Université de Bordeaux

Master 2 : Cryptologie et Sécurité Informatique

Projet de fin d'études

Wave - Un procédé de signature à base de codes correcteurs

Suzanne Lansade
Eva Palandjian

Encadrant:
Gilles Zemor

Février, 2020

Sommaire

Introduction

Dans le milieu des années 70, la cryptograhie à clé publique fut marquée par l'arrivée du chiffrement RSA ainsi que du protocole d'échange de clés Diffie-Hellman. De nos jours, ces deux méthodes sont les plus utilisées en pratique. Le chriffrement RSA repose sur le problème de la factorisation, tandis que le protocole DH repose sur le problème de la résolution du logarithme discret. La cryptographie à clef publique repose donc principalement sur ces deux problèmes, tous les deux des instances du problème dit du sous-groupe caché dans un groupe abélien. Il n'existe pas à ce jour d'attaque classique contre ce problème.

En revanche, il existe des attaques utilisant des algorithmes quantiques. Dans le contexte de l'éventualité de l'arrivée dans les décénies à venir de l'ordinateur quantique, le National Institute of Standard Technology (NIST) lança en 2017 un appel pour la standardisation de systèmes à clef publique, afin de trouver des alternatives à RSA et DH. Le papier sur lequel nous avons travaillé propose un système de signature à base de code correcteurs. Les codes correcteurs, que l'on sait déjà utiliser depuis les années 70 pour le chiffrement (ex: crypto-système de McEliece) présentent une alternative quantiquement sûr puisqu'ils reposent sur le problème du décodage, un problème NP-complet. En revanche, les recherches pour utiliser des codes dans des systèmes de signatures sont peu concluantes, en témoigne le tableau des soumissions au NIST pour le second tour.

	Proportion	Échange de clefs	Signature
Réseaux	46%	9	3
Codes	28%	7	0
Fonctions de hachage	4%	0	1
Multi-varié	15%	0	4
Isogénies	4%	1	0
Preuves ZK	4%	0	1

Figure 1: Comparaison des soumissions au NIST du second tour.

Alors pourquoi est-il si compliqué de faire des signatures avec des codes correcteurs ?
Premièrement, il est difficile de tomber dans l'ensemble des syndromes facilement décodables, dans le sens où il est difficile de créer une fonction de hachage qui envoie le message m dans l'ensemble des syndromes possibles. En effet, pour un décodage au sens stricte, il faudrait un syndrome s associé à un unique mot c du code, le plus proche de m.
Cela ne pose pas de problème lorsqu'on chiffre, puisque notre chiffré sera justement le syndrome. Il suffit donc de choisir un vecteur erreur en conséquance, et de le transformer en le syndrome associé. Le déchiffrement consistera alors à retrouver le vecteur erreur à partir du syndrome.
En revanche, pour la signature, il faudra trouver un moyen de transformer notre message en un syndrome qui sera dans l'ensemble des syndromes associés à un vecteur erreur de poids minimal.
Tous les systèmes de signatures basés sur des codes sont pour l'instant soit cassés, soit inutilisables dans la pratique.

La solution de Wave est d'enlever la restriction au mot le plus proche. On cherche maintenant une famille de codes permettant de trouver, pour un mot quelconque, un des mots de code à distance ω. En particulier, la distance de décodage ω est très grande, ce qui assure typiquement de l'existence d'un mot de code à cette distance. Ainsi, l'ensemble des syndromes utilisables est l'ensemble de tous les syndromes.
Nous expliquerons dans ce rapport comment fonctionne le système de signature Wave. Nous montrerons ensuite comment s'assurer qu'il ne fait pas fuiter d'information grâce à une uniformisation des sorties, puis nous démontrerons qu'il est sûr pour la sécurité EUF-CMA.

1 Le schéma de signature Wave

Nous allons détailler dans cette section le schéma de signature Wave. C'est un schéma de type hache et signe à base de codes correcteurs. Pour des raisons de clarté nous oublierons dans un premier temps la problématique du hachage. Nous la réintroduirons en fin de rapport afin de proposer une preuve formelle de la sécurité du schéma, où la fonction de hachage est alors nécessaire. Pour l'instant, nous considérerons que l'entrée de l'algorithme de signature est déjà un syndrome.

Le schéma de signature Wave s'appuie sur une famille de codes appelés des codes $(U, U+V)$-généralisés. La structure de ces codes nous permettra de proposer un algorithme de décodage $\mathcal{D}$ utilisant une trappe T et donnant un avantage par rapport à un algorithme de décodage générique. Ce système s'appuie aussi sur la notion de fonctions GPVM, que nous détaillerons.

1.1 La famille de codes (U,U+V)-généralisés

Pour définir la famille de code utilisée dans le schéma de signature Wave, on partira de deux codes de longueur $n/2$ aléatoires de dimensions respectives k_U et k_V. Nous avons la définition suivante :

Définition 1.1. (Code $(U, U+V)$) Un code $(U, U+V)$ est un code de longueur n et de dimension $k = k_U + k_V$ et tel que :

$$(U, U+V) = \{(u, u+v) \text{ tel que } u \in U \text{ et } v \in V\}$$

En revanche, pour le système de signature Wave, nous n'utiliserons pas cette définition. En effet, nous allons voir que l'on peut facilement tirer des informations sur la structure de ces codes.

Définition 1.2. Le *hull* d'un code $\mathcal{C}$ est défini comme $hull(\mathcal{C}) := \mathcal{C} \cap \mathcal{C}^{\perp}$

Proposition 1.3. *Soit $\mathcal{C}$ un code aléatoire binaire de longueur n et de dimension k, alors on s'attend à avoir $dim(hull(\mathcal{C})) \sim \mathcal{O}(1)$. De plus :*

$$\mathbb{P}\big(dim(hull(\mathcal{C})\big) \leq t) \ \geq \ 1 - \mathcal{O}(2^{-t})$$

Proposition 1.4. *Soit UV un code $(U, U+V)$-binaire de longueur n tel que $k_U > k_V$, alors nous avons avec probabilité $1 - \mathcal{O}(2^{k_U - k_V})$*

$$dim(hull(UV)) = k_U - k_V$$

Preuve. Montrons que nous avons $dim(hull(U, U+V)) = k_U - k_V$ avec probabilité $1 - \mathcal{O}(2^{k_U - k_V})$. Par définition du *hull*, nous avons :

$$\begin{aligned} hull(U, U+V) &= (U, U+V) \cap (U, U+V)^{\perp} \\ &= (U, U+V) \cap (V^{\perp} + U^{\perp}, V^{\perp}) \end{aligned}$$

Ainsi $\forall u \in U$ et $\forall v \in V$ tels que $(u, u+v) \in hull(U, U+V)$, il existe $u^\perp \in U^\perp$ et $v^\perp \in V^\perp$ tels que

$$\begin{cases} u = v^\perp + u^\perp \\ u + v = v^\perp \end{cases} \iff \begin{cases} v = u^\perp \\ u + v = v^\perp \end{cases}$$

Donc $v \in V \cap U^\perp$. De plus, nous avons

$$\dim(V) + \dim(U^\perp) = \frac{n}{2} + k_V - k_U < \frac{n}{2}$$

Alors, avec probabilité $1 - 0(2^{k_V - k_U})$ sur le choix des codes, nous aurons $V \cap U^\perp = 0$ et $v = u^\perp = 0$. Ainsi, les vecteurs de $hull(U, U+V)$ seront de la forme (x, x) où $x \in U \cap V^\perp$ avec probabilité $1 - 0(2^{k_V - k_U})$. De même on a,

$$\dim(U) + \dim(V^\perp) = \frac{n}{2} + k_U - k_V < \frac{n}{2}$$

Alors avec probabilité $1 - 0(2^{k_V - k_U})$, on aura

$$\dim(\text{hull}((U, U+V))) = \dim(U \cap V^\perp) = k_U - k_V$$

ce qui conclue la preuve. □

Ainsi nous ne pouvons pas utiliser les codes $(U, U+V)$ binaires où $\dim(U) > \dim(V)$ puisque nous pouvons facilement les distinguer de codes aléatoires. En effet, il suffit de calculer la dimension de leur *hull* puis de le comparer au résultat de la proposition 1.3. Pour résoudre ce problème, nous poserons $q = 3$ pour toute la suite du rapport, nous pouvons alors travailler avec $\dim(U) > \dim(V)$.
Dans la pratique, les sorties de l'algorithme de signature seraient permutées afin de cacher la structure du code. Ainsi, comme vu dans la preuve, le fait que des éléments de la forme (x, x) apparaissent avec bonne probabilité pose un problème puisque cela révèle des informations sur cette permutation, partie intégrante du secret.
Pour pallier à cela nous utiliserons une nouvelle famille de codes, les codes $(U, U+V)$-généralisés :

Définition 1.5. (codes $(U, U+V)$-généralisés) Soient n un entier pair et a,b,c,d quatres vecteurs de $\mathbb{F}_q^{n/2}$ tels que pour tout $i \in [\![1, n/2]\!]$:

$$a_i c_i \neq 0 \tag{1}$$

$$a_i d_i - b_i c_i \neq 0 \tag{2}$$

Soient U et V deux codes définis comme précédemment. Le code $(U, U+V)$-généralisé correspond à l'ensemble :

$$\{(a.u + b.v, c.u + d.v) \text{ tel que } u \in U \text{ et } v \in V\}$$

où $x.y$ est le produit coordonnée par coordonnée des x_i et y_i.

Les conditions sur les vecteurs a, b, c, d permettent de garantir que :

- toutes les coordonnées de $u \in U$ apparaîtront deux fois, ce qui sera nécessaire pour utiliser la structure du code dans notre algorithme de décodage (par l'équation (1)).
- la dimension du code $(U, U+V)$-généralisé sera la somme des dimensions des codes U et V (par l'équation (2)).

Sans perte de généralité, nous posons pour toute la suite les vecteurs a,b,c,d tels que $a_i d_i - b_i c_i = 1$ pour tout $i \in [\![1, n/2]\!]$ Nous pourrons toujours nous ramener au cas général.

Proposition 1.6. *Soient U, V, a, b, c et d définis comme précédemment. Soit UV le code $(U, U+V)$-généralisé associé. Alors*

$$k = \dim(UV) = k_U + k_V.$$

De plus soient:

- $G_U \in \mathbb{F}_q^{k_U \times n/2}$ *(respectivement* $G_V \in \mathbb{F}_q^{k_V \times n/2}$*)*
- $H_U \in \mathbb{F}_q^{(n/2-k_U) \times n/2}$ *(respectivement* $H_V \in \mathbb{F}_q^{(n/2-k_V) \times n/2}$*)*

les matrices génératrices et de parité des codes U et V.
Soient A, B, C, D de $\mathbb{F}_q^{n\times n}$ les matrices diagonales de diagonales respectives les vecteurs a, b, c et d.

Alors la matrice de $\mathbb{F}_q^{(k_U+k_V)\times n}$:

$$G := \left(\begin{array}{c|c} G_U A & G_U C \\ \hline G_V B & G_V D \end{array}\right)$$

et la matrice $\mathbb{F}_q^{(n-k_U-k_V)\times n}$:

$$H := \left(\begin{array}{c|c} H_U D & -H_U B \\ \hline -H_V C & H_V A \end{array}\right)$$

sont les matrices génératrice et de parité du code UV.

Preuve. Remarquons d'abord que G engendre bien le code UV. Remarquons aussi que

$$G = \left(\begin{array}{c|c} G_U A & G_U C \\ \hline G_V B & G_V D \end{array}\right) = \left(\begin{array}{c|c} G_U & 0 \\ \hline 0 & G_V \end{array}\right)\left(\begin{array}{c|c} A & C \\ \hline B & D \end{array}\right)$$

Par définition des matrices G_V et G_U, la matrice $\left(\begin{array}{c|c} G_U & 0 \\ \hline 0 & G_V \end{array}\right)$ est de rang k_U+k_V. De plus les matrices A, B, C, D étant diagonales, le déterminant de la matrice $\left(\begin{array}{c|c} A & C \\ \hline B & D \end{array}\right)$ est le produit des $(a_i d_i - b_i c_i)$ pour $i \in [\![1, n/2]\!]$, et donc non-nul par définition des vecteurs a, b, c, d. On a donc bien $k = k_U + k_V$.
On remarque aussi que $GH^T = 0$ et que H est de rang plein par le même raisonnement que précédemment, ce qui conclut la preuve. □

1.2 Le principe de signature

Notre schéma de signature utilisera donc les codes $(U, U+V)$-généralisés et la fonction syndrome comme fonction à sens unique (sous l'hypothèse de la difficulté de résoudre le problème du décodage).

Nous allons définir la notion de fonctions GPV en moyenne (GPVM). Pour cela, introduisons d'abord la notion de distance statistique.

Définition 1.7. Soient X et Y deux variables aléatoires à valeurs dans le même espace ϵ. Soient $\mathcal{D}_X$ et $\mathcal{D}_Y$ leurs distributions respectives. On définit la distance statistique entre ces deux distributions comme :

$$\rho(\mathcal{D}_X, \mathcal{D}_Y) := \frac{1}{2} \sum_{x \in \epsilon} |\mathcal{D}_X(x) - \mathcal{D}_Y(x)|.$$

Définition 1.8. (Fonctions GPVM). On appelle fonction GPV en moyenne une paire d'algorithmes (`Trapdoor`,`InvertAlg`) ainsi qu'un triplet de fonctions $(n(\lambda), k(\lambda), \omega(\lambda))$ en fonction d'un paramètre de sécurité λ, tels que :

- `Trapdoor` est un algorithme probabiliste et polynomial en 1^λ et renvoyant le couple (H, T) où $H \in \mathbb{F}_q^{(n-k)\times n}$ de rang $n-k$ et T est la trappe associée.
- `InvertAlg` est un algorithme probabiliste et polynomial prenant en entrée la trappe T et un syndrôme $s \in \mathbb{F}_q^{n-k}$, et renvoyant $e \in \mathbb{F}_q^n$ de poids ω tel que $eH^T = s$.

De plus, pour *presque toute* matrice H renvoyée par `Trapdoor`, la fonction est :

1. bien distribuée :
$$\rho(eH^T, s) \in \text{negl}(\lambda)$$
où e est pris uniformément dans l'ensemble des mots de poids ω et de longueur n et s est pris uniformément dans $\mathbb{F}_q^{n-k}$.
2. sans fuite d'information *en moyenne* :
$$\rho(\texttt{InvertAlg}(s, T), e) \in \text{negl}(\lambda)$$
où e est pris uniformément dans l'ensemble des mots de poids ω et de longueur n et s est pris uniformément dans $\mathbb{F}_q^{n-k}$.
3. À sens unique sans la trappe :
$$\mathbb{P}(\mathcal{A}(H, s) = e \mid eH^T = s) \in \text{negl}(\lambda)$$
pour tout algorithme probabiliste polynomial $\mathcal{A}$.

C'est une définition relaxée des fonctions GPV.

Nous générons notre paire de clés publique et privée comme :

$$(pk, sk) = (H, T) \leftarrow \texttt{Trapdoor}(\lambda)$$

Plus précisément, dans notre cas, nous aurons donc pour clé publique la matrice de parité du code $(U, U+V)$-généralisé, notée H, et pour clé privée, les matrices de parité des codes U et V, respectivement notées H_U et H_V.
Notre algorithme `InvertAlg` sera un algorithme utilisant la structure des codes $(U, U+V)$-généralisés afin d'inverser la fonction syndrome avec un avantage par rapport à un algorithme de décodage générique.
Le troisième point de la définition est bien vérifié, puisqu'il découle directement de la difficulté d'inverser la fonction syndrome. De plus, la fonction syndrome est en effet bien distribuée, même si nous ne le montrerons pas dans ce rapport. Enfin, le caractère sans fuite d'information de notre système sera discuté dans la deuxième partie du rapport.
Nous pouvons maintenant définir notre système de signature.

`Sign`$^{sk}(s)$:
  e $\leftarrow$ `InvertAlg`(s,T)
  `renvoie` e

`Verify`$^{pk}(s, e')$:
  `Si` $e'H^T = s$ `et` $|e'| = \omega$
    `renvoie 1`
  `renvoie 0`

1.3 Le décodage avec trappe

En partant de l'hypothèse que la matrice de parité $\mathbf{H}$ du code $(U, U+V)$-généralisé ressemble à une matrice aléatoire, la difficulté de créer une fausse signature sans connaître la trappe T est exactement celle de résoudre le problème du décodage d'un code aléatoire, que l'on sait difficile. Nous allons expliciter dans cette section l'algorithme d'inversion de la fonction syndrome connaissant la trappe, et discuter de sa difficulté en fonction du poids ω de $\mathbf{e}$.

Notons $\mathcal{S}_{\omega,n}$ l'ensemble des mots de poids ω et de longueur n. On notera dans la suite $\mathcal{S}_\omega$ s'il n'y a pas d'ambiguïté sur la longueur. On rappelle que l'algorithme `InvertAlg` cherche à inverser la fonction syndrome :

$$\begin{array}{rccc} f_{\omega,\mathbf{H}} & : & \mathcal{S}_{\omega,n} & \rightarrow & \mathbb{F}_q^{n-k} \\ & & \mathbf{e} & \mapsto & \mathbf{eH^T} \end{array}$$

On rappelle aussi que la fonction $f_{\omega,\mathbf{H}}$ avec $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}$ s'inverse génériquement si $\omega \in [\![\omega^-_{easy}, \omega^+_{easy}]\!]$, où :

$$\omega^-_{easy} := \frac{q-1}{q}(n-k) \qquad \text{et} \qquad \omega^+_{easy} := k + \frac{q-1}{q}(n-k).$$

De plus la fonction $f_{\omega,\mathbf{H}}$ admet un inverse pour toute entrée $s \in \mathbb{F}_q^{n-k}$ si $\omega \in [\![\omega^-, \omega^+]\!]$, où :

$$\omega^- := \min\left\{\omega \in [\![0,n]\!], \binom{\omega}{n}(q-1)^\omega \geq q^{n-k}\right\}$$

$$\omega^+ := \max\left\{\omega \in [\![0,n]\!], \binom{\omega}{n}(q-1)^\omega \geq q^{n-k}\right\}$$

Nous voulons donc un moyen d'inverser la fonction syndrome pour $\omega \in [\![\omega^-_{UV}, \omega^+_{UV}]\!]$ avec ω^-_{UV} et ω^+_{UV} tels que :

$$[\![\omega^-_{easy}, \omega^+_{easy}]\!] \subsetneq [\![\omega^-_{UV}, \omega^+_{UV}]\!] \subset [\![\omega^-, \omega^+]\!]$$

Afin d'expliciter le décodage, introduisons la fonction :

$$\begin{array}{rccc} \varphi_{\mathbf{a},\mathbf{b},\mathbf{c},\mathbf{d}} & : & \mathbb{F}_q^{n/2} \times \mathbb{F}_q^{n/2} & \to & \mathbb{F}_q^{n/2} \times \mathbb{F}_q^{n/2} \\ & & (\mathbf{x},\mathbf{y}) & \mapsto & (a.\mathbf{x}+b.\mathbf{y}, c.\mathbf{x}+d.\mathbf{y}) \end{array}$$

Si cette fonction respecte les conditions sur les vecteurs a, b, c, d définies dans la définition 1.5, on dit qu'elle est UV-normalisée. Dans ce cas on peut vérifier qu'elle est bijective d'inverse :

$$\begin{array}{rccc} \varphi^{-1}_{a,b,c,d} & : & \mathbb{F}_q^{n/2} \times \mathbb{F}_q^{n/2} & \to & \mathbb{F}_q^{n/2} \times \mathbb{F}_q^{n/2} \\ & & (\mathbf{x},\mathbf{y}) & \mapsto & (d.\mathbf{x}-b.\mathbf{y}, -c.\mathbf{x}+a.\mathbf{y}) \end{array}$$

Ainsi, pour chaque vecteur $\mathbf{e}$ de $\mathbb{F}_q^n$, on peut associer deux vecteurs $\mathbf{e_U}$ et $\mathbf{e_V}$ de $\mathbb{F}_q^{n/2}$ tels que

$$(\mathbf{e_U}, \mathbf{e_V}) = \varphi^{-1}_{a,b,c,d}(\mathbf{e}).$$

Proposition 1.9. *Inverser $f_{\omega,\mathbf{H}}$ pour un certain $\mathbf{s} \in F_q^{n-k}$ est équivalent à trouver $\mathbf{e} \in \mathbb{F}_q^n$ tel que:*

$$\mathbf{e}_U \mathbf{H}_U^T = \mathbf{s}^U \qquad et \qquad \mathbf{e}_V \mathbf{H}_V^T = \mathbf{s}^V$$

où $\mathbf{s} = (\mathbf{s}^U, \mathbf{s}^V)$ avec $\mathbf{s}^U \in \mathbb{F}_q^{n/2-k_U}$ et $\mathbf{s}^V \in \mathbb{F}_q^{n/2-k_V}$.

Preuve. Nous voulons montrer que le syndrome de $\mathbf{e}$ vaut $(\mathbf{e}_U\mathbf{H}_U^T, \mathbf{e}_V\mathbf{H}_V^T)$. Remarquons d'abord que nous avons

$$\begin{aligned}\mathbf{e} &= \varphi(\mathbf{e}_U, \mathbf{e}_V) \\ &= (\mathbf{a}.\mathbf{e}_U + \mathbf{b}.\mathbf{e}_V,\ \mathbf{c}.\mathbf{e}_U + \mathbf{d}.\mathbf{e}_V) \\ &= (\mathbf{e}_U A + \mathbf{e}_V B,\ \mathbf{e}_U C + \mathbf{e}_V D)\end{aligned}$$

avec A, B, C, D les matrices diagonales définie auparavant.
De plus, notons que

$$H^T := \left(\begin{array}{c|c} D^T H_U^T & -C^T H_V^T \\ \hline -B^T H_U^T & A^T H_V^T \end{array}\right)$$

Nous avons $\mathbf{e}\mathbf{H}^T = \mathbf{s} = (\mathbf{s}^U,\ \mathbf{s}^V)$. Ainsi, en calculant le syndrome de $\mathbf{e}$ à l'aide des expressions précédentes, nous obtenons :

$$\begin{cases} (\mathbf{e}_U A + \mathbf{e}_V B)D^T\mathbf{H}_U^T - (\mathbf{e}_U C + \mathbf{e}_V D)B^T\mathbf{H}_U^T = \mathbf{s}^U \\ -(\mathbf{e}_U A + \mathbf{e}_V B)C^T\mathbf{H}_V^T + (\mathbf{e}_U C + \mathbf{e}_V D)A^T\mathbf{H}_V^T = \mathbf{s}^V \end{cases}$$

$$\iff \begin{cases} \mathbf{e}_U(AD^T - CB^T)\mathbf{H}_U^T + \mathbf{e}_V(BD^T - CB^T)\mathbf{H}_U^T = \mathbf{s}^U \\ \mathbf{e}_U(CA^T - AC^T)\mathbf{H}_V^T + \mathbf{e}_V(DA^T - BC^T)\mathbf{H}_V^T = \mathbf{s}^V \end{cases}$$

Comme A, B, C et D sont des matrices diagonales, elles sont égales à leur transposées et leur produit est commutatif. Nous avons alors :

$$\begin{cases} \mathbf{e}_U(AD - BC)\mathbf{H}_U^T = \mathbf{s}^U \\ \mathbf{e}_V(AD - BC)\mathbf{H}_V^T = \mathbf{s}^V \end{cases}$$

Rappelons que $\forall i \in [\![1, n/2]\!]$ nous avons $a_i d_i - b_i c_i = 1$.
Ainsi $AD - BC = I_{n/2}$, ce qui conclut la preuve.

□

Au lieu de décoder $\mathbf{e}$ directement, on peut donc décoder $\mathbf{e}_U$ et $\mathbf{e}_V$ indépendemment. On a alors l'algorithme suivant :

$\texttt{InvertAlg}(\mathbf{s}, \mathbf{T})$:
$(\mathbf{s}_U, \mathbf{s}_V) = s$
$\mathbf{e}_U = \texttt{DecodeU}(\mathbf{s}_U)$
$\mathbf{e}_V = \texttt{DecodeV}(\mathbf{s}_V)$
$\texttt{renvoie}\ \varphi_{\mathbf{a},\mathbf{b},\mathbf{c},\mathbf{d}}(\mathbf{e_U}, \mathbf{e_V})$

Si l'on choisit un algorithme générique pour DecodeU et DecodeV, alors nous obtiendrons un vecteur $\mathbf{e}$ de poids $\omega \in [\![\omega^-_{easy}, \omega^+_{easy}]\!]$. Nous allons montrer comment utiliser les propriétés des codes $(U, U+V)$-généréralisés pour permettre un décodage hors de cet intervalle.

Remarque 1.10. Pour tout $\mathbf{e} = \varphi_{\mathbf{a},\mathbf{b},\mathbf{c},\mathbf{d}}(\mathbf{e_U}, \mathbf{e_V})$, on a pour tout $i \in [\![1, n/2]\!]$:

$$\begin{cases} a_i \mathbf{e}_U(i) + b_i \mathbf{e}_V(i) &= \mathbf{e}(i) \\ c_i \mathbf{e}_U(i) + d_i \mathbf{e}_V(i) &= \mathbf{e}(i + n/2) \end{cases}$$

Choisir la valeur de $\mathbf{e}_U$ en fonction de la valeur de $\mathbf{e}_V$ nous permettra donc d'influer sur le poids de $\mathbf{e}$. On aura alors :

$\texttt{InvertAlg}(\mathbf{s}, \mathbf{T})$:
$(\mathbf{s}_U, \mathbf{s}_V) = s$
$\mathbf{e}_V = \texttt{DecodeV}(\mathbf{s}_V)$
$\mathbf{e}_U = \texttt{DecodeU}(\mathbf{s}_U, \mathbf{e}_V)$
$\texttt{renvoie}\ \varphi_{\mathbf{a},\mathbf{b},\mathbf{c},\mathbf{d}}(\mathbf{e_U}, \mathbf{e_V})$

Proposition 1.11. *Soit $\mathbf{e}_V$ une sortie de* DecodeV*. Soit* DecodeU *un algorithme prenant en entrée $\mathbf{s}^U$ et $\mathbf{e}_V$ et renvoyant $\mathbf{e}_U$ tel que $\mathbf{e}_U \mathbf{H}_U^T = \mathbf{s}^U$ et tel que pour k_U positions de $\mathbf{e}_U$*

$$\begin{cases} a_i \mathbf{e}_U(i) + b_i \mathbf{e}_V(i) &\neq 0 \\ c_i \mathbf{e}_U(i) + d_i \mathbf{e}_V(i) &\neq 0 \end{cases}$$

Alors $\mathbf{e} = \varphi_{\mathbf{a},\mathbf{b},\mathbf{c},\mathbf{d}}(\mathbf{e_U}, \mathbf{e_V})$ a au moins $2k_U$ coordonnées non nulles. De plus les $n - k_U$ autres coordonnées sont uniformément distribuées sur $\mathbb{F}_q$.
On a alors

$$\mathbb{E}(|\mathbf{e}|) = \frac{q-1}{q} n + \frac{2k_U}{q}$$

et on peut donc espérer obtenir en temps polynomial des erreurs de poids :

$$\omega^{+}_{UV} = \begin{cases} \frac{q-1}{q}n + \frac{2k}{q} & si\ k \leq \frac{n}{2} \\ n & sinon \end{cases}$$

Proposition 1.12. *Soit* $\mathbf{e}_V$ *une sortie de* `DecodeV`. *Soit* `DecodeU` *un algorithme prenant en entrée* $\mathbf{s}^U$ *et* $\mathbf{e}_V$ *et renvoyant* $\mathbf{e}_U$ *tel que* $\mathbf{e}_U\mathbf{H}_U^T = \mathbf{s}^U$ *et tel que pour* k_U *positions de* $\mathbf{e}_U$

$$\begin{cases} a_i\mathbf{e}_U(i) + b_i\mathbf{e}_V(i) &= 0 \\ c_i\mathbf{e}_U(i) + d_i\mathbf{e}_V(i) &= 0 \end{cases} \tag{3}$$

On peut alors espérer obtenir en temps polynomial des erreurs de poids:

$$\omega^{-}_{UV} = \begin{cases} \frac{q-1}{q}(n-2k) & si\ k \leq \frac{n}{2q} \\ \\ \frac{2(q-1)^2}{(2q-1)q}(n-k) & sinon \end{cases} \tag{4}$$

Preuve. Il n'existe de solution au système (3) que si $\mathbf{e}_V(i) = 0$ car pour tout i on a $a_id_i - b_ic_i \neq 0$. De ce fait, à l'inverse du cas où nous souhaitions des erreurs de gros poids, l'ensemble d'indices où l'on peut gagner deux fois est réduit à $n/2 - |\mathbf{e}_V|$. Ainsi le poids minimal que nous pouvons espérer pour $\mathbf{e}_V$ est $|\mathbf{e}_V|_{min} := \frac{q-1}{q}(n/2 - k_V)$. Ainsi :

- Si $k_U \leq n/2 - |\mathbf{e}_V|_{min}$, nous pouvons obtenir des erreurs $\mathbf{e}$ telles que :
  - $2k_U$ coordonnées sont nulles.
  - Les autres coordonnées sont uniformément distribuées.
- Sinon, nous pouvons obtenir des erreurs $\mathbf{e}$ telles que :
  - $2(n/2 - |\mathbf{e}_V|_{min})$ sont nulles.
  - $k_U - (n/2 - |\mathbf{e}_V|_{min})$ autres coordonnées sont nulles tandis que $k_U - (n/2 - |\mathbf{e}_V|_{min})$ sont non nulles.
  - Les autres coordonnées sont uniformément distribuées.

Nous allons maintenant détailler le calcul de ω^{-}_{UV}. Rappelons que notre code a une dimension $k = k_U + k_V$ et que nous cherchons à obtenir une erreur de la forme $\mathbf{e} = (\mathbf{e}_U, \mathbf{e}_U + \mathbf{e}_V)$ avec k_U coordonnées nulles.
On commence par décoder $\mathbf{e}_V$ avec un algorithme générique. Nous pouvons

donc découper ce dernier en fonction de son poids de cette façon :

$$\mathbf{e}_V = (0, \mathbf{e}''_V) \text{ avec } 0 \in \mathbb{F}_q^{n/2-|\mathbf{e}_V|} \text{ et } \mathbf{e}''_V(i) \neq 0 \ , \forall i$$

On écrit alors $\mathbf{e}_U$ à l'aide du même découpage :

$$\mathbf{e}_U = (\mathbf{e}'_U, \mathbf{e}''_U) \text{ avec } \mathbf{e}'_U \in \mathbb{F}_q^{n/2-|\mathbf{e}_V|} \text{ et } \mathbf{e}''_U \in \mathbb{F}_q^{|\mathbf{e}_V|}$$

Nous allons dans cette démonstration, montrer la deuxième partie de l'égalité. Supposons que $k_U \geq n/2 - |\mathbf{e}_V|$, ce qui signifie que nous voulons choisir plus de coordonnées nulles que n'en dispose $\mathbf{e}'_U$. Nous allons donc, lors de notre algorithme, fixer toutes les coordonnées de $\mathbf{e}'_U$ à zéro, et il nous restera à annuler $k_U - (n/2 - |\mathbf{e}_V|)$ coordonnées de $\mathbf{e}''_U$. Nous obtenons alors

$$|\mathbf{e}_U| = \frac{q-1}{q}(n/2 - k_U)$$
$$|\mathbf{e}_U + \mathbf{e}_V| = k_U - n/2 + |\mathbf{e}_V| + \frac{q-1}{q}(n/2 - k_U)$$

En effet, comme nous annulons $k_U - n/2 + |\mathbf{e}_V|$ coordonnées de $\mathbf{e}''_U$ et que $\mathbf{e}''_V(i) \neq 0 \ \forall i$, ces mêmes coordonnées sont non nulles pour $\mathbf{e}''_U + \mathbf{e}''_V$. Il reste alors $n/2 - k_U$ coordonnées qui seront non nulles si $\mathbf{e}''_U(i) \neq -\mathbf{e}''_V(i)$.
Nous avons alors une erreur $\mathbf{e}$ de poids :

$$|\mathbf{e}| = |\mathbf{e}_U| + |\mathbf{e}_U + \mathbf{e}_V| = 2\frac{q-1}{q}(n/2 - k_U) + k_U - n/2 - |\mathbf{e}_V|$$

Ici, nous supposons que notre algorithme de décodage nous renvoie $\mathbf{e}_V$ de poids minimum, nous avons donc

$$|\mathbf{e}_V| = |\mathbf{e}_V|_{min} = \frac{q-1}{q}(n/2 - k_V) = \frac{q-1}{q}(n/2 - k + k_U)$$

Ainsi nous obtenons une erreur dont le poids est :

$$|\mathbf{e}| = 2\frac{q-1}{q}(n/2 - k_U) + k_U - n/2 - \frac{q-1}{q}(n/2 - k + k_U)$$

$$= n\left(\frac{3}{1}\frac{q-1}{q} - \frac{1}{2}\right) + k_U\left(1 - \frac{q-1}{q}\right) - k\frac{q-1}{1}$$

$$= n\frac{2q-3}{2q} + k_U\frac{1}{q} + k\frac{1-q}{1}$$

Pour minimiser le poids de $\mathbf{e}$ nous devons donc prendre k_U le plus petit possible. Comme nous avons supposé avoir $k_U \geq n/2 - |\mathbf{e}_V|$, nous prenons donc

$$k_U = \frac{n}{2} - |\mathbf{e}_V|$$

$$= \frac{n}{2} - \frac{q-1}{q}(n/2 - k + k_U)$$

$$= \frac{n}{2q} + k\frac{q-1}{q} - k_U\frac{q-1}{q}$$

Ce qui nous donne alors $k_U = \frac{n}{2(2q-1)} + k\frac{q-1}{q}$.

Par définition, nous avons $k_U \leq k$, nous pouvons se servir de cette condition pour minorer k :

$$\frac{n}{2(2q-1)} + k\frac{q-1}{q} \leq k$$

$$\frac{n}{2(2q-1)} \leq k\frac{q-1}{2q-1}$$

$$\frac{n}{2} \leq k$$

Nous nous trouvons donc bien dans le deuxième cas : lorsque $k \geq n/2$. Finalement, nous pouvons calculer le poids minimum de $\mathbf{e}$:

$$
\begin{aligned}
|\mathbf{e}| &= n\frac{2q-3}{2q} + \left(\frac{n}{2(2q-1)} + k\frac{q-1}{q}\right)\frac{1}{q} + k\frac{1-q}{1} \\
&= n\left(\frac{2q-3}{2q} + \frac{1}{2(2q-1)q}\right) + k\left(\frac{q-1}{(2q-1)q} + \frac{1-q}{1}\right) \\
&= n\,\frac{4q^2-8q+4}{2q(2q-1)} + k\,\frac{-2q^2+4q-2}{(2q-1)q} \\
&= n\,\frac{2(q-1)^2}{q(2q-1)} + k\,\frac{-2(q-1)^2}{(2q-1)q} \\
&= \frac{2(q-1)^2}{q(2q-1)}(n-k)
\end{aligned}
$$

Nous avons donc bien le résultat souhaité lorsque $k \geq \frac{n}{2q}$. La démonstration du cas où $k \leq \frac{n}{2q}$ étant équivalente, nous ne la ferons pas ici.

□

On remarque que l'on peut prouver la proposition 1.11 similairement.

Nous récapitulons dans la figure 2 la difficulté du décodage en fonction du poids ω.

La connaissance de la trappe apporte donc bien un avantage puisqu'elle permet un décodage pour des erreurs de poids ne permettant pas de décodage générique.

Remarque 1.13. En pratique, notre système de signature comporterait une fonction de hachage. Ainsi, la fonction de signature prendrait en entrée un mot m quelconque. Puis un aléa r serait pris aléatoirement dans $\{0,1\}^{\lambda_0}$. Ce serait alors $s \leftarrow \texttt{Hash}(m, r)$ qui serait signé. `Sign` renverrait ensuite le couple $(\mathbf{e}, r)$.

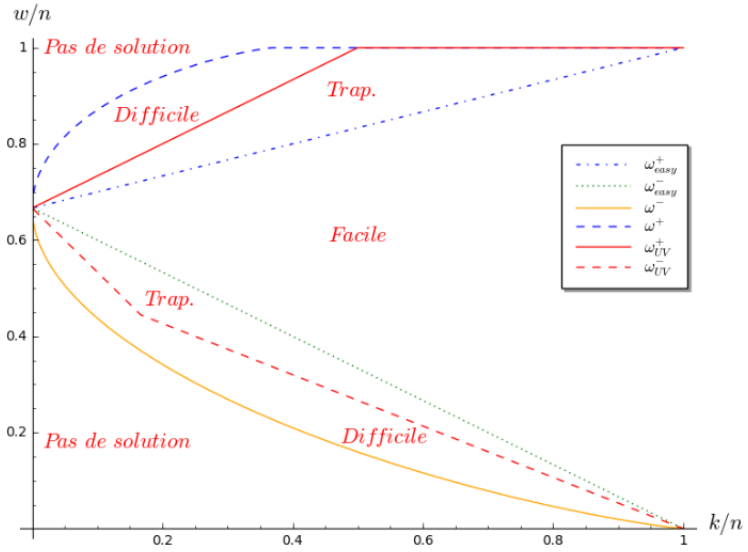

Figure 2: Comparaison des distances w/n avec et sans trappe en fonction du rendement.

La fonction `Verify` prendrait en entrée $(m, (\mathbf{e}', r))$, récupèrerait $s \leftarrow \texttt{Hash}(m, r)$ puis ferait ses vérifications. Ainsi un attaquant ne pourrait pas falsifier une signature à partir d'un vecteur $\mathbf{e}$ choisi aléatoirement dans S_ω et en récupérant $s = \mathbf{e}\mathbf{H}^T$. Sans la fonction de hachage, il aurait alors signé s.

1.4 Implémentation et choix de paramètres

Nous avons choisi d'implémenter le système de signature Wave en `C`. Nous avons également fait le choix d'implémenter nous-même toutes les fonctions nécessaires à la manipulation des matrices.
Pour implémenter la clé privée, nous avons créé une structure contenant $\mathbf{H}_U$, $\mathbf{H}_V$, S, et P. La clé publique, quant à elle, est une simple matrice puisqu'il s'agit de $S\mathbf{H}P$ avec $\mathbf{H}$ la matrice de parité du code $(U, U+V)$-généralisé.
Nous avons donc écrit une fonction qui génère une paire de clé publique et privée, une fonction qui signe un message et une fonction qui vérifie la signature. Notre fonction de signature appelle `InvertAlg` qui, pour cacher la structure de $\mathbf{H}$, appelle la fonction `DecodeUV` sur $\mathbf{s}(S^{-1})^T$ et renvoie la sortie récupérée multipliée par P. Nous obtenons alors bien une signature

valide tout en ayant masqué les données lors des calculs.
Pour obtenir λ bits de sécurité, les paramètres sont choisis de la façon suivante :

$$n = \frac{\lambda}{0.0154} \quad \omega = 0.9261n \quad k_U = 0.07978\frac{n}{2} \quad k_V = 0.4201\frac{n}{2}$$

Nous les donnons ici sans démonstration, celle-ci peut-être trouvée dans la thèse de Thomas Debris-Alazard section 5.3.5.
Lors de nos tests sur le nombre de rejets effectués, nous trouvons en effet des résultats cohérents avec ce qui précède, nous avons en moyenne 1 rejet toutes les 20 signatures.

2 Uniformisation des sorties

2.1 Une fuite d'information

Afin de vérifier le deuxième point de la définition des fonctions GPVM, il est nécéssaire que les $\mathbf{e} \in f^{-1}_{w,\mathbf{H}}(\mathbf{s})$ ne révèlent pas d'information sur la structure du code $(U, U+V)$-généralisé utilisé.
Or, si la sortie $\mathbf{e_V}$ de `DecodeV` n'est pas uniforme, alors des corrélations entre les coordonnées $\mathbf{e}_i$ et $\mathbf{e}_{i+n/2}$ du vecteur $\mathbf{e}$ apparaissent.

Par exemple, prenons le cas où $q = 3$, et où pour tout $i \in [\![1, n/2]\!]$, $a_i = c_i = d_i = 1$ et $b_i = 0$, et où `DecodeV` est l'algorithme de Prange.
On a alors pour tout $\mathbf{e} = (\mathbf{e_U}, \mathbf{e_U} + \mathbf{e_V})$

$$|\mathbf{e_V}| = \#\left\{1 \leq i \leq n/2 \mid e_i \neq e_{i+n/2}\right\}$$

Proposition 2.1. *Si le vecteur $\mathbf{e_V}$ est obtenu par l'algorithme de Prange, alors il est de poids moyen $\frac{2}{3}(\frac{n}{2} - k_V)$.*

Alors, pour tout $i \in [\![1, n/2]\!]$, on a :

$$\mathbb{P}(\mathbf{e}_i \neq \mathbf{e}_{i+n/2}) = \frac{2}{3(n/2)}(n/2 - k_V)(1 + o(1))$$

En revanche, pour les autres paires (i, j), on a :

$$\mathbb{P}(\mathbf{e}_i \neq \mathbf{e}_j) = \frac{4wn - 3w^2 - w}{n(n-1)}$$

Ces deux probabilités n'ont donc aucune raison d'être égales. On a donc une fuite d'information. En effet, dans la pratique et afin de cacher la structure, on effectue une permutation sur les coordonnées de $\mathbf{e}$ lors de la signature. Si un attaquant récupère suffisemment de signatures, il pourra donc en analysant la fréquence des $\mathbf{e}_i \neq \mathbf{e}_j$ retrouver cette permutation. Il est donc nécéssaire pour la sécurité du schéma de s'assurer de l'uniformité des sorties de l'algorithme `sign`.

2.2 La méthode du rejet

Afin de s'assurer un $\mathbf{e}$ uniforme dans son ensemble, nous allons :

- choisir $\mathbf{e}_V$ de façon à ce qu'il soit uniforme dans son ensemble
- mettre des conditions de rejet sur $\mathbf{e}_U$ en fonction du poids de $\mathbf{e}_V$ afin de supprimer le biais sur l'ensemble

$$m_1(x) := \#\left\{1 \leq i \leq n/2 \ ; \ |(x_i, x_{i+n/2})| = 1\right\}$$

Avant d'expliciter nos algorithmes, il est nécéssaire d'introduire quelques notations et définitions.

Notation 2.2. On notera :

- $\mathbf{e}^{unif}$ la variable aléatoire tirée uniformément dans l'ensemble S_w
- $\mathbf{e}_V^{unif}$ la variable aléatoire tirée uniformément dans les mots de $\mathbb{F}_q^{n/2}$
- $\mathbf{e}_U^{unif}$ la variable aléatoire tirée uniformément dans les mots de $\mathbb{F}_q^{n/2}$ conditionné au vecteur $\mathbf{e}_V^{unif}$

Définition 2.3. (uniforme en poids et m_1-uniforme)

- `DecodeV` est dit uniforme en poids si ses sorties $\mathbf{e}_V$ sont telles que $\mathbb{P}(\mathbf{e}_V)$ n'est fonction que du poids de $\mathbf{e}_V$ quand $\mathbf{s}^V$ est tiré uniformément dans son ensemble.
- `DecodeU` est dit m_1-uniforme si ses sorties $\mathbf{e}_U$ sont telles que $\mathbb{P}(\mathbf{e}_U \mid \mathbf{e}_V)$ n'est fonction que du poids de $\mathbf{e}_V$ et de $m_1(\varphi(\mathbf{e}_U, \mathbf{e}_V))$.

Lemme 2.4. *Soit* $\mathbf{e}$ *la sortie de* `InvertAlg` *avec* $\mathbf{s}^U$ *et* $\mathbf{s}^V$ *choisis uniformément dans leurs ensembles. Soit* `DecodeV` *uniforme en poids et* `DecodeU` m_1*-uniforme. Si pour tout* y *et* z

$$|\mathbf{e}_V| \sim |\mathbf{e}_V^{unif}| \quad et \quad \mathbb{P}(m_1(\mathbf{e}) = z \mid |\mathbf{e}_V| = y) = \mathbb{P}(m_1(\mathbf{e}^{unif}) = z \mid |\mathbf{e}_V^{unif}| = y)$$

Alors

$$\mathbf{e} \sim \mathbf{e}^{unif}.$$

Preuve. Nous allons montrer qu'avec les hypothèses précédentes nous avons $\forall x \in S_\omega \quad \mathbb{P}(\mathbf{e} = x) = \mathbb{P}(\mathbf{e}^{unif} = x)$.
Soit $x \in S_\omega$:

$$\begin{aligned}\mathbb{P}(\mathbf{e} = x) &= \mathbb{P}(\mathbf{e}_U = x_U, \mathbf{e}_V = x_V) \\ &= \mathbb{P}(\mathbf{e}_U = x_U | \mathbf{e}_V = x_V)\mathbb{P}(\mathbf{e}_V = x_V)\end{aligned}$$

Notre but étant de faire apparaître les expressions énoncées lors du lemme, nous allons exprimer ces deux probabilités en fonction de $|x_V| = y$ et de $m_1(x) = z$.

$$\begin{aligned}\mathbb{P}(\mathbf{e}_V = x_V) &= \mathbb{P}(\mathbf{e}_V = x_V, |\mathbf{e}_V| = y) \\ &= \mathbb{P}(\mathbf{e}_V = x_V \,|\, |\mathbf{e}_V| = y)\ \mathbb{P}(|\mathbf{e}_V| = y) \\ &= \frac{\mathbb{P}(|\mathbf{e}_V| = y)}{n(y)}\end{aligned}$$

avec $n(y) := \#\left\{ \mathbf{e} \in \mathbb{F}_3^{n/2} \,\middle|\, |\mathbf{e}| = y \right\}$. De la même façon, nous obtenons :

$$\begin{aligned}\mathbb{P}(\mathbf{e}_U = x_U |\ |\mathbf{e}_V| = y) &= \mathbb{P}(\mathbf{e}_U = x_U,\ m_1(\mathbf{e}) = z \mid\ |\mathbf{e}_V| = y) \\ &= \mathbb{P}(\mathbf{e}_U = x_U \mid\ m_1(\mathbf{e}) = z,\ |\mathbf{e}_V| = y)\mathbb{P}(m_1(\mathbf{e}) = z \mid |\mathbf{e}_V| = y) \\ &= \frac{\mathbb{P}(m_1(\mathbf{e}) = z \mid |\mathbf{e}_V| = y)}{n(y, z)}\end{aligned}$$

avec $n(y, z) := \#\left\{ \mathbf{e} \in \mathbb{F}_3^{n/2} \,\middle|\, m_1(\mathbf{e}) = z \text{ et } |\mathbf{e}_V| = y \right\}$.

Ainsi nous obtenons

$$\begin{aligned}\mathbb{P}(\mathbf{e}=x) &= \frac{\mathbb{P}(m_1(\mathbf{e})=z \mid |\mathbf{e}_V|=y)}{n(y,z)}\frac{\mathbb{P}(|\mathbf{e}_V|=y)}{n(y)} \\ &= \frac{\mathbb{P}(m_1(\mathbf{e}^{unif})=z \mid |\mathbf{e}_V^{unif}|=y)}{n(y,z)}\frac{\mathbb{P}(|\mathbf{e}_V^{unif}|=y)}{n(y)} \\ &= \mathbb{P}(\mathbf{e}_U^{unif}=x_U|\mathbf{e}_V^{unif}=x_V)\mathbb{P}(\mathbf{e}_V^{unif}=x_V) \\ &= \mathbb{P}(\mathbf{e}^{unif}=x)\end{aligned}$$

□

Ainsi, pour que $\mathbf{e}$ soit uniformément distribué sur S_ω, il suffit de choisir `DecodeV` de façon à ce que ses sorties soient uniformes sur $\mathbb{F}_q^{n/2}$ puis d'ajouter une condition de rejet sur les sorties de `DecodeU` de façon à ce que $m_1(\mathbf{e})$ conditionné à $|\mathbf{e}_V|$ soit distribué comme $m_1(\mathbf{e}^{unif})$ conditionné à $|\mathbf{e}_V^{unif}|$.
On peut alors introduire l'algorithme suivant :

Algorithme 1 DecodeUV$(\varphi, \mathbf{s}, \mathbf{H}_V, \mathbf{H}_U)$

Entrées: φ, $\mathbf{s} \in \mathbb{F}_q^{n-k}$ un syndrome, $\mathbf{H}_V \in \mathbb{F}_q^{(\frac{n}{2}-k_V)\times\frac{n}{2}}$, $\mathbf{H}_U \in \mathbb{F}_q^{(\frac{n}{2}-k_U)\times\frac{n}{2}}$
Sortie: $\mathbf{e} = \varphi(e_U, e_V)$ avec $\mathbf{e}_U\mathbf{H}_U^T = \mathbf{s}^U$ et $\mathbf{e}_V\mathbf{H}_V^T = \mathbf{s}^V$

1: $\mathbf{e}_V \leftarrow \text{DecodeV}(\mathbf{s}^V, \mathbf{H}_V)$
2: **Faire**
3: $\quad \mathbf{e}_U \leftarrow \text{DecodeU}(\varphi, \mathbf{e}_V, \mathbf{s}^U, \mathbf{H}_U)$
4: $\quad \mathbf{e} \leftarrow \varphi(\mathbf{e}_U, \mathbf{e}_V)$
5: **Tant que** $\text{rand}([0,1]) > r(m_1(\mathbf{e}), |\mathbf{e}_V|)$
6: **Retourne** $\mathbf{e}$

Avec :

$$r(s,t) := \frac{1}{M(t)} \frac{q^{unif}(s,t)}{q(s,t)}$$

$$q(s,t) := \mathbb{P}(m_1(\mathbf{e}) = s \mid |\mathbf{e}_V| = t)$$

$$q^{unif}(s,t) := \mathbb{P}(m_1(\mathbf{e}^{unif}) = s \mid |\mathbf{e}_V^{unif}| = t)$$

$$M(t) := \max_{0 \leq s \leq t} \frac{q^{unif}(s,t)}{q(s,t)}$$

Pour montrer que la sortie de notre algorithme `DecodeUV` suit bien une distribution uniforme, énonçons le lemme suivant :

Lemme 2.5. *Soient X et X^{unif} deux variables aléatoires à valeur dans un même ensemble $\mathcal{X}$ telles que :*

- *X suit une distribution quelconque,*
- *X^{unif} suit une distribution uniforme.*

Pour tout $x \in \mathcal{X}$ on pose :

$$r(x) := \frac{1}{M} \frac{\mathbb{P}(X^{unif} = x)}{\mathbb{P}(X = x)}$$

$$M := \max_{y \in \mathcal{X}} \frac{\mathbb{P}(X^{unif} = y)}{\mathbb{P}(X = y)}$$

Alors la variable aléatoire Y définie telle que:

1. *On tire $x \in \mathcal{X}$ selon la distribution X*
2. *On tire θ uniformément dans l'intervalle $[\![0, 1]\!]$,*

3. *Si* $\theta \leq r(x)$, *alors* Y *prend la valeur* x.

4. *Sinon, on recommence.*

Alors la variable Y *suit une loi uniforme.*

Preuve. Remarquons d'abord que pour tout x:

$$0 \leq r(x) = \frac{1}{M}\frac{\mathbb{P}(X^{unif} = x)}{\mathbb{P}(X = x)} = \left(\inf_{y \ in \mathcal{X}} \frac{\mathbb{P}(X = y)}{\mathbb{P}(X^{unif} = y)}\right)\frac{\mathbb{P}(X^{unif} = x)}{\mathbb{P}(X = x)} \leq 1$$

Donc les coordonnées de r sont donc bien des probabilités.

On remarque aussi que,

$$\begin{aligned}\mathbb{P}\text{ ("y soit accepté")} &:= \mathbb{P}(\ \theta \leq r(x) \quad \text{et} \quad X = x\ ) \\ &:= \mathbb{P}(\theta \leq r(x)) \times \mathbb{P}(X = x)\end{aligned}$$

De plus, la probabilité que Y soit égal à x est égale à la probabilité que x ai été tiré et qu'il ai été accepté. Pour tout $x \in \mathcal{X}$, on a alors,

$$\begin{aligned}\mathbb{P}(Y = x) &:= \frac{\mathbb{P}(\theta \leq r(x)) \times \mathbb{P}(X = x)}{\sum_{y\in\mathcal{X}} \mathbb{P}(\theta \leq r(y)) \times \mathbb{P}(X = y)} \\ &:= \frac{r(x) \times \mathbb{P}(X = x)}{\sum_{y\in\mathcal{X}} r(y) \times \mathbb{P}(X = y)} \qquad \Big(\text{car} \quad \mathbb{P}(\theta \leq r(x)) = r(x)\Big) \\ &:= \frac{\mathbb{P}(X^{unif} = x)}{M \times \sum_{y\in\mathcal{X}} 1/M \times \mathbb{P}(X^{unif} = y)} \qquad \Big(\text{car} \quad \sum_{y\in\mathcal{X}} \mathbb{P}(X^{unif} = y) = 1\Big) \\ &:= \mathbb{P}(X^{unif} = x)\end{aligned}$$

□

On peut alors énoncer la proposition suivante :

Proposition 2.6. *Si* `DecodeV` *est uniforme en poids et si* `DecodeU` *est* m_1*-uniforme, alors on a* $\mathbf{e} \sim \mathbf{e}^{unif}$.

Preuve. Soit $\mathbf{e}'_U$ le vecteur obtenu à la sortie de la boucle dans `DecodeUV`. Notons pour tout $i, j \in [\![1, n]\!]$:

$$q'(i,j) := \mathbb{P}(m_1(\mathbf{e}'_U) = i \mid |\mathbf{e}'_V| = j)$$

On a alors par le lemme 2.12

$$q'(i,j) = q^{unif}(i,j)$$

De plus la sortie de l'algorithme `DecodeV` est uniforme, ce qui conclut la preuve. □

2.3 Choix des algorithmes de décodage

Nous allons maintenant détailler les algorithmes `DecodeU` et `DecodeV` utilisés lors de `DecodeUV`.
Nous n'avons aucune contrainte sur la sortie de $\mathbf{e}_V$ si ce n'est qu'elle doit être uniforme. Nous avons donc choisi, contrairement à ce qui est spécifié dans l'article que nous avons étudié, de choisir $\mathbf{e}_V$ directement de façon à ce qu'il soit aléatoire. Celui est décrit ci-dessous :

Algorithme 2 DecodeV($\mathbf{s}^V$)

1: c mot aléatoire du code V
2: $\mathbf{s} \leftarrow$ le syndrome $\mathbf{s}^V$ paddé avec des zéros
3: $\mathbf{e}_V \leftarrow \mathbf{s} + c$
4: **Retourne** $\mathbf{e}_V$

Nous prenons donc un mot aléatoire du code que nous ajoutons au syndrome voulu (celui-ci ayant été complété avec des zéros pour être de la bonne longueur). Nous obtenons alors bien une erreur choisi uniformément dans les mots à distance $\mathbf{s}^V$ du code.

En revanche, dans l'algorithme `DecodeU`, nous avons des contraintes supplémentaires sur les coordonnées de $\mathbf{e}_U$ pour garantir un poids de $\mathbf{e}$ élevé. De plus, nous ajoutons le paramètre $d \in [0, k_U]$ dans le but de minimiser le nombre de

Algorithme 3 DecodeU$(\varphi, \mathbf{e}_V, \mathbf{s}^U, \mathbf{H}_U)$

$t \leftarrow |\mathbf{e}_V|$
$k_0 \hookleftarrow \mathcal{D}_U^t$
Faire
  $\mathcal{I} \leftarrow$ ensemble d'information de $\langle \mathbf{H}_U \rangle^\perp$
  $\mathcal{J} \subset \mathcal{I}$ de taille $k-d$ tel que $|\mathbf{e}_V|_{\mathcal{J}} = k_0$
  $x_U \hookleftarrow \{x \in \mathbb{F}_3^{n/2} | \forall j \in \mathcal{J}, x_j \notin \{-\frac{b_i}{a_i}\mathbf{e}_{V_i}, -\frac{d_i}{c_i}\mathbf{e}_{V_i}\}\}$
  $\mathbf{e}_U \leftarrow$ PRANGE $(\mathbf{H}_U, \mathbf{s}^U, \mathcal{I}, x_U)$
Tant que $|\varphi(\mathbf{e}_U, \mathbf{e}_V)| \neq \omega$
Retourne $\mathbf{e}_U$

rejets, en effet en fixant $k_U - d$ positions de $\mathbf{e}_U$, nous dépendrons moins de la structure du code et l'erreur se rapprochera donc plus de l'uniforme.
Lors de `DecodeU`, nous faisons varier les ensembles $\mathcal{I}$ et $\mathcal{J}$ pour garantir un maximum de possibilité dans le choix de $\mathbf{e}_U$.
Puis nous appelons `PRANGE` qui est ici une version modifié du décodage par ensemble d'information dans lequel nous choisissons $\mathbf{e}_U$ égal à x_U sur les positions de $\mathcal{I}$. Nous répétons tout cela jusqu'à obtenir une erreur du bon poids.

2.4 Estimation du nombre de rejet

Nous souhaitons estimer le nombre de rejets effectués dans notre algorithme `DecodeUV` utilisant les algorithmes `DecodeU`, `DecodeV` définis précédemment. Pour cela, introduisons la définition suivante:

Définition 2.7. (Bon ensemble). Soient $d \leq k \leq n$, $\mathbf{H} \in \mathbb{F}_3^{(n-k)\times n}$ et $\varepsilon \subseteq [\![1, n]\!]$ de taile $k-d$. On dit que ε est un bon ensemble pour $\mathbf{H}$ si $\mathbf{H}_{\overline{\varepsilon}}$ est de rang plein. Sinon, on dit que ε est un mauvais ensemble.

Pour estimer ce nombre de rejets moyen, nous allons comparer $\mathbf{e}$ la sortie de `DecodeUV` avec $\mathbf{e}^{unif}$ une erreur aléatoire uniforme de poids ω.
Nous savons que la sortie de `DecodeV` est uniforme, nous allons donc étudier la sortie de `DecodeU`. Nous introduisons pour cela `VarDecodeU` qui fonctionne de la même façon que `DecodeU` quand $\mathcal{J}$ est un bon ensemble pour $\mathbf{H}$ et qui renvoie une erreur aléatoire selon la distribution $\mathcal{D}_U^t$ sur $\mathcal{J}$ dans le cas contraire. Il n'y a donc aucune raison que la sortie soit une solution du problème de décodage lorsque $\mathcal{J}$ n'est pas un bon ensemble pour $\mathbf{H}$. Nous

pouvons facilement voir que `VarDecodeU` est m_1-uniforme.
La sortie de l'algorithme `DecodeUV` utilisant `VarDecodeU` est alors uniforme, on la note $\mathbf{e}^{unif}$.

Définition 2.8.

- $J^{unif}_{x_V,l}$ et un ensemble choisi uniformément tel que $J^{unif}_{x_V,l} \subseteq [\![1, n/2]\!]$, il est de cardinal $k_U - d$ et $\#J^{unif}_{x_V,l} \cap Supp(x_V) = k_0$.
- $J^{\mathbf{H}_U}_{x_V,l}$ est défini de la même façon avec une contrainte supplémentaire : il fait parti des bons ensembles pour $\mathbf{H}_U$.

Pour plus de simplicité, nous les noterons par la suite J^{unif} et $J^{\mathbf{H}_U}$.

Pour pouvoir compter le nombre de rejets, nous allons avoir besoin des lemmes suivants dont les preuves sont en annexes.

Lemme 2.9. *Nous pouvons majorer la distance statistique entre la sortie de l'algorithme utilisant* `DecodeU` *et celle de l'algorithme utilisant* `VarDecodeU` *de la façon suivante :*

$$\rho\left(\mathbf{e}, \mathbf{e}^{unif}\right) \leq \sum_{x_V,l} \rho\left(J^{\mathbf{H}_U}, J^{unif}\right) \mathbb{P}\left(k_0 = l, \mathbf{e}_V = x_V\right)$$

Lemme 2.10. *L'espérance de la différence statistique entre l'ensemble J^{unif} choisi uniformément et l'ensemble $J^{\mathbf{H}_U}$ choisi parmi les bons ensembles pour $\mathbf{H}$ peut être majorée comme suit :*

$$\mathbb{E}\left(\rho\left(J^{unif}, J^{\mathbf{H}_U}\right)\right) \leq \frac{3^{-d}}{2}$$

Lemme 2.11. *(Inégalité de Markov).*
Soit Z une variable aléatoire supposée presque sûrement positive ou nulle, alors

$$\forall a > 0 \quad \mathbb{P}(Z > a) \leq \frac{\mathbb{E}(Z)}{a}$$

Nous pouvons maintenant énoncer le théorème suivant :

Théorème 2.12. *Soit* $\mathbf{e}$ *la sortie de* `DecodeUV` *et* $\mathbf{e}^{unif}$ *la variable aléatoire uniformément distribuée parmi les mots de poids* ω*. On a :*

$$\mathbb{P}\left(\rho(\mathbf{e}, \mathbf{e}^{unif}) > 3^{-d/2}\right) \leq \frac{3^{-d/2}}{2}$$

Preuve.

$$\begin{aligned}
&\mathbb{P}\big(\rho(\mathbf{e}, \mathbf{e}^{unif}) > 3^{-d/2}\big) \\
&\quad \leq 3^{d/2}\mathbb{E}(\rho\left(\mathbf{e}, \mathbf{e}^{unif}\right)\big) \quad \text{par l'inégalité de Markov} \\
&\quad \leq 3^{d/2}\mathbb{E}\left(\sum_{x_V, l} \rho\left(J^{\mathbf{H}_U}, J^{unif}\right)\mathbb{P}\left(k_0 = l, \mathbf{e}_V = x_V\right)\right) \text{ d'après le lemme 2.9} \\
&\quad \leq 3^{d/2}\left(\sum_{x_V, l} \frac{3^{-d}}{2}\mathbb{P}\left(k_0 = l, \mathbf{e}_V = x_V\right)\right) \quad \text{par (2.10)} \\
&\quad = 3^{d/2}\frac{3^{-d}}{2}\left(\sum_{x_V, l} \mathbb{P}\left(k_0 = l, \mathbf{e}_V = x_V\right)\right) \\
&\quad = \frac{3^{-d/2}}{2}
\end{aligned}$$

□

La probabilité d'avoir un rejet équivaut à la probabilité d'avoir une distance significative entre $\mathbf{e}$ et $\mathbf{e}^{unif}$. D'après le théorème 2.12, nous voyons donc qu'en faisant augmenter d, nous serons en mesure d'effectuer très peu de rejets.

3 Sécurité du schéma

Pour montrer la sécurité du schéma, nous allons dans un premier temps montrer que si la matrice de parité du code considéré est difficile à distinguer d'une matrice aléatoire, alors le schéma est sûr au sens EUF-CMA.

Nous discuterons ensuite de la difficulté de distinguer notre matrice de parité permutée d'une matrice aléatoire.

3.1 Sécurité EUF-CMA

Nous allons montrer que le schéma est sûr au sens EUF-CMA (Existential Unforgeability under Chosen Message Attacks). Pour cela nous ferons une réduction au problème DOOM.

3.1.1 Définitions

Soit $\mathcal{A}$ un adversaire ayant accès à N_{sign} signatures de son choix. Soit les trois algorithmes suivants :

Algorithme 4 Init(λ)

1: $(pk, sk) \Longleftarrow \text{Gen}(1^\lambda)$
2: $\mathbf{H}_{pk} \Longleftarrow pk$
3: $(\varphi, \mathbf{H}_U, \mathbf{H}_V) \Longleftarrow sk$
4: **Retourne** $\mathbf{H}_{pk}$

Algorithme 5 Sign(s)

1: $\mathbf{e} \leftarrow \mathcal{D}_{\varphi,\mathbf{H}_U,\mathbf{H}_V}(\mathbf{s})$
2: **Retourne** $\mathbf{e}$

Algorithme 6 Fin((s, e))

1: **Retourne** $(\mathbf{e}\mathbf{H}_{pk}^T = s) \wedge (|\mathbf{e}| = \omega)$

Le jeu EUF-CMA se déroule comme suit. $\mathcal{A}$ fait appel à `Init`. Il peut ensuite faire N_{sign} requêtes à `sign`. Le jeu est dit réussi si $\mathcal{A}$ est capable de donner (s, e) accepté par `Fin` et tel que s n'est jamais été demandé à `Sign`.
On définit alors le succès EUF-CMA comme :

$$Succ_{Wave}^{EUF-CMA}(t, N_{sign}) := \max_{\mathcal{A};|A|\leq t} (\mathbb{P}(\mathcal{A} \text{ réussit le jeu EUF-CMA de Wave})).$$

Le protocole est alors sûr au sens EUF-CMA si ce succès est négligeable.

Nous souhaitons donc montrer que notre système est sûr au sens EUF-CMA. Pour cela, nous allons dans la section suivante majorer ce succès par rapport au succès d'un problème connu, le problème DOOM.

3.1.2 Réduction au problème DOOM

Le problème DOOM. Soient des paramètres (n, q, k, ω, N), où N est un entier.

> **I :** $\mathbf{H}$ une matrice uniforme de $\mathbb{F}_q^{(n-k)\times n}$ et $(\mathbf{s}_1, ..., \mathbf{s}_N)$ une liste de N syndromes.
> **Q :** Décoder l'un des syndromes à la distance $w := \lfloor \omega n \rfloor$.

On définit alors le succès de DOOM comme :

$$Succ^{DOOM(n,q,k,N)}(t) := \max_{\mathcal{A};|A|\leq t} (\mathbb{P}(\mathcal{A}(\mathbf{H}, \mathbf{s}_1, ..., \mathbf{s}_n) = \mathbf{e} \text{ tel que}$$

$$\mathbf{e}\mathbf{H}^T = \mathbf{s}_j \text{ pour un certain } j \in \{1, ..., N\})).$$

La réduction à ce problème est naturelle pour un schéma de signature. Il suffit de s'assurer que l'on peut considérer notre matrice de parité comme une matrice d'un code aléatoire. Ainsi, si $\mathcal{A}$ arrive à créer une signature valide sans connaître l'algorithme de décodage $\mathcal{D}_{\varphi,\mathbf{H}_U,\mathbf{H}_V}$, il a bien résolu le problème du décodage. Puisqu'on a montré préalablement que les sorties $\mathbf{e}$ de `sign` sont uniformément distribuées sur S_ω, la connaissance des N_{sign} couples $(\mathbf{s}, \mathbf{e})$ ne donnent pas d'avantage à $\mathcal{A}$. Ainsi la réduction se fait naturellement.

3.1.3 Preuve formelle de la réduction

Afin de faire une preuve formelle de la sécurité EUF-CMA, nous allons changer le jeu en rajoutant une fonction de hachage. L'attaquant $\mathcal{A}$ peut maintenant faire N_{hash} appelle à la fonction de hachage et ainsi obtenir des couples (m,s). Il cherchera alors à créer une signature valide pour l'un d'entre eux. On retrouve bien la question du problème DOOM.
De plus, la fonction de signature prends maintenant en entrée un message quelconque. Elle prend ensuite un aléa r dans $\{0,1\}^{\lambda_0}$. Le tout est alors

donné à la fonction de hachage qui renvoit un syndrome valide.

Nous allons aussi introduire un système de jeux qui nous permettra de réduire la sécurité d'un système à un problème P. Soit $\mathcal{A}$ un attaquant et $\mathcal{R}$ un rival. Soient $G_0, G_1, ..., G_N$ un ensemble de jeux et soit $\mathbb{P}(G_i)$ la probabilité pour $\mathcal{A}$ de répondre au défi posé par $\mathcal{R}$ pour le jeu G_i. $\mathbb{P}(G_0)$ est alors la probabilité de cassé le système considérer et $\mathbb{P}(G_N)$ la probabilité de répondre au problème P.
L'idée est de changer pas à pas les jeux G_0 à G_N de façon à ce que :

$$\forall i \in 0, ..., N-1, |\mathbb{P}(G_i) - \mathbb{P}(G_i + 1)| \in negl(\lambda) \Longrightarrow |\mathbb{P}(G_0) - \mathbb{P}(G_N)| \in negl(\lambda)$$

où λ est un paramètre de sécurité. Autrement dis, les changements sur les jeux ne changent qu'à un facteur négligeable près les probabilités de succès de l'attaquant $\mathcal{A}$.
Il n'est pas possible de changer le comportement de $\mathcal{A}$ puisqu'il est quelconque, en revanche nous pouvons modifier celui de R.

Théorème 3.1. *(Réduction de sécurité).*
Soit N_{sign} le nombre de requêtes faites à l'oracle de signature. Soit λ le paramètre de sécurité et $\lambda_0 = \lambda + 2\log_2(N_{sign})$. On a :

$$Succ_{Wave}^{EUF-CMA}(t, N_{sign}) \leq 2Succ^{DOOM(n,q,k,N)}(t) + \rho(\mathcal{D}_{rand}, \mathcal{D}_{pub})(t) +$$

$$f(\mathcal{U}_\omega, \mathcal{D}_\omega^{\mathbf{H}_{pk}}) + g(\epsilon) + c + \frac{N_{hash}}{2}\sqrt{\epsilon} + \frac{1}{2^\lambda}$$

où ϵ est une fonction en n qui décroît exponentiellement, $\mathcal{D}_{rand}$ est la distribution des matrices prises aléatoirement dans $\mathbb{F}_q(n-k) \times n$, $\mathcal{D}_{pub}$ celle des matrices prises aléatoirement dans l'ensemble des matrices de parité d'un code $(U, U+V)$-généralisé (et U, V de dimension respectives k_U et k_V), f l'espérance de la distance calculatoire entre les deux distributions, g linéaire en ϵ et c une constante.

Preuve. (On trouvera le détail des preuves de probabilité en annexe de ce rapport.) On rappelle que G_0 correspond à notre jeu pour la sécurité EUF-CMA de Wave.

- G_1 : Le jeu G_1 est identique au jeu G_0 sauf si l'évènement

$$F := \{\text{Un même aléa r a été tiré lors de deux requêtes d'un même message à l'oracle de signature}\}.$$

On a alors

$$\mathbb{P}(G_0) \leq \mathbb{P}(G_1) + \mathbb{P}(F)$$

Or pour $\lambda_0 = \lambda + 2\log_2(N_{sign})$, la probabilité que l'évènement F se produise est majorée par $\frac{1}{2^\lambda}$. C'est donc négligeable et le changement est autorisé.

- G_2 : Le passage au jeu G_2 permet d'empêcher $\mathcal{A}$ de faire appel à l'oracle de signature sur les syndrome du problème DOOM. L'idée est de créer une liste suffisemment grande L_m d'aléas tous différents. On modifie alors la fonction `hash` de cette façon :

  1. Si `hash` est appelée par la fonction `sign`, alors les aléas seront pris successivement dans L_m et associés à un vecteur erreur $\mathbf{e}_{m,r}$ (stocké) pris uniformément dans S_ω. Elle renvoie alors $\mathbf{s} = \mathbf{e}_{m,r}\mathbf{H}^T$.
  2. En revanche si `hash` est appelée hors de la fonction `sign` par $\mathcal{A}$, alors elle son comportement dépendra de l'aléa. Si r est dans L_m elle se comporte comme si elle avait été appelée par `sign` et renvoie $\mathbf{e}_{m,r}\mathbf{H}^T$. Sinon elle renvoie successivement les syndromes du problème DOOM.

  On prend donc dans la fonction `sign` toujours le r suivant de L_m. On a alors changé le jeu en supprimant le cas où deux mêmes r sont tirés lors de la signature. Cela ne pose pas de problème grace au passage à G_1. Le passage au jeu G_2 permettra ainsi de s'assurer par la suite que $\mathcal{A}$ n'a pas fait d'appel à `sign` sur les syndromes du problème DOOM. On a alors

  $$\mathbb{P}(G_1) \leq \mathbb{P}(G_2) + \frac{N_{hash}}{2}\sqrt{\epsilon}$$

  où ϵ est une fonction en n qui décroît exponentiellement. C'est donc bien négligeable.

- G_3 : Le jeu G_3 permet à l'oracle de signature de se passer de l'algorithme de décodage, et donc de la trappe T. Il sera nécessaire pour remplacer la matrice du code (U,U+V)-généralisé par la matrice aléatoire de l'instance du problème DOOM. Pour passer au jeu G_3, on modifie la

sortie de `sign`. Au lieu de renvoyer le couple $(\mathbf{e}, r)$ où $\mathbf{e} = D_{\varphi,H_U,H_V}$, on renvoit le couple $(\mathbf{e}_{m,r}, r)$ préalablement stocké.
La différence de succès de dépand que de ω et des différence de distribution entre $\mathcal{U}_\omega$ et $\mathcal{D}_\omega^{\mathbf{H}_{pk}}$, où $\mathcal{U}_\omega$ et la distribution uniforme sur S_ω et où $\mathcal{U}_\omega$ et $\mathcal{D}_\omega^{\mathbf{H}_{pk}}$ est la distribution des couples (e, r) où r est un aléa uniforme dans $\{0,1\}^{\lambda_0}$ et e est la sortie de l'algorothme de décodage avec trappe sur une entrée s prise uniformément dans $\mathbb{F}_q^{n-k}$. On a alors

$$\mathbb{P}(G_2) \leq \mathbb{P}(G_3) + f(\mathcal{U}_\omega, \mathcal{D}_\omega^{\mathbf{H}_{pk}}) + g(\epsilon) + c$$

où f et g sont linéaires et c un certaine constante.

- G_4 : On peut maintenant remplacer $\mathbf{H}_{pk}$ par $\mathbf{H}_0$. Ce changement ne pose pas de problème puisque `sign` n'utilise plus la trappe. En revanche, nous avons créé un distingueur entre la distribution $(:= \mathcal{D}_{rand})$ des matrices prises aléatoirement dans $\mathbb{F}_q(n-k) \times n$ et la distribuction $(:= \mathcal{D}_{pub})$ des matrices prises aléatoirement dans l'ensemble des matrices de parité d'un code $(U, U+V)$-généralisé où U (resp. V) est un $[n/2, k_U]$-code (resp. $[n/2, k_V]$-code). On a alors

$$\mathbb{P}(G_3) \leq \mathbb{P}(G_4) + \rho(\mathcal{D}_{rand}, \mathcal{D}_{pub})(t)$$

- G_5 : On change ici la procédure de fin. On rajoute à la vérification la condition $r \notin L_m$. Ainsi on est bien sûr que $\mathcal{A}$ réussi le jeu s'il répond au problème DOOM. Alors la probabilité que $\mathcal{A}$ réussisse G_5 est exactement la probabilité que $\mathcal{A}$ réussisse G_4 et $r \notin L_m$. On a alors

$$\mathbb{P}(G_4) \leq 2\mathbb{P}(G_5) + \rho(\mathcal{D}_{rand}, \mathcal{D}_{pub})(t)$$

où $\mathbb{P}(G_5)$ est exactement la probabilité pour $\mathcal{A}$ de renvoyer $\mathbf{e}_j \in S_\omega$ et tel que $\mathbf{e}_j \mathbf{H}_0^T = \mathbf{s}_j$ pour un certain indice j du problème DOOM. On a donc

$$\mathbb{P}(G_5) \leq Succ_{DOOM}^{n,k,N_{hash},\omega}(t)$$

En rassemblant toutes les inégalités on termine la preuve. □

3.2 Indistinguabilité des codes (U,U+V)-généralisés

Le schéma Wave est, d'après ce qui précède, sûr à condition que l'on ne puisse pas distinguer une matrice de parité d'un code (U,U+V)-généralisé

d'une matrice de parité aléatoire.

Nous avons, tout au long de ce rapport, simplifié les notations et les algorithmes. Afin de discuter ce problème de distinction, il est nécéssaire d'en réintroduire certaines.
La trappe du schéma Wave étant la structure du code $(U, U+V)$-généralisé utilisé, nous devons nous assurer qu'on ne peut pas la retrouver à partir de la clé secrète. Nous introduisons donc une matrice $S \in \mathbb{F}_q^{(n-k)\times(n-k)}$ inversible et une matrice de permutation $P \in \mathbb{F}_q^{n\times n}$, la clé publique devient alors $pk = SHP$. Les matrices S et P faisant donc partie de la clé privée. Ainsi, on peut espérer que la matrice de parité de laisse plus apparaître cette structure.

Lorsque $\dim(U) < \dim(V)$, il est possible de montrer que ce problème est NP-complet à l'aide d'une réduction au problème du mariage tri-dimensionnel lui-même NP-complet.
Malheureusement notre algorithme de décodage ne fonctionne pas dans le cas où $\dim(U) < \dim(V)$. En effet, afin d'obtenir des erreurs de grands poids et d'assurer qu'il sera difficile de décoder sans trappe, on a fixé $2k_U$ coordonnées de e. Si $\dim(U) < \dim(V)$, alors le fait que e soit de poids supérieur ou égal à $2k_U$ ne suffit plus à sortir de l'intervalle $[\![\omega^-_{easy}, \omega^+_{easy}]\!]$.

Nous devons donc nécéssairement nous placer dans où $\dim(U) > \dim(V)$. Or nous avons montré précédemment qu'il est alors facile de distinguer un code binaire $(U, U+V)$-généralisé d'un code aléatoire. On fixe alors $q > 2$. La démonstration du caractère NP-complet de Wave lorsque $q = 3$ et $\dim(U) > \dim(V)$ reste non achevée pour l'instant. Les auteurs du papier pensent néanmoins que la généralisation des codes $(U, U+V)$ comme nous l'avons présentée dans la définition 1.5 et notamment le grand choix des vecteurs a, b, c, et d offrent la possibilité d'étendre la réduction au problème du mariage tri-dimensionnel.

Conclusion

Le schéma de signature Wave présente une solution originale au problème de signature à base de codes correcteurs. Il est le premier, et actuellement seul, cryptosystème utilisant des codes avec un décodage en grande distance. La méthode du rejet, ajoutée afin de garantir des signatures uniformément distribuées, empêche bien de tirer la moindre information sur la structure du code à partir de la signature e. De plus, c'est une solution très efficace, puisque le nombre de rejet effectués est surprenemment faible.
Dans notre implémentation de la signature, nous avons effectivement constaté un très petit nombre de rejets.
En revanche, il y a encore peu de certitudes sur l'indistinguabilité des matrices de parités, ce qui remet en question les preuves de sécurité. On peut cependant espérer que la grande liberté sur le choix des vecteurs a,b,c,d laisse espérer que réduction de sécurité dans les cas qui nous intéressent soit possible.

Quelques preuves supplémentaires

Démonstration du lemme 2.9

Nous allons avoir besoin dans la suite de la proposition suivante que nous ne démontrerons pas dans ce rapport :

Proposition .2. *Soient X et Y deux variables aléatoires dans le même espace A et Z une variable aléatoire dans l'espace B indépendante de X et Y. Alors, pour toute fonction f, nous avons :*

$$\rho(f(X,Z), f(Y,Z)) \leq \rho(X,Y)$$

Preuve du lemme 2.9. Rappelons que nous voulons montrer l'inégalité :

$$\rho\left(\mathbf{e}, \mathbf{e}^{unif}\right) \leq \sum_{x_V, l} \rho\left(J^{\mathbf{H}_U}, J^{unif}\right) \mathbb{P}\left(k_0 = l, \mathbf{e}_V = x_V\right)$$

En écrivant $\mathbf{e} = (\mathbf{e}_U,\ \mathbf{e}_V)$ et $\mathbf{e}^{unif} = (\mathbf{e}_U^{unif},\ \mathbf{e}_V^{unif})$, nous pouvons majorer leur distance statistique

$$\begin{aligned}
\rho(\mathbf{e}, \mathbf{e}^{unif}) &= \rho\Big((\mathbf{e}_U, \mathbf{e}_V), (\mathbf{e}_U^{unif}, \mathbf{e}_V^{unif})\Big) \\
&\leq \sum_{x_U, x_V} \Big|\mathbb{P}\Big((\mathbf{e}_U, \mathbf{e}_V) = (x_U, x_V)\Big) - \mathbb{P}\Big((\mathbf{e}_U^{unif}, \mathbf{e}_V^{unif}) = (x_U, x_V)\Big)\Big| \\
&= \sum_{x_U, x_V} \Big|\mathbb{P}(\mathbf{e}_V = x_V)\mathbb{P}(\mathbf{e}_U = x_U | \mathbf{e}_V = x_V) - \mathbb{P}(\mathbf{e}_V^{unif} = x_V)\mathbb{P}(\mathbf{e}_U^{unif} = x_U | \mathbf{e}_V^{unif} = x_V)\Big| \\
&= \sum_{x_U, x_V} \mathbb{P}(\mathbf{e}_V = x_V)\Big|\mathbb{P}(\mathbf{e}_U = x_U | \mathbf{e}_V = x_V) - \mathbb{P}(\mathbf{e}_U^{unif} = x_U | \mathbf{e}_V^{unif} = x_V)\Big|
\end{aligned}$$

Le passage de la troisième ligne à la quatrième se fait car $\mathbb{P}(\mathbf{e}_V = x_V) = \mathbb{P}(\mathbf{e}_V^{unif} = x_V)$.
Rappelons que k_0 est le poids de $\mathbf{e}_V$ sur J, en ajoutant une condition sur la valeur de k_0 aux probabilités, nous obtenons donc la somme suivante :

$$\mathbb{P}(\mathbf{e}_U = x_U | \mathbf{e}_V = x_V) - \mathbb{P}(\mathbf{e}_U^{unif} = x_U | \mathbf{e}_V^{unif} = x_V) =$$

$$\sum_l \Big[\mathbb{P}(\mathbf{e}_U = x_U | k_0 = l, \mathbf{e}_V = x_V) - \mathbb{P}(\mathbf{e}_U^{unif} = x_U | k_0 = l, \mathbf{e}_V^{unif} = x_V)\Big] \mathbb{P}\left(k_0 = l | \mathbf{e}_V = x_V\right)$$

Nous obtenons donc, comme majoration de la distance statistique, une somme sur les indices x_U, x_V et l. Nous allons maintenant étudier la somme sur les indices x_U.
Les ensembles $\mathcal{J}$ étant inclus dans un ensemble d'information, l'aléa interne

de `DecodeU` et de `VarDecodeU` ne dépendent pas du choix de J^{unif} et $J^{\mathbf{H}_U}$. Nous pouvons voir que $\mathbb{P}(\mathbf{e}_U = x_U | k_0 = l, \mathbf{e}_V = x_V)$ ne dépend que de x_U et $J^{\mathbf{H}_U}$ et $\mathbb{P}(\mathbf{e}_U^{unif} = x_U | k_0 = l, \mathbf{e}_V^{unif} = x_V)$ ne dépend que de x_U et J^{unif}. De plus, J^{unif} et $J^{\mathbf{H}_U}$ vivent dans le même espace. Ainsi d'après la proposition .2, nous avons

$$\sum_{x_U} \mathbb{P}(\mathbf{e}_U = x_U | k_0 = l, \mathbf{e}_V = x_V) - \mathbb{P}(\mathbf{e}_U^{unif} = x_U | k_0 = l, \mathbf{e}_V^{unif} = x_V) \quad \leq \quad \rho(J^{\mathbf{H}_U}, J^{unif})$$

En combinant les différentes équations obtenues, nous avons alors :

$$\begin{aligned}
\rho\left(\mathbf{e}, \mathbf{e}^{unif}\right) &\leq \sum_{x_V, l} \rho\left(J^{\mathbf{H}_U}, J^{unif}\right) \mathbb{P}(\mathbf{e}_V = x_V) \mathbb{P}\left(k_0 = l | \mathbf{e}_V = x_V\right) \\
&= \sum_{x_V, l} \rho\left(J^{\mathbf{H}_U}, J^{unif}\right) \mathbb{P}\left(k_0 = l, \mathbf{e}_V = x_V\right)
\end{aligned}$$

□

Démonstration du lemme 2.10

Lemme .3. *Soient d et m deux entiers tels que $d < m$ et $M \hookleftarrow \mathbb{F}_3^{(m-d)\times m}$ alors $\mathbb{P}(rg(M) < m - d) \leq \frac{1}{2.3^d}$*

Preuve. Soit P la probabilité que M ne soit pas de rang plein, c'est-à-dire de rang inférieur à $m-d$. Notons V_i le sous-espace vectoriel engendré par les i premières lignes de M. Si M n'est pas de rang plein c'est qu'au moins une de ses lignes est une combinaison linéaire des précédentes, c'est-à-dire qu'il existe un i tel que $\dim(V_i) = \dim(V_{i-1}) = i - 1$. On a alors

$$\begin{aligned}
P &\leq \sum_{i=1}^{m-d} \mathbb{P}(\dim(V_i) = \dim(V_{i-1}) = i - 1) \\
&= \sum_{i=1}^{m-d} \mathbb{P}(\dim(V_i) = i - 1 | \dim(V_{i-1}) = i - 1) \mathbb{P}(\dim(V_{i-1}) = i - 1) \\
&\leq \sum_{i=1}^{m-d} \mathbb{P}(\dim(V_i) = i - 1 | \dim(V_{i-1}) = i - 1)
\end{aligned}$$

Si $\dim(V_{i-1}) = i - 1$, alors $\dim(V_i) = i - 1$ est équivalent à dire que la ligne i de la matrice est une combinaison linéaire des $i - 1$ précédentes. Nous avons

donc :

$$\mathbb{P}(\dim(V_i) = i-1 | \dim(V_{i-1}) = i-1) = \frac{3^{i-1}}{3^m} = \frac{1}{3^{m-i+1}}$$

Nous pouvons donc conclure avec :

$$\begin{aligned} P &\leq \sum_{i=1}^{m-d} \frac{1}{3^{m-i+1}} \\ &= \frac{1}{2.3^{-d}} - \frac{1}{2;3^m} \\ &\leq \frac{1}{2.3^{-d}} \end{aligned}$$

□

Preuve du lemme 2.10. Nous voulons ici montrer que

$$\mathbb{E}\left(\rho\left(J_{x_V,l}^{unif}, J_{x_V,l}^{\mathbf{H}_U}\right)\right) \leq \frac{3^{-d}}{2}$$

Notons $n_{U,d}$ le nombre de sous-ensembles de $[\![1, n/2]\!]$ de taille $k_U - d$ ie $n_{U,d} = \binom{n/2}{k_U - d}$. On a alors

$$\rho(J^{unif},\ J^{\mathbf{H}_U}) = \frac{N}{n_{U,d}}$$

avec $N = \#\{J \subset [\![1,\ n/2]\!] \text{ de taille } k_U - d \text{ mauvais pour } \mathbf{H}_U\}$.
Notons X_i l'événement "le sous-ensemble ε_i est un mauvais ensemble pour $\mathbf{H}_U$", c'est-à-dire la matrice $M_{\overline{\varepsilon_i}}$ n'est pas de rang plein. D'après le lemme .3, nous avons $\mathbb{P}(X_i = 1) \leq \frac{1}{2.3^d}$ De plus, nous pouvons maintenant écrire $N = \sum_{i=1}^{n_{U,d}} X_i$.

Ainsi nous avons :

$$\begin{aligned}
\mathbb{E}\left(\rho\left(J^{unif}_{x_V,l}, J^{\mathbf{H}_U}_{x_V,l}\right)\right) &= \mathbb{E}\left(\frac{N}{n_{U,d}}\right) \\
&= \mathbb{E}\left(\frac{\sum_{i=1}^{n_{U,d}} X_i}{n_{U,d}}\right) \\
&= \sum_{i=1}^{n_{U,d}} \mathbb{E}\left(\frac{X_i}{n_{U,d}}\right) \\
&= \sum_{i=1}^{n_{U,d}} \frac{\mathbb{P}(X_i = 1)}{n_{U,d}} \\
&\leq \frac{1}{2.3^d}
\end{aligned}$$

□

Démonstration du théorème 3.1

Preuve du théorème 3.1.

- **Passage de G_0 à G_1 :** Montrons que

$$\mathbb{P}(F) \leq \frac{1}{2^\lambda}$$

. Remarquons d'abord que la probabilité de n'avoir aucune collision sur t tirages indépendants et uniforme dans un ensemble de taille n est majoré par t^2/n (admis). Dans notre cas, $t = N_{sign}$ et $n = 2^{\lambda_0}$. Ainsi avec $\lambda_0 = \lambda + 2\log_2 N_{sign}$, on en déduit que

$$\mathbb{P}(F) \leq \frac{N^2_{sign}}{2^{\lambda_0}} = \frac{1}{2^{\lambda_0 - 2\log_2 N_{sign}}} = \frac{1}{2^\lambda}$$

ce qui conclu la preuve.

- **Passage de G_1 à G_2 :** Montrons que

$$\mathbb{P}(G_1) \leq \mathbb{P}(G_2) + \frac{N_{hash}}{2}\sqrt{\epsilon}$$

où ϵ est une fonction en n qui décroît exponentiellement. Nous utiliserons la proposition suivante (admise) :

Proposition .4. *Soient les variables aléatoires discrètes X_i et Y_i où $i \in \{1,2\}$ de même domaine $\mathcal{A}_i$. Notons pour $a_i \in \mathcal{A}_i$:*

- *$p(.|a_i)$ la distribution de X_2 conditionnée à l'évènement $X_1 = a_i$*
- *$q(.|a_i)$ la distribution de Y_2 conditionnée à l'évènement $Y_1 = a_i$*

Alors nous avons,

$$\rho(X_1, X_2, Y_1, Y_2) \leq \sup_{a_i \in \mathcal{A}_i} \rho(p(.|a_i), q(.|a_i)) + \rho(X_1, Y_1).$$

Les distributions des jeux G_1 et G_2 diffèrent par les sorties de la fonction de hachage. Dans le jeu G_1 les sorties X_i sont uniformément distribuées dans $\mathbb{F}_q^{n-k}$. Dans le jeu G_2, les sorties Y_i valent $\mathbf{eH}^T$ avec e uniformément distribué dans S_ω si $r \in L_m$, sinon Y_i est uniformément distribuée. On a,

$$\begin{aligned}\mathbb{P}(G_1) - \mathbb{P}(G_2) &= \sum_{\mathbf{H}} \mathbb{P}(\mathbf{H}_{pk} = \mathbf{H})(\mathbb{P}(G_1|\mathbf{H}_{pk} = \mathbf{H}) - \mathbb{P}(G_1|\mathbf{H}_{pk} = \mathbf{H}) \\ &\leq \mathbb{E}_{\mathbf{H}_{pk}}(\rho(X_1, ..., X_{N_{hash}}, Y_1, ..., Y_{N_{hash}}))\end{aligned} \tag{5}$$

Proposition .5. *Soit X la distribution uniforme dans $\mathbb{F}_q^{n-k}$ et Y la distribution de $\mathbf{eH}^T$ avec e uniformément distribué dans S_ω. Alors*

$$\mathbb{E}_{H_{pk}}(\rho(X, Y)) \leq \frac{1}{2}\sqrt{\epsilon}$$

où ϵ est une borne qui décroit exponentiellement avec n.

Par les deux propositions précédentes et par un raisonnement par récurrence, on termine la preuve.

- **Passage de G_2 à G_3 :** Nous avons

$$\mathbb{P}(G_2) \leq \mathbb{P}(G_3) + f(\mathcal{U}_\omega, \mathcal{D}_\omega^{\mathbf{H}_{pk}}) + g(\epsilon) + c$$

où f et g sont linéaires et c un certaine constante. La fonction en ϵ découle d'un raisonnement similaire au raisonnement précédent. Ensuite, à constante près, les différences de distributions entre les jeux G_2 et G_3 ne dépendent que de l'espérance de la distance statistique entre les

distributions $\mathcal{U}_\omega$ et $\mathcal{D}_\omega^{\mathbf{H}_{pk}}$, où $\mathcal{U}_\omega$ et la distribution uniforme sur S_ω et où $\mathcal{U}_\omega$ et $\mathcal{D}_\omega^{\mathbf{H}_{pk}}$ est la distribution des couples (e, r) où r est un aléa uniforme dans $\{0,1\}^{\lambda_0}$ et e est la sortie de l'algorothme de décodage avec trappe sur une entrée s prise uniformément dans $\mathbb{F}_q^{n-k}$.

- **Passage de G_3 à G_4 :** La différence entre ces deux jeux ne vient que de la distance entre la distributions des matrices de parité de nos codes $(U, U+V)$-généralisés et celle des matrices prises uniformément dans $\mathbb{F}_q^{n-k}$.

- **Passage de G_4 à G_5 :** Le jeu G_5 diffère du jeu G_4 uniquement par la vérification finale. Soit le couple (e, r) la réponse de $\mathcal{A}$ comme signature d'un message m. Sa probabilité de succès pour le jeu G_5 est exatement la probabilité de l'évènement "réussit G_4 et $r \notin L_m$".
  Si la signature est valise, alors m n'a jamais été demandé à l'oracle de signature, donc $\mathcal{A}$ n'a jamais eu accès aux éléments de L_m. Donc les évenements sont indépendants et nous avons,

$$\mathbb{P}(G_5) = (1 - 2^{-\lambda_0})^{N_{sign}} \mathbb{P}(G_4).$$

  où $\lambda_0 \leq \log_2({N^2}_{sign})$. Donc

$$(1 - 2^{-\lambda_0})^{N_{sign}} \geq (1 - \frac{1}{{N^2}_{sign}})^{N_{sign}} \geq \frac{1}{2}$$

  D'où $\mathbb{P}(G_5) \geq \frac{1}{2}\mathbb{P}(G_4)$.

□