Master CSI 2 2013-2014

Cryptanalyse — M1MA9W06

Responsable : G. Castagnos

Projet sur la cryptanalyse algébrique de A5/2

À faire en binôme. Vos programmes clairs et bien commentés ainsi qu'un rapport contenant les réponses aux questions (démonstrations, explications du code Sage) sont à rendre avant le vendredi 6 décembre 23:59 par mail à `guilhem.castagnos@math.u-bordeaux1.fr`

Le projet consiste en une cryptanalyse de l'algorithme de chiffrement par flot A5/2 en s'inspirant de la méthode proposée par Barkan, Biham et Keller en 2003 [1].

A5/2 est constitué de 4 LFSR, notés LFSR_1, LFSR_2, LFSR_3 et LFSR_4, de longueurs respectives $19, 22, 23$ et 17 et de polynômes de rétroactions respectifs $\mathrm{P}_1 = x^{19} + x^{18} + x^{17} + x^{14} + 1$, $\mathrm{P}_2 = x^{22} + x^{21} + 1$, $\mathrm{P}_3 = x^{23} + x^{22} + x^{21} + x^8 + 1$ et $\mathrm{P}_4 = x^{17} + x^{12} + 1$. Il utilise une clef $\mathrm{K} = (\mathrm{K}_0, \ldots, \mathrm{K}_{63})$ de 64 bits et un $\mathrm{IV} = (\mathrm{IV}_0, \ldots, \mathrm{IV}_{21})$ de 22 bits et il produit une suite chiffrante z de longueur fixe $\mathrm{N} = 228$ bits. On notera à chaque instant $\mathrm{R}_1 = (\mathrm{R}_{1,0}, \ldots, \mathrm{R}_{1,18}), \mathrm{R}_2 = (\mathrm{R}_{2,0}, \ldots, \mathrm{R}_{1,21}), \mathrm{R}_3 = (\mathrm{R}_{3,0}, \ldots, \mathrm{R}_{3,22})$ et $\mathrm{R}_4 = (\mathrm{R}_{4,0}, \ldots, \mathrm{R}_{4,16})$, les registres des 4 LFSR.

La phase d'initialisation, $\mathsf{A5/2-init}$, est décrite dans l'encadré suivant :

> Mettre à zéro les registres $\mathrm{R}_1, \mathrm{R}_2, \mathrm{R}_3, \mathrm{R}_4$
>
> Pour i allant de 0 à 63,
>
> Mettre à jour les LFSR_1, LFSR_2, LFSR_3, LFSR_4 en ignorant leurs sorties
>
> $\mathrm{R}_{1,18} = \mathrm{R}_{1,18} + \mathrm{K}_i, \mathrm{R}_{2,21} = \mathrm{R}_{2,21} + \mathrm{K}_i, \mathrm{R}_{3,22} = \mathrm{R}_{3,22} + \mathrm{K}_i, \mathrm{R}_{4,16} = \mathrm{R}_{4,16} + \mathrm{K}_i$
>
> Pour i allant de 0 à 21,
>
> Mettre à jour les LFSR_1, LFSR_2, LFSR_3, LFSR_4 en ignorant leurs sorties
>
> $\mathrm{R}_{1,18} = \mathrm{R}_{1,18} + \mathrm{IV}_i, \mathrm{R}_{2,21} = \mathrm{R}_{2,21} + \mathrm{IV}_i, \mathrm{R}_{3,22} = \mathrm{R}_{3,22} + \mathrm{IV}_i, \mathrm{R}_{4,16} = \mathrm{R}_{4,16} + \mathrm{IV}_i$
>
> On fixe à 1 certains bits des registres : $\mathrm{R}_{1,3} = 1, \mathrm{R}_{2,5} = 1, \mathrm{R}_{3,4} = 1, \mathrm{R}_{4,6} = 1$.

On note $\mathsf{Maj}(a, b, c)$ la fonction majorité. On rappelle que $\mathsf{Maj}(a, b, c) = 0$ si et seulement si la majorité des bits a, b et c vaut 0, par exemple $\mathsf{Maj}(0, 1, 0) = 0$ et $\mathsf{Maj}(1, 1, 0) = 1$.

A5/2 utilise la fonction de mise à jour $\mathsf{A5/2-step}$ décrite dans l'encadré suivant :

1. *Instant Ciphertext-Only Cryptanalysis of GSM encrypted communication*, Elad Barkan , Eli Biham , Nathan Keller, CRYPTO 2003

Calculer $m = \mathsf{Maj}(\mathrm{R}_{4,6}, \mathrm{R}_{4,13}, \mathrm{R}_{4,9})$
Si $\mathrm{R}_{4,6} = m$, le LFSR_1 est mis à jour en ignorant sa sortie
Si $\mathrm{R}_{4,13} = m$, le LFSR_2 est mis à jour en ignorant sa sortie
Si $\mathrm{R}_{4,9} = m$, le LFSR_3 est mis à jour en ignorant sa sortie
Le LFSR_4 est mis à jour en ignorant sa sortie
Calculer $y_1 = \mathrm{R}_{1,0} + \mathsf{Maj}(\mathrm{R}_{1,3}, \mathrm{R}_{1,4} + 1, \mathrm{R}_{1,6})$
Calculer $y_2 = \mathrm{R}_{2,0} + \mathsf{Maj}(\mathrm{R}_{2,8}, \mathrm{R}_{2,5} + 1, \mathrm{R}_{2,12})$
Calculer $y_3 = \mathrm{R}_{3,0} + \mathsf{Maj}(\mathrm{R}_{3,4}, \mathrm{R}_{3,9} + 1, \mathrm{R}_{3,6})$
Le bit de sortie de $\mathsf{A5/2 - step}$ est $y_1 + y_2 + y_3$

Après l'exécution phase d'initialisation, $\mathsf{A5/2 - init}$, la production de $\mathrm{N} = 228$ bits de suite chiffrante, se fait de la manière suivante :

— Exécuter 99 fois la fonction $\mathsf{A5/2 - step}$ en ignorant son bit de sortie
— Exécuter $\mathrm{N} = 228$ fois la fonction $\mathsf{A5/2 - step}$, en utilisant ses bits de sortie pour produire la suite chiffrante, z.

1 Programmer (avec Sage) le chiffrement A5/2. Il sera utile pour la suite de diviser le code en plusieurs fonctions (initialisation, mise à jour de l'état, production de suite chiffrante). Pour vérifier votre code, on pourra trouver dans

`http://www.math.u-bordeaux1.fr/~gcastagn/Cryptanalyse/A5_2-test-vector.sage`

un exemple de suite chiffrante créée par A5/2.

2 On se place juste après la phase d'initialisation. On suppose connu le registre R_4 du LFSR_4. On pose $\mathrm{R}_1 = (x_0, \ldots, x_{18}), \mathrm{R}_2 = (x_{19}, \ldots, x_{40})$ et $\mathrm{R}_3 = (x_{41}, \ldots, x_{63})$ où les x_i sont des inconnues (sauf 3 x_i que l'on sait être égaux à 1). Montrer que durant toutes les étapes de la production de suite chiffrante de A5/2, on peut exprimer les contenus des registres $\mathrm{R}_1, \mathrm{R}_2, \mathrm{R}_3$ au moyen d'équations linéaires en les x_i.

3 Écrire un code Sage permettant d'exprimer ces équations linéaires. Pour cela, déclarer les inconnues par `BPR = BooleanPolynomialRing(64,'x'); v = BPR.gens()` et utiliser les matrices de rétroaction des LFSR.

4 Donner la forme algébrique normale de la fonction booléenne Maj. En déduire que les bits de suite chiffrante z peuvent s'exprimer par des équations quadratiques en les x_i.

5 Écrire un code Sage permettant d'exprimer ces N équations quadratiques.

6 Montrer qu'au plus 655 monômes de degré au plus deux peuvent apparaître dans ces équations (sachant que l'on connaît la valeur de trois x_i). Créer avec Sage la liste M de ces monômes. On note $\mathrm{L} = 655$ la longueur de M.