Master 2 ACC / Paris 8

DM 1 Interaction Codes-Cryptographie

Prof: Borello Martino

26 Octobre 2022

DM 1: (signature) et Programme (Décodage de Patterson et Chiffrement McEliece)

Dans ce DM ils nous est donné d'étudier et d'implémenter une signature basée sur schémas de chiffrement de Niederreiter.
D'abord nous allons commencé par rappeler l'algorithme du cryptosystéme de Niederreiter.

Algorithm 1: Cryptosystéme de Niederreiter

1. **Génération de clé:**

   - On considérons un code $\mathcal{C}[n,k]$ sur $\mathbb{F}_q$ ayant un algorithme de décodage γ.
   - On construit une matrice de contrôle de parité $H'(n-k) \times n$ du code $\mathcal{C}$.
   - On choisi de manière aléatoire une matrice inversible $Q(n-k) \times (n-k)$ sur $\mathbb{F}_q$.
   - On choisi de manière aléatoire une matrice de permutation $P(n) \times (n)$ sur $\mathbb{F}_q$.

   • Clés public: $H = QH'P$ et t la capacité de correction.

   • Clés privé: (P, H', Q, γ)

2. **Chiffrement:**
   input : Message $x \in \mathbb{F}_q^n$ tel que $wt(x) = t$, clés public H
   output: Le chiffré $y \in \mathbb{F}_q^{n-k}$

   - On calcule $y = Hx^T$.

3. **Déchiffrement:**
   input : Chiffré $y \in \mathbb{F}_q^(n-k)$, clés privé (P, H', Q, γ)
   output: Le message $x \in \mathbb{F}_q^n$

   - On calcul $Q^{-1}y = H'Px^T$
   - On retrouve Px^T avec l'algorithme de décodage γ
   - On retrouve x en calculant $P^{-1}(Px^T) = x^T$.

Le cryptosystéme Niederreiter utilise un syndrome comme texte chiffré, et le message est un vecteur d'erreur au lieu d'un mot de code.

Pour transformer un schémas de chiffrement a clés public en schémas de signature il suffit de prendre la fonction de déchiffrement avec la clés privée comme fonction de signature et la fonction de chiffrement avec la clés privée comme fonction de vérification. Cependant dans le cas de de Niederreiter la conversion ne s'applique pas aussi simplement, car contrairement aux schémas comme RSA, le schémas de Niederreiter n'est pas inversible. La raison est que si l'on considère un syndrome aléatoire, il correspond à un vecteur d'erreur de poids supérieur à t.

coût de la signature	$t!t^2m^3$	$237 = 12 \times 10^{11}$ opération
longueur de la signature	$(t-1) \times m + log_2 t$	131 bits
coût de la vérification	t^2m	1296 opération
taille de la clé publique	$tm2^m$	9 Mbits
coût de la meilleure attaque publiée	$2^{tm}(\frac{1}{2} + \mathcal{O}(1))$	$\approx 2^{80}$

Problème:
Si nous prenons $y \in F_2^n$, choisit de manière aléatoire et $\mathcal{C}[n,k]$ définit sur $\mathbb{F}_2$ dont nous sommes capables de décoder t erreurs, la probabilité que y soit décodable est très faible.

Solution:
On va utiliser une fonction de hachage h qui retourne des valeurs dans $\mathbb{F}_2^{n-k}$. Et plus important encore on a besoin d'une famille de codes denses ayant un algorithme de décodage efficace et sûre face aux attaques structurelles : **codes de Goppa** binaires de paramètres $[2^m, 2^m - tm, 2t+1]$ sur $\mathbb{F}_2$.
Pour t petit, la probabilité pour un élément aléatoire d'être décodable (dans une boule de rayon t centrée sur les mots du code) est à peu près $\frac{1}{t!}$

Paramètres:
En prenant $n = 2^m, m = 16, t = 9$, on a une 1 chance sur $9! = 362880$ d'avoir un mot décodable.

- inconvénients:
  on doit décoder plusieurs mots ($t!$) avant de trouver un mot qui convient: 70 fois plus lent que RSA.
  t faible implique des paramètres très importants: clé publique de 9 Mbits.
- avantages:
  meilleure attaque: attaque par ensemble d'information; schéma de signature non basé sur un problème de la théorie des nombres.

Avec les paramètres choisis nous avons une probabilité de $\frac{1}{9!}$ pour décoder chaque syndrôme. Il va donc falloir essayer de décoder environ 9! syndromes aléatoires. Pour ce faire, nous allons simplement utiliser un compteur i et le hacher avec le message M: le haché du syndrome obtenu dépendra alors de i, en changeant i nous pouvons avoir plusieurs syndromes possibles. L'algorithme du schéma de signature CFS est donné ci-dessous.

KANE Seyni
kaneseyni.sn@gmail.com

Algorithm 2: Schéma de signature CFS

1. **Génération de clé:**
   Cette phase se fait comme dans l'algorithme 1

2. **Signature:**
   input : Message M, clés privé (P, H', Q, γ)

   output: Signature (i_0, x), i_0 avec valeur du compteur et $x \in \mathbb{F}_2^n$ un mot tels que $Hx^T = h(M\|i_0)$

$i \leftarrow 0$;
while $\gamma(Q^{-1}h(M||i)) == \tau$ **do**
  $i \leftarrow i + 1$;
$x' \leftarrow \gamma(Q^{-1}h(M||i))$;
$x \leftarrow P^{-1}x'$

3. **Vérification:**
   input : Message M, signature (i_0, x), clés public H

   output: $\widehat{b} \in \{0, 1\}$: signature valide si $\widehat{b} = 1$, invalide sinon

On calcule $s' = Hx^T$ et $s = h(h(M)\|i_0)$;
if $s' == s$ **then**
  $\widehat{b} \leftarrow 1$
else
  $\widehat{b} \leftarrow 0$

References :

1 How to achieve a McEliece-based Digital Signature Scheme; Nicolas Courtois et al.

2 Post-quantum Cryptography: Code-Based Signatures; Pierre-Louis Cayrel et al.

3 Polycopié Borello Martino https://www.math.univ-paris13.fr/~borello/interactionscodescrypto/20222023/interactions-codes-crypto.html

4 Polycopié Pierre-Louis CAYREL http://cayrel.net/-code-based-cryptography-