n-1)/2} \lambda_1(\mathcal{L}) \,.$$

Remarques. Cela reste exponentiellement loin de la longueur minimale $\lambda_1(\mathcal{L})$.

$\Longrightarrow$ pas une solution optimale en temps polynomial pour SVP, SIS, etc.

Remarque : nous étudierons LLL dans le cours d'Algorithmes Arithmétiques II.

Plan

Contexte mathématique. Soient N, p, q des entiers.

On cosidère les éléments de l'anneau de polynômes $\mathcal{R} := \mathbb{Z}_q[X]/(X^N - 1)$. Ils ont des représentants uniques sous la forme

$$A(X) = a_0 + a_1 X + \cdots + a_{N-1} X^{N-1}.$$

On note $\boldsymbol{a} := (a_0, \ldots, a_{N-1}) \in \mathbb{Z}_q^N$ le vecteur correspondant.

On définit $\boldsymbol{c} := \boldsymbol{a} \star \boldsymbol{b}$ comme le vecteur correspondant au produit $A(X)B(X)$ dans $\mathcal{R}$:

$$A(X)B(X) = C(X) \in \mathcal{R} \iff \boldsymbol{a} \star \boldsymbol{b} = \boldsymbol{c} \in \mathbb{Z}_q^N$$

Remarque. On associe souvent au vecteur $\boldsymbol{a} \in \mathbb{Z}_q^n$ la matrice circulante

$$\boldsymbol{M}(\boldsymbol{a}) = \begin{pmatrix} a_0 & a_1 & & & a_{N-1} \\ a_{N-1} & a_0 & a_1 & & a_{N-2} \\ a_{N-2} & \ddots & \ddots & & \\ & & \ddots & \ddots & \\ & & & \ddots & \\ a_1 & a_2 & & a_{N-1} & a_0 \end{pmatrix}.$$

Intérêts :

- calculatoire
- permet de représenter $\star$ comme un produit vecteur-matrice :

$$\boldsymbol{a} \star \boldsymbol{b} = \boldsymbol{a}\boldsymbol{M}(\boldsymbol{b})$$

On veut maintenant associer une **norme euclidienne** à des éléments $A(X) \in \mathcal{R}$, donc à des "vecteurs" $\boldsymbol{a} \in \mathbb{Z}_q^N$.

$$\|\boldsymbol{a}\| = \sqrt{\sum_{i=0}^{N-1} a_i^2}$$

Problème : la norme dépend du choix du représentant de $a_i \bmod q$...

Par exemple, pour $q = 101$, on a $\sqrt{1^2} = 1 \neq \sqrt{(-100)^2} = 100$.

On choisit une **représentation « centrée »** :

$$\mathbb{Z}_q \coloneqq \left\{ \lceil -\tfrac{q-1}{2} \rceil, \dots, \lfloor \tfrac{q-1}{2} \rfloor \right\}$$

Exemple (important) :

$$\mathbb{Z}_3 \coloneqq \{-1, 0, 1\}$$

Le système NTRU : paramètres du système

Le cryptosystème **NTRUEncrypt**, souvent abrégé **NTRU** (*N-th degree Truncated polynomial Ring Units*) a été proposé en 1996 par Hoffstein, Pipher et Silverman.

On va présenter une **première version** de NTRU. Il y a eu beaucoup de variantes par la suite, pour améliorer certains points de sécurité.

Les valeurs de N, p et q sont des **paramètres publics** tels que p et q sont premiers entre eux.
Typiquement, N vaut plusieurs centaines, et $q \gg p$, avec par exemple q de plusieurs milliers et $p = 3$.

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

Pour d_1, d_2 tels que $d_1 + d_2 \leq N$, on note $\mathcal{L}(d_1, d_2)$ le sous-ensemble de $\mathcal{R}$ ayant :

- exactement d_1 coefficients à 1,
- exactement d_2 coefficients à -1,
- le reste des coefficients à 0.

On considère enfin d_f, d_g et d trois entiers strictement inférieurs $N/2$ (également des **paramètres publics**).

NTRU : GÉNÉRATION DE CLEFS

1. Tirer uniformément $F(X) \in \mathcal{L}(d_f, d_f - 1)$.
2. Tester si $F(X)$ est inversible dans $\mathcal{R}$, modulo$q$ et modulo p :
   - **Si** ce n'est pas le cas, revenir à l'étape 1.
   - **Sinon,** calculer les inverses correspondants.
3. Tirer uniformément $G(X) \in \mathcal{L}(d_g, d_g)$.
4. Calculer $H(X) = G(X)(F(X)^{-1} \bmod q) \mod q$.
5. La **clé publique** est $H(X)$, la **clé privée** est $F(X)$.

Remarques :

1. En pratique, pour la clé privée, on stocke aussi $F(X)^{-1} \bmod p$.
2. $F(X)$ et $G(X)$ sont des polynômes de "petite" norme euclidienne :

$$\|F(X)\| = \sqrt{2d_f - 1} < \sqrt{N} \quad \text{et} \quad \|G(X)\| = \sqrt{2d_g} < \sqrt{N}\,.$$

3. La clé publique $H(X)$ est constituée de N coefficients dans $\mathbb{Z}_q$, qui ne sont pas nécessairement petits. Sa norme sera donc proportionnelle à $\sqrt{qN}$.

Le système NTRU : chiffrement et déchiffrement

L'espace des **clairs** est $\mathcal{M} = \mathbb{Z}_p^N$ avec, souvent, en pratique $p = 3$.
L'espace des **chiffrés** est $\mathbb{Z}_q^N$, avec $q \gg p$.

NTRU : CHIFFREMENT

Pour chiffrer un message $\boldsymbol{m} \in \mathbb{Z}_p^N$:

1. Tirer uniformément $R(X) \in \mathcal{L}(d, d)$.
2. Associer à $\boldsymbol{m}$ le polynôme $M(X) \in \mathcal{R}$ correspondant.
3. Calculer et retourner $Y(X) = pR(X)H(X) + M(X) \mod q$ (ou le vecteur $\boldsymbol{y} \in \mathbb{Z}_q^N$ correspondant).

NTRU : DÉCHIFFREMENT

Pour déchiffrer un chiffré $\boldsymbol{y} \in \mathbb{Z}_q^N$:

1. Reconstruire le polynôme $Y(X) \in \mathcal{R}$ correspondant au vecteur $\boldsymbol{y}$.
2. Calculer $A(X) = Y(X)F(X) \mod q$.
3. Calculer et retourner $M'(X) = A(X)(F^{-1} \bmod p) \mod p$ (ou le vecteur $\boldsymbol{m}'$ correspondant).

Le système NTRU : exemple

Exemple pour $N = 11$, $p = 3$, $q = 101$, et les paramètres de degrés $d_f = 3$, $d_g = 2$ et $d = 2$.

Génération de clés.

- On engendre $F(X) \in \mathcal{L}(3,2)$ jusqu'à ce qu'il soit inversible modulo p et modulo q. Par exemple, on obtient les coefficients :

$$\boldsymbol{f} = (0,1,1,0,-1,1,0,0,0,-1,0)$$

- On engendre $G(X) \in \mathcal{L}(2,2)$ aléatoirement, puis on calcule $H(X) = G(X)(F(X)^{-1} \mod q) \in \mathcal{R}$. On obtient les coefficients :

$$\boldsymbol{g} = (0,0,1,-1,0,0,0,0,1,0,0)$$
$$\boldsymbol{h} = (24,16,-32,17,-50,-45,33,-25,-47,40,-29)$$

- Pour la clé privée, on peut également stocker en mémoire $F(X)^{-1} \mod p$, dont les coefficients sont :

$$(\boldsymbol{f}^{-1} \mod p) = (1,1,1,1,1,-1,-1,-1,0,1,1)$$

Chiffrement. Supposons que l'on chiffre $M(X)$ donné par les coefficients :

$$\boldsymbol{m} = (1,1,1,1,-1,0,-1,-1,-1,0,0)$$

- On engendre $R(X) \in \mathcal{L}(2,2)$ aléatoirement, puis on calcule $Y(X) = pR(X)H(X) + M(X)$. On obtient respectivement les coefficients :

$$\boldsymbol{r} = (0,0,1,0,0,0,-1,0,0,1,0)$$
$$\boldsymbol{y} = (-47,-44,0,19,-40,21,-21,-22,35,-18,19)$$

Déchiffrement.

- On calcule $A(X) = F(X)Y(X) \mod q$, puis $M'(X) = (F^{-1}(X) \mod p)A(X) \mod p$. On obtient respectivement les coefficients :

$$\boldsymbol{a} = (2,-3,2,-1,5,-2,3,-1,-3,-1,2)$$
$$\boldsymbol{m}' = (1,1,1,1,-1,0,-1,-1,-1,0,0)$$

Le système NTRU : validité

Résumé.

1. $\mathtt{sk} = f$ inversible modulo p et q,
   $\mathtt{pk} = g \star (f^{-1} \mod q)$.
2. Chiffrement de $m \in \mathbb{Z}_p^N$: $y = p\,r \star h + m \mod q$ avec $r \in \mathcal{L}(d,d)$.
3. Déchiffrement : $m' = ((f^{-1} \mod p) \star (f \star y \mod q)) \mod p$.

Validité. On a

$$f \star y \equiv f \star (p\,r \star h + m) \equiv p\,r \star (f \star h) + f \star m \equiv p\,r \star g + f \star m \mod q$$

Si tous les coefficients (calculés dans $\mathbb{Z}$) de $p\,r \star g + f \star m$ sont dans $\{-\frac{q-1}{2}, \ldots, \frac{q-1}{2}\}$, alors on a

$$f \star y = p\,r \star g + f \star m \quad (\text{dans } \mathbb{Z})$$

donc

$$f \star y \equiv f \star m \mod p\,.$$

En mutlipliant par l'inverse de f modulo p, on obtient bien le message m.

Conséquence. Afin que le déchiffement fonctionne, on choisit N, p et q de sorte que les coefficients a_i de $r \star g + f \star m$ soient tels que $|a_i| \leq \frac{q-1}{2}$ avec très bonne probabilité.

Le système NTRU : sécurité (élémentaire)

Il faut choisir l'ensemble des paramètres pour **éviter deux attaques élémentaires**.

1. **Attaque exhaustive sur la clé privée :** à partir de $H(X)$, tester toutes les valeurs de $F(X)$ jusqu'à ce que $F(X)H(X) = G(X) \mod q$.

   Combien de tests ? C'est-à-dire, combien d'éléments $F(X) \in \mathcal{L}(d_f, d_f - 1)$?

   $$|\mathcal{L}(d_f, d_f - 1)| = \binom{N}{d_f} \times \binom{N - d_f}{d_f - 1} \simeq \frac{N!}{(d_f!)^2 (N - 2d_f)!}$$

   Une amélioration (compromis temps-mémoire de type *meet-in-the-middle*) permet d'améliorer l'attaque en temps

   $$\sqrt{|\mathcal{L}(d_f, d_f - 1)|} \simeq \frac{1}{d_f!} \sqrt{\frac{N!}{(N - 2d_f)!}}$$

2. **Attaque exhaustive sur le message :** on teste tous les $R(X)$ jusqu'à ce que $Y(X) - R(X)H(X) \mod q$ ait de petits coefficients (donc, soit probablement égal à $M(X)$).

   Un calcul similaire donne une attaque en temps :

   $$\sqrt{|\mathcal{L}(d, d)|} \simeq \frac{1}{d!} \sqrt{\frac{N!}{(N - 2d)!}}$$

Le système NTRU : sécurité (élémentaire)

Résumé.

1. $\mathtt{sk} = \boldsymbol{f}$ inversible modulo p et q,
   $\mathtt{pk} = \boldsymbol{g} \star (\boldsymbol{f}^{-1} \mod q)$.
2. Chiffrement de $\boldsymbol{m} \in \mathbb{Z}_p^N$: $\boldsymbol{y} = p\,\boldsymbol{r} \star \boldsymbol{h} + \boldsymbol{m} \mod q$ avec $\boldsymbol{r} \in \mathcal{L}(d, d)$.
3. Déchiffrement : $\boldsymbol{m}' = ((\boldsymbol{f}^{-1} \mod p) \star (\boldsymbol{f} \star \boldsymbol{y} \mod q)) \mod p$.

Rappel. On a demandé que p et q soient premiers entre eux, pour une raison de sécurité.

Exercice. Supposons (par exemple) que p divise q. Quel est le **souci de sécurité** ?

Réponse. Simplement, l'attaquant calcule $\boldsymbol{y} \mod p$ et retrouve le message $\boldsymbol{m}$.

Le système NTRU : sécurité (approfondie)

Observation. Attaquer la clé de NTRU s'apparente à résoudre un problème CVP.

Soit $\mathcal{A}$ le réseau (dans $\mathbb{Z}^{2N}$) engendré par la matrice

$$\boldsymbol{A} = \begin{pmatrix} \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{M}(\boldsymbol{h})^\top & q\boldsymbol{I} \end{pmatrix} \in \mathbb{Z}^{2N\times 2N}$$

Comme $\boldsymbol{A}\cdot(\boldsymbol{u},\boldsymbol{v})^\top = (\boldsymbol{u},\boldsymbol{h}\star\boldsymbol{u}+q\boldsymbol{v})^\top$, on peut écrire que

$$\mathcal{A} = \{(\boldsymbol{a},\boldsymbol{b})^\top \in (\mathbb{Z}^N)^2, \boldsymbol{a}\star\boldsymbol{h} \equiv \boldsymbol{b} \mod q\}$$

Comme on sait que, par définition, $\boldsymbol{f}\star\boldsymbol{h} \equiv \boldsymbol{g} \mod q$, on obtient $(\boldsymbol{f},\boldsymbol{g}) \in \mathcal{A}$.

Par ailleurs, $\|(\boldsymbol{f},\boldsymbol{g})\|_2 = \sqrt{2d_g + 2d_f - 1} < \sqrt{2N}$, tandis que dans un réseau aléatoire de dimension $2N$ sur $\mathbb{Z}_q$, un vecteur aléatoire aurait une norme proportionnelle à $\sqrt{qN}$.

Si $q \gg 1$, le vecteur $(\boldsymbol{f},\boldsymbol{g})$ est donc **particulièrement court** dans le réseau $\mathcal{A}$. On peut donc le chercher avec un algorithme qui résout le problème CVP.

Conséquence. Si CVP est facile en dimension $2N$, alors on peut attaquer le chiffrement NTRU. Pas de preuve pour la réciproque...

En pratique

Il existe beaucoup d'autres attaques à éviter.

Pour obtenir une sécurité IND-CPA, voire IND-CCA2, il faut également incorporer des **fonctions de hachage**.

Dans la **proposition NTRU** sélectionnée pour standardisation au NIST, et pour une sécurité « quantique » de 128 bits :

- $N = 509$, $q = 2048$ et $p = 3$,
- clés publique/privée de taille 699/935 octets (compressées),
- chiffrés de taille 699 octets,
- chiffrement et déchiffrement rapides ($< 2\,000\,000$ cycles).

Remarque. Il existe une signature fondée sur NTRU, intitulée **Falcon** et sélectionnée pour standardisation au NIST.

Plan

Le problème LWE

Problème LWE (*learning with errors*). Soient q un nombre premier, n un entier, et $\mathcal{E}$ une variable aléatoire de distribution $\pi_{\mathcal{E}}$, à valeurs dans $\mathbb{Z}_q$.

Instance. Une séquence de m échantillons $(\boldsymbol{a}_i, b_i) \in \mathbb{Z}_q^{n+1}$, où

- chaque $\boldsymbol{a}_i$ est choisi uniformément dans $\mathbb{Z}_q^n$,
- $b_i = e_i + \sum_{j=1}^{n} a_{i,j} s_j \mod q$, avec e_i tiré selon $\mathcal{E}$
- les $s_j \in \mathbb{Z}_q$ sont quelconques.

But. Trouver $\boldsymbol{s}$.

Remarque. Si l'on note $\boldsymbol{A} \in \mathbb{Z}_q^{m \times n}$ la matrice des $(a_{i,j})$, alors le problème LWE s'apparente à

$$\text{trouver } \boldsymbol{s} \text{ tel que } \boldsymbol{A}\boldsymbol{s} + \boldsymbol{e} \equiv \boldsymbol{b} \mod q, \quad \text{avec } \boldsymbol{e} \sim \mathcal{E}^m.$$

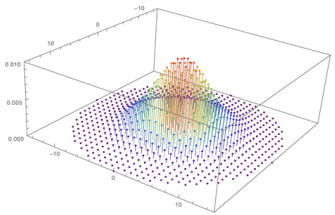
Difficulté. Il existe des paramètres pour lesquels le problème LWE est difficile :

- m « petit » devant n,
- distribution $\mathcal{E}$ centrée en 0

Paramètres publics.

- Des entiers n et m, un nombre premier q,
- Une variable aléatoire $\mathcal{E}$ sur $\mathbb{Z}_q$, telle que de petites valeurs sont très probables.

En pratique, on utilise une Gaussienne discrète centrée d'écart type σ petit comparé à q.



REGEV : GÉNÉRATION DE CLEFS

1. Tirer $\boldsymbol{s}$ uniformément dans $\mathbb{Z}_q^n$.
2. Tirer uniformément une matrice $\boldsymbol{A} \in \mathbb{Z}_q^{m \times n}$.
3. Tirer $\boldsymbol{e} = (e_1, \ldots, e_m)^\top$, où les e_i sont tirés selon $\mathcal{E}$.
4. Calculer $\boldsymbol{b} = \boldsymbol{A}\boldsymbol{s} + \boldsymbol{e} \mod q$.
5. La clé publique est $(\boldsymbol{A}, \boldsymbol{b})$, la clé privée est $\boldsymbol{s}$.

Chiffrement de Regev

L'espace des **clairs** est $\{0,1\}$. L'espace des **chiffrés** est $\mathbb{Z}_q^{n+1}$. On note $\langle \cdot, \cdot \rangle$ le produit scalaire.

REGEV : CHIFFREMENT

Pour chiffrer **un bit** $x \in \{0,1\}$.

1. Définir $\boldsymbol{r} = (r_1, \ldots, r_m)$, où les r_i sont tirés uniformément dans $\{0,1\}$
2. Si $x = 0$, alors définir $y = (\boldsymbol{r}\boldsymbol{A}, \langle \boldsymbol{r}, \boldsymbol{b} \rangle)$
3. Si $x = 1$, alors définir $y = (\boldsymbol{r}\boldsymbol{A}, \lfloor \frac{q}{2} \rfloor + \langle \boldsymbol{r}, \boldsymbol{b} \rangle)$
4. Retourner y.

REGEV : DÉCHIFFREMENT

Pour déchiffrer $(\boldsymbol{u}, v)$.

1. Calculer $z = v - \sum_{j=1}^{n} u_j s_j$.
2. Si $|z| < \left| \lfloor \frac{q}{2} \rfloor - z \right|$, retourner $x' = 0$.
3. Sinon, retourner $x' = 1$.

Validité. Que vaut $z = v - \sum_{j=1}^{n} u_j s_j$?
En notation matricielle, si les vecteurs sont notés comme des vecteurs colonnes :

$$\begin{aligned} z = v - \boldsymbol{u}^\top \cdot \boldsymbol{s} &= \left(\boldsymbol{r}^\top \cdot \boldsymbol{b} + x \lfloor \tfrac{q}{2} \rfloor\right) - \left(\boldsymbol{r}^\top \cdot \boldsymbol{A}\right) \cdot \boldsymbol{s} \\ &= \boldsymbol{r}^\top \cdot (\boldsymbol{b} - \boldsymbol{A}\boldsymbol{s}) + x \lfloor \tfrac{q}{2} \rfloor = \boldsymbol{r}^\top \cdot \boldsymbol{e} + x \lfloor \tfrac{q}{2} \rfloor \end{aligned}$$

Comme $\mathcal{E}$ suit une loi gaussienne discrète centrée en 0 et de petite variance, on a

$$\boldsymbol{r}^\top \cdot \boldsymbol{e} \ll q\,.$$

Ainsi, on retrouve x suivant si z est proche de 0 ou de $\lfloor \frac{q}{2} \rfloor$.

Remarque. C'est donc un déchiffrement probabiliste.

Sécurité. La sécurité du chiffrement de Regev repose sur deux points :

1. impossibilité de distinguer $\boldsymbol{A}\boldsymbol{s} + \boldsymbol{e}$ d'un élément uniforme de $\mathbb{Z}_q^m$ (variante décisionnelle du problème LWE)
2. impossibilité de distinguer $(\boldsymbol{r}^\top \boldsymbol{A}, \boldsymbol{r}^\top \boldsymbol{b})$ d'un élément uniforme de $\mathbb{Z}_q^{n+1}$. Un théorème (*leftover hash lemma*) l'assure dès lors que m n'est pas trop grand, typiquement $m \ll n \log_2 q$.

Chiffrement de Regev : efficacité et variantes

Constat. Le chiffrement de Regev est peu pratique : un seul bit $x \in \{0,1\}$ est chiffré en $(n+1)\log q$ bits : $(\boldsymbol{u}, v) \in \mathbb{F}_q^{n+1}$.

On peut gagner en efficacité en

- remplaçant les vecteurs $\boldsymbol{s}$, $\boldsymbol{e}$ et $\boldsymbol{b}$ par des matrices à $k \simeq n$ colonnes ;
- parallélisant le chiffrement (même $\boldsymbol{r}$ pour plusieurs bits x).

Néanmoins, cela augmente la taille des clés.

Variantes. Il existe des variantes avec des réseaux **structurés**. L'idée est de remplacer la matrice aléatoire uniforme $\boldsymbol{A}$ par une matrice toujours aléatoire mais plus structurée, afin d'obtenir des **clés plus courtes** et des **calculs plus efficaces**.

- On peut prendre $\boldsymbol{A}$ circulante, comme $\boldsymbol{M}(\boldsymbol{a})$ vue précédemment : il suffit alors de stocker $\boldsymbol{a}$ (clé plus courte). Cela correspond à considérer l'anneau de polynômes $\mathbb{F}_q[x]/(x^n - 1)$.
- On peut choisir d'autres anneaux de polynômes $\mathbb{F}_q[x]/(\Phi(x))$: ce sont les familles de chiffrements « Ring-LWE » et « Module-LWE ».

Par exemple, le cryptosystème **CRYSTALS-Kyber** a été retenu par le NIST, et propose des tailles de clé ≤ 2 ko pour une sécurité de 128 bits.

`https://pq-crystals.org/kyber/index.shtml`

Fin de la partie cours

Questions ?