ACCQ207
Morceaux choisis de cryptographie mathématique
Fonctions booléennes

Adèle Mortier

Juin 2015

1 Introduction

Principe de base de la cryptographie à clef:

$$\xrightarrow{\text{message clair}} \text{chiffrement} \xrightarrow{\text{message chiffré}} \text{déchiffrement} \xrightarrow{\text{message clair}}$$

K_D : clef de déchiffrement (doit être secrète)
K_C : clef de chiffrement (secrète en cryptographie symétrique, publique en cryptographie asymétrique)

Remarque 1. *On peut passer de K_D à K_C par un algorithme polynomial. En revanche, le passage de K_C à K_D est un problème difficile.*

1.1 Cryptographie asymétrique

Aussi appelée cryptographie à clef publique. Pour envoyer son message, Alice recherche la clef publique de Bob dans un annuaire (public lui aussi). Le message est ensuite déchiffré par Bob au moyen d'un clef secrète qu'il possède.

$$\begin{tikzcd}
\text{Alice} & & \text{Bob} \\
m \arrow[r, "K_{PB}"] & c \arrow[r, "K_{SB}"] & m \\
& \text{Annuaire public} \arrow[ul, "\text{clef publique de Bob}"] &
\end{tikzcd}$$

1.2 Cryptographie symétrique

Aussi appelée cryptographie à clef secrète. Au cours d'une première communication à très haut niveau de sécurité, Alice et Bob échangent leurs clefs respectives. Puis Alice peut poursuivre la communication en encodant avec sa clef de chiffrement K_{CA}, pour que Bob déchiffre avec la clef de déchiffrement K_{DA} qu'il a obtenue au cours de la première communication. Et vice-versa.

$$\begin{tikzcd}
\text{Alice} \arrow[r, leftrightarrow, "\text{échange de clefs (communication de haute sécurité)}"] & \text{Bob}
\end{tikzcd}$$

$$\begin{tikzcd}
\text{Alice} & & \text{Bob} \\
\text{m} \arrow[r, "K_{CA}"] & \text{c} \arrow[r, "K_{DA}"] & \text{m} \\
\text{m'} & \text{c'} \arrow[l, "K_{DB}"'] & \text{m'} \arrow[l, "K_{CB}"']
\end{tikzcd}$$

2 Cryptographie symétrique"moderne"

Elle comprend la cryptographie par flot (à la volée), et la cryptographie par blocs. Le cryptographie par flot traite des messages sur $0, 1$ (ou encore, $\mathbb{F}_2$). Elle se base sur le chiffrement de Vermann.

2.1 Chiffrement de Vermann

Il consiste en la somme binaire du message à transmettre avec une clef. Il comprte cependant deux inconvénients: la cef doit être ausi longue que le message, et elle doit changer à chaque chiffrement (car il est aisé, possédant un unique couple message/mot code, de la retrouver).

2.2 Version améliorée

L'idée est ici d'utiliser un clef de "petite taille" K (typiquement, 80 bit ou plus), pour générer une clef "plus longue" $\tilde{K}$ à l'aide d'un générateur pseudo aléatoire. La clef ainsi obtenue doit satisfaire un certain nombre de critères: - $\tilde{K}$ doit être de grande période (la taille minimale du motif qui se répète dans la clef doit être grande) - $\tilde{K}$ doit être équilibrée (autant de "0" que de "1") - $\tilde{K}$ doit être reproductible à l'aide de K - $\tilde{K}$ doit être non prédictible (étant donné le début, il doit être impossible de déduire le bit suivant) - $\tilde{K}$ doit posséder de bonne propriété statistiques Quand au générateur pseudo aléatoire, il se base sur la structure LFSR. Si le LFSR compte L registres, il existe exactement $2^L - 1$ états possibles des registres, non identiquement nuls. La période de la clef $\tilde{K}$

sera grande si le polynôme choisi pour les coefficients du LFSR est primitif. Mais cette technique pour simuler un générateur aléatoire est aujourd'hui caduque.

2.3 Versions améliorées de la version améliorée

On pourrai penser à utiliser un grand nombre de LFSR en parallèle, pour ensuite sommer ou multiplier les différentes clefs obtenues. Mais celaposedes problèmes de complexité linéaire (en cas de sommation), ou de sortie trop *sparse* (en cas de multiplication). Une autre idée, plus fructueuse, est appelée modèle combiné. Il s'agit cette fois encore de faire marcher en parallèle plusieurs LFSR, mais de leur appliquer en sortie une fonction booléenne, du type:

$$\begin{array}{rcl} f : \mathbb{F}_2^n & \longleftarrow & \mathbb{F}_2 \\ (x_1, ..., x_n) & \mapsto & f(x_1, ..., x_n) \end{array}$$

Ou bien, on peut aussi user d'un modèle dit filtré, qui consiste cette fois à ne faire fonctionner qu'un seul LFSR, et à ensuite appliquer la fonction booléenne à chaque registre.

Rem.Le deux modèles sont équivalents, mais les attaques ne fonctionnent pas avec la même complexité! La sécurité du système repose sur le choix de la fonction booléenne, qui nous le verrons, doit vérifier certains critères cryptographiques essentiels.

2.4 Fonctions booléennes

Définitions 1. *On appelle* ***support*** *de f et on note supp(f) l'ensemble des éléments de $\mathbb{F}_2^n$ dont l'image par f est "1" (ou "vrai"):*

$$supp(f) = \{(x_1, ..., x_n) \in \mathbb{F}_2^n, f(x_1, ..., x_n) = 1\}$$

On appelle ***poids de Hamming*** *de f et on note $wt(f)$ le cardinal de l'ensemble précédent:*

$$wt(f) = card\,(supp(f))$$

On appelle ***fonction indicatrice*** *de $a \in \mathbb{F}_2^n$, et on note δ_a, l'application:*

$$\begin{array}{rcl} \delta_a : \mathbb{F}_2^n & \longrightarrow & \mathbb{F}_2 \\ x & \mapsto & \left\{ \begin{array}{l} 1 \ si \ x = a \\ 0 \ sinon \end{array} \right. \end{array}$$

L'ensemble des fonctions booléennes sur $\mathbb{F}_2^n$ est noté $\mathbb{B}_n$:

$$\mathbb{B}_n = \{f : \mathbb{F}_2^n \longrightarrow \mathbb{F}_n\} \tag{1}$$

C'est un espace vectoriel de dimension 2^n. Son cardinal est 2^{2^n}.
Avec les notations qui précèdent, toute fonction booléenne f peut s'écrire sous la forme:

$$f : x \mapsto \sum_{a \in \mathbb{F}_2^n} f(a).\delta_a(x) \tag{2}$$

Remarque 2. *On peut montrer que l'ensemble des monômes forment une base de $\mathbb{B}_n$ en tant que $\mathbb{F}_2$-espace vectoriel.*

$$(x_1...x_n) \mapsto x_1^{u_1}...x_n^{u_n}, \text{ où } \forall i \in [1,n], u_i \in \{0,1\}$$

$x_1^{u_1}...x_n^{u_n}$ est noté m_u, son degré vaut $\sum_{i\in[1,n]} u_i = wt(u)$

2.4.1 Représentation en forme normale algébrique

On peut représenter une fonction booléenne de façon univoque, à l'aide de la **forme normale algébrique** (algebraic normal form - ANF): c'est une représentation polynômiale multivariée à coefficients dans $\mathbb{F}_2$. Elle s'écrit:

$$f(x_1...x_n) = \sum_{u=(u_1...u_n)\in\mathbb{F}_2^n} a_u \prod_{i=1}^{n} x_i^{u_i}$$

Où le degré relatif à chacune des variables est au plus 1.

Exemple 1.

$$\begin{aligned} f(x_1, x_2, x_3) &= x_1x_2 + x_1 + x_2x_3x_1 \\ deg(f) &= max\{wt(1,1,0), wt(1,0,0), wt(1,1,1)\} \\ &= max(2,1,3) \\ &= 3 \end{aligned}$$

Remarque 3. *Si $deg(f) = 0$ on dit que f est constante, s'il vaut 1, on dit qu'elle est affine.*

Algorithme 1. *(déduction de l'ANF par le support)*

x_1	x_2	x_3	f			
0	0	0	1	1	1	1
0	0	1	1	1	1	0
0	1	0	0	0	1	1
0	1	1	1	1	0	1
1	0	0	0	1	1	1
1	0	1	1	0	0	1
1	1	0	0	0	1	1
1	1	1	1	0	0	1

$$\begin{aligned} supp(f) &= \{(0,0,0), (0,0,1, (0,1,1), (1,00), (1,1,1)\} \\ f(x_1, x_2, x_3) &= 1 + x_1 + x_2 + x_1x_2 + x_1x_3 + x_2x_3 + x_1x_2x_3 \end{aligned}$$

On écrit d'abord la table de vérité de la fonction booléenne f souhaitée. Puis à chaque itération, on découpe les subdivisions de la colonne précédente en deux parties égales. On recopie dans la colonne courante chaque première partie des subdivisions ainsi obtenues. Et dans chaque seconde partie des subdivisions de la colonne courante, on écrit à la hauteur k la somme binaire du k-ième élément de la première partie et du k-ième élément de la seconde partie (lus dans la colonne précédente). A la fin, on obtient les coefficients de l'ANF de la fonction booléenne f désirée.

2.4.2 Représentation polynômiale

En remarquant que $\mathbb{F}_2^n \simeq \mathbb{F}_{2^n}$ (le corps fini à 2^n éléments), on peut représenter une fonction booléenne sous une forme polynômiale univariée, à coefficients dans un sous-corps de $\mathbb{F}_n$:

$$f(n) = \sum_{i \in \sigma_n} Tr^{\theta(i)}\left(a_i x^i\right) + \epsilon\left(1 + x^{2^n - 1}\right)$$

Avec:
σ_n l'ensemble des représentants des classes cyclotomiques binaires modulo $2^n - 1$.
$\theta(1)$ le cardinal de la classe cyclotomique de i
ϵ le poids deHamming de f modulo 2
Tr^k la trace(absolue) sur $\mathbb{F}_2^n$, relativement au corps $\mathbb{F}_2$:

$$Tr^k : x \mapsto x + x^2 + x^{2^2} + ... + x^{2^{k-1}} \tag{3}$$

Exemple 2. *$n = 4$. Cherchons σ_4. $2^n - 1 = 15$, donc on travaille dans un espace à 15 éléments. On détermine leurs classes cyclotomiques:*

$$\begin{aligned}
\mathcal{C}(0) &= \{0\} \\
\mathcal{C}(1) &= \{1, 2, 4, 8\} = \mathcal{C}(2) = \mathcal{C}(4) = \mathcal{C}(8) \\
\mathcal{C}(3) &= \{3, 6, 12, 9\} = \mathcal{C}(6) = \mathcal{C}(12) = \mathcal{C}(9) \\
\mathcal{C}(5) &= \{5, 10\} = \mathcal{C}(10) \\
\mathcal{C}(7) &= \{7, 14, 11^{**}, 13^{*}\} = \mathcal{C}(7) = \mathcal{C}(14) = \mathcal{C}(11) = \mathcal{C}(13)
\end{aligned}$$

** 13 est obtenu en faisant* $14 \times 2 = 28 = 13[15]$
*** 11 est obtenu en faisant* $13 \times 2 = 26 = 11[15]$
Puis on peut calculer leurs cardinaux:

$$\begin{aligned}
\sigma(0) &= 0 \\
\sigma(1) &= 4 \\
\sigma(3) &= 4 \\
\sigma(5) &= 2 \\
\sigma(7) &= 4
\end{aligned}$$

D'où $\sigma_4 = \{0,1,3,5,7\}$

$$\begin{array}{rcl} f & : & \mathbb{F}_16 \longrightarrow \mathbb{F}_2 \\ f(x) & = & Tr^{\sigma(1)}(a_1x^1) + Tr^{\sigma(3)}(a_3x^3) + Tr^{\sigma(5)}(a_1x^5) + Tr^{\sigma(7)}(a_1x^7) + \epsilon(1+x^{15}) \\ & = & Tr^4(a_1x) + Tr^4(a_3x^3) + Tr^2(a_1x^5) + Tr^4(a_1x^7) + \epsilon(1+x^{15}) \end{array}$$

2.4.3 La transformée de Fourier discrète

Soit $f : \mathbb{F}_2^n \longrightarrow \mathbb{F}_2$. La transformée de Fourier discrète de f est définie par:

$$f \mapsto \hat{f} \text{ t.q } \hat{f} : a \mapsto \sum_{x \in \mathbb{F}_2^n} f(x)(-1)^{a.x}$$

Où $a.x$ désigne le produit scalaire canonique dans $\mathbb{F}_2^n$ ($\sum_{i=1}^n a_i x_i$)

Algorithme 2. *(Transformée de Walsh à partir de la table de vérité)*

x_1	x_2	x_3	f	$(-1)^f$			
0	0	0	1	1	0	0	-2
0	0	1	1	1	-2	-2	2
0	1	0	0	0	0	0	-2
0	1	1	1	1	0	-2	2
1	0	0	0	1	-2	0	-2
1	0	1	1	0	0	-2	2
1	1	0	1	0	2	-4	-2
1	1	1	0	0	-2	2	-6

A écrit d'abord la table de vérité de f, puis la quantité $(-1)^f$[2], qui vaut -1 quand f vaut 1 et 0 quand f vaut 0. Puis à chaque étape, on redécoupe le tableau comme dans le premier algorithme. Cette fois, on met dans la partie supérieure la somme binaire des élément des parties supérieure et inférieure pris deux à deux. Dans la partie inférieure, on fait de même mais en effectuant une différence des éléments supérieurs par les élément inférieurs. La dernière colonne correspond aux coefficients de la transformée de Walsh de f.