Université Paris 8
A.A. 2018–2019

Interactions codes/cryptographie

Martino Borello

4 décembre 2018

Table des matières

Introduction

Les interactions entre la théorie des codes et la cryptographie sont nombreuses et on peut dire que les deux disciplines se sont enrichies de ces relations.

La théorie des codes, qui a débuté à la fin des années '40, est généralement considérée comme une science mature. La cryptographie d'autre part, au moins dans le secteur public, est une science en phase de développement rapide. Il existe de nombreux problèmes de cryptographie auxquels les techniques bien développées et les résultats de la théorie des codes peuvent être appliqués avec succès.

Dans les schémas de partage de secret, les codes se révèlent utiles pour généraliser les structures d'accès au secret. En outre, la théorie des codes peut jouer un rôle fondamental dans les systèmes d'authentification. Mais c'est sûrement dans la cryptographie basée sur les codes que l'interaction entre les deux disciplines montre particulièrement son importance : au cours des dernières années, il y a eu une quantité importante de recherches sur les ordinateurs quantiques. Si des ordinateurs quantiques à grande échelle sont construits, ils pourront rompre la plupart des cryptosystèmes à clé publique actuellement utilisés. Cela compromettrait sérieusement la confidentialité et l'intégrité des communications numériques. L'objectif de la cryptographie post-quantique est de développer des systèmes cryptographiques sécurisés contre les ordinateurs quantiques et classiques et inter-opérer avec les protocoles et les réseaux de communication existants. La cryptographie basée sur les codes est l'une des possibles solutions pour concevoir des cryptosystèmes post-quantiques.

Chapitre 1

Partage de secret

Le **partage de secret** consiste à répartir un secret – par exemple une clé ou un mot de passe – entre plusieurs dépositaires. Dans le schéma le plus simple de partage de secret on a que le secret ne peut être découvert que si un nombre suffisant de dépositaires mettent en commun les informations qu'ils ont reçues et, en revanche, un nombre inférieur de dépositaires n'apporte aucune information sur le secret.

Exemple : l'accès à un local renfermant des matières très sensibles ne peut être autorisé que si au moins deux personnes se présentent avec leurs clés, afin d'éviter qu'une personne isolée ne puisse les manipuler ou les dérober sans contrôle.

Un tel schéma de partage de secret a été inventé par Adi Shamir en 1979 [21].

1.1 Le schéma de Shamir

On l'a déjà vu dans le cours de Mathématiques et théorie de l'information. C'est une méthode simple et élégante, basée sur le fait qu'il existe un et un seul polynôme $q(x) \in K[x]$ (où K est un corps) de degré $t-1$ au plus défini par un ensemble de t points $P_1 \equiv (x_1, y_1), \ldots, P_t \equiv (x_t, y_t)$ (c'est-à-dire tel que $q(x_i) = y_i$ pour tout i).

Le secret dans ce schéma est un élément $a_0 \in K$. La personne qui détient le secret engendre un polynôme

$$q(x) := a_0 + a_1 x + \ldots a_{t-1} x^{t-1}$$

de degré $t-1$ et avec les coefficients $a_1, \ldots, a_{t-1}$ choisis au hasard dans K. On a $n \geq t$ dépositaires, représentés par des éléments $x_1, \ldots, x_n$ dans $K - \{0\}$. La personne qui détient le secret donne à chaque dépositaire la valeur $y_i := q(x_i)$. On a que t dépositaires ensembles peuvent reconstituer $q(x)$ (interpolation de

Lagrange) et calculer le secret $a_0 = q(0)$, tandis que $t' < t$ dépositaires n'ont aucune information sur le secret (le montrer comme exercice).

Le nombre t s'appelle le **seuil de reconstitution**, i.e. le nombre minimal de participants requis pour reconstituer le secret.

Exercice 1.1. *Soient $x_1, \ldots, x_t$ des dépositaires. Montrer que*

$$a_0 = \sum_{i=1}^{t} y_i \cdot \beta_i, \text{ avec } \beta_i := \prod_{j=0, j\neq i}^{t} \frac{x_j}{x_j - x_i}$$

1.2 Lien avec les codes MDS

Comme observé par Robert J. McEliece et Dilip V. Sarwate dans [18], le schéma de Shamir est strictement lié aux codes MDS et il y a plusieurs avantages à considérer le partage de secret de Shamir dans le contexte de la théorie des codes.

Le schéma de partage de secret est le suivant : soit C un $[n+1, t, n-t+2]_q$ code MDS (la borne de Singleton est vraie avec égalité : $t = n+1-(n+2-t)+1$). Le secret est un élément $a_0 \in \mathbb{F}_q$. La personne qui détient le secret choisit au hasard $t-1$ éléments $a_1, \ldots, a_{t-1}$ dans $\mathbb{F}_q$.
Rappel : pour les codes MDS on a le résultat suivant.

Lemme 1.1. *Soit C un $[n+1, t, n-t+2]_q$ code. Pour tout $(a_0, \ldots, a_{t-1}) \in \mathbb{F}_q^t$ et $\{i_0, \ldots, i_{t-1}\} \subseteq \{0, \ldots, n\}$, il existe un et un seul $c \in C$ tel que $c_{i_j} = a_j$ pour tout $j \in \{0, \ldots, t-1\}$.*

Démonstration. Soit $\pi_{i_0,\ldots,i_{t-1}} : C \to \mathbb{F}_q^t$ l'application $c \mapsto (c_{i_0}, \ldots, c_{i_{t-1}})$. Elle est injective : en effet, si $\pi_{i_0,\ldots,i_{t-1}}(c) = \pi_{i_0,\ldots,i_{t-1}}(c')$ pour $c, c' \in C$, alors $\mathrm{d}(c, c') \leq n+1-t$, ce qui implique $c = c'$. Puisque $\#C = q^t$, elle est surjective aussi, de manière que le lemme est montré. □

La personne qui détient le secret peut donc déterminer le seul mot $c \in C$ tel que $\pi_{0,\ldots,t-1}(c) = (a_0, \ldots, a_{t-1})$. Après, elle donne à chaque dépositaire l'un des n chiffres $c_1, \ldots, c_n$. Grâce au Lemme 1.1, on a que t dépositaire ensembles peuvent reconstituer c et connaître donc c_0.

Ce schéma est une généralisation du schéma de Shamir :

Exercice 1.2. *Soient $x_1, \ldots, x_n \in \mathbb{F}_q - \{0\}$, avec $n \geq t$. Montrer que le code*

$$C := \{(q(0), q(x_1), \ldots, q(x_n)) \mid q(x) \in \mathbb{F}_q[x] \text{ de degré au plus } t-1\}$$

est un $[n+1, t, n-t+2]_q$ code (Suggestion : pour ce qui concerne la dimension, montrer l'injectivité d'une application d'évaluation).

1.3 Le schéma de Massey

Dans les deux schémas précédents, ce qui caractérise l'accès au secret est le nombre de dépositaires qui doivent être d'accord. Dans certaines applications, on pourrait vouloir que même des coalitions avec un nombre différent de dépositaires peuvent accéder au secret.

Par exemple, on voudrait un schéma pour partager un secret entre Alice, Bob, Carol et David, tel que Alice et Bob peuvent reconstituer le secret, comme aussi Bob, Carol et David, mais aucune coalition de dépositaires ne contenant pas l'une de ces deux coalitions autorisées peut obtenir des informations sur le secret. Une solution possible à ce problème est

- choisir le secret comme le premier chiffre c_1 d'un mot du $[5,3]_q$ code qui a matrice de parité
$$H := \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 \end{bmatrix};$$
- choisir deux éléments c_2 et c_4 au hasard dans $\mathbb{F}_q$;
- reconstituer le mot $c = (c_1, c_2, \mathbf{c}_3, c_4, \mathbf{c}_5)$ (Exercice : montrer que c'est possible) ;
- donner c_2, c_3, c_4, c_5 à Alice, Bob, Carol et David respectivement.

Puisque H est la matrice de parité de C, on a que $c_1 = -c_2 - c_3$, de manière que Alice et Bob peuvent reconstituer le secret, et $c_1 = -c_3 + c_4 + c_5$, de manière que Bob, Carol et David peuvent reconstituer le secret.

Exercice 1.3. *Montrer qu'aucune coalition de dépositaires ne contenant pas l'une de ces deux coalitions autorisées peut obtenir des informations sur le secret.*

On a donc traité notre cas particulier. On va montrer que cela peut être généralisé.

1.3.1 Mots minimaux et leurs propriétés

On commence par deux définitions.

Définition 1.1. *Soient $v := (v_1, \ldots, v_n), v' := (v'_1, \ldots, v'_n) \in \mathbb{F}_q^n$. On dit que v **couvre** v' si pour tout i on a $v'_i \neq 0 \Rightarrow v_i \neq 0$.*

Par exemple $(1,1,0,1,0)$ couvre $(0,2,0,1,0)$ (vecteurs dans $\mathbb{F}_3^5$).

Définition 1.2. *Un mot c d'un code C est dit **minimal** si*

1. *c est un mot non nul dont la composante non nulle la plus à gauche est égale à 1,*
2. *c ne couvre aucun autre mot dont la composante non nulle la plus à gauche est égale à 1.*

Il découle immédiatement de la définition que tous les mots de poids minimal dont la composante non nulle la plus à gauche est égale à 1 sont mots minimaux.

Exercice 1.4. *Soit C un $[n, k, n-k+1]_q$ code (MDS). Montrer qu'il y a $\binom{n}{n-k+1}$ mots de poids minimal dont la composante non nulle la plus à gauche est égale à 1 et ces sont tous les mots minimaux du code.*

Exercice 1.5. *Soient $c, c' \in C$ deux mots minimaux. Montrer que*

$$\{i \mid c_i = 0\} \neq \{j \mid c'_j = 0\}.$$

Lemme 1.2. *Tout mot non nul c qui n'est pas minimal couvre un mot minimal c' tel que*

$$\min\{i \mid c_i \neq 0\} = \min\{j \mid c'_j \neq 0\}.$$

Démonstration. c n'est pas minimal, donc il couvre un mot minimal c' et il existe un scalaire λ' tel que $c - \lambda' c'$ est un mot de poids plus petit du poids de c. On répète l'argument... (conclure par exercice, voir [15]). □

1.3.2 Description du schéma

Soit C un $[n+1, t]_q$ code avec matrice génératrice $[I_t|A]$ (où I_t est la matrice identité). On suppose que

1. le secret est un élément $c_0 \in \mathbb{F}_q$;
2. la personne qui détient le secret choisi au hasard $c_1, \ldots, c_{t-1} \in \mathbb{F}_q$ et calcule $c = (c_0, \ldots, c_{t-1})[I_t|A]$.
3. la personne qui détient le secret donne aux n dépositaires l'un des chiffres $c_1, \ldots, c_n$ de c.

Définition 1.3. *Une **structure d'accès** d'un schéma de partage de secret est l'ensemble des sous-ensembles $\{i_1, \ldots, i_m\}$ de $\{1, \ldots, n\}$ tels que, pour tout $c \in C$, $c_{i_1}, \ldots, c_{i_m}$ détermine c_0 de manière unique (par combinaison linéaire), mais, si on enlève l'un des indices, on a aucune information sur le secret.*

Clairement une coalition de dépositaires peut déterminer le secret si et seulement si elle "contient" une structure d'accès.

Théorème 1.1. *Soit $M := \{h \in C^\perp \mid h_0 = 1$ et h est minimal$\}$. La structure d'accès d'un schéma de partage de secret donnée par C est*

$$\mathcal{S} := \{\{i > 0 \mid h_i \neq 0\} \mid h \in M\}.$$

Démonstration. Soit $\{i_1, \ldots, i_m\}$ un élément de la structure d'accès. Par définition, on a que

$$c_0 = \lambda_1 c_{i_1} + \ldots + \lambda_m c_{i_m}$$

pour certains $\lambda_1, \ldots, \lambda_m \in \mathbb{F}_q$ et pour tout $c \in C$, ce qui équivaut à

$$h := (1, 0, \ldots, 0, \underbrace{-\lambda_1}_{i_1-\text{ème}}, 0, \ldots, 0, \underbrace{-\lambda_m}_{i_m-\text{ème}}, 0, \ldots, 0) \in C^\perp.$$

De plus, puisque $\{i_1, \ldots, i_m\}$ est un élément de la structure d'accès, il n'existe pas $h' \in C^\perp$ tel que $h'_0 = 1$ et h couvre h', i.e., par le Lemme 1.2, $h \in M$. Donc $\{i_1, \ldots, i_m\} \in \mathcal{S}$.

Vice versa, si $\{i_1, \ldots, i_m\} \in \mathcal{S}$, alors il existe $h \in C^\perp$ tel que $h_0 = 1$ et $h_j \neq 0$ si et seulement si $j \in \{i_1, \ldots, i_m\}$, de manière que

$$c_0 = -h_{i_1} c_{i_1} + \ldots - h_{i_m} c_{i_m}$$

pour tout $c \in C$. Ainsi, $\{i_1, \ldots, i_m\}$ appartient à la structure d'accès. □

Exercice 1.6. *Revoir l'exemple au début à la lumière de ce théorème.*

Remarque 1.1. *On a réduit donc le problème de la réalisation d'une structure d'accès au problème de construire un code linéaire dont les mots minimaux avec premier chiffre égal à* 1 *correspondent aux sous-ensembles de la structure. Cela suggère que le problème de partage de secret peut être donc affronté à travers les outils bien développés de la théorie des codes.*

Exercice 1.7. *Montrer que la structure d'accès donnée par le* $[6, 3, 3]_3$ *code avec matrice de parité*

$$H := \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 2 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 2 & 2 \end{bmatrix}$$

est $\{\{1, 2, 4\}, \{1, 3, 4\}, \{2, 3, 4\}, \{5\}\}$ (*ce code admet-il une matrice génératrice en forme systématique ?*).
Déterminer le secret s *sachant que* $(c_0, 0, 1, 1, 2, 0)$ *est un mot de code.*

Une version plus faible, mais utilisée fréquemment en littérature, du Théorème 1.1, est la suivante.

Exercice 1.8. *Soit* C *un* $[n + 1, t]_q$ *code. Alors dans le schéma de Massey, les dépositaires* $\{i_1, \ldots, i_m\} \subseteq \{1, \ldots, n\}$ *peuvent avoir accès au secret si et seulement s'il existe un mot*

$$h = (1, 0, \ldots, 0, h_{i_1}, 0, \ldots, 0, h_{i_m}, 0, \ldots, 0) \in C^\perp$$

avec au moins l'un des h_{i_j} *différent de* 0.

1.4 Structure d'accès éparse

Dans [1], les auteurs montrent une faiblesse du système de Shamir dans le contexte réel : supposons que dans une banque avec 6 caissiers l'accès au coffre de banque soit permis seulement à 3 caissiers ensemble. Un jour le coffre de banque est retrouvé vide. La police trouve que l'un des caissiers a un alibi très forte. Cela n'apporte pas trop d'information, parce que il y a encore $\binom{5}{3} = 10$ coalitions minimales de possible coupables.

Supposons maintenant que la structure d'accès au coffre soit

$$\{1,2,4\},\{3,4,5\},\{2,5,6\},\{1,3,6\}.$$

Alors si 1 a un alibi, forcement 5 est coupable et la police peut le persuader de nommer ses complices. On peut voir facilement que cela est vrai même si n'importe quel caissier (à la place de 1) a un alibi.
En outre, même si cette structure est plus éparse que celle de Shamir, elle garantit une certaine résilience : si par exemple un caissier est malade, l'accès au coffre de banque est assuré. Si deux caissier sont malades, l'accès est assuré dans le 80% des cas (exercice).

Comment généraliser cet exemple ?

Pour généraliser l'exemple ci-dessus on a besoin d'introduire un objet classique de la combinatoire : soit X un ensemble à v éléments, qu'on appelle **points**.

Définition 1.4. *Un t-(v,k,λ) **système de blocs** (ou t-design) est une collection de sous-ensembles à k points, dits **blocs**, tels que tout sous-ensemble à t points de X est contenu exactement en λ blocs.*

La structure d'accès

$$\{1,2,4\},\{3,4,5\},\{2,5,6\},\{1,3,6\}$$

est un 1-$(6,3,2)$ *design*. Ce type de structure s'appelle **démocratique** (de degré 1).

On peut noter (exercice) que cette structure est donnée (schéma de Massey) par le $[7,4,3]$ code de Hamming $\mathcal{H}_3$ avec matrice génératrice

$$G := \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}.$$

Ce type de lien entre code et *designs* est assez étudié. Pour en parler, on a besoin de définir trois quantités, associées à un code C :

— $A_i(C) := \#\{c \in C \mid \mathrm{wt}(c) = i\}$;
— $\omega(C) := \{i \mid A_i > 0\}$;
— $\delta_i(C) := \{\{j \mid c_j \neq 0\} \mid c \in C \text{ de poids } i\}$.

On peut donc énoncer le résultat fondamental.

Théorème 1.2. *Soit $C^\perp$ un $[n+1, n-k+1, d]_q$ code tel que $\omega(C^\perp) = \{0, d, n+1\}$. Supposons qu'il existe un entier t, $1 < t < d$, tel qu'il y a au plus $d-t$ poids non nuls de C dans l'intervalle de 1 à $n+1-t$. Soit en particulier t le plus grand entier avec cette propriété. Alors le schéma de partage de secret basé sur le code C (schéma de Massey) a les propriétés suivantes :*

1. *il y a $\frac{d \cdot A_d(C^\perp)}{(n+1)\cdot(q-1)}$ éléments dans la structure d'accès, composés par $d-1$ dépositaires chacun ;*

2. *tout sous-ensemble de* $t-1$ *dépositaires fait partie exactement de*

$$\frac{\binom{d}{t}\cdot A_d(C^\perp)}{\binom{n+1}{t}\cdot(q-1)}$$

éléments de la structure d'accès.

Démonstration. Le résultat est une conséquence du théorème de Assmus-Mattson, qui implique que $\delta_d(C^\perp)$ est un t-$(n+1,d,\lambda)$ *design*, et d'autres résultats classiques de la théorie des *designs* (Voir [1] et ses références). □

Les conditions du Théorème 1.2 sont suffisants pour avoir un t-*design*, mais elles ne sont pas faciles à contrôler. Clairement avoir t grand implique une structure plus démocratique, mais cela a une implication sur la taille de la structure.

Exercice 1.9. *Soit* C *le* $[8,4,4]$ *code de Hamming étendu* $\hat{\mathcal{H}}_3$. *On a que* $C = C^\perp$, $\omega(C) = \{0,4,8\}$, $A_0 = A_8 = 1$ *et* $A_4 = 14$. *Pour* $t = 3$, *on a au plus* $1 = 4-3$ *poids non nuls dans l'intervalle de* 1 *à* 5. *Alors le théorème nous donne qu'il y a*

$$7 = \frac{4\cdot 14}{8}$$

éléments dans la structure d'accès, composés par 3 *dépositaires chacun, et tous sous-ensemble de* 2 *dépositaires fait partie exactement de*

$$1 = \frac{\binom{4}{3}\cdot 14}{\binom{8}{3}}$$

élément de la structure d'accès.

Puisque dans les cas concrètes on a souvent besoin de 2-*designs*, on montre le résultat suivant.

Lemme 1.3. *Soit* C *un* $[n,k,d]_q$ *code avec* $\omega(C) \subseteq \{0,d,n\}$, $k \geq 2$ *et une matrice génératrice* G *dont aucune colonne est le vecteur nul. Alors* $\delta_d(C)$ *est un* 2-*design si et seulement si la distance minimale* $d^\perp$ *de* $C^\perp$ *est au moins* 3.

Démonstration. Pour $\{i,j\} \subseteq \{1,\dots,n\}$, $i \neq j$, soit

$$N_{i,j} := \#\{b \in \delta_d(C) \mid \{i,j\} \subseteq b\}.$$

On doit montrer que $N_{i,j}$ est constant si et seulement si $d^\perp > 2$, i.e. si et seulement si les colonnes de G sont deux à deux linéairement indépendantes.

Soient $\underline{g}_i$ et $\underline{g}_j$ les colonnes i-ième et j-ième de G respectivement et soit $G_{i,j}$ la matrice $\left[\underline{g}_i, \underline{g}_j\right]$. On indique avec $(G_{i,j})_r$, pour $r \in \{1,\dots,k\}$, les lignes de $G_{i,j}$ et on appelle $\varphi_{i,j} : \mathbb{F}_q^k \to \mathbb{F}_q^2$, l'application linéaire $(\lambda_1,\dots,\lambda_k) \mapsto \sum_{r=1}^k \lambda_r (G_{i,j})_r$.

Si $\underline{g}_i, \underline{g}_j$ sont linéairement indépendantes alors il existe deux lignes de $G_{i,j}$ linéairement indépendantes, ce qui implique que l'image de $\varphi_{i,j}$ a dimension

2, i.e. le noyau de $\varphi_{i,j}$ a dimension $k-2$ et cardinalité q^{k-2}, de manière que $(q-1)^2 \cdot q^{k-2}$ (# images·# pré-images par image) mots de C n'ont aucun zéro dans les coordonnées i et j. Ainsi

$$N_{i,j} = (q-1)^2 \cdot q^{k-2} - A_n(C).$$

Si $\underline{g}_i, \underline{g}_j$ ne sont pas linéairement indépendantes alors on obtient de la même manière (exercice) que

$$N_{i,j} = (q-1) \cdot q^{k-1} - A_n(C).$$

Cela implique que $N_{i,j}$ est constant si et seulement si et seulement si les colonnes de G sont deux à deux linéairement indépendantes. □

Exercice 1.10. *Montrer que dans un* 2-(v,k,λ) *design il y a*

$$\frac{\lambda \cdot \binom{v}{2}}{\binom{k}{2}}$$

blocs en total, et

$$\frac{\lambda \cdot (v-1)}{k-1}$$

blocs qui contient un point donné. Peut-on généraliser ces résultats ?

Exercice 1.11. *Soit C un $[n,k,d]_q$ code avec $\omega(C) = \{0,d,n\}$, $k \geq 2$ et une matrice génératrice G dont aucune colonne est le vecteur nul et supposons que la distance minimale $d^\perp$ de $C^\perp$ soit au moins 3. Montrer que le schéma de partage de secret basé sur le code $C^\perp$ (schéma de Massey) a les propriétés suivantes :*

1. *tout dépositaire fait partie exactement de*

$$\frac{A_d(C) \cdot d \cdot (d-1)}{(q-1) \cdot n \cdot (n-1)}$$

   éléments de la structure d'accès ;

2. *il y a $\frac{A_d(C) \cdot d}{(q-1) \cdot n}$ éléments dans la structure d'accès, composés par $d-1$ dépositaires chacun.*

Donc pour avoir une bonne structure d'accès on cherche des codes de distance minimal au moins 3 dont le dual a au plus trois poids non nuls.

On peut avoir deux poids non nuls (codes **équidistants**) ou trois poids non nuls (codes à **deux-poids**, parce qu'on ne compte pas le poids 0).

Une classification de ces codes est donnée dans [1]. Pour les codes équidistants, les plus significatifs du point de vue du partage du secret sont les duaux des codes de Hamming, appelés *simplex-codes*.

Un exemple de codes à deux-poids est la famille des codes de Reed-Muller de premier ordre $\mathcal{R}_2(1,m)$, qui sont des codes de longueur 2^m et avec poids non nuls $0, 2^{m/2}, 2^m$.

1.5 Développements

Problèmes [8] :

1. Étant donné un code linéaire, comment déterminer la structure d'accès du schéma de partage de secret basé sur ce code ?
2. Étant donnée une structure d'accès, comment construire un code linéaire de sorte que le schéma de partage de secret correspondant à cette structure d'accès, tout en minimisant le taux de transmission ?

Attaquer le premier problème est plus ou moins équivalent à déterminer l'ensemble de tous les mots minimaux du code (**problème de couverture** du code linéaire). C'est un problème très difficile pour les codes linéaires généraux, et n'a été résolu que pour quelques classes de codes linéaires spéciaux. Le deuxième problème dépend des solutions au premier, et il est également très difficile en général. Jusqu'à présent, aucune solution générale n'est connue. Intuitivement, seulement des codes linéaires bien structurés peuvent donner des schémas de partage de secret avec des bonnes structures d'accès. Ainsi, la construction de codes linéaires avec certaines propriétés est une direction intéressante dans l'étude des schémas de partage de secret.

Un résultat très utilisé récemment est le Lemme de Ashikhmin–Barg [2, Lemme 2.1 point 3)], qui avec nos notations a la forme suivante.

Lemme 1.4 (Ashikhmin–Barg)**.** *Soit C un $[n, k, d]_q$ code et c un mot dont la composante non nulle la plus à gauche est égale à 1. Si*

$$\mathrm{wt}(c) < d \cdot \frac{q}{q-1}$$

alors c est minimal. Ainsi, si

$$\max_{c \in C} \mathrm{wt}(c) < d \cdot \frac{q}{q-1}$$

alors tout mot dont la composante non nulle la plus à gauche est égale à 1 est minimal.

Démonstration. Si c n'est pas minimal, alors, par le Lemme 1.2, il couvre un mot minimal c' tel que $c \neq ac'$ pour tout $a \in \mathbb{F}_q^*$. On considère les $q-1$ mots

$$c_a = c - ac', \qquad a \in \mathbb{F}_q^*.$$

On a que (exercice)

$$S := \sum_{a \in \mathbb{F}_q^*} \mathrm{wt}(c_a) = (q-1) \cdot \mathrm{wt}(c) - \mathrm{wt}(c').$$

Ainsi le poids moyen des vecteurs est $S/(q-1)$ et l'un des vecteurs, disons $c_{\overline{a}}$, a poids au plus $S/(q-1)$:

$$\mathrm{wt}(c_{\overline{a}}) \leq \frac{S}{q-1} = \mathrm{wt}(c) - \frac{\mathrm{wt}(c')}{q-1} < d \cdot \frac{q}{q-1} - \frac{d}{q-1} = d,$$

ce qui est une contradiction, car $c_{\overline{a}} \in C$ non nul. □

Soit p un nombre premier, m un entier positif, $q := p^m$ et Tr la fonction trace de $\mathbb{F}_q$ sur $\mathbb{F}_p$ ($x \mapsto x+x^p+\ldots+x^{p^{m-1}}$). Pour un sous-ensemble $D := \{d_1, \ldots, d_n\}$ de $\mathbb{F}_q$ (*defining set*), soit

$$C_D := \{(\mathrm{Tr}(xd_1), \ldots, \mathrm{Tr}(xd_n)) \mid x \in \mathbb{F}_q\} \subseteq \mathbb{F}_p^n$$

un code linéaire (exercice). Dans ce contexte les codes avec

$$D = \{x \in \mathbb{F}_q^* \mid \mathrm{Tr}(f(x)) = 0\}$$

pour $f : \mathbb{F}_q \to \mathbb{F}_q$ sont très étudiés dans le contexte décrit ci-dessus (voir par exemple [13]).

Un autre développement possible est présenté dans [11].

Chapitre 2

Systèmes d'authentification

Un système d'authentification est une méthode visant à fournir la preuve (ou des éléments de preuve) qu'une entité (personne, service ou machine) est bien celle qu'elle prétend être.

Alice veut transmettre à Bob un message (m, t), où t est une étiquette associée à m, dépendante d'une clé k connue par Alice et Bob, qui doit permettre à Bob de vérifier l'identité d'Alice.

Un schéma d'authentification systématique est donné par

$$(M, T, K, E := \{E_k \mid k \in K\})$$

où $E_k : M \to T$ est la fonction de codage associé à la clé k.

Alice et Bob choisissent une clé $k \in K$. Alice calcule $t = E_k(m)$ et envoie (m, t) à Bob, lequel peut vérifier si $t = E_k(m)$.

Il y a deux attaques possibles :

1. l'attaquant peut essayer de forger (m, t) à partir de zéro, en espérant qu'il soit accepté par Bob. Cela s'appelle le **attaque d'usurpation d'identité**
2. l'attaquant peut observer un couple valide (m, t) et essayer de le modifier, c'est à dire utiliser une clé k' telle que $E_{k'}(m) = t$ pour forger un message $(m', t' = E_{k'}(m'))$. Cela s'appelle **attaque de substitution**.

Les probabilités maximales de succès de ces attaques sont

$$P_I := \max_{m \in M, t \in T} \frac{\#\{k \in K \mid E_k(m) = t\}}{\#K}$$

pour l'attaque d'usurpation d'identité et

$$P_S := \max_{m \neq m' \in M, t, t' \in T} \frac{\#\{k \in K \mid E_k(m) = t \text{ et } E_k(m') = t'\}}{\#\{k \in K \mid E_k(m) = t\}}$$

pour l'attaque de substitution.

En général on suppose que M et K ont une distribution uniforme et on veut que $k \mapsto E_k$ soit bijective.

Proposition 2.1. *On a*

$$P_I \geq \frac{1}{\#T} \quad et \quad P_S \geq \frac{1}{\#T}$$

Démonstration. Clairement, pour tout $m \in M$, on a

$$K = \bigsqcup_{t \in T} \{k \in K \mid E_k(m) = t\},$$

de manière que

$$\sum_{t \in T} \frac{\#\{k \in K \mid E_k(m) = t\}}{\#K} = 1.$$

Ainsi, P_I, qui est le maximum, est au moins aussi grand que la moyenne (sur T). L'autre inégalité suit de la même manière (exercice). □

Un système d'authentification est dit **optimal** si on a l'égalité.

2.1 Une construction générale basée sur les codes correcteurs

On va présenter une construction générale basée sur les codes correcteurs d'erreurs qui a été proposée dans [12].

Soit C un code de longueur n et de cardinalité μ sur un alphabet A, i.e. $C \subset A^n$ et $\#C = \mu$. On suppose que $(A, +)$ soit un groupe abélien avec q éléments. On indique avec $c_i = (c_{i,0}, \ldots, c_{i,n-1})$, $i \in \{0, \ldots, \mu - 1\}$, tous les mots de C.

Définition 2.1. *Un* ***système d'authentification cartésien*** *est donné par*

$$(M, T, K, E) := (\mathbb{Z}/\mu\mathbb{Z}, A, \mathbb{Z}/n\mathbb{Z} \times A, \{E_k \mid k \in K\})$$

où, pour tout $(k_1, k_2) \in K$ *et* $m \in M$*, on a* $E_k(m) = c_{m,k_1} + k_2$.

Sous quelles conditions (sur le code C) a-t-on que $k \mapsto E_k$ est une bijection (exercice) ?

2.1.1 Codes linéaires

Cette construction générale peut-être spécialisée à des cas particulier, par exemple en ajoutant l'hypothèse de linéarité : soit C un $[n, \kappa, d]_q$ code. On a donc

$$(M, T, K, E) := (\mathbb{Z}/q^\kappa\mathbb{Z}, \mathbb{F}_q, \mathbb{Z}/n\mathbb{Z} \times \mathbb{F}_q, \{E_k \mid k \in K\})$$

où, pour tout $(k_1, k_2) \in K$ et $m \in M$, on a $E_k(m) = c_{m,k_1} + k_2$. Dans ce cas, on a le résultat suivant.

Théorème 2.1.

$$P_I = \frac{1}{q} \quad et \quad P_S = \max_{0 \neq c \in C} \max_{u \in \mathbb{F}_q} \frac{N(c,u)}{n} \geq 1 - \frac{d}{n}.$$

où $N(c,u) := \#\{i \mid c_i = u\}$.

Démonstration. Soit $m \in M$ et $t \in T$ fixés. Si on fixe k_1 aussi, on a exactement un k_2 tel que $c_{m,k_1} + k_2 = t$, de manière que $\#\{k \in K \mid E_k(m) = t\} = \#\mathbb{Z}/n\mathbb{Z} = n$. Alors

$$P_I = \frac{n}{n \cdot q} = \frac{1}{q}.$$

De plus, pour tout $t, t' \in T$ et $m, m' \in M$, $m \neq m'$, on a

$$\#\{k \in K \mid E_k(m) = t \text{ et } E_k(m') = t'\} =$$

$$= \#\{k \in K \mid c_{m,k_1} + k_2 = t \text{ et } c_{m',k_1} + k_2 = t'\} =$$

$$= \#\{k \in K \mid c_{m,k_1} + k_2 = t \text{ et } t - t' = c_{m,k_1} - c_{m',k_1}\}$$

$$= \#\{k_1 \in \mathbb{Z}/n\mathbb{Z} \mid t - t' = c_{m,k_1} - c_{m',k_1}\} = N(c_m - c_{m'}, t - t')$$

de manière que

$$P_S = \max_{0 \neq c \in C} \max_{u \in \mathbb{F}_q} \frac{N(c,u)}{n} \geq \max_{0 \neq c \in C} \frac{N(c,0)}{n} = \frac{n-d}{d}$$

□

Cette construction exige donc que tous les éléments de $\mathbb{F}_q$ se produisent plus ou moins également souvent dans chaque mot non nul de C. Ainsi la distance minimale ne doit pas être trop grande (mais aussi pas trop petite, selon la dernière inégalité).

Exercice 2.1. *Trouver P_I et P_S dans le cas où $C = \hat{\mathcal{H}}_3$.*

Exercice 2.2. *Trouver P_I et P_S dans le cas où $C = \mathcal{H}_m^{\perp}$.*

D'autres exemples de codes sont présentés dans [12].

2.1.2 Codes non linéaires

On commence par une définition.

Définition 2.2. *Une $(q, \kappa; \lambda)$ difference matrix $D = (d_{i,j})$, avec $d_{i,j} \in A$, est une matrice $\kappa \times q\lambda$ telle que, pour tout $1 \leq h < l \leq \kappa$, la liste*

$$d_{h,1} - d_{l,1}, d_{h,2} - d_{l,2}, \ldots, d_{h,q\lambda} - d_{l,q\lambda}$$

contient chaque élément de A exactement λ fois.

Exemples : si $A = \mathbb{F}_2$ on a que

$$H(2) = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$$

est une $(2, 2; 1)$ *difference matrix* et

$$H(4) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

est une $(2, 4; 2)$ *difference matrix*.

Soit C un code de longueur n et de cardinalité μ sur un alphabet A, i.e. $C \subset A^n$ et $\#C = \mu$. On suppose que $(A, +)$ soit un groupe abélien avec q éléments. On indique avec $c_i = (c_{i,0}, \ldots, c_{i,n-1})$, $i \in \{0, \ldots, \mu - 1\}$, tous les mots de C. Soit

$$X(C) = \begin{bmatrix} c_0 \\ \vdots \\ c_{\mu-1} \end{bmatrix}$$

la matrice $\mu \times n$ sur A dont les lignes sont les mots de C. On peut montrer (exercice), avec des arguments similaires à ceux qui sont dans la preuve du Théorème 2.1, le résultat suivant.

Théorème 2.2. *Le système d'authentification donné par C a*

$$P_I = P_S = \frac{1}{q}$$

si et seulement si $X(C)$ est une $(q, \mu; n/q)$ difference matrix sur A.

Les systèmes ci-dessus sont optimaux. Il est donc intéressant d'étudier des exemples de codes qui ont cette propriété. Cet étude est strictement lié à l'étude des matrices d'Hadamard généralisées.

Définition 2.3. *Une* matrice d'Hadamard généralisé $GH(q, \lambda)$ *est une difference matrix de paramètres $(q, q\lambda; \lambda)$. Une matrice d'Hadamard $H(4n)$ est une $GH(2, 2n)$ sur $A \cong (\{-1, 1\}, \cdot)$.*

Dans [6], les auteurs présentent une manière de construire des matrices d'Hadamard généralisés (qui a été introduite pour la première fois probablement par W. de Launey en 1992 [10]) : soit f une fonction d'un groupe abélien $(G, +)$ d'ordre r sur le groupe abélien $(A, +)$ d'ordre q et supposons que q divise r. Soit $G = \{g_0, \ldots, g_{r-1}\}$ et

$$D(f) = \begin{bmatrix} f(g_0 + g_0) & f(g_0 + g_1) & \cdots & f(g_0 + g_{n-1}) \\ f(g_1 + g_0) & f(g_1 + g_1) & \cdots & f(g_1 + g_{n-1}) \\ \vdots & \vdots & & \vdots \\ f(g_{r-1} + g_0) & f(g_{r-1} + g_1) & \cdots & f(g_{r-1} + g_{n-1}) \end{bmatrix}$$

une matrice $r \times r$ sur A.
On a que $D(f)$ est une $GH(q, r/q)$ si et seulement si, pour tout $g \in G$, on a que

$$f(x+g) - f(x)$$

prends comme valeur chaque élément de A exactement r/q fois. Cette propriété est équivalente à la non linéarité parfaite de f (voir [6]). Cela nous fournit des exemples de fonctions. Par exemple : soit $f : \mathbb{F}_q^{2t} \to \mathbb{F}_q$, définie par

$$f(x_1, x_2, \ldots, x_{2t}) = x_1x_2 + x_3x_4 + \cdots + x_{2t-1}x_t.$$

On a que f est parfaitement non linéaire (voir [6]), de manière que $D(f)$ est une (q, q^{2t}, q^{2t-1}) *difference matrix*, i.e. une $GH(q, q^{2t-1})$.

Exercice 2.3. *Soit $q = 3$ et $t = 1$. Vérifier que la matrice obtenue en choisissant $f(x_1, x_2) = x_1x_2$ est une $(3, 9, 3)$ difference matrix. Quelle est donc la valeur de P_I et de P_S dans ce cas ?*

2.2 Un autre exemple

On donne un autre exemple de système d'authentification, qui utilise la notion de non linéarité des fonctions sur les corps finis. On va pas développer cette théorie ici, mais on le présente quand même pour suggérer d'autres possibilités de construction.

Soit $\mathbb{F}_{2^h}$ un sous-corps de $\mathbb{F}_{2^n}$ et soit $\varphi : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ une fonction.

Définition 2.4. *La mesure de non linéarité de φ est*

$$\mathrm{NL}(\varphi) := \min_{\alpha \in \mathcal{A}} \mathrm{d}(\varphi, \alpha)$$

où $\mathcal{A}$ est l'ensemble des fonctions affines de $\mathbb{F}_{2^n}$ sur soi-même et $\mathrm{d}(\varphi, \alpha) = \#\{x \in \mathbb{F}_{2^n} \mid \varphi(x) \neq \alpha(x)\}$.

Soit

$$M = \mathbb{F}_{2^n} \times \mathbb{F}_{2^n}$$

l'ensemble des messages,

$$T := \mathbb{F}_{2^h}$$

l'ensemble des étiquettes,

$$K := \mathbb{F}_{2^n} \times \mathbb{F}_{2^h}$$

l'ensemble des clés et $E := \{E_k \mid k \in K\}$, où

$$E_k(m) := \mathrm{Tr}(m_1\varphi(k_1) + m_2k_1) + k_2$$

pour $k = (k_1, k_2) \in K$ et $m = (m_1, m_2) \in M$. Tr est la fonction trace de $\mathbb{F}_{2^n}$ sur $\mathbb{F}_{2^h}$

$$\mathrm{Tr}(a) = \sum_{j=0}^{\frac{n}{h}-1} a^{2^{h \cdot j}}.$$

Lemme 2.1. *La fonction $k \mapsto E_k$ est une bijection entre K et E.*

Démonstration. Il suffit de montrer que si $k = (k_1, k_2)$ et $k' = (k'_1, k'_2)$ sont deux clés telles que $E_k = E_{k'}$ alors $k = k'$ (injectivité).

$E_k((0,0)) = k_2 = k'_2 = E_{k'}((0,0))$.

$E_k((0,a)) = \mathrm{Tr}(ak_1) + k_2 = \mathrm{Tr}(ak'_1) + k'_2 = E_{k'}((0,a))$, de manière que $\mathrm{Tr}(a(k_1 - k'_1)) = 0$ pour tout $a \in \mathbb{F}_{2^n}$, ce qui implique (exercice) $k_1 = k'_1$. □

Lemme 2.2. *On a*

$$P_I = \frac{1}{2^h}.$$

Démonstration. La fonction

$$\mathbb{F}_{2^n} \to \{(k_1, k_2) \in K \mid \mathrm{Tr}(m_1\varphi(k_1) + m_2 k_1) + k_2 = t\}, \quad k_1 \mapsto (k_1, k_2)$$

est bijective (pour tout k_1 on a exactement un choix pour k_2), de manière que

$$P_I := \max_{m \in M, t \in T} \frac{\#\{k \in K \mid E_k(m) = t\}}{\#K} = \frac{2^n}{2^{n+h}}.$$

□

Lemme 2.3 ([7])**.**

$$P_S \leq \frac{1}{2^h} + \left(1 - \frac{1}{2^h}\right)\left(1 - \frac{\mathrm{NL}(\varphi)}{2^{n-1}}\right).$$

Chapitre 3

Cryptographie post-quantique

Les ordinateurs quantiques, qui fonctionnent avec des qubits (bits quantiques), ont la capacité d'être dans plusieurs états simultanément. Au moins un algorithme conçu pour utiliser un circuit quantique, l'algorithme de Shor, rendrait possible de nombreux calculs combinatoires hors de portée d'un ordinateur classique en l'état actuel des connaissances.

La plupart des algorithmes cryptographiques à clé publique classiques ne sont pas résistant face à de telles attaques, par exemple le cryptosystème RSA, dont la sécurité tombe si la factorisation peut être résolue facilement, ne l'est pas, puisque l'algorithme de Shor pour factoriser un nombre fonctionne en temps polynomial. Pour être plus précis, la factorisation et le logarithme discret (et sa version elliptique, ce qui élimine aussi la cryptographie à base de couplages), qui sont deux problèmes sur lesquels repose la sécurité des algorithmes de chiffrement classiques (chiffrement RSA, cryptosystème El Gamal, etc.), sont résolus par cette attaque.

La **cryptographie basée sur les codes correcteurs** est une solution pour concevoir des cryptosystèmes post-quantiques introduite en 1978 par Robert McEliece [17]. Le problème difficile sous-jacent venant de la théorie des codes provient du fait qu'il est difficile, étant donné une famille de codes, de déterminer quel code a été utilisé pour le codage, la clef privée étant la matrice génératrice du code (permettant de corriger l'erreur), et la clef publique étant une version randomisée de cette matrice, permettant de générer des mots du code, sans pouvoir décoder. Ainsi pour chiffrer un message, l'expéditeur rajoute une erreur à son message encodé, ce qui le rend inintelligible pour une personne ne disposant pas de la matrice génératrice. La principale difficulté de la cryptographie basée sur les codes correcteurs est donc de trouver une famille de codes efficaces et exhibant la difficulté nécessaire. Une famille qui possède ces propriétés est celle des codes de Goppa.

3.1 Codes de Goppa

Soit m un entier positif et $K = \mathbb{F}_{q^m}$ un corps fini de cardinalité q^m. Pour $g(x) \in K[x]$ (polynôme de Goppa) on peut définir l'anneau quotient

$$Q_g := K[x]/\langle g(x)\rangle.$$

Lemme 3.1. *Soit $a \in K$ tel que $g(a) \neq 0$. Alors $(x-a)$ est inversible dans Q_g et*

$$(x-a)^{-1} = -\frac{1}{g(a)} \cdot \frac{g(x)-g(a)}{x-a}.$$

Démonstration. Par Euclide on a $g(x) = f(x)(x-a) + g(a)$, d'où $f(x) = \frac{g(x)-g(a)}{x-a}$. De plus $f(x)(x-a) \equiv -g(a) \bmod g(x)$ de manière que

$$f(x) \cdot (-g(a)^{-1}) \equiv (x-a)^{-1} \bmod g(x).$$

L'unicité de l'élément inverse est à montrer comme exercice. □

Définition 3.1. *Soit $A = \{a_1, \ldots, a_n\}$, une liste ordonnée d'éléments dans K, avec $a_i \neq a_j$ pour tout $i \neq j$ et $g(a_i) \neq 0$ pour tout i. Alors le code de Goppa associé à $g(x)$ et A est*

$$\Gamma(A, g(x)) = \left\{ c \in \mathbb{F}_q^n \;\middle|\; \sum_{i=1}^{n} \frac{c_i}{x-a_i} = 0 \text{ in } Q_g \right\}$$

Souvent on prend $A = \{a \mid g(a) \neq 0\}$. Si $g(x)$ est irréductible, le code de Goppa correspondant est appelé irréductible.

La longueur de $\Gamma(A, g(x))$ est $n = \#A$ par définition. Soit $r = \deg(g(x))$. À partir du Lemme 3.1, on peut déduire qu'une matrice de parité pour $\Gamma(A, g(x))$ est

$$H := \begin{bmatrix} 1 & 1 & \cdots & 1 \\ a_1 & a_2 & \cdots & a_n \\ a_1^2 & a_2^2 & \cdots & a_n^2 \\ \vdots & \vdots & \ddots & \vdots \\ a_1^{r-1} & a_2^{r-1} & \cdots & a_n^{r-1} \end{bmatrix} \cdot \begin{bmatrix} g(a_1)^{-1} & 0 & \cdots & 0 \\ 0 & g(a_2)^{-1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & g(a_n)^{-1} \end{bmatrix}$$

(attention que c'est une matrice avec entrées dans K et pas dans $\mathbb{F}_q$) qui nous permet de montrer les résultats suivants :

$$\dim(\Gamma(A, g(x))) \geq n - mr \quad \text{et} \quad \mathrm{d}(\Gamma(A, g(x))) \geq r + 1.$$

On peut montrer (voir [16]) que dans le cas binaire, si $g(x)$ n'a pas de racines multiples, on a

$$\mathrm{d}(\Gamma(A, g(x))) \geq 2r + 1.$$

Exercice 3.1. *Soit $m = 3$, $q = 2$, $\alpha \in \mathbb{F}_8$ tel que $\alpha^3 = \alpha + 1$. Soit*

$$g(x) := x^2 + x + 1 \in \mathbb{F}_8[x],$$

qui est irréductible. Soit $A := \{0, 1, \alpha, \alpha^2, \ldots, \alpha^6\} = \mathbb{F}_8$. Montrer qu'une matrice génératrice pour $\Gamma(A, g(x))$ est donnée par

$$G := \left[\begin{smallmatrix} 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \end{smallmatrix}\right].$$

Notons que sa dimension est $2 = 8 - 3 \cdot 2$ et sa distance minimale est $5 = 2 \cdot 2 + 1$.

3.1.1 Décodage

Soit $c = (c_1, \ldots, c_n) \in \Gamma(A, g(x))$, avec $g(x)$ irréductible de degré r, et supposons qu'on reçoit $y = (y_1, \ldots, y_n)$. L'erreur est $e = y - c = (e_1, \ldots, e_n)$. Soit $M := \{i \mid e_i \neq 0\}$. Supposons que $1 \leq \#M \leq r$. On appelle **polynôme syndrome** le polynôme suivant :

$$S_y(x) := \sum_{i=1}^{n} \frac{e_i}{x - a_i} \left(= \sum_{i=1}^{n} \frac{y_i}{x - a_i} \right) \text{ in } Q_g.$$

Ce polynôme peut être calculé par le destinataire en utilisant y et la matrice de parité.

Définition 3.2. *Le polynôme **localisateur de l'erreur** dans y pour $\Gamma(A, g(x))$ est*

$$\sigma_y(x) := \prod_{i \in M} (x - a_i).$$

Pour les codes binaires ce polynôme est suffisant pour décoder. Sinon il faut calculer un autre polynôme (voir [19]). L'**algorithme de Patterson** pour calculer le polynôme localisateur de l'erreur suit les passages suivantes :

- Calcule $v(x) := \sqrt{S_y(x)^{-1} - x}$ in Q_g ;
- Calcule $a(x)$ et $b(x)$ tels que $\deg(a(x)) \leq r/2$, $\deg(b(x)) \leq (r-1)/2$ et $v(x) = a(x)/b(x)$ in Q_g.
- $\sigma_y(x) := a(x)^2 + x \cdot b(x)^2$.

Voir [19] pour la preuve de l'exactitude de cet algorithme (on peut en avoir une idée plus claire dans le cas binaire si on observe que $S_y(x) = \sum_{i \in M} \frac{1}{x - a_i} = \frac{n(x)}{\sigma_y(x)}$ pour un certain polynôme $n(x)$, de manière que...). Une fois qu'on a $\sigma(x)$, on peut le factoriser pour localiser les erreurs et les corriger. On peut montrer que le temps d'exécution de l'algorithme de Patterson est $\mathcal{O}(nr)$ (donc c'est un algorithme très rapide).

Remarque 3.1. *Notons que le décodage dépend fortement de la connaissance de A et de $g(x)$.*

3.2 Le cryptosystème de McEliece

Le cryptosystème presenté par Robert J. McEliece dans [17] est basé sur les codes de Goppa. En particulier, il considère les codes de Goppa irréductibles sur $\mathbb{F}_2$: soit $g(x)$ un polynôme dans $\mathbb{F}_{2^m}[x]$ irréductible de degré r ; alors $\Gamma(\mathbb{F}_{2^m}, g(x))$ est un code de longueur 2^m, dimension $k \geq n - rm$ et il peut corriger r erreurs.

Voici les passages du cryptosystème :

1. Alice choisi m et r et elle cherche avant tout un polynôme $g(x)$ dans $\mathbb{F}_{2^m}[x]$ irréductible de degré r au hasard. Cela est faisable, parce qu'il y a environ $1/r$ polynômes irréductibles de degré r dans $\mathbb{F}_{2^m}[x]$ et il existe un algorithme rapide pour tester l'irréductibilité (voir [3]).
2. Alice calcule une matrice génératrice G pour $\Gamma(\mathbb{F}_{2^m}, g(x))$.
3. Alice sélectionne aléatoirement une matrice binaire $k \times k$ non singulière S et une matrice binaire $n \times n$ de permutation P (on rappelle qu'une matrice de permutation est une matrice binaire telle qu'il y a un et un seul 1 par ligne et par colonne). Elle calcule $G' = SGP$. Notons que G' est une matrice génératrice d'un code équivalent à $\Gamma(\mathbb{F}_{2^m}, g(x))$. Le couple (G', r) est la **clé publique**.
4. Bob doit communiquer un message $m \in \mathbb{F}_2^k$ à Alice (si jamais le message est plus long, il le coupe en blocs). Il choisi au hasard un vecteur $e \in \mathbb{F}_2^n$ de poids r et il calcule $c := mG' + e$, qui est le message chiffré, et il l'envoie à Alice.
5. Alice reçoit c et elle calcule $cP^{-1} = mSG + eP^{-1}$. Puisque eP^{-1} a poids r, elle trouve mS en utilisant l'algorithme de Patterson et finalement elle calcule $m = mSS^{-1}$.

Exercice 3.2. *Soient*

$$G := \left[\begin{smallmatrix} 1&1&0&0&1&0&1&1 \\ 0&0&1&1&1&1&1&1 \end{smallmatrix}\right],\ S := \left[\begin{smallmatrix} 1&1 \\ 0&1 \end{smallmatrix}\right],\ P := \left[\begin{smallmatrix} 0&1&0&0&0&0&0&0 \\ 1&0&0&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0 \\ 0&0&1&0&0&0&0&0 \\ 0&0&0&0&0&1&0&0 \\ 0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&0&0&1 \\ 0&0&0&0&0&0&1&0 \end{smallmatrix}\right],$$

$$m := \left[\begin{smallmatrix} 1&0 \end{smallmatrix}\right] \text{ et } e := \left[\begin{smallmatrix} 0&1&0&0&0&1&0&0 \end{smallmatrix}\right],$$

Calculer c. Déchiffrer $c' := \left[\begin{smallmatrix} 0&0&0&0&1&1&1&1 \end{smallmatrix}\right]$.

Il est clair qu'implémenter cet algorithme est assez simple. Il nous reste d'étudier sa sécurité. Combien est-il difficile pour Eve déterminer m, en connaissant G' et c ?

Deux attaques possibles :

1. Eve essaie de reconstruire G (et donc $g(x)$) à partir de G', afin de pouvoir utiliser l'algorithme de Patterson. Cela semble être sans possibilité concrète de succès, parce qu'il y a trop de possibilités pour G, S et P.

2. Eve essaie de reconstruire m à partir de c, sans essayer d'avoir G. Cela semble être peut-être plus prometteur, mais on voit bien que le problème fondamental qu'il faut résoudre c'est de décoder un code linéaire presque arbitraire. Dans [4], ils montrent que cela est un problème NP-complet, de manière que si le paramètres du code sont assez grands, on s'attend qu'un attaque ne soit pas faisable.

Remarque 3.2. *Dans* [17]*, McEliece prend en considération* $n = 1024 = 2^{10}$ *et* $r = 50$*. Dans ce cas, il y a environ* 10^{149} *polynômes de Goppa irréductibles possibles et un nombre astronomique de choix possibles pour* S *et* P*. Le deuxième attaque (si on le fait par force brute) est également très difficile.*

Remarque 3.3. *Un attaque plus prometteur est le suivant : on choisi* k *des* n *coordonnées au hasard, en espérant qu'aucune est touchée par l'erreur, et on calcule le message à partir de ces coordonnées. La probabilité de n'avoir pas une erreur dans le* k *coordonnées choisies est environ*

$$\left(1 - \frac{r}{n}\right)^k$$

La complexité de résoudre un système de k *équations en* k *inconnus est* $\mathcal{O}(k^3)$*, ce qui nous donne que cet attaque est en tout cas difficile avec le paramètres ci-dessus. On note en plus que cet attaque implique que, pour une meilleure sécurité du système,* r *doit être grand. On va voir d'autres attaques de ce type (décodage par ensemble d'information).*

Remarque 3.4. ***Avantages*** *: facile à implémenter et sûr à des attaques quantiques.* ***Désavantages*** *: largeur de la clé.*

Remarque 3.5. *On peut substituer les codes de Goppa avec d'autres codes, à condition qu'ils aient un algorithme de décodage efficient et qu'ils soient indistinguables des codes choisis au hasard. Codes de Reed-Solomon généralisés et leur sous-codes, codes de Reed-Muller, codes algébriques-géométriques, codes concaténés ont été proposés, mais toujours un attaque efficace a été trouvé. Seulement le codes de Goppa semblent être sûrs.*

3.3 Le cryptosystème de Niederreiter

Le cryptosystème présenté par Harald Niederreiter en 1986 est une variation du cryptosystème de McEliece, avec un chiffrement dix fois plus rapide.

Voici les passages du cryptosystème :

1. Alice choisi m et r et elle cherche un polynôme $g(x)$ dans $\mathbb{F}_{2^m}[x]$ irréductible de degré r au hasard.
2. Alice calcule une matrice de parité H pour $\Gamma(\mathbb{F}_{2^m}, g(x))$.
3. Alice sélectionne aléatoirement une matrice binaire $(n-k) \times (n-k)$ non singulière S et une matrice binaire $n \times n$ de permutation P. Elle calcule $H' = SHP$. Notons que H' est une matrice de parité d'un code équivalent à $\Gamma(\mathbb{F}_{2^m}, g(x))$. Le couple (H', r) est la **clé publique**.

4. Bob doit communiquer un message $m' \in \mathbb{F}_2^r$ à Alice. Il l'encode comme un vecteur m dans $\in \mathbb{F}_2^n$ de poids au plus r et il calcule $c := H'm^T$, qui est le message chiffré, et il l'envoie à Alice.
5. Alice reçoit c et elle calcule $S^{-1}c = HPm^T$. Puisque Pm^T a poids r, elle trouve Pm^T en utilisant l'algorithme de Patterson et finalement elle calcule $m = P^{-1}Pm^T$.

Donc le cryptosystème de Niederreiter est une sorte de dual du cryptosystème de McEliece. Sa sécurité est équivalent.

Théorème 3.1. *Les cryptosystèmes de Niederreiter et de McEliece ont une sécurité équivalente (et cela ne dépend pas du code C choisi).*

Démonstration. Soit (G', r) la clé publique du cryptosystème de McEliece. On peut facilement calculer une matrice de parité H' par le code avec matrice génératrice G'. Dans le cryptosystème de McEliece le message chiffré est $c = mG' + e$. Si on multiplie chaque coté par H'^T, on obtient

$$z = cH'^T = mG'H'^T + eH'^T = eH'^T,$$

car $G'H'^T = 0$. Donc $z^T = H'e^T$, avec e un vecteur de poids au plus r. Si un attaquant peut casser le cryptosystème de Niederreiter, il peut obtenir e et donc m, c'est-à-dire il peut casser le cryptosystème de McEliece.

Vice versa, soit (H', r) la clé publique du cryptosystème de Niederreiter. On peut facilement calculer une matrice génératrice G' par le code avec matrice de parité H'. Dans le cryptosystème de Niederreiter le message chiffré est $c = H'm^T$, avec m de poids au plus r. Soit $z \in \mathbb{F}_2^n$ un vecteur quelconque tel que $c = H'z^T$. Ce vecteur est forcement de la forme $z = m'G' + m$ pour quelque $m' \in \mathbb{F}_2^k$, qu'on est capable de trouver si on est capable de casser le cryptosystème de McEliece. Si on trouve m', on obtient $m = z - m'G'$, de manière qu'on a cassé le cryptosystème de Niederreiter. □

3.4 Un schéma de signature

La signature numérique est un mécanisme visant à garantir l'intégrité d'un document électronique et d'en authentifier l'auteur. Pour longtemps on a cru que McEliece ne pouvait pas être utilisé pour la signature. Cependant, dans [9], les auteurs montrent qu'il est en effet possible de construire un schéma de signature basé sur le cryptosystème de Niederreiter.

Afin d'obtenir une signature numérique efficace, nous avons besoin de deux choses : un algorithme capable de calculer une signature pour n'importe quel document de telle sorte qu'ils identifient leur auteur de manière unique, et un algorithme de vérification rapide accessible à tous. Un cryptosystème à clé publique peut être utilisée comme schéma de signature comme suit :

1. hacher (avec un algorithme de hachage public) le document à signer ;
2. déchiffrer le document haché comme s'il s'agissait d'un texte chiffré ;

3. ajoutez le message déchiffré au document en tant que signature.

Pour vérifier la signature on applique simplement la fonction de chiffrement publique à la signature et on vérifie que le résultat est bien le message haché. Dans le cas du cryptosystème de McEliece ou de Niederreiter, le point 2 échoue. La raison en est que si l'on considère un message haché aléatoire, il correspond généralement à une erreur de poids supérieur à r. En d'autres termes, il est difficile de générer un texte chiffré aléatoire à moins qu'il ne soit explicitement produit en tant que sortie de l'algorithme de chiffrement.

Le schéma de Courtois, Finiasz et Sendrier spécifie alors une manière déterministe de modifier le message haché jusqu'à ce que l'on trouve un message qui peut être déchiffré. Le choix des paramètres du code est lié à la probabilité qu'un message aléatoire soit déchiffrable. Courtois, Finiaz et Sendrier suggèrent les valeurs des paramètres $n = 2^{16}$ et $r = 9$, pour lesquels la probabilité de déchiffrer un message aléatoire est environ $\frac{1}{9!}$. Par conséquent, un message déchiffrable est trouvé après un nombre attendu de 9! tentatives. On ajoute un compteur, i, au message original (avant hachage), pour produire un message haché légèrement modifié. Soit i_0 la première valeur de i pour laquelle le message haché est déchiffrable. Dans ce cas, le message déchiffré est un mot, z, de longueur n et de poids 9, de sorte que Hz^T est égal au message haché avec le compteur i_0 ajouté. La signature sera z combinée avec la valeur i_0 qui sert pour la vérification. Cette signature est jointe au message original.

3.5 Variantes avec d'autres codes

Niederreiter avait suggéré d'utiliser les codes de Reed-Solomon généralisés à la place des codes de Goppa. On va rappeler leur définition et on va montrer un attaque développé par Sidelnikov et Shestakov. Cela veut être un exemple des attaques possibles à des variantes des cryptosystèmes de McEliece et Niederreiter.

3.5.1 Codes de Reed-Solomon généralisés

Soit $\mathbb{F}_q$ un corps fini et $1 \leq k < n \leq q$ des entiers. Soit $\alpha = (\alpha_1, \ldots, \alpha_n) \in \mathbb{F}_q^n$, avec $\alpha_i \neq \alpha_j$ pour tout $i \neq j$, et $\beta = (\beta_1, \ldots, \beta_n) \in \mathbb{F}_q^n$, avec $\beta_i \neq 0$ pour tout i.

Définition 3.3. *Le code de Reed-Solomon généralisé de longueur n et dimension k, associé à α et β, est*

$$\mathrm{GRS}_{n,k}(\alpha, \beta) := \{(\beta_1 p(\alpha_1), \ldots, \beta_n p(\alpha_n)) \mid p(x) \in \mathbb{F}_q[x], \deg(p(x)) < k\}.$$

Une matrice génératrice canonique pour $\mathrm{GRS}_{n,k}(\alpha, \beta)$ est

$$G := \begin{bmatrix} \beta_1 & \ldots & \beta_n \\ \beta_1\alpha_1 & \ldots & \beta_n\alpha_n \\ \vdots & \ddots & \vdots \\ \beta_1\alpha_1^{k-1} & \ldots & \beta_n\alpha_n^{k-1} \end{bmatrix}$$

et sa distance minimale est $n-k+1$ (code MDS, exercice).

Proposition 3.1. *Le code dual d'un code généralisé de Reed-Solomon est un code généralisé de Reed-Solomon. En particulier*

$$\mathrm{GRS}_{n,k}(\alpha,\beta)^{\perp} = \mathrm{GRS}_{n,n-k}(\alpha,\gamma),$$

où

$$\gamma_i = \beta_i^{-1} \prod_{j=1, j\neq i}^{n} (\alpha_i - \alpha_j)^{-1}.$$

Démonstration. Exercice (suggestion : utiliser l'interpolation de Lagrange). □

Proposition 3.2. *Soient α et β des éléments de $\mathbb{F}_q^n$ définis comme ci-dessus. Alors*

$$\mathrm{GRS}_{n,k}(\alpha,\beta) = \mathrm{GRS}_{n,k}(a\alpha + b, c\gamma),$$

pour tout $a, b, c \in \mathbb{F}_q$, $a \neq 0$ et $c \neq 0$. Ainsi on peut toujours supposer que α_1 et α_2 (par exemple) soient deux éléments de $\mathbb{F}_q$ fixés.

Démonstration. Exercice. □

Finalement, le décodage d'un code de Reed-Solomon généralisé est similaire au décodage d'un code de Goppa et c'est facile décoder si on connaît α et β.

3.5.2 Attaque de Sidelnikov et Shestakov

Le but de l'attaque de Sidelnikov et Shestakov, présenté en [22], est de récupérer α et β à partir la connaissance de la matrice "brouillée" G'. Plus précisément, on veut récupérer des permutés $\alpha' = \alpha P$ et $\beta' = \beta P$, où $G' = SGP$, de manière que GP soit la matrice génératrice canonique de $\mathrm{GRS}_{n,k}(\alpha', \beta')$. Par la Proposition 3.2, on peut supposer $\alpha'_1 = 0$, $\alpha'_2 = 1$ et $\beta'_1 = 1$. On suppose en plus que $n \geq k+2$.

On récupére avant tout α'. On a que $G' = [A|B]$ avec A une matrice $k \times k$ inversible (cela est toujours vrai parce que les codes de Reed-Solomon généralisés sont MDS et donc chaque ensemble de k colonnes a rang k). On multiplie par A^{-1} et on obtient donc

$$B := A^{-1}G' = \begin{bmatrix} 1 & & 0 & b_{1,k+1} & \cdots & b_{1,n} \\ & \ddots & & \vdots & & \vdots \\ 0 & & 1 & b_{k,k+1} & \cdots & b_{k,n} \end{bmatrix}$$

On sait que chaque ligne $b_1, \ldots, b_k$ de B correspond à un polynôme $p_{b_i}(x)$ dans $\mathbb{F}_q[x]$ de degré au plus $k-1$. De plus, puisque la matrice est en forme systématique, $p_{b_i}(\alpha'_j) = 0$ pour tout $j \in \{1, \ldots, k\}$, $j \neq i$, de manière que

$$p_{b_i}(x) = c_{b_i} \cdot \prod_{j=1, j\neq i}^{k} (x - \alpha'_j)$$

pour un certain $c_{b_i} \in \mathbb{F}_q$.

Puisque le code est MDS, on a $b_{i,j} \neq 0$ pour tout $k+1 \leq j \leq n$ et tout $2 \leq i \leq k$, de manière que

$$\frac{b_{1,j}}{b_{i,j}} = \frac{\beta'_j p_{b_1}(\alpha'_j)}{\beta'_j p_{b_i}(\alpha'_j)} = \frac{c_{b_1}(\alpha'_j - \alpha'_i)}{c_{b_i}(\alpha'_j - \alpha'_1)} = \frac{c_{b_1}(\alpha'_j - \alpha'_i)}{c_{b_i}\alpha'_j}$$

est bien défini. On appelle $\mu_{i,j} := \frac{b_{1,j}}{b_{i,j}}$ (qui est connu) et $\lambda_i := \frac{c_{b_1}}{c_{b_i}}$ (qui est à deviner, parmi tout élément de $\mathbb{F}_q$). Puisque $\alpha'_2 = 1$, on a

$$\mu_{2,j} = \lambda_2(1 - (\alpha'_j)^{-1})$$

et donc

$$\alpha'_j = \frac{\lambda_2}{\lambda_2 - \mu_{2,j}}, \text{ pour tout } k+1 \leq j \leq n.$$

Pour les autres indices, on a (on rappelle que $n \geq k+2$),

$$\begin{cases} \mu_{i,k+1}\alpha'_{k+1} = \lambda_i(\alpha'_{k+1} - \alpha'_i) \\ \mu_{i,k+2}\alpha'_{k+2} = \lambda_i(\alpha'_{k+2} - \alpha'_i) \end{cases}$$

d'où on obtient

$$\alpha'_i = \frac{\alpha'_{k+1}\alpha'_{k+2}(\mu_{i,k+1} - \mu_{i,k+2})}{\mu_{i,k+1}\alpha'_{k+1} - \mu_{i,k+2}\alpha'_{k+2}}$$

pour tout $3 \leq i \leq k$. Donc on a obtenu α', à condition d'avoir choisi bien λ_2.

Le deuxième pas consiste à récupérer β'.

Soit G_{k+1} la matrice $k \times (k+1)$ des premières $k+1$ colonnes de GP et G'_{k+1} la matrice $k \times (k+1)$ des premières $k+1$ colonnes de G'. Soit $c = (c_1, \ldots, c_{k+1})$ une solution non triviale de $G'_{k+1}c^T = 0$. Alors on sait que $G_{k+1}c^T = 0$ (en effet $G'_{k+1} = SG_{k+1}$). Ainsi,

$$\begin{bmatrix} c_1 & \cdots & c_{k+1} \\ c_1\alpha'_1 & \cdots & c_{k+1}\alpha'_{k+1} \\ \vdots & & \vdots \\ c_1{\alpha'}_1^{k-1} & \cdots & c_{k+1}{\alpha'}_{k+1}^{k-1} \end{bmatrix} \begin{bmatrix} \beta'_1 \\ \beta'_2 \\ \vdots \\ \beta'_{k+1} \end{bmatrix} = 0,$$

de manière qu'on peut déterminer $\beta'_1, \ldots, \beta'_{k+1}$, à une constante multiplicative près. Puisque $\beta'_1 = 1$, on les a donc déterminé. Pour déterminer les autres β'_i, on peut déterminer S : en effet soit G_k la matrice $k \times k$ des premières k colonnes de GP et G'_k la matrice $k \times k$ des premières k colonnes de G'. Alors $S = G'_k G_k^{-1}$. Finalement $GP = S^{-1}G'$, d'où on peut récupérer les derniers β_i.

Exercice 3.3. *Reconstituer α et β dans $\mathbb{F}_7^4$ tels que*

$$G := \begin{bmatrix} 1 & 0 & 6 & 3 \\ 0 & 1 & 2 & 6 \end{bmatrix}$$

soit une matrice génératrice de $\mathrm{GRS}_{4,2}(\alpha, \beta)$.

L'attaque de Sidelnikov et Shestakov utilise la vulnérabilité liée au fait que le code public est équivalent à un code de Reed-Solomon généralisé. Cela permet d'annuler les effets de la matrice de "brouillage" S. Connaître un code équivalent au code secret suffit donc à récupérer le message. Pour éviter cette attaque, la matrice publique ne doit pas donner un code qui est équivalent au code secret. Il y a eu des propositions dans les dernières années, et il y a toujours une recherche active sur ce sujet.

3.5.3 Le produit de Schur

Un outil très utilisé pour les attaques aux cryptosystèmes basés sur les codes est le produit de Schur, qui permet très souvent de distinguer entre un code avec structure algébrique et un code linéaire quelconque (c'est-à-dire, choisi au hasard).

Définition 3.4. *Pour $x = (x_1, \dots, x_n), y = (y_1, \dots, y_n) \in \mathbb{F}_q^n$ on appelle* produit étoile *de x et y le vecteur*

$$x \star y = (x_1 y_1, \dots, x_n y_n).$$

Pour deux codes $\mathcal{C}_1, \mathcal{C}_2 \subseteq \mathbb{F}_q^n$ on appelle produit de Schur *de $\mathcal{C}_1$ et $\mathcal{C}_2$ le code*

$$\mathcal{C}_1 \star \mathcal{C}_2 := \langle x \star y \mid x \in \mathcal{C}_1, y \in \mathcal{C}_2 \rangle_{\mathbb{F}_q}.$$

En particulier, on appelle $\mathcal{C}^{(2)} := \mathcal{C} \star \mathcal{C}$ le carré de Schur de $\mathcal{C}$.

Le produit de Schur est commutatif et bilinéaire (exercice). Cela implique que, si $b_1, \dots, b_k$ est une base de $\mathcal{C}$, alors

$$\mathcal{C}^{(2)} = \langle b_i \star b_j \mid i, j \in \{1, \dots, k\}, j \geq i \rangle_{\mathbb{F}_q},$$

de manière que

$$\dim \mathcal{C}^{(2)} \leq \min \left\{ \binom{k+1}{2}, n \right\}.$$

On peut montrer le résultat suivant.

Proposition 3.3. *Soit $n > \binom{k+1}{2}$ et soit $\mathcal{C}$ un code choisi avec probabilité uniforme parmi tous les codes de longueur n et dimension k. Alors $\dim \mathcal{C}^{(2)} = \binom{k+1}{2}$ presque sûrement (i.e. avec probabilité 1).*

Démonstration. Voir la Proposition 2 de [14]. □

D'autre part, très souvent des codes avec une structure algébrique ont une dimension beaucoup plus petite. Par exemple, on peut montrer facilement (exercice) que

$$GRS_{n,k}(\alpha, \beta) \star GRS_{n,k'}(\alpha, \beta') = GRS_{n,k+k'-1}(\alpha, \beta \star \beta'),$$

de manière que $\dim GRS_{n,k}(\alpha, \beta)^{(2)} = \min\{2k-1, n\}$. Cette remarque si simple aide énormément l'attaquant d'un cryptosystème basé sur le codes (du type

McEliece) à distinguer le code utilisé. En effet, si on utilise seulement une matrice de permutation (ou éventuellement une matrice monomiale) on ne change pas la dimension du carré de Schur du code utilisé, car

$$(xP) \star (yP) = (x \star y)P$$

(ou éventuellement la matrice dont les coefficients sont les carrés des coefficients de P, si P est monomiale). Donc l'attaquant peut distinguer, par exemple, un code GRS d'un code choisi au hasard.

3.5.4 Masquage à deux poids

Dans [5], les auteurs proposent une méthode visant à masquer la structure algébrique du code utilisé dans le cryptosystème du type McEliece. Cette méthode diffère du schéma classique pour l'utilisation d'une matrice P non pas de permutation mais issue de l'ensemble suivant :

$$\mathcal{W}_n := \{P \in \mathrm{GL}_n(\mathbb{F}_q) \mid \text{toute ligne a poids 2}\}.$$

Pour créer une telle matrice, on peut considérer les matrices de la forme $I_n + M$ avec M une matrice monomiale. Elle ne sont pas forcement dans $\mathcal{W}_n$, mais elle le sont avec une très bonne probabilité (voir [5] pour plus de détails).

Lemme 3.2. *Pour tout $v \in \mathbb{F}_q^n$ et tout $P \in \mathcal{W}_n$, on a*

$$\mathrm{wt}(vP) \leq 2\mathrm{wt}(v).$$

Démonstration. On peut clairement écrire toute matrice $P \in \mathcal{W}_n$ comme une somme de deux matrices monomiales, disons $P = M_1 + M_2$. On a donc que

$$\mathrm{wt}(vP) = \mathrm{wt}(vM_1 + vM_2) \leq \mathrm{wt}(vM_1) + \mathrm{wt}(vM_2) = 2\mathrm{wt}(v).$$

□

Le cryptosystème suit les passages suivantes :

1. Alice choisi un $[n, k]$ code $\mathcal{C}$ r-correcteurs, avec sa matrice génératrice G, dont elle connaît un algorithme de décodage efficace.
2. Alice sélectionne aléatoirement une matrice binaire $S \in \mathrm{GL}_k(\mathbb{F}_q)$ et une matrice $P \in \mathcal{W}_n$. Elle calcule $G' = SGP^{-1}$. Le couple (G', r) est la **clé publique**.
3. Bob doit communiquer un message $m \in \mathbb{F}_2^k$ à Alice. Il choisi au hasard un vecteur $e \in \mathbb{F}_2^n$ de poids au plus $r/2$ et il calcule $c := mG' + e$, qui est le message chiffré, et il l'envoie à Alice.
4. Alice reçoit c et elle calcule $cP = mSG + eP$. Puisque eP a poids au plus r, par le Lemme 3.2, elle trouve mS en utilisant l'algorithme de décodage qu'elle connaît et finalement elle calcule $m = mSS^{-1}$.

Puisque P n'est pas une matrice monomiale, on obtient un code qui n'est pas équivalent à $\mathcal{C}$. On peut aussi formuler l'analogue de Niederreiter pour cette variante à deux poids (voir [5]).

Il semble que la multiplication par P soit un bon masquage de la structure algébrique du code. Cela devient clair en calculant le carré de Schur du code donné par la clé publique et le comparant au carré de Schur du code donné par la clé privée. Les auteurs de [5] fournissent des données de ces calculs, qui semblent montrer que, graçe à ce masquage, on obtient un carré de Schur qui a très souvent la dimension d'un carré de Schur d'un code au hasard. Plusieurs centaines de simulations donnent que, si on considère un code $GRS_{n,k}$ avec $2k-1 < n$, en multipliant par P^{-1} on obtient souvent un code dont le carré de Schur a dimension maximale, i.e. $\min\left\{\binom{k+1}{2}, n\right\}$.

3.6 Décodage par ensembles d'information

Le décodage par ensemble d'information (en anglais *Information Set Decoding*) est une technique qui s'oppose aux techniques de recherches exhaustives (qui énumèrent les mots de code ou les vecteurs d'erreur) en utilisant, comme idée principale, le fait qu'il suffit, pour décoder un vecteur v de trouver un ensemble de k positions d'information ne contenant aucune erreur.

On rappelle qu'un ensemble d'information d'un $[n,k,d]_q$ code $\mathcal{C}$ est un sous-ensemble I de $\{1,\dots,n\}$ de cardinalité k tel que le k colonnes d'une matrice génératrice G correspondantes aux indices contenu dans I ont rang k (i.e. la sous-matrice G_I qui composée par ces colonnes est inversible).

Le décodage par ensemble d'information, dans sa forme la plus simple, prend comme entrée un vecteur $y \in \mathbb{F}_q^n$ qui a distance w du code $\mathcal{C}$ (appelons c le mot de $\mathcal{C}$ le plus proche à y). On prend un ensemble d'information I au hasard en espérant que y et c soient égaux sur les coordonnées indexées par I. Si cela est vrai, $c = y_I G_I^{-1} G$, où y_I est la restriction de y aux coordonnées indexées par I. Sinon, on répète le procédé en choisissant un autre I.

Les algorithmes de Lee-Brickell et de Stern permettent une erreur (de taille controllée) dans les coordonnées indexées par I.

On va utiliser, dans ce qui suit, la notation suivante : pour tout $a \in I$, on appelle $\mathbf{g}_a$ l'unique ligne de $G_I^{-1}G$ dont la colonne indexée par a a un coefficient égale à 1.

3.6.1 L'algorithme de Lee-Brickell

On présente ici l'algorithme de Lee-Brickell dans sa forme généralisée donnée dans [20].

Soit p un entier tel que $0 \leq p \leq w$.

1. On choisit un ensemble d'information I au hasard.
2. On remplace y par $y - y_I G_I^{-1} G$.

3. Pour tout sous-ensemble $A = \{a_1, \ldots, a_p\} \subseteq I$ de p éléments et pour tout $m = (m_1, \ldots, m_p) \in (\mathbb{F}_q^*)^p$, on calcule

$$e = y - \sum_{i=1}^{p} m_i \mathbf{g}_{a_i}.$$

Si e a poids w, alors e est l'erreur cherchée. Sinon, on répète le procédé.

Si $p = 0$, le pas 3 consiste en vérifier si $y - y_I G_I^{-1} G$ a poids w (ce qui correpond à la versione la plus simple du décodage par ensemble d'information). Si $p > 0$, on est en train de vérifier tout possible e qui contiennent p coordonnées non nulles dans I.

Comment choisir un ensemble d'information I au hasard ? Il y a au moins deux méthodes : choisir un sous-ensemble de $\{1, \ldots, n\}$ de cardinalité k au hasard et vérifier, à travers l'élimination de Gauss, que la matrice correspondante est inversible, ou bien choisir les colonnes une par une et vérifier à chaque choix le rang.

Dans [20] certains simplifications par rapport aux calculs du pas 3 sont expliquées.

3.6.2 L'algorithme de Stern

L'algorithme de Stern a été originellement inventé pour trouver un mot de poids donné dans un code linéaire binaire. On donne ici sa généralisation pour un corps arbitraire présenté dans [20] : étant donnés $y \in \mathbb{F}_q^n$, un code $\mathcal{C} \subseteq \mathbb{F}_q^n$ et un entier positif w, on veut trouver un vecteur $e \in \mathbb{F}_q^n$ de poids w contenu dans la classe $y + \mathcal{C}$ (si $y = 0$, cet algorithme sert pour trouver un mot de poids w).

L'algorithme de Stern utilise deux paramètres p et ℓ (une discussion précise sur ces paramètres est présentée dans [20]) et divide l'ensemble d'information I dans deux ensembles X et Y de la même cardinalité et cherche des mots ayant exactement poids p dans les colonnes indexées par X, exactement poids p dans les colonnes indexées par Y et exactement poids 0 dans le colonnes indexées par un ensemble de cardinalité ℓ choisi au hasard dehors les colonnes indexées par I (en étant e un vecteur de poids petit, cela nous donne une restriction importante et raisonnable).

Soit p un entier tel que $0 \leq p \leq w$. Soit ℓ un entier tel que $0 \leq \ell \leq n - k$. Supposons, pour simplifier la discussion, que k soit pair.

1. On choisit au hasard un ensemble d'information I.
2. On remplace y par $y - y_I G_I^{-1} G$.
3. On choisit au hasard un sous-ensemble $X \subseteq I$ de cardinalité $k/2$.
4. On appelle $Y = I \setminus X$.
5. On choisit au hasard un sous-ensemble Z de $\{1, \ldots, n\} \setminus I$ de cardinalité ℓ.

6. Pour tout sous-ensemble $A = \{a_1, \ldots, a_p\} \subseteq X$ de p éléments on considère
$$\mathcal{V}_A := \{y - \sum_{i=1}^{p} m_i \mathbf{g}_{a_i} \mid m = (m_1, \ldots, m_p) \in (\mathbb{F}_q^*)^p\}$$
et pour tout $\alpha \in \mathcal{V}_A$ on calcule la restriction $\alpha_Z \in \mathbb{F}_q^\ell$.
7. Pour tout sous-ensemble $B = \{b_1, \ldots, b_p\} \subseteq Y$ de p éléments on considère
$$\mathcal{V}_B := \{\sum_{i=1}^{p} m'_i \mathbf{g}_{a_i} \mid m' = (m'_1, \ldots, m'_p) \in (\mathbb{F}_q^*)^p\}$$
et pour tout $\beta \in \mathcal{V}_B$ on calcule la restriction $\beta_Z \in \mathbb{F}_q^\ell$.
8. Pour tout couple (A, B) et tout couple $(\alpha, \beta) \in (\mathcal{V}_A, \mathcal{V}_B)$ tel que $\alpha_Z = \beta_Z$, on calcule
$$e = \alpha - \beta = y - \sum_{i=1}^{p} m_i \mathbf{g}_{a_i} - \sum_{i=1}^{p} m'_i \mathbf{g}_{b_i}.$$
Si e a poids w, alors e est l'erreur cherchée. Sinon, on répète le procédé.

On peut trouver dans [20] une analyse de la complexité et de la probabilité de succès de cet algorithme.

Chapitre 4

Exercices

Exercice 4.1. *Soit $\mathcal{C} \subseteq \mathbb{F}_2^6$ un code avec matrice génératrice*

$$G := \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}.$$

Déterminer la structure d'accès donnée par C. Est-elle démocratique ?

Solution : la matrice de parité de $\mathcal{C}$ est

$$H := \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

On appelle $0, 1, 2, 3, 4, 5$ les coordonnées. L'ensemble des mots qui ont la coordonnée 0 égale à 1 est

$$\{(1,0,1,0,1,0), (1,1,0,0,0,1), (1,1,0,1,1,0), (1,0,1,1,0,1)\}$$

et on peut bien vérifier qu'ils sont tous minimaux. Donc la structure d'accès donnée par C est

$$\{\{2,4\}, \{1,5\}, \{1,3,4\}, \{2,3,5\}\}.$$

Elle est démocratique dans le sens que tout dépositaire appartient exactement à 2 coalitions, mais elle n'est pas démocratique selon la définition donnée dans le cours, parce que les coalitions ont un nombre différent d'éléments.

Exercice 4.2. *Soit C un $[n, k, n-k+1]_q$ code (MDS). Montrer qu'il y a $\binom{n}{n-k+1}$ mots de poids minimal dont la composante non nulle la plus à gauche est égale à 1 et ces sont tous les mots minimaux du code.*

Solution : Un mot de poids minimal a exactement $k-1 = n-(n-k+1)$ zéros. Par le Lemme 1.1. du cours, un mot d'un code MDS est déterminé de façon unique par ses valeurs dans k coordonnées. Donc les mots de poids minimal dont la composante non nulle la plus à gauche est égale à 1 sont en bijection avec les sous-ensembles de $k-1$ coordonnées, qui sont $\binom{n}{n-k+1}$.

Soit c un mot de poids minimal dont la composante non nulle la plus à gauche est égale à 1. Soit $c' \in C$, $c' \neq c$, tel que c couvre c'. Alors c' a au moins $k-1$ zéros, de manière que soit $c' = 0$ soit c' est un mot de poids minimal. Si $c' \neq 0$ et la composante non nulle la plus à gauche de c' est égale à 1, alors c' est forcement égale à c, par le Lemme 1.1.. Donc c est minimal.

Soit c un mot minimal. Elle a $h \leq k-1$ composantes égales à 0. Par le Lemme 1.1. il existe un mot c' de poids minimal qui a composantes égales à 0 dans les h coordonnées et dans $k-1-h$ coordonnées différentes. En multipliant c' pour un scalaire, on obtient c'', avec composante non nulle la plus à gauche égale à 1. Par construction c couvre c'', et donc $c = c''$, par définition. Ainsi c a poids minimal.

Exercice 4.3. *Soit $\mathcal{C} \subseteq \mathbb{F}_3^6$ un code linéaire avec matrice génératrice*

$$G := \begin{bmatrix} 0 & 0 & 1 & 1 & 2 & 2 \\ 1 & 2 & 0 & 1 & 2 & 0 \end{bmatrix}.$$

Calculer P_I et P_S pour le système d'authentification cartésien basé sur le code $\mathcal{C}$. Est-ce que le système est optimal ?

Solution : Par le Théorème 2.1 on a

$$P_I = \frac{1}{3} \quad \text{et} \quad P_S = \max_{0 \neq c \in \mathcal{C}} \max_{u \in \mathbb{F}_3} \frac{N(c,u)}{6}.$$

où $N(c,u) := \#\{i \mid c_i = u\}$. Il suffit donc de calculer $N(c,u)$ pour tout $0 \neq c \in \mathcal{C}$ et tout $u \in \mathbb{F}_3$. La liste de $0 \neq c \in \mathcal{C}$ est

$$\{(0,0,1,1,2,2), (1,2,0,1,2,0),$$
$$(0,0,2,2,1,1), (2,1,0,2,1,0),$$
$$(1,2,1,2,1,2), (2,1,1,0,0,2),$$
$$(2,1,2,1,2,1), (1,2,2,0,0,1)\}$$

d'où on voit que $N(c,u) = 2$ ou $N(c,u) = 3$. Donc $P_S = \frac{3}{6} = \frac{1}{2} > \frac{1}{3}$, de manière que le système n'est pas optimal.

Exercice 4.4. *On choisi au hasard un polynôme $p(x)$ parmi tous les polynômes de degré 2 dans $\mathbb{F}_q[x]$. Montrer qu'il y a*

$$\frac{1}{2} \cdot (q-1)^2 \cdot q$$

polynôme irréductibles de degré 2 dans $\mathbb{F}_q[x]$, de manière que la probabilité que $p(x)$ soit irréductible est

$$\frac{1}{2} \cdot \frac{q-1}{q}.$$

Donner les paramètres possibles d'un code de Goppa binaire $\Gamma(\mathbb{F}_8, p(x))$, avec $p(x)$ irréductible de degré 2.

Solution : un polynôme $p(x)$ de degré 2 dans $\mathbb{F}_q[x]$ est de la forme ax^2+bx+c, avec $a, b, c \in \mathbb{F}_q$ et $a \neq 0$. Donc il y a $(q-1) \cdot q^2$ polynômes en total.
Soient $e_1, \ldots, e_q$ les éléments de $\mathbb{F}_q$. Un polynôme de degré 2 est réductible si et seulement s'il est de la forme

$$p_{a,i,j}(x) := a(x-e_i)(x-e_j),$$

avec $a \in \mathbb{F}_q - \{0\}$ et $i, j \in \{1, \ldots, q\}$. Clairement

$$p_{a,i,j}(x) = p_{a,j,i}(x),$$

de manière qu'on peut supposer $i \leq j$. Donc le nombre de polynômes réductible est

$$(q-1) \cdot (1+2+3+\ldots q) = (q-1) \cdot \frac{(q+1) \cdot q}{2}.$$

Finalement, le nombre des polynômes irréductible est égal au nombre total des polynômes moins le nombre des polynômes réductibles :

$$(q-1) \cdot q^2 - (q-1) \cdot \frac{(q+1) \cdot q}{2} = (q-1) \cdot \frac{2q^2 - q^2 - q}{2} = \frac{1}{2} \cdot (q-1)^2 \cdot q.$$

Ainsi la probabilité que $p(x)$ soit irréductible est

$$\frac{\frac{1}{2} \cdot (q-1)^2 \cdot q}{(q-1) \cdot q^2} = \frac{1}{2} \cdot \frac{q-1}{q}.$$

Les paramètres possibles d'un code de Goppa binaire $\Gamma(\mathbb{F}_8, p(x))$ sont : longueur égale à 8, dimension $k \geq 8 - 3 \cdot 2 = 2$ et distance minimale $d \geq 2 \cdot 2 + 1 = 5$ par le cours. De plus, en utilisant la borne de Singleton on a que les paramètres possibles sont $[8, 2, 5]$, $[8, 2, 6]$, $[8, 2, 7]$, $[8, 3, 5]$, $[8, 3, 6]$ et $[8, 4, 5]$. Puisque il n'existe pas de code MDS binaire non trivial, $[8, 2, 7]$, $[8, 3, 6]$ et $[8, 4, 5]$ ne sont pas possibles.

Bibliographie

[1] R. Anderson, C. Ding, T. Helleseth et T. Klove. *How to build robust shared control systems*. Designs, Codes and Cryptography, 15 (2), 111–124, 1998.

[2] A. Ashikhmin et A. Barg. *Minimal vectors in linear codes*. IEEE Transactions on Information Theory 44 (5), 2010–2017, 1998.

[3] E. R. Berlekamp. *Algebraic coding theory*. World Scientific Publishing Co, 2015.

[4] E. Berlekamp, R. McEliece et H. Van Tilborg. *On the inherent intractability of certain coding problems (Corresp.)*. IEEE Transactions on Information Theory 24 (3), 384–386, 1978.

[5] J.Bolkema, H.Gluesing-Luerssen, C.A.Kelley, K.E.Lauter, B.Malmskog et J.Rosenthal. *Variations of the McEliece Cryptosystem*. Algebraic Geometry for Coding Theory and Cryptography (129–150). Springer, Cham, 2017.

[6] C. Carlet et C. Ding. *Highly nonlinear mappings*. Journal of complexity 20 (2), 205–244, 2004.

[7] C. Carlet, C. Ding et H. Niederreiter. *Authentication schemes from highly nonlinear functions*. Information Theory, 2006 IEEE International Symposium on. IEEE, 2006.

[8] C. Carlet, C. Ding et J. Yuan. *Linear codes from perfect nonlinear mappings and their secret sharing schemes*. IEEE Transactions on Information Theory 51 (6), 2089–2102, 2005.

[9] N. Courtois, M. Finiasz et N. Sendrier. *How to achieve a McEliece-based digital signature scheme*. Asiacrypt. Vol. 2248. 2001.

[10] W. de Launey. *Generalized Hadamard matrices which are developed modulo a group*. Discrete Math. 104, 49–65, 1992.

[11] R.de la Cruz, A. Meyer et P. Solé. *An extension of Massey scheme for secret sharing*. Information Theory Workshop (ITW), 2010.

[12] C. Ding, T. Helleseth, T. Klove, X. Wang. *A generic construction of Cartesian authentication codes*. IEEE transactions on information theory, 53 (6), 2229–2235, 2007.

[13] Z. Heng, Y. Qin et L. Chengju. *Three classes of linear codes with two or three weights*. Discrete Mathematics 339 (11) 2832–2847, 2016.

[14] I.Márquez-Corbella et R.Pellikaan. *Error-correcting pairs for a public-key cryptosystem.* Journal of Physics : Conference Series. Vol. 855. No. 1. IOP Publishing, 2017.

[15] J. L. Massey. *Minimal codewords and secret sharing.* Proceedings of the 6th joint Swedish-Russian international workshop on information theory, 1993.

[16] F. J. MacWilliams et N. J. A. Sloane. *The theory of error-correcting codes.* Elsevier, 1977.

[17] R. J. McEliece. *A public-key cryptosystem based on algebraic coding theory.* DSN Progress Report 42–44, 114–116, 1978.

[18] R. J. McEliece et D. V. Sarwate. *On sharing secrets and Reed-Solomon codes.* Communications of the ACM 24 (9), 583-584, 1981.

[19] N. Patterson. *The algebraic decoding of Goppa codes.* IEEE Transactions on Information Theory, 21(2), 203–207, 1975.

[20] C.Peters. *Information-set decoding for linear codes over* $\mathbb{F}_q$. International Workshop on Post-Quantum Cryptography. Springer, Berlin, Heidelberg, 2010.

[21] A. Shamir. *How to share a secret.* Communications of the ACM 22 (11), 612–613, 1979.

[22] V. M. Sidelnikov et S. O. Shestakov. *On insecurity of cryptosystems based on generalized Reed-Solomon codes.* Discrete Mathematics and Applications 2 (4), 439–444, 1992.