Nouvelles constructions utilisant des codes correcteurs d'erreurs en cryptographie à clef publique

thèse soutenue par

Matthieu Finiasz

le

1er octobre 2004

Plan

Partie I

Introduction à la cryptographie

Histoire de la cryptographie

Des algorithmes secrets aux clefs publiques

- Algorithmes secrets :
  - difficiles à changer en cas de fuite.
- Algorithmes publiques/clef secrètes :
  - changement de clef facile,
  - reste le problème du partage de clef.
- [Diffie, Hellman 1976] cryptosystème à clef publique :
  - la clef de chiffrement est publique,
  - la clef de déchiffrement est secrète,
  - il n'y a plus besoin d'échanges sécurisé.

La cryptographie à clef publique

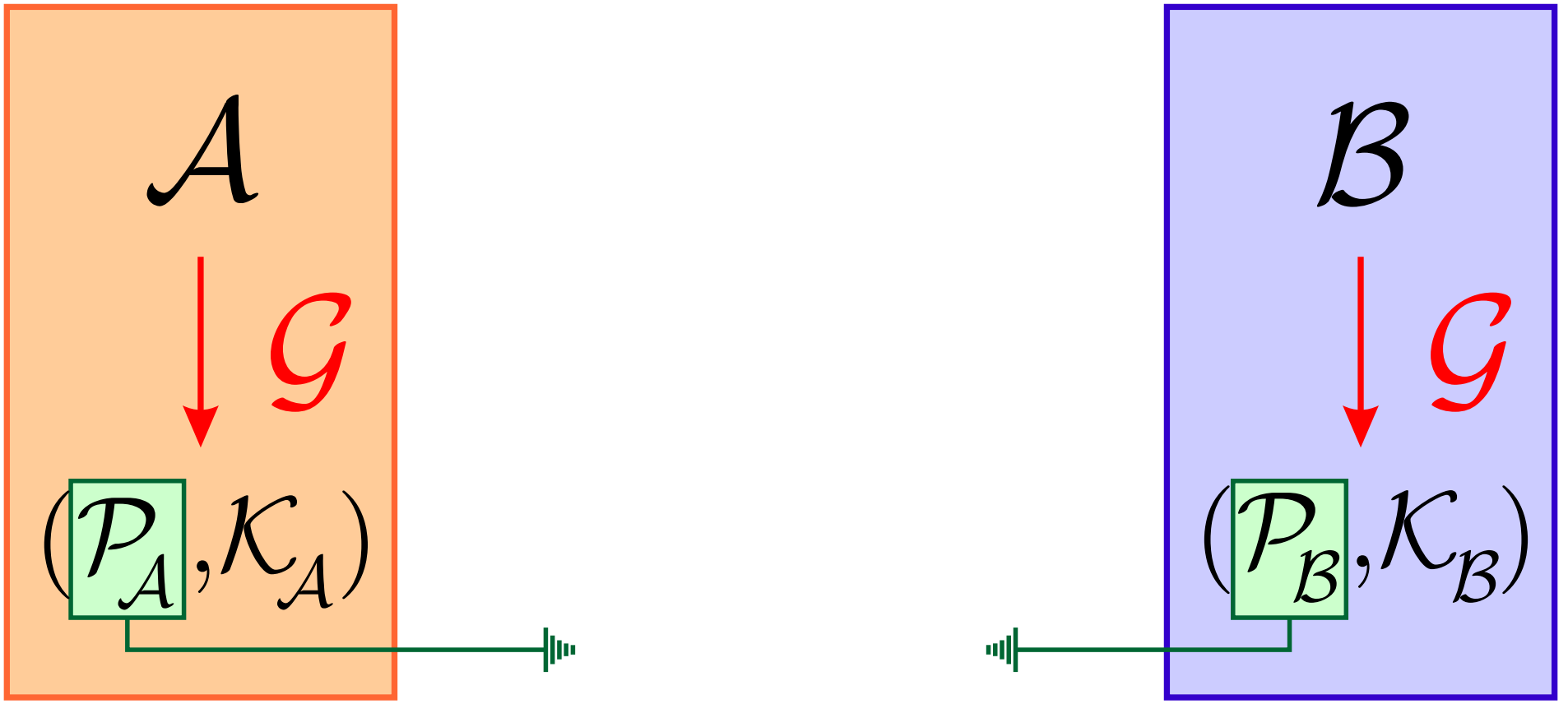
Fonctionnement

Le système est composé de trois algorithmes publiques :

- Le générateur de clefs $\mathcal{G}$: crée un couple clef publique/clef privée $(\mathcal{P}, \mathcal{K})$.
- Le chiffrement $\mathcal{E}$: utilise la clef publique pour chiffrer.
- Le déchiffrement $\mathcal{D}$: utilise la clef privée pour déchiffrer.

La cryptographie à clef publique

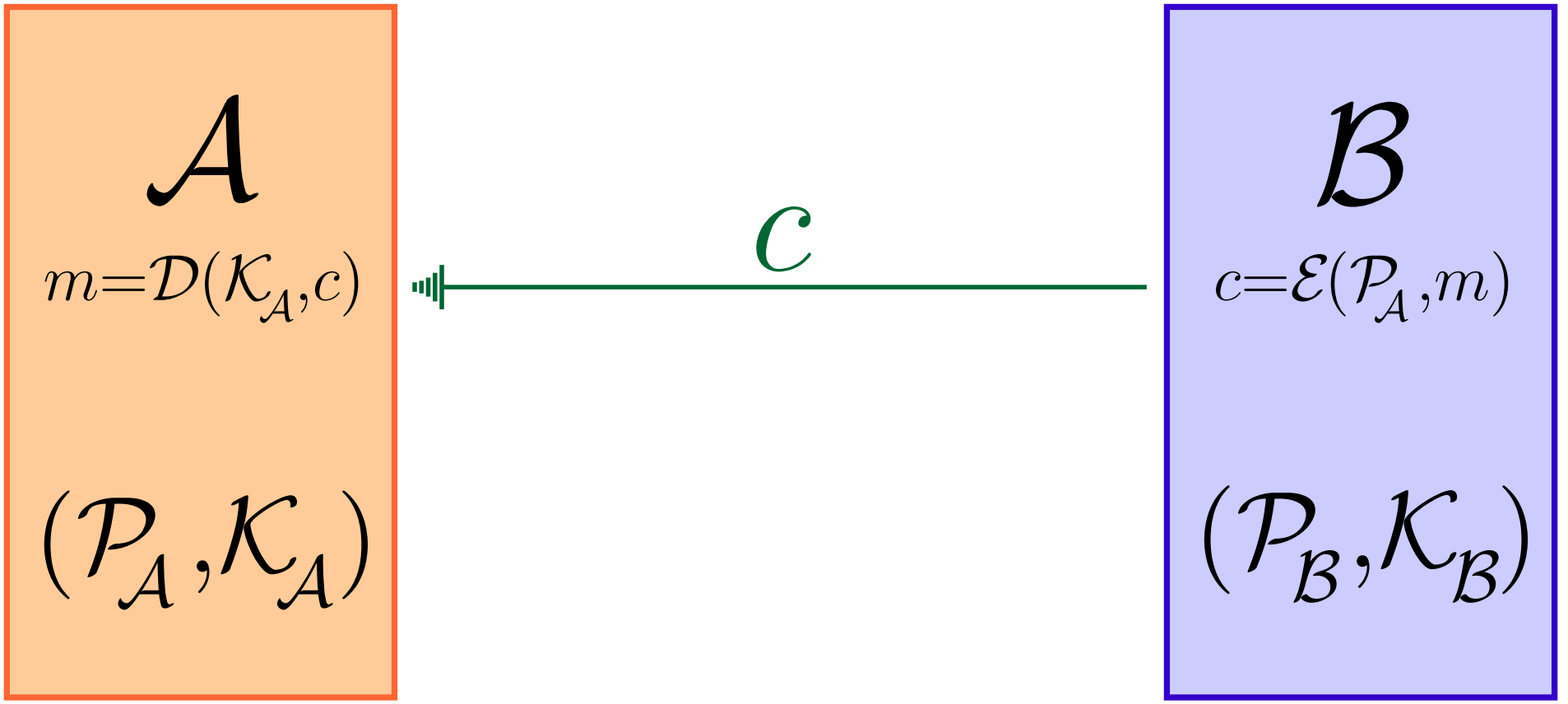
Fonctionnement (suite)



- $\mathcal{A}$ et $\mathcal{B}$ génèrent leurs clefs et les rendent publiques.

La cryptographie à clef publique

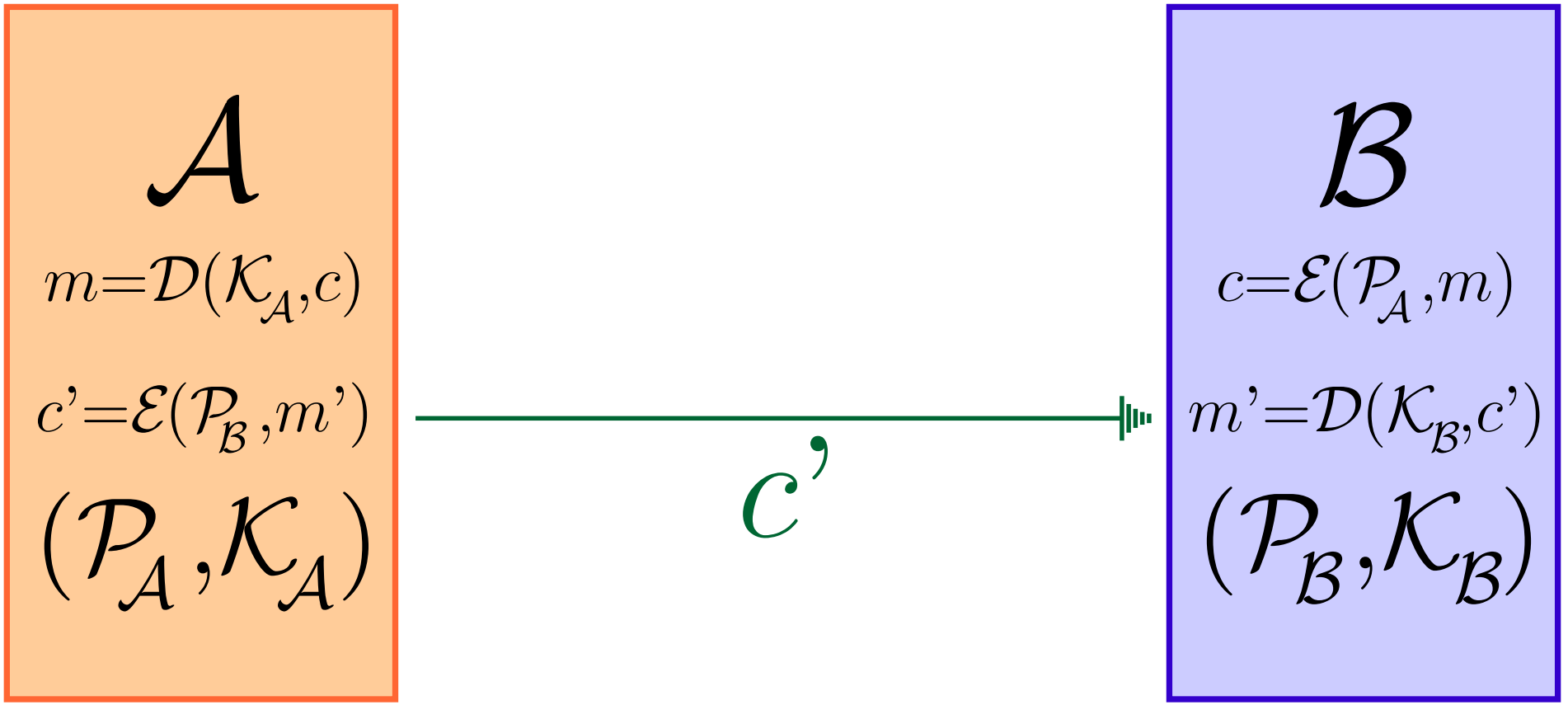
Fonctionnement (suite)



- $\mathcal{B}$ veut envoyer un message m : il le chiffre et le diffuse.
  - $\mathcal{A}$ le déchiffre avec sa clef privée.

La cryptographie à clef publique

Fonctionnement (suite)

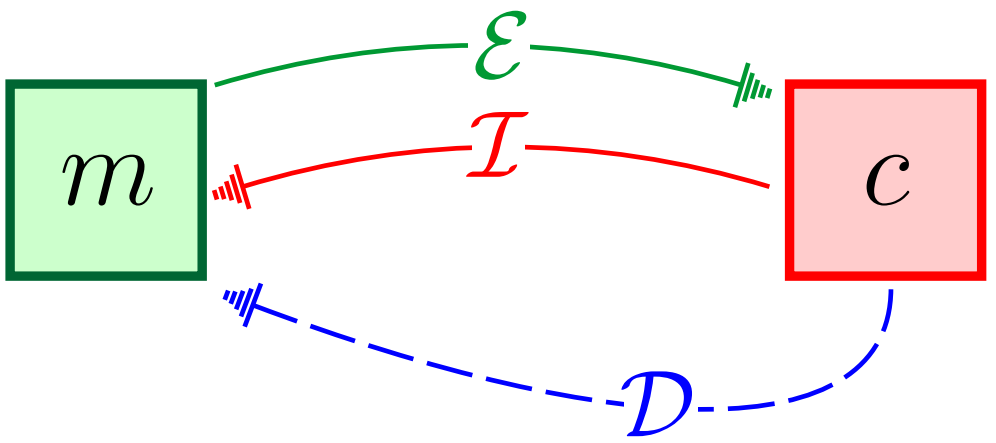


- $\mathcal{A}$ répond par le chiffré c'.
  - $\mathcal{B}$ le déchiffre et lit la réponse m'.

La cryptographie à clef publique

Mise en œuvre

- Il faut utiliser une fonction à sens unique avec trappe.

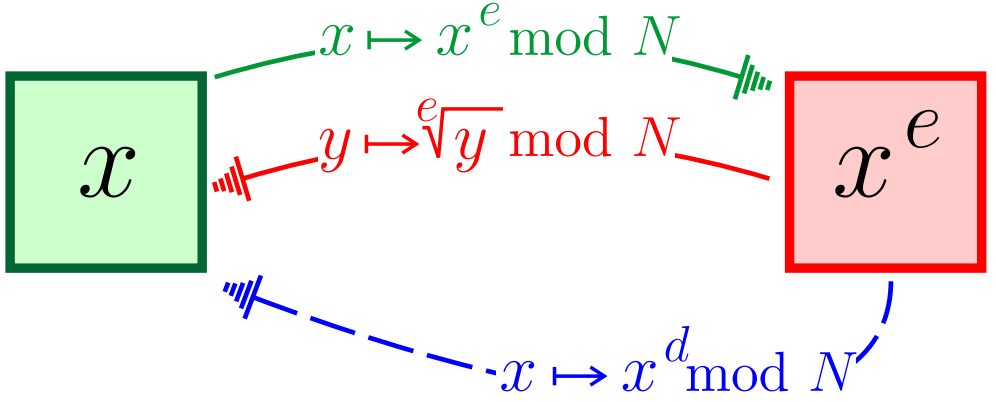


- $\mathcal{E}$: fonction à sens unique publique
- $\mathcal{I}$: problème d'inversion difficile
- $\mathcal{D}$: la trappe utilisant un secret

La cryptographie à clef publique

Mise en œuvre

- Il faut utiliser une fonction à sens unique avec trappe.



- $\mathcal{E}$: exponentiation modulaire
- $\mathcal{I}$: problème d'extraction de racine
- $\mathcal{D}$: utilise l'exposant secret d

- Retrouver d à partir de e et N doit aussi être difficile.

La cryptographie à clef publique

Les systèmes connus

- Les systèmes utilisés manquent de diversité :
  - toujours des problèmes de théorie des nombres.
    - RSA : factorisation / extraction de racine
    - Diffie-Hellman/El Gamal : logarithmes discrets
    - Courbes elliptiques : logarithmes sur des courbes
- Heureusement :
  - Sac-à-dos, NTRU, HFE...
  - McEliece : décodage d'un code aléatoire/ distinguabilité des codes de Goppa.
  - Mais il en faudrait d'autres !

Partie II

Introduction aux codes correcteurs d'erreurs

Les codes correcteurs d'erreurs

En quoi cela consiste...

Lors d'une communication sur un canal bruité :

- à l'émission : ajouter une redondance à un message,
- à la réception : détecter les erreurs et les corriger.

La théorie des codes étudie comment :

- corriger un maximum d'erreurs pour un taux de transmission donné,
- coder et décoder à moindre coût,
- s'adapter à des formes d'erreurs particulières...

Les codes linéaires

C'est la plus ancienne des catégories de codes correcteurs d'erreurs :

- ▷ codes en blocs
- ▷ définis par leur matrice génératrice

$$\mathcal{G} = \begin{pmatrix} 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \end{pmatrix}$$

$00 \longrightarrow 00000$	$10 \longrightarrow 10110$
$01 \longrightarrow 01011$	$11 \longrightarrow 11101$

Les codes linéaires

C'est la plus ancienne des catégories de codes correcteurs d'erreurs :

- ▷ codes en blocs
- ▷ définis par leur matrice génératrice

$$\mathcal{G} = \begin{pmatrix} 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \end{pmatrix}$$

$$\text{'a'} = 97 = 01100001$$
$$01|10|00|01 \longrightarrow 01011|10110|00000|01011$$

Les codes linéaires

Terminologie

- le code $\mathcal{C}$: espace vectoriel engendré par $\mathcal{G}$.
- la dimension k : dimension de $\mathcal{C}$.
- la longueur n : nombre de colonnes de $\mathcal{G}$.
- Le poids de Hamming : nombre de coordonnées non nulles d'un mot.
- la distance minimale d : distance de Hamming minimale entre deux mots du code $\mathcal{C}$.

Les codes linéaires

Terminologie (suite)

- la matrice de parité $\mathcal{H}$ a pour noyau le code $\mathcal{C}$
  - $c \in \mathcal{C} \iff \mathcal{H}c^t = 0$.
  - $\mathcal{H}$ est de taille $(n-k) \times n$.
- le syndrome $\mathcal{S}_x$ d'un mot x est $\mathcal{S}_x = \mathcal{H}x^t$
  - les éléments du code ont un syndrome nul.
  - si $x = c + e$ et $c \in \mathcal{C}$, alors $\mathcal{S}_x = \cancel{\mathcal{H}c^t} + \mathcal{H}e^t = \mathcal{S}_e$.
- Décoder : trop de sens différents...
  - Maximum de vraisemblance : trouver le mot qui a la meilleure probabilité d'avoir été émis.

Quelques codes connus

Les codes de Hamming : $[2^r - 1, 2^r - r - 1, 3]$

- les colonnes de $\mathcal{H}$ sont toutes différentes.
- permettent de corriger une seule erreur.

- Pour $r = 3$, c'est un code $[7, 4, 3]$.
- Si le canal fait $\frac{1}{1000}$ d'erreurs :
  - sans code : probabilité d'erreur de 0.4%
  - avec code : probabilité d'erreur de 0.002%

Quelques codes connus

Les codes de Hamming : $[2^r - 1, 2^r - r - 1, 3]$

- ▷ les colonnes de $\mathcal{H}$ sont toutes différentes.
- ▷ permettent de corriger une seule erreur.

Les codes de Reed-Solomon : $[n, k, n - k + 1]$

- ▷ codes sur $\mathbb{F}_{2^m}$ avec $n \leq 2^m$.
- ▷ code d'évaluation des polynômes de degré $< k$ sur $\mathbb{F}_{2^n}$.

Les codes de Reed-Muller (évaluation de polynômes multivariés), de Goppa (très bons codes binaires)...

Problèmes difficiles liés aux codes

Quelques exemples

- Syndrome Decoding :

  Entrée : une matrice $\mathcal{H}$, un syndrome $\mathcal{S}$ et un poids w.

  Problème : trouver e de poids w avec $\mathcal{H}e^t = \mathcal{S}$.

  ▷ NP-complet [Berlekamp, McEliece, van Tilborg 1978]

- Parameterized Bounded Decoding :

  Entrée : une matrice $\mathcal{H}$ de taille $r \times n$, un syndrome $\mathcal{S}$.

  Problème : trouver e de poids $f(r, n)$ avec $\mathcal{H}e^t = \mathcal{S}$.

  ▷ NP-complet pour tout f "raisonnable".

Partie III

L'utilisation de codes en cryptographie

Le chiffrement de McEliece
[McEliece 1978]

On va utiliser un code de Goppa et cacher sa structure.

- ▷ bons codes binaires : corrigent $\frac{n-k}{\log_2 n}$ erreurs,
- ▷ efficaces à décoder : Euclide, Berlekamp-Massey...

- Chiffrement :
  - ◇ on code le message en mot de code,
  - ◇ on y ajoute une erreur de poids corrigible.
- Déchiffrement :
  - ◇ celui qui connaît la structure décode les erreurs,
  - ◇ sinon il faut décoder dans un code quelconque.

Le chiffrement de McEliece

Les algorithmes

- Génération de clef :
  - générer un code de Goppa $\Gamma(g, \mathcal{L})$ et sa matrice génératrice $\mathcal{G}$.
  - calculer $\mathcal{G}' = \mathcal{Q}\mathcal{G}\mathcal{P}$: qui semble aléatoire.
    - $\mathcal{G}'$ est la clef publique/$\Gamma(g, \mathcal{L})$, $\mathcal{Q}$ et $\mathcal{P}$ sont privés.
- Chiffrement d'un message m :
  - calculer $c = m\mathcal{G}'$ et générer e une erreur de poids t.
  - transmettre le chiffré $c + e$.

Le chiffrement de McEliece

Déchiffrement

- Déchiffrement (légitime) de $c+e$:
  - $c' = (c+e)\mathcal{P}^{-1} = m\mathcal{Q}\mathcal{G}\cancel{\mathcal{P}\mathcal{P}^{-1}} + e\mathcal{P}^{-1} = (m\mathcal{Q})\mathcal{G} + e'$.
  - on décode e' dans $\Gamma(g, \mathcal{L})$ et on trouve $m\mathcal{Q}$.
  - on utilise $\mathcal{Q}^{-1}$ pour retrouver m.
- Décryptement (illégitime) de $c+e$:
  - La structure de $\mathcal{G}'$ est cachée :
    - décoder t erreurs dans un code quelconque (SD).
  - $\mathcal{G}'$ est un matrice de Goppa permutée :
    - retrouver la structure cachée (beaucoup de codes).

La variante de Niederreiter

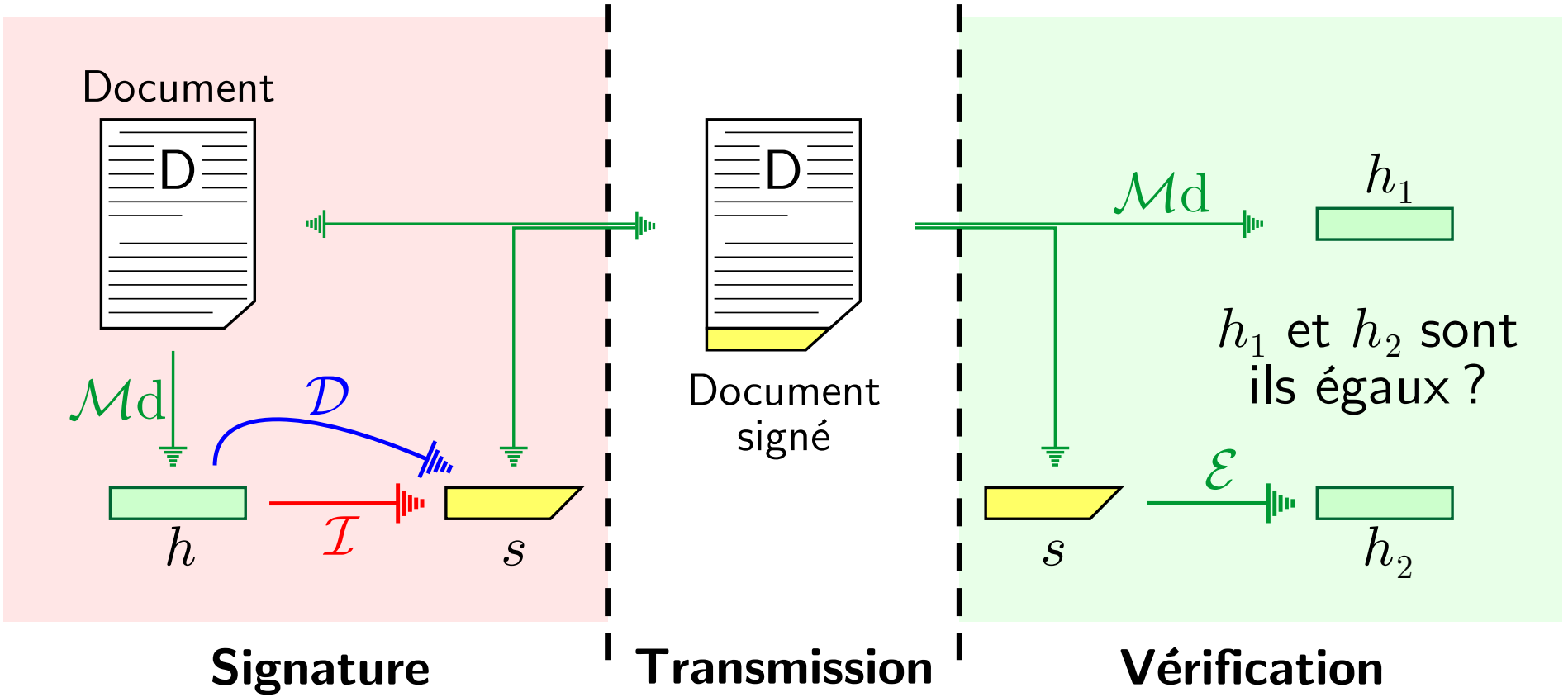
[Niederreiter 1986]

Dual de McEliece avec la matrice de parité.

- Clef publique : $\mathcal{H}' = \mathcal{Q}\mathcal{H}\mathcal{P}$.
- Le message m est codé dans une erreur e_m de poids t et le syndrome $\mathcal{S}_m = \mathcal{H}e_m$ est le cryptogramme.
- ▷ C'est cette version qui est utilisée dans la signature...

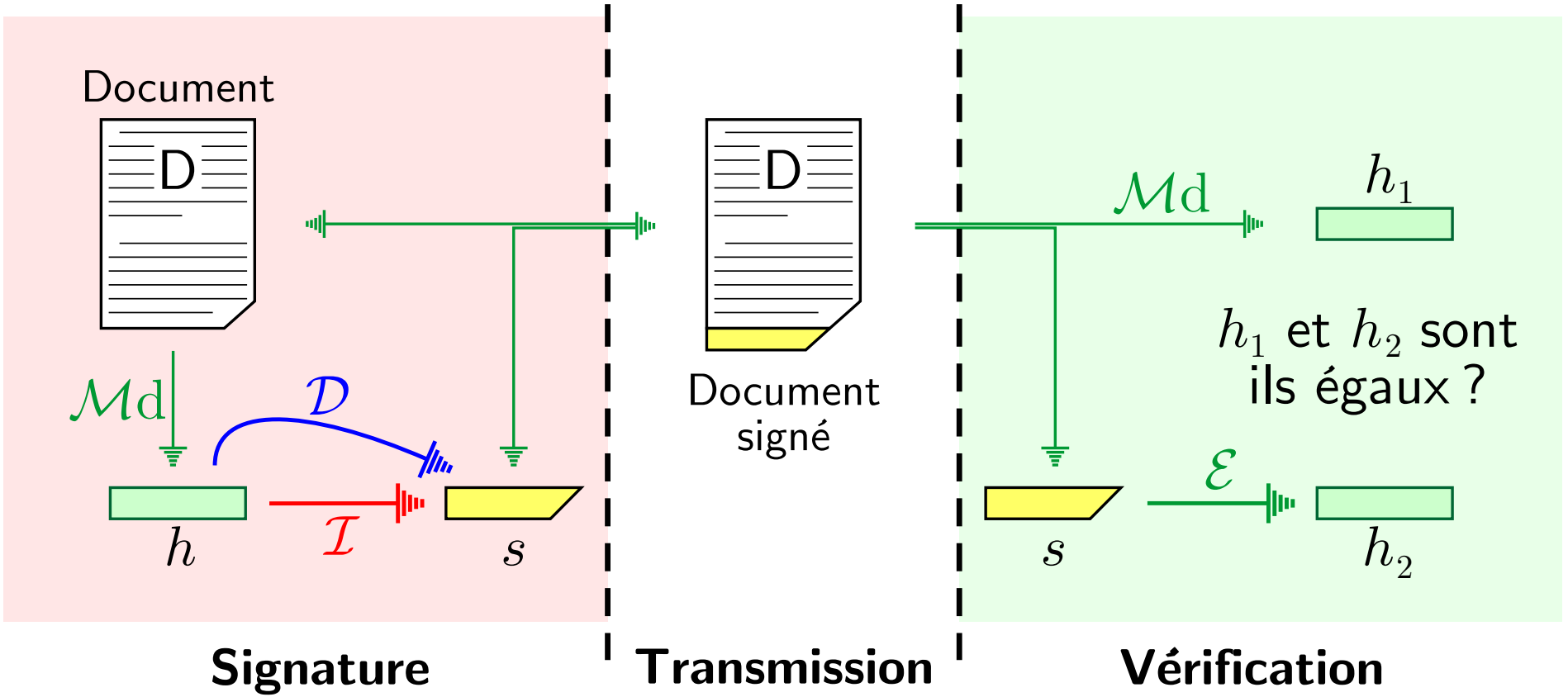
Signature numérique

Construction standard



Signature numérique

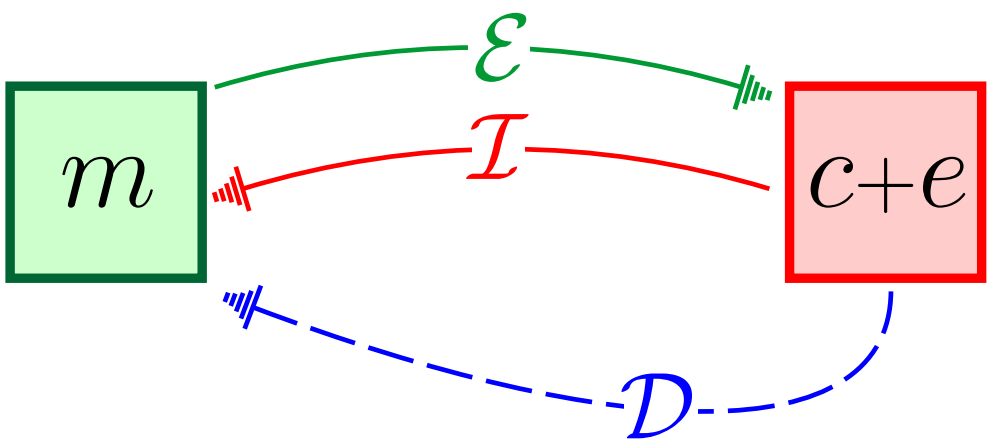
Construction standard



Il faut pouvoir déchiffrer un h quelconque.

La signature McEliece

[Courtois, Finiasz, Sendrier – Asiacrypt 2001]

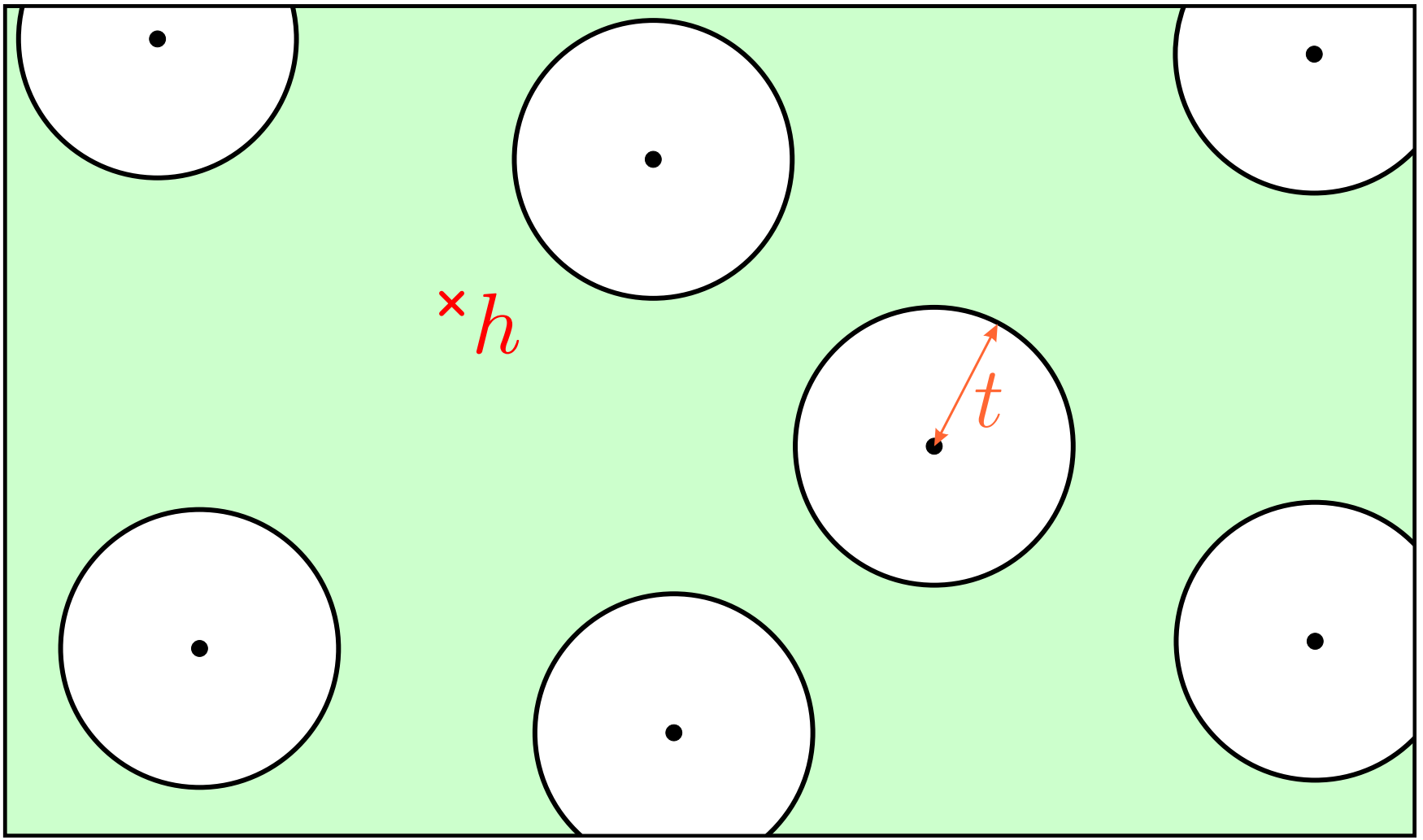


- $\mathcal{E}$: codage et ajout d'un bruit aléatoire.
- $\mathcal{I}$: décodage d'un code quelconque.
- $\mathcal{D}$: algorithme de décodage utilisant la structure cachée.

- $\mathcal{E}$ n'est pas surjective :
  - $\mathcal{D}$ ne s'applique qu'à un sous-ensemble des mots,
  - c'est cela qui a longtemps posé problème...

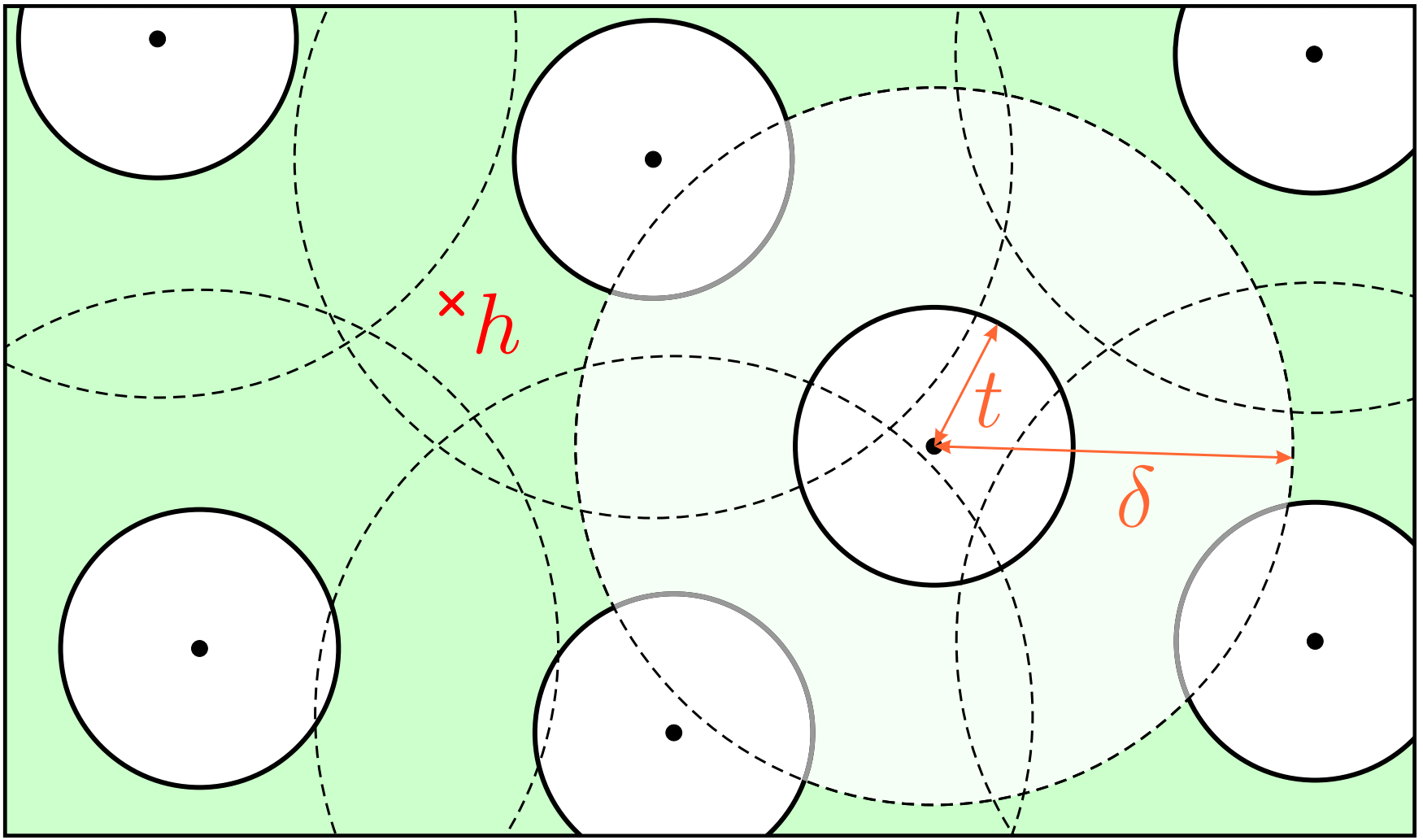
La signature McEliece

Comment faire ?



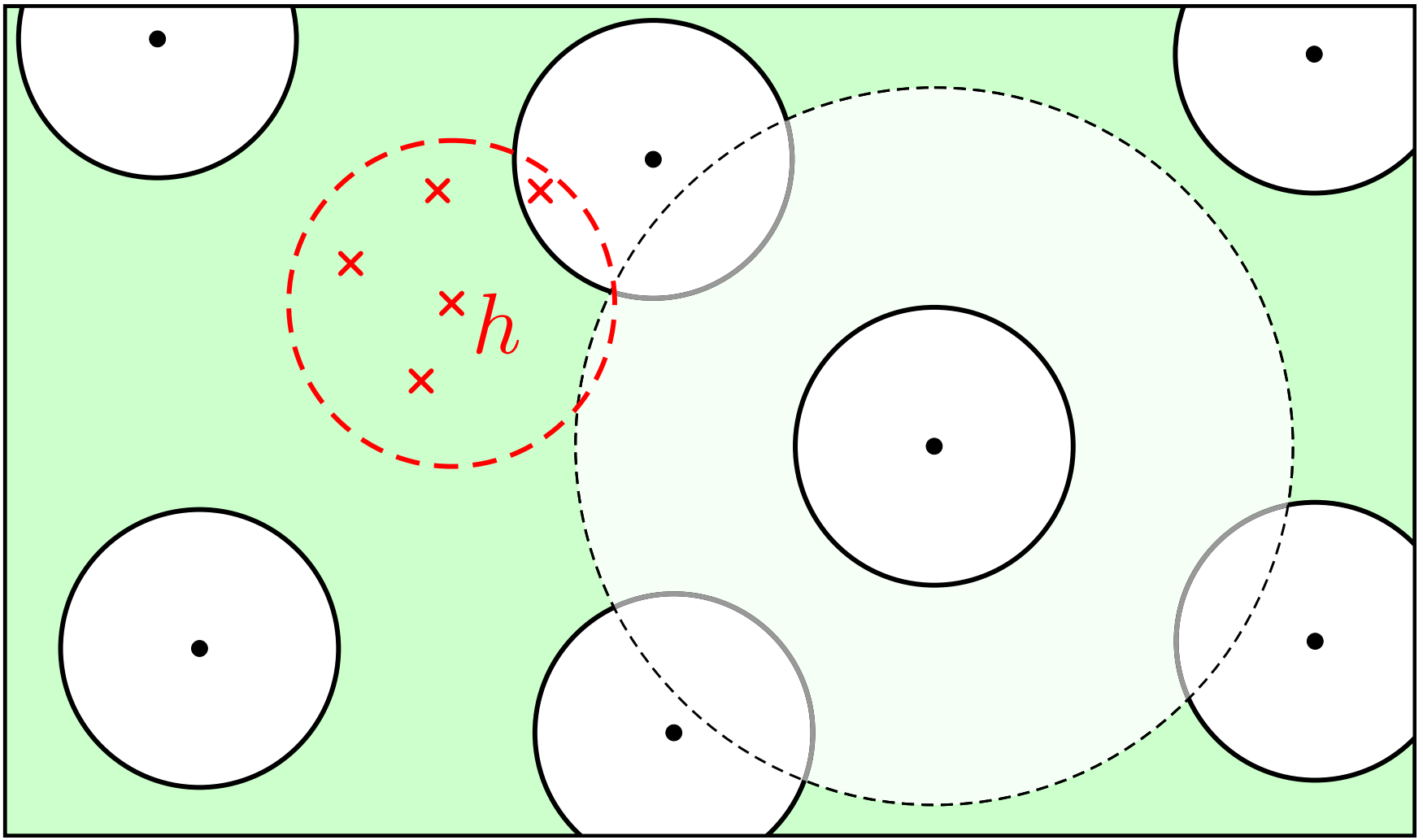
La signature McEliece

Décodage jusqu'au rayon de recouvrement



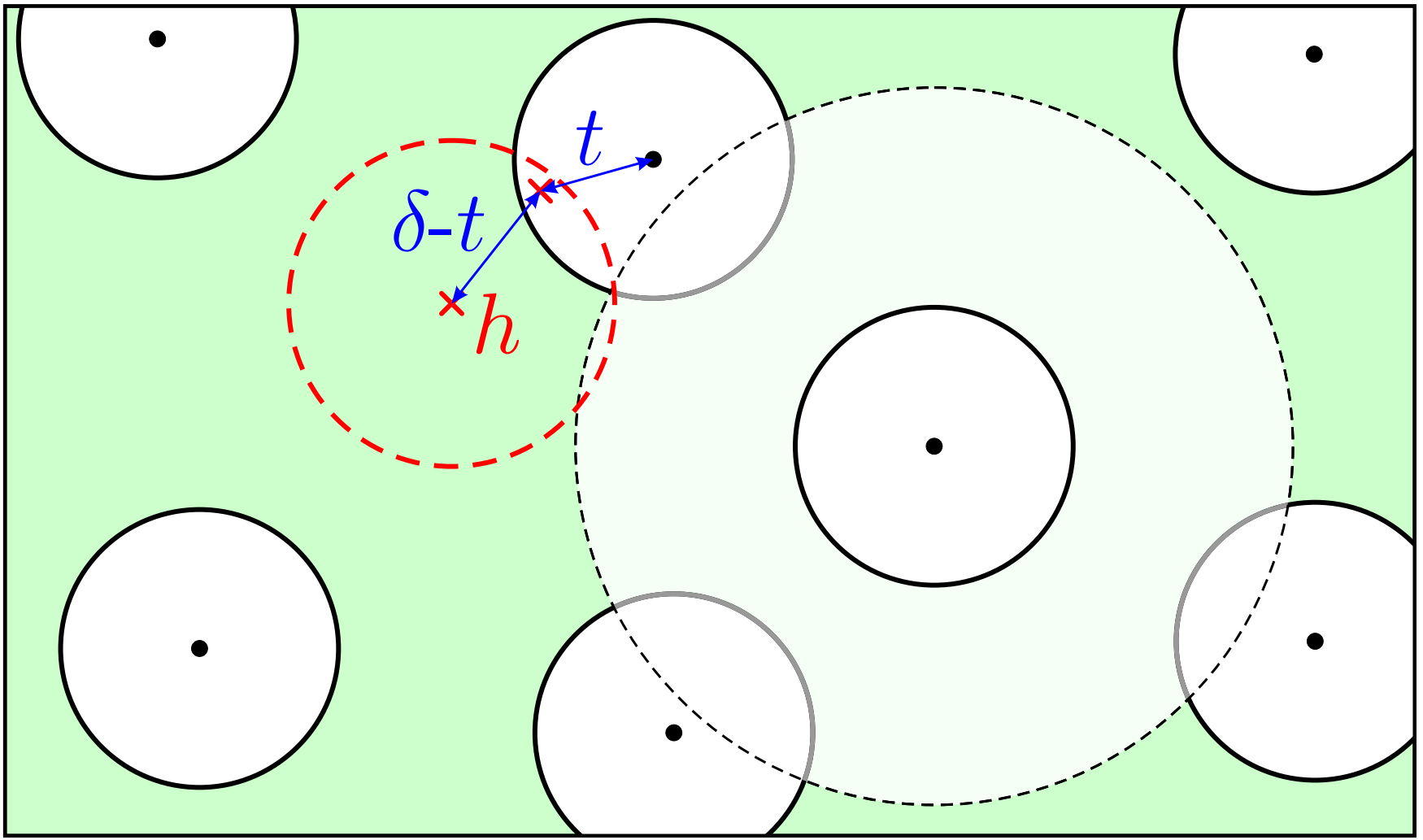
La signature McEliece

Décodage jusqu'au rayon de recouvrement



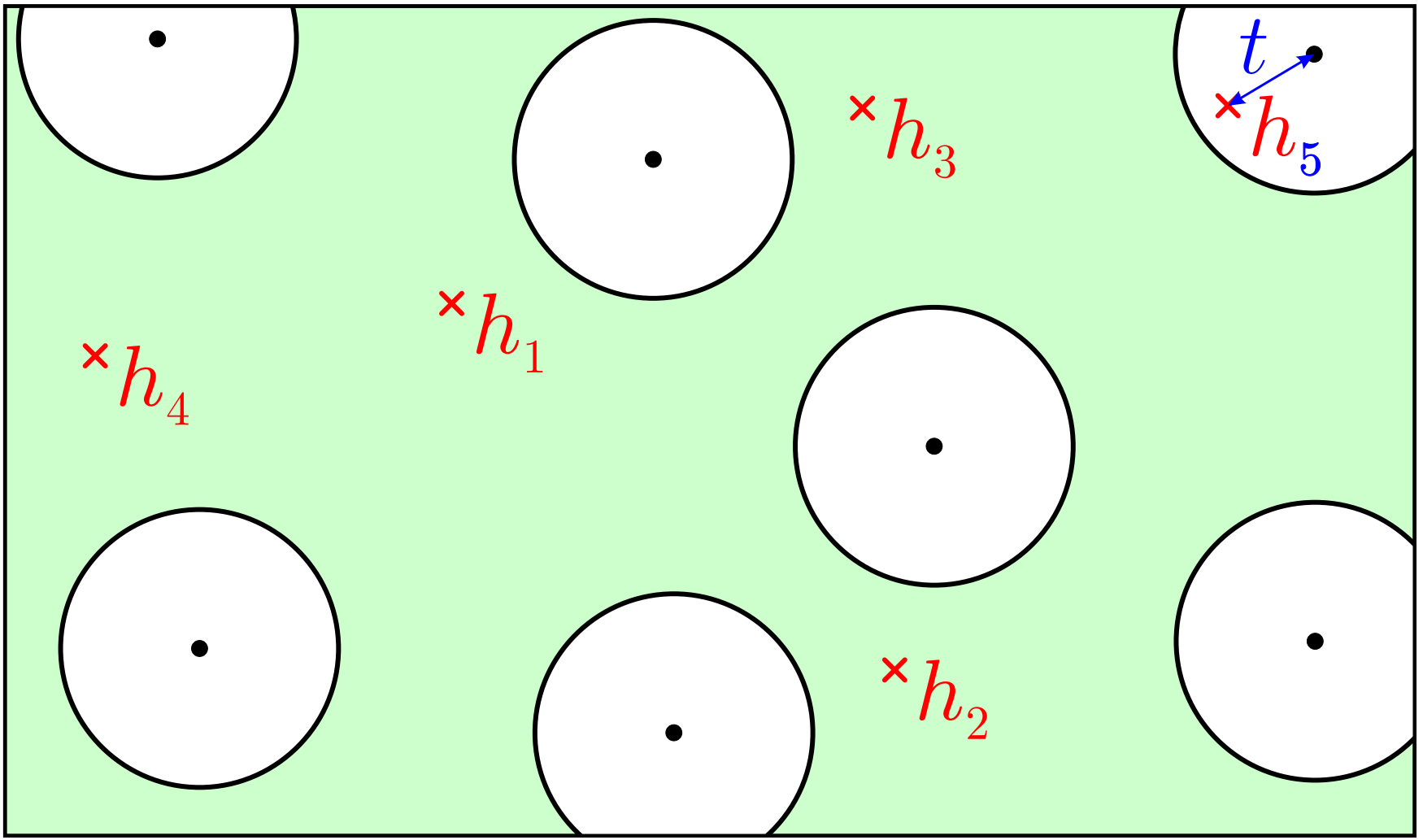
La signature McEliece

Décodage jusqu'au rayon de recouvrement

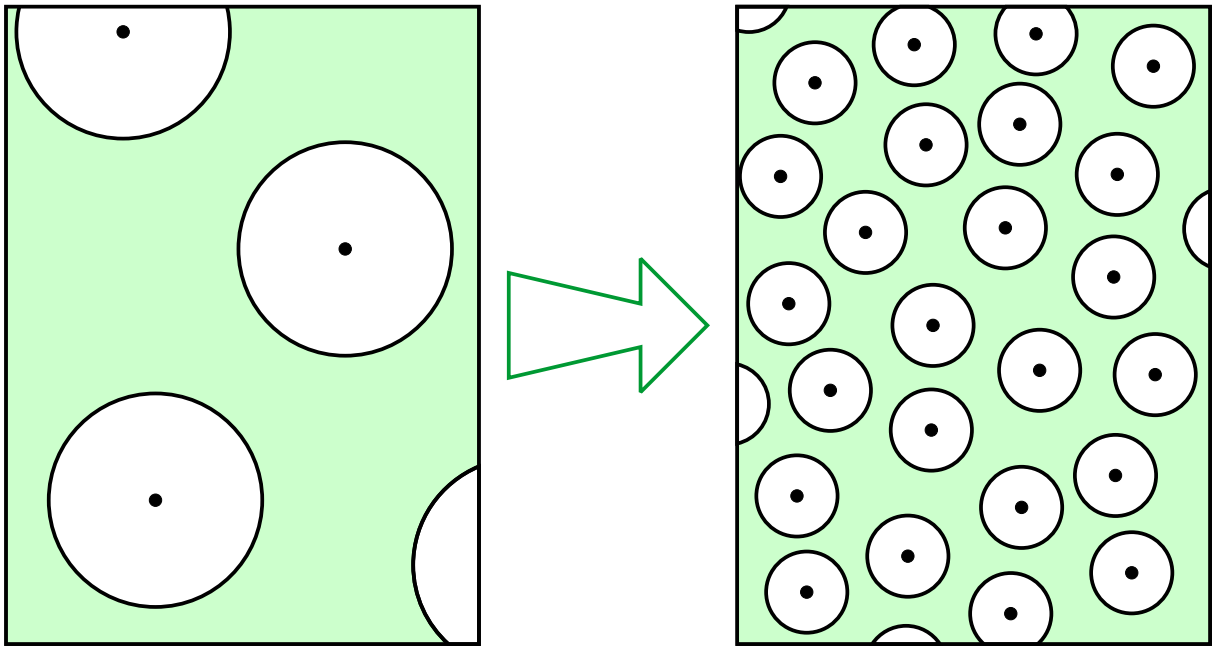


La signature McEliece

Utilisation d'un compteur



- Il faut une bonne sécurité et un petit nombre d'essais :
  - des paramètres conviennent : $n = 2^{16}$, $t = 9$.
  - McEliece proposait : $n = 2^{10}$ et $t = 50$.



La signature McEliece

Conclusion

- Il faut une bonne sécurité et un petit nombre d'essais :
  - des paramètres conviennent : $n = 2^{16}$, $t = 9$.
- Les deux solutions sont quasi équivalentes :
  - signer coûte le temps de $9!$ décodages (une minute),
  - les signatures font de 150 à 80 bits.
- Une implantation FPGA est en cours :
  - le temps de signature est de moins d'une seconde.

- Nouveau schéma qui a de plus de bonnes propriétés.

Mots de poids minimum des Goppa

[Finiasz – ISIT 2003]

- En reprenant l'algorithme de décodage précédent :
  - on peut trouver des mots de poids minimum,
  - on peut estimer le nombre de tels mots.

- Nouveau résultat sur la distribution des poids :
  - les bornes théoriques concernent les poids moyens.

[Levy-dit-Vehel, Litsyn]

Un nouveau chiffrement à clef publique

[Augot, Finiasz – Eurocrypt 2003]

Pour les codes de Reed-Solomon :

- ▷ la distance minimale est de $n - k + 1$,
- ▷ on sait décoder (par liste) $n - \sqrt{nk}$ erreurs,
- ▷ au-delà, le décodage est NP-dur.

  [Goldreich, Rubinfeld, Sudan 1995] & [Vardy 2004]

- Faire reposer la sécurité sur ce problème.
- On suppose que décoder dans un code RS+1 engendré par un Reed-Solomon et un mot aléatoire est difficile.

Polynomial Reconstruction

Fonctionnement

- Clef publique : $c + E$ avec $poids(E) = W$
- Clef privée : (c, E)
- Chiffrement : e de poids w

$$\mathcal{E}(m) = y = \overbrace{\overline{eval(m) + \alpha\big(c + E\big)}}^{\text{mot du RS+1}} + e$$

- Déchiffrement : on poinçonne le code sur E

$$\bar{y} = \overline{\overline{eval(m)}} + \alpha\bar{c} + \bar{e}$$

Polynomial Reconstruction

Fonctionnement

- Clef publique : $c + E$ avec $poids(E) = W$
- Clef privée : (c, E)
- Chiffrement : e de poids w

$$\mathcal{E}(m) = y = \overbrace{\overline{eval(m)} + \alpha\big(c + E\big)}^{\text{mot du RS+1}} + e$$

- Déchiffrement : on poinçonne le code sur E

$$\bar{y} = \overline{\overline{eval(m)}} + \alpha\bar{c} + \bar{e}$$

Décoder dans RS+1 est polynomial... [Coron 2003]

Partie IV

Une nouvelle famille de fonctions de hachage

[Augot, Finiasz, Sendrier – ePrint 2003]

Fonction de hachage

Définition

- Une fonction de hachage sert à garantir l'intégrité d'un document.
- C'est une fonction de $\mathbb{F}_2{}^*$ dans $\mathbb{F}_2{}^r$:
  - associe à un document $\mathbf{D}$ un haché $h(\mathbf{D})$ de r bits.
- Utilisées pour :
  - les signatures,
  - les téléchargements...

Fonction de hachage

Propriétés

- Une fonction de hachage doit être :
  - à sens unique :

$$x \mapsto \mathbf{D} \text{ tel que } h(\mathbf{D}) = x$$

  - sans $2^{\text{ème}}$ antécédent :

$$\mathbf{D}_1 \mapsto \mathbf{D}_2 \text{ tel que } h(\mathbf{D}_1) = h(\mathbf{D}_2)$$

  - sans collisions :

$$\mathbf{D}_1 \text{ et } \mathbf{D}_2 \text{ tels que } h(\mathbf{D}_1) = h(\mathbf{D}_2)$$

- Une fonction qui vérifie cela pourra garantir l'intégrité.

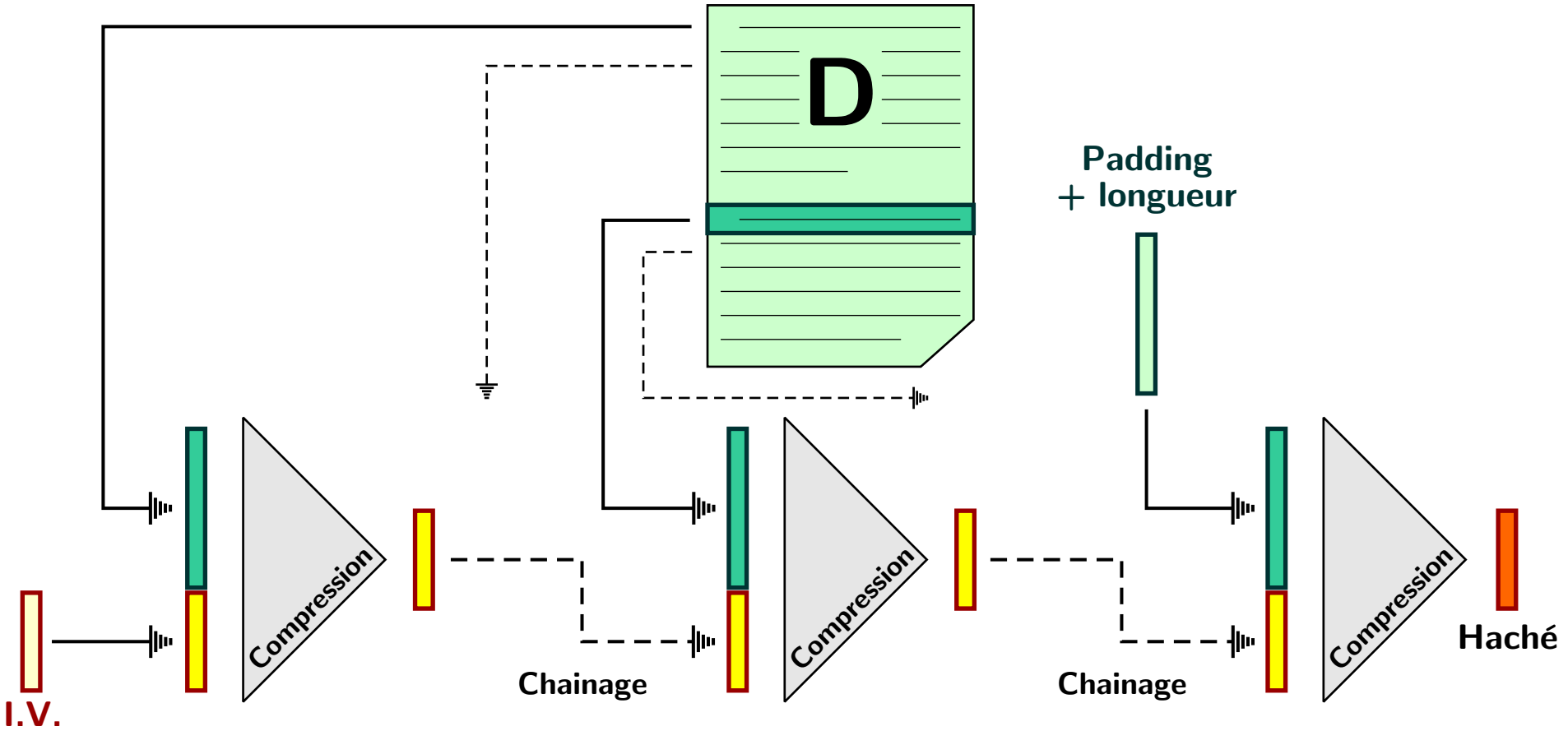
Les fonctions connues

- Plusieurs systèmes existants ont des failles :
  - MD4, MD5
  - SHA-0, SHA-1, SHA-256...
  - RIPEMD, RIPEMD-128, RIPEMD-160...
  - HAVAL-128, HAVAL-160...

- On veut une construction rapide où les trois problèmes se ramènent à des problèmes difficiles.

La construction classique

[Damgård & Merkle – Crypto 1989]



- La sécurité de la fonction de compression suffit.

Utilisation d'un syndrome

On se ramène au problème de Syndrome Decoding :

- ▷ on code l'entrée en mot de poids w et longueur n,
- ▷ la sortie est son syndrome par une matrice de parité $\mathcal{H}$ de taille $r \times n$ tirée aléatoirement.

- Inverser = trouver un mot de poids et syndrome donnés.
- Collision = deux mots de même syndrome = un mot de code de poids double ou moins.
- On peut coder $\log_2 \binom{n}{w}$ bits dans un mot.

Utilisation d'un syndrome

On se ramène au problème de Syndrome Decoding :

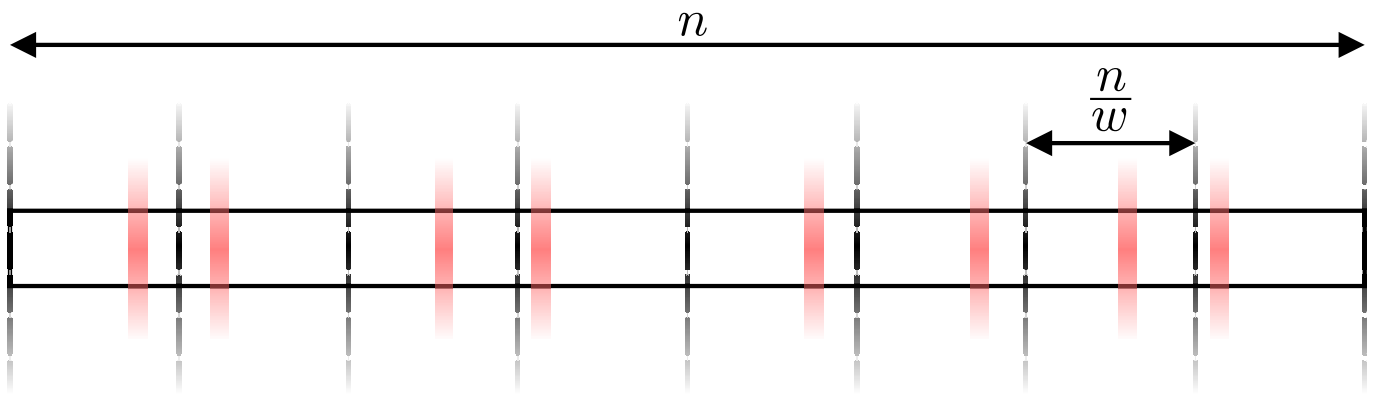
- ▷ on code l'entrée en mot de poids w et longueur n,
- ▷ la sortie est son syndrome par une matrice de parité $\mathcal{H}$ de taille $r \times n$ tirée aléatoirement.

- ▶ Inverser = trouver un mot de poids et syndrome donnés.
- ▶ Collision = deux mots de même syndrome = un mot de code de poids double ou moins.

Problème : le codage est trop lent.

Avec des mots réguliers

- Un mot régulier de longueur n et poids w possède un bit non nul dans chaque bloc de $\frac{n}{w}$ positions.



- Le codage est alors très rapide :
  - chaque groupe de $\log_2 \frac{n}{w}$ bits donne une position.
  - $\binom{n}{w}^w$ mots réguliers différents (au lieu de $\binom{n}{w}$).

La fonction de compression

Description

Données :

- une matrice $\mathcal{H}$ de taille $r \times n$,
- un poids w tels que $w \log_2 \frac{n}{w} > r$.

Fonctionnement :

- convertir les $w \log_2 \frac{n}{w}$ bits d'entrée (r de chaînage le reste pris dans $\mathbf{D}$) en un mot régulier e,
- calculer $h = \mathcal{H}e^t$.

Étude de la sécurité

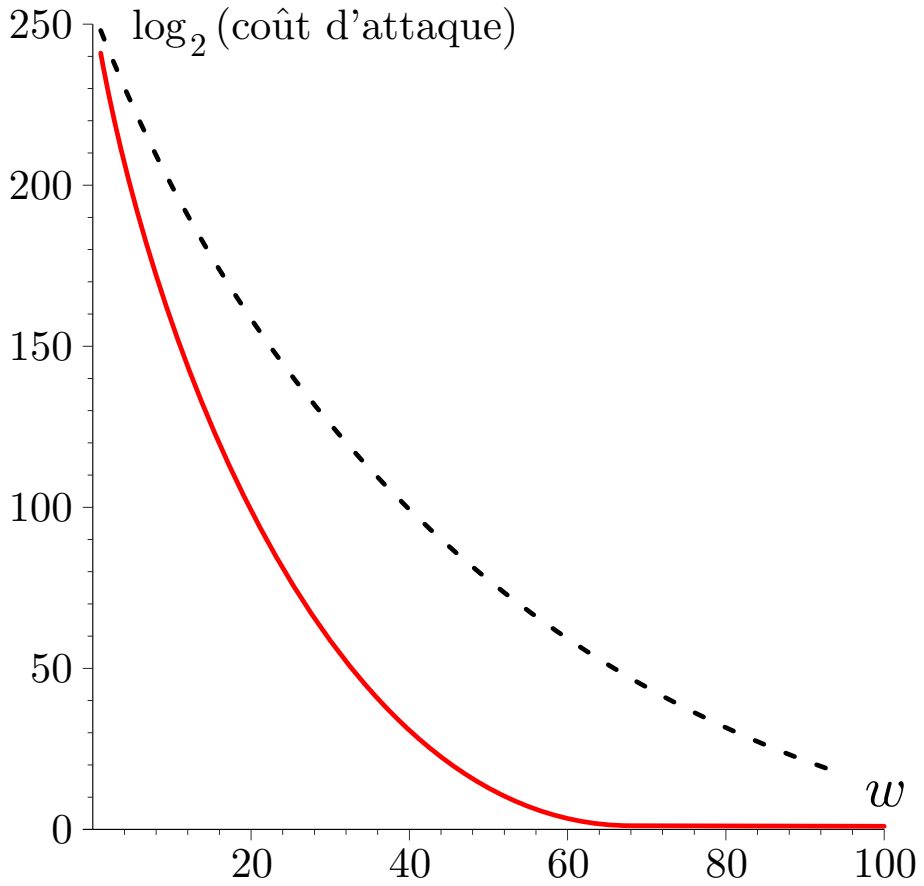
La théorie...

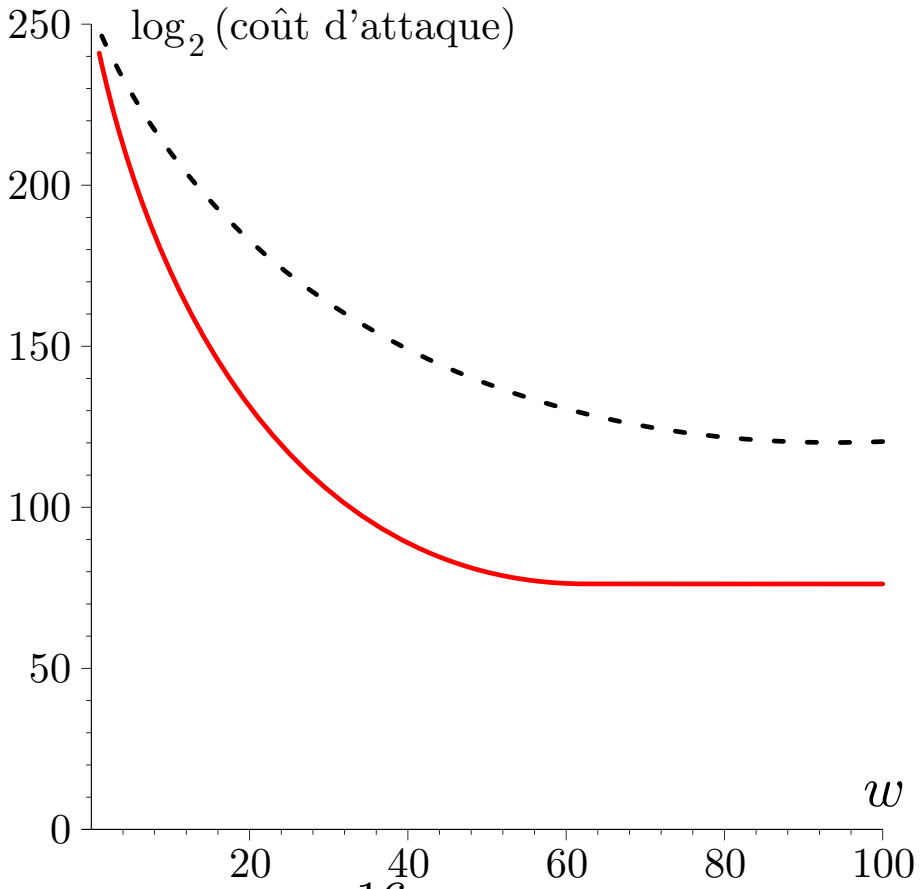
- Inverser : chercher un mot régulier de syndrome donné.
  - C'est un problème NP-complet.
- Chercher une collision : un mot de code 2-régulier.
  - C'est aussi NP-complet.
- Pour une instance donnée ?

Étude de la sécurité

La pratique...

- Par recherche d'ensembles d'information :





paramètres : $r = 256, n = 2^{16}$

Paradoxe des anniversaires généralisé

[Wagner 2002]

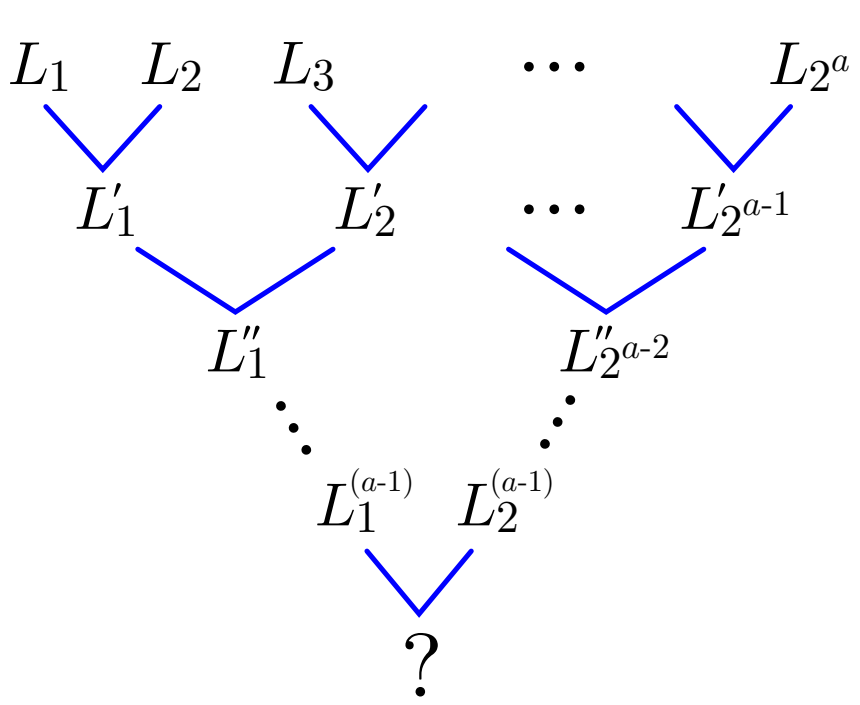
- Soient Ω_1,...,Ω_w des ensembles de N colonnes.
  - on cherche des $x_i \in \Omega_i$ tels que $x_1 \oplus ... \oplus x_w = 0$.

- Paradoxe des anniversaires classique :
  - on construit deux listes L_1 et L_2 d'éléments $x_1 \oplus ... \oplus x_{w/2}$ et $x_{w/2+1} \oplus ... \oplus x_w$,
  - on cherche une collision entre L_1 et L_2.

Paradoxe des anniversaires généralisé

[Wagner 2002]

- Soient Ω_1,...,Ω_w des ensembles de N colonnes.
  - ▷ on cherche des $x_i \in \Omega_i$ tels que $x_1 \oplus ... \oplus x_w = 0$.
- On va couper en 2^a listes.



Combinaisons de $\frac{w}{2^a}$ termes.

$\frac{w}{2^{a-1}}$ termes, avec $\frac{r}{a+1}$ zéros.

$\frac{w}{2^{a-2}}$ termes, avec $\frac{2r}{a+1}$ zéros.

$\frac{w}{2}$ termes, avec $\frac{(a-1)\,r}{a+1}$ zéros.

Étude de la sécurité

[Coron, Joux 2004]

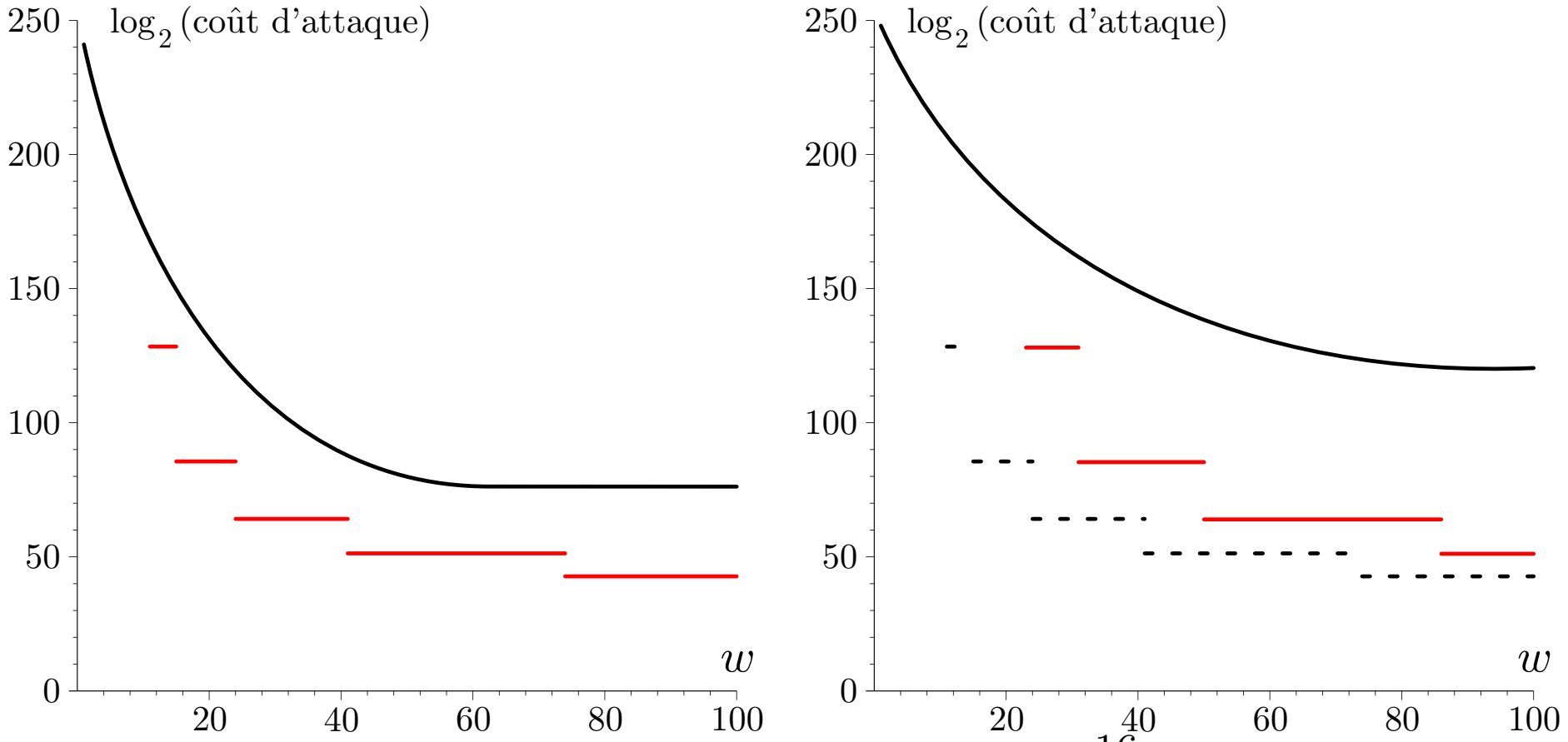
- On annule $\frac{r}{a+1}$ valeurs à chaque fusion de listes
  - si les listes contiennent $2^{\frac{r}{a+1}}$ éléments elle gardent la même taille et il reste une solution à la fin,
  - il faut construire 2^a listes de $2^{\frac{r}{a+1}}$ éléments.
- Pour trouver une collision avec les mots réguliers :
  - dans chaque bloc on peut construire $\binom{\frac{n}{w}}{2}+1$ éléments,
  - on construit 2^a listes de $\left[\binom{\frac{n}{w}}{2}+1\right]^{\frac{w}{2^a}}$ éléments.

Si : $\frac{r}{w}\log_2\left[\binom{\frac{n}{w}}{2}+1\right] \geq \frac{2^a}{a+1}$ on attaque en $\mathcal{O}(2^{\frac{r}{a+1}})$

Étude de la sécurité

Résultats

- On améliore nettement l'attaque :

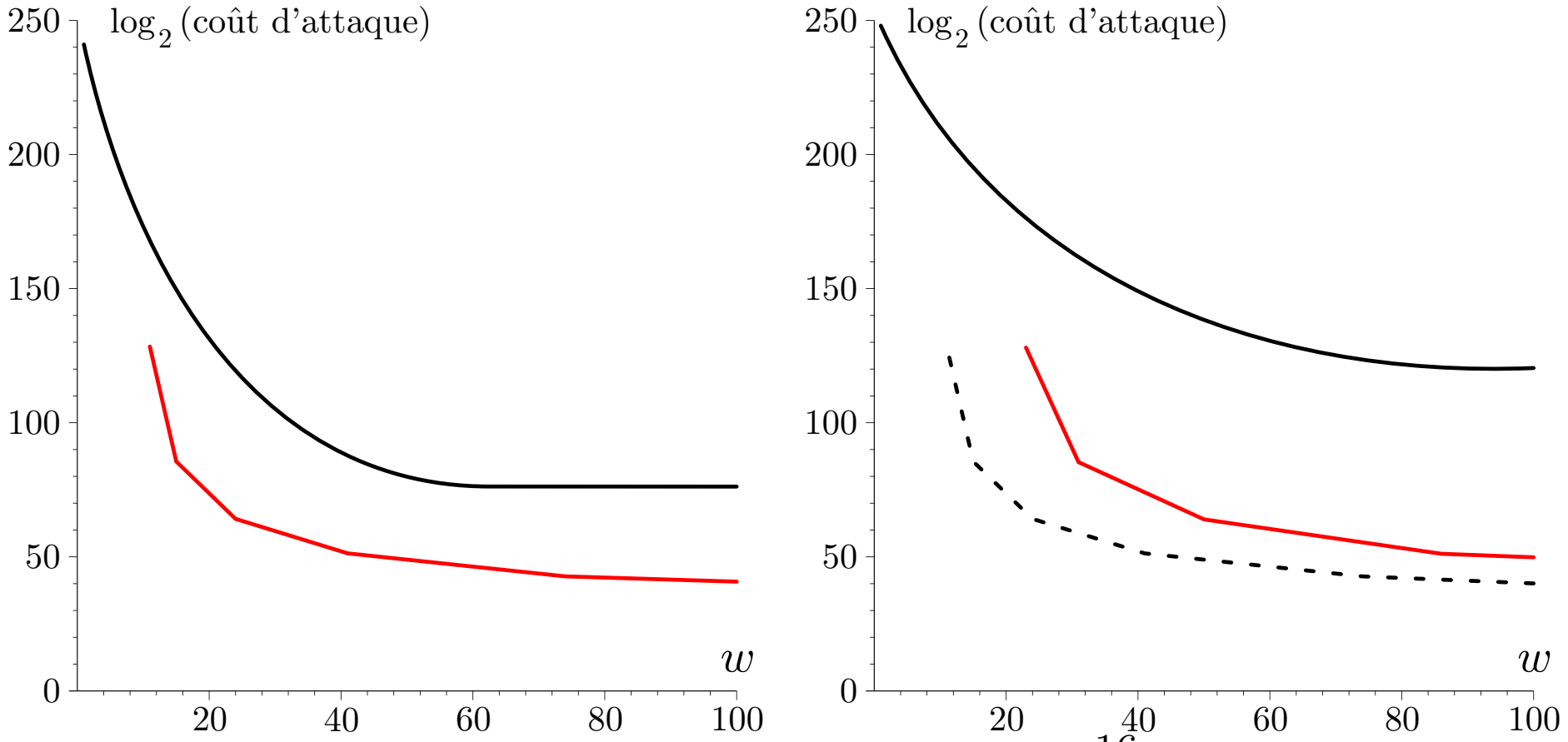


paramètres : $r = 256, n = 2^{16}$

Étude de la sécurité

Résultats

- On améliore nettement l'attaque :

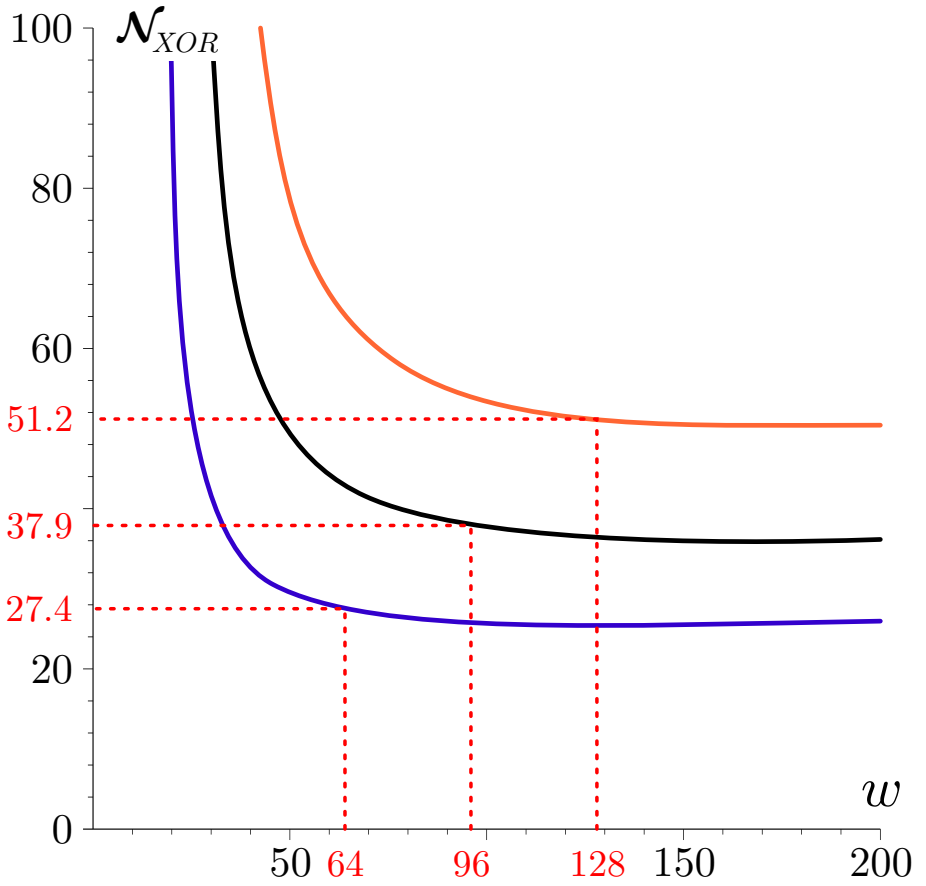


paramètres : $r = 256, n = 2^{16}$

Recherche de paramètres

Efficacité/sécurité

- Il faut être efficace :
  - nombre de XORs par bit de document.
- Il faut $2^{\frac{r}{a+1}} \geq 2^{80}$:
  - on fixe un r et un a et on cherche n et w.



paramètres : $r = 192, n = 2^{16}$ $r = 256, n = 2^{16}$ *et* $r = 256, n = 2^{14}$

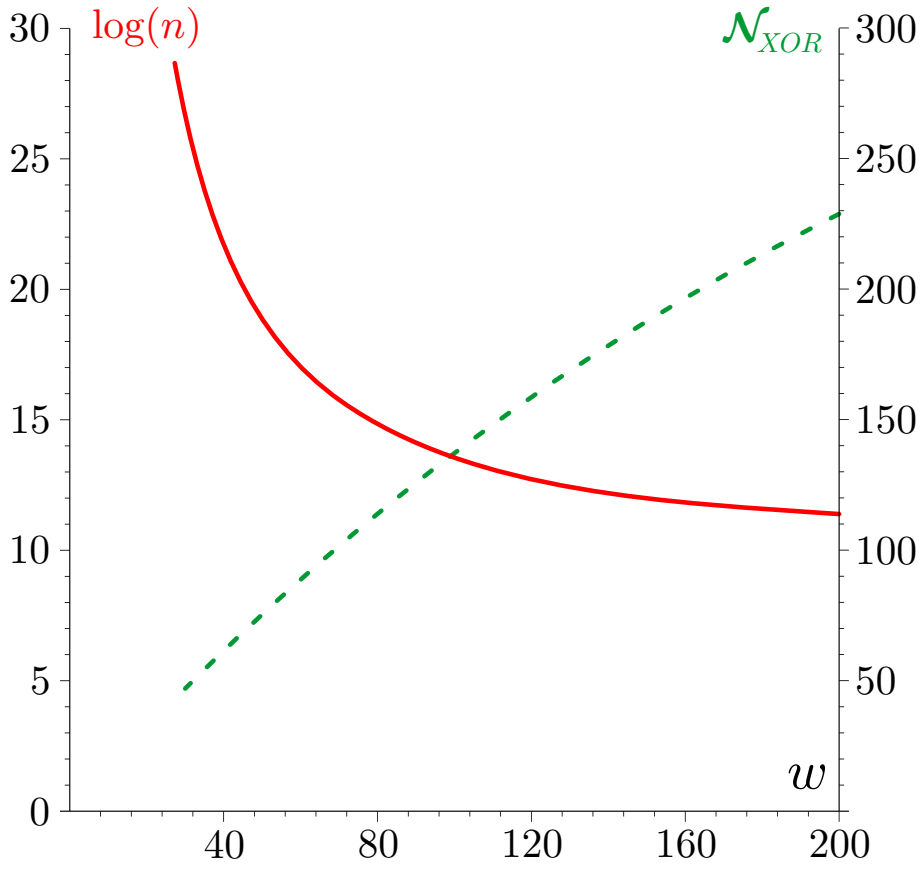
Recherche de paramètres

Valeurs possibles pour a

- On veut fixer une valeur maximale pour a :
  - choisir n et w afin qu'un attaquant ne puisse pas utiliser un a plus grand.
- On ne peut pas limiter a à 1 ou 2 :
  - sinon la fonction ne compresse plus.
- On peut trouver des paramètre limitant a à 3 :
  - on ne compresse toujours pas assez ($\mathcal{N}_{XOR} \simeq 2r$).

Recherche de paramètres

Valeurs acceptables



pour : $r = 400$ *et* $a = 4$

$\log_2\left(\frac{n}{w}\right)$	w	$\mathcal{N}_{XOR}$	taille de $\mathcal{H}$
16	41	64.0	$\sim$ 1 Gbit
15	44	67.7	550 Mbits
14	47	72.9	293 Mbits
13	51	77.6	159 Mbits
12	55	84.6	86 Mbits
11	60	92.3	47 Mbits
10	67	99.3	26 Mbits
9	75	109.1	15 Mbits
8	85	121.4	8.3 Mbits
7	98	137.1	4.8 Mbits
6	116	156.8	2.8 Mbits
5	142	183.2	1.7 Mbits
4	185	217.6	1.1 Mbits

Étude asymptotique

- On part de l'équation : $\frac{r}{w}\log_2\left[\binom{\frac{n}{w}}{2}+1\right] \geq \frac{2^a}{a+1}$
- Si on augmente r, w et n linéairement :
  - $w = \omega \times r$ et $n = \nu \times r$
  - a reste constant
  - sécurité : $\mathcal{O}(2^{\frac{r}{a+1}})$
  - taille des blocs : constante
  - coût du hachage : $\mathcal{N}_{XOR} = \frac{rw}{w\log_2\frac{n}{w}-r} = \mathcal{O}(r)$
  - taille de $\mathcal{H}$: $\mathcal{O}(r^2)$

Bilan

On a construit une famille de fonctions de hachage.

- Relativement rapides : quelques dizaines de Mbits/s.
- Dont la sécurité se réduit à des problèmes NP-complets.
- Faciles à redimensionner.
- Très simples à programmer.
- Qui requièrent beaucoup de mémoire.
- Qui ne peuvent produire que des grands hachés.

Partie V

Conclusion

- Nouvelles applications :
  - signatures très courtes,
  - nouveau système de chiffrement à clef publique,
  - fonctions de hachage rapides avec réduction de sécurité.
- Il reste d'autres problèmes à exploiter...