ALGORITHME SPECIFIQUE DE CRYPTANALYSE

[1]Naima HADJ-SAID – [1]Adda. ALI-PACHA– [1]Aicha TEKKOUK – [2]Maachou BACHIR RAHO

[1]Université des Sciences et de la Technologie d'Oran –USTO, BP 1505 El M'Naouer Oran 31036 ALGERIE
[2] Université d'Oran Es-Senia – BP 1524 El M'Naouer Oran 31036 ALGERIE

Tél. /Fax : 213 – 041 / 46 26 85
E.Mail : nim_hadj@yahoo.fr; alipacha@yahoo.com

Résume :

La cryptographie symétrique et en particulier le chiffrement continu à la base des registres à décalage à rétroaction linéaire à pour principe de générer une suite pseudo aléatoire qui est additionnée modulo 2 avec le texte en clair pour donner un texte chiffré.

Caser ce système de chiffrement c'est retrouve la clé de chiffrement qui est matérialisée par le polynôme générateur et l'état initial des registres (RDRL) afin de prévoir complètement la suite pseudo aléatoire. Ceci est réalisé par l'algorithme de Berlekamp-Massey qui permet de déterminer la complexité linéaire d'une suite finie (dans notre cas c'est des cryptogrammes), ainsi que le polynôme de rétroaction d'un RDRL de longueur minimale engendrant la même suite pseudo aléatoire qui a été utilisé lors du chiffrement.

Meilleures sont les résultats, si nos cryptogrammes sont présentes à l'algorithme sous formes d'échantillons chevauchés c'est-à-dire que deux échantillons consécutifs ont en commun un certains nombre de bits de données (les derniers bits d'un échantillon sont les premier de l'autre).

Mots Clés : *Cryptographie, cryptanalyse, chiffrement à flot, polynôme primitif, algorithme Massey Berlekamp.*

1. Introduction :

La cryptographie, est l'étude des méthodes permettant de transmettre des données de manière confidentielle. Afin de protéger un message, on lui applique une transformation spéciale qui le rend incompréhensible ; c'est ce qu'on appelle le *chiffrement* [7], qui, à partir d'un texte en clair donne un texte chiffré ou cryptogramme, ceci a pour but de développer des procédés :

- ✓ Préserver la confidentialité des messages.
- ✓ Vérifier l'intégrité des messages.

Qui permettent à deux personnes de communiquer tout en protégeant leurs messages.

De nombreux systèmes de chiffrement ont été inventés et qu'on peut les classer en deux catégories : système à clé publique et système clé privée [4, 5]. L'algorithme à clé privée (figure 1), aussi appelé algorithme *symétrique,* est caractérise par le fait qu'on utilise la même clé pour chiffrer ET déchiffrer.

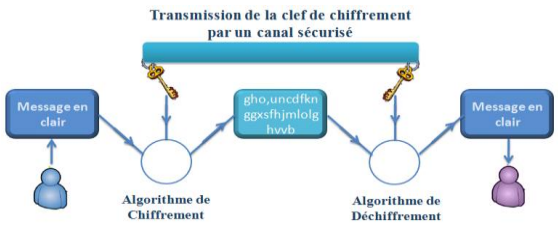

Figure 1. : Chiffrement symétrique

Deux types de chiffrements à clé secrète se manifestent :

- ***Le chiffrement par bloc :*** Les chiffrements par bloc traitent les données par morceaux.
- ***Le chiffrement par flot :*** Le chiffrement à flot (en continu) permet de chiffrer bit par bit. La fonction de chiffrement varie au fur et à mesure de son application au message en clair.

On s'intéresse particulièrement dans ce travail, au chiffrement à flot à la base des générateurs qui sont constitués par des registres à décalage à rétroaction linéaire RDRL [8] usuellement désigné par l'abréviation anglo-saxonne LFSR (*Linear Feedback Shift Register*) produisant un flux pseudo aléatoire (suite chiffrante). Ce flux aléatoire sera combiné avec l'information à chiffrer par l'opération XOR pour donner les cryptogrammes. A la réception le même flux est généré puis combiné avec le flux reçu pour reconstituer l'information en claire.

Les LFSR sont des dispositifs extrêmement rapides et d'implémentation peu coûteuse pour engendrer des suites ayant de bonnes qualités statistiques, notamment une période élevée. Même si la période est très élevée le générateur sera toujours périodique, donc prévisible, de ce fait, il est évidemment impossible d'utiliser la suite produite par un LFSR comme suite chiffrante dans un chiffrement à flot.

En effet, si les coefficients du LFSR sont publics (ce qui est généralement le cas quand le LFSR est implémenté sous forme d'un circuit électronique), il suffit à un attaquant qui connaît L bits consécutifs de la suite d'appliquer la relation de récurrence pour retrouver tous les bits suivants.

Dans le cas où les coefficients de rétroaction sont secrets, l'algorithme de Berlekamp-Massey permet de les reconstituer, ainsi que l'état initial, à partir de 2L bits de suite chiffrante.

2. Chiffrement Continu à la base des générateurs pseudos aléatoires:

2.1 Registres à Décalage à Rétroaction Linéaire :

Un LFSR est composé (figure 2) de deux éléments un registre à décalage, et une fonction de rétroaction P satisfaisante une relation de récurrence linéaire.

Il permet d'engendrer des suites (séquences) binaires possédant (sous certaines conditions) une période élevée et de bonnes propriétés statistiques.

Un LFSR binaire de longueur L est composé d'un registre à L cellules (cases mémoires) contenant chacune un bit, et d'une horloge contrôlant le mouvement des données (ce mouvement a lieu de la gauche vers la droite figure 2).

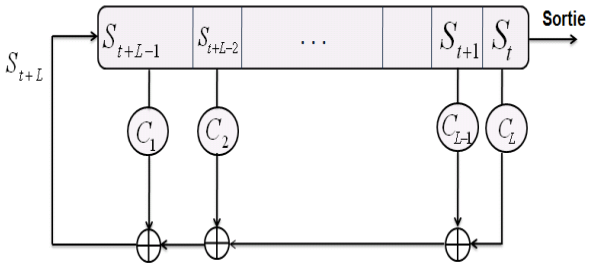

Figure 2 : Registre à décalage à rétroaction linéaire

Les coefficients C_i peuvent être nuls ou égalent à '1' selon le polynôme choisi; mais ils ne sont pas tous nuls. Les L bits contenus dans le registre forment l'état interne du LFSR. Ces L cellules sont initialisées par L bits S_0, ..., S_{L-1} placés de droite à gauche; il y a 2^L possibilités d'initialisation (la clé). Notons que si la suite initiale S_0, ..., S_{L-1} est nulle, alors toute la suite est nulle.

Ce registre à décalage est contrôlé par une horloge externe. Au cours de chaque unité de temps, chaque bit est décalé d'une cellule vers la droite. Le contenu de la cellule la plus à droite, S_t, sort du registre, alors que la cellule la plus à gauche reçoit le bit de rétroaction, s_{t+L}. La valeur de ce dernier est obtenue par une combinaison linéaire des valeurs des autres cellules, dont les coefficients sont des éléments fixés qui valent 0 ou 1 et appelés *coefficients de rétroaction du LFSR* :

$$S_{t+L} = \sum_{i=1}^{L} C_i S_{t+L-i} \quad pour tout\ t \geq 0$$

Où la somme est une somme modulo 2 dans le cas d'un LFSR binaire.

La suite pseudo aléatoire (s_n) $n \geq 0$ produite par un LFSR de longueur L est constituée des bits de sortie successifs du LFSR ; donc cette suite a une récurrence linéaire homogène d'ordre L.

$$sj = c_1 s_{j-1} \oplus c_2 s_{j-2} \oplus \ldots \oplus c_{L-1} s_{j-(L-1)}$$
$$\ldots$$
$$\oplus c_L s_{j-L} \quad \text{avec } j>0$$

Toute suite binaire à récurrence linéaire homogène d'ordre L est ultimement périodique, et sa plus petite période T est inférieure ou égale à 2^L-1.Si sa période égal (2^L-1) elle est appelée suite de longueur maximum. Le polynôme caractéristique associé à une suite de longueur maximum [2] est appelé polynôme primitif.

Le polynôme de rétroaction est dit irréductible, si la suite engendrée par le LFSR quelle que soit son initialisation (à part l'état nul) ne peut pas être engendrée par un LFSR plus court. Un polynôme est dit irréductible s'il ne peut s'écrire comme produit de deux polynôme (on ne peut pas le factorise).

Ainsi, le polynôme de rétroaction *du LFSR*, est défini par :

$$P(x) = 1 + \sum_{i=1}^{L} C_i \ X^i$$

Ces polynômes donnent des périodes maximales.

2.2 Exemple de chiffrement

Un agent nommé A veut envoyer un message secret à son copain B en

utilisant la clé produite par un LFSR. Dans cet exemple, nous considérons le registre de polynôme de rétroaction $P(X)=x^4+x^3+1$ ce polynôme est irréductible primitif (figure 3).

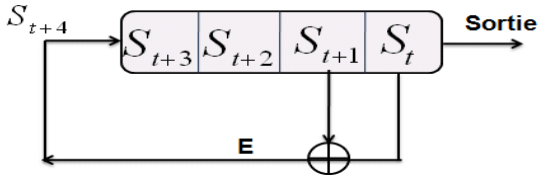

Figure 3 : LFSR ($P(X) =x^4 +x^3 +1$)

La première connexion du registre (à gauche) correspond au terme x, la dernière connexion (à droite) correspond au terme x^4. Le terme 1, toujours présent dans un polynôme primitif, n'est pas associé à une connexion [3].

Le cryptage dépend non seulement de l'algorithme, représenté par le câblage de la contre réaction, mais aussi de l'état initial du registre à décalage. On Prend la situation de départ suivante:

$$(S_0, S_1, S_2, S_3) = (1, 0, 1, 1).$$

A chaque top d'horloge le bit introduit par décalage dans le registre (à gauche) est déterminé par la relation de récurrence :

$S_{t+4}= S_{t+1}+S_t$.

C'est un LFSR à 4 bits dérivé au troisième et en quatrième bit, la suite pseudo aléatoire produite par ce dernier est : 101111000100110

(Table 1).

t	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S_t	1	0	1	1	1	1	0	0	0	1	0	0	1	1	0	1
S_{t+1}	0	1	1	1	1	0	0	0	1	0	0	1	1	0	1	0
S_{t+2}	1	1	1	1	0	0	0	1	0	0	1	1	0	1	0	1
S_{t+3}	1	1	1	0	0	0	1	0	0	1	1	0	1	0	1	1
E	1	1	0	0	0	1	0	0	1	1	0	1	0	1	1	1

Table 1 : États successifs du LFSR de la figure 3

2.2.1 Chiffrement d'un texte :

Soit le message en clair suivant « RENDEZ VOUS DEMAIN MATIN » à chiffrer par le système généré par la figure 3.

Ce message sera transformé en code ASCII comme suit :

Message clair en binaire :

01010010010001010100111001000100010001010101101000100000010101100100111101010101010100110010000001000100010100110101000001010010010100001001101000010000001001101010000010101000100100101001110001000000.

Suite Chiffrante (Clé) :

10111100010011010111100010011010111100010011010111100010011010111100010011010111100010011010111100010011010111100010011010111.

Message crypté en binaire :

1110111000001000001101101101111010
1101000110111111000010001111011000
1011100000101101101010001111010101
1100011011011010111111110100000100
0011011010111010101111000111010010
11011000100010100010101111 0111

Maintenant on a le message crypté mais sous forme binaire on doit le convertir pour trouver le message crypté et l'envoyer à B. Donc le message crypté sera :

Message crypté : BSd6o$M^USoNUL9N{CFS.

2.2.2 Chiffrement d'une image :

Pour chiffrer une image, on prend par exemple une image satellitaire représentant la partie nord de la ville d'Oran ALGERIE figure 4 (d'une taille 400*400 pixels) et un LFSR basé sur polynôme (x^3+x+1) qui est irréductible et primitif.

On remarque que l'image cryptée est brouillée.

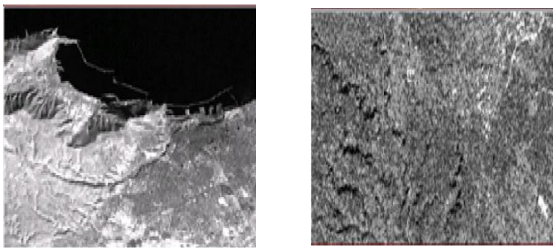

Image claire Image cryptée

Figure 4 : la ville d'Oran

3. Cryptanalyse:

La cryptanalyse s'oppose, en quelque sorte, à la cryptographie. En effet, si déchiffrer consiste à retrouver le texte clair au moyen d'une clé, cryptanalyser c'est tenter de se passer de cette dernière (trouver le clair sans connaissance de la clé). La cryptanalyse est tout simplement l'art de rendre clair un texte crypté sans avoir connaissance de la clef utilisée. Briser un chiffrement revient à trouver une faiblesse dans le chiffrement qui peut être exploitée avec une complexité inférieure à une attaque de force brute.

Il faut distinguer les termes "Secret" et "Robustesse" d'un algorithme. Le secret de l'algorithme revient à cacher les concepts de celui-ci, ainsi que les méthodes utilisées (fonctions mathématiques). La robustesse quant à elle désigne la résistance de l'algorithme à diverses attaques. Il existe plusieurs familles d'attaques cryptanalytiques [6, 9], les plus connues étant l'analyse fréquentielle, la cryptanalyse différentielle et la cryptanalyse linéaire.

- ✓ **L'analyse fréquentielle :** L'analyse fréquentielle, découvert au IXe siècle par Al-Kindi, examine les répétitions des lettres du message chiffré afin de trouver la clé. Elle est inefficace contre les chiffrements modernes. Elle est principalement utilisée contre les chiffrements

mono-alphabétiques qui substituent chaque lettre par une autre et qui présentent un biais statistique.

- ✓ **Cryptanalyse différentielle :** La cryptanalyse différentielle est une analyse statistique des changements dans la structure de la méthode de chiffrement après avoir légèrement modifié les entrées, elle consiste à étudier la manière dont les différences entre les données en entrée affectent les différences de leurs sorties. Elle montre un comportement prédictible et elle exploite ainsi ces propriétés afin de retrouver la clé secrète. **C**ette attaque date de 1990 (présentée à la conférence*Crypto 90*). Elle est due à Eli Biham et Adi Shamir ;
- ✓ **Cryptanalyse linéaire :** La cryptanalyse linéaire, due à Mitsuru Matsui, consiste à faire une approximation linéaire de la structure interne de la méthode de chiffrement. Elle remonte à 1993 et s'avère être l'attaque la plus efficace sur DES (Data Encryption Standard). Les algorithmes plus récents sont insensibles à cette attaque.

Il existe plusieurs sortes d'attaques qu'on va les énumérés par ordre décroissant de leur difficulté pour une cryptanalyse.

- ➢ ***Attaque exhaustive :*** Une attaque *exhaustive* essaie toutes les fonctions de décryptage possible en parcourant l'espace des clefs, jusqu'à ce qu'on ressemble à un texte clair. Le temps moyen de cette attaque est égal au temps d'un déchiffrement multiplié par la moitié de la taille de l'espace des clés.
- ➢ ***Texte chiffré connu*** (*ciphertext-only*) **:** où l'attaquant connaît plusieurs messages cryptés ((*e k)* (*m i*)) . il cherche le clair et si possible la clé; il peut faire des hypothèses sur les messages originaux qu'il ne possède pas. La cryptanalyse est plus ardue de par le manque d'informations à disposition.
- ➢ ***Texte clair connu*** (*known-plaintext attack*)**:** où l'attaquant connaît plusieurs couples message en clair – message crypté : (*mi, ek* (*m i*)) et il cherche la clé, la cryptanalyse linéaire fait partie de cette catégorie.
- ➢ ***Texte clair choisi statique ou adaptative*** (*chosen-plaintext attack*) **:** où l'attaquant peut connaître des couples (*mi, ek* (*m i*)), il peut choisir les clairs de façon que son attaque réussisse avec une meilleure probabilité (elle peut être *adaptative* si les choix peuvent être faits en fonction des résultats des attaques sur les couples précédents si le choix fait au début l'attaque est statique), avec l'algorithme que l'on peut dès lors considérer comme une boîte noire. La cryptanalyse différentielle est un exemple d'attaque à texte clair choisi.

- ***Texte chiffré choisi*** (*chosen-ciphertext attack*)**:** où l'attaquant peut connaître des couples (*ci* , *d k* (*c i*)) de son choix , c'est à dire il possède des messages chiffrés et demande la version en clair de certains de ces messages pour mener l'attaque.

3.1 Attaque réussie d'un crypto système :

Lorsque le cryptanalyste parvient à retrouver soit les messages en clair soit la clé de déchiffrement de manière systématique et effective (sans soudoyer quelqu'un qui possède ses informations) on dit que le système est cassé.

Enfin, les attaques se distinguent selon les aspects du système sur lequel elles portent. On peut attaquer le système en le considérant comme une boîte noire. Cette approche correspond au degré le plus faible de connaissances sur le système. Elle lui reste entièrement extérieure en ne considérant que ses sorties et éventuellement ses entrées. On classe la recherche exhaustive de clé dans cette catégorie. À l'inverse, l'étude détaillée des propriétés internes d'un chiffrement permet de produire des attaques qui exploitent tous les aspects structurels de l'algorithme. Les cryptanalyses les plus connues et dont sont déduits des critères de conception appartiennent à cette catégorie.

On distingue plusieurs niveaux de réussite selon la connaissance que l'on obtient du crypto-système :

- ***Le cassage complet,*** où l'attaquant trouve la clé de déchiffrement *k* .
- ***L'obtention globale,*** où l'attaquant trouve un algorithme équivalent à l'algorithme de déchiffrement, mais qui ne nécessite pas la connaissance de la clef de déchiffrement.
- ***L'obtention locale,*** où l'attaquant trouve le texte en clair *d k* (*c*) correspondant à un message chiffré.
- ***l'obtention d'information,*** où l'attaquant obtient quelques indications sur le texte en clair ou la clef (certains bits de la clef, un renseignement sur la forme du texte en clair,...)

3.2 Attaque sur chiffrements à flot:

La famille de chiffrements à flot conçus à partir de LFSR est certainement celle qui a fait l'objet des études les plus nombreuses. Comme son nom l'indique, le registre à rétroaction linéaire produit une suite binaire que l'on sait relier de manière linéaire à son initialisation.

Un LFSR de polynôme irréductible primitif semble être un bon candidat pour un générateur pseudo-aléatoire. En effet, les suites non nulles qu'il engendre ont une période grande, et une complexité linéaire L bien égale à T. Elles ont de bonnes propriétés statistiques. Malheureusement un tel

générateur n'est pas cryptographiquement sûr; il est évidemment impossible d'utiliser la suite produite par ce LFSR comme une suite chiffrante dans un chiffrement à flot.

En effet, si les coefficients du LFSR sont publics (ce qui est généralement le cas quand le LFSR est implémenté sous forme d'un circuit électronique), il suffit à un attaquant qui connaît L bits consécutifs de la suite d'appliquer la relation de récurrence pour retrouver tous les bits suivants. Dans le cas où les coefficients de rétroaction sont secrets, l'algorithme de Berlekamp-Massey permet de les reconstituer, ainsi que l'état initial, à partir de 2L bits de suite.

La complexité linéaire d'une suite infinie $s=(s_n)_{n \geq 0}$, notée $\Lambda(s)$, est le plus entier Λ tel que s peut être engendrée par un LFSR de longueur Λ; elle est infinie si aucun LFSR ne permet de l'engendrer. Par convention, la complexité linéaire de la suite nulle est égale à 0. La complexité linéaire d'une suite vérifiant une relation de récurrence linéaire correspond au degré de son polynôme minimal.

La complexité linéaire d'une suite finie $s^n = s_0 s_1 \ldots s_{n-1}$ de n éléments, notée $\Lambda(s^n)$, est la longueur du plus petit LFSR qui permette de produire une suite dont les n premiers éléments correspondent à s^n. Pour toute suite finie s^n de longueur n, le LFSR de longueur $\Lambda(s^n)$ qui engendre s^n est unique si et seulement si $n \geq 2\,\Lambda(s^n)$.

La complexité linéaire d'une suite infinie S et celle de la suite finie composée de ses n premiers éléments sont liées par la propriété suivante. Si la suite infinie s est de complexité linéaire Λ, alors s^n est également de complexité linéaire Λ si $n \geq 2\,\Lambda$. Dans ce cas, s^n correspond aux n premiers éléments produit par l'unique LFSR de longueur Λ qui engendre s.

Une notion plus fine est celle de profil de complexité linéaire d'une suite infinie $s=s_0 s_1 \ldots$. On désigne par ce terme la suite $(\Lambda(s^n))_{n \geq 1}$ composée des complexités linéaires de chacune des sous-suites $s^n = s_0 \ldots s_{n-1}$ formées par les n premiers éléments de s.

3.3 Algorithme Massey Berlekamp :

L'algorithme de Berlekamp-Massey [1] permet de déterminer la complexité linéaire d'une suite finie , ainsi que le polynôme de rétroaction d'un LFSR de longueur minimale qui permet de l'engendrer alors on peut trouver l'état initial du registre ainsi qu'on peut prévoir complètement la suite de sortie du LFSR. Cet algorithme est introduit en 1969 par le cryptologue américain James Massey.

L'algorithme de Belekamp-Massey permet de calculer un couple (L, P) associé à une suite donnée, avec $L = \Lambda(s)$ (L minimal). La suite des L bits s'appelle aussi le profil de complexité

de la suite S. Le temps de calcul de cet algorithme est en N^2 avec N =2L.

On défini la notion de profil de complexité d'une suite S, c'est la suite Lk = L(sk). C'est donc, un autre critère pour caractériser une suite pseudo-aléatoire, son profil de complexité devrait être assez proche de celui d'une véritable suite aléatoire, c'est-à-dire $L_k \approx k/2$.

Initialisation :P(X) ←1, P′(X) ←1, Λ ←0, m←−1,

Pour t de 0 à n−1 faire
$d \leftarrow s_t + \sum_{i=1}^{\Lambda} p_i\, s_{t-i}$.
Si d ≠ 0 alors
T(X) ← P(X).
$P(X) \leftarrow P(X) + P'(X)\, X^{t-m}$.
si 2Λ ≤ t alors
Λ ← t + 1 − Λ.
m ← t.
P′(X) ← T(X).
Retourner Λ et P.

Exemple : Soit la suite choisie est issue d'un RDRL de longueur 3; alors il suffit du connaitre 6 bits consécutifs du cryptogramme pour appliquer Massey Berlekamp. Le tableau (table 2) décrit le déroulement de l'algorithme de Berlekamp-Massey appliqué à la suite binaire de longueur 6, $s_0 \ldots s_5$ = 100100. Les valeurs de Λ et P obtenues à la fin de l'itération t correspondent à la complexité linéaire de la suite $s_0 \ldots s_t$ et au polynôme de rétroaction d'un LFSR de taille minimale permettant de l'engendrer.

t	St	d	Λ	P(x)	m	P'(x)
			0	1	-1	1
0	1	1	1	1+x	0	1
1	0	1	1	1	0	1
2	0	0	1	1	0	1
3	1	1	3	$1+x^3$	3	1
4	0	0	3	$1+x^3$	3	1
5	0	0	3	$1+x^3$	3	1

Table 2 : Déroulement de l'algorithme de Berlekamp-Massey appliqué à $s_0 \ldots s_5$ = 100100

T=0→St=0 suivant la relation

$d = d \leftarrow s_t + \sum_{i=1}^{\Lambda} p_i\, s_{t-i} = St = 1$

Puisque d=1 alors T(X) ← P(X)→T(x)=1

Suivant la relation $P(X) \leftarrow P(X) + P'(X)\, X^{t-m}$.

$P(x) = 1 + x^{0-(-1)} = 1 + x^1$

Vérifier la condition 2Λ ≤ t :2*0<=0.

condition vérifiée alors : $\Lambda \leftarrow t + 1 - \Lambda$.

Donc Λ=0+1- 0= 1 ; m ← t. Donc m=0 ;

$P'(X) \leftarrow T(X) = 1$

T=1→St=0 suivant le relation

$d = d \leftarrow s_t + \sum_{i=1}^{\Lambda} p_i\, s_{t-i} = S_t + C_1 S_0 = 0 + 1*1 = 1$

Puisque d=1alors T(X) ← P(X)→ T(x)=1+x

Suivant la relation $P(X) \leftarrow P(X) + P'(X)\, X^{t-m}$.

$P(x) = 1 + x + x^{1-(0)} = 1$

Vérifier la condition 2Λ ≤ t :2*1<=1.

condition non vérifiée

T=2→S_t=0 suivant le relation

$d = d \leftarrow s_t + \sum_{i=1}^{\Lambda} p_i\, s_{t-i} = S_t + C_1 S_1 + C_2 S_0 = 0$

T=3→ainsi de suite…Le LFSR de longueur 4 ayant pour polynôme de rétroaction X^3+1 génère donc la suite donnée.

Pour attaquer un cryptogramme par la méthode de Massey Berlekamp ; il suffit de connaitre n bits consécutifs de ce cryptogramme ; avec n est égale deux fois la complexité du RDRL; mais si on a aucune information sur ce RDRL il est considéré comme une boite noire ; pour cela il faut échantillonner le cryptogramme et chaque échantillon a une taille de n bits.

Si on a un RDRL dont la complexité linéaire L= n; il suffit de connaitre (2*n) bits pour trouver le bon polynôme ; alors il faut échantillonner le cryptogramme avec chaque échantillon a une taille de (2*n) bits; si on diminue la taille ça ne donne pas des bons résultats; si on donne la juste taille; ça nous donne des bons résultats; et si on augmente la taille ça nous donne toujours des bons résultats ; parce que dans le déroulement de Massey Berlekamp après la $n^{ième}$ itération les résultats restent les mêmes.

3.3.1 Décryptage des données:

L'objectif de l'attaquant c'est de trouver la clé de chiffrement associé à un texte chiffré donné, puisque la donnée de la clé fournit la possibilité de calculer tous les textes en clairs associés à tous les textes chiffrés. Dans ce cas, il faut utiliser la *cryptanalyse différentielle*.

On a un message crypté et on aucune information sur le générateur ; il est considéré comme une boîte noire; on a seulement ses sorties et éventuellement et ses entrées.

Grâce à l'algorithme de Massey berlekamp **(**cryptanalyse *différentielle***)** on peut trouver la suite complète et deviner l'état initial.

3.3.2 Exemple de cryptanalyse

Soit un LFSR de longueur 4 de polynôme de rétroaction X^4+X^3+1 ; L=4, et on prend comme état initial $s_0 \ldots s_4$ = 1011 ; La suite produite par ce LFSR est : 101111000100110. Et soit :

Message clair : SALUT

Message crypté : FX!Pù,

On va échantillonner ce cryptogramme en cinq échantillons chacun a une taille de 8 bits (2*L avec L=4); ensuite on va appliquer Massey Berlekamp sur ces échantillons.

Message crypté :

1. 01111000 /
2. 10111100/
3. 01011110/
4. 01100100/
5. 10111100/.

Echantillon 1 : 01111000

t	St	d	Λ	P(x)	m	P'(x)
			0	1	-1	1
0	0	0	0	1	-1	1
1	1	1	0	$1+x^2$	1	1
2	1	1	2	$1+x+x^2$	1	1
3	1	1	2	1+x	1	1
4	1	0	2	1+x	1	1
5	0	1	4	$1+x+x^4$	5	1+x
6	0	0	4	$1+x+x^4$	5	1+x
7	0	0	4	$1+x+x^4$	5	1+x

Donc l'échantillon 1 (01111000) nous donne le polynôme **$p(x)= X^4+X+1$**

Ainsi de suite pour les autres échantillons :

Echantillon 2 : 10111100 nous donne le polynôme **$p(x)= X^4+X^3+1$**

Echantillon 3 : 10111100 → **$p(x)= X^4+X^3+1$**

Echantillon 4 : 01100100 → **$p(x)= X^5+X^2+X+1$**

Echantillon 5 : 10111100 → **$p(x)= X^4+X^3+1$**

Pourcentage du polynôme : Massey Berlekamp trouve trois polynômes par différent pourcentage :

1. $p_1(x)= X^4+X+1$ →1/5= 20% (un seul échantillon parmi 5 échantillons nous donne ce polynôme).
2. $p_2(x)= X^5+X^2+X+1$ →1/5=20% (un seul échantillon parmi 5 échantillons nous donne ce polynôme).
3. $p_3(x)= X^4+X^3+1$ →3/5=60% (trois échantillons parmi 5 échantillons nous donne ce polynôme). On peut le qualifier comme favori (L=4).

Trouver l'état initial :

On a exactement 2^4 suites différentes, satisfaisant la même relation de récurrence, associées aux 2^4 possibilités. Pour décrypter le texte il suffit de tester les 2^4 possibilités afin de trouver un texte clair compréhensive, qu'il est possible qu'il soit le texte chiffré.

1. Pour l'initialisation (s_0, s_1, s_2, s_3)= (0, 1,1 ,1) ; La suite produite par ce LFSR suivant ce polynôme ; S=011110001001101; cette initialisation nous donne le texte décrypte hgyt/
2. Pour l'initialisation (s_0, s_1, s_2, s_3)= (1, 1, 1,0) ; La suite produite par ce LFSR suivant ce polynôme ; S=111000100110101; cette initialisation nous donne le texte décrypte bty_ç
3. Pour l'initialisation (s_0, s_1, s_2, s_3)= (1, 1, 1,1) ; La suite produite par ce LFSR suivant ce polynôme ; S=111100010011010; cette initialisation nous donne le texte décrypte ^ ;:es

Ainsi de suite jusqu'à trouver …

4. Pour l'initialisation (s_0, s_1, s_2, s_3)= (1, 0, 1,1) ; La suite produite par ce LFSR suivant ce polynôme ; S=101111000100110 ; cette initialisation nous donne le texte décrypte SALUT.

4. Résultats et Interprétation :

L'étude a été réalisée [10] sur un micro-ordinateur ayant comme caractéristiques un Micro processeur Intel Pentium 4 (2.4 GHz) et une de RAM 2Go.

Les données seront **c**ryptées par un LFSR dont le polynôme de rétroaction est irréductible primitif $p(x)=x^{10}+x^5+1$ (L=10). Les résultats obtenus seront présentés sous forme de courbe où d'histogramme. La courbe présente : L'application de Massey Berlekamp sur 20 échantillons du cryptogramme et le résultat (c'est-à-dire un polynôme) obtenu pour chaque échantillon. L'axe « *Echantillonnages du cryptogramme* » présente les vingt premiers échantillons d'un cryptogramme ; c'est à dire 1 : échantillon 1 ; 2 : échantillon 2 …ect . L'axe *polynôme de rétroaction* » présente le polynôme trouvé pour chaque échantillon Par contre, l'histogramme présente : le nombre d'apparition d'un polynôme particulier sur le nombre total d'échantillons.

5.1 Tests sur l'Algorithme Massey Berlekamp

On va échantillonner le cryptogramme; avec chaque échantillon a une taille égale à 10 bit (< 2*L) **Figure 5A**; ensuite on va prendre une taille de l'échantillon 20 bit (2*L=20 ; =2*L) **Figure 5B**. Et en fin on va prendre une taille de l'échantillon 32 bit. . (2*L=20 ; >2*L) **Figure 5C**, ensuite on va appliquer la méthode Massey Berlekamp sur touts les échantillons (en commencent par le premier bit).

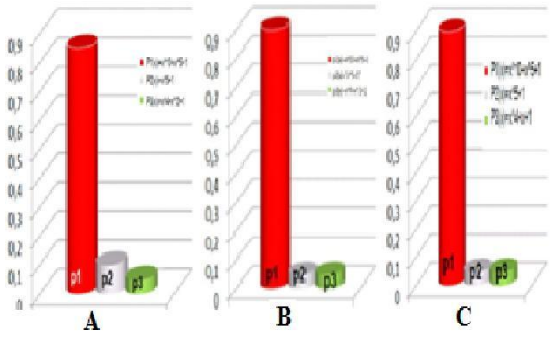

Figure 5

Les résultats de l'algorithme Massey Berlekamp fait ressortir le polynôme $p_2(x)=x^{10}+x^5+1$ avec un pourcentage de pus de 80% par rapport aux autres polynômes trouvés ce qui implique que les données sont probablement à 80% cryptées avec le polynôme $p_2(x)$. On remarque aussi que l'algorithme, au-delà d'un nombre d'échantillon égal à $n \geq 2L$, donne les mêmes résultats, mais il n'est pas efficace dans le cas ou $n < L$.

Dans la pratique cryptographique les LFSR sont des boites noires, on ne connaît pas le polynôme donc on prend n (la taille de l'échantillon) aléatoirement. Dans le cas général le premier bit de la séquence à décrypter n'est pas connu, on est obligé à commencer notre décryptage par un bit aléatoire. On va échantillonner le cryptogramme (figure 6) avec chaque échantillon a une taille égale à 10 bit ; ensuite on va appliquer la méthode Massey Berlekamp sur touts les

échantillons (en commencent par un bit aléatoirement).

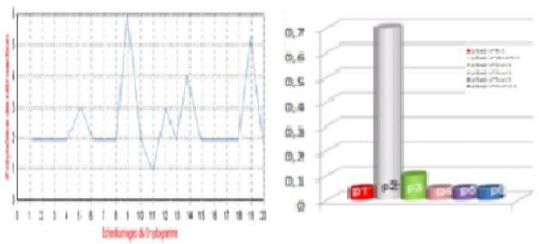

Figure 6 :

Les résultats de l'algorithme Massey Berlekamp fait ressortir le polynôme $p_2(x)=x^{10}+x^5+1$ avec un pourcentage de pus de 60% par rapport aux autres polynômes trouvés ce qui implique que les données sont probablement à 60% cryptées avec le polynôme $p_2(x)$.

5.2 Séquence de bit chevauchée :

Dans ce qui suit-on propose un nouveau modèle qui est celui de chevauchement des bits entrant dans l'algorithme de Massey Berlekamp. C'est-à-dire qu'avec un chevauchement de n bit. (les derniers n bits d'un échantillon sont les premiers n bits de l'échantillon qui le suive (figure 7).

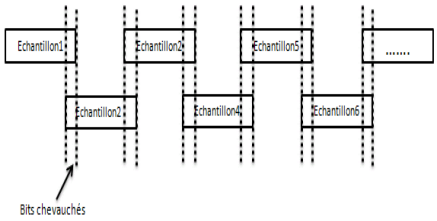

Figure7:Chevauchement des Données

On va échantillonner le cryptogramme ; avec chaque échantillon a une taille égale à 32 bit ; ensuite on va appliquer la méthode Massey Berlekamp sur touts les échantillons (en commencent par le premier bit) ; avec un chevauchement de 1 bit (**Figure 8A)**.

Par suite on va faire prendre un chevauchement de 4 bits (c'est à dire les derniers quatre bits d'un échantillon c'est les premiers quatre bits de l'échantillon suivant) **Figure 8B**, et enfin on va faire prendre un chevauchement de 10 bits (c'est à dire les derniers dix bits d'un échantillon c'est les premiers dix bits de l'échantillon suivant) **Figure 8C**.

On constate que l'ajout de fait de chevauchement améliore davantage les résultats. Et chaque fois qu'on augmente la taille de l'échantillon ; Massey Berlekamp trouve le bon polynôme avec un pourcentage plus élevé.

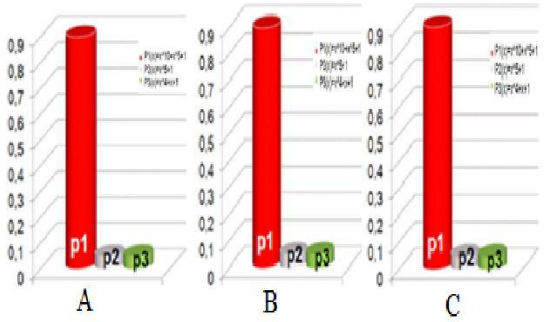

Figure 8

6. Conclusion :

L'algorithme de Berlekamp-Massey permet de reconstituer les coefficients du LFSR, ainsi que l'état initial, à partir de n bits de suite chiffrante si et

seulement si n ≥ 2 Λ (s^n).mais on ne connaît pas la valeur de Λ pour cela on a choisi n aléatoirement avec une grande valeur ; la $t^{ième}$ itération détermine en fait un LFSR de longueur minimale qui engendre les t premiers éléments de s^n.

Même lorsque le polynôme de rétroaction du registre est choisi de manière appropriée, la complexité linéaire de la suite produite reste généralement trop faible pour se prémunir d'une attaque par l'algorithme de Berlekamp-Massey. Afin de surmonter cet obstacle et d'augmenter la taille de l'espace des clefs, c'est-à-dire le nombre d'initialisations possibles, on utilise *m* LFSRs en parallèle, et on combine leurs sorties par une fonction booléenne appelée fonction de combinaison. Dans la pratique, les LFSRs ne sont donc pas utilisés de manière isolée, mais essentiellement sous la forme des registres combinés ou filtrés.

Les résultats obtenus et les essais réalisés montrent que cette méthode est efficace pour casser les systèmes de chiffrement basés sur les LFSR. Nous avons remarqué que le chevauchement améliore les résultats. Mais, si on ne tombe pas sur la bonne partie de suite chiffrée ça nous coûte un temps de calcule> N^2. (N le nombre de bit consécutifs choisis). Le temps de chiffrement dépasse six heures ; et le temps décryptage dépasse 24 heures.

Le temps et la complexité du calcul augmente si le polynôme de rétroaction utilisé dans le chiffrement est irréductible, et primitif (parce que la période sera maximal) les coefficients de rétroaction du LFSR sont bien choisis.

7. Bibliographie

[1] B. Schneier,'' Applied Cryptography-Protocols, Algorithms and Source Code in C'', John Wiley & Sounds, Inc, New York, Second Edition, 1996.

[2] Thomas Pornin, « Implantation et optimisation des primitives cryptographiques », Groupe de Recherche En Complexité et Cryptographiale 2001.

[3] Matthieu Finiasz Nouvelles constructions utilisant des codes correcteurs d'erreurs en cryptographie 2004.

[4] Christine Bachoc, « Cours de cryptographie symétrique », Université Bordeaux I Master CSI 2004-2005.

[5] Marion VIDEAU', « Critères de sécurité des algorithmes de chiffrement à clé secrète », L'université PARIS 2005.

[6] Christophe Giraud, « Attaques de cryptosystèmes embarqués », Université de Versailles Saint-Quentin 2007.

[7] Abderrahmane NITAJLE, « Cryptosystème », Université de Caen Département de Mathématiques France 2007.

[8] Eric Wegrzynowski, « Chiffrement à flot », Université de Lille 2009.

[9].Simon Guillem-Lessard, "Cryptanalyse ", 2002, Département des mathématiques et de l'informatique, Université du Québec à Trois-Rivière téléchargée le 10/04/10 de : http://www.uqtr.ca/~delisle/Crypto/cryptanalyse

[10] Aicha TEKKOUK, 'Etude et Implémentation d'une méthode cryptanalyse pour le chiffrement continu', Mémoire de magister, Département d'Informatique, Université des Sciences et de la Technologie d'Oran, Juin 2010.