Université de Limoges
L.A.C.O.
École Doctorale STS

Preuves d'Analyse et de Sécurité en Cryptologie à Clé Secrète

THÈSE

présentée et soutenue publiquement le 30 septembre 2002

pour l'obtention du

Doctorat de l'Université de Limoges

Spécialité : Mathématiques
(Arrêté du 30 mars 1992)

par

Marine MINIER

Composition du jury

Président :	Jean-Pierre Borel
Rapporteurs :	Pascale Charpin Jacques Patarin
Examinateurs :	François Arnault Anne Canteaut Bart Preneel
Directeurs de thèse :	Henri Gilbert Thierry Berger

Thèse effectuée à France Télécom R&D, 38-40, rue du Général Leclerc, 92794 ISSY LES MOULINEAUX CEDEX 9

Mis en page avec la classe thloria.

Remerciements

Cette thèse n'aurait pu se faire sans Henri Gilbert, aussi c'est en tout premier lieu à lui que s'adressent ces remerciements. Son goût pour la recherche notamment en cryptanalyse et sa constante volonté de dépassement m'ont permis tout au long de ces trois années de comprendre ce qu'est le travail, l'intuition et l'intelligence de la recherche. Merci donc à Henri.

Merci également à Thierry Berger qui a accepté d'être mon directeur de thèse à la faculté des sciences de Limoges. Ses conseils m'ont été précieux et m'ont aidée à comprendre le lien entre codes et cryptographie. Ses cours m'ont également permis d'apprécier l'algèbre linéaire à sa juste valeur.

Je tiens à exprimer toute ma gratitude à Pascale Charpin, responsable du projet CODES, pour l'intérêt qu'elle a porté à mes travaux et pour avoir accepté d'être rapporteur de cette thèse.

J'adresse mes sincères remerciements à Jacques Patarin qui a accepté d'être rapporteur de cette thèse. Ses travaux m'ont permis de comprendre ce que sont les preuves de sécurité de la cryptologie à clé secrète et ont été d'une aide précieuse pour élaborer la troisième partie de cette thèse.

Merci à François Arnault qui a suivi mon travail à la faculté de Limoges et qui a su plusieurs fois orienter mes idées et mes intuitions. Je le remercie également pour les cours de cryptographie qu'il m'a donnés.

Je remercie également Anne Canteaut pour avoir accepté d'être membre de ce jury et pour son chaleureux accueil à l'INRIA. Ses conseils et ses remarques m'ont été très utiles pour rédiger la version finale de ce manuscrit.

Je tiens également à remercier Bart Preneel et Jean-Pierre Borel qui ont accepté d'être membres de ce jury. C'est un très grand honneur pour moi. Merci pour l'intérêt qu'ils portent à mes travaux.

Merci aussi à tous les membres du département DTL/SSR de France Télécom R&D à Issy-Les-Moulineaux où j'ai effectué cette thèse. Tous ont contribué à rendre ces trois années passionnantes. Je remercierai en particulier David Arditti notre chef à tous pour sa gentillesse, Laurent Frisch pour ses judicieux conseils mathématiques, Gilles Macario-Rat pour ses énigmes mathématiques, Sarah Nataf, Nicolas Prud'homme et Melaine Broudic pour leurs conseils informatiques, Sylvie Camus pour ses conseils sur les fonctions booléennes et son aide en LaTeX, Olivier Charles pour ses points de vue toujours intéressants, Sébastien Allard pour nos discussions littéraires et enfin Thierry Baritaud pour sa disponibilité.

Je tiens également à remercier les amis qui m'ont donné l'énergie de continuer, par

ordre alphabétique donc :

Anaïs, Anne, Atlantis, Audrey, Bertrand, Blandine, Bruno, Célio, Chloé, Christine, Clément, Cyprien, Damien, Didier, Emmanuel, Étienne, Flavie, Frédérique, Gaucelm, Guillaume, Isabelle, Isabelle, Jérôme, Julien, Karine, Kik, Landry, Laurent, Marc, Marie, Nadège, Nathalie, Pascal, Pépou, Pierre, Samuel, Sébastien, Sophie, Stéphanie, Sylvain, Valérie, Véronique, Yann, tous les membres de ma fanfare,...

Et ma mère, en tant que relectrice naïve et parce que même si elle n'a rien compris, est très fière de moi.

Enfin, bonne lecture à ceux qui auront le courage d'aller plus loin.

À la mémoire de mon père.

Table des matières

Introduction

Cette thèse n'aurait pu voir le jour sans le contexte cryptographique né de la compétition AES [AES98] qui s'est déroulée entre 1997 et le premier octobre 2001 et qui a débouché sur le choix de Rijndael [DR98] comme nouveau standard de chiffrement à clé secrète américain pour le 21ème siècle. L'idée était alors d'étudier la sécurité des différents candidats de l'AES. Dans un second temps, l'Union Européenne a également lancé un appel d'offre nommé NESSIE [Nes01a] afin d'évaluer un certain nombre de primitives cryptographiques soumises. L'objet de cette thèse devenait alors évident.

La première partie de cette thèse, composée de deux chapitres, tente, après une rapide introduction à la cryptologie, de faire une description la plus complète possible des connaissances actuelles en matière d'algorithmes de chiffrement par blocs en cryptographie à clé secrète. Dans un premier temps, nous décrivons les méthodes classiques les plus usitées qui permettent de construire des algorithmes itératifs telles que le schéma de Feistel [Nat77] ou les réseaux de substitutions-permutations. Nous nous intéressons ensuite aux différents types de cryptanalyses connues, les plus marquantes étant les attaques différentielle [BS91] et linéaire [Mat93], avant d'aborder les méthodes développées pour résister à ces différents types de cryptanalyses.

La deuxième partie est consacrée aux trois cryptanalyses construites durant cette thèse contre les algorithmes Crypton [Lim98], candidat de l'AES, Rijndael [DR99], l'AES et SFLASH [PCG00], schéma de signature à clé publique soumis à NESSIE. La première cryptanalyse qui porte sur une version à huit étages de l'algorithme Crypton possède une parenté évidente avec la partitioning cryptanalysis introduite par C. Harpes et J. Massey [HM97] mais reste cependant d'un type nouveau. Nous avons appelé la méthode utilisée cryptanalyse stochastique. Cette méthode nous permet de monter deux attaques contre Crypton l'une utilisant les principes différentielles, l'autre les principes linéaires. La deuxième cryptanalyse est une amélioration de l'attaque dite par saturation sur une version à sept étages de l'AES. Cette cryptanalyse, que nous présentons dans le contexte des autres attaques connues à ce jour contre l'AES, est certainement l'élément le plus marquant de cette thèse. Elle permet d'attaquer sept étages de l'AES sous des clés de 192 et 256 bits avec une complexité nettement inférieure à une recherche exhaustive. La troisième cryptanalyse présentée dans cette partie concerne SFLASH, schéma de signature à clé publique dont la structure est très proche de celle utilisée dans la construction des algorithmes de chiffrement par blocs. Cette cryptanalyse, qui utilise une faiblesse particulière des fonctions de SFLASH, s'appuie sur des principes d'attaque également utilisés en clé secrète, notamment la cryptanalyse différentielle d'ordre supérieur.

La troisième et dernière partie de cette thèse est consacrée a l'étude des preuves de sécurité des algorithmes de chiffrement par blocs dans le modèle dit de Luby et Rackoff [LR88]. Nous présentons tout d'abord les notions de base nécessaires à la compréhension de telles preuves, notamment la preuve simplifiée par J. Patarin [Pat91] du principal résultat démontré dans [LR88], à savoir qu'il existe une borne supérieure sur la probabilité d'un attaquant de distinguer, à l'aide de q entrées/sorties, un schéma de Feistel à trois étages d'un schéma idéal. Le deuxième chapitre est consacré à la théorie de la décorrélation introduite par S. Vaudenay dans [Vau98a]. Cette théorie tente d'unifier les différentes notions de sécurité des algorithmes de chiffrement par blocs à l'aide d'une généralisation de la définition de fonction universelle [CW79]. Nous développons ensuite les nouveaux résultats de sécurité prouvée obtenus durant cette thèse sur deux variantes du schémas de Feistel, pour des versions à 4 et 5 étages.

Première partie

Notions Générales en Cryptologie

Chapitre 1

Généralités sur la Cryptologie

La cryptologie, étymologiquement "la science du caché" par extension "la science du secret", est composée de deux disciplines complémentaires : la cryptographie et la cryptanalyse. La cryptographie est l'art de cacher l'information. Elle répond aux besoins suivants :

- Confidentialité : garantir que les données échangées ne sont dévoilées qu'aux personnes voulues.
- Intégrité : garantir que les données échangées ne sont pas altérées intentionnellement ou non.
- Authenticité : prouver l'identité d'une personne ou l'origine d'un document.
- Signature : permettre à une personne de "signer" informatiquement un document sans qu'elle puisse le renier par la suite.

Quant à la cryptanalyse, c'est le pendant naturel de la cryptographie : lorsqu'un système cryptographique est utilisé, certains vont tenter de l'attaquer pour mettre en défaut sa sécurité. L'ensemble des attaques sur les cryptosystèmes est la cryptanalyse.

1.1 Petit Historique de la Cryptographie

L'histoire de la cryptographie est indissociable de celle de l'écriture. Les premières traces de son utilisation datent de 2000 ans avant Jésus-Christ en Égypte. Durant des siècles, la cryptographie se fonda sur deux grands principes :

- La permutation : on modifie dans le texte clair l'ordre des lettres.
  Le plus ancien exemple d'utilisation de permutation dans un chiffrement est dû aux Spartiates. Le procédé de chiffrement était un bâton appelé scytale sur lequel était enroulé un ruban de parchemin ou de papyrus. Le texte était alors écrit en continu sur le ruban. On envoyait le ruban au destinataire qui pour lire le texte n'avait plus qu'à l'enrouler sur un scytale de même diamètre.
- La substitution : dans un alphabet, on remplace une lettre par une autre (substitution monoalphabétique) ou une suite de deux lettres ou plus par une autre (substitution polyalphabétique).
  L'alphabet de César reste le plus fameux exemple de ce type de chiffrement. Il consiste en un décalage de 3 positions des lettres de l'alphabet.

Ce n'est qu'en 1949 que Shannon [Sha49], dans la théorie de l'information, formalise les deux concepts utiles à la construction de cryptosystèmes qu'il nomma respectivement diffusion (chaque bit du texte clair doit influencer le plus grand nombre possible de bits du chiffré) et confusion (la structure liant le chiffré et le clair doit être très complexe).

L'apparition et le développement des ordinateurs et des réseaux de télécommunications au cours des dernières décennies a sorti la cryptographie de son cadre strictement diplomatique et militaire. La diffusion d'information à l'échelle de la planète a rendu nécessaire la création de systèmes cryptographiques communs réputés très sûrs, car la puissance de calcul des ordinateurs personnels permet aujourd'hui à chacun de s'essayer à la cryptanalyse. De nouveaux problèmes liés à ces nouvelles technologies sont apparus comme le contrôle d'accès,...

C'est dans ce contexte qu'est né en 1975 l'un des premiers grands algorithmes de la cryptographie à clé secrète : le DES "Data Encryption Standard", très influencé par la théorie de l'information et choisi comme standard de chiffrement par le gouvernement américain en 1977 [Nat77]. Un an plus tard, en 1976, est apparue une nouvelle forme de cryptographie : la cryptographie à clé publique introduite par Diffie et Hellman [DH76].

Depuis, la recherche en cryptographie n'a cessé d'avancer tant en clé publique qu'en clé secrète. Citons ici le choix, en octobre 2000, du nouveau standard de chiffrement américain pour le XXIème siècle : l'AES "Advanced Encryption Standard", en remplacement du DES.

1.2 Chiffrement et Déchiffrement

Le premier objectif de la cryptographie est d'assurer la confidentialité des données. Supposons que deux individus veuillent communiquer entre eux sur un canal non sûr sans que personne d'autre ne puisse lire le contenu du message ; ces deux individus, que nous appellerons Alice et Bob comme le veut la tradition cryptographique, doivent pour se protéger des éventuelles attaques du dénommé Oscar utiliser un système cryptographique. Si Alice veut envoyer un message à Bob, elle chiffre le texte du message grâce à une fonction de chiffrement E utilisant une clé k. Elle envoie alors le message chiffré que seul Bob sera capable de déchiffrer à l'aide d'une fonction de déchiffrement D et d'une donnée supplémentaire (la clé par exemple). On peut alors formaliser la notion de fonction de chiffrement et de déchiffrement de la façon suivante :

Définition 1 *Soit M l'ensemble des textes clairs, C l'ensemble des textes chiffrés, K l'ensemble des clés. Une fonction de chiffrement E associe à tout couple (k, m) de $K \times M$ un élément $c = E(k,m)$ de C. La fonction de déchiffrement correspondante, paramétrée par la clé de déchiffrement d, associe alors au couple (d, c) de $K \times C$ le message m de M, elle vérifie : $D(d, E(k,m)) = m$.*

1.3 Deux Méthodes pour Échanger de l'Information

Jusqu'en 1976, la seule méthode connue pour échanger de l'information était la cryptographie à clé secrète ; pour que deux personnes puissent échanger un message chiffré, elles devaient partager un même secret connu d'elles seules. En 1976, est apparue une nouvelle forme de chiffrement dite à clé publique où les protagonistes ne sont plus tenus de partager une même clé.

1.3.1 Chiffrement Symétrique

Si Alice et Bob décident d'utiliser pour communiquer un principe de chiffrement symétrique, ils doivent partager la même clé k connue d'eux seuls qui est identique pour le chiffrement et le déchiffrement :

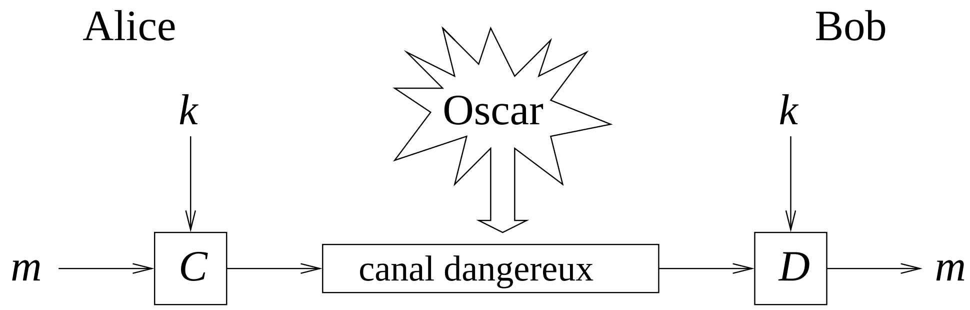


Fig. 1.1 – Chiffrement Symétrique

Un des exemples les plus connus de cryptographie à clé secrète est le chiffrement de Vernam, inventé par ce dernier en 1917. C'est un des seuls chiffrements inconditionnellement sûrs (c'est à dire pour lequel la connaissance du chiffré ne permet d'obtenir aucune information sur le texte clair). Si Alice veut envoyer un message m de longueur n à Bob en utilisant cette méthode, elle doit générer une clé aléatoire k de longueur n et calculer le chiffré à envoyer $c = m \oplus k$, où $\oplus$ représente le ou exclusif bit à bit. Pour déchiffrer le message c, Bob qui possède la même clé k qu'Alice n'a alors plus qu'à calculer $m = c \oplus k$. Pour que ce chiffrement soit inconditionnellement sûr, il ne faut utiliser la clé k qu'une seule fois, d'où le deuxième nom du chiffrement de Vernam : "One time pad". L'inconvénient majeur de cet algorithme est la longueur de la clé : on doit avoir $k \geq$ taille(m).

1.3.2 Chiffrement Asymétrique

En 1976, Diffie et Hellman décrivent le principe d'un nouveau type d'échange d'information : la cryptographie asymétrique ou à clé publique. Si Alice veut envoyer un message confidentiel à Bob, elle se procure (par exemple dans un annuaire public) la clé publique K_P de Bob et chiffre le message m à l'aide d'un algorithme de chiffrement asymétrique et de la clé K_P. Elle envoie ensuite sur le canal non sûr le message chiffré c que seul Bob peut déchiffrer grâce à la clé secrète correspondante K_S, connue de lui seul.

Citons ici l'exemple de l'algorithme RSA, introduit par Rivest, Shamir et Adleman en 1978 dans [RSA78]. Pour créer sa clé publique K_P, Bob calcule deux entiers N et e où N est le produit de deux grands nombres premiers p et q et e est tel que $0 < e \leq N$ et pgcd$(e, \varphi(N)) = 1$ où φ est la fonction indicatrice d'Euler et donc $\varphi(N) = (p-1)(q-1)$. Il calcule également l'entier d tel que $0 \leq d \leq N$ et $ed \equiv 1 \bmod \varphi(N)$. Il rend publics

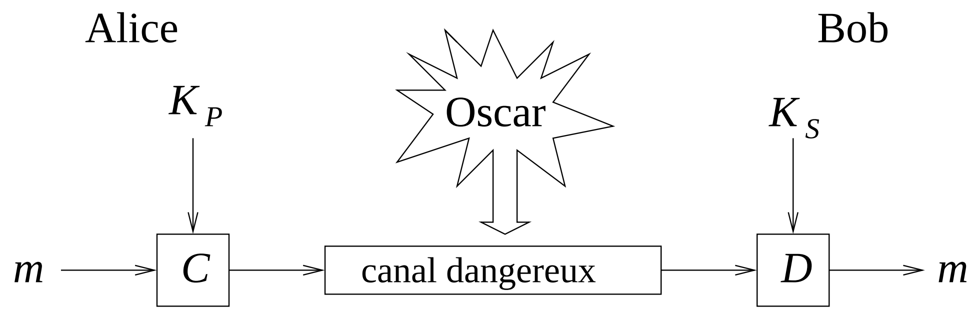


FIG. 1.2 – Chiffrement Asymétrique

e et N et garde secrète sa propre clé d. Lorsqu'Alice veut envoyer un message m à Bob, elle va chercher le couple (N,e) et calcule $c = m^e \bmod N$ qu'elle envoie. Bob reçoit c et déchiffre le message d'Alice en calculant $c^d = m^{ed} = m \bmod N$ à l'aide de sa clé secrète.

Cet algorithme repose sur la difficulté du problème de la factorisation de grands nombres.

A priori, la cryptographie à clé publique apparaît comme la meilleure méthode de chiffrement car pour que deux personnes communiquent sur un canal non sûr, ils n'ont pas besoin de pré-échanger une clé. Cependant, les algorithmes à clé publique sont lents. Par exemple, l'algorithme RSA chiffre au moins mille fois plus lentement que le DES (Data Encryption Standard), ancien standard de chiffrement à clé secrète. La méthode la plus efficace et la plus usitée pour échanger de l'information sur un canal non sûr est de ce fait une combinaison des deux méthodes de chiffrement : on chiffre le message à envoyer à l'aide d'un algorithme symétrique et d'une clé secrète, clé transmise au destinataire à l'aide d'un algorithme asymétrique. Il est donc indispensable de continuer à étudier les algorithmes de chiffrement symétrique, ce qui est d'ailleurs l'objet de cette thèse.

1.4 Deux Méthodes pour Chiffrer en Clé Secrète

Les deux méthodes les plus usitées en cryptographie à clé secrète sont le chiffrement à flot et le chiffrement par blocs.

1.4.1 Chiffrement à Flot

Le principe des algorithmes à flot informatiquement sûrs repose sur le partage d'une clé K de longueur constante et non sur le partage d'une clé de longueur égale à celle du message à chiffrer. L'algorithme utilisé chiffre "à la volée" le flot de message clair par un ou-exclusif bit à bit (ou une opération similaire) avec une suite chiffrante (k_i) pour produire un flot de texte chiffré, comme dans le cas du chiffrement de Vernam. La suite chiffrante $k = (k_i)$ est produite par un générateur pseudo-aléatoire G à partir de la clé K et d'un vecteur d'initialisation IV.

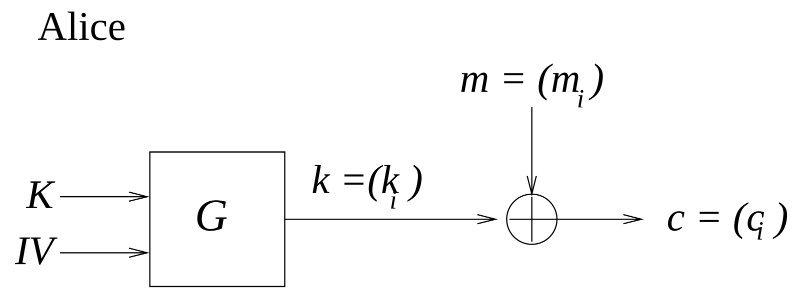


FIG. 1.3 – Principe du Chiffrement à Flot

1.4.2 Chiffrement par Blocs

On oppose souvent chiffrement à flot et chiffrement par blocs et pour cause : le premier utilise une suite chiffrante, le deuxième une fonction. Le chiffrement par blocs consiste à découper en blocs de longueur fixe n le message m à chiffrer, puis à traiter chaque bloc séparément. Il faut que n soit suffisamment grand pout éviter les attaques par dictionnaire. Il existe plusieurs méthodes, appelées modes, pour chiffrer un message m à l'aide d'un algorithme de chiffrement par blocs E_k et d'une clé k. Le mode le plus simple est le mode ECB (de l'anglais Electronic Code Book) : le message m est découpé en blocs de taille n ; chaque bloc est chiffré séparément et on concatène ensuite les blocs de chiffré obtenus. Un autre mode de chiffrement très employé est le mode CBC (de l'anglais Cipher Block Chaining) : le message clair est ici encore découpé en blocs de longueur n, mais au lieu de chiffrer simplement un bloc, on chiffre le bloc i préalablement combiné par ou exclusif avec le chiffré du bloc précédent : si m_i désigne le i-ème bloc et c_{i-1} le $(i-1)$-ème chiffré, on obtient c_i par la relation $c_i = E_k(m_i \oplus c_{i-1})$ avec $c_0 = E_k(m_0 \oplus IV)$ où IV est un vecteur d'initialisation.

1.5 La Cryptanalyse ou Que Fait Oscar ?

Même si la cryptanalyse s'est développée parallèlement à la cryptographie, il est nécessaire de préciser certains concepts spécifiques à cette discipline. Jusqu'à une époque assez récente, la sécurité d'un chiffrement reposait tout autant sur la non-divulgation de l'algorithme utilisé que sur la clé. Ce n'est plus le cas aujourd'hui et on attend d'un algorithme, conformément au *principe de Kerckhoff*, qu'il reste solide lorsque l'attaquant en connaît la spécification et ignore seulement la clé utilisée.

La cryptanalyse d'un système peut être alors soit partielle (l'attaquant découvre le texte clair correspondant à un ou plusieurs messages chiffrés interceptés), soit totale (l'attaquant peut déchiffrer tous les messages, par exemple en trouvant la clé). Il existe plusieurs types d'attaques selon les moyens dont dispose l'attaquant :

- *Attaques à chiffré connu* : l'attaquant a seulement accès à des messages chiffrés.
- *Attaques à clair connu* : l'attaquant dispose d'un ou plusieurs messages clairs et des chiffrés correspondants.
- *Attaques à clair choisi* : l'attaquant choisit des clairs et peut obtenir les chiffrés correspondants.

– *Attaques à chiffré choisi* : l'attaquant peut déchiffrer les messages de son choix.

L'attaque la plus simple et la plus brutale est la recherche exhaustive. L'attaquant teste l'ensemble des clés possibles sur un cryptogramme donné dont il est supposé connaître au moins partiellement le clair ; il a découvert la bonne clé lorsque le déchiffrement redonne le clair attendu. La complexité d'une recherche exhaustive sur un algorithme de chiffrement par blocs dont la taille des clés est n bits est de l'ordre de 2^n chiffrements. Il est donc nécessaire lorsque l'on souhaite créer un algorithme de chiffrement de prendre des clés suffisamment longues pour se prémunir contre ce type d'attaque. Malheureusement, ce n'est pas la seule condition à vérifier. C'est pourquoi, la construction de tels algorithmes doit s'appuyer sur des problèmes mathématiques difficiles.

Ces concepts restent très généraux mais permettent de mieux comprendre les notions qui seront abordées dans la suite de cette thèse, notamment aux chapitres 2 et 3.

Chapitre 2

État de l'Art

Dans ce chapitre, nous allons présenter les principales notions utilisées en cryptologie à clé secrète, tant dans la construction des fonctions de chiffrement par blocs qu'en cryptanalyse. Il peut paraître étonnant de ne pas aborder ici les preuves de sécurité notamment celles du modèle de Luby et Rackoff [LR88]. Elles seront abordées dans le premier chapitre de la dernière partie en introduction des nouveaux résultats de sécurité prouvée obtenus durant cette thèse. Nous prions donc le lecteur de se reporter au début de la troisième partie pour plus de détails en ce qui concerne les preuves de sécurité de la cryptographie à clé secrète.

2.1 Méthodes Classiques dans le Choix des Primitives Cryptographiques

Pour répondre aux exigences de la cryptographie symétrique, un algorithme de chiffrement par blocs E_k doit satisfaire plusieurs propriétés. Rappelons les plus importantes : l'algorithme doit être facilement calculable et inversible pour n'importe quelle valeur de la clé, tout en offrant une bonne sécurité pratique.

2.1.1 Structure Générale

Les méthodes de construction restent très empiriques mais la plus usitée est l'utilisation d'algorithmes itératifs. Ces algorithmes sont composés de r étages et à chaque étage, ils font agir la même fonction f paramétrée par une sous-clé k_i. Les sous-clés k_1,..., k_r sont différentes à chaque étage et générées à partir de la clé secrète k (voir figure 2.1).

Notons que, mis à part quelques cas pathologiques, plus le nombre d'étages est grand, plus on augmente la diffusion et la confusion.

2.1.2 Schéma de Feistel

En 1977, le gouvernement américain adopta comme standard de chiffrement le DES (Data Encryption Standard) développé par IBM [Nat77]. Le premier algorithme proposé, nommé Lucifer, rejeté à cause d'une faille de sécurité (voir par exemple [BAB96]), avait

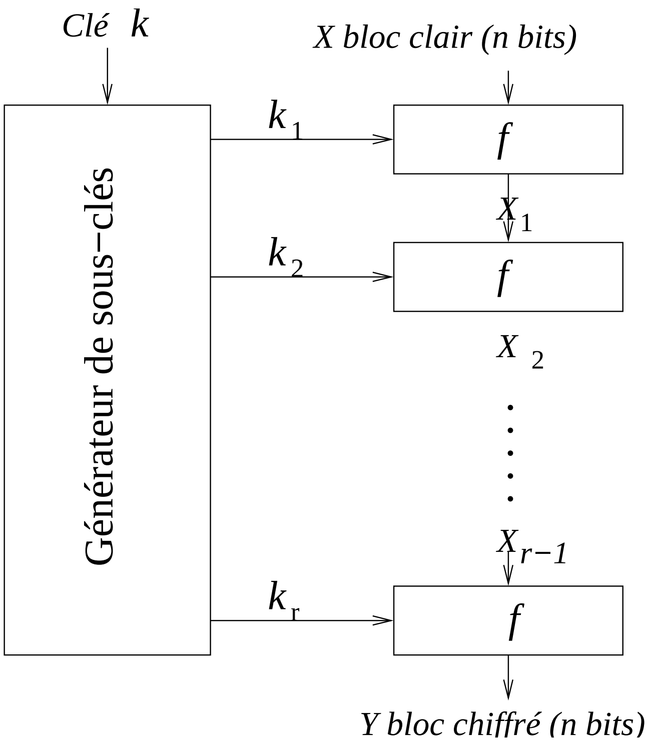


FIG. 2.1 – Structure Générale d'un Chiffrement par Blocs Itératif

été créé par Horst Feistel. Le DES utilise le même schéma qui a gardé le nom de son inventeur. Un chiffrement de Feistel est un chiffrement itéré qui utilise à chaque étage le même schéma. Rappelons la définition d'un schéma de Feistel :

Définition 2 *Soit f_1 une fonction de $I_n = \{0,1\}^n$ dans lui-même et x^0, x^1, x^2, x^3 quatre éléments de I_n. On définit le schéma de Feistel Ψ lié à f_1 sur $I_{2n} = \{0,1\}^{2n}$ de la manière suivante :*

$$\forall (x^0, x^1) \in (I_n)^2, \Psi(f_1)[(x^0, x^1)] = (x^2, x^3) \Leftrightarrow \begin{cases} x^2 = x^1 \\ x^3 = f_1(x^1) \oplus x^0 \end{cases}$$

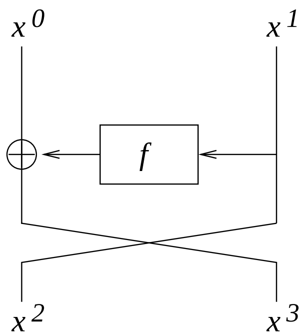


FIG. 2.2 – Schéma de Feistel

Ψ représente un schéma de Feistel sur un étage (voir figure 2.2). On peut alors généraliser la définition de Ψ à plusieurs étages : soit f_1, f_2,..., f_r, r fonctions de $\{0,1\}^n$ vers $\{0,1\}^n$, on définit $\Psi^r(f_1, \cdots, f_r)$ de $\{0,1\}^{2n}$ vers $\{0,1\}^{2n}$ par :

$$\Psi^r(f_1, ..., f_r) = \Psi(f_r) \circ \cdots \circ \Psi(f_2) \circ \Psi(f_1).$$

Précisons quelques propriétés élémentaires de ce schéma [Pat91] :

- Dans tous les cas, $\Psi(f_1)$ est une permutation de I_{2n}.
- La fonction réciproque de $\Psi(f_1)$ est très facile à calculer :

  $$\Psi(f_1)^{-1} = \sigma \circ \Psi(f_1) \circ \sigma$$

  où σ désigne la permutation de I^{2n} qui inverse les parties droite et gauche d'un mot de I^{2n}.

Citons ici l'exemple le plus connu des schémas de Feistel : le DES, qui chiffre des blocs de 64 bits avec une clé de longueur 56 bits. L'algorithme se compose d'une permutation initiale, de 16 itérations d'une fonction étage f de $\{0,1\}^{32}$ dans lui-même dans un schéma de Feistel, et d'une transformation finale composée de la permutation σ qui échange les moitiés droite et gauche du mot de 64 bits et de la permutation initiale inversée. Nous nous contenterons de décrire la fonction f. Elle se compose d'une fonction d'expansion affine E, suivie d'un x-or avec la sous-clé k_i (longue de 48 bits) de l'étage, puis d'une fonction S et d'une permutation P (voir figure 2.3). Si on note (x^{i-1}, x^i) les deux mots de 32 bits d'entrée, le mot de sortie est alors (x^i, x^{i+1}) avec $x^{i+1} = x^{i-1} \oplus f(x^i) = x^{i-1} \oplus P(S(E(x^i) \oplus k_i))$.

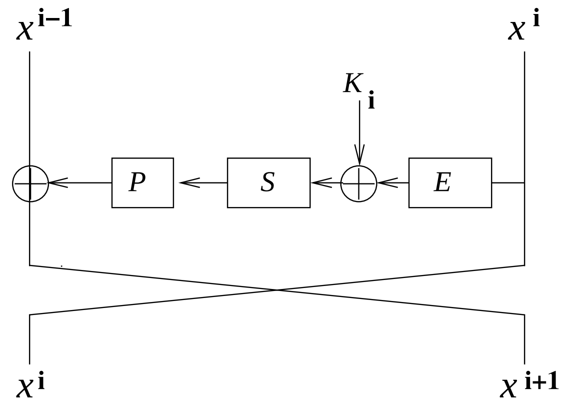


Fig. 2.3 – Un Étage du DES

L'expansion affine E transforme le mot d'entrée de longueur 32 bits x^i noté $(x_1 x_2 \cdots x_{32})$ en mots de 48 bits $y = (y_1 y_2 \cdots y_{48})$. Le mot de 48 bits est formé en prenant les bits de $(x_1 x_2 \cdots x_{32})$ dans l'ordre suivant : 32, 1, 2, 3, 4, 5, 4, 5, 6, 7, 8, 9, 8, 9, 10, 11, 12, 13, 12, 13,..., 28, 29, 28, 29, 30, 31, 32, 1.

L'addition de clé est un simple x-or bit à bit entre les 48 bits de la clé k_i et les 48 bits de y.

La fonction S fait appel à huit boîtes S différentes qui chacune transforment un mot de 6 bits en mot de 4 bits. Nous ne détaillerons pas ici les huit boîtes S, décrites dans [Nat77].

Quant à P, c'est une simple permutation entre les bits d'un mot de $\{0,1\}^{32}$ [Nat77].

Il est utile de préciser que même si le DES est encore très usité aujourd'hui, la faible longueur de sa clé (56 bits) ne le prémunit plus contre une recherche exhaustive. Ce n'est que très récemment qu'il a été remplacé par un nouveau standard, l'AES, qui utilise un réseau de substitutions-permutations.

2.1.3 Réseau de Substitutions-Permutations

Un réseau de substitutions-permutations est également un système de chiffrement itératif. A chaque étage, on applique au bloc d'entrée une substitution non linéaire puis une fonction généralement linéaire appelée abusivement permutation. Ces réseaux prennent au mot les deux concepts définis par Shannon. La substitution, en général représentée par une boîte S, garantit, si elle est bien choisie, une bonne confusion (faire disparaître les structures tant linéaires qu'algébriques du chiffrement) et la permutation garantit une bonne diffusion de l'information (faire en sorte que chaque bit de sortie soit influencé par le plus grand nombre possible de bits d'entrée). Précisons également que l'usage des mots substitution et permutation est abusif. On peut effectivement parler de substitution en ce qui concerne les boîtes S même si il peut y avoir confusion avec le terme de permutation mais le terme de permutation est beaucoup trop approximatif pour désigner l'application linéaire qui suit. Cependant, l'emploi de ces termes restent classique en cryptographie, nous continuerons donc à les utiliser dans la suite de ce paragraphe.

Nous développerons ici deux exemples d'algorithmes de chiffrement utilisant des réseaux de substitutions-permutations, SAFER K-64 et Rijndael.

SAFER K-64 est un algorithme orienté octet créé par J. M. Massey [Mas94] qui chiffre des blocs de 64 bits. Il est constitué de 6 étages. L'algorithme de génération de clé produit 13 sous-clés. Deux sont utilisées à chaque étage et la 13ème intervient dans la transformation finale. Nous ne présenterons ici que le chiffrement proprement dit. Nous ne nous intéresserons ni à la génération des sous-clés ni au processus de déchiffrement. La fonction étage, présentée à la figure 2.4, se compose d'une partie non linéaire et d'une partie linéaire.

Le + représente ici l'addition modulo 256. P est une permutation définie par $P(x) = (45^x \bmod 257) \bmod 256$. La fonction Q n'est autre que P^{-1}. La fonction 2-PHT est linéaire à deux entrées et deux sorties :

$$\text{2-PHT}(x,y) = (2x+y \mod 256, x+y \mod 256).$$

La transformation finale consiste juste à appliquer la première opération de la partie non linéaire en utilisant la 13ème sous-clé.

Intéressons-nous à présent au nouveau standard de chiffrement AES, nommé Rijndael, créé par J. Daemen et V. Rijmen [DR98]. Cet algorithme utilise des blocs de longueur 128, 192 ou 256 bits et des clés de longueur 128, 192 ou 256 bits. Il est composé d'un

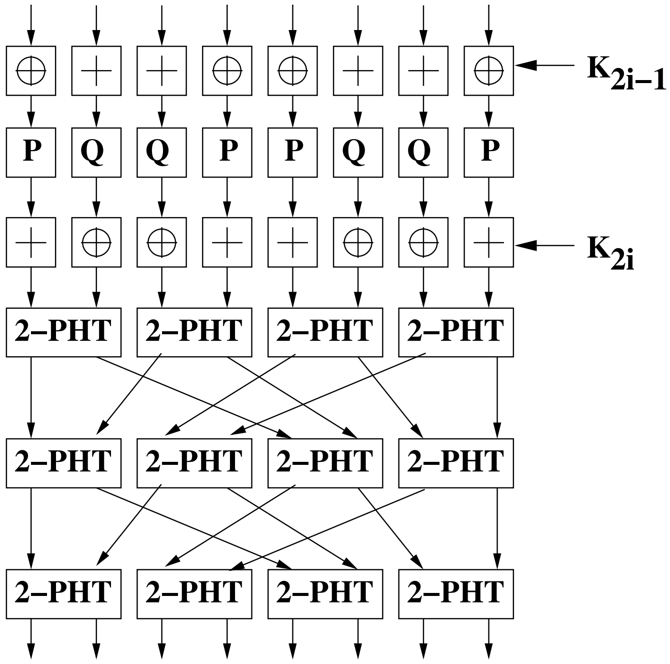


FIG. 2.4 – i-ème Étage de SAFER

nombre variable d'étages - 10, 12 ou 14 - qui dépend de la longueur de la clé et de celle des blocs à chiffrer. Le dernier étage est une transformation finale qui diffère légèrement des autres étages. La clé secrète permet de produire une sous-clé par étage, de la même taille que les blocs à chiffrer. Ces derniers sont représentés par une matrice d'octets 4×4, 4×6 ou 4×8. Rijndael utilise une structure parallèle mettant en oeuvre quatre transformations. :

- ByteSub : substitution non linéaire de $GF(256)$ dans $GF(256)$ faisant appel à une boîte S choisie pour garantir de bonnes propriétés de confusion.
- ShiftRow : application linéaire qui fait subir à chacune des lignes de la matrice une rotation circulaire d'un certain nombre d'octets vers la gauche (rotation de 0 pour la première ligne, de 1 octet pour la deuxième, de 2 octets pour la troisième et 3 pour la quatrième).
- MixColumn : multiplication matricielle dans le corps $GF(256) \approx GF(2)[x]/(x^8+x^4+x^3+x+1)$ où chaque octet est représenté par un polynôme de degré 7) portant sur chaque colonne de la matrice à chiffrer. C'est une application linéaire garantissant une bonne diffusion. La matrice utilisée est la suivante :

$$M = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix}$$

- Key-Addition : addition de clé classique. On x-ore chaque octet de la matrice à chiffrer avec un octet de la sous-clé de l'étage correspondant.

Il existe beaucoup d'autres exemples d'algorithmes utilisant une structure parallèle. On peut citer Crypton [Lim98], lui aussi candidat de l'AES, SAFER+ [MKK98] et SA-

FER++ [MKK00], nouvelles versions de SAFER, Serpent [ABK98], autre candidat de l'AES.

Nous venons de voir deux des méthodes les plus connues pour créer des fonctions d'étage d'algorithmes de chiffrement à clé secrète. Même si les schémas présentés semblent différents dans leur conception, ces deux méthodes font appel aux mêmes propriétés pour assurer une bonne sécurité et aux mêmes types de fonctions. En général, elles "enchaînent" ce qu'on appelle une boîte S, c'est à dire une permutation non linéaire avec une fonction de diffusion linéaire, même dans le cas d'un schéma de Feistel qui n'est finalement qu'un cas particulier d'un réseau de substitutions-permutations.

2.1.4 La Génération des Sous-Clés

Intéressons nous à présent à l'autre partie importante d'un algorithme de chiffrement par blocs : la génération des sous-clés à partir de la clé secrète. Généralement, celle-ci peut se décomposer en deux étapes : l'expansion de la clé et la génération des sous-clés à partir de la clé étendue. Ces opérations sont en général très simples ; on étend la clé secrète par des opérations élémentaires (permutation, décalage, addition de constante,...) en une clé étendue de longueur égale à la somme de celles de toutes les sous-clés. Puis, on "découpe" la clé étendue (de manière plus ou moins simple) en sous-clés.

Nous présenterons ici l'exemple de génération des 16 sous-clés (une par étage) de 48 bits du DES même si l'algorithme de génération de clés du DES ne comporte pas de phase d'expansion proprement dite (on peut cependant considérer que les rotations successives de la clé maître génère la clé étendue). La clé maître K du DES a une longueur de 56 bits. La première étape de la génération des sous-clés consiste à découper K en deux parties de 28 bits chacune puis à lui appliquer des rotations successives décalant vers la gauche d'un nombre α_i, dépendant du numéro i du tour, chacune des moitiés obtenues au tour $i-1$. Les deux moitiés obtenues au tour i sont ensuite concaténées pour former un nouveau bloc de 56 bits qui grâce à la transformation PC génère la sous-clé K_i du tour i longue de 48 bits.

La transformation PC est donc une fonction de compression qui transforme un mot de 56 bits en un mot de 48 bits dont l'ordre des bits de sortie est :

PC							
14	17	11	24	1	5	3	28
15	6	21	10	23	19	12	4
26	8	16	7	27	20	13	2
41	52	31	37	47	55	30	40
51	45	33	48	44	49	39	56
34	53	46	42	50	36	29	32

Précisons également que α_i vaut 1 pour les tours 1,2,9 et 16 et 2 sinon.

2.2 Cryptanalyse

Dans les méthodes de construction des primitives cryptographiques, le choix de la fonction étage repose sur les deux grands principes de confusion et de diffusion. Cependant, la sécurité de ces algorithmes reste encore empirique ; si une attaque efficace n'a pas été trouvée contre un algorithme donné, on le juge sûr. La cryptanalyse s'est attachée à définir des classes d'attaques génériques les plus large possibles et il est aujourd'hui possible de prémunir un algorithme contre les classes d'attaques connues que nous allons présenter ici.

2.2.1 Principe Général

Le principe général de la cryptanalyse repose sur la notion d'attaques par distingueur. Un distingueur $\mathcal{A}$ est en fait une machine de Turing probabiliste qui cherche par un jeu de questions/réponses à distinguer un système de chiffrement C d'un autre système de chiffrement souvent idéal noté C^*. On note p la probabilité que le distingueur, après qu'on lui a fourni un certain nombre d'informations, renvoie la réponse "1" (c'est à dire que la suite de réponses fournies aux différentes requêtes permettent de dire que le chiffrement étudié est bien l'algorithme C). On note p^* la probabilité correspondante pour C^*. Il s'agit alors d'étudier le biais probabiliste qui existe éventuellement entre ces deux chiffrements, c'est ce qu'on nomme l'avantage du distingueur et qui correspond à la quantité $Adv_{\mathcal{A}}(C, C^*) = |p - p^*|$.

Lors d'une attaque, on cherche donc un distingueur particulier souvent donné par des relations particulières liant les clairs et/ou les chiffrés de l'algorithme C qui fournit un avantage le plus grand possible, c'est à dire que l'on cherche des relations fournies par le chiffrement C qui ont une probabilité de se produire la plus éloignée possible de la probabilité uniforme. Ces relations peuvent être de tout type (dans le cas de la cryptanalyse différentielle, il s'agit d'un x-or entre deux blocs d'entrée dont on étudie les probabilités de transition pour un certain nombre de tours,...).

Plus le nombre d'étage impliqué dans la relation est grand et plus le biais est important, plus on pourrat efficacement obtenir de l'information sur les sous-clés et la clé maître. Pour tester une clé (ou une partie seulement de cette clé selon le type de relation utilisé dans le distingueur), il suffit de regarder, sur un nombre suffisant de messages, si le biais obtenu réellement pour cette clé particulière après passage dans le distingueur est très proche du biais théorique. C'est ce qu'on appelle un test d'hypothèse. Celui-ci fait donc essentiellement appel à des tests statistiques où la probabilité de réussite, liée au biais impliqué par la relation dans le distingueur, permet de calculer la complexité et le nombre de textes nécessaires au succès d'une telle attaque.

Le principal but des attaques présentées ici est de construire un distingueur à partir d'une relation portant sur les $(r-2)$ étages du milieu ou sur les $(r-1)$ premiers étages, puis, à partir de ce distingueur, de faire des tests d'hypothèse sur tout ou partie des première et/ou dernière clés.

2.2.2 Cryptanalyse Différentielle

La cryptanalyse différentielle, introduite par E. Biham et A. Shamir en 1991 [BS91], est une attaque à message clair choisi applicable aux algorithmes à clé secrète itératifs. Le principe général de cette attaque consiste à considérer des couples de clairs X et X' présentant une différence ΔX fixée et à étudier la propagation de cette différence initiale à travers le chiffrement. Les différences sont définies par une loi de groupe, en général le x-or bit à bit.

Cette attaque utilise la faiblesse potentielle de la fonction itérée f dans une dérivation à l'ordre 1.

Introduisons tout d'abord les notations dont nous aurons besoin pour tenter de formaliser les concepts de la cryptanalyse différentielle.

On note $X = X_0$ et $X' = X'_0$ un couple de textes clairs et $X_1,\cdots,\ X_{r-1},\ Y = X_r$ les chiffrés des étages intermédiaires (respectivement $X'_1,\cdots,\ X'_{r-1},\ Y' = X'_r$) en supposant que la fonction d'étage f prend en entrée/sortie des blocs de n bits. On note $\Delta X_i = X_i - X'_i$ les différences des étages intermédiaires et $k_1,\ k_2,\cdots,\ k_{r-1},\ k_r$ les sous-clés des différents étages, supposées prises chacune dans un même espace noté $\mathcal{K}$.

On cherche donc à construire un distingueur à partir d'une relation liant ΔX_0 et généralement $\Delta X_{(r-1)}$ qui ait une probabilité de se produire aussi éloignée que possible de la probabilité uniforme. Cette relation doit dans la mesure du possible être indépendante ou peu dépendante des sous-clés de chaque étage.

Pour calculer les probabilités liées à ces différences, on utilise deux définitions. La première introduite par Lai et Massey [LM91] ne tient pas compte des valeurs des différences intermédiaires contrairement à celle proposée par Biham et Shamir dans [BS91].

Définition 3 *On appelle différentielle sur i tours la donnée d'un couple* $(\alpha,\ \beta)$ *tel que*

$$\alpha = X_0 - X'_0 = \Delta X_0 \text{ et } \beta = X_i - X'_i = \Delta X_i$$

On définit alors la probabilité conditionnelle associée appelée probabilité de la différentielle en supposant que le texte clair X_0 *et les sous-clés* $k_1,\ \cdots,\ k_i$ *sont indépendantes et uniformément distribuées :*

$$P(\Delta X_i = \beta | \Delta X_0 = \alpha).$$

Définition 4 *On appelle caractéristique différentielle sur i tours la donnée d'un* $(i+1)$*-uplet* $(\alpha_0,\ \alpha_1,\cdots,\alpha_{i-1},\alpha_i)$ *tel que*

$$\begin{aligned}
\alpha_0 &= X_0 - X'_0 = \Delta X_0 \\
\alpha_1 &= X_1 - X'_1 = \Delta X_1 \\
&\vdots \\
\alpha_{i-1} &= X_{i-1} - X'_{i-1} = \Delta X_{i-1} \\
\alpha_i &= X_i - X_i = \Delta X_i
\end{aligned}$$

On définit alors la probabilité conditionnelle associée appelée probabilité de la caractéristique différentielle sous l'hypothèse que le texte clair X_0 *et les sous-clés* $k_1,\ \cdots,\ k_i$ *soient indépendants et uniformément distribués :*

$$P(\Delta X_1 = \alpha_1 \wedge \cdots \wedge \Delta X_i = \alpha_i | \Delta X_0 = \alpha_0)$$

Une caractéristique différentielle sur i étages détermine donc une séquence de i différentielles sur un tour chacune.

Pour rendre la cryptanalyse différentielle praticable, on fait généralement l'hypothèse suivante, dite hypothèse d'équivalence stochastique décrite dans [LM91].
Hypothèse d'équivalence stochastique [LM91] :
Pour une différentielle (α, β) au tour $(r-1)$, on suppose :

$$P(\Delta X_{r-1} = \beta | \Delta X_0 = \alpha) \approx P(\Delta X_{r-1} = \beta | \Delta X_0 = \alpha, k_1 = \omega_1, \cdots k_{r-1} = \omega_{r-1})$$

pour la plupart des valeurs $(\omega_1, \cdots, \omega_{r-1})$.

On utilise également souvent la propriété dite de Markov également présentée dans [LM91] qui est vérifiée ou non par le chiffrement étudié :

Un algorithme de chiffrement possède la propriété de Markov [LM91] si, sur un étage, la probabilité de toute différentielle (ou de toute caractéristique ce qui revient au même ici) est indépendante du choix du texte clair. Formellement, on doit avoir :

$$\forall \gamma, P(\Delta X_1 = \beta | \Delta X_0 = \alpha, X_0 = \gamma) = P(\Delta X_1 = \beta | \Delta X_0 = \alpha)$$

en supposant que la différentielle (α, β) est différente du couple $(0, 0)$ et que la sous-clé k de l'étage est uniformément distribuée.

Dans la plupart des cas, cette propriété est vérifiée. Citons cependant le cas du DES qui ne vérifie pas cette propriété.

Cette dernière propriété est la base du théorème suivant fondé sur les propriétés des chaînes de Markov [LM91] (à savoir que les distributions de probabilités de sortie d'un étage ne dépendent que des distributions de probabilités d'entrée de cet étage) ; ce qui permet de calculer réellement les probabilités mises en jeu par une caractéristique différentielle ou une différentielle.

Théorème 1 (LM91) *Pour un chiffrement de Markov, en supposant les sous-clés indépendantes et uniformément distribuées, la probabilité d'une caractéristique différentielle sur $(r-1)$ étages est égale au produit des probabilités (supposé non nulles) des $(r-1)$ caractéristiques de chaque étage qui la constituent.*

Si on note $p[\alpha_{i-1}, \alpha_i]$ la probabilité de la caractéristique à l'étage i, on a :

$$\begin{aligned} p[\alpha_{i-1}, \alpha_i] &= P(\Delta X_i = \alpha_i | \Delta X_{i-1} = \alpha_{i-1}) \\ &= P(f(X_{i-1}, k_i) + f(X_{i-1} + \alpha_{i-1}, k_i) = \alpha_i) \end{aligned}$$

D'après le théorème précédent, si le chiffrement possède la propriété de Markov, la probabilité d'une caractéristique différentielle sur $(r-1)$ étages est :

$$P(\Delta X_1 = \alpha_1 \wedge \cdots \wedge \Delta X_{(r-1)} = \alpha_{(r-1)} | \Delta X_0 = \alpha_0) = \prod_{i=1}^{r-1} p[\alpha_{i-1}, \alpha_i]$$

La probabilité d'une différentielle sur $(r-1)$ tours peut donc être calculée :

$$P(\Delta X_{(r-1)} = \alpha_{(r-1)} | \Delta X_0 = \alpha_0) = \sum_{\alpha_1} \sum_{\alpha_2} \cdots \sum_{\alpha_{(r-2)}} \prod_{i=1}^{r-1} p[\alpha_{i-1}, \alpha_i]$$

Une autre méthode pour calculer cette probabilité consiste à utiliser la matrice des probabilités de transition de la chaîne de Markov homogène (i.e. de même matrice de transition pour toutes les valeurs de i) que représentent les ΔX_i. On note $\mathcal{M}_i$ la matrice des probabilités de transition à l'étage i. L'entrée (α_k, α_j) de la matrice $\mathcal{M}_i$ correspond à la probabilité $P(\Delta X_i = \alpha_k | \Delta X_{i-1} = \alpha_j)$ de l'étage i. La probabilité d'une différentielle (α_u, α_v) sur les l premiers tours est alors égale au coefficient d'indice (u, v) de la matrice $\mathcal{N} = \prod_{i=1}^{l} \mathcal{M}_i$.

Une fois que l'on a obtenu une relation différentielle (α, β), notée $\mathcal{R}$ sur $(r-1)$ tours qui a de bonnes chances de se produire, on peut créer le distingueur utilisant cette relation et tester les hypothèses de clé pour un algorithme à r étages :

Pour i variant de 1 à m faire
  Choisir un couple $(X_0, X_0 + \alpha)$ et chiffrer ce couple.
  Déterminer toutes les valeurs possibles k de k_r
  pour lesquelles $\Delta X_{(r-1)} = \beta$.
  compteur[k] reçoit compteur[k] + 1.
Fin Pour.

où le compteur compteur[k] est initialisé à 0. Précisons que dans de nombreux cas, on ne détermine pas la clé k_r en entier mais seulement un nombre l de bits de celle-ci déterminé par la relation $\mathcal{R}$, ce qui peut éventuellement suffire pour une attaque complète si l'on peut terminer par une recherche exhaustive ou encore dans le cas où il existe plusieurs distingueurs qui déterminent plusieurs groupes différents de l bits de la clé. On peut parfois n'utiliser qu'un distingueur sur $(r-2)$ étages (selon les cas, les $r-2$ premiers, les $r-2$ étages du milieu ou les $r-2$ derniers) si l'on parvient à effectuer une recherche exhaustive pertinente sur des bits des sous-clés des deux autres étages.

Il reste alors à trouver une bonne approximation de la valeur m qui représente le nombre de couples de clairs à chiffrer.

Comme décrit dans [Gil97], pour calculer m, on suppose que le nombre d'occurences de la différentielle attendue obéit à une loi binomiale. On note p la probabilité fonction de m que le phénomène attendu se produise pour le vrai chiffrement et p^* la probabilité également fonction de m que le phénomène attendu se produise pour un chiffrement aléatoire. Lorsque l'on teste les valeurs de clé pour une valeur de m suffisante, on peut s'attendre à ce que la "bonne" clé soit celle pour laquelle la différentielle attendue apparaît le plus grand nombre de fois.

On distingue alors deux grands cas pour le calcul de m selon l'ordre de grandeur du rapport signal sur bruit S/N, ici égal à $\frac{p-p^*}{p^*}$. On souhaite que la probabilité de réussite de l'algorithme soit de plus de 99% et la probabilité de fausses alarmes (c'est à dire de retenir une mauvaise clé) soit très faible.

$$\text{Si S/N} \gg 1 \text{ alors } m \approx \frac{4.6}{p}.$$

$$\text{Si S/N} \ll 1 \text{ alors } m \approx \frac{23 \cdot p^* \cdot (1 - p^*)}{(p - p^*)^2}.$$

En règle générale, la cryptanalyse différentielle relève du premier cas S/N $\gg 1$ et la cryptanalyse linéaire du second.

Pour permettre des calculs simples de différentielles et du nombre de clairs impliqués dans une attaque différentielle, on définit le coefficient DP^F où F est une fonction d'un ensemble E vers un ensemble E' par :

$$\mathrm{DP}^F(\alpha, \beta) = \Pr(F(X) - F(X') = \beta | X - X' = \alpha)$$

avec X, X' appartenant à E et tirés au hasard, α appartenant à E et β appartenant à E'.

Dans le cas où la fonction F est paramètrée par une clé k prise dans un espace $\mathcal{K}$, le coefficient DP dépend de la clé :

$$\mathrm{DP}^{F_k}(\alpha, \beta) = \Pr(F_k(X) - F_k(X') = \beta | X - X' = \alpha)$$

On calcule alors le coefficient EDP en moyenne sur l'espace des clés :

$$\mathrm{EDP}^F(\alpha, \beta) = E_{k \in \mathcal{K}}(\mathrm{DP}^{F_k}(\alpha, \beta))$$

où la fonction E désigne l'espérance mathématique sur l'espace des clés.

Lorsque l'on souhaite monter une cryptanalyse différentielle sur la fonction F_k, considérée ici comme le chiffrement à attaquer, on crée un distingueur à partir d'une différentielle (α, β) pour des valeurs α et β maximisant le coefficient EDP. La probabilité p que le phénomène attendu ici se produise pour la fonction F_k est :

$$p = \mathrm{EDP}^F(\alpha, \beta)$$

Une bonne approximation de m est alors

$$m \approx 4.6 \cdot \frac{1}{\mathrm{EDP}^F(\alpha, \beta)}.$$

En pratique, si on considère que la fonction d'étage f est constituée d'une série de boîtes S S_i, d'une permutation linéaire et d'une addition de clé, on cherchera une propriété de la permutation linéaire pouvant être itérée sur les $r-1$ premiers étages. Le calcul de la probabilité des différentielles sur un étage ne se fera alors plus qu'au travers des boîtes S_i à l'aide des différents coefficients $\mathrm{EDP}^{S_i}(\alpha, \beta)$ pour des valeurs de α et β à déterminer. La probabilité de caractéristiques sur $(r-1)$ étages est elle-même calculée comme le produit des différentes probabilités sur un étage, c'est à dire comme un produit des coefficients EDP^{S_i}.

Notons également qu'il existe un certain nombre d'attaques inspirées de la cryptanalyse différentielle. On peut citer ici l'exemple de la cryptanalyse utilisant des différences impossibles. On cherche dans cette attaque à déterminer des différentielles qui n'ont aucune chance de se produire afin de construire un distingueur à partir de ce type de relation. L'attaque par différentielles impossibles la plus connue est celle menée par L. Knudsen dans l'article décrivant l'algorithme DEAL [OK98] créé par R. Outerbridge et L. Knudsen, candidat malheureux de l'AES. Cette attaque a également été amélioré par S. Lucks dans [Luc99].

Nous développerons dans la section 2.2.4 des cryptanalyses elles aussi inspirées de la cryptanalyse différentielle comme la cryptanalyse différentielle d'ordre supérieur et la cryptanalyse différentielle tronquée.

2.2.3 Cryptanalyse Linéaire

Cette cryptanalyse, introduite par A. Corfdir et H. Gilbert pour cryptanalyser l'algorithme FEAL [CG91] et appliquée par M. Matsui au DES [Mat93], est une attaque à texte clair connu qui consiste à chercher des approximations linéaires de la fonction d'étage f_{k_i}, définie de $\{0,1\}^n$ dans lui-même. On cherche donc des relations de la forme :

$$a \cdot f_{k_i}(X) = b \cdot X + c \cdot k_i \text{ ou } a \cdot f_{k_i}(X) = b \cdot X + c \cdot k_i + 1$$

vérifiées pour le plus grand nombre de valeurs de X possibles où $\cdot$ représente le produit scalaire modulo 2 et a, b, et c appartiennent à $\{0,1\}^n$. k_i représente, comme dans le paragraphe précédent, la sous-clé de l'étage i. On peut donc voir la cryptanalyse linéaire comme une projection sur l'ensemble $\{0,1\}$ de la relation liant les blocs d'entrée et de sortie de la fonction f_{k_i}.

On cherche alors des relations de ce type portant sur un nombre d'étages l. Notons X_i le chiffré du i-ème étage et p_i la probabilité linéaire de l'étage i définie par

$$p_i(a_i, b_i) = \frac{1}{2} \pm \epsilon_i \text{ où } \epsilon_i = \frac{|\#\{X_{i-1} | a_i \cdot X_i = b_i \cdot X_{i-1}\} - 2^{n-1}|}{2^n}.$$

Une relation sur l étages est déduite en sommant les relations sur un étage. Pour cela, on cherche des suites de coefficients a_i et b_i dans lesquelles les coefficients des termes intermédiaires X_i s'éliminent par sommation.

Pour calculer la probabilité linéaire $p_{1 \cdots l}$ sur l étages, il suffit comme dans le cas d'une caractéristique différentielle de multiplier les corrélations associées aux différentes relations et égales à $2 \cdot \epsilon_i$ entre elles :

$$p_{(1 \cdots l)} = \frac{1}{2} \pm \epsilon_{(1 \cdots l)} \text{ avec } \epsilon_{(1 \cdots l)} = \left(\prod_{i=1}^{l} \epsilon_i \right) \cdot 2^{l-1}$$

Le but est alors de rendre cette probabilité le plus éloignée possible de $\frac{1}{2}$ pour obtenir le meilleur biais possible et donc le meilleur distingueur possible.

On cherche donc une relation sur $(r-1)$ étages $L(X_0, X_{r-1}, [k_1, \cdots, k_{r-1}])$ de la forme :

$$L(X_0, X_{r-1}, [k_1, \cdots, k_{r-1}]) = a \cdot X_0 + b \cdot X_{r-1} = c \cdot [k_1, \cdots, k_{r-1}] \quad (*)$$

ayant une probabilité de se produire p le plus éloignée possible de $\frac{1}{2}$ et où $[k_1, \cdots, k_{r-1}]$ représente l'ensemble des sous-clés des $r-1$ étages.

À partir de cette relation qui représente un distingueur, on cherche à construire une attaque. Deux algorithmes ont été présentés par Matsui dans [Mat93]. Le premier s'attache à dériver le coefficient $c \cdot [k_1, \cdots, k_{r-1}]$. Sa probabilité de réussite est relativement élevée mais il ne permet d'attaquer que $(r-1)$ étages. Nous nous intéresserons donc ici au deuxième algorithme décrit dans [Mat93] qui représente un distingueur sur r étages et qui teste le biais de la relation $(*)$ pour les différentes hypothèses sur les valeurs de la sous-clé du dernier étage k_r en utilisant uniquement la valeur absolue des déviations par rapport à la probabilité uniforme.

Pour impliquer la sous-clé du dernier étage dans la relation $(*)$, il suffit de considérer X_{r-1} comme le déchiffré sur un étage de X_r. On obtient donc une relation du type :

$$a \cdot X_0 + b \cdot f_{k_r}^{-1}(X_r) = c \cdot [k_1, \cdots, k_{r-1}]$$

On utilise alors l'algorithme suivant pour tester les hypothèses de clés :

> **Première Étape :** Chiffrer N clairs pour obtenir une liste de (X_0, X_r)
> **Deuxième Étape :**
> Pour chaque valeur $k_r^{(i)}$ de k_r faire
> T_i reçoit le nombre de clairs tel que
> $a \cdot X_0 + b \cdot f_{k_r^{(i)}}^{-1}(X_r) = 0$
> T_{max} reçoit la valeur maximale des T_i et max le i correspondant
> T_{min} reçoit la valeur minimale des T_i et min le i correspondant
> Si $|T_{max} - \frac{N}{2}| > |T_{min} - \frac{N}{2}|$ alors
> k_r reçoit $k_r^{(max)}$
>   Si $p > \frac{1}{2}$ alors $c \cdot [k_1, \cdots, k_{r-1}] = 0$
>   Si $p < \frac{1}{2}$ alors $c \cdot [k_1, \cdots, k_{r-1}] = 1$
> Si $|T_{max} - \frac{N}{2}| < |T_{min} - \frac{N}{2}|$ alors
> k_r reçoit $k_r^{(min)}$
>   Si $p > \frac{1}{2}$ alors $c \cdot [k_1, \cdots, k_{r-1}] = 1$
>   Si $p < \frac{1}{2}$ alors $c \cdot [k_1, \cdots, k_{r-1}] = 0$

Il reste donc à évaluer la probabilité de réussite d'un tel algorithme. Pour cela, on utilise les lemmes suivants présentés dans [Mat93] qui donnent une bonne approximation de ce taux :

Lemme 1 *Soit N le nombre de clairs donné. Le taux de succès de l'algorithme ne dépend alors que du nombre de bits de clé impliqué dans l'expression $b \cdot f_{k_r}^{-1}(X_r)$ et de $\sqrt{N}|p - \frac{1}{2}|$.*

Lemme 2 *Soit $q^{(i)}$ les probabilités de réussite de l'équation suivante pour chaque valeur $k^{(i)}$ de k_r :*

$$b \cdot f_{k_r}^{-1}(X) = b \cdot f_{k^{(i)}}^{-1}(X)$$

alors si les $q^{(i)}$ sont indépendants, le taux de succès de l'algorithme est :

$$\int_{x=-2\sqrt{N}|p-\frac{1}{2}|}^{\infty}\left(\prod_{k^{(i)}\neq k_r}\int_{-x-4\sqrt{N}(p-\frac{1}{2})q^{(i)}}^{-x+4\sqrt{N}(p-\frac{1}{2})(1-q^{(i)})}\frac{1}{\sqrt{2\pi}}e^{-y^2/2}dy\right)\frac{1}{\sqrt{2\pi}}e^{-x^2/2}dx$$

Même si les $q^{(i)}$ ne sont pas indépendants dans le cas qui nous intéresse, ce lemme donne une bonne approximation du taux de succès de l'algorithme. Pour plus de lisibilité et comme présenté dans [Mat93], nous donnons le tableau suivant qui fait correspondre le nombre de texte clairs N et le taux de succès de l'algorithme :

N	$2\lvert p-\frac{1}{2}\rvert^{-2}$	$4\lvert p-\frac{1}{2}\rvert^{-2}$	$8\lvert p-\frac{1}{2}\rvert^{-2}$	$16\lvert p-\frac{1}{2}\rvert^{-2}$
Taux de Succès	48.6 %	78.5 %	96.7 %	99.9 %

Précisons également que ce distingueur peut être étendu à $(r+1)$ étages en ajoutant un étage au début et en considérant X_0 comme le chiffré sur un étage d'un texte clair P afin de faire intervenir la sous-clé k_0 du premier étage ajouté. Dans ce cas il ne reste plus qu'à appliquer à k_0 la deuxième étape de l'algorithme. Les taux de succès sont alors identiques et la complexité varie très peu.

Le distingueur présenté ici n'est qu'un exemple d'attaque utilisant les relations linéaires. La meilleure attaque connue du DES présentée dans [Mat94] permet de trouver les 56 bits de la clé secrète à l'aide de 2^{43} couples de clairs/chiffrés connus avec un taux de réussite de 85% grâce à un distingueur utilisant une relation portant sur les 14 tours centraux du DES. La complexité de cette attaque est donc de 2^{43} chiffrements DES. Cette attaque a été améliorée expérimentalement par P. Junod et S. Vaudenay (voir [Jun01]) : si la complexité reste la même que celle de l'attaque de Matsui, à savoir 2^{43} chiffrements DES, alors seulement $2^{42.5}$ clairs connus suffisent à cette attaque. On peut également en utilisant 2^{43} clairs connus obtenir une attaque de complexité 2^{39} chiffrements DES.

Comme dans le cas différentiel pour simplifier les calculs de recherche d'expressions linéaires, on définit le coefficient LP^F où F est une fonction de E dans E' par :

$$\mathrm{LP}^F(a,b) = (2\Pr(a\cdot X = b\cdot F(X)) - 1)^2$$

avec X appartenant à E et tiré au hasard, a appartenant à E et b à E'. On définit également l'ensemble L_F par

$$\mathrm{L}_F(a,b) = \{X\in E / a\cdot X = b\cdot F(X)\}.$$

Dans le cas où la fonction F est paramétrée par une clé k prise dans un espace $\mathcal{K}$, le coefficient LP dépend de la clé :

$$\mathrm{LP}^{F_k}(a,b) = (2\Pr(a\cdot X = b\cdot F_k(X)) - 1)^2$$

On calcule alors le coefficient ELP :

$$\mathrm{ELP}^F(a,b) = E_{k\in\mathcal{K}}(\mathrm{LP}^{F_k}(a,b))$$

où la fonction E désigne l'espérance mathématique sur l'espace des clés.

Lorsque l'on souhaite monter une cryptanalyse linéaire sur la fonction F_k^r à r étages, considérée ici comme le chiffrement à attaquer, on crée un distingueur sur $r-1$ étages à partir d'une relation linéaire sur $(r-1)$ étages dont la probabilité est $ELP^{F^{r-1}}(a,b)$, a et b étant les valeurs qui maximisent le coefficient ELP et F^{r-1} représentant la fonction F_k^r sur ses $(r-1)$ premiers étages. Le nombre de clairs N à chiffrer pour qu'une telle attaque réussisse avec un taux de succès de 99.9 % est alors égal à

$$N = \frac{16}{ELP^{F^{r-1}}(a,b)}.$$

En pratique et comme dans le cas de la cryptanalyse différentielle, si un algorithme de chiffrement est composé de r étages identiques constitués d'une série de boîtes S S_i, d'une permutation linéaire et d'une addition de clé, on cherche alors une relation linéaire parcourant les $(r-1)$ premiers tours du chiffrement dont la probabilité est calculée d'abord sur un étage grâce aux coefficients ELP^{S_i} et sur $r-1$ étages en multipliant entre elles ces différentes probabilités sur un étage.

2.2.4 Autres Exemples d'Attaques

Les deux grandes classes d'attaques présentées précédemment sont connues depuis longtemps. Il existe beaucoup d'autres attaques et d'autres classes d'attaques. Dans ce paragraphe, je ne m'intéresserai pas aux attaques dédiées qui ne sont pas généralisables, mais uniquement aux attaques génériques apparues après les cryptanalyses différentielle et linéaire.

Cryptanalyse Différentielle d'Ordre Supérieur

Cette attaque s'inspire largement de la cryptanalyse différentielle. Au lieu de considérer une dérivée d'ordre un selon une seule variable comme c'est le cas de la cryptanalyse différentielle, cette attaque s'intéresse à une dérivée d'ordre supérieur comme son nom l'indique. Cette cryptanalyse peut donc être considérée comme multivariable. Plus formellement et comme défini dans [Knu95], si on a une fonction f : S $\rightarrow$ T, où (S, $+$) et (T, $+$) sont des groupes abéliens, la dérivée de f au point a est définie par $\Delta_a f(x) = f(x+a) - f(x)$ et la dérivée i-ème de f aux points $a_1, \cdots, a_i$ est définie de manière récursive par $\Delta^{(i)}_{a_1,\cdots,a_i} f(x) = \Delta_{a_i}(\Delta^{(i-1)}_{a_1,\cdots,a_{i-1}} f(x))$. Une différentielle d'ordre i est alors la donnée d'un $i+1$-uplet $(\alpha_1, \cdots, \alpha_i, \beta)$ tel que $\Delta^{(i)}_{\alpha_1,\cdots,\alpha_i} f(x) = \beta$. Dans le cas d'une fonction f de $GF(2)^m$ dans $GF(2)^n$, il est nécessaire d'assurer l'indépendance linéaire des $a_1, \cdots, a_i$ afin d'éviter une dérivée d'ordre i nulle. Pour cela, on définit la différentielle d'ordre i de f sur le sous-espace vectoriel de $GF(2)^m$ $\mathcal{V}$ de dimension i par

$$\Delta_{\mathcal{V}} f(x) = \bigoplus_{\nu \in \mathcal{V}} f(x \oplus \nu)$$

L'existence de telles différentielles à travers le chiffrement permet de construire un distingueur même si des exemples d'une telle attaque sur des algorithmes de chiffrement sont peu nombreux. Précisons que ce type de cryptanalyse ne s'applique que lorsque le degré des expressions des valeurs intermédiaires est anormalement bas.

Citons ici l'exemple de l'attaque par différentielles d'ordre supérieur due à H. Tanaka, K. Hisamatsu et T. Kaneko [THK99] contre une version réduite de MISTY1 nommée M'1. MISTY est un algorithme de chiffrement par blocs introduit en 1996 par M. Matsui [Mat97] dont la sécurité contre les attaques linéaire et différentielle est prouvée. Il représente un bon compromis entre sécurité prouvée et rapidité d'exécution notamment en hardware. MISTY prend en entrée/sortie des blocs de 64 bits et itère sur huit étages d'un schéma de Feistel une fonction FO ainsi qu'une fonction linéaire FL très rapide. Il existe deux variantes de MISTY, MISTY1 et MISTY2 qui garantissent les mêmes bornes de sécurité prouvée. Nous décrirons ici MISTY1 qui contrairement à MISTY2 ne fait appel à la fonction FL que tous les deux étages et la version réduite de celui-ci, M'1 (voir figure 2.5). MISTY1 est construit de façon récursive sur trois niveaux. Les fonctions du premier niveau, nommées FI_{ij}, prennent en entrée un bloc de 16 bits qu'elles découpent en deux blocs de 7 et 9 bits et font intervenir sur chacun des deux petits blocs deux boîtes S S_7 et S_9 ainsi que des additions de clés et des opérations simples d'expansion et de réduction à travers une variante d'un schéma de Feistel, appelé schéma de MISTY. Les fonctions du deuxième niveau nommées FO_i qui portent sur des blocs de 32 bits font intervenir, selon le schéma de MISTY, trois fonctions FI_{ij} et des additions de clés. Le troisième niveau est un schéma de Feistel sur 8 étages. La version réduite M'1 correspond alors à une version de MISTY1 réduite à 5 étages sans aucun appel à la fonction FL (voir figure 2.5).

M'1 peut être attaqué en utilisant une propriété sur 4 étages construite à partir d'une différentielle d'ordre 7 [THK99]. On note G_K la fonction de chiffrement d'un bloc P de 64 bits vers la moitié droite de la sortie du quatrième étage dépendant de la clé maître K et G_K^{L7} la restriction de cette fonction aux 7 bits les plus à gauche de la sortie de 32 bits obtenue. P, le texte clair de 64 bits, est décomposé en trois parties : $(w_{25}||x||w_{32})$ où w_{25} est un mot de 25 bits, x un mot de 7 bits et w_{32} un mot de 32 bits. On peut alors énoncer ainsi la propriété utilisant une différentielle d'ordre 7 qui servira à construire un distingueur sur 5 étages :

$$\bigoplus_{x \in GF(2^7)} G_K^{L7}(P) = c$$

où c est une constante complètement indépendante de la clé K.

Cette attaque sur 5 étages permet d'obtenir de l'information sur 32 bits de clé à l'aide de 1408 clairs choisis et une complexité de 2^{17} chiffrements de M'1.

Cette attaque est rendue possible par le fait que le degré algébrique de la boîte S_7 est petit, il vaut 3. Plus exactement, la propriété qui rend possible une telle attaque est liée au fait que le degré du produit de deux éléments traversant la boîte S_7 n'atteint jamais 7, (voir section 2.3.2).

Cryptanalyse Utilisant des Différentielles Tronquées

Cette méthode consiste à considérer non pas des transitions entre des valeurs de différences entièrement spécifiées sous la forme de mots de un bloc mais à regarder des transitions entre des ensembles de valeurs de différences tels que ceux obtenus en fixant

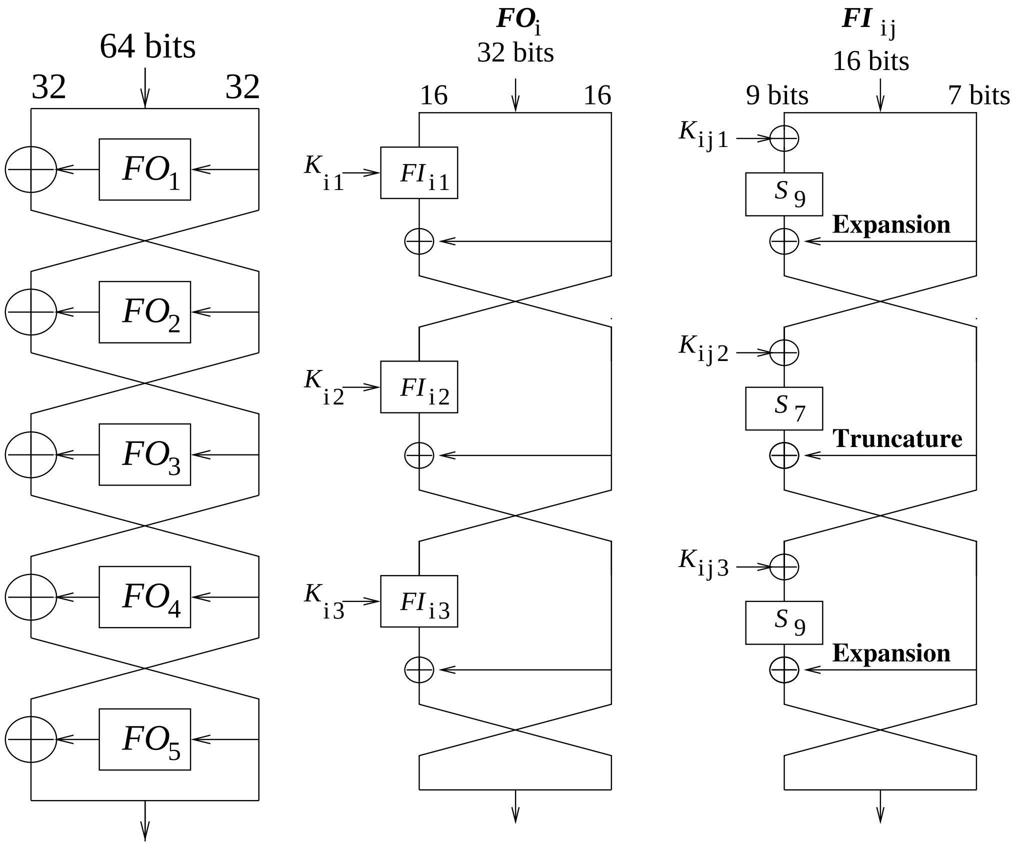


FIG. 2.5 – Le Chiffrement M'1 sur 5 Étages d'un Schéma de Feistel

une partie des mots de différence et en laissant la valeur du reste de ces mots varier librement.

Je prendrai ici l'exemple de la cryptanalyse utilisant des différentielles tronquées menée par M. Matsui et T. Tokita contre une version à huit étages de l'algorithme E2 sans transformations initiale et finale décrite dans [MT99] et qui utilise des transitions d'octets, à savoir que l'on ne s'intéresse ici qu'à des différences entre des octets de même position qu'on caractérise par la valeur 0 si les deux octets sont égaux et par la valeur 1 sinon.

E2 [KMA+98], candidat de l'AES soumis par les laboratoires de NTT (Nippon Telegraph and Telephone), est un algorithme de chiffrement par blocs qui prend en entrée/sortie des blocs de 128 bits. Il utilise un schéma de Feistel sur douze étages précédé d'une transformation initiale IT et suivi d'une transformation finale FT. La fonction étage F (voir figure 2.6), qui prend en entrée/sortie des blocs de 64 bits se compose d'une addition de clé (simple x-or bit à bit) sur chacune des huit octets, d'une boîte S agissant sur des octets, répétée huit fois, d'une partie linéaire définie par :

$$\begin{aligned}
d_1 &= c_2 \oplus c_3 \oplus c_4 \oplus c_5 \oplus c_6 \oplus c_7\\
d_2 &= c_1 \oplus c_3 \oplus c_4 \oplus c_6 \oplus c_7 \oplus c_8\\
d_3 &= c_1 \oplus c_2 \oplus c_4 \oplus c_5 \oplus c_7 \oplus c_8\\
d_4 &= c_1 \oplus c_2 \oplus c_3 \oplus c_5 \oplus c_6 \oplus c_8\\
d_5 &= c_1 \oplus c_2 \oplus c_4 \oplus c_5 \oplus c_6\\
d_6 &= c_1 \oplus c_2 \oplus c_3 \oplus c_6 \oplus c_7\\
d_7 &= c_2 \oplus c_3 \oplus c_4 \oplus c_7 \oplus c_8\\
d_8 &= c_1 \oplus c_3 \oplus c_4 \oplus c_5 \oplus c_8
\end{aligned}$$

où $c_1, \cdots, c_8$ représentent les huit octets d'entrée de la partie linéaire et $d_1, \cdots, d_8$ les huit octets de sortie de cette même transformation, d'une nouvelle addition de clé, de la boîte S agissant de nouveau 8 fois et d'une transformation de sortie qui décale d'une place vers la gauche tous les octets, le premier allant à la huitième place.

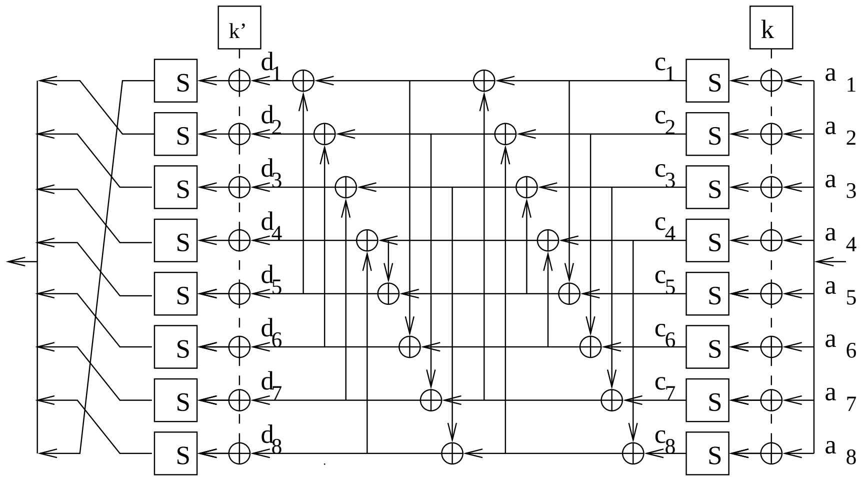


FIG. 2.6 – La Fonction Étage de E2

Le principe de la cryptanalyse menée contre une version réduite de E2 consiste à regarder sur la fonction d'étage comment se propagent des différences entre octets, sans tenir compte de la valeur de cette différence en distinguant seulement les cas où elle est ou non égale à 0. Les auteurs nomment les propriétés étudiées "caractéristiques d'octets". Les deux caractéristiques utiles pour la cryptanalyse sur la version de E2 réduite à huit étages sont :

$$\begin{aligned}
(00000000) &\rightarrow (00000000)\\
(10010100) &\rightarrow (10010100)
\end{aligned}$$

où l'expression entre parenthèse que nous noterons de façon générique $(\Delta a_1, \cdots, \Delta a_8)$ porte sur huit octets de différence entre deux blocs d'entrée ou de sortie de la fonction d'étage de E2 et indique si ces octets de différences sont ou non égaux à zéro.

La première caractéristique se produit avec une probabilité égale à un. Quant à la deuxième, elle se produit avec une probabilité égale à 2^{-16} dans le cas où $\Delta a_1 = \Delta a_4 = \Delta a_6$. Elle peut être utilisée pour construire un distingueur sur 7 étages du schéma de Feistel de E2 : la probabilité que si la différence d'entrée sur les octets est de la forme (10010100 00000000) avec $\Delta a_1 = \Delta a_4 = \Delta a_6$, la différence de sortie soit de la même forme, est de 2^{-104}. Cette probabilité tient également compte des effets du x-or entre deux éléments de la forme (10010100) à la fin du 6ème étage du schéma de Feistel. On souhaite en effet obtenir un élément de la forme (00000000) en sortie. La probabilité de (10010100) $\oplus$ (10010100) = (00000000) est de $\left(\frac{1}{255}\right)^3 \approx 2^{-24}$. Il reste cependant un problème, car la probabilité des fausses alarmes, c'est à dire de l'occurence d'un couple de valeurs de sortie dont la différence soit de la forme (10010100 00000000), est elle aussi égale à 2^{-104}. Cette difficulté peut être surmontée en constatant que dans le cas où $\Delta a_1 = \Delta a_4 = \Delta a_6$ dans la différence (10010100) la probabilité de traverser la deuxième fonction d'étage n'est pas de 2^{-16} mais de $9,3 \cdot 2^{-16}$ en moyenne. Cela est dû au fait qu'en moyenne, on obtient de meilleurs résultats pour la probabilité $\Pr[S(x \oplus \Delta x) \oplus S(x) = S(y) \oplus S(y \oplus \Delta y)]$ de la boîte S lorsque $\Delta x = \Delta y$.
Donc sur sept étages, si on chiffre 2^{104} couples de clairs choisis $P = (P_1, \cdots, P_{16})$ et $P' = (P'_1, \cdots, P'_{16})$ avec $\Delta P_1 = \Delta P_4 = \Delta P_6$ et tels que tous les autres octets soient égaux alors le nombre de chiffrés qui ont une différence de la forme (10010100 00000000) est 9,3.

Dans le cas d'une attaque sur 8 étages, on utilise la caractéristique sur 7 étages et on déchiffre le dernier étage en testant toutes les hypothèses de la sous-clé du huitième étage. La bonne clé est celle qui apparaît le plus souvent. Pour être sûr d'éviter les fausses alarmes, il est nécessaire de chiffrer $2^{107} = 8 \times 2^{104}$ clairs de la forme souhaitée. La complexité d'une telle attaque peut être réduite à une valeur inférieure à 2^{128} en écartant les paires impossibles et en introduisant une méthode de comptage à 2^{64} entrées sur la sous-clé du deuxième étage.

Nous venons de voir une cryptanalyse différentielle tronquée orientée octets qui s'intéresse non à la valeur de différence entre octets mais juste aux patterns d'égalité entre octets.

Attaques par Saturation ou Cryptanalyse Intégrale

L'attaque dite par saturation a été introduite par J. Daemen, V. Rijmen et L. Knudsen dans [DKR97] mais ce n'est qu'en 2001 que ce type d'attaque a pris ce nom dans un article de S. Lucks [Luc01]. Notons également que le nom proposé par L. Knudsen dans [Knu02] à FSE'02 et plébiscité par les participants de cette conférence est "Cryptanalyse Intégrale". Il s'agit ici de "saturer" un ou plusieurs octets, c'est à dire de faire prendre à une partie du message toutes les valeurs possibles et d'utiliser les propriétés induites par la fonction itérée (par exemple sa semi-bijectivité, c'est à dire qu'après un certain nombre d'étages, les octets du chiffré obtenu peuvent être vus comme une somme de bijection des octets du clair) pour créer un distingueur. Ces attaques peuvent également être vues comme un cas particulier de la classe des attaques différentielles d'ordre supérieur.

Deux attaques de ce type, qui seront développées au chapitre 4, sont à ce jour connues contre Rijndael.

Je prendrai ici l'exemple de trois étages de SAFER que j'ai étudié pendant cette thèse.

Rappelons que SAFER prend en entrée un bloc de 64 bits, découpé en 8 octets. Si on fait prendre à deux octets d'entrée toutes les valeurs possibles comprises entre 0 et 255, les autres octets d'entrée étant fixés, au bout de deux tours, la somme modulo 256 des $(256)^2$ valeurs est nulle. En effet, si on ne sature qu'un seul octet à l'entrée du chiffrement, au bout de deux tours, on obtient des ensembles de 32 valeurs pour chacun des huit octets, cela étant lié à l'alternance des x-or avec les clés et des additions classiques et au choix des coefficients de la matrice de la partie linéaire. Il est donc nécessaire de saturer deux octets pour obtenir une propriété exploitable dans un distingueur sur deux tours et ainsi de l'information sur la clé du troisième étage en testant les valeurs de clé pour laquelle la somme précédente est nulle. La valeur de clé qui apparaît le plus souvent est celle recherchée. Une telle attaque nécessite, en théorie, le chiffrement de $(256)^2$ clairs choisis. Pour être sûr d'éviter les fausses alarmes, il est plus prudent de répéter deux fois cette opération en changeant la valeur des 6 octets d'entrée constants.

Dans leur article [BS01], A. Biryukov et A. Shamir ont donné une première formalisation de ce type de phénomènes, en étudiant les propriétés des schémas du type $S_1A_1S_2A_2S_3$ où S_i désigne un étage composé de k boîtes S qui prennent en entrée/sortie des mots de m bits et A_i représente une transformation affine inversible de vecteurs de longueur $n = k \cdot m$ sur $GF(2)$. On peut donc écrire : $A_i(x) = L_i \cdot x \oplus B_i$ où L_i est une matrice $n \times n$ de $GF(2)$ et B_i un vecteur de taille n de $GF(2)$. Les auteurs de [BS01] démontrent, dans ce cas, que la saturation d'un mot de taille m en entrée entraîne en sortie de l'application $S_1A_1S_2A_2S_3$ la propriété dite d'équilibre, à savoir que $\bigoplus_{i=0}^{2^m-1} S_3(A_2(S_2(A_1(S_1(((\lambda_1, \lambda_2, \cdots, \lambda_{k-1}, i)))))) = 0$ où les λ_j sont des constantes de $GF(2^m)$. Cette propriété ne peut être appliquée à SAFER, car sa partie linéaire ne peut s'exprimer comme une transformation sur $GF(2)^n$.

Attaques par Interpolation

L'attaque par interpolation a été introduite par T. Jacobsen et L. Knudsen dans [JK97]. Cette attaque monovariable est applicable aux algorithmes dont les sorties peuvent être exprimées comme un polynôme de faible degré algébrique n en les entrées, ce qui peut par exemple se produire lorsque le degré de l'algorithme étudié est égal au produit des degrés algébriques des boîtes S de chaque étage.

La propriété d'interpolation ainsi obtenue peut être utilisée comme un distingueur à r étages dans une attaque contre un chiffrement à $r + 1$ étages ou plus. Les attaques par interpolation peuvent être considérées comme un cas particulier monovariable des attaques par linéarisation, applicables lorsque le nombre de monômes des expressions multivariables des sorties en fonction des entrées est suffisamment faible.

Dans l'article initial, T. Jacobsen et L. Knudsen donnent l'exemple d'un chiffrement nommé $\mathcal{PURE}$ composé de r étages d'un schéma de Feistel, dont la fonction d'étage F_K est définie de $GF(32)$ dans lui-même par : $F_K(x) = f(x \oplus k)$ avec $f(x) = x^3$ dans $GF(32)$. Ce chiffrement est prouvé sûr contre les cryptanalyses différentielle et linéaire pour un nombre suffisant d'étages. En revanche, l'attaque par interpolation fonctionne pour un nombre d'étages allant jusqu'à 32.

Pour cette attaque par rencontre dans le milieu, on exprime tout d'abord la sortie de l'étage s que l'on notera z $(s \leq r - 1)$ comme un polynôme $g(x)$ fonction du texte clair x

composé de au plus n_1 coefficients et dont on peut estimer le degré ici égal à au plus $n_1 - 1$ par multiplication du degré des boîtes S successivement traversées. On exprime ensuite cette même sortie z comme un polynôme $h(y)$ en la sortie y du dernier étage composé de au plus n_2 coefficients. On a alors $g(x) = h(y)$. Cette équation dont le nombre de coefficients inconnus est $n = n_1 + n_2$ permet de construire un système en au plus $n - 2$ inconnues car on suppose que les termes constants valent 0 et que les coefficients de plus haut degré valent 1 afin d'obtenir une solution unique et non-trivial. Le nombre de paires de clairs/chiffrés nécessaires pour résoudre ce système est alors égal à $n - 2$. On chiffre un texte clair de plus afin de vérifier que le système trouvé est le bon. Pour obtenir la valeur de y, on applique l'inverse de la fonction du r-ième étage au chiffré C pour chaque valeur des b bits de la clé k_r impliqués dans la relation entre C et y, et pour chacune de ces valeurs on calcule les coefficients du système. La bonne valeur est celle pour laquelle la vérification faite à partir de la paire supplémentaire de clair/chiffré fonctionne. La complexité d'une telle attaque est donc de $2^{b-1}(n-1)$ chiffrements pour $(n-1)$ paires de clairs/chiffrés.

Dans le cas de l'algorithme $\mathcal{PURE}$ à 32 étages, on fixe la partie droite de 32 bits de l'entrée et on exprime la sortie de gauche du 22ème étage comme un polynôme de degré au plus 3^{20} de la partie gauche de l'entrée. Puis, on exprime la sortie du 31ème étage comme un polynôme dont le nombre de coefficients est au plus de 3^{19}. Le nombre de coefficients de l'équation obtenue par l'égalité des deux polynômes est d'au plus 2^{32}. Cette attaque par interpolation permet de retrouver 32 bits d'information sur la clé du dernier étage. Elle utilise 2^{32} clairs choisis et sa complexité est de 2^{63} calculs.

2.3 Résistance aux Classes d'Attaques

Peu après la découverte des attaques différentielle et linéaire, la recherche en cryptographie s'est tout naturellement orientée vers l'investigation de méthodes permettant de résister à ces attaques. Nous présenterons donc ici différentes techniques de preuves qui permettent de prémunir les algorithmes contre ces attaques.

2.3.1 Sécurité Prouvable contre les Cryptanalyses Différentielle et Linéaire

Dans leur article [NK95], K. Nyberg et L. Knudsen donnent une borne supérieure de sécurité prouvée pour un schéma de Feistel de trois étages et plus dans le cas où la fonction étage est une permutation et de quatre étages et plus dans le cas où la fonction étage est une fonction, les sous-clés étant supposées indépendantes et uniformément distribuées. Soit :

$$\begin{aligned}
f &: GF(2)^m \to GF(2)^n, m \geq n \\
E &: GF(2)^n \to GF(2)^m, \text{ une expansion affine} \\
F &: GF(2)^n \times GF(2)^m \to GF(2)^n \\
&\quad (X, k_i) \to f(E(X) \oplus k_i)
\end{aligned}$$

où k_i appartenant à $GF(2)^m$ désigne la sous-clé de l'étage i. On considère un chiffrement de type DES résultant d'un schéma de Feistel à r étages qui prend en entrée des blocs de taille $2n$ et applique la fonction F à la moitié du bloc courant. On définit alors p_{max} comme la meilleure probabilité d'une différentielle sur un tour :

$$p_{max} = \max_\beta \max_{\alpha_R \neq 0} \mathrm{DP}^F(\alpha, \beta)$$

où α_R représente la partie droite de α.

Théorème 2 *Dans un chiffrement de type DES utilisant la fonction F comme fonction étage avec des clés k_i indépendantes et uniformément distribuées, la probabilité d'une différentielle sur s tours ($s \geq 4$) est inférieure ou égale à $2p_{max}^2$:*

$$\forall \alpha \neq 0 \,, \forall \beta \,, P(\Delta X_s = \beta | \Delta X_0 = \alpha) \leq 2p_{max}^2$$

où ΔX_0 représente la différence d'entrée et ΔX_s la différence à la sortie de l'étage s.

Théorème 3 *Si f est une permutation dans un chiffrement de type DES et si les sous-clés sont indépendantes et uniformément distribuées, alors la probabilité d'une différentielle sur s tours ($s \geq 3$) est inférieure ou égale à $2p_{max}^2$.*

Notons que pour le cas de la résistance à la cryptanalyse linéaire étudiée par K. Nyberg dans [Nyb94], on obtient les mêmes bornes en définissant

$$p_{max} = \max_\alpha \max_{\beta \neq 0} \mathrm{LP}^F(\alpha, \beta)$$

Cette borne a été améliorée à p_{max}^2 par K. Aoki et K. Ohta [AO97] dans le cas différentiel comme dans le cas linéaire avec des fonctions d'étage bijectives.

M. Matsui a conçu son chiffrement MISTY [Mat97] pour qu'il résiste de façon efficace aux cryptanalyses différentielle et linéaire à l'aide de cette théorie. Rappelons tout d'abord que MISTY agit sur trois niveaux (voir figure 2.5). Le premier est un schéma de Feistel classique dont la fonction étage est FO_i. Le deuxième représente la fonction FO_i et est un schéma de Feistel modifié sur trois étages faisant intervenir trois fois une fonction FI_{ij}. Le troisième niveau représente la fonction FI_{ij} sur trois étages d'un schéma de Feistel modifié séparant en deux blocs de taille n_1 et n_2 le bloc d'entrée et faisant intervenir deux boîtes S S_7 et S_9 et des opérations élémentaires de troncature et d'expansion à 0 (voir figure 2.5). M. Matsui prouve tout d'abord une borne supérieure pour chaque fonction FI_{ij} : si f_1, f_2 et f_3 sont les trois fonctions étages bijectives du schéma de Feistel modifié représentant une fonction FI_{ij} et si $p_k = EDP^{f_k}$ (respectivement ELP^{f_k}) alors $EDP^{FI_{ij}}$ (respectivement $ELP^{FI_{ij}}$) est plus petit que $\max\{p_1p_2, p_2p_3, 2^{n_1-n_2}p_1p_3\}$. De même, pour chaque fonction FO_i, si les trois fonctions FI_{ij} sont bijectives et si pour tout j valant 1, 2 ou 3 on a $EDP^{FI_{ij}} \leq p$ (respectivement $ELP^{FI_{ij}} \leq p$) alors $EDP^{FO_i} \leq p^2$ (respectivement $ELP^{FO_i} \leq p^2$).

Les fonctions étage de FI sont ici les deux boîtes S S_7 et S_9, S_9 intervenant deux fois, et on a :

$$LP^{S_7} = DP^{S_7} = 2^{-6} \text{ et } LP^{S_9} = DP^{S_9} = 2^{-8}.$$

On obtient donc pour chaque fonction FI_{ij} une borne $ELP^{FI_{ij}} = EDP^{FI_{ij}} \leq 2^{-14}$. De même, pour chaque fonction FO_i, en appliquant le deuxième résultat aux trois fonctions FI_{ij}, on obtient : $ELP^{FO_i} = EDP^{FO_i} \leq 2^{-28}$. Et donc sur trois étages d'un schéma de Feistel composé de trois fonctions FO_i, on obtient une borne supérieure pour les probabilités différentielle et linéaire égale à 2^{-56}, ce qui prémunit MISTY composé de 8 étages contre les attaques différentielle et linéaire.

2.3.2 Sécurité des Boîtes S

Les cryptologues se sont intéressés aux conditions que devaient remplir la fonction d'étage itérée f, considérée ici comme une fonction de $GF(2)^m$ dans $GF(2)^n$, pour résister aux cryptanalyses différentielle et linéaire. Dans le cas de la cryptanalyse différentielle, on cherche à rendre le coefficient

$$\delta_f = \max_{\alpha \neq 0, \beta} \mathrm{DP}^f(\alpha, \beta)$$

le plus petit possible. Dans le cas linéaire, on s'intéresse au coefficient

$$\lambda_f = \max_{\alpha, \beta \neq 0} \mathrm{LP}^f(\alpha, \beta).$$

Précisons qu'en règle générale, le caractère non linéaire de la fonction d'étage f provient de la ou des boîtes S utilisées. Il suffit donc d'étudier les coefficients δ_S et λ_S uniquement pour la ou les boîtes S.

De plus, on a les propriétés suivantes :

- Les valeurs δ_f et λ_f restent inchangées si l'on compose f à gauche et à droite avec des permutations linéaires.
- Si f est bijective, on a :
$$\delta_{f^{-1}} = \delta_f \text{ et } \lambda_{f^{-1}} = \lambda_f$$

Dans de nombreux algorithmes récents comme Rijndael, MISTY, E2 ou Camellia, la ou les boîtes S sont en général la ou les composées d'une permutation linéaire et d'une exponentiation, c'est à dire d'une permutation de $GF(2^m)$ qui à la variable x fait correspondre la sortie x^α avec $\mathrm{pgcd}(\alpha, 2^m - 1) = 1$ afin de conserver le caractère bijectif de l'opération. L'étude de la résistance de la fonction étage f aux cryptanalyses différentielle et linéaire se ramène donc souvent à l'étude des exposants particuliers utilisés dans les boîtes S en cherchant toujours à rendre minimum les coefficients δ_S et λ_S.

On a les résultats suivants sur les meilleures bornes qui puissent être atteintes (voir par exemple [CV94]) :

Théorème 4 *Pour toute fonction f de $GF(2)^m$ dans $GF(2)^n$, on a :*

- $\delta_f \geq 2^{-n}$. *En cas d'égalité, f est dite parfaitement non-linéaire (PN).*
- $\lambda_f \geq 2^{-m}$. *En cas d'égalité, f est dite courbe (B).*

Les fonctions qui atteignent l'une ou l'autre des deux bornes sont identiques. En effet, d'après [Nyb91], on a :

Théorème 5 *Une fonction est parfaitement non-linéaire si et seulement si elle est courbe.*

Des fonctions qui atteignent ces bornes n'existent que pour certaines valeurs de m et n [Nyb91] :

Théorème 6 *Les fonctions B ou PN n'existent que si m est pair et si $m \geq 2n$.*

On s'intéresse également à des fonctions ayant des propriétés légèrement moins fortes, étudiées notamment dans [CV94].

Théorème 7 *Pour toute fonction f de $GF(2)^m$ dans $GF(2)^n$, on a :*

- $\delta_f \geq 2^{1-m}$. *En cas d'égalité, f est dite presque parfaitement non-linéaire (APN).*

– $\lambda_f \geq 2^{1-m}\left(1+\frac{(2^{n-m}-1)(2^m-1)}{2^n-1}\right)$. *En cas d'égalité, f est dite presque courbe (AB).*

Les conditions sur l'existence de telles fonctions sont les suivantes [Nyb91] :

Théorème 8

– *Des fonctions AB n'existent que si $m = n$ et n est impair et ces fonctions vérifient également la propriété APN.*
– *Des fonctions APN n'existent que si $m = 2$ et $n = 1$ ou $n \geq m$.*

Un exemple de fonction presque parfaitement non-linéaire et presque courbe est la fonction $f(x) = x^{2^{2i}-2^i+1}$ sur $GF(2^n)$ avec n impair et pgcd$(n, i) = 1$, dont les propriétés ont été étudiées par T. Kasami dans [Kas71].

Dans le cas où $n = 7$ et $i = 2$, on obtient l'exposant $d = 13$ auquel on associe la classe C de ses conjugués (c'est à dire l'ensemble des valeurs $2^i d$ et $2^i d^{-1}$ modulo $(2^7 - 1)$ pour i variant de 0 à 6) qui définissent également des fonctions presque parfaitement non-linéaires et presque courbes d'après les propriétés vues précédemment. On a donc

$$C_{n=7,13} = \{81, 35, 70, 13, 26, 52, 104, 69, 11, 22, 44, 88, 49, 98\}$$

L'exposant 11 qui appartient à cette classe représente la fonction puissance de Welch définie par $f(x) = x^{2^{\frac{n-1}{2}}+3}$ qui est également presque courbe et donc presque parfaitement non-linéaire [CCD00a]. Ces propriétés particulières de la fonction, conjecturées par Welch, ont été démontrées par A. Canteaut, P. Charpin et H. Dobbertin dans une série de leur travaux [CCD99] et [CCD00b]. L'exposant le plus intéressant de $C_{n=7,13}$ reste pourtant 81 qui est celui utilisé par M. Matsui dans la boîte S_7 de Misty [Mat97]. D'après ce qui précède, il est évident que cet exposant garantit à la boîte S_7 une bonne résistance aux cryptanalyses différentielle et linéaire. Cependant, d'autres propriétés indésirables de ces fonctions optimales utilisées dans les boîtes S notamment celles de MISTY rendent des versions réduites des algorithmes de chiffrement par blocs reposant sur ces boîtes vulnérables à d'autres formes de cryptanalyses, notamment la cryptanalyse différentielle d'ordre supérieur, MISTY ne dérogeant pas à la règle comme nous avons pu le voir dans la section 2.2.4.

Rappelons ici que les attaques différentielles d'ordre supérieur ne sont possibles que dans le cas où le degré de non-linéarité des valeurs intermédiaires est faible. On obtient une borne supérieure de ce degré grâce au spectre de Walsh, comme démontré dans [CV02].

Définition 5 *Soit F une fonction de $GF(2)^n$ dans lui-même. Pour tout α de $GF(2)^n$, on définit la fonction linéaire φ_α de $GF(2)^n$ dans $GF(2)$ par $\varphi_\alpha(x) = \alpha \cdot x$ où $\cdot$ désigne le produit scalaire usuel. Le spectre de Walsh de la fonction F est alors l'ensemble*

$$\{\mathcal{F}(\varphi_\alpha \circ F + \varphi_\beta,\ \alpha \in GF(2)^n\ \{0\},\ \beta \in GF(2)^n\}$$

où $\mathcal{F}(f)$ désigne la corrélation de f avec la fonction nulle définie par :

$$\mathcal{F}(f) = \sum_{x \in GF(2)^n} (-1)^{f(x)} = 2^n - w(f)$$

où f est une fonction booléenne de $GF(2)^n$ dans $GF(2)$ et où $w(f)$ désigne le poids de Hamming de f.

Définition 6 *Le spectre de Walsh d'une fonction F de $GF(2)^n$ dans lui-même est dit 2^l-divisible si toutes ses valeurs sont divisibles par 2^l.*

Théorème 9 *Soit F une fonction de $GF(2)^n$ dans lui-même. Si le spectre de Walsh de F est 2^l-divisible alors $deg(F) \leq n - l + 1$.*

A. Canteaut et M. Videau démontre également le théorème suivant :

Théorème 10 *Soit F une fonction de $GF(2)^n$ dans $GF(2)^n$ telle que son spectre de Walsh soit 2^l-divisible alors, le degré du produit de t composantes booléennes de F est au plus $n - l + t$.*

Dans le cas particulier de MISTY, la fonction utilisée dans la boîte S_7 est AB, on en déduit donc d'après [CV94] et comme rappelé dans [CV02] que les valeurs contenues dans son spectre de Walsh sont 0 et $2^{\frac{n+1}{2}}$ uniquement et que celui-ci est donc 2^4-divisible. On en déduit donc d'après le théorème précédent, que le degré du produit de deux composantes booléennes de la boîte S_7 de MISTY est inférieur ou égal à 5. Ce résultat est également démontré dans [BF00]. C'est cette propriété particulière de la boîte S_7 (voir [BF00] et [CV02]) qui rend M'1, version réduite de MISTY, vulnérable à l'attaque différentielle d'ordre 7 décrite au début de la section 2.2.4. Précisons également comme prouvé dans [CV02] que l'exposant choisi dans la boîte S de l'AES (l'inversion dans le corps $GF(2^8)$) est l'une des rares fonctions a ne pas avoir cette propriété indésirable, à savoir que tout en étant une fonction APN, elle n'a pas un spectre de Walsh 2^l-divisible, ce qui prémunit l'AES contre une attaque différentielle d'ordre supérieur.

Si les propriétés APN et AB garantissent une protection efficace contre les cryptanalyses différentielle et linéaire, il ne faut pas oublier que d'autres attaques existent. Il est également nécessaire lorsque l'on construit une boîte S à l'aide d'une fonction puissance de choisir un exposant dont le degré algébrique est suffisamment grand pour se prémunir contre une attaque par interpolation.

2.4 Nouvelles Propositions pour Résister aux Classes d'Attaques Connues

Les critères vus précédemment portent essentiellement sur les propriétés de confusion que doit remplir une bonne fonction étage. Nous nous sommes plus particulièrement intéressés au rôle de la boîte S dans la résistance aux attaques connues. L'autre grand critère que doit respecter un bon algorithme de chiffrement par blocs, déjà défini par Shannon en 1949, est la diffusion. Nous allons donc présenter dans cette section deux méthodes qui permettent de garantir à la fonction étage une bonne diffusion. La première méthode introduite par S. Vaudenay dans [SV94] fait appel à ce qu'il nomme des multipermutations, généralisation du graphe FFT dont une illustration est le réseau de la partie linéaire de l'algorithme SAFER. S. Vaudenay utilise le principe des multipermutations dans l'algorithme CS-Cipher. La deuxième méthode est la théorie nommée "Wide Trail Strategy" dont l'idée principale est de rendre maximal le nombre de boîtes S "actives", c'est à dire d'assurer une diffusion maximale des caractéristiques différentielles ou linéaires. C'est cette théorie qui a conduit au choix de la partie linéaire de l'AES.

2.4.1 Les Multipermutations

Le concept de multipermutation a été introduit par S. Vaudenay et C. Schnorr dans un article présenté à Eurocrypt'94 [SV94], l'idée étant de créer des boîtes de diffusion parfaite. Son lien avec ce qu'on nomme le graphe FFT (transformation de Fourier rapide) semble évident. Le graphe FFT ou schéma papillon fait dans sa forme la plus simple correspondre 2 entrées à 2 sorties, chaque sortie étant influencée à travers des permutations par les deux entrées. La diffusion est donc ici maximale : toute sortie est liée au nombre maximal d'entrées. On peut généraliser ce graphe à 2^n entrées et 2^n sorties à l'aide d'un réseau de connexions reliant n étages de 2^{n-1} boîtes 2×2 tel que celui utilisé dans la partie linéaire de SAFER.

Cependant, les critères que doivent vérifier les multipermutations sont plus forts que ceux vérifiés dans le cas particulier de SAFER où la boîte élémentaire 2×2 n'est pas une multipermutation, L'idée principale des multipermutations étant de maximiser le nombre de sorties modifiées lors de la modification d'un certain nombre d'entrées.

On définit donc une multipermutation comme suit :

Définition 7 *Soit f une fonction de l'ensemble Z^r dans Z^n. On dit que f est une (r,n)-multipermutation sur Z si deux $(r+n)$-uplets différents de la forme $(x, f(x))$ ne peuvent coïncider sur r positions différentes.*

Cela signifie en particulier que si deux entrées x et x' de taille r diffèrent sur une position, les sorties correspondantes de taille n doivent être différentes sur n positions, c'est à dire que toutes les entrées influencent toutes les sorties. La diffusion est donc bien maximum.

Plus généralement, la modification de m entrées entraîne la modification d'un nombre de sorties au moins égal à $n + 1 - m$.

On peut également donner la définition équivalente suivante liée aux codes correcteurs d'erreurs :

Définition 8 *Une (r,n)-multipermutation sur Z est une fonction f de Z^r dans Z^n telle que l'ensemble des $(r+n)$-uplets de la forme $(x, f(x))$ soit un code de distance minimale $n+1$.*

Rappelons qu'un code correcteur d'erreur de distance d sur Z est un sous-espace de dimension k de l'espace Z^n où deux vecteurs de l'espace d'arrivée ont une distance de Hamming au moins égale à d, d étant le plus petit nombre vérifiant cette propriété. On caractérise donc un code par la donnée de l'espace Z et du triplet $[n, k, d]$. Pour un code de longueur n et de dimension k données, la distance minimale est majorée par la borne dite de Singleton :

$$d \leq n - k + 1$$

Dans le cas où l'ensemble Z est un corps de caractéristique q, si on note f une (r,n)-multipermutation linéaire, la donnée de l'ensemble des mots $(x, f(x))$ définit un code MDS (Maximal Distance Separable) de longueur $(n+r)$ et de taille q^r, c'est à dire un code qui atteint la borne de Singleton et dont la distance minimale est maximale.

Pour une meilleure compréhension de ce qu'est une multipermutation, l'exemple le plus simple est une multipermutation à une entrée et à n sorties qui est en fait une famille de n permutations de l'entrée.

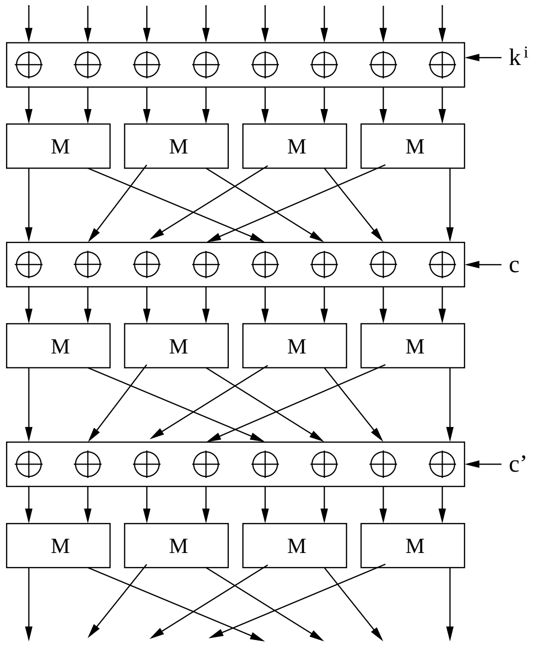


FIG. 2.7 – Un Étage de CS-Cipher

Un exemple de mise en oeuvre de la notion de multipermutation est le chiffrement CS-Cipher décrit dans [SV98], composé de 8 étages, qui prend en entrée/sortie des mots de 64 bits découpés en octets.

Un étage (voir figure 2.7) se compose de trois transformations répétées trois fois. La première est un x-or avec une sous-clé pour le premier sous-étage et avec des constantes pour les deux autres sous-étages. La deuxième transformation M est une $(2,2)$-multipermutation prenant en entrée/sortie des couples d'octets, suivie d'une permutation P sur les octets tenant le rôle de boîte S. La troisième transformation est une permutation des octets selon un graphe FFT sur les octets qui permet une diffusion maximale de l'information.

La fonction M prend en entrée une chaîne de 16 bits x scindée en deux octets $x_l||x_r$ et calcule $M(x) = y_l||y_r$ avec $y_l = P(\phi(x_l) \oplus x_r)$ et $y_r = P(R_l(x_l) \oplus x_r)$ où $\phi(x_l) = (R_l(x_l) \wedge 55_{hex}) \oplus x_l)$, R_l étant la rotation à gauche de 1 bit. Il s'agit bien d'une $(2,2)$-multipermutation. La modification d'une entrée entraîne la modification de toutes les sorties. La fonction inverse nécessaire au déchiffrement définie par $(x_l||x_r) = M^{-1}(y_l||y_r)$ peut être calculée de la manière suivante :

$$x_l = \phi'(y_l) \oplus P(y_r) \text{ et } x_r = R_l(x_l) \oplus P(y_r)$$

avec $\phi'(x) = (R_l(x) \wedge aa_{hex}) \oplus x$.

La diffusion dans CS-Cipher est donc maximale, car elle combine les propriétés des multipermutations à celles d'un schéma FFT.

2.4.2 La "Wide Trail Strategy" et le "Branch Number"

Nous présentons ici une deuxième théorie sur les méthodes de diffusion de la partie linéaire développée dans [Dae95], [DR99] et [DR02] .
L'idée de départ exposée dans la thèse de J. Daemen [Dae95], connue sous le nom de "Wide Trail Strategy", est de choisir dans un premier temps des boîtes S avec des coefficients DP et LP très petits, puis de choisir une partie linéaire qui propage très bien les différences, c'est à dire qui implique le passage des caractéristiques différentielles ou linéaires dans un grand nombre de boîtes S. La fonction d'étage est alors choisie de telle façon que si, à une certaine étape, le nombre de boîtes S actives est petit, à l'étage suivant il doit être beaucoup plus grand.

Cette idée formulée en 1995 reste floue mais a été formalisée plus précisément dans [DR99] et dans [DR02]. Dans toute cette section, on notera F la transformation linéaire agissant sur un vecteur d'octets à n composantes dont on étudie les propriétés. F sera composée de deux transformations : θ une permutation linéaire de n octets qui garantit la plus haute diffusion possible et π une transformation qui garantit une très grande dispersion, c'est à dire qui "éloigne" les éléments actifs les uns des autres.

Nous nous intéresserons tout d'abord aux critères de choix de θ qui dans le cas de Rijndael correspond à la transformation MixColumn. Pour cela, on définit alors le poids d'un vecteur d'octets a comme étant son nombre d'octets non nuls. On note ce poids $W(a)$. On définit alors :

Définition 9 *Le nombre de branches d'une transformation linéaire ϕ est*

$$Nb = min_{a \neq 0}(W(a) + W(\phi(a))).$$

Cette valeur peut être vue comme une mesure de la puissance de diffusion de l'application ϕ. Lorsque ce nombre est maximal, on dit qu'il vérifie la propriété MDS (Maximal Distance Separable), c'est à dire que $Nb = n + 1$.

Dans le cas de l'algorithme Rijndael, le design de la transformation MixColumn répond aux différents impératifs de cette stratégie de construction. MixColumn ici égale à θ est la donnée de la matrice M, décrite dans la section 2.1.3 et qui définit un code MDS. En effet, la matrice circulante M de taille 4×4, construite comme une matrice de code cyclique à partir d'un polynôme de degré 3, prend en entrée/sortie un vecteur d'octets de taille $n = 4$. On peut voir cette transformation comme un code linéaire sur le corps $GF(2^8)$ dont les mots de code sont les éléments $(x, M \cdot x)^T$ où x est un vecteur d'octets de taille 4. Les paramètres de ce code sont donc $[8, 4, d]$. Ce code est MDS et atteint la borne de Singleton : $d = 5$. Sa matrice génératrice est de la forme $[IM]$ où I est la matrice identité de taille 4×4. Le nombre de branches associé à la transformation MixColumn est, d'après [DR02], égal à la distance minimum du code généré, on a donc ici $Nb = d = 5$ qui est maximum. On peut également voir cette transformation comme une (4,4)-multipermutation.

Quant à la transformation π qui compose également la partie linéaire, elle correspond à l'opération ShiftRow décrite à la section 2.1.3 et consiste en un décalage des lignes de la matrice à chiffrer. Le but d'une telle application est une dispersion maximale.

Précisons également que dans [DR02], une borne inférieure est démontrée en ce qui concerne le nombre de boîtes S actives sur 4 tours d'un algorithme composé à chaque étage d'une série de boîtes S, d'une partie linéaire composée des deux transformations décrites précédemment et d'une addition de clé classique. Cette borne vaut $(Nb)^2$. Rijndael qui satisfait à toutes ces conditions a donc un nombre de boîtes S actives sur 4 étages au moins égal à 25 ce qui rend impossibles les attaques différentielle et linéaire compte tenu des très petits coefficients DP et LP des boîtes S choisies.

La stratégie présentée ici permet une bonne résistance aux cryptanalyses linéaire et différentielle. Cependant, la structure orientée octet de Rijndael liée au choix de cette stratégie de construction rend ce dernier vulnérable à d'autres attaques, notamment les attaques par saturation. Celles-ci seront décrites dans le chapitre 2 de la deuxième partie de cette thèse.

Deuxième partie

Cryptanalyses

Introduction

En février 1999, une attaque menée par les laboratoires RSA contre le DES a permis de retrouver la clé maître utilisée en un peu plus de 22 heures.

Déjà en 1997, sachant le DES vulnérable, le N.I.S.T. (National Institute for Standard and Technology) lançait un appel pour choisir un nouvel algorithme de chiffrement par blocs dont la longueur des clés de 128, 192 et 256 bits devait permettre de l'utiliser au cours du 21ème siècle. Ce nouvel algorithme porterait le nom d'AES (Advanced Encryption Standard). 15 propositions émanant des quatre coins du monde ont été soumises. En août 1999, après une première phase de sélection, 5 finalistes ont été retenus, il s'agissait de Mars [CBD+98], Serpent [ABK98], Twofish [SKW+98], RC6 [RRSY98] et Rijndael [DR98]. C'est finalement Rijndael qui a été choisi comme AES le 2 octobre 2000.

Peu après, en novembre 2000, l'Union Européenne a lancé un projet de choix de primitives cryptographiques touchant tous les domaines de la cryptographie : algorithmes par flot, schémas de signature, algorithmes de chiffrement par blocs,... 39 cryptosystèmes ont été soumis dans toutes ces catégories ; 24 ont été retenus après une première phase de sélection en 2001 [Nes01a].

Le contenu de toute cette thèse, et particulièrement les cryptanalyses présentées ici, sont indissociables de l'effervescence cryptographique née de ces différents challenges.

Nous présentons dans cette partie trois cryptanalyses menées durant cette thèse contre les algorithmes Crypton [Lim98], candidat malheureux de l'AES, Rijndael [DR98], l'AES et SFLASH [PCG00], un schéma de signature à clé publique qui a passé avec succès la première phase de sélection de NESSIE.

Dans un premier chapitre, nous décrirons l'algorithme Crypton ainsi que deux cryptanalyses statistiques menées contre Crypton utilisant respectivement des méthodes différentielles et linéaires. La particularité de ces deux attaques provient dans le cas différentiel d'une partition de l'espace des différences en non pas deux classes comme c'est le cas dans une cryptanalyse différentielle classique (où on considère à chaque étage la valeur de différence de la caractéristique considérée et le reste de l'espace) mais en 16 classes de valeurs possibles et dans le cas linéaire d'une projection non pas sur deux valeurs (0 ou 1) mais sur 16 valeurs. Ces cryptanalyses qui ont un lien de parenté avec la partitioning cryptanalysis de C. Harpes et J. Massey [HM97] sont d'un genre nouveau. Nous leur avons donné le nom de cryptanalyses stochastiques.

La deuxième cryptanalyse présentée ici est également d'un nouveau type et concerne l'algorithme Rijndael, le nouvel AES. Elle utilise une propriété de dépendance particulière entre les entrées et les sorties déterminée par un petit nombre de constantes sur une version réduite à 3 étages de Rijndael.

La troisième cryptanalyse concerne le schéma de signature SFLASH. Il peut paraître étonnant de trouver dans cette thèse dont l'intitulé porte sur la cryptologie à clé secrète un schéma de signature à clé publique. Il est à noter cependant que les structures utilisées dans SFLASH sont très proches de celles développées en clé secrète. En effet, SFLASH se compose d'une première application linéaire suivie d'une fonction quadratique dont le rôle peut s'apparenter à celui d'une série de boîtes S et d'une deuxième application linéaire. On peut donc considérer SFLASH comme une fonction d'étage d'un algorithme de chiffrement à clé secrète.

La cryptanalyse présentée ici s'appuie d'ailleurs sur des principes d'attaques utilisées en clé secrète, notamment, la cryptanalyse différentielle d'ordre supérieur appliquée ici à un petit corps qui a la mauvaise idée de rester invariant après passage dans les fonctions de signature et de vérification de SFLASH. Cette propriété est liée au choix particulier des coefficients des deux applications linéaires qui composent SFLASH.

Chapitre 1

Cryptanalyse de Crypton

Crypton est un algorithme de chiffrement par blocs candidat de l'AES soumis par C. H. Lim en août 1998 [Lim98]. Une version modifiée nommée Crypton v1.0 a été présentée à la conférence FSE'99 [Lim99], le changement principal résidant dans le choix de nouvelles boîtes S.

Dans ce chapitre, nous décrirons deux attaques sur une version de Crypton réduite à 8 étages s'appuyant sur une généralisation des cryptanalyses différentielle et linéaire. Les études précédemment menées en cryptanalyse contre cet algorithme sont dues à l'auteur de Crypton [Lim98] en ce qui concerne les cryptanalyses linéaire et différentielle et à C. D'Halluin [DBRP99] qui attaque une version de Crypton à 6 étages grâce à une transposition de la Square Attack. On doit également à S. Vaudenay la découverte de clés faibles [BGG+99].

Les deux attaques présentées ici, liées à des propriétés itératives de la partie linéaire, utilisent ce que nous appellerons une propriété stochastique, fondée non pas sur une partition de l'espace étudié en deux classes comme c'est le cas pour les cryptanalyses différentielle, linéaire ou différentielle tronquée, mais sur une répartition des valeurs de différences sur des classes contenant chacune 16 éléments et une partition en 16 sous espaces vectoriels de l'espace des entrées.

Ces cryptanalyses ont une certaine parenté avec les travaux de Lai et Massey [LM91] concernant les processus markoviens de la cryptanalyse différentielle, de S. Vaudenay pour l'étude de détermination de biais par le test du χ^2 [Vau95], et avec la "partitioning cryptanalysis" développée par C. Harpes et J. Massey dans [HM97] qui utilise une partition de l'espace des clairs et une partition de l'espace des chiffrés de l'avant dernier tour exploitable si les probabilités de transition des classes de la première partition vers les classes de la seconde partition diffèrent des probabilités qu'induit une permutation aléatoire.
Les deux cryptanalyses présentées ici représentent surtout un des premiers exemples concrets d'application de ces résultats théoriques.

Dans le cas des cryptanalyses dites classiques (différentielle, linéaire, différentielle tronquée,...), l'étude des caractéristiques fait intervenir à chaque étage une matrice de taille 2×2 contenant les probabilités de transition associées aux valeurs d'entrée et de sortie correspondant à la caractéristique étudiée et à l'espace complémentaire associé. Ici, dans les attaques stochastiques qui utilisent une partition des blocs ou des sous-ensembles de valeurs de différences en plus de deux classes, la matrice des probabilités de transition d'un étage est de taille supérieure à 2, chaque élément $p_{i,j}$ de la matrice représentant la probabilité de transition d'une classe d'entrée de la partition vers une classe de sortie de la partition ou d'une valeur particulière de différence appartenant à la partition vers une valeur particulière de différence appartenant également à la partition. Il faut bien sûr que les probabilités détectées soient le plus éloignées possible de la probabilité uniforme et ce pour toutes les valeurs de clé. La première étape de l'attaque consiste à trouver une bonne partition de classes et de bons sous-ensembles de différences. On cherche alors à calculer les probabilités de transition sur k étages.

Nous décrirons dans la première partie le principe général de cryptanalyse utilisée ici, que nous nommerons cryptanalyses stochastiques car il s'agit d'étudier les parcours de certaines probabilités de manière statistique. Dans le cas de Crypton décrit dans la deuxième partie, la propriété itérative particulière d'invariance liée au choix de la partie linéaire qui sera vue à la section 3 ne fait intervenir que 4 boîtes S actives par tour, ce qui permet de limiter les effets d'uniformisation statistique et donc de rendre "praticables" (ou du moins moins complexes qu'une recherche exhaustive) les attaques qui seront décrites dans les parties 4 et 5 de ce chapitre. La section 6 est consacrée à l'étude de la transposition de ces attaques sur Crypton v1.0 et montrent qu'elles sont rendues impossibles par le choix judicieux de nouvelles boîtes S.

1.1 Cryptanalyse Stochastique

Dans cette section, nous allons tenter de formaliser les notions de cryptanalyse que nous appellerons stochastique. Stochastique car l'idée est de modéliser le chiffrement à l'aide de matrices de probabilités de transition entre classes de valeurs intermédiaires.

1.1.1 Cryptanalyse Stochastique Utilisant des Différences

La première cryptanalyse présentée ici utilise les principes de la cryptanalyse différentielle, mais au lieu de partitionner l'espace des valeurs de différences en deux classes, on le partitionne en plusieurs classes à chaque étage.

Soit C un algorithme de chiffrement par blocs sur r tours, faisant intervenir à chaque étage la fonction f. On définit un nombre n de sous-ensembles A_{i_j} où l'indice i représentera l'indice allant de 1 à n et j le numéro du tour. On note m_{i_j} le cardinal de chacun de ces ensembles.

L'idée est alors d'étudier les propagations des différences d'un sous-ensemble à un autre sous-ensemble après passage dans la fonction étage et ce sur plusieurs tours.

Soit δ une différence d'entrée à l'étage j et δ' la différence après passage dans la fonction f, c'est à dire la différence d'entrée de l'étage $j+1$. À toute classe A_{i_j} de différences d'entrée de l'étage j et à toute classe $A_{i'_{j+1}}$ de différences de sortie de l'étage j, on peut associer la probabilité de transition $p_{ii'}$ de A_{i_j} vers $A_{i'_{j+1}}$ définie par :

$$p_{ii'} = \Pr[\delta' = f(x \oplus \delta) \oplus f(x) \in A_{i'_{j+1}} \mid \delta \in A_{i_j}]$$

où x est une valeur d'entrée de l'étage j. Ces probabilités peuvent être exprimées en fonction des $m_{i_j} \cdot m_{i'_{j+1}}$ probabilités différentielles associées à chaque couple (δ, δ') de différences de $A_{i_j} \times A_{i'_{j+1}}$.

Les coefficients $p_{ii'}$ définissent une matrice de transition $M_{j \to j+1}$ des classes de différences d'entrée vers les classes de différences de sortie de f. ($p_{ii'}$ représente le coefficient associé à la colonne i et à la ligne i').

Pour obtenir des probabilités de transition sur deux étages, sous l'hypothèse d'indépendance des étages et des clés de chaque étage et en supposant que les suites de différences représentent une chaîne de Markov homogène (voir première partie section 2.2.2, théorème 1), il suffit de multiplier entre elles les matrices des probabilités de transition :

$$M_{j \to j+2} = M_{j+1 \to j+2} \times M_{j \to j+1}.$$

On obtient ainsi sur deux étages les probabilités de transition d'une classe A_{i_j} de différences vers une autre classe $A_{i'_{j+2}}$ de différences.

On généralise ce résultat à un nombre supérieur d'étages en appliquant plusieurs fois le processus précédent. Ainsi pour $(r-1)$ étages, on multiplie entre elles les $(r-1)$ matrices de probabilités de transitions des r espaces traversés. On s'intéresse alors uniquement à l'appartenance de la différence de sortie au sous-ensemble d'arrivée.

En pratique, il est plus intéressant de prendre à l'entrée des $(r-1)$ étages une différence particulière d'un sous-ensemble particulier qui maximise la probabilité de transition vers un sous-ensemble particulier en sortie du premier étage.

Pour créer un distingueur sur r étages à partir d'une relation sur $(r-1)$ étages, on utilise le même principe que celui de la cryptanalyse différentielle classique en testant toutes les valeurs de la sous-clé du dernier étage. La bonne sous-clé est celle qui apparaît le plus souvent. Les fausses alarmes sont cependant plus nombreuses et le rapport signal sur bruit moins bon, il est donc nécessaire de prendre un nombre de clairs légèrement supérieur à celui d'une cryptanalyse différentielle classique.

Cette cryptanalyse est très proche de la cryptanalyse utilisant des différentielles tronquées mais nous pensons que l'utilisation des matrices de probabilités de transition donne des résultats plus précis.

1.1.2 Cryptanalyse Stochastique Utilisant des Relations Linéaires

Dans une cryptanalyse linéaire classique, il s'agit en général de projeter des applications affines de $GF(2)^n$ appliquées au chiffrement sur l'ensemble $GF(2)$. L'idée de la cryptanalyse stochastique utilisant des relations linéaires consiste à projeter des applications affines non sur $GF(2)$ mais sur un espace plus grand que l'on peut apparenter à $GF(2^k)$ où k est un entier naturel. C'est à dire qu'on partitionne l'espace sur un certain nombre de classes supérieur à 2. Dans la cryptanalyse de Crypton, il s'agit de $m = 16$ classes.

Soit C un chiffrement sur r tours faisant appel à chaque étage à une fonction f qui prend en entrée/sortie des éléments de $GF(2)^n$. On note X_i le chiffré de l'étage i. Soit A un ensembleb de classes de cardinal m. On cherche alors des applications linéaires Φ de $GF(2)^n$ dans A qui pour toutes valeurs l et l' de A maximisent ou minimisent les probabilités de transition définies par :

$$p_{ll'} = \frac{\{X_i \in GF(2)^n, \Phi(X_i) = l \text{ et } \Phi(X_{i+1}) = l'\}}{\{X_i \in GF(2)^n, \Phi(X_i) = l\}}$$

Cela est équivalent à la recherche d'une relation linéaire sur un étage. Une fois que l'on a trouvé une bonne relation linéaire pour un étage, on calcule le biais par rapport à la moyenne, ici égale à $\frac{1}{m^2}$, pour chaque l et chaque l'. Il ne reste plus qu'à placer ces résultats dans une matrice M de taille $m \times m$ selon les valeurs de l et l'.

Pour obtenir les biais de transition impliqués par la relation Φ sur deux étages, on multiplie la matrice M par elle-même comme dans le cas d'une cryptanalyse stochastique différentielle. Cependant, dans ce cas, les résultats ne sont pas indépendants des sous-clés de chaque étage qui interviennent dans la relation Φ. Il est alors nécessaire de multiplier la matrice M^2 par une matrice de taille elle aussi $m \times m$ qui représente une permutation de A dans A afin de prendre en compte l'influence des bits de clés mis en jeu dans Φ.

Pour déterminer les biais de transition sur $(r-1)$ étages, il suffit de multiplier $(r-2)$ fois la matrice M avec elle-même. On multiplie ici encore la matrice M par toutes les matrices de permutations de taille $m \times m$ pour tenir compte des déviations dues aux bits de clé de chaque étage. On calcule ensuite la moyenne de tous les résultats ainsi obtenus.

Pour construire un distingueur sur r étages, on utilise la relation précédente sur $(r-1)$ étages et on teste à l'aide d'un opérateur statistique (nous emploierons ici le test du χ^2) la clé du dernier étage en effectuant un déchiffrement partiel. La bonne clé est celle pour laquelle l'indicateur du χ^2 est le plus haut.

Le nombre de clairs N nécessaire à cette attaque est donné par les équations suivantes décrites dans [HG97].

Soit $(X_j)_{j=1\cdots N}$ un ensemble de N variables aléatoires supposées indépendantes. On suppose également que les X_j prennent leurs valeurs dans l'intervalle $[0,\cdots,m-1]$. On définit alors les quantités $n_i = \#\{j|X_j = i\}$ pour i allant de 0 à $m-1$ et l'indicateur du χ^2 par

$$S_{\chi^2} = \frac{m}{N}\sum_{i=0}^{m-1}\left(n_i - \frac{N}{m}\right)^2.$$

Le test du χ^2 compare alors la valeur de cet indicateur avec celle que fournirait une distribution multinômiale non biaisée. Si les n_i sont distribués selon une distribution multinômiale de paramètres $\left(\frac{1}{m},\cdots,\frac{1}{m}\right)$, l'espérance et l'écart-type de l'indicateur S_{χ^2} sont donnés par les relations suivantes :

$$E\left(S_{\chi^2}\right) \to \mu = m-1 \text{ et } \sigma\left(S_{\chi^2}\right) \to \sqrt{2\cdot(m-1)}$$

En revanche, si les X_j sont distribués selon une loi multinômiale biaisée de paramétres $(p_0,...,p_{m-1})$, l'espérance de l'indicateur S_{χ^2} est alors :

$$\mu' = \mu + m(N-1)\sum_{i=0}^{m-1}\left(p_i - \frac{1}{m}\right)^2.$$

L'ordre de grandeur du nombre N d'échantillons nécessaires pour que le test du χ^2 détecte un biais est donné par : $\mu' - \mu >> \sigma$.

Ce qui nous donne l'ordre de grandeur de N :

$$N >> \frac{\sqrt{2 \cdot (m-1)}}{m \sum_{i=0}^{m-1} \left(p_i - \frac{1}{m}\right)^2}.$$

Dans le cas qui nous intéresse, l'expression $\sum_{i=0}^{m-1} \left(p_i - \frac{1}{m}\right)^2$ correspond à la somme des éléments d'une colonne de la matrice $m \times m$ utilisée. Dans le cas de Crypton, on a $m = 16$.

Nous venons de voir les définitions des attaques stochastiques que nous allons utiliser pour monter des cryptanalyses contre Crypton. Nous allons donc dans un premier temps décrire l'algorithme.

1.2 Description de l'Algorithme

Crypton [Lim98] est un algorithme coréen présenté pour l'AES par C. H. Lim pour Futur Systems Inc.. C'est un algorithme itératif qui chiffre des blocs de 128 bits sous des clés de 128, 192 ou 256 bits. Il se compose d'une addition de clé initiale, suivie de $r = 12$ étages identiques puis d'une transformation finale ϕ. La fonction de déchiffrement est identique à la fonction de chiffrement. On représente le bloc A de 128 bits à chiffrer par une matrice 4×4 d'octets :

$$A = \begin{pmatrix} a_{0,3} & a_{0,2} & a_{0,1} & a_{0,0} \\ a_{1,3} & a_{1,2} & a_{1,1} & a_{1,0} \\ a_{2,3} & a_{2,2} & a_{2,1} & a_{2,0} \\ a_{3,3} & a_{3,2} & a_{3,1} & a_{3,0} \end{pmatrix} \begin{array}{c} A[0] \\ A[1] \\ A[2] \\ A[3] \end{array}$$

où chaque $a_{i,j}$ est un octet.

Chaque étage est composé d'une substitution d'octets γ, d'une permutation de bits noté π, d'une transposition d'octets τ et d'une addition de clé σ_k, faisant intervenir à chaque étage une sous-clé produite par l'algorithme de génération de clés.

1.2.1 La Substitution d'Octets γ

La substitution d'octets γ fait appel à deux boîtes S S_0 et S_1 réciproques l'une de l'autre qui prennent en entrée/sortie des mots de 8 bits :
Dans le cas d'un tour impair, on a $B = \gamma_0(A)$ où γ_0 est défini par :

$$\begin{pmatrix} b_{0,3} & b_{0,2} & b_{0,1} & b_{0,0} \\ b_{1,3} & b_{1,2} & b_{1,1} & b_{1,0} \\ b_{2,3} & b_{2,2} & b_{2,1} & b_{2,0} \\ b_{3,3} & b_{3,2} & b_{3,1} & b_{3,0} \end{pmatrix} \xleftarrow{\gamma_0} \begin{pmatrix} S_1(a_{0,3}) & S_0(a_{0,2}) & S_1(a_{0,1}) & S_0(a_{0,0}) \\ S_0(a_{1,3}) & S_1(a_{1,2}) & S_0(a_{1,1}) & S_1(a_{1,0}) \\ S_1(a_{2,3}) & S_0(a_{2,2}) & S_1(a_{2,1}) & S_0(a_{2,0}) \\ S_0(a_{3,3}) & S_1(a_{3,2}) & S_0(a_{3,1}) & S_1(a_{3,0}) \end{pmatrix}$$

Dans le cas d'un tour pair, on a $B = \gamma_1(A)$, égal à γ_0 à la position des boîtes S près :

$$\begin{pmatrix} b_{0,3} & b_{0,2} & b_{0,1} & b_{0,0} \\ b_{1,3} & b_{1,2} & b_{1,1} & b_{1,0} \\ b_{2,3} & b_{2,2} & b_{2,1} & b_{2,0} \\ b_{3,3} & b_{3,2} & b_{3,1} & b_{3,0} \end{pmatrix} \xleftarrow{\gamma_1} \begin{pmatrix} S_0(a_{0,3}) & S_1(a_{0,2}) & S_0(a_{0,1}) & S_1(a_{0,0}) \\ S_1(a_{1,3}) & S_0(a_{1,2}) & S_1(a_{1,1}) & S_0(a_{1,0}) \\ S_0(a_{2,3}) & S_1(a_{2,2}) & S_0(a_{2,1}) & S_1(a_{2,0}) \\ S_1(a_{3,3}) & S_0(a_{3,2}) & S_1(a_{3,1}) & S_0(a_{3,0}) \end{pmatrix}$$

La relation $\gamma_1(\gamma_0(A)) = \gamma_0(\gamma_1(A)) = A$ permet une inversion simple de γ dans le cas du déchiffrement.
Signalons également les meilleurs biais observés sur les deux boîtes S S_0 et S_1 pour la cryptanalyse différentielle et la cryptanalyse linéaire :

$$DP_{S_0} = DP_{S_1} = 2^{-5} \text{ et } LP_{S_0} = LP_{S_1} = 2^{-4}.$$

Ces biais ne sont pas optimaux car sinon, on devrait avoir $LP = DP = 2^{-6}$.

1.2.2 La Permutation π et la Transposition d'Octets τ

Comme pour la transformation γ, la permutation π est légèrement modifiée selon la parité du tour. Dans le cas d'un tour impair, on a $B = \pi_0(A)$ où π_0 est défini par :

$$\begin{array}{rcl} T & = & A[0] \oplus A[1] \oplus A[2] \oplus A[3], \\ B[0] & \leftarrow & (A[0] \wedge MI_0) \oplus (A[1] \wedge MI_1) \oplus (A[2] \wedge MI_2) \oplus (A[3] \wedge MI_3) \oplus T, \\ B[1] & \leftarrow & (A[0] \wedge MI_1) \oplus (A[1] \wedge MI_2) \oplus (A[2] \wedge MI_3) \oplus (A[3] \wedge MI_0) \oplus T, \\ B[2] & \leftarrow & (A[0] \wedge MI_2) \oplus (A[1] \wedge MI_3) \oplus (A[2] \wedge MI_0) \oplus (A[3] \wedge MI_1) \oplus T, \\ B[3] & \leftarrow & (A[0] \wedge MI_3) \oplus (A[1] \wedge MI_0) \oplus (A[2] \wedge MI_1) \oplus (A[3] \wedge MI_2) \oplus T, \end{array}$$

avec $MI_0 = c0300c03, MI_1 = 03c0300c, MI_2 = 0c03c030, MI_3 = 300c03c0$ en notation hexadécimal. Les effets de la transformation π_1 dans le cas d'un tour pair sont similaires et on a $\pi_0^{-1} = \pi_0$ et $\pi_1^{-1} = \pi_1$. Notons également que si on omet l'addition de T, π est une symétrie suivie d'une rotation opérant sur les 16 colonnes de mots de 2 bits constituant un bloc (voir figure 1.1).
La transposition d'octets τ est une simple transposition matricielle qui échange les $a_{i,j}$ et les $a_{j,i}$.

1.2.3 L'Addition de Clé σ_k

L'addition de clé σ_k est un simple x-or bit à bit sur les lignes du bloc à chiffrer : $B[i] = A[i] \oplus k[i]$ pour i variant de 0 à 3.

On note alors $\rho_{1,k}(A)$ la transformation d'étage dans le cas d'un tour pair, on a : $\rho_{1,k}(A) = (\sigma_k \circ \tau \circ \pi_1 \circ \gamma_1)(A)$. Dans le cas d'un tour impair, on définit de même la fonction d 'étage $\rho_{0,k}(A)$ par $\rho_{0,k}(A) = (\sigma_k \circ \tau \circ \pi_0 \circ \gamma_0)(A)$.

Comme toutes les transformations d'un étage sont des involutions sauf les transformations γ_i qui sont inverses l'une de l'autre, on a :

$$\begin{array}{rcl} \rho_{1,k}^{-1}(A) & = & (\gamma_0 \circ \pi_1 \circ \tau \circ \sigma_k)(A) \\ \rho_{0,k}^{-1}(A) & = & (\gamma_1 \circ \pi_0 \circ \tau \circ \sigma_k)(A) \end{array}$$

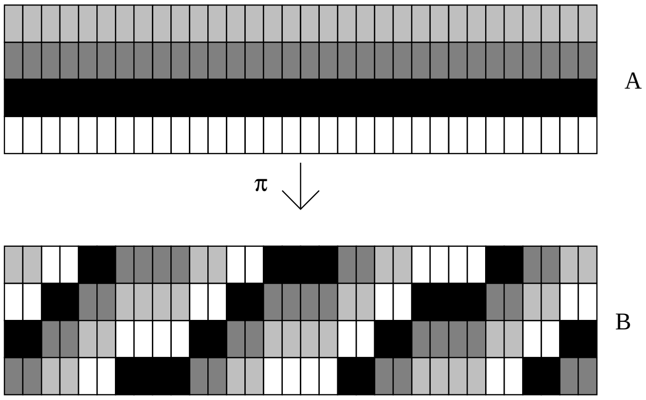


FIG. 1.1 – Transformation π_0 sans le X-or avec T

1.2.4 La Transformation Finale ϕ

Cette transformation a été créée pour rendre le chiffrement et le déchiffrement identiques exception faite des clés. Si on suppose que le nombre d'étages r est pair, on a : $\phi_1 = \tau \circ \pi_1 \circ \tau$ (dans le cas d'un nombre de tours impairs, on pose $\phi_0 = \tau \circ \pi_0 \circ \tau$). La procédure de chiffrement s'écrit alors :

$$E_k = \phi_1 \circ \rho_{1,k_e^r} \circ \rho_{0,k_e^{r-1}} \circ \cdots \circ \rho_{1,k_e^2} \circ \rho_{0,k_e^1} \circ \sigma_{k_e^0}$$

où k_e^i désigne la sous-clé de l'étage i.

On pose alors :

$$k_d^{r-i} = \begin{cases} \phi_1(k_e^i) & \text{si } i \text{ est pair} \\ \phi_0(k_e^i) & \text{si } i \text{ est impair} \end{cases}$$

en remarquant que $\phi_1 \circ \sigma_{k_e^i} = \sigma_{k_d^{r-i}} \circ \phi_1$, ce qui entraîne la propriété suivante :

$$\phi_1 \circ \rho_{1,k_e^r} = \sigma_{k_d^0} \circ \tau \circ \gamma_1$$

. On cherche alors à calculer la fonction de déchiffrement $D_k = E_k^{-1}$:

$$\begin{aligned} D_k &= \left(\phi_1 \circ \rho_{1,k_e^r} \circ \rho_{0,k_e^{r-1}} \circ \cdots \circ \rho_{1,k_e^2} \circ \rho_{0,k_e^1} \circ \sigma_{k_e^0}\right)^{-1} \\ D_k &= \sigma_{k_e^0} \circ \rho_{0,k_e^1}^{-1} \circ \rho_{1,k_e^2}^{-1} \circ \cdots \circ \rho_{0,k_e^{r-1}}^{-1} \circ \gamma_1 \circ \tau \circ \sigma_{k_d^0} \end{aligned}$$

car $\phi_1 \circ \rho_{1,k_e^r} = \sigma_{k_d^0} \circ \tau \circ \gamma_1$, on en déduit que :

$$\begin{aligned} D_k &= \sigma_{k_e^0} \circ \left(\gamma_1 \circ \pi_0 \circ \tau \circ \sigma_{k_e^1}\right) \circ \left(\gamma_0 \circ \pi_1 \circ \tau \circ \sigma_{k_e^2}\right) \\ &\quad \circ \cdots \circ \left(\gamma_1 \circ \pi_0 \circ \tau \circ \sigma_{k_e^{r-1}}\right) \circ \gamma_1 \circ \tau \circ \sigma_{k_d^0} \end{aligned}$$

Étudions à présent la transformation particulière $\sigma_{k_e^0} \circ \left(\gamma_1 \circ \pi_0 \circ \tau \circ \sigma_{k_e^1}\right)$. On a :

$$\begin{aligned} \sigma_{k_e^0} \circ \left(\gamma_1 \circ \pi_0 \circ \tau \circ \sigma_{k_e^1}\right) &= \phi_1 \circ \phi_1 \circ \sigma_{k_e^0} \circ \left(\gamma_1 \circ \pi_0 \circ \tau \circ \phi_0 \circ \phi_0 \sigma_{k_e^1}\right) \\ &= \phi_1 \circ \sigma_{k_d^r} \circ \tau \circ \pi_1 \circ \tau \circ \gamma_1 \circ \tau \circ \tau \circ \pi_0 \circ \tau \circ \sigma_{k_e^1} \\ &= \phi_1 \circ \sigma_{k_d^r} \circ \tau \circ \pi_1 \circ \gamma_1 \circ \phi_0 \circ \sigma_{k_e^1} \\ &= \phi_1 \circ \rho_{1,k_d^r} \circ \phi_0 \circ \sigma_{k_e^1} \end{aligned}$$

car $\tau \circ \gamma_1 \circ \tau = \gamma_1$. Il ne reste alors plus qu'à "remonter" le reste de la fonction de déchiffrement en ajoutant des transformations $\tau \circ \tau$ après chaque γ_i sauf pour celui du premier tour. On obtient :

$$D_k = \phi_1 \circ \rho_{1,k_d^r} \circ \rho_{0,k_d^{r-1}} \circ \cdots \circ \rho_{1,k_d^2} \circ \rho_{0,k_d^1} \circ \sigma_{k_d^0}$$

Le chiffrement et le déchiffrement sont donc bien identiques moyennant l'utilisation de clés différentes.

1.2.5 Génération des Sous-Clés

Je ne rentrerai pas ici dans le détail de la génération des sous-clés. Précisons simplement que celle-ci comporte deux phases : l'expansion à 256 bits de la clé initiale par rajout de zéros et la génération des sous-clés à partir des transformations $\sigma_{P\ ou\ Q}$, τ, π_i et γ_i où P et Q sont deux constantes.

Signalons surtout que J. Borst et S. Vaudenay [BGG$^+$99] ont découvert indépendamment des clés faibles. Une clé de 256 bits de la forme xxxxyyyyxxxxyyyyzzzzttttzzzztttt où x, y, z et t sont des octets laisse invariants les textes de la forme ababcdcdababcdcd où a, b, c et d sont également des octets. Même si ces clés particulières ne représentent pas un sérieux danger pour l'algorithme, il faut en éviter 2^{32}.

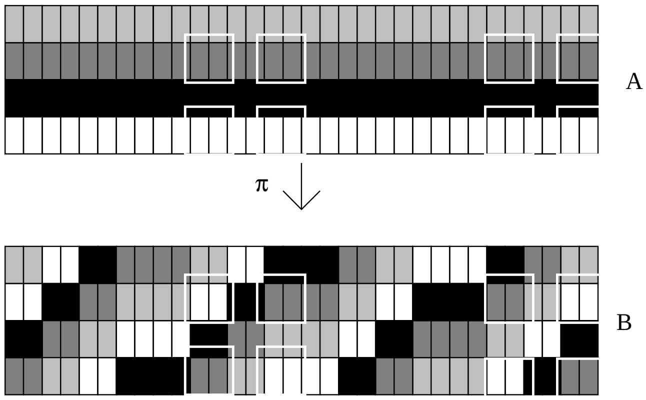
1.3 Propriétés Utilisées dans les Attaques

Nous allons présenter dans ce paragraphe deux propriétés (l'une impliquant des différences, l'autre des relations linéaires) de la partie linéaire de Crypton pouvant être exploitées pour monter des attaques contre cet algorithme. Ces deux propriétés décrites dans [BGG$^+$99] découlent de la mauvaise diffusion de la transformation $\tau \circ \pi$, ou plus précisément, de l'invariance de certains groupes de deux carrés de mots de deux bits.

Soit $a_{i,j}$ un octet du bloc A. On découpe alors $a_{i,j}$ en quatre mots de deux bits et on note $a_{i,j} = (a_{3,i,j}, a_{2,i,j}, a_{1,i,j}, a_{0,i,j})$. $a_{k,i,j}$ (avec $i \in [0 \cdots 3]$, $j \in [0 \cdots 3]$ et $k \in [0 \cdots 3]$) désignera donc le mot de deux bits de la ligne i, de la colonne j et de position k dans l'octet. On définit alors le "carré" de mots de deux bits associé au triplet (k, i, j) par $(a_{k,i,j}, a_{k,i+2,j}, a_{k,i,j+2}, a_{k,i+2,j+2})$, tous les indices étant pris modulo 4. Sous certaines conditions, ce carré reste invariant après passage dans γ_0, γ_1, π_0, π_1 et τ, moyennant un changement des indices (k, i, j).

Comme décrit dans [BGG$^+$99], nous pouvons voir que les transformations π_0 et π_1 si on omet T ne font que permuter les valeurs d'indice i du carré $a_{k,i,j}$. Ainsi les carrés associés au triplet (k, i, j) et $(k+2, i, j)$ sont transformés en les mêmes carrés associés à des triplets (k, i', j) et $(k+2, i', j)$:

Cette propriété d'invariance reste vraie dans le cas des fonctions π_0 et π_1 complètes à condition de prendre les valeurs sur une colonne égales, c'est à dire, plus formellement, dans le carré associé au triplet (k, i, j), de prendre $a_{k,i,j} = a_{k,i+2,j}$ et $a_{k,i,j+2} = a_{k,i+2,j+2}$. Si on se place dans ce cas de figure, les deux carrés associés aux triplets (k, i, j) et $(k+2, i, j)$ sont transformés après passage dans π_i en deux autres carrés d'indices (k, i', j') et



$(k+2, i', j')$ dont les valeurs en chaque position sont les mêmes que les précédentes à une constante près (donnée par T). Dans la suite, on nommera les deux carrés d'indice (k, i, j) et $(k+2, i, j)$ des carrés jumeaux.

1.3.1 Propriété Utilisant des Différences

Afin d'exploiter cette propriété dans une attaque différentielle, on considère la différence de deux blocs à chiffrer égale à zéro partout sauf en deux carrés jumeaux d'indices (k, i, j) et $(k+2, i, j)$ vérifiant $a_{k,i,j} = a_{k,i+2,j}$ et $a_{k,i,j+2} = a_{k,i+2,j+2}$. On peut par exemple dans le cas particulier où $i = 1$, $j = 1$ et $k = 1$ écrire cette différence sous la forme :

$$\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 00x_1x_200y_1y_2 & 0 & 00x'_1x'_200y'_1y'_2 \\ 0 & 0 & 0 & 0 \\ 0 & 00x_1x_200y_1y_2 & 0 & 00x'_1x'_200y'_1y'_2 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & a & 0 & b \\ 0 & 0 & 0 & 0 \\ 0 & a & 0 & b \end{pmatrix}$$

où x_1, x_2, x'_1, x'_2, y_1, y_2, y'_1 et y'_2 représentent des bits et a et b des octets.

Le passage d'une différence de cette forme dans la fonction π_i ne fait que changer la position du carré de différence. Le couple (i, j) se transforme donc en un couple (i', j').

Notons également que le passage dans la fonction τ ne fait que transposer les coefficients. Les carrés d'indices (k, i', j') et $(k+2, i', j')$ sont transformés en les carrés d'indices (k, j', i') et $(k+2, j', i')$. Ce que l'on représente de façon matricielle dans l'exemple précédent par

$$\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & a & 0 & b \\ 0 & 0 & 0 & 0 \\ 0 & a & 0 & b \end{pmatrix} \stackrel{\tau}{\Rightarrow} \begin{pmatrix} b & 0 & b & 0 \\ 0 & 0 & 0 & 0 \\ a & 0 & a & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Précisons également que le passage d'une telle différence dans la partie non linéaire γ de Crypton ne fait intervenir que 4 boîtes S.

1.3.2 Propriété de Type Linéaire

Elle est fondée sur les mêmes propriétés d'invariance que celles décrites précédemment. Elle fait également intervenir deux "carrés jumeaux" définis par la donnée des deux triplets d'indices (k, i, j) et $(k+2, i, j)$. De la même manière, si on omet l'addition du terme T, ces deux triplets d'indices sont transformés en deux autres triplets d'indices (k, i', j') et $(k+2, i', j')$. Dans le cas d'une étude linéaire, on définit la fonction $\Phi_{k,i,j}$ comme étant le x-or des mots de 2 bits d'un carré d'indices (k, i, j) : $\phi_{k,i,j} = a_{k,i,j} \oplus a_{k,i+2,j} \oplus a_{k,i,j+2} \oplus a_{k,i+2,j+2}$. On a alors, si (k, i', j') est l'image après passage dans π_i de (k, i, j), $\phi_{k,i,j} = \phi_{k,i',j'}$, même en tenant compte du terme T dont la valeur s'auto-compense par définition de ϕ. De même, on a : $\phi_{k+2,i,j} = \phi_{k+2,i',j'}$. On peut donc en déduire que la valeur de la fonction $\Phi_{k,i,j} = 4 \cdot \phi_{k+2,i,j} \oplus \phi_{k,i,j}$ est égale à $\Phi_{k,i',j'} = 4 \cdot \phi_{k+2,i',j'} \oplus \phi_{k,i',j'}$ toujours en tenant compte de T.

De même, le passage dans τ laisse cette valeur inchangée.

Il existe donc une combinaison linéaire portant sur le x-or de valeurs de 2 bits de deux carrés jumeaux, impliquant seulement 4 octets du bloc à chiffrer, dont la valeur reste inchangée après passage dans la partie linéaire de Crypton. Cette propriété qui corrèle les entrées et sorties de la partie linéaire peut être exploitée pour monter une attaque de type linéaire ou plus précisément une cryptanalyse utilisant une partition des valeurs en 16 classes.

Précisons également que dans les deux propriétés présentées précédemment le passage dans γ_i des deux "carrés jumeaux" ne fait intervenir que 4 boîtes S ce qui permet de limiter les effets d'uniformisation des biais statistiques, différentielles et linéaires produits par la partie non linéaire.

1.4 Attaque Stochastique Utilisant les Propriétés Différentielles

1.4.1 Calcul des Probabilités de Transition

Cette première cryptanalyse est fondée sur la propriété décrite dans la section 1.2.1 et utilise le même type de différences, c'est à dire des différences sur un octet de la forme $00x_1x_200y_1y_2$ ou $x_1x_200y_1y_200$ incluses dans un carré de différences toutes de la même forme. On introduit donc deux ensembles de différences possibles entre deux octets :

$$D_1 = \{0,1,2,3,16,17,18,19,32,33,34,35,48,49,50,51\}$$
$$D_2 = \{0,4,8,12,64,68,72,76,128,132,136,140,192,196,200,204\}$$

D_1 correspondant aux différences de la forme $00x_1x_200y_1y_2$ et D_2 à celles de la forme $x_1x_200y_1y_200$.

La valeur 0 est contenue dans les deux ensembles car elle vérifie bien les propriétés des deux ensembles mais ne peut apparaître en pratique car cela signifierait que la différence d'entrée ou de sortie est nulle, ce qui ne peut pas se produire. Ce choix particulier de D_1 et D_2 nous permet d'utiliser les propriétés d'invariance de π_i sur des carrés jumeaux de différence de la forme

$$\Delta_1 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & \delta_1 & 0 & \delta_2 \\ 0 & 0 & 0 & 0 \\ 0 & \delta_1 & 0 & \delta_2 \end{pmatrix}$$

où Δ_1 est représenté par une matrice 4×4 d'octets avec $\delta_1 \in D_1$ et $\delta_2 \in D_1$
ou de la forme

$$\Delta_2 = \begin{pmatrix} 0 & \delta_1 & 0 & \delta_2 \\ 0 & 0 & 0 & 0 \\ 0 & \delta_1 & 0 & \delta_2 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

avec cette fois, $\delta_1 \in D_2$ et $\delta_2 \in D_2$. La position des carrés de différence n'est pas la même pour les deux ensembles afin de garantir que la propriété d'invariance soit bien vérifiée après passage dans π_i.

Par abus de langage, on identifiera le couple de différences (δ_1, δ_2) et le carré de différence correspondant Δ_1 ou Δ_2. Ces deux matrices ne sont que des exemples des carrés de différences que l'on peut utiliser, toutes les positions dans la matrice pouvant être exploitées dans cette attaque.

D'après le paragraphe précédent, il existe δ'_1 et δ'_2, valeurs de D_1 telles que :

$$\tau(\pi_i(\Delta_1)) = \begin{pmatrix} \delta'_2 & 0 & \delta'_2 & 0 \\ 0 & 0 & 0 & 0 \\ \delta'_1 & 0 & \delta'_1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

On peut alors déduire qu'avec une certaine probabilité les carrés jumeaux représentés par le couple (δ'_1, δ'_2) se transforme après passage dans les boîtes S en un couple (δ_3, δ_4) tel que

$$\gamma_i(\tau(\pi_i(\Delta_1))) = \begin{pmatrix} \delta_3 & 0 & \delta_4 & 0 \\ 0 & 0 & 0 & 0 \\ \delta_3 & 0 & \delta_4 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

On peut alors utiliser de nouveau la propriété d'invariance de $\gamma_i(\tau(\pi_i(\Delta_1)))$ à travers π_i et τ pour un autre carré jumeau. On obtient donc avec une certaine probabilité p' l'invariance de Δ_1 sur un tour complet. En utilisant la même construction, on obtient également l'invariance de Δ_2 sur un tour avec une certaine probabilité p''.

Pour tout couple (δ_1, δ_2) de D_1 et tout couple (δ_3, δ_4) de D_1, on calcule alors la probabilité de passage sur un tour d'obtenir un couple de sortie (δ_3, δ_4) pour un couple d'entrée (δ_1, δ_2). Toutes ces probabilités sont indépendantes de la clé et dépendent uniquement de la distribution des différences des boîtes S.

$$\begin{aligned} Pr[\Delta' = (\delta_3, \delta_4) | \Delta = (\delta_1, \delta_2)] &= Pr[\delta_1 \to \delta_3] \cdot Pr[\delta_2 \to \delta_3] \\ &\quad \cdot Pr[\delta_1 \to \delta_4] \cdot Pr[\delta_2 \to \delta_4] \\ &= \frac{d_{\delta_1\delta_3}}{256} \cdot \frac{d_{\delta_2\delta_3}}{256} \cdot \frac{d_{\delta_1\delta_4}}{256} \cdot \frac{d_{\delta_2\delta_4}}{256} \end{aligned}$$

où $d_{\delta_i \delta_j}$ désigne le nombre d'octets tel que : $\delta_j = S(x) \oplus S(x \oplus \delta_i)$, S représentant la boîte S de Crypton appropriée.

On obtient une matrice M 256×256 de probabilités de transition sur un tour en calculant $Pr[\Delta' = (\delta_3, \delta_4)|\Delta = (\delta_1, \delta_2)]$ pour les 256 valeurs possibles du couple d'entrée (δ_1, δ_2) et pour les 256 valeurs possibles du couple de sortie (δ_3, δ_4) avec $\delta_1, \delta_2, \delta_3, \delta_4 \in D_1$. Les colonnes de la matrice M représentent alors toutes les probabilités de transition du couple (δ_1, δ_2) vers toutes les sorties possibles, c'est à dire tous les couples (δ_3, δ_4). On utilise le même raisonnement pour les probabilités de transition liées à l'ensemble de différence D_2.

On considère à présent des différentielles sur deux tours de la fonction d'étage de Crypton, c'est à dire des transitions entre une valeur d'entrée (δ_1, δ_2) et une valeur de sortie au bout de deux tours (δ_3, δ_4) en autorisant toutes les valeurs possibles pour la valeur intermédiaire de la fin du premier tour. Une borne inférieure sur les probabilités de telles différentielles est donnée en sommant toutes les valeurs intermédiaires possibles qui appartiennent à D_1 :

$$p_{i,j} = \sum_{k=0}^{255} p_{i,k} p_{k,j}$$

où les $p_{i,j}$ représentent les coefficients de la matrice M de taille 256×256.

On a ainsi tenu compte de toutes les possibilités de transition intermédiaires comme vu dans la section 1.1.1. Il ne reste plus qu'à appliquer les résultats de la partie 1.1.1, ce qui est possible car dans le cas étudié ici, les hypothèses d'indépendance des étages et des clés restent vraies et on peut dire que les suites de différences sur plusieurs étages représentent bien une chaîne de Markov homogène. On peut donc calculer M^n qui représente les probabilités de transition de différences entre l'entrée et la sortie de n étages de manière complètement indépendante de la clé. Par exemple, pour des probabilités de transition sur 6 étages, on calcule M^6. De plus, d'un point de vue cryptanalytique, seule l'appartenance du couple de différence de sortie à l'ensemble $D_1 \times D_1$ nous intéresse. On peut donc sommer les valeurs de la "meilleure" colonne de la matrice ainsi obtenue afin de tenir compte de toutes les sorties envisageables.

1.4.2 Procédure d'Attaque et Résultats Pratiques

Nous présentons ici une attaque sur une version de Crypton à huit étages composée de l'addition de clé initiale, 8 étages internes et de la transformation finale. Sous les hypothèses heuristiques vues à la section 1.1.1 (indépendance des étages et indépendance des sous-clés mises en jeu à chaque étage), on peut calculer les probabilités des meilleurs couples de différences sur plusieurs étages. Dans le cas d'un schéma composé de six étages internes de Crypton, si la différence d'entrée est le couple (128,128) appartenant à l'ensemble $D_2 \times D_2$ et formant une matrice de différence de la forme Δ_2, la probabilité que la sortie soit une différence (δ_1, δ_2) appartenant également à $D_2 \times D_2$ est de $2^{-112.62}$, meilleure probabilité observée.

L'attaque présentée ici utilise cette probabilité sur 6 étages internes afin de construire un distingueur sur 6 tours internes utilisant la relation précédente et ayant une probabilité de distinguer le chiffrement d'une permutation aléatoire égale à $2^{-112.62}$. On ajoute au-dessus de ce distingueur l'addition de clé initiale σ_0 et un étage interne ainsi qu'un huitième étage interne et la transformation finale Φ_e après les 6 étages du distingueur. Cette attaque permet d'obtenir de l'information sur 4 octets de la clé K_0 utilisés lors de l'addition de clé σ_0 et sur 4 octets de la clé K_8, clé du huitième étage, à l'aide d'une recherche exhaustive sur des textes clairs munis d'une structure particulière.

Le premier tour interne peut être ici considéré comme une permutation ne faisant pas intervenir les clés en raison de l'occurrence tardive de l'addition de clé du premier tour. On peut donc contrôler les différences jusqu'à cette addition de clé. On tient ensuite de la relation sur six étages et on ajoute un huitième étage en testant les hypothèses sur la sous-clé du huitième étage à l'aide d'un distingueur.

Afin d'améliorer la recherche exhaustive des 32 bits de la clé K_0, on regroupe les textes clairs choisis en structure de 2^{16} éléments, il est alors nécessaire de connaître la valeur des 4 octets de la clé K_0 qui seront déterminés par la recherche exhaustive. Deux clairs appartiennent à la même structure si leurs images après le premier γ, c'est à dire leurs images à la moitié du premier tour sont égales partout sauf sur les 16 bits de $\Delta_2 = (128, 128)$. Si X et X' sont des éléments qui appartiennent à la même structure alors, avec une probabilité égale à $2^{-112.62}$, leurs images à l'entrée du huitième tour sont de la forme Y et $Y \oplus \Delta$ où Δ est une différence de l'ensemble D_2. Il ne reste alors plus qu'à filtrer les paires (X, X') de clairs dont les chiffrés C et C' sont tels que $C \oplus C'$ est nul partout excepté dans le carré d'éléments formant le Δ_2 attendu.

On chiffre N paires de textes choisis réparties en structures contenant chacune 2^{15} paires et on ne garde que les chiffrés vérifiant la condition précédente. Il ne reste alors plus qu'à tester les 4 octets de la clé K_0 et les 4 octets de la clé K_8 présents dans la différence Δ_2 en faisant un déchiffrement partiel du dernier tour et de la transformation finale. La valeur qui apparaît le plus souvent est la bonne. Il faut cependant prendre en compte un certain nombre de "fausses alarmes", c'est à dire de chiffrés qui ont la bonne forme sans pour autant être de vrais candidats. La probabilité d'apparition de ces fausses alarmes est de 2^{-96}. Il est alors nécessaire pour s'en préserver d'augmenter le nombre de clairs. De plus, le rapport signal/bruit étant faible, il est nécessaire de chiffrer $N = 8 \times 2^{112{,}62} = 2^{115{,}62}$ clairs choisis.

Dans cette attaque, on obtient donc 32 bits d'information sur K_0 et 32 bits d'information sur K_8 en chiffrant $N = 2^{115{,}62}$ clairs pour une complexité de $2^{115{,}62}$ chiffrements de Crypton et de 2^{96} calculs annexes. On peut également à partir des mêmes clairs et chiffrés, en répétant cette attaque sur d'autres positions de carrés, retrouver d'autres bits d'informations sur les clés testés voire, pour un nombre suffisant de telles opérations, la clé entière.

Cette attaque est plus rapide qu'une recherche exhaustive et que toutes les attaques différentielles connues car la probabilité de la meilleure caractéristique sur huit tours est 2^{-120}.

1.5 Attaque Stochastique Utilisant une Partition des Classes

1.5.1 Calcul des Probabilités de Transition

La cryptanalyse présentée dans ce paragraphe est fondée sur la propriété linéaire vue précédemment. On considère à présent un bloc courant A représenté par :

$$A = \begin{pmatrix} * & * & * & * \\ * & X & * & Y \\ * & * & * & * \\ * & Z & * & T \end{pmatrix}$$

où le quartet d'octets (X, Y, Z, T) est associé à une paire d'indices (i, j). On peut alors partitionner l'ensemble des blocs d'entrée en 16 classes selon la valeur de $\Phi_0[X, Y, Z, T] = \Phi_{0,i,j}$ exprimée sur 4 bits (ou selon la valeur également sur 4 bits de $\Phi_1[X, Y, Z, T] = \Phi_{1,i,j}$).

De la propriété vue à la section 1.2.2, il découle que la valeur de Φ_0 ou la valeur de Φ_1 reste inchangée après passage dans la partie linéaire de Crypton, si les valeurs de Φ_0 ou de Φ_1 à la sortie sont calculées à partir du carré de sorties (i', j') déduit de (i, j).

Il est également facile de voir que l'addition de clé σ ne fait qu'ajouter à $\Phi_0[X, Y, Z, T]$ ou à $\Phi_1[X, Y, Z, T]$ une constante sur 4 bits impliquant 4 octets de clé associés au quartet (X, Y, Z, T) de valeur $\Phi_0[K_X, K_Y, K_Z, K_T]$ ou $\Phi_1[K_X, K_Y, K_Z, K_T]$.

Il ne reste alors plus qu'à étudier le comportement des valeurs de classes des fonctions $\Phi_0[X, Y, Z, T]$ et $\Phi_1[X, Y, Z, T]$ après passage dans la partie non linéaire γ de Crypton, composée des deux boîtes S S_0 et S_1. Pour cela, pour chaque boîte S, on calcule les quantités :

$$\begin{gathered} \#\{X, Y, Z, W \in [0, 255]^4 \ / \ \Phi_0[X, Y, Z, T] = a \\ \text{et } \Phi_0[S_\epsilon(X), S_\epsilon(Y), S_\epsilon(Z), S_\epsilon(T)] = b\} - (256)^3 \\ \text{et} \\ \#\{X, Y, Z, W \in [0, 255]^4 \ / \ \Phi_1[X, Y, Z, T] = a \\ \text{et } \Phi_1[S_\epsilon(X), S_\epsilon(Y), S_\epsilon(Z), S_\epsilon(T)] = b\} - (256)^3 \end{gathered}$$

où ϵ vaut 0 ou 1 selon la valeur de i et j et a et b appartiennent à $[0, 15]$. Ces valeurs représentent les biais par rapport à la valeur moyenne $(256)^3$. On obtient ainsi quatre matrices 16×16 (une pour Φ_0 et S_1 (voir annexe), une pour Φ_0 et S_0, une pour Φ_1 et S_1 et une pour Φ_1 et S_1) dont les colonnes sont indexées par la valeur a et les lignes par b. La valeur associée à la colonne a et à la ligne b représente, à un facteur multiplicatif valant $(256)^4$ près, le biais de la probabilité de transition entre la classe des valeurs d'entrée associée à la valeur a et la classe des valeurs de sorties à la valeur b.

On peut donc déduire de ce qui précède le comportement de Φ_0 ou de Φ_1 pour un tour complet de Crypton grâce à une matrice 16×16 de probabilités de transition obtenue en multipliant une des quatre matrices précédentes par une matrice de permutation elle aussi de taille 16×16 liée aux quatre octets de clé impliqués dans le calcul de Φ_0 ou de Φ_1 lors du passage dans σ, comme vu à la section 1.1.2. Le coefficient de cette matrice associé à la ligne b et à la colonne a vaut 1 si $b = a \oplus \Phi_0[K_X, K_Y, K_Z, K_T]$ dans le cas d'une étude sur Φ_0.

Sous l'hypothèse heuristique que le comportement du chiffrement est presque markovien, on peut multiplier entre elles les matrices de probabilités de transition afin d'exprimer les transitions de la fonction d'étage de Crypton sur n tours. La matrice obtenue est dépendante des sous-clés et donc de la clé, plus précisément elle dépend de 4 combinaisons linéaires des bits des sous-clés de chaque tour. Cependant, l'ordre de grandeur des biais de transition est le même pour la plupart des valeurs des bits de clé et devient encore plus grand dans quelques cas particuliers comme celui où tous les mots de 4 bits des sous-clés impliqués dans le calcul de Φ_i sont nuls, ceci étant lié au fait que la meilleure transition est celle qui fait correspondre 0 à 0.

1.5.2 Principe de l'Attaque et Résultats Pratiques

La cryptanalyse présentée ici est une attaque à clairs choisis portant sur une version de Crypton à 8 étages, c'est à dire comprenant l'addition de clé initiale et la transformation finale. Pour ce faire, on calcule les probabilités de transition attendues sur 6 tours internes (ce qui représentera un distingueur sur 6 étages internes de Crypton) afin de tester 4 octets de la clé de K_0, qui déterminent la valeur de Φ_i à l'entrée des 6 tours testés à une constante inconnue près, et 4 octets de la clé K_8, qui détermine la valeur de Φ_i à la sortie des mêmes 6 tours.

On choisit tout d'abord un couple d'indices (i, j) et on chiffre N textes clairs choisis de façon à ce que après la première addition de clé et le premier tour on ait $\Phi_{i,j} = \alpha \oplus \Phi_\epsilon(K_1)$, c'est à dire que la valeur d'entrée des 6 tours internes soit $\Phi_{i,j}$ en supposant que la distribution à la sortie des 6 tours internes des probabilités induites par Φ_ϵ soit suffisamment déséquilibrée.

On utilise un test du χ^2 comme décrit dans la section 1.1.2 pour tester les 4 octets de la clé K_0 et les 4 octets de la clé K_8. On teste les 4 octets de K_0 selon la valeur attendue de $\Phi_{i,j}$. Pour chacune des 2^{32} valeurs possibles des 4 octets de K_8 impliquées dans le calcul de Φ_ϵ, on déchiffre partiellement les chiffrés C répartis selon les valeurs possibles des mots de 4 bits impliqués dans le calcul de Φ_ϵ pour obtenir les valeurs possibles à la sortie des 6 tours sur lesquels portent le test. Une fois le déchiffrement partiel effectué, on calcule (en au plus 2^{32} opérations) la distribution des valeurs de sortie en fonction de la valeur inconnue supposée $\alpha \oplus \Phi_\epsilon(K_1)$ déduite de l'hypothèse sur les 4 octets de clé de K_0, notée également Φ et la valeur du χ^2 associée à la distribution des 16 valeurs possibles. On suppose que la valeur de l'indicateur donné par le χ^2 est plus haute pour la bonne valeur de la clé. On utilise le même type de test pour déterminer la valeur d'entrée Φ et les 4 octets de la clé K_0.

Le nombre N de clairs à chiffrer nécessaires à cette attaque est inversement proportionnel à la somme des carrés des biais attendus à priori. On peut donc déduire N des biais calculés à partir des matrices décrites dans le paragraphe précédent. Le meilleur résultat est obtenu pour Φ_1. On en déduit donc que en moyenne sur les 6-uplets de mots de 4 bits de clé, N vaut environ 2^{112}. Pour le meilleur 6-uplet de mots de 4 bits de clé (le 6-uplet nul), un nombre de clairs égal à 2^{104} suffit.

On obtient donc 32 bits d'information sur la clé K_0 et 32 bits d'information sur la clé K_8 à partir d'un nombre de clairs choisis égal à $N = 2^{112}$. La complexité d'une telle attaque est donc de 2^{112} chiffrements de Crypton et d'environ 2^{100} calculs annexes. On peut également en répétant cette attaque reconstituer la clé entière en utilisant environ 2^{116} clairs choisis.

1.6 Résultats Concernant Crypton v1.0

Crypton v1.0 a été introduit par Chae Hoon Lim en 1999 dans [Lim99] afin de modifier la génération des sous-clés et de changer les boîtes S.

Au lieu d'utiliser deux boîtes S comme dans la version initiale, Crypton v1.0 utilise 4 boîtes S S_0, S_1, S_2, S_3 qui vérifient $S_0^{-1} = S_2$ et $S_1^{-1} = S_3$. Les transformations γ_0 et γ_1 deviennent donc :

$$B = \gamma_0(A) \Leftrightarrow b_{i,j} = S_{i+j \bmod 4}(a_{i,j})$$
$$B = \gamma_1(A) \Leftrightarrow b_{i,j} = S_{i+j+2 \bmod 4}(a_{i,j})$$

En revanche, les relations $\gamma_0^{-1} = \gamma_1$ et $\gamma_1^{-1} = \gamma_0$ restent vraies et le chiffrement et le déchiffrement sont toujours identiques.

Les nouvelles boîtes S de Crypton v1.0, sont déduites d'une même boîte S, qui est une involution d'octets, choisie pour ses bonnes propriétés de diffusion, le but visé étant ici de réduire le nombre de caractéristiques différentielles et linéaires de poids faible les plus probables et donc d'accélérer la diffusion, même si les coefficients DP et LP de ces nouvelles boîtes sont les mêmes que pour S_0 et S_1.

Selon nos calculs, ces nouvelles boîtes S permettent une résistance bien meilleure contre les attaques présentées ici.

Si on essaye d'appliquer la première attaque utilisant les propriétés différentielles sur 6 tours à Crypton v1.0, en considérant une différence d'entrée égale à $\Delta_1 = (49, 49)$ alors la probabilité que la sortie au bout de 6 tours soit un couple de la forme $(\delta_1,\ \delta_2)$ avec $(\delta_1, \delta_2) \in D_1 \times D_1$ est $2^{-151.23}$, en tenant compte de toutes les valeurs intermédiaires possibles avec une alternance d'éléments de Δ_1 et d'éléments de Δ_2. La valeur $2^{-151{,}23}$ représente la meilleure probabilité de transition sur 6 tours de Crypton v1.0 ce qui est largement plus que la meilleure probabilité de transition de 6 tours de Crypton, égale à $2^{-112.62}$.

Dans le cas d'une cryptanalyse stochastique utilisant une partition en 16 classes, le nombre de textes clairs nécessaires à l'attaque est d'au moins 2^{135}, cette valeur étant calculée en utilisant Φ_1 et le meilleur 6-uplet de mots de clés de 4 bits. Cette valeur est également bien supérieure à celle trouvée dans le cas de la version initiale de Crypton et supérieure au nombre de clairs (2^{128}).

Crypton v1.0 résiste donc bien mieux aux attaques présentées ici. Cela semble être une conséquence directe des critères de choix des nouvelles boîtes S. En effet, les nouvelles boîtes S étant choisies pour diminuer le nombre de caractéristiques différentielles et linéaires de poids faible les plus probables à chaque tour, les coefficients de la matrice représentant les probabilités de transition sont plus petits et les performances des attaques s'en ressentent. On peut donc dire que face aux deux attaques présentées ici le changement de boîtes S semble judicieux.

1.7 Conclusion

Nous venons de voir deux attaques stochastiques portant sur une version de Crypton à huit tours qui sont plus rapides qu'une recherche exhaustive et plus efficaces que toutes les attaques connues.

La première attaque est proche d'une attaque différentielle tronquée mais nous pensons que l'utilisation de matrices de probabilités donne des résultats plus précis.

Ces deux attaques ne menacent pas la sécurité de Crypton dans sa version complète mais mettent cependant en évidence une propriété particulière de la partie linéaire de Crypton qui peut être vue comme une faiblesse. La nouvelle version de Crypton grâce au choix judicieux des nouvelles boîtes S résistent beaucoup mieux à ces attaques.

Annexe : Matrice des Biais de Transition pour Φ

	0	1	2	3	4	5	6	7
0	204800	-36864	124928	-28672	-53248	77824	-18432	69632
1	-28672	-83968	-4096	-94208	-94208	63488	-118784	36864
2	-55296	53248	-28672	45056	112640	-94208	53248	-86016
3	-69632	102400	-94208	75776	28672	-45056	53248	-51200
4	-114688	53248	-112640	45056	-36864	-94208	6144	-86016
5	-20480	67584	-12288	110592	143360	-47104	135168	-53248
6	34816	-36864	77824	-28672	-92160	77824	-102400	69632
7	86016	-86016	77824	-92160	-45056	28672	-36864	67584
8	28672	-135168	14336	-126976	-57344	12288	-71680	4096
9	102400	-2048	94208	-12288	-61440	145408	-53248	118784
10	-55296	36864	-61440	28672	112640	-77824	86016	-69632
11	-86016	102400	-77824	75776	45056	-45056	36864	-51200
12	-77824	118784	-59392	110592	106496	4096	116736	12288
13	-53248	-14336	-77824	28672	12288	-129024	36864	-135168
14	34816	-53248	45056	-45056	-92160	94208	-69632	86016
15	69632	-86016	94208	-92160	-28672	28672	-53248	67584
	8	9	10	11	12	13	14	15
0	4096	-77824	-30720	-86016	-114688	36864	-116736	45056
1	143360	-47104	151552	-36864	-20480	100352	-28672	61440
2	-75776	94208	-69632	102400	51200	-53248	12288	-61440
3	-45056	61440	-53248	22528	86016	-86016	94208	-79872
4	4096	94208	-14336	102400	106496	-53248	161792	-61440
5	-126976	30720	-102400	53248	4096	-83968	-20480	-77824
6	63488	-77824	118784	-86016	-38912	36864	-61440	45056
7	61440	-45056	36864	-38912	-102400	69632	-77824	96256
8	90112	-12288	88064	-20480	-36864	135168	-55296	143360
9	45056	-129024	20480	-118784	-86016	18432	-61440	-20480
10	-75776	77824	-102400	86016	51200	-36864	45056	-45056
11	-61440	61440	-36864	22528	102400	-86016	77824	-79872
12	-73728	-4096	-75776	4096	20480	-118784	43008	-126976
13	-61440	112640	-69632	135168	102400	-2048	110592	4096
14	63488	-94208	86016	-102400	-38912	53248	-28672	61440
15	45056	-45056	53248	-38912	-86016	69632	-94208	96256

TAB. 1.1 – Matrice des distributions pour S_1 et Φ_0

Chapitre 2

Cryptanalyse de Rijndael

Ce chapitre est consacré à l'étude de Rijndael [DR99], choisi comme AES le 2 octobre 2000 [AES98].

Nous allons dans un premier temps décrire cet algorithme symétrique de chiffrement par blocs et les propriétés particulières qu'il utilise.

Nous nous intéresserons dans un deuxième temps aux cryptanalyses menées à ce jour contre des versions de Rijndael à 6 et 7 tours. Toutes les cryptanalyses présentées ici utilisent la structure orientée octet de l'algorithme et mettent en lumière des propriétés particulières de celui-ci. La première attaque, nommée "Square Attack", est dûe aux auteurs eux-mêmes et est décrite dans l'article initial [DR98]. Elle utilise une propriété particulière sur trois étages internes permettant de construire un distingueur sur ces mêmes trois étages. La deuxième, dûe à Ferguson et al. [FKS+00], est une amélioration de la "Square Attack". Notons également que les mêmes auteurs décrivent une related key attack. La troisième attaque est dûe à Stefan Lucks [Luc00] et exploite une faiblesse du cadencement de clé pour améliorer la "Square Attack". La quatrième attaque présentée ici que l'on nomme la "Gilbert-Minier attack", décrite dans [GM00], s'appuie également sur une propriété portant sur trois tours internes de Rijndael qui exploite des collisions entre des fonctions partielles. Cette propriété plus forte que celle utilisée dans la "Square Attack" permet de construire un distingueur sur quatre étages internes et donc de monter une attaque sur une version de Rijndael à sept étages utilisant 2^{32} textes clairs choisis et ayant une complexité de 2^{144} chiffrements Rijndael sous des clés de taille 192 et 256 bits. Nous présentons également une variante de cette attaque pour des clés de 128 bits dont la complexité totale est légèrement inférieure à celle d'une recherche exhaustive.

Nous présenterons également quelques tentatives d'attaques n'ayant pas abouti et utilisant des méthodes polynômiales.

2.1 Description de l'Algorithme

2.1.1 Généralités

Rijndael est un algorithme itératif de chiffrement par blocs qui chiffre des blocs de longueur 128, 192 ou 256 bits sous des clés de taille 128, 192 ou 256 bits. Il est composé d'un nombre variable d'étages : 10, 12 ou 14, fonction de la longueur de la clé et de celle des blocs à chiffrer. Il utilise une structure parallèle mettant en oeuvre quatre transformations où les blocs à chiffrer sont représentés par des matrices d'octets :

- ByteSub : substitution non linéaire faisant appel à une boîte S pour garantir de bonnes propriétés de confusion.
- ShiftRow : application linéaire qui décale les lignes de la matrice.
- MixColumn : application linéaire (multiplication matricielle dans le corps $GF(256)$, fondée sur les codes MDS) garantissant une bonne diffusion.
- RKadd : addition de clé classique.

On peut donc représenter la fonction de chiffrement par le schéma de la figure 2.1.

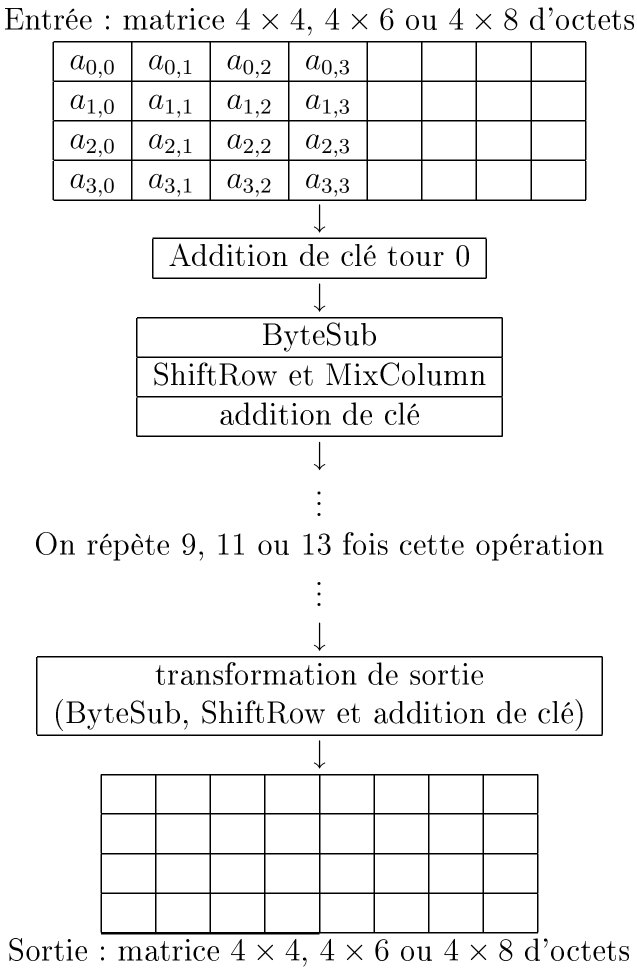


FIG. 2.1 – Schéma de Chiffrement de Rijndael

La fonction de déchiffrement de Rijndael se déduit très facilement de la fonction de chiffrement. Il suffit, avec la même structure générale, d'utiliser les fonctions inverses des précédentes et d'inverser leur position dans le tour.

Dans la suite, nous nous contenterons d'étudier le cas de blocs à chiffrer de longueur 128 bits que nous représenterons par la matrice 4×4 d'octets suivante :

A =	$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$
	$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$
	$a_{2,0}$	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$
	$a_{3,0}$	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$

2.1.2 La Transformation ByteSub

Définition

L'opération ByteSub fait intervenir une boîte S, permutation non linéaire de $GF(256)$ dans lui-même, de la façon suivante :

$b_{0,0}$	$b_{0,1}$	$b_{0,2}$	$b_{0,3}$		$S(a_{0,0})$	$S(a_{0,1})$	$S(a_{0,2})$	$S(a_{0,3})$
$b_{1,0}$	$b_{1,1}$	$b_{1,2}$	$b_{1,3}$	=	$S(a_{1,0})$	$S(a_{1,1})$	$S(a_{1,2})$	$S(a_{1,3})$
$b_{2,0}$	$b_{2,1}$	$b_{2,2}$	$b_{2,3}$		$S(a_{2,0})$	$S(a_{2,1})$	$S(a_{2,2})$	$S(a_{2,3})$
$b_{3,0}$	$b_{3,1}$	$b_{3,2}$	$b_{3,3}$		$S(a_{3,0})$	$S(a_{3,1})$	$S(a_{3,2})$	$S(a_{3,3})$

Construction de la boîte S

La boîte S utilisée dans la transformation ByteSub est la composée de deux transformations :

- L'inversion (avec $0^{-1} = 0$) dans le corps $GF(256)$ représenté sous la forme $GF(256) \approx GF(2)[x]/(x^8+x^4+x^3+x+1)$. Cette opération permet de garantir la plus forte non linéarité possible, c'est à dire de rendre minimaux les coefficients utilisés en cryptanalyse différentielle et en cryptanalyse linéaire. De plus, le haut degré du polynôme engendré par l'inversion permet de se prémunir contre les attaques par interpolation décrites dans la section 2.2.4. On peut également noter que le choix de cet exposant prémunit l'AES contre une attaque différentielle d'ordre supérieur, comme décrit dans [CV02] et mentionné dans la section 2.3.2.
- Une transformation affine de $GF(2)^8$ dans $GF(2)^8$ de la forme $y = A \cdot x + B$ où A est une matrice 8×8 à coefficients dans $GF(2)$ et où B est un vecteur colonne de taille 8 également à coefficients dans $GF(2)$. Les octets d'entrée et de sortie sont représentés sous forme d'un vecteur colonne de taille 8 à coefficients dans $GF(2)$:

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \times \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

  L'utilisation d'une transformation affine est destinée à cacher les propriétés algébriques remarquables propres à l'inversion. Cette dernière a été également choisie pour éviter la présence de points fixes (i.e. tels que $S(x) = x$).

Précisons également les valeurs des coefficients DP et LP de la boîte S :

$$DP_S = 2^{-6} \text{ et } LP_S = 2^{-6}.$$

2.1.3 ShiftRow

L'opération ShiftRow consiste à faire subir une permutation circulaire vers la gauche aux lignes de la matrice à chiffrer :

m	n	o	p	pas de décalage $\longleftarrow$	m	n	o	p
i	j	k	l	décalage de 1 $\longleftarrow$	j	k	l	i
d	e	f	g	décalage de 2 $\longleftarrow$	f	g	d	e
w	x	y	z	décalage de 3 $\longleftarrow$	z	w	x	y

2.1.4 MixColumn

MixColumn peut être représenté comme une multiplication matricielle portant sur chaque colonne de la matrice représentant le bloc d'entrée :

$$\begin{bmatrix} b_{0,i} \\ b_{1,i} \\ b_{2,i} \\ b_{3,i} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} a_{0,i} \\ a_{1,i} \\ a_{2,i} \\ a_{3,i} \end{bmatrix} \quad \text{pour } i \text{ allant de 0 à 3}$$

Pour le calcul, on représente le corps $GF(256)$ par $GF(2)[x]/(x^8+x^4+x^3+x+1)$ comme précédemment et chaque octet par un polynôme de degré au plus 7. La multiplication est donc définie comme une multiplication de polynômes binaires modulo $x^8+x^4+x^3+x+1$. Par exemple, la multiplication par $'02'$ correspond à la multiplication par x, les chiffres entre côtes étant exprimés en hexadécimal. On note $\bullet$ cette multiplication.

On peut alors représenter chaque colonne de la matrice d'entrée ou de sortie comme un polynôme de degré au plus 3 à coefficient dans $GF(256)$. La transformation MixColumn peut être vue comme la multiplication modulo x^4+1 des polynômes associés à chacune des colonnes par les décalés successifs du polynôme constant $c(x) = '03'x^3 + '01'x^2 + '01'x + '02'$.

Il est également important de noter que MixColumn vérifie la propriété MDS, c'est à dire rend maximum le "nombre de branches", i.e. le nombre d'octets actifs en entrée/sortie de la partie linéaire, notion introduite par J. Daemen dans sa thèse [Dae95] et développée dans [DR99] présentée dans la section 2.4.2 de la première partie de cette thèse. Dans le cas de Rijndael, les vecteurs colonnes d'octets sont de taille 4 et le nombre de branches est maximal et vaut 5. Précisons que la notion de "nombre de branches" représente une mesure de la puissance de diffusion de la partie linéaire du chiffrement.

Précisons également que sur 4 tours et comme décrit dans la section 2.4.2, le nombre minimum de boîtes S actives est d'au moins 25 ce qui compte tenu des coefficients DP_S et LP_S de la boîte S rend les attaques différentielles et linéaires impraticables. En effet, sur 4 étages, il n'y a pas de différentielle qui ait une probabilité de se produire supérieure à 2^{-150} et il n'y a pas de relation linéaire qui ait une probabilité de se produire supérieure à 2^{-75}.

2.1.5 RKadd

L'addition de clé est un simple Xor bit à bit sur tous les octets de la matrice :

$$\begin{array}{|c|c|c|c|}\hline b_{0,0} & b_{0,1} & b_{0,2} & b_{0,3} \\ \hline b_{1,0} & b_{1,1} & b_{1,2} & b_{1,3} \\ \hline b_{2,0} & b_{2,1} & b_{2,2} & b_{2,3} \\ \hline b_{3,0} & b_{3,1} & b_{3,2} & b_{3,3} \\ \hline \end{array} = \begin{array}{|c|c|c|c|}\hline a_{0,0} & a_{0,1} & a_{0,2} & a_{0,3} \\ \hline a_{1,0} & a_{1,1} & a_{1,2} & a_{1,3} \\ \hline a_{2,0} & a_{2,1} & a_{2,2} & a_{2,3} \\ \hline a_{3,0} & a_{3,1} & a_{3,2} & a_{3,3} \\ \hline \end{array} \oplus \begin{array}{|c|c|c|c|}\hline k_{0,0} & k_{0,1} & k_{0,2} & k_{0,3} \\ \hline k_{1,0} & k_{1,1} & k_{1,2} & k_{1,3} \\ \hline k_{2,0} & k_{2,1} & k_{2,2} & k_{2,3} \\ \hline k_{3,0} & k_{3,1} & k_{3,2} & k_{3,3} \\ \hline \end{array}$$

2.1.6 Le Cadencement de Clé

Cette opération permet de générer les sous-clés du schéma de chiffrement à partir de la clé maître. Il est nécessaire de décrire cette opération car deux attaques l'utilisent, l'une due à S. Lucks [Luc00], l'autre due à Ferguson et al. [FKS+00]. Le cadencement de clé se déroule en deux étapes : d'abord l'expansion de clé, où la clé de chiffrement est transformée en une clé étendue dont la longueur est égale au nombre de bits nécessaires à la formation de toutes les sous-clés, ensuite la sélection des sous-clés générées à partir de la clé étendue.

L'Expansion de Clé

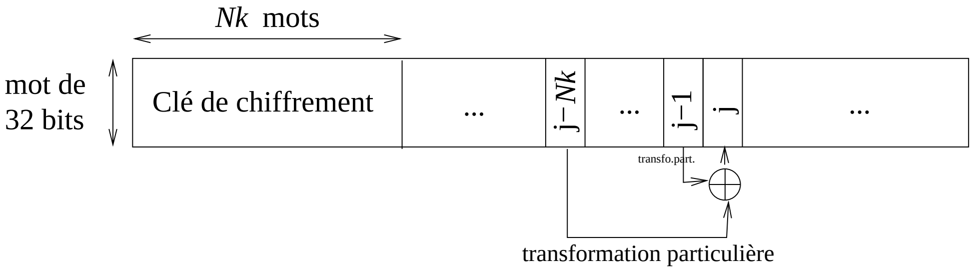
On note :
- Nr le nombre de tours.
- Nb le nombre de colonnes des blocs à chiffrer dans la représentation matricielle, c'est à dire la longueur des blocs divisée par 32.
- Nk le nombre de colonnes de la clé de chiffrement ($= \frac{\text{longueur de clé}}{32}$).

La clé étendue est représentée par un tableau de mots de 4 octets qui sont notés $W[0]$ à $W[Nb*(Nr+1)-1]$. On note CC la clé de chiffrement, représentée de la même manière. On peut alors représenter l'expansion de clé par la procédure récursive suivante :
- Les Nk premières valeurs de la clé étendue (c'est à dire les Nk premiers mots de 4 octets) reçoivent les Nk premières valeurs de la clé de chiffrement.
- Pour les valeurs de j multiples de Nk, on applique la transformation suivante : $W[j] \leftarrow W[j-Nk] \oplus Sub4Byte(W[j-1]^{<<1}) \oplus Rcon[\frac{j}{Nk}]$ où $Sub4Byte$ est la boîte S de Rijndael du paragraphe 1.2 appliquée à chacun des quatre octets, $^{<<1}$ désigne la rotation circulaire d'un octet vers la gauche et $Rcon$ une table de constantes.
- Pour les valeurs i telles que $i \bmod Nk \neq 0$ et $i \bmod Nk \neq 4$ pour $Nk = 8$, on utilise la relation : $W[i+j] \leftarrow W[i+j-Nk] \oplus W[i+j-1]$.
- Si dans le cas $Nk = 8$ on a $i \bmod Nk = 4$, alors on a $W[i] \leftarrow W[i-Nk] \oplus Sub4Byte(W[i-1])$

L'expression algorithmique en est :

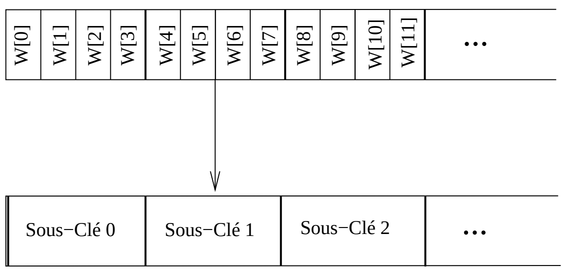
Pour i allant de 0 à $(Nk-1)$ faire
$W[i] \leftarrow CC[i]$
Fin Pour
Pour i allant de Nk à $Nb*(Nr+1)-1$ faire
Si (i mod Nk=0) alors
$W[i] \leftarrow W[i-Nk] \oplus Sub4Byte(W[i-1]^{<<1}) \oplus Rcon[\frac{i}{Nk}]$
Sinon Si $(Nk > 6)$ et (i mod Nk=4) alors
$W[i] \leftarrow W[i-Nk] \oplus Sub4Byte(W[i-1])$
Sinon $W[i] \leftarrow W[i-Nk] \oplus W[i-1]$
Fin Si
Fin Si
Fin Pour

On peut résumer l'expansion de clé par le schéma suivant :



Sélection des Sous-Clés

La sous-clé i, de taille Nb est simplement donnée par les mots $W[Nb*i]$ à $W[Nb*(i+1)-1]$. On obtient par exemple le schéma suivant si $Nb=4$:



2.1.7 Conclusion

Nous venons de donner toutes les spécifications de Rijndael. Il est à noter que dans toutes les transformations utilisées, une seule est non linéaire modulo 2 : la boîte S ou plus exactement, l'inversion dans $GF(256)$ contenue dans cette boîte S. C'est peut être ce que l'on peut actuellement considérer comme la principale "faiblesse" de Rijndael.

2.2 Les Attaques Connues sur Rijndael

Nous allons voir dans ce paragraphe différentes attaques montées à ce jour contre Rijndael. Aucune n'a une efficacité réelle, elles ne font que mettre en évidence quelques unes des propriétés particulières de l'algorithme. Leur but est peut être de tenter de relever le "défi" proposé par J. Daemen et V. Rijmen dans [DR98], lorsqu'ils disent : " For the different block lengths of Rijndael no extensions to 7 rounds [of a known attack] faster than exhaustive key search have been found."

Comme je l'ai signalé précédemment, Rijndael, grâce à ses bonnes propriétés de confusion (dues à la boîte S) et de diffusion (liées au choix de la partie linéaire) résiste très efficacement aux différentes classes d'attaques connues, à savoir la cryptanalyse différentielle et différentielle d'ordre supérieur, la cryptanalyse linéaire et les attaques par interpolation.

2.2.1 La "Square Attack"

Cette attaque est la première d'une nouvelle classe d'abord nommée par S. Lucks [Luc01] "attaque par saturation" car il s'agit ici de "saturer" un octet en lui faisant prendre les 256 valeurs possibles et d'utiliser les propriétés induites sur cet octet par la semi-bijectivité de la fonction d'étage itérée trois fois (chaque octet de sortie étant en fait une somme de bijections des octets d'entrée). C'est finalement le nom "Integral Cryptanalysis" proposé par L. Knudsen [Knu02] qui a été retenu par les participants de FSE'02.

L'attaque proposée par les auteurs de l'algorithme dans les spécifications soumises à l'AES est une variante de l'attaque proposée par les mêmes auteurs sur un autre de leurs algorithmes au design similaire, Square [DKR97]. Il s'agit ici de créer un distingueur sur trois tours pour monter une attaque sur 6 étages qui utilise 2^{32} textes clairs et qui nécessite l'exécution de 2^{72} chiffrements.

Définissons tout d'abord l'ensemble Λ qui contient 256 blocs à chiffrer (c'est à dire 256 matrices d'octets de taille 4×4) tous distincts. Deux blocs appartiennent au même ensemble Λ si ils sont égaux partout excepté sur un octet de position prédéfinie :

$$\forall x, y \in \Lambda : \begin{cases} x_{i,j} \neq y_{i,j} \text{ pour un } i \text{ et un } j \text{ déterminés} \\ x_{i,j} = y_{i,j} \text{ pour toutes les autres valeurs de } i \text{ et } j \end{cases}$$

i et j variant de 0 à 3 dans le cas qui nous intéresse. L'ensemble présenté ici ne contient qu'un seul octet actif mais il est à noter qu'on peut définir des ensembles Λ avec plusieurs octets actifs, par exemple, en considérant comme actifs tous les octets d'une colonne. Dans tous les cas, les transformations ByteSub et RKadd transforment un ensemble Λ en un autre ensemble Λ avec le même nombre d'octets actifs.

Regardons à présent comment utiliser les propriétés de semi-bijectivité des transformations internes de Rijndael sur trois étages (voir figure 2.4) pour monter une attaque sur 4 étages. On considère donc un ensemble Λ avec un seul octet actif et on étudie l'évolution des octets actifs sur trois tours. Le premier MixColumn transforme l'octet actif en une colonne d'octets actifs, diffusés sur les quatre colonnes par le ShiftRow du deuxième tour et transformés en une pleine matrice d'octets actifs à la fin du MixColumn du deuxième

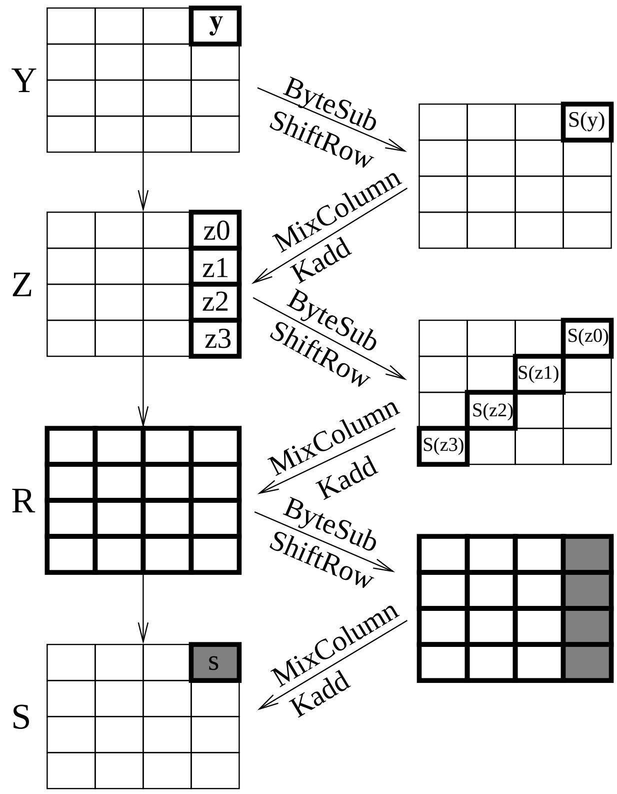


FIG. 2.2 – 3 Tours Internes de Rijndael

tour. Jusqu'à l'entrée du MixColumn du troisième tour, les transformations impliquées constituent une bijection. On utilise à présent une propriété due à la semi-bijectivité du MixColumn du troisième tour et à la bonne répartition des éléments d'un ensemble Λ quelconque. On note a tout octet actif à l'entrée du troisième MixColumn. On a, pour un octet particulier :

$$\bigoplus_{b=MixColumn(a),a\in GF(256)} b_{i,j} = 0 \quad (1)$$

On en déduit donc que tous les octets à l'entrée du quatrième tour sont équilibrés. On considère alors que le quatrième tour est le dernier, c'est à dire qu'il ne contient pas de MixColumn. On a donc une relation du type $a_{i,j} = ByteSub(b_{i',j'} \oplus k_{i,j})$ entre chaque octet d'entrée et de sortie. On peut alors déterminer la valeur de l'octet $k_{i,j}$ par la boucle suivante :

choisir une valeur pour $k_{i,j}$
calculer les valeurs des $b_{i',j'}$ pour tous les éléments de Λ
Si les $b_{i',j'}$ vérifient la propriété (1) alors
    renvoyer $k_{i,j}$
Sinon
    refaire pour une autre valeur de $k_{i,j}$
Fin Si

On peut utiliser cet algorithme pour déterminer tous les octets de la dernière sous-clé k.

Cette attaque nécessite le chiffrement de 2^8 textes clairs, en théorie tout du moins. Pour être sûr d'éliminer toutes les fausses alarmes sur l'octet de clé, on préfère augmenter légèrement la complexité de l'attaque en chiffrant 2^9 textes clairs choisis, c'est à dire deux ensemble Λ.

On peut étendre ce résultat en ajoutant un cinquième tour à la fin. Pour calculer la valeur des $b_{i,j}$ à partir du cinquième tour et non plus du quatrième, on a besoin de connaître 4 octets supplémentaires de la dernière clé. Comme dans l'attaque sur 4 tours, on élimine les mauvaises clés en vérifiant la propriété (1). Pour éviter toutes les fausses alarmes de clés, on utilise 4 ensembles Λ. On a besoin de tester 2^{40} valeurs de clé. Cette attaque nécessite donc 2^{11} textes clairs choisis et l'exécution de 2^{40} chiffrements.

On peut également étendre cette attaque en ajoutant un tour au début. L'idée est d'obtenir un ensemble Λ à la sortie du premier tour contenant un seul octet actif. Pour ce faire, on a besoin de connaître la valeur de 4 octets de clé utilisés avant l'entrée du deuxième tour. On considère donc un ensemble de 2^{32} textes clairs et grâce à l'hypothèse sur les quatres octets de clé, on en sélectionne 256 pour former un ensemble Λ. On peut alors appliquer l'attaque sur 4 étages. On répète ainsi l'attaque pour différentes hypothèses sur les 4 octets de clés. Cette attaque nécessite 2^{32} textes clairs choisis et l'exécution de 2^{40} chiffrements.

On peut combiner les deux extensions à l'attaque sur 4 étages pour obtenir une attaque sur 6 étages utilisant 2^{32} textes clairs choisis et l'exécution de 2^{72} chiffrements.

2.2.2 L'Attaque de S. Lucks Présentée à la Troisième Conférence AES

Il est à noter que l'on peut étendre à 7 étages la Square Attack pour des longueurs de clé égales à 192 et 256 bits simplement en cherchant les 16 octets de la clé du dernier étage. Cela ajoute une recherche exhaustive sur les 128 bits de la dernière clé pour une complexité totale de 2^{200} chiffrements sans changer le nombre de textes clairs nécessaires.

L'amélioration de cette attaque sur 7 étages de Rijndael exploite une petite faiblesse du cadencement de clé. En effet, on a la propriété particulière suivante : si Nk représente le nombre de colonnes d'une matrice à chiffrer et $W[i]$ la colonne de la clé étendue au rang i, il est très facile de voir que si les clés $W[i]$ et $W[i-1]$ sont connues, on peut en déduire $W[i-Nk]$.

Cette remarque ainsi que le fait qu'on puisse permuter l'ordre des opérations internes de chaque tour de Rijndael permet d'améliorer la Square Attack sur 7 étages. En effet, la connaissance complète de la sous-clé du septième étage permet de calculer d'autres octets des sous-clés nécessaires à l'attaque.

On en déduit alors une attaque sur 7 tours pour des clés de longueur 192 bits. Cette attaque nécessite la connaissance de 2^{32} clairs et environ 2^{184} chiffrements de Rijndael.

2.2.3 L'attaque présentée à FSE 2000

Dans [FKS+00], Ferguson et al. ont présenté plusieurs attaques sur Rijndael : la première utilise une amélioration de la "Square Attack" grâce à une nouvelle technique appelée méthode des "sommes partielles" en sommant non plus sur 256 éléments mais sur 2^{32} valeurs, la seconde est une "Related Key Attack" sur 9 étages de Rijndael sous des clés de 256 bits.

Le facteur de complexité gagné ici est essentiellement lié au fait de ne plus avoir à deviner les 4 octets de clé du premier étage. Pour cela, au lieu de sommer sur 256 valeurs, on somme sur 2^{32} valeurs, qui peuvent être regroupées en 2^{24} ensembles de 256 éléments chacun, chaque ensemble représentant un ensemble Λ à un seul octet actif. Il ne reste alors plus qu'à deviner les 5 octets de clés des deux derniers tours pour faire un déchiffrement partiel sur l'octet impliqué dans la relation (1) en sommant non sur 256 valeurs mais sur les 2^{32} valeurs.

On note alors k_0, k_1, k_2, k_3, k_4 les cinq octets de clé des deux derniers tours utilisés dans la Square Attack, k_4 étant l'octet de clé de l'avant dernier tour. Notons également $c_{i,j}$ le j-ème octet du i-ème chiffré.

On a alors besoin pour remonter les deux derniers tours (le dernier ne contenant pas de MixColumn) de calculer :

$$\sum_i S^{-1}\left[S_0\left[c_{i,0} \oplus k_0\right] \oplus S_1\left[c_{i,1} \oplus k_1\right] \oplus S_2\left[c_{i,2} \oplus k_2\right] \oplus S_3\left[c_{i,3} \oplus k_3\right] \oplus k_4\right]$$

où S_0, S_1, S_2, S_3 représentent l'inverse de la boîte S multipliée par une composante de InvMixColumn.

Afin d'améliorer la complexité de l'attaque, on calcule des sommes partielles pour chaque chiffré c :

$$x_k := \sum_{j=0}^{k} S_j\left[c_j \oplus k_j\right]$$

pour des valeurs de k variant de 0 à 3. On utilise alors la transformation $(c_0, c_1, c_2, c_3) \rightarrow (x_k, c_{k+1}, \cdots, c_3)$ pour déterminer les valeurs des différents k_i. On peut donc découper le calcul global en 4 phases de recherche d'octets de clé faisant appel chacune à 2^{48} calculs. Cela permet de faire diminuer la complexité de la Square Attack : il suffit de faire appel à la boîte S 2^{52} fois, ce qui correspond à environ 2^{44} chiffrements de Rijndael pour $6 * 2^{32}$ clairs (ce nombre prenant en compte les fausses alarmes).

On peut utiliser l'amélioration de la Square Attack combinée à la même remarque que celle de S. Lucks, à savoir que la connaissance complète d'une sous-clé permet de calculer des colonnes d'octets de la sous-clé précédente, pour monter des attaques contre des versions de Rijndael à 7 et 8 étages.

Pour 7 tours, en utilisant une meilleure organisation des clairs et les remarques précédentes, la meilleure attaque présentée pour des clés de longueur 128 bits utilise $2^{128} - 2^{119}$ clairs et nécessite 2^{120} chiffrements. On peut étendre ce résultat à 8 tours dans le cas de clés de longueur 256 bits ; on obtient alors une complexité de 2^{204} chiffrements pour le même nombre de clairs.

Les auteurs présentent également une attaque à clés reliées contre une version à 9 étages de Rijndael pour des longueurs de clé de 256 bits. Cette attaque repose sur la mauvaise diffusion du cadencement de clé et utilise des ensembles Λ à un octet actif pour générer des différences entre les clés ainsi qu'entre les clairs à chiffrer. Elle nécessite 2^{224} chiffrements de Rijndael et 2^{77} clairs.

2.2.4 La "Gilbert-Minier Attack"

Lors de la troisième conférence AES, une autre attaque à clairs choisis a été présentée par Henri Gilbert et moi-même [GM00]. Cette cryptanalyse utilise une propriété sur trois étages internes de Rijndael légèrement plus forte que celle de la "Square Attack", qui permet de construire un distingueur sur 4 étages et de monter une attaque sur une version de Rijndael à 7 étages. L'idée est d'utiliser un "goulet d'étranglement" qui exploite la structure d'octets de Rijndael.

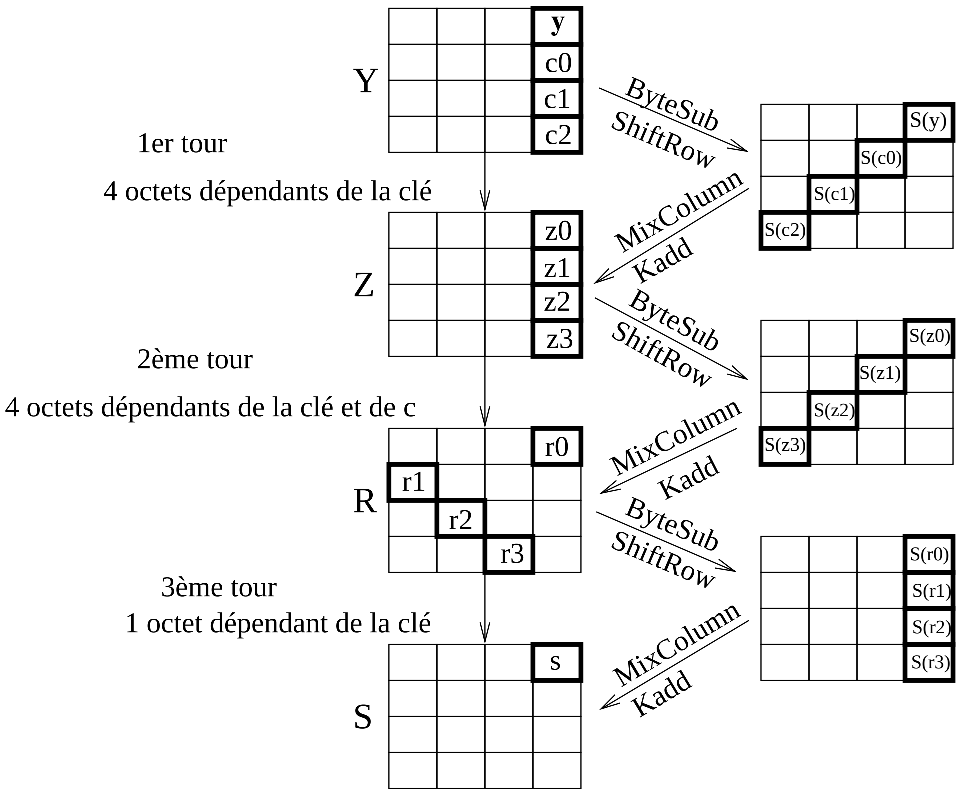


Fig. 2.3 – 3 Tours Internes de Rijndael

Cette attaque utilise le fait qu'il existe une dépendance entre des octets d'entrée et des octets de sortie sur trois étages internes de Rijndael, déterminée par un petit nombre de constantes tant qu'il n'y a pas (pour 1 ou 2 étages) ou peu (3 étages) de repliements dans les diffusions. La figure 2.2 représente trois étages de Rijndael associés à 3 sous-clés inconnues de 128 bits. Y, Z et R représentent les blocs de 128 bits à l'entrée de chaque tour et S le bloc de 128 bits à la sortie du troisième tour. On note $y = Y_{0,3}$, $z_0 = Z_{0,3}$, $z_1 = Z_{1,3}$, $z_2 = Z_{2,3}$, $z_3 = Z_{3,3}$, et ainsi de suite. On note c le triplet d'octet de Y défini par ($c_0 = Y_{1,3}$, $c_1 = Y_{2,3}$, $c_2 = Y_{3,3}$). Précisons encore que toutes les valeurs de Y sont constantes sauf y. On peut donc voir toutes les autres valeurs introduites à la figure 2.2 comme des fonctions de y dépendant de la sous-clé et du choix de c. On les notera donc $z_0^c[y], \cdots, z_3^c[y]$, $r_0^c[y], \cdots, r_3^c[y]$, $t_0^c[y], \cdots, t_3^c[y]$ et $s^c[y]$, y décrivant l'ensemble $[0, \cdots, 255]$. On a les propriétés suivantes :

- l'application $y \to (z_0^c[y], z_1^c[y], z_2^c[y], z_3^c[y])$ est une fonction déterminée par la valeur de quatre octets dépendants de la clé et indépendants du triplet c.
- le quartet d'octets $(r_0^c[y], \cdots, r_3^c[y])$ est une fonction du quartet $(z_0^c[y], \cdots, z_3^c[y])$ déterminée par quatre octets dépendants de c et de la sous-clé de l'étage.
- l'octet s peut être exprimé comme une fonction de $r_0^c[y]$, $r_1^c[y]$, $r_2^c[y]$ et $r_3^c[y]$ et dépend d'un seul octet inconnu lié à la sous-clé de l'étage.

On peut donc en déduire que la fonction $s^c[y]$ est entièrement déterminée par 4 octets dépendants de la clé et du triplet de constantes c et par 5 autres octets dépendants de la clé mais indépendants de c. La fonction partielle $s^c[y]$ est donc déterminée par un nombre réduit d'octets inconnus. On pouvait envisager d'exploiter directement cette propriété. Il nous a cependant semblé plus efficace de l'exploiter au travers de la recherche de collisions définies de la façon suivante :

Définition 10 *Soit c' et c'' deux triplets de constantes. Si $\forall y \in [0, \cdots, 255]$, on a $s^{c'}[y] = s^{c''}[y]$, alors on dit qu'on a une collision.*

En fait, ce n'est pas une, mais 256 collisions qu'on obtient ainsi. Si on fait l'hypothèse heuristique que les octets inconnus se comportent comme des fonctions aléatoires, alors, par le paradoxe des anniversaires, si on prend un ensemble C de $(256)^2$ triplets c, la probabilité d'obtenir une collision est non négligeable. Il est facile de vérifier par simulation qu'il existe de nombreuses collisions.

On peut alors grâce à ces collisions sur trois étages construire un distingueur sur 4 étages. On rajoute donc un quatrième étage qu'on "remonte".

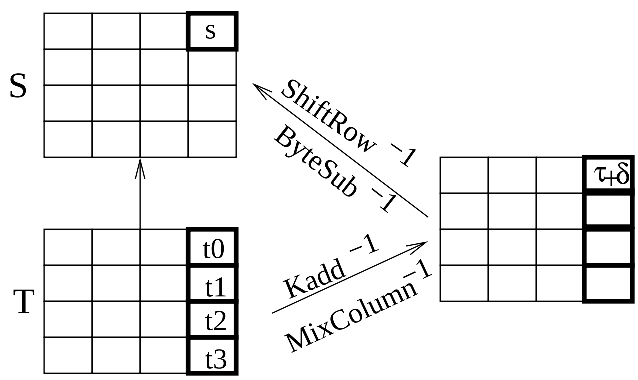


Fig. 2.4 – Étage 4 de Rijndael

On considère le déchiffrement du quatrième tour. On peut alors exprimer s : $s =$

$S^{-1}[(0E \cdot t_0 + 0B \cdot t_1 + 0D \cdot t_2 + 09 \cdot t_3) + \delta]$ où S désigne la boîte S de Rijndael et δ une constante liée à la sous-clé de l'étage. On a alors la propriété suivante :

Propriété 1 *Il existe une collision entre $s^{c'}[y]$ et $s^{c''}[y]$ si et seulement si pour tout y de [0,...,255], on a :*

$$0E \cdot t_0^{c'} + 0B \cdot t_1^{c'} + 0D \cdot t_2^{c'} + 09 \cdot t_3^{c'} = 0E \cdot t_0^{c''} + 0B \cdot t_1^{c''} + 0D \cdot t_2^{c''} + 09 \cdot t_3^{c''}.$$

Pour simplifier les notations, on note $0E \cdot t_0^c + 0B \cdot t_1^c + 0D \cdot t_2^c + 09 \cdot t_3^c = \tau^c[y]$. On peut, pour limiter les calculs à effectuer, tester l'égalité sur un nombre limité de valeurs de y par exemple 16, le nombre d'alarmes étant alors négligeable. On obtient ainsi un distingueur sur quatre étage qui requiert environ 2^{20} clairs choisis, moins de 2^{20} chiffrements et qui fonctionne de la manière suivante :

Choisir un ensemble C contenant environ 2^{16} triplets c
Choisir un ensemble Λ de 16 valeurs de y
Pour chaque valeur du triplet c faire
  Pour y dans Λ faire
    Calculer $\tau^c[y]$
Chercher dans la liste précédente les collisions
sur $\tau^{c'}[y]$ et $\tau^{c''}[y]$

Nous allons à présent voir comment étendre cette attaque à 7 tours en ajoutant un tour au début et deux tours à la fin. On "gagne" le tour du début en utilisant la même méthode que pour la "Square Attack" moyennant 4 octets de clés que nous appellerons k_{ini}. Nous supposerons également que le tour final ne contient pas l'opération MixColumn, même si cela ne change en rien la complexité de l'attaque présentée ici.

Pour "remonter" les deux derniers étages, on utilise une recherche exhaustive améliorée en "coupant" en deux les équations de collision. On pose :

$$\tau_1^{c'} = 0E \cdot t_0^{c'} + 0B \cdot t_1^{c'} \text{ et } \tau_2^{c'} = 0D \cdot t_2^{c'} + 09 \cdot t_3^{c'}.$$

Si on a une collision sur c' et c'', on a $\tau^{c'} = \tau^{c''}$ c'est à dire $\tau_1^{c'} + \tau_2^{c'} = \tau_1^{c''} + \tau_2^{c''}$ donc $\tau_1^{c'} + \tau_1^{c''} = \tau_2^{c'} + \tau_2^{c''}$. τ_1 est lié à t_0 et t_1, τ_2 est lié à t_2 et t_3. t_0 et t_1 dépendent dans les deux dernières colonnes du chiffrement des deux derniers tours de 10 octets de clé inconnus que nous appellerons k_{τ_1} tandis que t_2 et t_3 dépendent dans les deux premières colonnes du chiffrement des deux derniers tours de 10 octets de clés nommés k_{τ_2}. Cet artifice permet de partager la recherche exhaustive en deux et de gagner un facteur d'environ 2^{80}.

Nous pouvons alors utiliser l'algorithme suivant de recherche des 24 octets de $(k_{ini}, k_{\tau_1}, k_{\tau_2})$ pour des longueurs de clés de 192 et 256 bits (dans le cas d'une clé de 128 bits, nous verrons une autre méthode plus performante) :

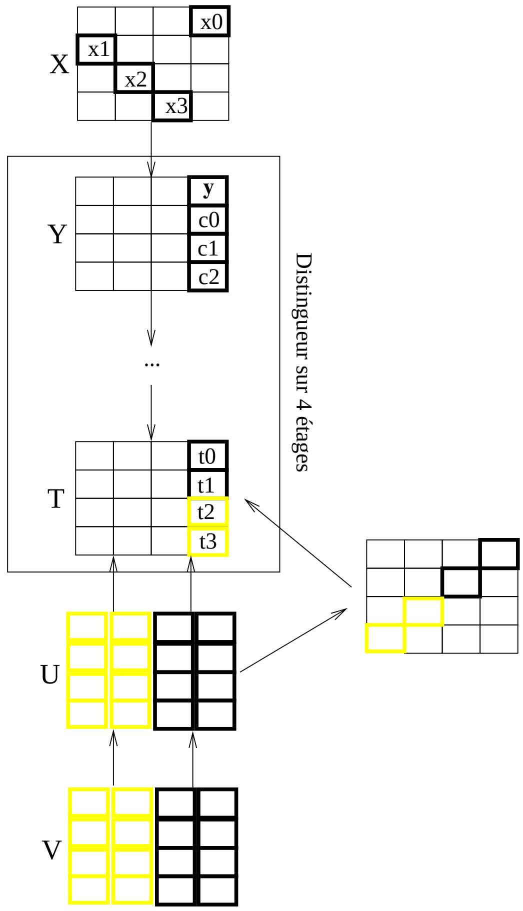


FIG. 2.5 – Attaque sur 7 Étages de Rijndael

Chiffrer les $(256)^4$ textes clairs
Pour k_{ini} variant de (0,0,0,0) à (255,255,255,255) faire
  Partitionner les $(256)^4$ textes clairs choisis
  en $(256)^3$ ensemble Λ^c selon la valeur de c.
  Choisir parmi ces $(256)^3$ ensemble Λ^c $(256)^2$ valeurs de c
  Pour chaque paire (c', c'') faire
    Pour k_{τ_1} variant de (0,...,0) à (255,...,255) faire
      Calculer $(\tau_1^{c'} \oplus \tau_1^{c''})_{y=0..15}$ à partir des chiffrés
      Les mettre dans une table $\mathrm{T}_{k_{ini},c',c''}[k_{\tau_1}]$
    Fin Pour
    Pour k_{τ_2} variant de (0,...,0) à (255,...,255) faire
      Calculer $(\tau_2^{c'} \oplus \tau_2^{c''})_{y=0..15}$ à partir des chiffrés
      Chercher dans la table si les mêmes 16-uplets apparaissent
      Si oui refaire le calcul pour les 256 valeurs de y
        Si égalité retourner $(k_{ini}, k_{\tau_1}, k_{\tau_2})$
      Sinon continuer
      Fin Si
    Fin Pour
  Fin Pour
Fin Pour

Il est à noter que toute la première partie peut être effectuée indépendamment, c'est à dire en précalcul. Cette attaque nécessite 2^{32} textes clairs choisis et l'exécution d'environ 2^{144} opérations (moins complexes que des chiffrements). Sa probabilité de réussite et d'environ 1/2 (car on ne prend que la moitié des valeurs possibles).

Elle détermine entièrement la dernière sous-clé. On peut donc se ramener pour trouver les autres sous-clés à une attaque sur 6 tours, beaucoup moins onéreuse.

On peut en appliquant la propriété particulière du cadencement de clé utilisée par S. Lucks améliorer la complexité de cette attaque d'un petit facteur dans le cas d'une clé de 192 bits. En effet, l'attaque présentée par S. Lucks permet de se limiter à une recherche exhaustive sur 8 octets de k_{τ_2} à la place des 10 octets normalement requis car la connaissance des deux premières colonnes de la sous-clé du dernier étage détermine complètement, en tenant compte du ShiftRow du dernier étage, les deux autres octets de la sous-clé de l'avant dernier étage qui composent également k_{τ_2}, ceci grâce à la propriété particulière du cadencement de clé.

Je vais à présent m'intéresser à l'attaque dédiée au cas où la taille de clé est de 128 bits qui est très légèrement plus rapide que la recherche exhaustive mais qui requiert beaucoup de précalculs et donc de mémoire.

Cette attaque est fondée sur la remarque suivante : les quatres valeurs r_i sont entièrement déterminées par la connaissance de 12 octets de clés que nous nommerons $\phi(K)$. L'idée est donc de précalculer pour un certain nombre de valeurs de c (environ $(256)^2$) et pour chaque valeur possible de $\phi(K)$ les collisions sur s, c'est à dire les valeurs c' et c'' telles que $\forall y \in [0, \cdots, 15], s^{c'}[y] = s^{c''}[y]$. On garde alors en mémoire dans une table à $(256)^{12}$ entrées (correspondant à toutes les valeurs possibles de $\phi(K)$) les valeurs c' et c'' pour lesquelles on obtient une collision.

On calcule les chiffrés des 2^{32} textes clairs qui nous intéressent.

On procède alors à une deuxième phase de précalcul. Pour chaque valeur de k_{ini} et chaque valeur de c, par un déchiffrement du premier étage, on trouve les clairs correspondants aux seize premières valeurs de y auxquels on associe leurs chiffrés respectifs. On garde alors en mémoire selon la valeur de k_{ini} et de c, la liste des $V^c[y]$ pour les seize premières valeurs de y.

On effectue ensuite une recherche exhaustive sur la clé maître K. Pour tester une valeur de K, on calcule les valeurs de k_{ini}, k_{τ_1}, k_{τ_2} et $\phi(K)$ correspondantes. On cherche alors dans la première table la paire (c', c'') correspondant à la valeur de $\phi(K)$ trouvée. On cherche ensuite dans la deuxième table, selon la valeur de k_{ini} trouvée, les deux séries de valeurs $V^{c'}[y]$ et $V^{c''}[y]$ pour les 16 premières valeurs de y. Connaissant les valeurs de k_{τ_1} et k_{τ_2}, on teste ensuite, à partir des deux listes $V^{c'}[y]$ et $V^{c''}[y]$, par un déchiffrement partiel de Rijndael si on a $\tau^{c'}[y] = \tau^{c''}[y]$ pour la première valeur de y (ici égale à 0). Si c'est le cas, on teste si on a l'égalité pour la deuxième valeur de y et ainsi de suite pour les 16 premières valeurs de y. Si l'égalité est vérifiée pour toutes les valeurs testées, on teste K pour quelques textes clairs.

Nous pensons que les opérations pratiquées à chaque test de clé sont moins coûteuses qu'un chiffrement de Rijndael et donc que la complexité de cet algorithme est légèrement inférieure à celle d'une recherche exhaustive sur une clé de 128 bits.

2.3 Tableau Récapitulatif des Attaques

	Nb tours (lgr de clé)	**Type**	**Nb clairs**	**Complexité**
Attaque des Auteurs de Rijndael	4	Integral	2^9	2^9
	5	Integral	2^{11}	2^{40}
	6	Integral	2^{32}	2^{72}
Attaque de Ferguson et al.	6	Integral	$6 \cdot 2^{32}$	2^{44}
	7	Integral	$2^{128} - 2^{119}$	2^{120}
	8 (256)	Integral	$2^{128} - 2^{119}$	2^{204}
	9 (256)	Related Key	2^{77}	2^{224}
Attaque de S. Lucks	7	Integral	2^{32}	2^{184}
	7	Integral	2^{32}	2^{200}
Attaque Gilbert-Minier	7 (192-256)	Goulot d'étranglement	2^{32}	2^{144}
	7 (128)		2^{32}	$2^{128}(1-\epsilon)$

Dans le cas où la longueur de clé n'est pas précisée, on considère que l'attaque fonctionne pour toutes les longueurs possibles.

2.4 Autres Tentatives d'Attaques

Afin de tenter d'améliorer l'attaque présentée à la troisième conférence AES sur une version de 7 étages de Rijndael, je me suis intéressée aux deux derniers tours en essayant de "remonter" ces étages par des méthodes polynômiales. Malheureusement, aucune de ces méthodes n'a permis d'amélioration significative.

2.4.1 Notation

Soit C, la matrice du chiffré, K la clé du dernier étage, K' la clé de l'avant dernier étage. Soit, si on détaille les notations :

$$C = \begin{array}{|c|c|c|c|}\hline x_0 & x_1 & x_2 & x_3 \\ \hline x_4 & x_5 & x_6 & x_7 \\ \hline x_8 & x_9 & x_{10} & x_{11} \\ \hline x_{12} & x_{13} & x_{14} & x_{15} \\ \hline \end{array}$$

$$K = \begin{array}{|c|c|c|c|}\hline k_0 & k_1 & k_2 & k_3 \\ \hline k_4 & k_5 & k_6 & k_7 \\ \hline k_8 & k_9 & k_{10} & k_{11} \\ \hline k_{12} & k_{13} & k_{14} & k_{15} \\ \hline \end{array}$$

$$K' = \begin{array}{|c|c|c|c|}\hline k'_0 & k'_1 & k'_2 & k'_3 \\ \hline k'_4 & k'_5 & k'_6 & k'_7 \\ \hline k'_8 & k'_9 & k'_{10} & k'_{11} \\ \hline k'_{12} & k'_{13} & k'_{14} & k'_{15} \\ \hline \end{array}$$

Considérons à présent les deux derniers tours d'une version à 7 étages de Rijndael, où l'on remplace la boîte S par la fonction d'inversion, c'est à dire que l'on ne considère plus la transformation affine. Précisons également que le dernier étage étant la transformation finale, il ne contient pas de MixColumn. Pour faciliter la compréhension des équations, nous omettons également l'action des rotations de ShiftRow, qui ne jouent pas ici un rôle essentiel.

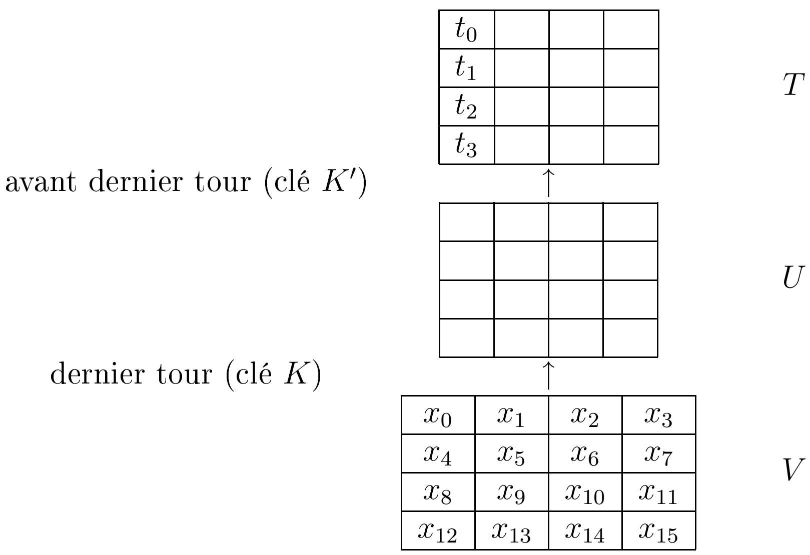


FIG. 2.6 – Dernier et avant dernier étage de Rijndael

Le but est alors d'utiliser les 256 équations linéaires du tour précédent de la forme : pour tout y de [0,...,255], on a $0E \cdot t_0^{c'} + 0B \cdot t_1^{c'} + 0D \cdot t_2^{c'} + 09 \cdot t_3^{c'} = 0E \cdot t_0^{c''} + 0B \cdot t_1^{c''} + 0D \cdot t_2^{c''} + 09 \cdot t_3^{c''}$ en exprimant de façon polynômiale chaque t_i en fonction de C, K et K' pour obtenir un test plus efficace que la recherche exhaustive d'un certain nombre d'octets de clé.

L'expression rationnelle liant t_0 et le chiffré du dernier tour est :

$$\begin{aligned} t_0 \quad = \quad & \left(0E \bullet \left(\frac{1}{x_3 \oplus k_3} \oplus k'_3\right) \oplus 0B \bullet \left(\frac{1}{x_6 \oplus k_6} \oplus k'_6\right) \oplus \right. \\ & \left. 0D \bullet \left(\frac{1}{x_9 \oplus k_9} \oplus k'_9\right) \oplus 09 \bullet \left(\frac{1}{x_{12} \oplus k_{12}} \oplus k'_{12}\right)\right)^{-1} \end{aligned}$$

Chaque t_i s'exprime comme une équation en 8 octets de clés inconnus. Les 4 t_i dépendent donc de 32 inconnues liées aux clés K et K'. Si on tente de linéariser le système de manière classique (c'est à dire que après avoir fait disparaître les fractions rationnelles par multiplication successive des dénominateurs et avoir simplifié l'expression obtenue, on considère que chaque monôme de la forme $k_m \cdots k'_p$ devient une inconnue), on obtient environ 2^{48} monômes soit 2^{48} inconnues. Cette méthode ne permet donc pas à cause du nombre très élevé de monômes et de la complexité de la résolution d'un système linéaire (environ (nb monômes)$^{2.495}$, voir [CW82]) d'améliorer l'algorithme de la section 2.2.1.

En revanche, si la clé du dernier tour est complètement déterminée, le nombre de monômes liés aux inconnues dans le système linéarisé tombe à 5 et à 30 pour chaque t_i si c'est la clé de l'avant dernier tour qui est connue. Malheureusement, la recherche exhaustive des sous-clés coûte trop cher.

Une amélioration sur une version à 7 tours de Rijndael semble donc compromise. je me suis donc intéressée à une version de Rijndael à 6 étages.

2.4.2 Attaque de Rijndael sur 6 Tours

Dans le cas d'une version à 6 étages avec des boîtes S complètes, le dernier tour représente la transformation finale, c'est à dire qu'il ne contient pas de MixColumn. On considère que la partie linéaire de la boîte S fait partie intégrante de la transformation linéaire de Rijndael, c'est à dire que moyennant une transformation sur la clé, on la considère juste comme une transformation bits à bits des octets qu'on étudie. La boîte S se compose alors uniquement de l'inversion dans le corps $GF(256)$. On a alors :

$$\tau = \bigoplus_{i=0}^{3} \alpha_i \bullet t_i = \bigoplus_{i=0}^{3} \frac{\alpha_i}{x_{3(i+1)} \oplus k_{3(i+1)}}$$

en tenant ici compte du ShiftRow du dernier étage. On cherche alors à résoudre l'équation $\forall y \in [0..255]$, $\tau^{c'}[y] = \tau^{c''}[y]$ en la linéarisant. Le nombre de monômes liés aux octets de clé est alors de 60. Il faut donc 60 équations pour la résoudre. D'où l'algorithme suivant :

Chiffrer les $(256)^4$ textes clairs
Pour k_{ini} variant de (0,0,0,0) à (255,255,255,255) faire
    Partitioner les $(256)^4$ textes clairs choisis
    en $(256)^3$ ensemble Λ^c selon la valeur de c.
    Choisir parmi ces $(256)^3$ ensemble Λ^c $(256)^2$ valeurs de c
    Pour chaque paire (c', c'') faire
        Résoudre les 60 équations linéaires pour y allant de 0 à 59
        Vérifier si collision pour les valeurs suivantes de y
            Si oui retourner $(k_{ini}, k_3, k_6, k_9, k_{12})$
            Sinon continuer
        Fin Si
    Fin Pour
Fin Pour

La complexité d'un tel algorithme est : $2^{32} \times 2^{16} \times (60)^{2.495} \times 16 \approx 2^{67}$, ce qui place une telle attaque en dessous de la "Square Attack".

Annexe 1 : Dénombrement des Transitions de MixColumn

Dans une tentative d'attaque qui cherchait à utiliser la structure particulière de MixColumn, transformation choisie pour ses bonnes propriétés de diffusion, nous avons étudié les biais de transition de MixColumn selon le nombre de zéros d'une colonne d'entrée vers le nombre de zéros d'une colonne de sortie. On pouvait s'attendre à ce que ces biais soient relativement éloignés de la probabilité uniforme étant donné la propriété MDS vérifiée par MixColumn. Nous cherchions donc à distinguer MixColumn d'une somme de permutations aléatoires. Les biais détectés ne sont pas suffisants pour être exploités dans une attaque. Il est néanmoins intéressant de les connaître :

Notons a, b, c et d, les entrées sur une colonne et t_0, t_1, t_2 et t_3 les sorties pour une colonne après MixColumn.

nombre de valeurs nulles dans (t_0, t_1, t_2, t_3) \ nombre de valeurs nulles dans (a, b, c, d)	0	1	2	3	4
0	4162506525	65358030	385050	1020	0
1	65422290	899130	4080	0	0
2	321300	67830	1020	0	0
3	510	510	0	0	0
4	0	0	0	0	1

Voila le dénombrement d'une somme de permutations aléatoires :

nombre de 0 dans (t_0, t_1, t_2, t_3) \ nombre de 0 dans (a, b, c, d)	0	1	2	3	4
0	4162570310	65295221	384090	1004	0
1	65295221	1024238	6025	16	0
2	384090	6025	35	0	0
3	1004	16	0	0	0
4	0	0	0	0	1

Annexe 2 : Les Polynômes de la Boîte S

Dans les dernières tentatives d'attaques vues dans ce chapitre, il a été nécessaire de calculer les différentes expressions polynômiales de la boîte S et de sa réciproque.

Le polynôme de la boîte S dont les coefficients exprimés en hexadécimal appartiennent à GF(256) est : $63 + 8fx^{127} + b5x^{191} + x^{223} + f4x^{239} + 25x^{247} + f9x^{251} + 9x^{253} + 5x^{254}$.

Le polynôme de l'inverse de la boîte S est : $52 + f3x^{1} + 7ex^{2} + 1ex^{3} + 90x^{4} + bbx^{5} + 2cx^{6} + 8ax^{7} + 1cx^{8} + 85x^{9} + 6dx^{10} + c0x^{11} + b2x^{12} + 1bx^{13} + 40x^{14} + 23x^{15} + f6x^{16} + 73x^{17} + 29x^{18} + d9x^{19} + 39x^{20} + 21x^{21} + cfx^{22} + 3dx^{23} + 9ax^{24} + 8ax^{25} + 2fx^{26} + cfx^{27} + 7bx^{28} + 4x^{29} + e8x^{30} + c8x^{31} + 85x^{32} + 7bx^{33} + 7cx^{34} + afx^{35} + 86x^{36} + 2fx^{37} + 13x^{38} + 65x^{39} + 75x^{40} + d3x^{41} + 6dx^{42} + d4x^{43} + 89x^{44} + 8ex^{45} + 65x^{46} + 5x^{47} + eax^{48} + 77x^{49} + 50x^{50} + a3x^{51} + c5x^{52} + 1x^{53} + bx^{54} + 46x^{55} + bfx^{56} + a7x^{57} + cx^{58} + c7x^{59} + 8ex^{60} + f2x^{61} + b1x^{62} + cbx^{63} + e5x^{64} + e2x^{65} + 10x^{66} + d1x^{67} + 5x^{68} + b0x^{69} + f5x^{70} + 86x^{71} + e4x^{72} + 3x^{73} + 71x^{74} + a6x^{75} + 56x^{76} + 3x^{77} + 9ex^{78} + 3ex^{79} + 19x^{80} + 18x^{81} + 52x^{82} + 16x^{83} + b9x^{84} + d3x^{85} + 38x^{86} + d9x^{87} + 4x^{88} + e3x^{89} + 72x^{90} + 6bx^{91} + bax^{92} + e8x^{93} + bfx^{94} + 9dx^{95} + 1dx^{96} + 5ax^{97} + 55x^{98} + ffx^{99} + 71x^{100} + e1x^{101} + a8x^{102} + 8ex^{103} + fex^{104} + a2x^{105} + a7x^{106} + 1fx^{107} + dfx^{108} + b0x^{109} + 3x^{110} + cbx^{111} + 8x^{112} + 53x^{113} + 6fx^{114} + b0x^{115} + 7fx^{116} + 87x^{117} + 8bx^{118} + 2x^{119} + b1x^{120} + 92x^{121} + 81x^{122} + 27x^{123} + 40x^{124} + 2ex^{125} + 1ax^{126} + eex^{127} + 10x^{128} + cax^{129} + 82x^{130} + 4fx^{131} + 9x^{132} + aax^{133} + c7x^{134} + 55x^{135} + 24x^{136} + 6cx^{137} + e2x^{138} + 58x^{139} + bcx^{140} + e0x^{141} + 26x^{142} + 37x^{143} + edx^{144} + 8dx^{145} + 2ax^{146} + d5x^{147} + edx^{148} + 45x^{149} + c3x^{150} + ecx^{151} + 1cx^{152} + 3ex^{153} + 2ax^{154} + b3x^{155} + 9ex^{156} + b7x^{157} + 38x^{158} + 82x^{159} + 23x^{160} + 2dx^{161} + 87x^{162} + eax^{163} + dax^{164} + 45x^{165} + 24x^{166} + 3x^{167} + e7x^{168} + 19x^{169} + e3x^{170} + d3x^{171} + 4ex^{172} + ddx^{173} + 11x^{174} + 4ex^{175} + 81x^{176} + 91x^{177} + 91x^{178} + 59x^{179} + a3x^{180} + 80x^{181} + 92x^{182} + 7ex^{183} + dbx^{184} + c4x^{185} + 20x^{186} + ecx^{187} + dbx^{188} + 55x^{189} + 7fx^{190} + a8x^{191} + c1x^{192} + 64x^{193} + abx^{194} + 1bx^{195} + fdx^{196} + 60x^{197} + 5x^{198} + 13x^{199} + 2cx^{200} + a9x^{201} + 76x^{202} + a5x^{203} + 1dx^{204} + 32x^{205} + 8ex^{206} + 1ex^{207} + c0x^{208} + 65x^{209} + cbx^{210} + 8bx^{211} + 93x^{212} + e4x^{213} + aex^{214} + bex^{215} + 5fx^{216} + 2cx^{217} + 3bx^{218} + d2x^{219} + fx^{220} + 9fx^{221} + 42x^{222} + ccx^{223} + 6cx^{224} + 80x^{225} + 68x^{226} + 43x^{227} + 9x^{228} + 23x^{229} + c5x^{230} + 6dx^{231} + 1dx^{232} + 18x^{233} + bdx^{234} + 5ex^{235} + 1bx^{236} + b4x^{237} + 85x^{238} + 49x^{239} + bcx^{240} + dx^{241} + 1fx^{242} + a6x^{243} + 6bx^{244} + d8x^{245} + 22x^{246} + 1x^{247} + 7ax^{248} + c0x^{249} + 55x^{250} + 16x^{251} + b3x^{252} + cfx^{253} + 5x^{254}$.

Il est tout à fait normal que le polynôme de la boîte S ne comporte que très peu de termes. En effet, l'opération d'inversion dans $GF(256)$ transforme x en x^{254}. Quant à l'application affine sur $GF(2)^8$, on peut l'exprimer dans $GF(256)$ comme un polynôme de la forme $\sum_{i=0}^{7} \alpha_i \bullet x^{2^i}$ avec $\alpha_i \in GF(256)$. Donc le passage à travers la boîte S transforme x en $y = \sum_{i=0}^{7} \alpha_i \bullet x^{254 \cdot 2^i \bmod 255}$ avec $\alpha_i \in GF(256)$.

Il est également normal de constater que ce n'est pas le cas de S^{-1}. En effet, si on note f l'opération d'inversion dans $GF(256)$ et g la transformation affine de la boîte S, on a : $S^{-1}(x) = (g \circ f)^{-1}(x) = f^{-1} \circ g^{-1}(x)$. En tant que inverse d'une transformation affine sur $GF(2)^8$, on peut écrire g^{-1}, également transformation affine, comme un polynôme sur $GF(256)$ de la forme $\sum_{i=0}^{7} \beta_i \bullet x^{2^i}$ avec $\beta_i \in GF(256)$. On a donc $S^{-1}(x) = \left(\sum_{i=0}^{7} \beta_i \bullet x^{2^i}\right)^{254}$, tous les exposants étant calculés modulo 256. Le nombre de termes de S^{-1} est donc très important.

Il nous a également paru intéressant de donner les polynômes de la transformation affine et de la transformation affine réciproque :

- Le polynôme de la transformation affine est : $63 + 5x + 9x^2 + f9x^4 + 25x^8 + f4x^{16} + x^{32} + b5x^{64} + 8fx^{128}$.

- Le polynôme de l'application linéaire réciproque est : $5 + 5x + fex^2 + 7fx^4 + 5ax^8 + 78x^{16} + 59x^{32} + dbx^{64} + 6ex^{128}$.

Chapitre 3

Cryptanalyse de SFLASH

Dans ce chapitre, nous aborderons la cryptanalyse menée contre SFLASH [GM02], schéma de signature à clé publique utilisant les principes de la cryptographie multivariable. Dans un premier temps, nous décrirons succinctement les bases de cette famille d'algorithmes et la fonction trappe qu'elle utilise [Pat00] et nous donnerons quelques exemples de cryptosystèmes. Nous nous intéresserons ensuite au schéma C^*, premier cryptosystème utilisant la cryptographie multivariable introduit en 1988 par Matsumoto et Imai [MI88], et à une de ses variantes C^{*--}, base du schéma de signature SFLASH [PCG00], proposé par J. Patarin, N. Courtois et L. Goubin en 2000 lors de la première conférence NESSIE, projet européen de choix de primitives cryptographiques [NES01a]. Nous présenterons enfin l'attaque menée contre SFLASH.

3.1 Généralités

La cryptographie multivariable est une famille de cryptosystèmes relevant de la cryptographie à clé publique qui utilisent des polynômes à plusieurs variables définis sur un petit corps ou un petit anneau fini K. Ces différents schémas ont en commun l'utilisation de un ou plusieurs systèmes d'équations polynômiales en plusieurs variables comme clé publique pour chiffrer un message ou vérifier une signature.

3.1.1 Premier Exemple Simple

Nous présentons ici un premier exemple simple tiré de [Pat00] : soit $x_0, \cdots, x_4$ et $y_0, \cdots, y_4$ dix éléments du corps $GF(2)$. On pose :

$$\begin{aligned}
y_0 &= x_0 \cdot x_1 + x_0 \cdot x_2 + x_0 \cdot x_3 + x_1 \cdot x_2 + x_1 \cdot x_4 + x_2 \cdot x_3 + x_2 \cdot x_4 \\
y_1 &= x_0 \cdot x_2 + x_0 \cdot x_3 + x_0 \cdot x_4 + x_1 \cdot x_3 + x_1 \cdot x_4 + x_2 \cdot x_3 + x_3 \cdot x_4 \\
y_2 &= x_0 \cdot x_1 + x_0 \cdot x_3 + x_1 \cdot x_2 + x_3 \cdot x_4 \\
y_3 &= x_0 \cdot x_2 + x_0 \cdot x_4 + x_1 \cdot x_2 + x_1 \cdot x_3 + x_2 \cdot x_3 \\
y_4 &= x_0 \cdot x_1 + x_0 \cdot x_2 + x_0 \cdot x_4 + x_1 \cdot x_4 + x_3 \cdot x_4
\end{aligned}$$

toutes ces équations étant calculées modulo 2. Si on connaît les valeurs des variables $x_0, \cdots, x_4$ prises dans $GF(2)$, il est très facile de calculer les y_i correspondants. En revanche, si on connaît les y_i, il est difficile de retrouver les x_i même en connaissant toutes les équations. Dans ce cas précis, certes, on peut facilement tester les 32 valeurs possibles des x_i pour retrouver les bonnes valeurs. Mais si on augmente le nombre de variables et/ou qu'on se place sur des corps ou des anneaux beaucoup plus gros, le problème devient difficile. C'est sur ce constat que repose la cryptographie multivariable.

3.1.2 La Difficulté du Problème MQ

On formalise la difficulté d'un tel problème relevant de l'algèbre de la manière suivante :

Définition 11 (le problème MQ) *Soit K un anneau. Soit f une fonction définie comme m polynômes quadratiques (pas forcément homogènes), en n variables éléments de K, chaque polynôme quadratique f_k étant défini par :*

$$f_k(a_0, \cdots, a_{n-1}) = \sum_{i=-1}^{n-1} \sum_{j=i}^{n-1} \lambda_{ijk} a_i a_j \text{ pour } k \text{ variant de } 0 \text{ à } m-1 \text{ et } a_{-1} = 1.$$

Le problème MQ_K est alors le suivant :
Soit $(b_0, \cdots, b_{m-1})$ une sortie. Trouver au moins une solution $a = (a_0, \cdots, a_{n-1})$ telle que $f(a) = b$.

Le problème de décision associé (c'est à dire l'existence d'une solution) est un problème NP-complet même dans le cas $K = GF(2)$.

Notons que l'on sait résoudre ce problème dans deux cas : lorsque le système est fortement sous-déterminé ($n >> m$) [CGMT02]et lorsque le système est fortement sur-déterminé ($n << m$). Dans ce dernier cas, on peut comme l'ont montré Kipnis et Shamir dans [KS99] diminuer la complexité de la résolution de ce problème par des méthodes de re-linéarisation. L'algorithme XL présenté dans [Cou01] qui utilise les mêmes principes pourrait même permettre de résoudre MQ avec une complexité sous-exponentielle dans certains cas particuliers.

L'un des seuls cas avérés où MQ est polynômial lorsque l'on utilise l'algorithme XL (voir [Cou01]) est $m = \epsilon n^2$ avec ϵ petit.

Notons également que le problème MQ n'est pas le seul posé par la cryptographie multivariable : il existe beaucoup d'autres problèmes algébriques jugés difficiles dans ce domaine comme celui des isomorphismes de polynômes (problème IP), le problème Min-Rank, etc.

3.1.3 Intérêt de l'Utilisation de la Cryptographie Multivariable

Les avantages de l'utilisation de la cryptographie multivariable peuvent être multiples : faible complexité, faible taille des clés ou faible longueur des signatures selon les cas et les cryptosystèmes employés, toujours en considérant que la meilleure attaque a une complexité inatteignable. On peut donc considérer la cryptographie multivariable comme une direction de recherche prometteuse pour l'instant.

3.2 Design des Schémas Multivariables

Les schémas à clé publique utilisant la cryptographie multivariable sont pour nombre d'entre eux construits selon les principes suivants : on définit tout d'abord une fonction F selon deux écritures, l'une sur une extension de corps K de degré m, l'autre sur l'espace vectoriel K^m. On noit ensuite cette fonction sous une ou plusieurs couches d'autres transformations comme des fonctions affines ou linéaires afin de cacher les structures algébriques du schéma. La clé publique de ce cryptosystème est alors donnée par la deuxième écriture de la fonction sous forme de m polynômes quadratiques tandis que les couches de transformations qui entourent la fonction F définissent en général la clé secrète. Si Bob envoie un message à Alice chiffré à l'aide de la clé publique d'Alice, seule cette dernière pourra déchiffrer le message à l'aide de sa clé secrète car par définition du problème MQ, il est difficile de trouver une solution au système composé des m équations publiques lorsque l'on ne connaît qu'une sortie de ce dit système.

Ces schémas à clé publique emploient largement des transformations très usitées en cryptographie à clé secrète telles que les boîtes S ou les transformations affines pour "noyer" l'instance du problème MQ.

Avant de donner des exemples de cryptosystèmes, il est nécessaire de définir les ensembles sur lesquels ces schémas sont construits.

3.2.1 Bases Algébriques

- Soit $K = GF(q)$ un corps fini (q puissance d'un nombre premier).
- Soit P un polynôme irréductible de degré n à coefficient dans K.
- Soit L une extension de K de degré n : $L = GF(q^n) \approx GF(q)[X]/(P(X))$.
- Chaque élément a de L peut alors être représenté par le n-uplet $(a_0, \cdots, a_{n-1})$ de $GF(q)^n$ où les a_i sont des éléments de $GF(q)$ définis par la représentation polynômiale de a : $a = \sum_{i=0}^{n-1} a_i X^i \bmod P(X)$. On note φ cette bijection canonique entre $GF(q)^n$ et $L = GF(q^n)$.

On se dote alors d'une fonction F polynômiale de $GF(q^n)$ dans $GF(q^n)$ qui peut s'écrire à la fois comme un polynôme univarié dans L : $b = F(a)$ mais également comme n polynômes en n variables sur K :

$$\begin{cases} b_0 = p_0(a_0, \cdots, a_{n-1}) \\ \cdots \\ b_{n-1} = p_{n-1}(a_0, \cdots, a_{n-1}) \end{cases}$$

où $(b_0, \cdots, b_{n-1})$ et $(a_0, \cdots, a_{n-1})$ représentent la décomposition de b et a sur $GF(q)^n$ par φ^{-1} et chaque p_i va de K^n dans K. Les polynômes $p_0, \cdots, p_{n-1}$ représentent les composantes de la fonction f :

$$f = \varphi^{-1} \circ F \circ \varphi.$$

où $(b_0, \cdots, b_{n-1})$ et $(a_0, \cdots, a_{n-1})$ représentent la décomposition par φ de b et a sur $GF(q)[X]/P(X)$.

On se dote également de deux transformations affines s et t en n variables à coefficients dans K :

$$(a_0, \cdots, a_{n-1}) = (s_0(x_0, \cdots, x_{n-1}), \cdots, s_{n-1}(x_0, \cdots, x_{n-1}))$$
$$(y_0, \cdots, y_{n-1}) = (t_0(b_0, \cdots, b_{n-1}), \cdots, t_{n-1}(b_0, \cdots, b_{n-1})).$$

On peut donc à partir de la donnée de la fonction F et des transformations s et t définir la fonction G de L dans L par :

$$G = t \circ f \circ s$$

ce qui donne une deuxième écriture de G sous la forme de n polynômes quadratiques de $GF(q)^n$ dans $GF(q)$:

$$G : \{y_i = g_i(x_0, \cdots, x_{n-1})\}_{i=0,\cdots,n-1}$$

La clé secrète d'Alice est alors constituée de s et t et éventuellement de F qui peut être secrète ou publique, tandis que sa clé publique est la donnée des n polynômes g_i.

3.2.2 Le Cryptosystème C^*

Le premier cryptosystème utilisant la cryptographie multivariable a été créé en 1988 par Matsumoto et Imai [MI88] et se nomme C^*. Il utilise les bases algébriques du paragraphe précédent :

- Soit $K = GF(q)$ un corps fini de caractéristique 2 (i.e. $q = 2$ ou $q = 2^m$).
- Soit P un polynôme irréductible de degré n à coefficient dans K. Soit $L = GF(q^n)$ une extension de K de degré n dont la représentation est $L \approx GF(q)[X]/(P(X))$. Chaque élément a de L peut alors être écrit comme un n-uplet $(a_0, \cdots, a_{n-1})$ de K^n où les a_i sont des éléments de K définis par la représentation polynômiale de a : $a = \sum_{i=0}^{n-1} a_i X^i \mod P(X)$. On note comme dans le paragraphe précédent φ cette bijection canonique entre K^n et $L = GF(q^n)$.
- La clé publique de C^* est la donnée de n fonctions quadratiques de K^n dans K qui définissent de manière globale une fonction G de K^n dans K^n.

$$\begin{array}{cccc} G : & K^n & \rightarrow & K^n \\ & x = (x_0, .., x_{n-1}) & \mapsto & y = (y_0, .., y_{n-1}) \end{array}$$

  avec

$$y_i = \sum_{0 \leq j < k \leq n-1} \rho_{ijk} x_j x_k + \sum_{0 \leq j \leq n-1} \sigma_{ij} x_j + \tau_i$$

  (en d'autres termes, la clé publique est la donnée des coefficients ρ_{ijk}, σ_{ij} and τ_i, éléments de K).
- La clé privée est quant à elle constituée de deux transformations affines secrètes de K^n, s et t, données par n^2+n coefficients de K. s et t déterminent une représentation secrète de G :

$$G = t \circ \varphi^{-1} \circ F \circ \varphi \circ s$$

  avec

$$\begin{array}{cccc} F : & L & \rightarrow & L \\ & a & \mapsto & b = a^{q^\theta + 1} \end{array}$$

(θ est un entier qui peut être public ou privé et qui est tel que $q^\theta + 1$ et $q^n - 1$ soient premiers entre eux).

Notons que la fonction F est bien quadratique puisqu'elle est le produit de deux automorphismes linéaires de L. De ce fait, $\varphi^{-1} \circ F \circ \varphi$ et G sont des fonctions quadratiques. L'inverse de F, F^{-1} est donné par $a \mapsto a^h$, où h est l'inverse de $q^\theta + 1$ modulo $q^n - 1$.

La connaissance de la clé privée permet le calcul complet de G^{-1}.

3.2.3 Attaque de C^*

Une attaque très efficace contre C^* a été découverte par J. Patarin [Pat95]. Elle est fondée sur la remarque suivante : l'équation $b = a^{q^\theta+1}$, représentation de la fonction F, entraîne $a^{q^{2\theta}} \cdot b = a \cdot b^{q^\theta}$ en multipliant l'équation précédente par $a^{q^{2\theta}}$. Cette nouvelle équation est bilinéaire en a et b à droite comme à gauche. Il existe donc des équations de la forme :

$$\sum_{0\leq j\leq n-1, 0\leq k\leq n-1} \gamma_{jk} x_j y_k + \sum_{0\leq j\leq n-1} \delta_j x_j + \sum_{0\leq j\leq n-1} \epsilon_j y_j + \eta = 0$$

où $(x_0, \cdots, x_{n-1})$ est un vecteur d'entrée de G appartenant à K^n et $(y_0, \cdots, y_{n-1})$ un vecteur de sortie. Le principal avantage de cette réécriture est la disparition des termes de degré deux de la forme $x_i x_j$ et $y_i y_j$. Chaque donnée d'un couple d'entrée/sortie permet de construire une équation linéaire en les coefficients inconnus. On peut alors reconstituer un espace vectoriel V composé d'équations de cette forme à l'aide de suffisamment de couples d'entrée/sortie de la fonction G. La recherche de l'antécédent par G d'un vecteur de sortie y se réduit alors essentiellement à la résolution d'un système linéaire. On peut ainsi déchiffrer des messages.

La complexité de cette attaque est de l'ordre de $m^2 n^4 \log(n)$ opérations.

3.2.4 Autres Propositions sur Les Mêmes Bases

Après cette attaque, d'autres cryptosystèmes utilisant la cryptographie multivariable ont été proposés. On peut citer l'exemple du cryptosystème HFE (pour Hidden Field Equations), introduit en 1996 par J. Patarin [Pat96] qui utilise à la place de la fonction F une fonction légèrement plus complexe - polynomiale et non monomiale - pour se prémunir contre l'attaque précédente. Cette nouvelle fonction n'est pas bijective mais l'introduction dans le clair d'une redondance fournie par une fonction de hachage permet cependant de déchiffrer.

Plusieurs transformations générales permettant de consolider des cryptosystèmes utilisant la cryptographie multivariable ont également été proposées par Jacques Patarin [Pat00] :

- enlever un certain nombre d'équations publiques pour rendre impossible la transformation du système d'équations quadratiques en un système univarié et pour cacher les propriétés algébriques du système.
- ajouter de nouvelles variables appelées "variables de vinaigre" mélangées aux variables du système appelées "variables d'huile" à l'aide de transformations affines

bijectives secrètes. On ne peut inverser le processus que si ces dernières transformations sont connues.

On peut également fixer un certain nombre de variables ou ajouter des équations pour noyer les propriétés algébriques du schéma.

Les méthodes les plus usitées restent cependant les deux premières ce qui a permis de créer de nouveaux cryptosystèmes tel HFEv- ou HFE- (le "v" représentant l'ajout de nouvelles "variables de vinaigre" , et le "-" la suppression d'équations publiques).

En 1998, J. Patarin, L. Goubin et N. Courtois dans [PGC98] présentaient une attaque contre C^{*-}, variante de C^*, la clé publique de C^{*-} étant alors constituée des $n-r$ premières équations quadratiques à coefficients dans K, déduites de la fonction publique G de l'instance complète de C^*. Ils prouvent dans cet article que C^{*-} est attaquable pour des petites valeurs de r, plus exactement les valeurs telles que $q^r \leq 2^{64}$. Cette dernière remarque amena à la construction d'un nouveau cryptosystème nommé C^{*--} utilisable seulement en signature et dont les paramètres q et r étaient choisis tels que $q^r \geq 2^{64}$.

3.2.5 D'autres Schémas Proposés

J. Patarin a également proposé d'autres types de schémas utilisant les mêmes propriétés [Pat00]. Citons par exemple les schémas "sur deux Rounds -" où toujours à partir d'un corps $GF(q)$ un message clair $x = (x_0, \cdots, x_{n-1})$ de $GF(q)^n$ est transformé en un message chiffré $y' = (y_0, \cdots, y_{m-1})$ appartenant à $GF(q)^m$ $(m \leq n)$ représentant les m premières composantes du vecteur $y = (y_0, \cdots, y_{n-1})$ de $GF(q)^n$. Pour chiffrer le message x, on calcule $y = t \circ \psi \circ s \circ \phi \circ r(x)$ où r, s et t sont trois bijections affines de $GF(q)^n$ dans lui-même et ϕ et ψ sont données chacune par n équations quadratiques sur $GF(q)$. On demande également que ϕ et ψ soient inversibles mais pas bijectives, c'est à dire que pour ces fonctions, on soit capable de trouver les antécédents des éléments qui nous intéressent. Les cinq composantes précédemment décrites restent secrètes. On ne publie en fait que les m premières équations globales de $t \circ \psi \circ s \circ \phi \circ r$.

Citons ici une attaque de E. Biham décrite dans [Bih00] contre ce type de schéma qui utilise le paradoxe des anniversaires et le fait que les fonctions ϕ et ψ ne soient pas des bijections afin de chercher des collisions internes. D. Ye, K. Lam et Z. Dai présentent également dans [YLD99], une cryptanalyse de ces schémas utilisant des méthodes de décomposition de la composée de deux fonctions quadratiques.

3.3 FLASH et SFLASH

Nous venons de voir quelques exemples de cryptosystèmes de la cryptographie multivariable. Nous allons à présent nous intéresser à deux schémas de signature nommés FLASH et SFLASH, tous deux instances de C^{*--}, soumis en 2000 à NESSIE [NES01a], projet européen de choix de primitives cryptographiques.

Nous décrirons ensuite l'attaque menée contre SFLASH présentée à Eurocrypt'2002 [GM02] qui utilise une faiblesse particulière de cet algorithme.

3.3.1 Description de SFLASH

SFLASH [PCG00] est une instance particulière de C^{*--}, proposée comme schéma de signature lors de la première conférence NESSIE et qui a passé avec succès la première phase de sélection [NES01a] contrairement à FLASH, autre schéma de signature fondé sur les mêmes bases théoriques et également instance particulière de C^{*--}. Il semble que les clés plus longues utilisées par FLASH pour un niveau de sécurité équivalent l'aient fait juger moins attractif que SFLASH.

La faible longueur des clés de SFLASH est liée à un choix particulier des fonctions s et t et des coefficients des équations quadratiques publiques. Les paramètres choisis pour SFLASH sont les suivants :

- Le corps de départ K est égal à $GF(2^7)$, on a donc $q^m = 2^7$. On note $K' = GF(2) = \{0, 1\}$, le sous-corps de K à deux éléments. Les éléments de K sont représentés par des 7-uplets d'éléments de K' grâce à la représentation polynômiale de K, $K \approx K'[X]/(X^7+X+1)$ où X^7+X+1 est un polynôme primitif à coefficient dans K'.
- L est une extension du corps K définie par $L \approx K[X]/(P(X))$ où $P(X)$ est le polynôme irréductible de $K[X]$ égal à $X^{37} + X^{12} + X^{10} + X^2 + 1$ à coefficients dans K'. On a $n = 37$, les éléments de L peuvent donc être représentés comme des 37-uplets d'éléments de K.
- La fonction F de L dans lui-même utilisée dans la représentation secrète de G est égale à $a \mapsto a^{128^{11}+1}$; θ est donc public et égal à 11.
- Le nombre r d'équations cachées est égal à 11. La condition de C^{*--} est donc bien vérifiée car $q^r = 2^{77} > 2^{64}$.
- Les deux transformations secrètes s et t de $K^n = K^{37}$ sont pour chacune la donnée d'une matrice $n \times n$ à coefficients dans K' et d'un vecteur colonne $n \times 1$ également à coefficients dans K'.

Sous toutes ces conditions, et à cause du choix de s, t et F, il est facile de voir que tous les coefficients des $n - r = 26$ premières équations publiques, déduites des $n - r$ premières coordonnées de la fonction G, appartiennent au sous-corps $K' = GF(2)$. C'est cette propriété qui rend la clé publique de SFLASH aussi compacte ; la taille de la clé publique est divisée par un facteur $m = 7$.

Précisons également que en plus de la donnée de s et t, la clé secrète de SFLASH contient une valeur secrète Δ de 80 bits, servant de germe pour la génération pseudo-aléatoire d'un des paramètres d'entrée de l'algorithme de signature.

Lorsqu'Alice veut signer un message M avec SFLASH (voir figure 3.1), elle utilise sa clé secrète (s, t, Δ) comme suit :

- Elle génère d'abord deux mots de 160 bits $M1 = SHA - 1(M)$ et $M2 = SHA - 1(M1)$, un mot V de 182 bits tel que $V = M1_{0\to 159}||M2_{0\to 21}$ et le mot $W = SHA - 1(V||\Delta)_{0\to 76}$ de 77 bits.
- V est divisé en $n-r = 26$ mots $y_0, \cdots, y_{25}$ de 7 bits chacun, représentant 26 éléments du corps K, et W est divisé en $r = 11$ autres mots $y_{26}, \cdots, y_{36}$ également de 7 bits, chacun représentant un élément de K. On note y le 26-uplet $(y_0, \cdots, y_{25})$ et y^* le 37-uplet $(y_0, \cdots, y_{25}, y_{26}, \cdots, y_{36})$.
- La fonction secrète $s^{-1} \circ \varphi^{-1} \circ F^{-1} \circ \varphi \circ t^{-1}$ est appliquée à y^*. Le 37-uplet x d'éléments

de K obtenu représente la signature de M. Afin de vérifier la signature x d'Alice, Bob n'a qu'à calculer $G(x)$ à l'aide des 26 équations quadratiques publiques déduites de G et à vérifier si la valeur ainsi obtenue est bien égale à y.

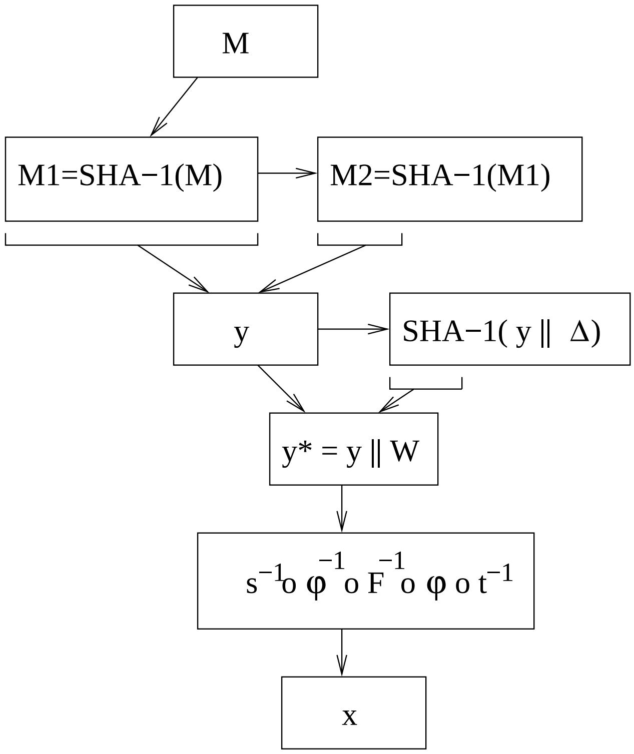


FIG. 3.1 – Le Schéma de Signature SFLASH

3.3.2 Cryptanalyse de SFLASH

Je vais décrire ici l'attaque contre SFLASH, présentée à Eurocrypt'2002 [GM02]. Cette attaque sur SFLASH utilise une propriété particulière de l'algorithme due au choix de s et t. La décision de réduire la taille des signatures a créé une faille de sécurité. Cette attaque n'est cependant pas applicable à FLASH, ni à la nouvelle version de SFLASH proposée en Octobre 2001 au comité NESSIE.

Premières Remarques

L'attaque contre SFLASH est liée à une première remarque très simple : l'espace vectoriel $K'^{37} = GF(2)^{37}$ reste invariant après passage dans G. En effet, puisque s, t et $\varphi^{-1} \circ F \circ \varphi$ sont construites pour laisser K'^n invariant, leur composé G et son inverse $s^{-1} \circ \varphi^{-1} \circ F^{-1} \circ \varphi \circ t^{-1}$ laissent également cet espace vectoriel invariant.

Dans la suite, on notera G' la fonction de K^n dans K^n définie par les $n-r$ premières équations quadratiques de G suivies de r équations quadratiques additionnelles définies également sur K^n. Ainsi, G' laisse également invariants les éléments de K'^n. On note g' et g les restrictions de G' et G au sous espace vectoriel invariant K'^n. g peut être vu comme une version miniature de SFLASH agissant sur l'espace vectoriel K'^n. Dans ce cas, l'exposant de F induit dans g et g' devient $q'^{\theta'}+1 = 2^3+1$.

L'idée directrice de l'attaque est d'utiliser les propriétés d'invariance sur l'espace vectoriel K'^n pour générer une instance complète de C^* et ensuite d'appliquer l'attaque de J. Patarin dédiée à cet algorithme (dans le cas de la restriction à K'^n, on peut remarquer que la condition nécessaire à la solidité de C^{*--} n'est plus vérifiée car on a $q'^r \leq 2^{64}$).

Afin d'obtenir une instance complète de C^*, on cherche à générer r équations additionnelles de la forme :

$$z_i(x) = \sum_{0\leq j<k\leq n-1} \alpha_{ijk}x_jx_k + \sum_{0\leq j\leq n-1} \beta_{ij}x_j$$

(où les α_{ijk} et β_{ij} appartiennent à K') qui avec les $n-r$ équations quadratiques publiques

$$y_i(x) = \sum_{0\leq j<k\leq n-1} \rho_{ijk}x_jx_k + \sum_{0\leq j\leq n-1} \sigma_{ij}x_j + \tau_i$$

représentent complètement une fonction G' et donc une instance de C^*. Plus formellement, on souhaite trouver r équations quadratiques additionnelles telles qu'il existe une transformation affine t' de K^n dans lui-même à coefficients dans K' telle que :

$$\forall x = (x_0, x_1, .., x_{n-1}) \in K^n,$$
$$(y_0(x), .., y_{n-r-1}(x), z_0(x), .., z_{r-1}(x)) = t' \circ \varphi^{-1} \circ F \circ \varphi \circ s(x) \ (1)$$

Il est facile de voir que ces équations supplémentaires sont des combinaisons linéaires des vraies équations cachées et des $n-r$ équations publiques si on omet l'addition avec la constante.

On applique alors l'attaque contre C^* aux n équations trouvées $(y_0(x), \cdots, y_{n-r-1}(x), z_0(x), \cdots, z_{r-1}(x))$ pour générer des signatures valides du message M. Ceci est possible car Δ, l'autre partie de la clé secrète, n'a pas d'influence sur y, et n'intervient pas dans les équations publiques de vérification.

Il est nécessaire de choisir des z_i linéairement indépendants des y_i connus afin d'obtenir un espace E de dimension 37. On cherche donc une base de $E =< y_0(x), \cdots, y_{n-r-1}(x), y_{n-r}(x), \cdots, y_{n-1}(x) >$, espace de dimension 37 à coefficients dans K' généré par les 37 équations quadratiques publiques et cachées sans leurs constantes, sachant que les 26 premiers vecteurs sont connus.

Première Tentative d'Attaque

Le but est donc de trouver les 703 coefficients appartenant à $GF(2)$ de onze fonctions quadratiques z_i pour compléter une fonction G'.

Pour cela, en travaillant sur l'espace K'^n, on peut "inverser" la fonction publique g : pour les 2^{37} $x = (x_0, \cdots, x_{36})$ appartenant à K'^{37}, on calcule chaque image $y- = (y_0, \cdots, y_{25})$ correspondante par les 26 équations quadratiques publiques. On partitionne toutes ces valeurs en classes selon la valeur de $y-$. Chaque classe, notée $classe(y-)$, contient 2^{11} valeurs x. Cette première étape nécessite environ 2^{37} calculs de vérification de signature et une place en mémoire égale au nombre de classes que l'on souhaite conserver.

On utilise alors le caractère équilibré de la fonction g' : pour toute équation quadratique supplémentaire $z(x)$ composant une base de l'espace E de dimension 37, pour chaque valeur de $y-$, $classe(y-)$ contient 2^{10} valeurs telles que $z(x) = 0$ et 2^{10} valeurs telles que $z(x) = 1$, on a donc :

$$\begin{aligned} \sum_{x \in class(y-)} z(x) &\equiv 0 \bmod 2 \\ \text{i.e.} \sum_{x \in class(y-)} \alpha_{jk} x_j x_k + \beta_j x_j &\equiv 0 \bmod 2 \end{aligned} \tag{3.1}$$

Ainsi, chaque classe déduite d'une valeur particulière de $y-$, produit une équation linéaire sur $GF(2)$ dont les 703 inconnues sont les coefficients de la fonction quadratique z.

En pratique, pour une classe donnée, on calcule sur $GF(2)$ pour chaque α_{jk}, $\sum_{x \in class(y-)} x_j x_k$ et pour chaque β_j , $\sum_{x \in class(y-)} x_j$ et cela pour un certain nombre N de classes. On souhaite générer un nombre d'équations supérieur au nombre d'inconnues, on choisit donc $N = 1000$ afin d'obtenir une matrice de taille 1000×703, représentant un système de 1000 équations linéaires dans $GF(2)$. Le noyau de la matrice ainsi obtenue est composé des solutions de l'équation (3.1) et donc en partie au moins des équations publiques et de leurs combinaisons linéaires. On s'attend donc à obtenir l'espace E de dimension 37, donné par les 26 équations publiques et les 11 équations cachées. Dans les simulations pratiquées, l'espace de solutions obtenu E' est quatre fois plus grand que E, sa dimension est 148. Nous avons également testé d'autres valeurs de r pour lesquelles le même phénomène est observé. Il existe donc des solutions parasites qui viennent "noyer" l'espace solution.

Rappel sur la Cryptanalyse Différentielle d'Ordre Supérieur

Soit f une fonction vectorielle booléenne de $\{0,1\}^n$ dans $\{0,1\}^n$ à n composantes f_i, soit V un sous-espace affine de $\{0,1\}^n$ de dimension d, on définit alors la dérivée de f_i à l'ordre d sur V par :

$$D_V(f_i) = \bigoplus_{v \in V} f_i(v).$$

On a alors la propriété suivante :

Propriété 1 *Si $deg(f_i) < d$ alors $D_V(f_i) = 0$ où $deg(f_i)$ représente le degré algébrique de la fonction f_i.*

Dans le cas qui nous intéresse, c'est à dire dans le cas de fonction vectorielle booléenne de $\{0,1\}^n$ dans lui-même à n composantes f_i définie par $f = \phi_1 \circ \varphi^{-1} \circ F \circ \varphi \circ \phi_2$ où la fonction F est une fonction puissance sur le corps $GF(2^n)$ i.e. $F(x) = x^\alpha$ où α est un entier naturel premier avec $2^n - 1$, le degré algébrique de f et donc des f_i est égal au poids de Hamming de α que l'on note $W(\alpha)$ (cette propriété provient de l'écriture de f sous sa forme algébrique normale et de l'écriture de α sous la forme d'une somme de puissance de 2).

Explication du Phénomène

Dans tout ce paragraphe, on notera $y-$ la donnée des 26 premières coordonnées dans K' de y^* données par les équations publiques : $y- = (y_0, \cdots, y_{25})$ et $\mathbf{y}$ la donnée d'un 37-uplet composé de $y-$ et de 11 autres éléments de K'. Précisons également que l'étude menée dans ce paragraphe porte uniquement sur l'espace vectoriel K'^n.

On peut expliquer la présence de solutions parasites de la manière suivante : lors de la première tentative d'attaque, on somme des fonctions booléennes sur un ensemble de 2^{11} valeurs ce qui revient à calculer une dérivée à l'ordre 11 en $\mathbf{y}$. Pour un $\mathbf{y}$ donné, on note V_{11} le sous espace affine de $GF(2^{37})$ défini par $(y_0, \cdots, y_{25}) \times GF(2^{11})$. On a donc cherché des éléments tel que :

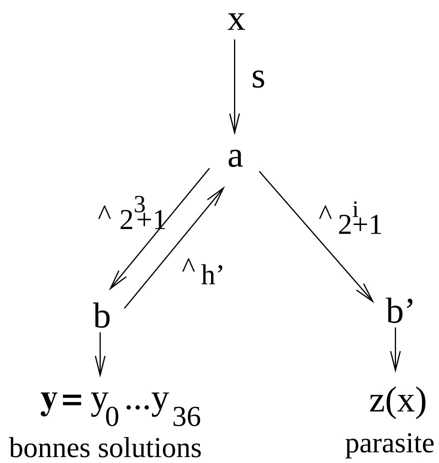
$$\sum_{x \in classe(y-)} z(x) = \sum_{\mathbf{y} \in V_{11}} z(g^{-1}(\mathbf{y})) =_{def} \sum_{\mathbf{y} \in V_{11}} z(\mathbf{y}) = 0$$

Les solutions parasites vérifient cette équation. Il est donc naturel de supposer que leur degré en $\mathbf{y}$ est inférieur à 11.

Si $z(x)$ est une "vraie" fonction quadratique solution du système, elle s'écrit comme une combinaison linéaire des composantes de $\mathbf{y}$ (où l'on omet les constantes) et le degré en $\mathbf{y}$ de $z(\mathbf{y})$ vaut 1.

En ce qui concerne les solutions parasites, on est conduit à rechercher des fonctions booléennes $z(x)$ qui soient quadratiques en x et dont le degré en $\mathbf{y}$ soit inférieur à 11. Ces propriétés suggèrent de rechercher des solutions parasites parmi les fonctions quadratiques de la forme $t' \circ \varphi^{-1} \circ f_{q'^i+1} \circ \varphi \circ s(x)$ où cette fois ci $f_{q'^i+1}$ est un exposant quadratique "quelconque" distinct de $f_{q'^{\theta'}+1}$. Pour une telle fonction, on a $z(\mathbf{y}) = t' \circ \varphi^{-1} \circ f_{q'^{\theta'}+1} \circ f_{h'} \circ \varphi \circ t^{-1}(\mathbf{y})$ et la condition pour que le degré de $z(\mathbf{y})$ en $\mathbf{y}$ soit inférieur à 11 est donc $W((q'^i + 1) \cdot h' \bmod 2^{37} - 1) < 11$.

Les solutions parasites sont donc des fonctions $z(\mathbf{y})$ telles que $W(h_i) < 11$ avec $h_i = h' \cdot (2^i + 1) \bmod 2^{37} - 1$.



Il ne reste plus alors qu'à calculer les différentes valeurs du poids de h_i pour i variant de 0 à 36. On obtient ainsi 6 exposants de poids respectifs 1, 4, 7, 10, 4 et 1 pour i valant 3, 9, 15, 21, 22, 28 et 34. Il est cependant nécessaire de noter que $(2^{28} + 1) =$

$2^{28} \cdot (2^9 + 1) \bmod 2^{37} - 1$, donc que h_9 et h_{28} sont égaux à une rotation circulaire près et qu'ils définissent le même ensemble de fonctions quadratiques car dans ce cas, les deux fonctions $t' \circ \varphi^{-1} \circ f_{q'^i+1} \circ \varphi \circ s$ sont égales à une transformation linéaire près. Cette remarque est également valable pour h_1 et h_{34} car $h_{34} = 2^{34} \cdot h_1 \bmod 2^{37} - 1$.

On obtient donc à une transformation linéaire près 4 exposants de la forme $2^i + 1$ conduisant à 4 fonctions quadratiques $z(\mathbf{y})$ de degré strictement inférieur à 11 (les poids respectifs étant 1, 4, 7 et 10) pour des valeurs de i égales à 3, 9, 15 et 21. Il y a donc bien 4 solutions (dont trois solutions parasites) qui chacune génère un espace de dimension 37. E' est donc composé de la somme directe de ces 4 espaces de dimension 37.

Résolution

On cherche à présent une méthode qui permette de détecter ces solutions parasites pour se ramener à l'espace E de dimension 37 attendu. Pour cela, on choisit une base $B = (e_0(x), \cdots, e_{147}(x))$ de E'. Les solutions $l_i(x)$ (vraies et fausses) du système précédent sont des combinaisons linéaires des éléments de cette base : $l_i(x) = \sum_{0\le j\le 147} \gamma_j e_j$. Si $l_i(x)$ est une vraie solution, son degré en $\mathbf{y}$ est 1, en revanche si $l_i(x)$ est une solution parasite son degré en $\mathbf{y}$ est compris entre 2 et 10 (ce qui est vrai aussi pour les éléments e_k de la base). En formant des produits de la forme $l_i(x) \cdot e_k$ pour un certain nombre de e_k, le degré maximal en $\mathbf{y}$ de ce produit est au plus 11 dans le cas où $l_i(x)$ est une vraie solution et au moins 12 dans le cas où $l_i(x)$ est une solution parasite. Ainsi, si on calcule des dérivées douzièmes de tels produits, les bonnes solutions seront celles pour lesquelles toutes ces dérivées douzièmes sont nulles. Cela nous donne donc plusieurs équations supplémentaires en les 148 inconnues γ_i. En pratique, pour que ce test soit concluant, il suffit de prendre les produits avec deux éléments de la base. Ceci s'est avéré vrai dans les simulations.

On note $V_{12}(\mathbf{y})$ l'ensemble des éléments $\mathbf{y}$ qui sont fixes sur leurs 25 premières coordonnées, c'est à dire l'ensemble des éléments $(y_0, \cdots, y_{24}) \times \mathrm{GF}(2)^{12}$ et $V_{12}(x)$ l'ensemble des images des éléments de $V_{12}(\mathbf{y})$ par la fonction g. On a :

$$V_{12}(x) = \text{ classe}(y_0, \cdots, y_{24}, 0) \cup \text{ classe}(y_0, \cdots, y_{24}, 1)$$

Pour k valant 0 et 1, les bonnes solutions vérifient les équations de la forme :

$$\sum_{0\le j\le 147} \gamma_j \left(\sum_{x\in V_{12}(x)} e_j(x) \cdot e_k(x) \right) \equiv 0 \bmod 2$$

On calcule ces deux équations pour un nombre N' de classes $V_{12}(\mathbf{y})$ au moins égal à 148 ($N' = 200$ devrait suffire) afin d'obtenir un système de taille $2N' = 400$ en 148 inconnues. Le noyau de ce système est bien E de taille 37, comme le confirme l'expérience.

Il ne reste alors plus qu'à générer une fonction G' en formant une base à l'aide de l'espace E et des 26 équations publiques pour obtenir une instance complète de C^*. On peut alors, grâce à l'attaque [Pat95], inverser la fonction G' et produire des signatures valides.

Complexité de l'Attaque

Cette attaque nécessite en tout premier lieu la recherche exhaustive de toutes les images des éléments x de K'^n par les équations publiques, soit le calcul de 2^{37} vérifications de signature à partir de la fonction publique g. On garde en mémoire tous ces éléments, en stockant les 2^{11} éléments des $N + 2N' = 1400$ classes dont on a besoin dans une table à 1400 entrées où chaque entrée est pointée par la valeur d'un $y-$, ce calcul étant moins cher qu'une vérification de signature SFLASH. La complexité totale de cette étape est donc inférieure à 2^{38} calculs de SFLASH et la taille en mémoire nécessaire est donc égale à une table à 1400 entrées composées pour chaque entrée de 2^{11} éléments.

La première étape de l'attaque est essentiellement la génération des $N = 1000$ équations en 703 variables et la résolution du système de taille $N \times 703$ par élimination gaussienne. La complexité de cette étape est donc majorée par $N \cdot 703 \cdot 2^{11} + \frac{N^3}{3} \leq 2^{31}$. La seconde étape qui nécessite la création de $2 \cdot N' = 400$ équations linéaires a une complexité du même type majorée par 2^{27}. La complexité totale de ces deux premières phases est donc inférieure à 2^{32} calculs de la fonction publique de SFLASH. La complexité de l'attaque de J. Patarin contre C^* est de l'ordre de 2^{27} calculs.

Pour forger une fausse signature d'un message M, on applique à celui-ci deux fois la fonction SHA-1 afin d'obtenir la valeur du $y-$ que l'on complète par 11 autres valeurs arbitrairement choisies. On signe alors ce message à l'aide de la fausse instance de C^* générée précédemment. La personne qui souhaite vérifier la signature à l'aide des équations publiques obtient alors bien la valeur de $y-$, la signature est donc jugée valide.

La complexité totale de cette attaque est inférieure à 2^{39} calculs de la fonction publique de SFLASH.

3.4 Conclusion

Nous venons de voir, après une rapide introduction à la cryptographie multivariable, une attaque dédiée contre le schéma de signature SFLASH liée à une faiblesse particulière de cet algorithme. Cette attaque n'est efficace ni contre FLASH à cause du choix différent des coefficients des transformations affines s et t ni contre la nouvelle version de SFLASH proposée en Octobre 2001.

Troisième partie

Preuves de Sécurité

Introduction

Dans cette partie, nous aborderons le deuxième grand thème de cette thèse à savoir les preuves de sécurité des algorithmes de chiffrement par blocs dans le modèle de Luby-Rackoff [LR88].

Dans l'étude de la sécurité des algorithmes de la seconde partie de cette thèse, nous avons rencontré des attaques par distingueur qui comparent un système de chiffrement C à un système de chiffrement idéal C^* à l'aide d'une procédure de test (le distingueur) fournissant, pour un nombre q de requêtes au chiffrement C ou au chiffrement C^*, une réponse positive avec des probabilités différentes respectivement notées p et p^*. Si on obtient un avantage ($|p - p^*|$) suffisamment grand, cela signifie que l'on peut différencier C et C^* et donc "attaquer" l'algorithme C. Ce type d'études permet d'établir des bornes inférieures sur la probabilité de distinguer, avec un nombre de clairs et une complexité donnée, un algorithme cryptographique d'un chiffrement idéal.

Luby et Rackoff dans [LR88] ont introduit une notion légèrement différente permettant de calculer des bornes supérieures sur la probabilité de distinguer, avec un nombre de clairs donné et des ressources de calcul illimitées, une construction cryptographique d'un chiffrement idéal : au lieu de distinguer des systèmes de chiffrement, ils ont utilisé la notion de distingueur sur des fonctions ou plus précisément ils ont comparé une fonction idéale à un schéma composé de fonctions idéales afin d'évaluer la qualité des schémas utilisés en chiffrement. Le principal exemple qu'ils ont développé est l'étude théorique d'un schéma de Feistel à trois étages. Ici, une fonction cryptographique dépendant de la clé peut être vue comme une fonction aléatoire associée avec une valeur de clé choisie aléatoirement. Ils prouvent la sécurité de ce schéma en comparant les distributions de probabilités entre la fonction de chiffrement engendrée par ce schéma et une fonction idéale. Ils démontrent ainsi une borne supérieure de sécurité sur l'avantage du distingueur obtenu. Après la publication de cet article, beaucoup d'autres études utilisant les principes de cette preuve de sécurité ont été menées. On peut citer ici la thèse de J. Patarin entièrement consacrée à ce sujet [Pat91] qui replace la sécurité dans le modèle de Luby-Rackoff dans le cadre de la sécurité inconditionnelle définie par Shannon, les travaux de U. Maurer [Mau92] plus abordables que l'article de Luby et Rackoff,... D'autres auteurs se sont également intéressés à d'autres schémas dérivés de celui de Feistel en prouvant leur sécurité dans le modèle de Luby-Rackoff, notamment M. Sugita [Sug96], [Sug97] ou à des modes opératoires d'algorithmes par blocs notamment M. Bellare, J. Kilian et P. Rogaway [BKR94]...

S. Vaudenay a alors tenté d'unifier les différents modèles de sécurité à l'intérieur d'une théorie dite de la décorrélation [Vau98a], [Vau98b], [Vau99a] à partir d'une tentative de généralisation de la définition de fonction universelle de [CW81], [CW79].

Dans cette partie, nous rappellerons tout d'abord quelques notions de base des preuves de sécurité dans le modèle de Luby-Rackoff, notamment le lien établi par J. Patarin entre les probabilités de transition d'une fonction aléatoire et l'avantage d'un distingueur appliqué à cette même fonction. Nous présenterons ensuite la théorie de la décorrélation développée par S. Vaudenay avant d'aborder les preuves concernant les schémas inspirés de Feistel et les nouveaux résultats obtenus pendant cette thèse [GM01].

Même si le modèle de sécurité étudié ici n'est pas réaliste, il donne un critère de sécurité pour les schémas étudiés.

Chapitre 1

Bases des Preuves de Sécurité

Nous nous intéresserons dans ce chapitre à toutes les notions indispensables à la compréhension des preuves de sécurité dans le modèle de Luby et Rackoff pour un algorithme de chiffrement par blocs considéré comme une fonction aléatoire. Après avoir procédé à quelques rappels élémentaires sur les probabilités d'une fonction aléatoire et formalisé la notion de distingueur, notamment les probabilités de transition entre des q-uplets d'entrée et de sortie de cette fonction, nous décrirons le lien établi par J. Patarin entre ces deux notions afin de simplifier la démonstration de Luby et Rackoff concernant la preuve de sécurité par distingueur d'un schéma de Feistel à trois étages considéré comme une fonction aléatoire.

1.1 Rappels Élémentaires sur les Fonctions Aléatoires

Si on considère un algorithme de chiffrement comme une fonction aléatoire, il est nécessaire de faire quelques rappels élémentaires sur les propriétés de ces fonctions aléatoires.

On note $F_{n,m}$ l'ensemble des fonctions de I_n dans I_m ; on a $\#F_{n,m} = 2^{n.2^m}$. On note P_n l'ensemble des permutations de I_n ; on a $\#P_n = 2^n!$.

Définition 12 *On définit une fonction aléatoire f^* comme parfaitement aléatoire si elle est prise aléatoirement dans $F_{n,m}$ selon la loi uniforme, c'est à dire si elle est associée à une distribution de probabilité uniforme sur $F_{n,m}$. On définit de même une permutation c^* parfaitement aléatoire sur I_n comme un élément de P_n distribué uniformément sur celui-ci.*

Si on considère une fonction cryptographique f_k liée à une clé aléatoire k de l'ensemble K, on peut voir la fonction $f = (f_k)_{k \in K}$ comme une fonction aléatoire de $F_{n,m}$.

L'étude de la sécurité des primitives cryptographiques repose essentiellement sur la recherche de collisions internes pour un q-uplet d'entrées ou de sorties.

Définition 13 (Probabilité de transitions multiples liée à f) *Soit f une fonction aléatoire de $F_{n,m}$, on définit $Pr[x \overset{f}{\mapsto} y]$ comme la probabilité de transition associée à un q-uplet $x = (x_1, \cdots, x_q)$ d'entrées de I_n et à un q-uplet $y = (y_1, \cdots, y_q)$ de sorties de I_m :*

$$Pr[x \overset{f}{\mapsto} y] = Pr[f(x_1) = y_1 \wedge f(x_2) = y_2 \wedge ... \wedge f(x_q) = y_q].$$

Précisons quelques propriétés élémentaires que nous utiliserons par la suite :

Propriété 2 *Soit f^* une fonction parfaitement aléatoire de $F_{n,m}$, x un q-uplet d'entrées distinctes de I_n et y un q-uplet de sorties (distinctes ou non) de I_m, on a alors : $Pr[x \overset{f^*}{\mapsto} y] = \frac{1}{|I_m|^q} = 2^{-m.q}$.*

Propriété 3 *Soit c^* une permutation parfaitement aléatoire de P_n, x un q-uplet d'entrées distinctes de I_n et y un q-uplet de sorties distinctes de I_n, on a alors : $Pr[x \overset{c^*}{\mapsto} y] = (|I_n| - q)!/|I_n|! = \frac{(2^n - q)!}{(2^n)!}$.*

Beaucoup de fonctions d'étage étant construites à l'aide de x-or, on utilise souvent la propriété suivante :

Propriété 4 *Soit c^* une permutation parfaitement aléatoire de P_n, x et x' deux éléments distincts de I_n et δ une valeur donnée de I_n, on a alors : $Pr[c^*(x) \oplus c^*(x') = \delta] \leq \frac{2}{2^n}$.*

Preuve : Si $\delta = 0$, $\Pr[c^*(x) \oplus c^*(x') = \delta] = 0$ puisque $x \neq x'$. Si $\delta \neq 0$, $\Pr[c^*(x) \oplus c^*(x') = \delta] = \frac{2^n \cdot (2^n - 2) \cdots 1}{2^n!} = \frac{1}{2^n - 1} \leq \frac{2}{2^n}$. Donc, $\Pr[c^*(x) \oplus c^*(x') = \delta] \leq \frac{2}{2^n}$.

1.2 Formalisation de la Notion de Distingueur

La deuxième notion essentielle à développer dans ce chapitre est celle de distingueur. C'est une machine de Turing probabiliste qui cherche à distinguer par un jeu de questions/réponses une fonction de chiffrement aléatoire f d'une fonction de chiffrement idéale f^*. De façon plus formelle, on a la définition suivante inspirée comme les suivantes de [Vau99] :

Définition 14 *Soit deux ensembles M_1 et M_2 et une fonction aléatoire f de M_1 vers M_2. Un distingueur pour la fonction f est une machine de Turing probabiliste $\mathcal{A}$ disposant d'un oracle $\mathcal{O}$ qui pour toute requête de M_1 envoie une réponse dans M_2. Après un certain nombre de calculs de l'oracle, le distingueur fournit la réponse "0" ou "1". On caractérise un distingueur par le nombre de requêtes q de l'oracle et le temps de calcul T. On mesure alors l'avantage de $\mathcal{A}$ à distinguer f de f^*, fonction de chiffrement aléatoire parfaite (i.e. tirée au sort selon la loi uniforme dans l'ensemble des fonctions de M_1 dans M_2), en calculant $Adv_{\mathcal{A}}(f, f^*) = |p - p^*|$ où p (respectivement p^*) représente la probabilité que $\mathcal{A}$ réponde 1 lorsque la fonction $\mathcal{O} = f$ est implémentée (respectivement $\mathcal{O} = f^*$).*

On peut également utiliser les distingueurs dans le cas de permutations aléatoires, on prend alors en compte les probabilités liées à ces permutations. Il existe deux types de distingueurs : les distingueurs dits "non-adaptatifs" qui préparent à l'avance toutes les requêtes et les distingueurs dits "adaptatifs" qui adaptent les requêtes en fonction des réponses précédentes, cette dernière notion étant bien évidemment plus forte.

Étant donné une classe Cl de distingueurs, on définit le meilleur avantage sur cette classe :

$$\mathrm{Adv}_{Cl}(f, f^*) = \max_{\mathcal{A} \in Cl} \mathrm{Adv}_{\mathcal{A}}(f, f^*).$$

On définit alors trois classes particulières de distingueurs :

Définition 15 (Cas d'un distingueur non-adaptatif) *Soit un entier q. On note Cl^q_{na} la classe des distingueurs non-adaptatifs limités à q textes clairs choisis. Pour construire une attaque faisant appel à cette classe de distingueurs, on choisit q textes clairs que l'on soumet à l'oracle et on obtient la réponse 0 ou 1.*

Définition 16 (Cas d'un distingueur adaptatif) *Soit un entier q. On note Cl^q_a la classe des distingueurs adaptatifs limités à q textes clairs choisis.*

Définition 17 (Cas d'un distingueur super-pseudo-aléatoire) *Soit q un entier. On note Cl^q_s la classe des distingueurs adaptatifs limités à q textes clairs ou chiffrés choisis. Pour attaquer un chiffrement C avec cette classe de distingueur, on peut utiliser soit le chiffrement C, soit son inverse C^{-1}.*

Notons que l'utilisation d'un distingueur super-pseudo-aléatoire n'a de sens que dans le cas de l'étude de permutations.

1.3 Distingueurs et Probabilités de Transitions d'une Fonction Aléatoire

Il est possible d'établir un lien entre $\Pr[x \overset{f}{\mapsto} y]$ et la notion de distingueur comme l'a le premier démontré J. Patarin dans [Pat91]. Le meilleur avantage $\mathrm{Adv}_{\mathcal{A}}(f, f^*)$ pour tout distingueur $\mathcal{A}$ à q requêtes de distinguer f de f^* est entièrement déterminé par $Pr[x \overset{f}{\mapsto} y]$, comme démontré dans [Pat91] :

Théorème 11 (Patarin) *Soit f une fonction aléatoire de $F_{n,m}$ et f^* une fonction aléatoire parfaite de $F_{n,m}$. Soit q un entier. On note $\mathcal{X}$ l'ensemble des $x = (x_1, \cdots, x_q)$ q-uplets de valeurs de I_n deux à deux distinctes. Si il existe un sous ensemble $\mathcal{Y}$ de ${I_m}^q$ et deux nombres réels positifs ϵ_1 et ϵ_2 tel que :*
1°) $|\mathcal{Y}| \geq (1 - \epsilon_1) \cdot |I_m|^q$ (i)
2°) $\forall x \in \mathcal{X},\ \ \forall y \in \mathcal{Y},\ Pr[x \overset{f}{\mapsto} y] \geq (1 - \epsilon_2) \cdot \frac{1}{|I_m|^q}$ (ii)
alors pour tout distingueur A adaptatif utilisant q requêtes, on a

$$Adv_A(f, f^*) \leq \epsilon_1 + \epsilon_2$$

Preuve : On se limite dans cette démonstration au cas d'un distingueur adaptatif A, déterminé par un aléa ω, faisant appel à q requêtes supposées distinctes deux à deux. Les calculs possibles de A peuvent être partitionnés selon la valeur de l'aléa ω et de la réponse $y = (y_1, \cdots, y_q)$. En effet, si on note $x = (x_1, \cdots, x_q)$ le q-uplet d'entrées, la i-ème requête x_i est entièrement déterminée par la valeur de ω et par $y_1, \cdots, y_{i-1}$. La réponse du distingueur A ne dépend donc que du couple (ω, y). On note $\mathcal{A}$ l'ensemble de tous les couples (ω, y) pour lesquels la réponse du distingueur est 1. On a :

$$Adv_A(f, f^*) = |p - p^*| \text{ avec } p = \sum_{(\omega,y)\in\mathcal{A}} \Pr[\omega] \cdot \Pr[x(\omega, y) \overset{f}{\mapsto} y]$$

$$\text{et } p^* = \sum_{(\omega,y)\in\mathcal{A}} \Pr[\omega] \cdot \Pr[x(\omega, y) \overset{f^*}{\mapsto} y]$$

On obtient ces égalités en partionnant les valeurs de ω et f acceptées par A selon la valeur de la liste y de réponses qu'elles déterminent.

On utilise alors la condition (ii) du théorème :
$\forall x \in \mathcal{X},\ \forall y \in \mathcal{Y},\ \Pr[x \overset{f}{\mapsto} y] \geq (1-\epsilon_2)\Pr[x \overset{f^*}{\mapsto} y]$ sur l'espace $\mathcal{Y}$ pour obtenir :

$$p^* - p \leq \sum_{(\omega,y)\in\mathcal{A},\ y\in\mathcal{Y}} \Pr[\omega]\epsilon_2 \Pr[x(\omega,y) \overset{f^*}{\mapsto} y] + \sum_{(\omega,y)\in\mathcal{A},\ y\notin\mathcal{Y}} \Pr[\omega]\Pr[x(\omega,y) \overset{f^*}{\mapsto} y]$$

On utilise alors la condition (i) pour borner la seconde partie du terme de droite de notre inégalité.
On réécrit la condition (i) de la manière suivante : $\epsilon_1 \geq 1 - |\mathcal{Y}|\Pr[x(\omega,y) \overset{f^*}{\mapsto} y]$ soit $\epsilon_1 \geq |\bar{\mathcal{Y}}|\Pr[x(\omega,y) \overset{f^*}{\mapsto} y]$. L'inégalité devient donc :

$$p^* - p \leq \sum_{(\omega,y)\in\mathcal{A},\ y\in\mathcal{Y}} \Pr[\omega]\epsilon_2 \Pr[x(\omega,y) \overset{f^*}{\mapsto} y] + \epsilon_1$$

De plus, on a :

$$\sum_{(\omega,y)\in\mathcal{A},\ y\in\mathcal{Y}} \Pr[\omega]\epsilon_2 \Pr[x(\omega,y) \overset{f^*}{\mapsto} y] \leq \epsilon_2$$

car

$$\sum_{(\omega,y)\in\mathcal{A},\ y\in\mathcal{Y}} \Pr[\omega]\Pr[x(\omega,y) \overset{f^*}{\mapsto} y] \leq \frac{|I_m|^q}{|I_m|^q}$$

En considérant l'inverse de ce distingueur, à savoir celui dont la réponse est 0, on démontre la même inégalité pour $p - p^*$.

On a donc bien :

$$Adv_A(f, f^*) \leq \epsilon_1 + \epsilon_2$$

□

L'utilisation de distingueurs permet donc l'étude théorique des fonctions de chiffrement, plus précisément des schémas de chiffrement construits à partir de fonctions d'étage idéales, en comparant leur distribution de probabilités de transition à celle d'une fonction aléatoire parfaite.

1.4 Exemple du Schéma de Feistel à Trois Étages

Nous présentons ici l'étude de la sécurité d'un schéma de Feistel à 3 étages, d'abord démontré par Luby et Rackoff en 1988 [LR88] puis par J. Patarin en 1991 [Pat91], et enfin par U. Maurer en 1992 [Mau92]. La démonstration présentée s'inspire de celle de [Pat91] qui utilise le théorème 11 et de celle de [Vau98].

Théorème 12 *Soient F_1^*, F_2^*, F_3^* trois fonctions aléatoires parfaites indépendantes définies de $2^{\frac{m}{2}}$ dans lui-même et soit f^* une fonction aléatoire parfaite de 2^m dans 2^m. On note $f = \Psi^3(F_1^*, F_2^*, F_3^*)$ le schéma de Feistel à trois étages lié aux fonctions F_1^*, F_2^* et F_3^*. Alors, pour tout distingueur A adaptatif faisant appel à q requêtes, on a :*

$$Adv_A(f, f^*) \leq q^2 \cdot 2^{-\frac{m}{2}}.$$

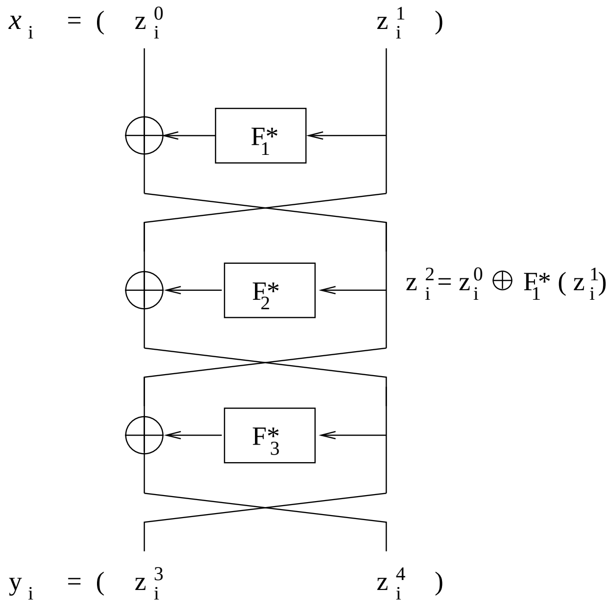


FIG. 1.1 – Schéma de Feistel sur 3 Étages

Preuve : On note $x = (x_i)_{i\in[1..q]} = (z_i^0, z_i^1)_{i\in[1..q]}$ le q-uplet de requêtes et $y = (y_i)_{i\in[1..q]} = (z_i^3, z_i^4)_{i\in[1..q]}$ le q-uplet de sorties correspondant. On note également $z^2 = (z_i^2)_{i\in[1..q]}$ le q-uplet d'éléments z_i^2 de $I_{\frac{m}{2}}$ avec $z_i^2 = z_i^0 \oplus F_1^*(z_i^1)$. On note également $z^0 = (z_i^0)_{i\in[1..q]}$, $z^1 = (z_i^1)_{i\in[1..q]}$,$z^3 = (z_i^3)_{i\in[1..q]}$ et $z^4 = (z_i^4)_{i\in[1..q]}$ les q-uplets correspondants aux différentes valeurs d'entrée et de sortie. On note $\mathcal{Z}$ l'ensemble des q-uplets de valeurs de z_i^2 deux à deux distinctes et comme dans le paragraphe précédent, $\mathcal{X}$ l'ensemble des $x = (x_1, \cdots, x_q)$ q-uplets de valeurs de I_m deux à deux distinctes.

On note $\mathcal{Y}$ l'ensemble défini par :

$$\mathcal{Y} = \{(y_1, \cdots, y_q); \forall i < j,\ z_i^3 \neq z_j^3\}$$

$\mathcal{Y}$ correspond donc à l'ensemble des q-uplets de sortie dont les valeurs de la partie gauche sont deux à deux distinctes. Le cardinal de cet ensemble est tel que :

$$|\mathcal{Y}| \geq \left(1 - \frac{q(q-1)}{2} \cdot 2^{-\frac{m}{2}}\right) 2^{mq}$$

On peut donc poser :

$$\epsilon_1 = \frac{q(q-1)}{2} \cdot 2^{-\frac{m}{2}}$$

pour rester dans le cadre du théorème 11.

À présent, pour tout q-uplet x de l'ensemble $\mathcal{X}$ et pour tout q-uplet y de l'ensemble $\mathcal{Y}$, on cherche à établir une borne inférieure pour la probabilité $\Pr[x \overset{f}{\mapsto} y]$:

$$\begin{array}{rll} \Pr[x \overset{f}{\mapsto} y] & = \sum_{z^2 \in I_{\frac{m}{2}}{}^q, z^3 \in I^q_{\frac{m}{2}}} & \Pr[(z^2 = z^0 \oplus F_1^*(z^1)) \wedge (z^3 = z^1 \oplus F_2^*(z^2)) \\ & & \wedge (z^4 = z^2 \oplus F_3^*(z^3))] \\ & \geq \sum_{z^2 \in \mathcal{Z}, z^3 \in \mathcal{Y}} & \Pr[(z^3 = z^1 \oplus F_2^*(z^2)) \wedge (z^4 = z^2 \oplus F_3^*(z^3))] \\ & & \cdot \Pr[(z^2 = z^0 \oplus F_1^*(z^1))] \quad (1) \end{array}$$

On cherche alors à estimer l'inégalité (1).
Pour cela, pour tout q-uplet z^2 dans $\mathcal{Z}$ et pour tout q-uplet z^3 de $\mathcal{Y}$, on évalue tout d'abord $\Pr[(z^3 = z^1 \oplus F_2^*(z^2)) \wedge (z^4 = z^2 \oplus F_3^*(z^3))] = \Pr[(z^1 \oplus z^3 = F_2^*(z^2))] \cdot \Pr[(z^2 \oplus z^4 = F_3^*(z^3))]$. Comme z^2 appartient à $\mathcal{Z}$ et comme z^3 appartient à $\mathcal{Y}$, alors

$$\Pr[(z^3 = z^1 \oplus F_2^*(z^2)) \wedge (z^4 = z^2 \oplus F_3^*(z^3))] = \left(2^{-\frac{m}{2}}\right)^q \cdot \left(2^{-\frac{m}{2}}\right)^q = 2^{-mq}$$

De plus, comme les z^2 appartiennent à $\mathcal{Z}$, on a :

$$\Pr[(z^2 = z_i^0 \oplus F_1^*(z_i^1))] \geq 1 - \frac{q(q-1)}{2} \cdot 2^{-\frac{m}{2}}$$

On obtient donc

$$\Pr[x \overset{f}{\mapsto} y] \geq 2^{-mq} \cdot \left(1 - \frac{q(q-1)}{2} \cdot 2^{-\frac{m}{2}}\right)$$

On pose alors :

$$\epsilon_2 = \frac{q(q-1)}{2} \cdot 2^{-\frac{m}{2}}.$$

Il ne reste alors plus qu'à appliquer le théorème 11 sur l'ensemble $\mathcal{Y}$, on obtient donc :

$$\begin{array}{rcl} Adv_A(f, f^*) & \leq & \epsilon_1 + \epsilon_2 \\ & \leq & \dfrac{q(q-1)}{2^{\frac{m}{2}}} \\ & \leq & \dfrac{q^2}{2^{\frac{m}{2}}} \end{array}$$

□

D'autres recherches ont été effectuées concernant les preuves de sécurité de schémas de Feistel dans le modèle de Luby et Rackoff. Nous donnerons ici deux autres résultats sans démonstration. Le premier concerne la comparaison d'un schéma de Feistel avec une permutation aléatoire parfaite et le deuxième l'utilisation d'un distingueur super-pseudo-aléatoire, car un schéma de Feistel est une permutation.

Théorème 13 *Soient F_1^*, F_2^*, F_3^* trois fonctions aléatoires parfaites indépendantes définies de $2^{\frac{m}{2}}$ dans lui-même et soit c^* une permutation aléatoire parfaite de 2^m. On note $c = \Psi^3(F_1^*, F_2^*, F_3^*)$ le schéma de Feistel à trois étages lié aux fonctions F_1^*, F_2^*, F_3^*. Alors, pour tout distingueur A adaptatif faisant appel à q requêtes, on a :*

$$Adv_A(c, c^*) \leq q^2 \cdot 2^{-\frac{m}{2}}.$$

Théorème 14 *Soient F_1^*, F_2^*, F_3^*, F_4^* quatre fonctions aléatoires parfaites indépendantes définies de $2^{\frac{m}{2}}$ dans lui-même et soit c^* une fonction aléatoire parfaite de 2^m. On note $c = \Psi^3(F_1^*, F_2^*, F_3^*)$ le schéma de Feistel à quatre étages lié aux fonctions F_1^*, F_2^*, F_3^* et F_4^*. Alors, pour tout distingueur A super-pseudo-aléatoire faisant appel à q requêtes, on a :*

$$Adv_A(c, c^*) \leq q^2 \cdot 2^{-\frac{m}{2}}.$$

Chapitre 2

La Théorie de la Décorrélation

La théorie de la décorrélation a été introduite par Serge Vaudenay en 1998 dans [Vau98a] et [Vau98b]. Elle généralise la notion de fonction universelle définie par Carter et Wegman dans [CW81] et [CW79]. Son but premier est de tenter d'unifier les différents résultats de cryptologie tant dans la résistance à des classes d'attaques que dans les différents modèles de preuves de sécurité à l'aide d'une théorie qui se veut plus globale.

Après avoir donné les définitions de base nécessaires à la compréhension de cette théorie, nous nous intéresserons à quelques exemples simples avant de présenter les résultats de la décorrélation concernant les preuves de sécurité puis la résistance contre les classes d'attaques connues. Nous aborderons enfin quelques exemples de la mise en oeuvre de cette théorie à travers une famille de schémas de chiffrement appelée PEANUT et à travers DFC [GGH+98], candidat de l'AES [AES98].

2.1 Premières Définitions

2.1.1 Matrice de Distribution à l'Ordre d

Définition 18 *Étant donné une fonction aléatoire F d'un ensemble M_1 vers un ensemble M_2 et un entier naturel d, on définit la "matrice de décorrélation d'ordre d" que l'on note $[F]^d$ comme étant la matrice dont les lignes sont indexées par les d-uplets $x = (x_1, ..., x_d)$ d'éléments de M_1 et les colonnes sont indexées par les d-uplets $y = (y_1, ..., y_d)$ d'éléments de M_2, et telle que*

$$[F]^d_{x,y} = Pr[F(x_i) = y_i; i = 1, ..., d].$$

La matrice de décorrélation d'une fonction F à l'ordre d contient les probabilités de transition de tous les d-uplets de valeurs d'entrée vers tous les d-uplets de valeurs de sortie. La matrice $[F]^d$ décrit toutes les possibilités de transition pour toutes les valeurs des espaces d'entrée et de sortie. $[F]^d$ appartient à l'ensemble des matrices $R^{|M_1|^d \times |M_2|^d}$.

De même, on peut définir la matrice de décorrélation à l'ordre d d'une permutation aléatoire C.

Propriété 5 *Si F et G sont deux fonctions aléatoires de distributions indépendantes, on* *a alors pour tout entier naturel d la relation suivante :*

$$[F \circ G]^d = [G]^d \times [F]^d$$

où $\times$ désigne le produit matriciel classique.

Propriété 6 *Si F^* désigne la fonction aléatoire idéale, on a :*

$$[F]^d \times [F^*]^d = [F^* \circ F]^d = [F^*]^d$$

Nous avons la même propriété pour une permutation aléatoire C et une permutation aléatoire parfaite C^*.

2.1.2 Distance et Biais de Décorrélation

Intuitivement, la décorrélation d'une fonction aléatoire F est la distance de $[F]^d$ à $[F^*]^d$ pour une fonction F^* de référence, c'est à dire ici une fonction de distribution uniforme. On calcule de même la décorrélation d'une permutation C comme étant la distance de $[C]^d$ à $[C^*]^d$ où C^* désigne une permutation de distribution uniforme.

Définition 19 *Etant donné deux fonctions aléatoires F et G d'un ensemble M_1 vers un ensemble M_2, un entier naturel d et une distance D sur l'ensemble de matrices $R^{M_1^d \times M_2^d}$, on appelle $D([F]^d, [G]^d)$ la "distance de décorrélation d'ordre d entre F et G". Lorsque cette distance est nulle, on dit que F et G ont la même décorrélation. Si G admet une distribution uniforme, on appelle $D([F]^d, [G]^d)$ le "biais de décorrélation d'ordre d de la fonction F" que l'on note $DecF^d(F)$. Si F et G sont bijectives et si la distribution de G admet une distribution uniforme, on appelle $D([F]^d, [G]^d)$ le "biais de décorrélation d'ordre d de la permutation F" que l'on note $DecP^d(F)$. Lorsqu'un biais de décorrélation est nul, on dit que la décorrélation est "parfaite".*

On se pose alors la question du choix de la norme et de la distance induite sur un ensemble de matrices.

2.1.3 Choix de la Norme et de la Distance

S. Vaudenay présente plusieurs résultats faisant appel à différentes normes, toutes définies comme étant des normes sur des espaces matriciels.

Définition 20 *Soit M_1 et M_2 deux ensembles et d un entier. Pour une matrice A de $R^{M_1^d \times M_2^d}$, on définit :*

$$|| A ||_2 = \sqrt{\sum_{x,y} (A_{x,y})^2} \tag{2.1}$$

$$||| A |||_\infty = \max_x \sum_y | A_{x,y} | \tag{2.2}$$

$$|| A ||_a = \max_{x_1} \sum_{y_1} \max_{x_2} \sum_{y_2} \cdots \max_{x_d} \sum_{y_d} | A_{(x_1,\cdots,x_d),(y_1,\cdots,y_d)} | \tag{2.3}$$

$$|| A ||_s = \max\left(\max_{x_1} \sum_{y_1} || \pi_{x_1,y_1}(A) ||_s, \max_{y_1} \sum_{x_1} || \pi_{x_1,y_1}(A) ||_s \right) \tag{2.4}$$

où la norme d'une matrice réduite à une entrée est la valeur absolue et où $\pi_{x_1,y_1}(A)$ est la matrice de $R^{M_1^{d-1}\times M_2^{d-1}}$ telle que

$$(\pi_{x_1,y_1}(A))_{(x_2,\cdots,x_d),(y_2,\cdots,y_d)} = A_{(x_2,\cdots,x_d),(y_2,\cdots,y_d)}$$

La norme (1) est la norme L_2. Les résultats sur la décorrélation concernant cette norme sont donnés dans [Vau98a]. La norme (2) est la norme de matrice associé à L_∞. Nous nous servirons essentiellement de cette norme étudiée dans [Vau99d]. Les deux dernières normes définies plus récemment dans [Vau99a] et [Vau99b] ont également permis de donner des résultats de sécurité intéressants développés dans la suite de cette présentation. On peut remarquer que : $||\cdot||_s \geq ||\cdot||_a \geq |||\cdot|||_\infty$.

Toutes ces normes vérifient l'inégalité

$$|| A\times B || \leq || A || \cdot || B || .$$

On peut démontrer la propriété suivante : si F_1 et F_2 sont deux fonctions aléatoires indépendantes, $\mathrm{DecF}^d(F_1\circ F_2) \leq \mathrm{DecF}^d(F_1)\cdot\mathrm{DecF}^d(F_2)$ en utilisant les propriétés 5 et 6 et l'inégalité additive des normes $||x||+||y|| \geq ||x+y||$. Cette inégalité se vérifie également dans le cas de permutations.

On peut alors calculer les biais de décorrélation grâce aux distances induites par les normes définies ci-dessus.

2.2 Premiers Exemples

Tous les résultats présentés ci-dessous sont calculés à partir de la norme $|||\cdot|||_\infty$.

2.2.1 Matrices de Décorrélation Élémentaires

On se limite ici au cas de matrices de décorrélation à l'ordre 2. On suppose que les fonctions et permutations utilisées dans ce paragraphe sont définies sur un groupe G et on pose $N = \#G$.

Dans le cas d'une fonction aléatoire parfaite F^*, sa matrice de décorrélation à l'ordre 2 vérifie $[F]^2_{(x_1,x_2),(y_1,y_2)} = \Pr[F(x_1)=y_1 \text{ et } F(x_2)=y_2]$ et vaut :

$$\begin{array}{cc} & \begin{array}{cc} \quad y_1=y_2 \quad & \quad y_1\neq y_2 \end{array} \\ \begin{array}{c} x_1=x_2 \\ \\ \\ x_1\neq x_2 \end{array} & \begin{pmatrix} \frac{1}{N} & \cdots & \frac{1}{N} & 0 & \cdots & 0 \\ \vdots & & \vdots & \vdots & & \vdots \\ \frac{1}{N} & \cdots & \frac{1}{N} & 0 & \cdots & 0 \\ \frac{1}{N^2} & \cdots & \cdots & \cdots & \cdots & \frac{1}{N^2} \\ \vdots & & & & & \vdots \\ \frac{1}{N^2} & \cdots & \cdots & \cdots & \cdots & \frac{1}{N^2} \end{pmatrix} \end{array}$$

Dans le cas d'une permutation aléatoire parfaite C^*, la matrice de décorrélation à l'ordre 2 vaut :

$$\begin{array}{cc}
 & \begin{array}{cc} \qquad y_1 = y_2 \qquad & \qquad y_1 \neq y_2 \qquad \end{array} \\
\begin{array}{c} x_1 = x_2 \\ \\ \\ x_1 \neq x_2 \end{array} &
\begin{pmatrix}
\frac{1}{N} & \cdots & \frac{1}{N} & 0 & \cdots & 0 \\
\vdots & & \vdots & \vdots & & \vdots \\
\frac{1}{N} & \cdots & \frac{1}{N} & 0 & \cdots & 0 \\
0 & \cdots & 0 & \frac{1}{N(N-1)} & \cdots & \frac{1}{N(N-1)} \\
\vdots & & \vdots & & & \vdots \\
0 & \cdots & 0 & \frac{1}{N(N-1)} & \cdots & \frac{1}{N(N-1)}
\end{pmatrix}
\end{array}$$

2.2.2 Chiffrement de Vernam

La fonction de chiffrement de Vernam définie sur un groupe G avec $N = \#G$ par $C(x) = x + K$ où K admet une distribution uniforme a une décorrélation d'ordre 1 parfaite. En revanche, on a

$$\mathrm{DecP}^2(C) = 2 - \frac{2}{N-1}$$

.

2.2.3 La Fonction de Chiffrement $C(x) = A \cdot x + B$

La fonction de chiffrement $C(x) = A \cdot x + B$ définie par une clef $K = (A, B)$ aléatoire avec $A \neq 0$ et de distribution uniforme admet une décorrélation (de permutation) d'ordre 2 parfaite.

2.2.4 La Fonction $F(x) = ((A \cdot x + B) \bmod p) \bmod 2^m$

Cette fonction de chiffrement est définie sur $\{0,1\}^m$ par une clef $K = (A, B) \in \{0,1\}^{2m}$ de distribution uniforme et la donnée d'un nombre premier $p \geq 2^m$, on a :

$$\mathrm{DecF}^2(F) = 4\epsilon + 2\epsilon^2$$

avec $p = 2^m(1 + \epsilon)$.

Cette fonction est celle utilisée dans DFC (voir [GGH$^+$98]) avec les valeurs suivantes : $m = 64$, $p = 2^{64} + 13$.

2.3 Résultats de Sécurité Revisités

On assimile dans tout le paragraphe qui suit une fonction de chiffrement à une permutation. Tous les résultats feront intervenir le calcul de DecP^d.

2.3.1 La Sécurité Inconditionnelle de Shannon

On ne peut évoquer la notion de sécurité dans le chiffrement symétrique sans parler de la théorie de l'information de Shannon [Sha49] et de sa définition de "sécurité parfaite". Pour cela, on définit l'entropie d'une variable aléatoire X qui représente l'information moyenne contenue dans un résultat par :

$$H(X) = -\sum_x \Pr[X = x] \log_2 \Pr[X = x]$$

.

Définition 21 *Une fonction de chiffrement aléatoire C assure une "confidentialité parfaite" sur un message aléatoire X si l'on a : $H(X|C(X)) = H(X)$.*

Cela signifie que la connaissance du chiffré n'apporte aucune information sur le message clair. Le plus bel exemple de chiffrement vérifiant cette définition est le chiffrement de Vernam traité en exemple si il est utilisé une seule fois et si K est aléatoire de distribution uniforme. D'où le théorème de Shannon :

Théorème 15 *Si une fonction de chiffrement aléatoire C assure une confidentialité parfaite sur un message aléatoire X, on a*

$$H(C) \geq H(X).$$

En terme de théorie de la décorrélation, cela signifie que la décorrélation à l'ordre 1 est parfaite. On peut également généraliser la notion de confidentialité parfaite pour des fonctions utilisées plusieurs fois. Le lien avec la décorrélation parfaite à l'ordre d est alors facile à faire. En effet, si on préserve la confidentialité parfaite d'une fonction utilisée deux fois, cela signifie que la décorrélation à l'ordre 2 de cette fonction est parfaite.

2.3.2 Décorrélation et Indistingabilité

On s'intéresse dans ce paragraphe au lien existant entre la théorie de la décorrélation et la notion de distingueur. S. Vaudenay a présenté des travaux faisant intervenir les normes (2), (3), (4) définies plus haut afin de relier chaque classe de distingueurs à une distance de décorrélation impliquant une norme particulière. On obtient les résultats suivants.

Théorème 16 (Cas d'un distingueur non-adaptatif) *Soit un entier d. On note Cl_{na}^d la classe des distingueurs non-adaptatifs limités à d textes clairs choisis. Pour construire une attaque faisant appel à cette classe de distingueurs, on choisit d textes clairs que l'on soumet à l'oracle et on obtient une série de réponses 0 ou 1. Pour toute fonction aléatoire F, pour toute fonction F^* de distribution uniforme, on a :*

$$Adv_{Cl_{na}^d}(F, F^*) = \frac{1}{2} \cdot DecF^d_{|||\cdot|||_\infty}(F).$$

où $Adv_{Cl_{na}^d}(F, F^*)$ représente comme dans le chapitre précédent le meilleur avantage sur la classe des distingueurs non-adaptatifs faisant appel à d requêtes.

De même, pour tout chiffrement C, pour tout chiffrement aléatoire parfait C^*, on a :

$$Adv_{Cl_{na}^d}(C, C^*) = \frac{1}{2} \cdot DecP^d_{|||\cdot|||_\infty}(C).$$

Éléments de Preuve : Nous donnons ici quelques éléments de preuve pour la première égalité. On a :

$$\mathrm{DecF}^d_{|||\cdot|||_\infty}(F) = \max_x \sum_y \mid \Pr[x \overset{F}{\mapsto} y] - \Pr[x \overset{F^*}{\mapsto} y] \mid$$

Dans le cas d'un multi-point $x = x^0$ qui maximise cette valeur, on tient compte de tous les distingueurs possibles, c'est à dire de ceux qui vont répondre 1 mais aussi de ceux qui vont répondre 0. La valeur maximale est obtenue pour le multi-point x^0 qui maximise cette relation dans la distribution de Dirac, c'est à dire pour le multi-point qui est tel que :

$$\sum_y \mid \Pr[x^0 \overset{F}{\mapsto} y] - \Pr[x^0 \overset{F^*}{\mapsto} y] \mid = 2 \cdot \mathrm{Adv}_{Cl^d_{na}}(F, F^*)$$

□

Théorème 17 (Cas d'un distingueur adaptatif) *Soit un entier d. On note Cl^d_a la classe des distingueurs adaptatifs limités à d textes clairs choisis. Pour toute fonction aléatoire F, pour toute fonction F^* de distribution uniforme, on a :*

$$Adv_{Cl^d_a}(F, F^*) = \frac{1}{2} \cdot DecF^d_{||\cdot||_a}(F).$$

De même, pour tout chiffrement C, pour tout chiffrement aléatoire parfait C^*, on a :

$$Adv_{Cl^d_a}(C, C^*) = \frac{1}{2} \cdot DecP^d_{||\cdot||_a}(C).$$

Théorème 18 (Cas d'un distingueur super-pseudo-aléatoire) *Soit d un entier. On note Cl^d_s la classe des distingueurs adaptatifs limités à d textes clairs ou chiffrés choisis. Pour attaquer un chiffrement C avec cette classe de distingueurs, on peut utiliser soit le chiffrement C, soit son inverse C^{-1}. Pour tout chiffrement C, pour tout chiffrement idéal C^*, on a :*

$$Adv_{Cl^d_s}(C, C^*) = \frac{1}{2} \cdot DecP^d_{||\cdot||_s}(C).$$

On obtient finalement les relations suivantes qui permettent de caractériser le chiffrement itératif $C_r \circ \cdots \circ C_1$ composé de r instances indépendantes C_1,...,C_r du même chiffrement C :

$$\mathrm{Adv}_{Cl^d_{na}}(C_r \circ \cdots \circ C_1, C^*) \leq 2^{r-1} \cdot \left(\mathrm{Adv}_{Cl^d_{na}}(C, C^*)\right)^r$$

$$\mathrm{Adv}_{Cl^d_a}(C_r \circ \cdots \circ C_1, C^*) \leq 2^{r-1} \cdot \left(\mathrm{Adv}_{Cl^d_a}(C, C^*)\right)^r$$

$$\mathrm{Adv}_{Cl^d_s}(C_r \circ \cdots \circ C_1, C^*) \leq 2^{r-1} \cdot \left(\mathrm{Adv}_{Cl^d_s}(C, C^*)\right)^r.$$

Cela signifie que le meilleur avantage pour distinguer un chiffrement donné d'un chiffrement aléatoire parfait C^* décroît exponentiellement avec le nombre de tours r.

2.3.3 Sécurité dans le Modèle de Luby-Rackoff

Grâce au paragraphe précédent, nous avons assez d'outils en main pour pouvoir relier la théorie de la décorrélation et la sécurité dans le modèle de Luby-Rackoff. Rappelons la définition d'un schéma de Feistel :
Soit F_1 une fonction de $\{0,1\}^{\frac{m}{2}}$ vers $\{0,1\}^{\frac{m}{2}}$. On définit la fonction $\Psi(F_1)$ de $\{0,1\}^m$ vers $\{0,1\}^m$ par :

$$\forall (L,R) \in (\{0,1\}^m)^2, \Psi(F_1)[L,R] = [S,T] \Leftrightarrow \left\{ \begin{array}{l} S = R \\ T = L \oplus F_1(R) \end{array} \right.$$

La fonction Ψ représente un schéma de Feistel sur un étage. On peut alors généraliser la définition de Ψ à plusieurs étages : Soit F_1, F_2,..., F_k, k fonctions de $\{0,1\}^{\frac{m}{2}}$ vers $\{0,1\}^{\frac{m}{2}}$, on définit Ψ^k de $\{0,1\}^m$ vers $\{0,1\}^m$ par :

$$\Psi^k(F_1, ..., F_k) = \Psi(F_k) \circ \cdots \circ \Psi(F_2) \circ \Psi(F_1).$$

(voir figure 1.1 du chapitre 1 dans le cas $k = 3$).

Le théorème de Luby-Rackoff, démontré en 1988 [LR88], s'énonce alors ainsi :

Théorème 19 *Pour un ensemble $M = \{0,1\}^m$, on considère trois fonctions aléatoires F_1, F_2, F_3 indépendantes et de distribution uniforme de $\{0,1\}^{\frac{m}{2}}$ vers $\{0,1\}^{\frac{m}{2}}$. On considère également une permutation aléatoire C^* sur M de distribution uniforme. Pour tout distingueur $\mathcal{A}$ entre C^* et $\Psi^3(F_1,F_2,F_3)$ limité à d requêtes, on a :*

$$Adv_{\mathcal{A}}(\Psi^3(F_1,F_2,F_3), C^*) \leq \frac{d^2}{2^{\frac{m}{2}}}.$$

Ce théorème peut être vu comme une conséquence du théorème équivalent suivant, liant un schéma de Feistel à trois étages à la théorie de la décorrélation.

Théorème 20 *Pour un ensemble $M = \{0,1\}^m$ (m pair), on considère trois fonctions aléatoires F_1, F_2, F_3 indépendantes de $\{0,1\}^{\frac{m}{2}}$ vers $\{0,1\}^{\frac{m}{2}}$ et de décorrélation parfaite à l'ordre d. On a*

$$DecP^d_{|||\cdot|||_\infty}(\Psi^3(F_1,F_2,F_3)) \leq \frac{2d^2}{2^{\frac{m}{2}}}.$$

On obtient également grâce aux résultats du paragraphe précédent une généralisation de ce théorème à r étages d'un schéma de Feistel. Ce résultat établi tout d'abord par Y. Zheng, T. Matsumoto et H. Imai dans [ZMI90] trouve tout naturellement sa place dans la théorie de la décorrélation [Vau99b] :

Théorème 21 *Soit F_1,..., F_r r fonctions aléatoires indépendantes ($r \geq 3$) sur $\{0,1\}^{\frac{m}{2}}$, selon la norme que l'on utilise, on obtient :*

Si $DecP^d_{|||\cdot|||_\infty}(F_i) \leq \epsilon$, *alors* $DecP^d_{|||\cdot|||_\infty}(\Psi^r(F_1,\cdots,F_r)) \leq \left(3\epsilon + 3\epsilon^2 + \epsilon^3 + 2d^2 \cdot 2^{-\frac{m}{2}}\right)^{\lfloor \frac{r}{3} \rfloor}$.

$$\textit{Si } DecP^d_{||\cdot||_a}(F_i) \leq \epsilon, \textit{ alors } DecP^d_{||\cdot||_a}(\Psi^r(F_1,\cdots,F_r)) \leq \left(3\epsilon + 2d^2 \cdot 2^{-\frac{m}{2}}\right)^{\lfloor \frac{r}{3} \rfloor}.$$

$$\textit{Si } DecP^d_{||\cdot||_s}(F_i) \leq \epsilon, \textit{ alors } DecP^d_{||\cdot||_s}(\Psi^r(F_1,\cdots,F_r)) \leq \left(4\epsilon + 2d^2 \cdot 2^{-\frac{m}{2}}\right)^{\lfloor \frac{r}{4} \rfloor}.$$

2.4 Résistance contre des Classes d'Attaques

Tous les résultats présentés dans ce paragraphe ont été développés par S. Vaudenay dans [Vau98b] et [Vau99d].

2.4.1 Attaques Différentielle et Linéaire

Avant d'établir un lien entre la décorrélation et les différents types d'attaques connues, rappelons quelques définitions :

Définition 22 *Soit C un chiffrement sur le groupe M, on définit le coefficient $DP^C(a,b)$ avec $a \neq 0$ utilisé en cryptanalyse différentielle par :*

$$DP^C(a,b) = Pr_X[C(X+a) = C(X)+b].$$

où X admet une distribution uniforme.
On définit également le coefficient $LP^C(a,b)$ utilisé en cryptanalyse linéaire par :

$$LP^C(a,b) = (2Pr_X[X \cdot a = C(X) \cdot b] - 1)^2.$$

On cherche alors les valeurs moyennes des coefficients DP et LP sur la distribution de la clef K. Pour cela, on définit une fonction de chiffrement C_K et les coefficients suivants :

$$\mathrm{EDP}^C(a,b) = E_K(\mathrm{DP}^{C_K}(a,b))$$

$$\mathrm{ELP}^C(a,b) = E_K(\mathrm{LP}^{C_K}(a,b))$$

où E_K désigne l'espérance sur la distribution de la clé.
On a alors, si $M = \{0,1\}^m$:

$$E_K(\mathrm{DP}^{C_K}(a,b)) = 2^{-m} \sum_{x_1 \neq x_2, y_1 \neq y_2} 1_{\substack{x_1 \oplus x_2 = a \\ y_1 \oplus y_2 = b}} \Pr\begin{bmatrix} x_1 \xrightarrow{C} y_1 \\ x_2 \rightarrow y_2 \end{bmatrix}$$

$$E_K(\mathrm{LP}^{C_K}(a,b)) = 1 - 2^{2-2m} \sum_{x_1 \neq x_2, y_1 \neq y_2} 1_{\substack{x_1 \cdot a = y_1 \cdot b \\ x_2 \cdot a \neq y_2 \cdot b}} \Pr\begin{bmatrix} x_1 \xrightarrow{C} y_1 \\ x_2 \rightarrow y_2 \end{bmatrix}$$

Nous aurons besoin tout au long de ce paragraphe de rendre minimales ces quantités. On considère donc :

$$\mathrm{EDP}^C_{max} = \max_{a \neq 0, b} \mathrm{EDP}^C(a,b))$$

$$\mathrm{ELP}^C_{max} = \max_{a \neq 0, b} \mathrm{ELP}^C(a,b))$$

On peut établir les majorations suivantes :

Théorème 22 *Pour une fonction de chiffrement aléatoire C sur $M = \{0,1\}^m$, on a*

$$EDP^C_{max} \leq \frac{1}{2} DecP^2_{|||\cdot|||_\infty}(C) + \frac{1}{2^m - 1}$$

$$ELP^C_{max} \leq 2 DecP^2_{|||\cdot|||_\infty}(C) + \frac{1}{2^m - 1}.$$

La décorrélation à l'ordre 2 permet donc de formaliser les notions d'attaques linéaires et différentielles. Pour aller plus loin, on définit les distingueurs liés à ces attaques : Dans le cas différentiel, le distingueur est le suivant :

1. pour i allant de 1 à n faire
   (a) Tirer aléatoirement un X et calculer $C(X)$ et $C(X+a)$
   (b) Si $C(X+a)+C(X)=b$ alors arrêter et répondre "1"
2. répondre "0"

où a représente en fait la différence d'entrée et b la différence de sortie.

On obtient alors le résultat suivant :

Théorème 23 *En utilisant le distingueur précédent de complexité n pour un chiffrement C et un chiffrement C^* de distribution uniforme sur $\{0,1\}^m$, on a*

$$Adv(C,C^*) \leq n \cdot \max\left(\frac{1}{2^m-1}, EDP^C(a,b)\right).$$

Dans le cas linéaire, on définit un distingueur linéaire caractérisé par a, b, n et un intervalle I :

1. initialiser un compteur u
2. pour i allant de 1 à n faire
   (a) Tirer aléatoirement un X et calculer $C(X)$
   (b) Si $X \cdot a = C(X) \cdot b$ alors incrémenter u
3. si $u \in I$, répondre 1 sinon répondre 0

On a alors :

Théorème 24 *En utilisant le distingueur précédent de complexité n pour un chiffrement C et un chiffrement C^* de distribution uniforme sur $\{0,1\}^m$, on a*

$$\lim_{n\to\infty} \frac{Adv(C,C^*)}{n^{\frac{1}{3}}} \leq 9.3\left(\max\left(\frac{1}{2^m-1}, ELP^C(a,b)\right)\right)^{\frac{1}{3}}.$$

2.4.2 Attaques Itérées d'Ordre d

Nous venons de voir comment les attaques différentielles et linéaire sont liées à la décorrélation d'ordre 2, nous allons à présent faire le lien entre attaques d'ordre d et décorrélation. Pour cela, nous allons définir un distingueur itératif d'ordre d pour un chiffrement C, suivant le test $\mathcal{T}$, un ensemble de procédures de décision T_i et un ensemble A :

1. Pour i allant de 1 à n faire
   (a) Choisir d requêtes $X=(X_1,...,X_d)$
   (b) Calculer $Y=(C(X_1,...,X_d))$
   (c) Selon la valeur de $\mathcal{T}(X,Y)$, $T_i \leftarrow 0$ ou 1

2. Si $(T_1, ..., T_n) \in A$ répondre "1" sinon répondre "0"

Théorème 25 *Soit C un chiffrement sur un espace de messages de taille m tel que $Dec^{2d}_{|||\cdot|||_\infty} \leq \epsilon$ pour un d donné tel que $d \leq m/2$. Pour toute attaque itérée utilisant le distingueur d'ordre d décrit ci-dessus, si les textes clairs sont indépendants, on a :*

$$Adv(C, C^*) \leq 3\left(\left(2\delta + \frac{5d^2}{2m} + \frac{3\epsilon}{2}\right)n^2\right)^{\frac{1}{3}} + \frac{n\epsilon}{2}$$

où δ représente la probabilité que pour deux X et X' indépendants il existe i et j tels que $X_i = X'_j$.

Ainsi, la décorrélation à l'ordre $2d$ fournit une sécurité prouvée contre des attaques itérées d'ordre d. Ce théorème est la forme la plus générale de la théorie de la décorrélation en ce qui concerne les différentes formes d'attaques.

2.5 Mise en Oeuvre

L'un des plus beaux exemples d'application de la théorie de la décorrélation est l'algorithme DFC [GGH+98], candidat de l'AES. Il utilise un module de décorrélation inspiré de la famille des PEANUT [Vau99c].

2.5.1 Les Chiffrements de la famille PEANUT

On caractérise les chiffrements de cette famille par les paramètres (m, r, d, p). Ce sont des chiffrements de Feistel sur r tours qui utilisent des blocs de longueur m avec m pair. d est l'ordre de décorrélation partielle du chiffrement et p est un nombre premier légèrement supérieur à $2^{\frac{m}{2}}$.

Le chiffrement est paramétré par une clé de longueur $\frac{mrd}{2}$ bits qui représente r listes (une pour chaque tour) de d nombres k_0, k_1, $\cdots$, k_{d-1} de longueur $\frac{m}{2}$-bits chacun. On définit alors une fonction F, utilisée à chaque tour, par :

$$F(x) = g(k_{d-1} \cdot x^{d-1} + k_{d-2} \cdot x^{d-2} + \cdots + k_1 \cdot x + k_0 \bmod p \bmod 2^{\frac{m}{2}})$$

où g est une permutation quelconque de l'ensemble $\{0,1\}^{\frac{m}{2}}$.

Une fois cette définition acquise, on peut estimer le biais de décorrélation (pour la norme infinie) des chiffrements de cette forme. Pour cela, on utilise le lemme suivant :

Lemme 3 *Soit F une fonction sur un tour de la famille des PEANUT de paramètres (m, r, d, p). Soit R une fonction aléatoire de distribution uniforme, on a alors :*

$$|||\, [F]^d - [R]^d \,|||_\infty \leq 2\left(p^d 2^{-\frac{md}{2}} - 1\right)$$

On peut alors en déduire le biais de décorrélation du chiffrement complet sur r tours en appliquant le théorème 21. On obtient alors le résultat suivant :

Théorème 26 *Soit C un chiffrement de la famille des PEANUT de paramètres (m, r, d, p) et soit C^* le chiffrement parfait. On a alors :*

$$|||\, [C]^d - [C^*]^d \,|||_\infty \leq \left(\left(1 + 2\left(p^d 2^{-\frac{md}{2}} - 1\right)\right)^3 - 1 + \frac{2d^2}{2^{\frac{m}{2}}}\right)^{\lfloor \frac{r}{3} \rfloor}$$

2.5.2 DFC

DFC est le candidat français de l'A.E.S. et surtout un exemple de la théorie de la décorrélation. Il utilise des clés de longueur 128, 192 ou 256 bits et fait partie de la famille des PEANUT avec comme paramètres : $m = 128$, $r = 8$, $d = 2$, $p = 2^{64} + 13$.

Le module de décorrélation est donc :

$$\mathrm{RF}_{a|b}(x) = \mathrm{CP}\left(ax + b \bmod (2^{64}+13) \bmod 2^{64}\right)$$

où a et b sont les deux moitiés d'une sous clé de 128 bits et où CP est une permutation sur l'ensemble des mots de 64 bits.

En appliquant les résultats de sécurité de la famille des PEANUT on obtient donc des bornes de sécurité prouvées pour DFC :

- Si la paire $a|b$ a une distribution uniforme, on a :

  $$||\ [\mathrm{RF}_{a|b}]^2 - [R]^2\ || \leq 2\left((2^{64}+13)^2 2^{-128} - 1\right) = 0,813 \cdot 2^{-58}$$

  où R est une fonction aléatoire de distribution uniforme sur l'ensemble des mots de 64 bits.
- On obtient également si on utilise la fonction RF dans un schéma de Feistel à trois étages un biais de décorrélation à l'ordre 2 inférieur à $0{,}641.2^{-56}$ et dans le cas d'un schéma à six étages un biais inférieur à $0{,}821.2^{-113}$.
- Dans le cas d'un distingueur différentiel (toujours dans le cas d'un schéma de Feistel à 6 étages), le nombre de clairs choisis nécessaires pour obtenir un avantage de 1% est d'au moins 2^{107}.
- De même, dans le cas d'un distingueur linéaire, le nombre de textes clairs connus nécessaires est d'au moins 2^{81}.

On peut noter ici que la condition "$a|b$ a une distribution uniforme" est garantie par l'utilisation dans la construction de clé de la fonction de chiffrement de DFC sur 4 tours d'un schéma de Feistel.

Dans le cas de DFC, la théorie de la décorrélation permet d'établir des bornes de sécurité prouvées suffisamment petites pour écarter toute possibilité d'attaques d'ordre 2.

2.6 Conclusion

Nous avons présenté ici les principaux résultats de la théorie de la décorrélation et les plus beaux exemples de sa mise en oeuvre. Elle permet de développer des preuves de sécurité non heuristiques.

Précisons également que des études plus récentes portant sur la décorrélation se sont intéressées à d'autres schémas que le traditionnel schéma de Feistel. L'article [MV00] porte en effet sur différents schémas utilisés par les candidats de l'AES et s'attache à calculer le nombre de tours nécessaires pour assurer une sécurité suffisante en termes de "pseudo-aléatoirité" (i.e. on considère ici l'avantage d'un distingueur adaptatif) et de "super-pseudo-aléatoirité" (dans le cas d'un distingueur super-pseudo-aléatoire) en considérant que la distance de décorrélation entre le schéma étudié composé de fonctions aléatoires parfaites et une permutation aléatoire parfaite est bornée par un coefficient égal au moins à

2^{128} afin de rendre toute attaque impraticable. Par exemple, dans le cas du schéma de RC6 [RRSY98], le nombre minimum d'étages pour assurer la "pseudo-aléatoirité" du schéma est de 5 en prenant $d << 2^{16}$ et de 8 si on s'intéresse à la "super-pseudo-aléatoirité" du schéma.

Chapitre 3

Nouveaux Résultats de Sécurité Prouvée

Dans ce chapitre, nous présenterons les résultats, établis dans l'article [GM01], concernant les preuves de sécurité dans le modèle de Luby-Rackoff de deux types de schémas dérivés de celui de Feistel que nous comparerons à des permutations aléatoires parfaites. Ces schémas génèrent des permutations de $2n$ bits à partir de permutations de n bits. Nous ne nous intéresserons ici qu'aux cas où les fonctions internes aux schémas sont des permutations, ce cas étant le plus usité dans la "cryptographie réelle".

Nous analyserons tout d'abord un schéma nommé schéma L utilisé par M. Matsui dans Misty [Mat97] et dans Kasumi [KAS99], variante de Misty, adopté récemment comme algorithme de chiffrement et d'intégrité dans le système de troisième génération de mobiles, puis à un autre schéma dual du précédent nommé schéma R.

De nombreuses alternatives aux schémas de Feistel ont déjà été étudiées. Citons ici l'exemple de celui proposé par X. Lai et J. Massey pour l'algorithme IDEA [LM90] ou ceux, versions parallélisables du schéma de Feistel, présentés par W. Aiello et R. Venkatesan dans [AV96].

Les résultats développés ici s'intéressent à des attaques par distingueur soit adaptatif soit super-pseudo-aléatoire pour des versions des schémas R et L sur 3, 4 et 5 étages. En ce qui concerne le schéma L, Y. Zheng, T. Matsumoto et H. Imai ont prouvé que ce schéma n'était pas pseudo-aléatoire dans sa version à 3 étages [ZMI90]. Quelques années plus tard, M. Sugita [Su96] et K. Sakurai et Y. Zheng [SZ97] ont établi que ce schéma était pseudo-aléatoire sur 4 étages alors que cette même version n'est pas super-pseudo-aléatoire. Nous développerons ici une preuve de sécurité sur la version à 5 étages du schéma L qui montre qu'il est super-pseudo-aléatoire. Quant au schéma R, il a plus rarement été étudié car il est moins usité, cependant, il est pseudo-aléatoire dans une version à 3 étages contrairement au schéma L. Il est donc également pseudo-aléatoire sur 4 étages bien que dans ce cas il ne soit pas super-pseudo-aléatoire. En revanche, nous prouvons que la version sur 5 étages est super-pseudo-aléatoire.

Les bornes trouvées dans toutes les démonstrations qui suivront sont très proches de celles établies par Luby et Rackoff pour le schéma de Feistel.

3.1 Préliminaires

3.1.1 Fonctions Aléatoires et Distingueurs

Nous supposons ici que le lecteur est familier des différentes notions utilisées dans les preuves de sécurité du type Luby-Rackoff. Tous les concepts utilisés ici ainsi que la plupart des notations sont ceux présentés dans le premier chapitre de cette partie.

Précisons cependant quelques notations abrégées utilisées dans les démonstrations qui vont suivre :

- I représentera l'ensemble $(I_n)^q$.
- $I^{\neq} = ((I_n)^q))^{\neq}$ représentera le sous-ensemble d'éléments de $(I_n)^q$ où les q-uplets de valeurs de I_n sont deux à deux distinctes. On note également $I^{=} = (I_n)^q \backslash I^{\neq}$.
- $\mathcal{X}$ représentera le sous-ensemble d'éléments de $(I_{2n})^q$ où les q-uplets $(x_1, \cdots, x_q)$ de valeurs de I_{2n} sont deux à deux distinctes.
- $\mathcal{Y}$ représentera le sous-ensemble d'éléments de $(I_{2n})^q$ où les q-uplets $(y_1, \cdots, y_q)$ de valeurs de I_{2n} sont deux à deux distinctes. Cet ensemble $\mathcal{Y}$ sera cependant redéfini à chacune de ses utilisations, sa définition exacte pouvant légèrement varier selon les besoins.

3.1.2 Description du Schéma L et du Schéma R

Nous allons à présent décrire les deux types de schémas que nous allons étudier, variantes de celui de Feistel, nommés schéma L et schéma R par Y. Kaneko, F. Sano et K. Sakurai dans [KSS97].

Ces deux schémas génèrent des permutations de $2n$ bits vers $2n$ bits à partir de permutations de n bits vers n bits (une par étage). Il est ici nécessaire d'utiliser des permutations de n bits vers n bits et non des fonctions comme c'est le cas pour le schéma de Feistel.

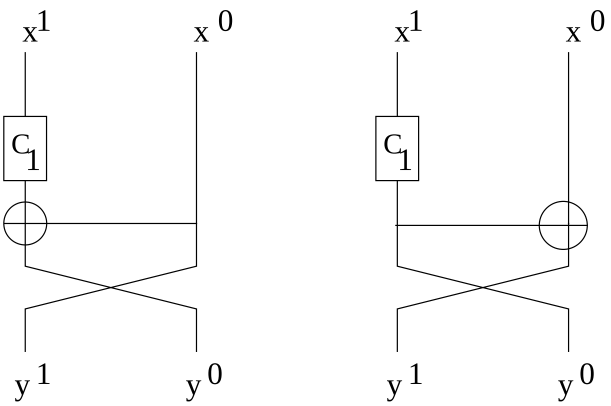


FIG. 3.1 – Schéma L sur un Tour à Gauche et Schéma R sur un Tour à Droite

On formalise l'écriture du schéma L, décrit ci-dessus, sur un étage de la manière suivante :
Soit c_1 une permutation de l'ensemble I_n. Cette permutation est alors transformée en une

permutation $\psi_L(c_1)$ de l'ensemble I_{2n} définie par

$$\psi_L(c_1)(x^1, x^0) = (x^0, c_1(x^1) \oplus x^0).$$

L'extension à r étages de ce schéma transforme r permutations $c_1, \cdots, c_r$ de I_n en une permutation de I_{2n} :

$$\psi_L(c_1, c_2, ..., c_r) = \psi_L(c_r) \circ \cdots \circ \psi_L(c_1).$$

Comme signalé précédemment, ce schéma est utilisé dans les fonctions FI et FO de Misty et de Kasumi notamment à cause de sa rapidité car on peut exécuter en parallèle deux des permutations c_i.

Quant au schéma construit à partir du schéma R, il transforme une permutations c_1 de I_n en une permutation $\psi_R(c_1)$ de I_{2n}

$$\psi_R(c_1)(x^1, x^0) = (c_1(x^1) \oplus x^0, c_1(x^1))$$

Comme précédemment, on peut généraliser à r étages un schéma R. Celui-ci transforme alors r permutations $c_1, \cdots, c_r$ de I_n en une permutation de I_{2n} définie par $\psi_R(c_1, c_2, ..., c_r) = \psi_R(c_r) \circ \cdots \circ \psi_R(c_1)$.

Dans les sections suivantes, on considérera plutôt les versions simplifiées $\psi'_L(c_1, c_2, ..., c_r)$ et $\psi'_R(c_1, c_2, ..., c_r)$ de $\psi_L(c_1, c_2, ..., c_r)$ et $\psi_R(c_1, c_2, ..., c_r)$ qui ne contiennent pas le X-or et l'inversion gauche-droite du dernier étage afin d'alléger les démonstrations, cette simplification n'affectant pas le contenu des preuves.

Il est important de remarquer que $\psi'_R(c_1, c_2, ..., c_r)$ et $\psi'_L(c_r^{-1}, c_{r-1}^{-1}, ..., c_1^{-1})$ sont des transformations inverses l'une de l'autre. Cela va considérablement simplifier les preuves de sécurité dans le cas d'un distingueur super-pseudo-aléatoire.

3.2 Résultats Concernant le Schéma L

Nous allons dans cette section comparer une fonction f de $2n$ bits vers $2n$ bits construite à partir d'un nombre r d'étages variable du schéma L ($f = \psi_L(c_1^*, c_2^*, ..., c_r^*)$ est générée à partir de r permutations aléatoires parfaites de n bits vers n bits notées $c_1^*, c_2^*, ..., c_r^*$) à une fonction aléatoire parfaite f^* de $2n$ bits vers $2n$ bits ou d'une permutation aléatoire parfaite c^* de I_{2n}.

3.2.1 Version sur Trois Étages du Schéma L : $\psi_L(c_1^*, c_2^*, c_3^*)$ n'est Pas une Fonction Pseudo-Aléatoire

Comme démontré par Y. Zheng, T. Matsumoto et H. Imai dans [ZMI90], la fonction $f = \psi_L(c_1^*, c_2^*, c_3^*)$ associée à trois étages du schéma L n'est pas pseudo-aléatoire.

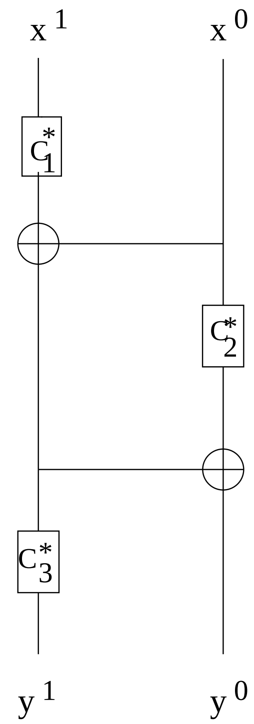


FIG. 3.2 – Version sur Trois Étages du Schéma L

En utilisant, les notations de la figure précédente qui représente la fonction $f = \psi'_L(c_1^*, c_2^*, c_3^*)$, il est facile de prouver ce résultat.

Il suffit de soumettre au chiffrement 4 entrées bien choisies pour distinguer f d'une fonction aléatoire parfaite f^* avec une très grande probabilité. On considère donc deux couples d'entrées (a, b) et (a', b) de $2n$ bits que l'on soumet au chiffrement. On note (c, d) et (c', d') les sorties correspondantes. On a alors $d \oplus d' = c_1^*(a) \oplus c_1^*(a')$, équation complètement indépendante de b. Il suffit alors de soumettre au chiffrement deux nouveaux couples (a, b') et (a', b') qui diffèrent uniquement sur la partie droite pour obtenir la même valeur de $d \oplus d'$. On peut donc, si on soumet au chiffrement 4 entrées judicieusement choisies, distinguer f d'une fonction aléatoire parfaite f^* de I_{2n} avec une probabilité proche de 1.

3.2.2 Version sur 4 Étages du Schéma L : $\psi_L(c_1^*, c_2^*, c_3^*, c_4^*)$ est une Fonction Pseudo-Aléatoire

La preuve que la version sur 4 étages du schéma L est pseudo-aléatoire a déjà été donnée par K. Sakurai et Y. Zheng dans [SZ97]. Cependant, afin d'établir une liste exhaustive des diverses propriétés des schémas L et R, nous redémontrons ce théorème énoncé ainsi :

Théorème 27 *Soit n un entier, c_1^*, c_2^*, c_3^*, c_4^* quatre permutations aléatoires parfaites de I_n dans I_n indépendantes et soit f^* une fonction aléatoire de distribution uniforme sur l'ensemble I_{2n}. On note $f = \psi_L(c_1^*, c_2^*, c_3^*, c_4^*)$ la permutation aléatoire associée à quatre étages d'un schéma de type L. Pour tout distingueur adaptatif $\mathcal{A}$ faisant appel à q requêtes,* *on a :*

$$Adv^q_{\mathcal{A}}(f, f^*) \leq \frac{7}{2} q^2 2^{-n}$$

Preuve :

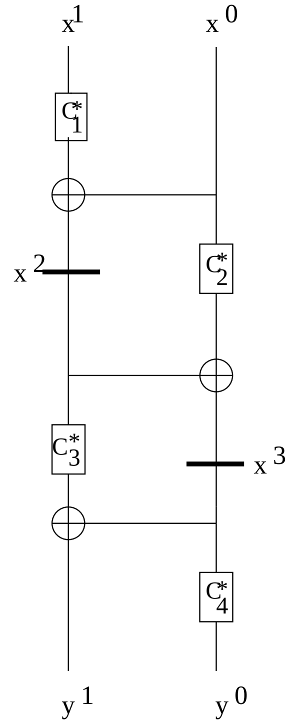


FIG. 3.3 – 4 Étages d'un Schéma L

on compare ici la permutation $f = \psi'_L(c_1^*, c_2^*, c_3^*, c_4^*)$ de la figure précédente (qui possède exactement les mêmes propriétés dans le modèle de sécurité de Luby et Rackoff que la version complète de $\psi_L(c_1^*, c_2^*, c_3^*, c_4^*)$) à une fonction aléatoire parfaite f^*.

Cette preuve est très proche de celle qui sera développée dans la section 3.3.1.

Introduisons tout d'abord quelques notations :

- On note $X = (X_i)_{i \in [1..q]} = (x_i^1, x_i^0)$ le q-uplet d'entrées de la fonction f où chaque X_i appartient à I_{2n}.
- On note également $Y = (Y_i)_{i \in [1..q]} = (y_i^1, y_i^0)_{i \in [1..q]}$ le q-uplet correspondant de sorties de la fonction f.
- Pour chaque calcul de $(y_i^1, y_i^0) = \psi'_L(c_1^*, c_2^*, c_3^*, c_4^*)(x_i^1, x_i^0)$, on note x_i^2 et x_i^3 les valeurs intermédiaires dont la position est précisée sur la figure 3.3 définies par :
$$x_i^2 = c_1^*(x_i^1) \oplus x_i^0$$
$$x_i^3 = c_2^*(x_i^0) \oplus x_i^2$$
- On note $(x_i^0)_{i \in [1..q]}$, $(x_i^1)_{i \in [1..q]}$, $(x_i^2)_{i \in [1..q]}$, $(x_i^3)_{i \in [1..q]}$, $(y_i^0)_{i \in [1..q]}$ et $(y_i^1)_{i \in [1..q]}$ les q-uplets de mots de n bits correspondants aux séries de valeurs liées à x^0, x^1, x^2, x^3, y^0, y^1 respectivement.
- On définit finalement l'ensemble $\mathcal{X}$ comme l'ensemble des q-uplets X composés d'éléments de I_{2n} deux à deux distincts (c'est à dire tels que pour tout i dans $[1..q]$

et tout j différent de i dans $[1..q]$, $x_i^1 \neq x_j^1$ ou $x_i^0 \neq x_j^0$) et l'ensemble $\mathcal{Y}$ comme l'ensemble des q-uplets Y de I_{2n} tel que les éléments de I_n composant le q-uplet y^1 correspondant soient deux à deux distincts : $\mathcal{Y} = \{(Y_1, \cdots, Y_q) \in (I^{2n})^q \ / \ y^1 \in I^{\neq},\ y^0 \in I\}$.

Nous cherchons alors une borne inférieure pour la taille de $\mathcal{Y}$ et pour la probabilité $\Pr[X \overset{f}{\mapsto} Y]$, probabilité de transition associée à un q-uplet X de $\mathcal{X}$ et à un q-uplet Y de $\mathcal{Y}$ afin d'être dans les conditions du théorème 11 de Patarin. On va donc chercher à montrer qu'il existe deux nombres réels ϵ_1 et ϵ_2 qui satisfont ces hypothèses.

Commençons par estimer $|\mathcal{Y}|$:

$$\begin{aligned} |\mathcal{Y}| &= |I^{2n}|^q \cdot (1 - \Pr[y^1 \notin I^{\neq}]) \\ &\geq |I^{2n}|^q (1 - \sum_{i,j \in [1..q], i \neq j} \Pr[y_i^1 = y_j^1]) \\ &\geq |I^{2n}|^q (1 - \frac{q(q-1)}{2}.2^{-n}) \end{aligned}$$

On peut donc prendre $\epsilon_1 = \frac{q(q-1)}{2.2^n}$.
À présent, pour tout q-uplet X de l'ensemble $\mathcal{X}$ et pour tout q-uplet Y de l'ensemble $\mathcal{Y}$, on cherche à établir une borne inférieure pour la probabilité $\Pr[X \overset{f}{\mapsto} Y]$:

$$\begin{aligned} \Pr[X \overset{f}{\mapsto} Y] &= \textstyle\sum_{x^2, x^3, x^3 \oplus y^1 \in I} && \Pr[(c_1^*(x^1) \oplus x^0 = x^2) \wedge (x^3 = c_2^*(x^0) \oplus x^2) \\ & && \wedge (c_3^*(x^2) \oplus x^3 = y^1) \wedge (c_4^*(x^3) = y^0)] \\ &\geq \textstyle\sum_{x^2, x^3, x^3 \oplus y^1 \in I^{\neq}} && \Pr[(c_1^*(x^1) \oplus x^0 = x^2) \wedge (x^3 = c_2^*(x^0) \oplus x^2)] \\ & && \cdot \Pr[(c_3^*(x^2) \oplus x^3 = y^1) \wedge (c_4^*(x^3) = y^0)] \quad (1) \end{aligned}$$

On cherche à présent à estimer la partie droite l'inégalité (1).
Pour cela, pour tout x^2, x^3, $x^3 \oplus y^1$ q-uplets de $I^{\neq}$, on calcule tout d'abord $\Pr[(c_3^*(x^2) \oplus x^3 = y^1) \wedge (c_4^*(x^3) = y^0)] = \Pr[(c_3^*(x^2) \oplus x^3 = y^1)] \cdot \Pr[(c_4^*(x^3) = y^0)]$. Comme x^2 et x^3 appartiennent à $I^{\neq}$, on peut alors appliquer la propriété 3 vue dans le chapitre 1 de cette partie concernant les permutations aléatoires, on a : $\Pr[(c_3^*(x^2) \oplus x^3 = y^1)] = \frac{(2^n - q)!}{2^n!}$. Pour les mêmes raisons, on a également $\Pr[(c_4^*(x^3) = y^0)] = \frac{(2^n - q)!}{2^n!}$. On en déduit donc que :

$$\Pr[(c_3^*(x^2) \oplus x^3 = y^1) \wedge (c_4^*(x^3) = y^0)] = \left(\frac{(2^n - q)!}{2^n!}\right)^2 \geq \frac{1}{2^{2nq}}$$

car on a bien $\left(\frac{(2^n - q)!}{2^n!}\right)^2 \geq \frac{1}{2^{2nq}}$.

L'inégalité (1) devient donc :

$$\Pr[X \overset{f}{\mapsto} Y] \geq \frac{1}{2^{2nq}} \cdot \sum_{x^2, x^3, x^3 \oplus y^1 \in I^{\neq}} \Pr[(c_1^*(x^1) \oplus x^0 = x^2) \wedge (x^3 = c_2^*(x^0) \oplus x^2)] \quad (i)$$

On cherche donc à estimer :

$$B = \sum_{x^2,x^3,x^3\oplus y^1\in I^{\neq}} \Pr[(c_1^*(x^1)\oplus x^0 = x^2)\wedge(c_2^*(x^0)\oplus x^2 = x^3)]$$

On a :

$$\begin{aligned}
B &= \Pr[(c_1^*(x^1)\oplus x^0)\in I^{\neq}\wedge(c_2^*(x^0)\oplus c_1^*(x^1)\oplus x^0)\in I^{\neq}\wedge(x^3\oplus y^1)\in I^{\neq}] \\
&= 1-\Pr\begin{bmatrix} (c_1^*(x^1)\oplus x^0\in I^{=}) \\ \vee(c_2^*(x^0)\oplus c_1^*(x^1)\oplus x^0)\in I^{=} \\ \vee(x^3\oplus y^1)\in I^{=} \end{bmatrix} \\
&\geq 1-\sum_{i,j,i\neq j}\Pr[c_1^*(x_i^1)\oplus x_i^0 = c_1^*(x_j^1)\oplus x_j^0] \\
&\quad -\sum_{i,j,i\neq j}\Pr[c_2^*(x_i^0)\oplus x_i^2 = c_2^*(x_j^0)\oplus x_j^2] \\
&\quad -\sum_{i,j,i\neq j}\Pr[x_i^3\oplus y_i^1 = x_j^3\oplus y_j^1]
\end{aligned}$$

On cherche donc à calculer $\Pr[c_1^*(x_i^1)\oplus x_i^0 = c_1^*(x_j^1)\oplus x_j^0]$, $\Pr[c_2^*(x_i^0)\oplus x_i^2 = c_2^*(x_j^0)\oplus x_j^2]$ et $\Pr[x_i^3\oplus y_i^1 = x_j^3\oplus y_j^1]$. Intéressons nous tout d'abord à $\Pr[c_1^*(x_i^1)\oplus x_i^0 = c_1^*(x_j^1)\oplus x_j^0]$. Grâce à la propriété 4 de la section 1.1 de cette partie et si $x_i^1\neq x_j^1$, étant donné une différence δ fixée (ici égale à $x_i^0\oplus x_j^0$), on a $\Pr[c_1^*(x_i^1)\oplus c_1^*(x_j^1)=\delta]\leq\frac{2}{2^n}$. D'autre part, si $x_i^1 = x_j^1$, alors $x_i^0\neq x_j^0$, et donc $\Pr[c_1^*(x_i^1)\oplus x_i^0 = c_1^*(x_j^1)\oplus x_j^0]=0$. Donc dans tous les cas, $\Pr[c_1^*(x_i^1)\oplus x_i^0 = c_1^*(x_j^1)\oplus x_j^0]\leq\frac{2}{2^n}$. Il ne reste plus qu'à appliquer cette propriété aux $\frac{q(q-1)}{2}$ paires d'indices (i,j) de $[1..q]$ avec $i\neq j$, on obtient alors $\sum_{i,j}\Pr[c_1^*(x_i^1)\oplus x_i^0 = c_1^*(x_j^1)\oplus x_j^0]\leq\frac{q(q-1)}{2^n}$. Pour les mêmes raisons, on a : $\sum_{i,j,i\neq j}\Pr[c_2^*(x_i^0)\oplus x_i^2 = c_2^*(x_j^0)\oplus x_j^2]\leq\frac{q(q-1)}{2^n}$ et $\sum_{i,j,i\neq j}\Pr[x_i^3\oplus y_i^1 = x_j^3\oplus y_j^1]\leq\frac{q(q-1)}{2^n}$. On obtient donc :

$$B\geq 1-\frac{3.q(q-1)}{2^n}\quad (ii)$$

en associant les inégalités (i) et (ii), on obtient :

$$\Pr[X\overset{f}{\mapsto}Y]\geq\left(1-\frac{3q(q-1)}{2^n}\right)\cdot\frac{1}{2^{2nq}}$$

En remarquant que $Pr[X\overset{f^*}{\mapsto}Y]=\frac{1}{2^{2nq}}$, on peut appliquer le théorème 11 de Patarin avec $\epsilon_1=\frac{q(q-1)}{2|I^n|}$ et $\epsilon_2=\frac{3q(q-1)}{|I^n|}$. On obtient donc :

$$\begin{gathered}
\mathrm{Adv}_{\mathcal{A}}^q(f,f^*)\leq\frac{7q(q-1)}{2.2^n} \\
\mathrm{Adv}_{\mathcal{A}}^q(f,f^*)\leq\frac{7q^2}{2.2^n}
\end{gathered}$$

□

3.2.3 Version sur 4 Étages du Schéma L : $\psi_L(c_1^*,c_2^*,c_3^*,c_4^*)$ n'est Pas une Permutation Super-Pseudo-Aléatoire

Comme l'ont remarqué K. Sakurai et Y. Zheng dans [SZ97], la version sur 4 étages d'un schéma de type L n'est pas une permutation super-pseudo-aléatoire. Il est en effet possible de distinguer $\psi_L(c_1^*, c_2^*, c_3^*, c_4^*)$ d'une permutation parfaitement aléatoire c^* à l'aide d'un petit nombre de requêtes de chiffrement et de déchiffrement.

Nous ne redonnerons pas ici de preuve de cette propriété car elle est une conséquence directe de la section 3.3.2 où nous démontrerons que $\psi_R(c_1^*, c_2^*, c_3^*, c_4^*)$ n'est pas une permutation super-pseudo-aléatoire. En effet, les permutations $\psi'_R(c_1, c_2, ..., c_r)$ et $\psi'_L(c_r^{-1}, c_{r-1}^{-1}, ..., c_1^{-1})$ étant inverses l'une de l'autre, le distingueur super-pseudo-aléatoire pour $\psi_R(c_1^*, c_2^*, c_3^*, c_4^*)$ peut être transformé en un distingueur pour $\psi_L(c_1^*, c_2^*, c_3^*, c_4^*)$ utilisant le même nombre de requêtes et ayant le même avantage.

3.2.4 Version sur 5 Étages du Schéma L : $\psi_L(c_1^*, c_2^*, c_3^*, c_4^*, c_5^*)$ est une Permutation Super-Pseudo-Aléatoire

Rappelons tout d'abord qu'un distingueur super-pseudo-aléatoire est un distingueur adaptatif qui peut faire appel à la fois au chiffrement c et à son inverse c^{-1}. Nous allons démontrer le théorème suivant afin de prouver que la permutation $\psi_L(c_1^*, c_2^*, c_3^*, c_4^*, c_5^*)$ est super-pseudo-aléatoire.

Théorème 28 *Soit n un entier, c_1^*, c_2^*, c_3^*, c_4^*, c_5^* cinq permutations aléatoires parfaites de I_n dans I_n indépendantes et soit c^* une permutation aléatoire de distribution uniforme sur l'ensemble I_{2n}. On note $c = \psi_L(c_1^*, c_2^*, c_3^*, c_4^*, c_5^*)$ la permutation aléatoire associée à cinq étages d'un schéma de type L. Pour tout distingueur super-pseudo-aléatoire $\mathcal{A}$ faisant appel à q requêtes, on a :*

$$Adv_{\mathcal{A}}^q(c, c^*) \leq \frac{9}{2} \cdot \frac{q^2}{2^n}$$

Afin de démontrer ce théorème, nous avons besoin d'une variante du théorème de Patarin (théorème 11) due également à Patarin énoncée dans [Pat91] portant sur les permutations (que nous donnons ici sans preuve) :

Lemme 4 *Soit m un entier, ϵ un nombre réel positif, c une permutation aléatoire de l'ensemble $\{0,1\}^m$, c^* une permutation aléatoire parfaite sur le même ensemble. On note $\mathcal{X}$ le sous-ensemble des q-uplets $(X_1, \cdots, X_q)$ d'éléments deux à deux distincts. Soit $\mathcal{A}$ un distingueur super-pseudo-aléatoire faisant appel à q requêtes.*
Si $Pr[X \overset{c}{\mapsto} Y] \geq (1-\epsilon) \cdot \frac{1}{|I_m|^q}$ pour tout X et Y q-uplets de $\mathcal{X}$ alors

$$Adv_{\mathcal{A}}^q(c, c^*) \leq \epsilon + \frac{q(q-1)}{2 \cdot 2^m}.$$

Preuve du Théorème :

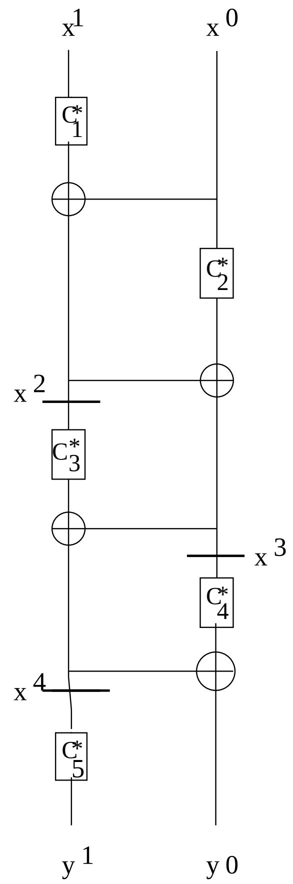


FIG. 3.4 – Cinq Tours d'un Schéma L

Le but de cette démonstration est de comparer la permutation $c = \psi'_L(c_1^*, c_2^*, c_3^*, c_4^*, c_5^*)$ de la figure précédente (dont les propriétés concernant l'étude d'un distingueur super-pseudo-aléatoire sont exactement les mêmes que celles de la version complète de la permutation $\psi_L(c_1^*, c_2^*, c_3^*, c_4^*, c_5^*)$) avec une permutation aléatoire parfaite c^* de I_{2n}. On introduit donc un certain nombre de notations :

- Soit $X = (x_i^1, x_i^0) \in \mathcal{X}$ un q-uplet de valeurs de I_{2n} deux à deux distinctes.
- Soit $Y = (y_i^1, y_i^0)$ un q-uplet de valeurs de I_{2n} deux à deux distinctes.
- On note $x^2 = (x_i^2)_{i=1..q}$, $x^3 = (x_i^3)_{i=1..q}$ et $x^4 = (x_i^4)_{i=1..q}$ les q-uplets de I_n de valeurs intermédiaires induites par les q calculs de la permutation c dont l'emplacement est précisé sur la figure précédente.

On cherche alors une borne inférieure pour la probabilité $\Pr[X \overset{c}{\mapsto} Y]$ afin d'appliquer le lemme de Patarin concernant les permutations. On a :

$$\begin{array}{lll} \Pr[X \overset{c}{\mapsto} Y] & = \sum_{x^2,x^3,x^4} & \Pr[(c_1^*(x^1) \oplus x^0 = x^2) \wedge (c_2^*(x^0) \oplus x^2 = x^3) \\ & & \wedge (c_3^*(x^2) \oplus x^3 = x^4) \\ & & \wedge (c_4^*(x^3) \oplus x^4 = y^0) \wedge (c_5^*(x^4) = y^1)] \\ & \geq \sum_{x^2,x^3 \in I^{\neq}} & \Pr[(c_1^*(x^1) \oplus x^0 = x^2) \wedge (c_2^*(x^0) \oplus x^2 = x^3)] \\ & \cdot \sum_{x^4} & \Pr\left[\begin{array}{c} (c_3^*(x^2) \oplus x^3 = x^4) \wedge \\ (c_4^*(x^3) \oplus x^4 = y^0) \wedge (c_5^*(x^4) = y^1) \end{array}\right] \quad (1) \end{array}$$

On cherche tout d'abord à établir une borne inférieure pour le calcul de $\sum_{x^4} \Pr[(c_3^*(x^2) \oplus x^3 = x^4) \wedge (c_4^*(x^3) \oplus x^4 = y^0) \wedge (c_5^*(x^4) = y^1)]$, x^2 et x^3 étant supposés fixés dans $I^{\neq}$. On définit pour cela l'ensemble Z formé de q-uplets x^4 :

$$Z = \{x^4 | x^4 \sim y^1 \wedge x^4 \oplus y^0 \in I^{\neq} \wedge x^4 \oplus x^3 \in I^{\neq}\}$$

où $x^4 \sim y^1$ signifie que $\forall i, j$ $x_i^4 = x_j^4$ si et seulement si $y_i^1 = y_j^1$. On note $q_1 \leq q$, le nombre de valeurs de y_i^1 distincts. Il existe alors une série d'indices $i_1, \cdots, i_{q_1}$ telle que $y_{i_1}^1, \cdots, y_{i_{q_1}}^1$ soient deux à deux distincts. Chaque indice $i_k \in \{i_1, \cdots, i_{q_1}\}$ détermine une classe telle que pour tout élément i de cette classe, on ait $y_i^1 = y_{i_k}^1$. Alors, $\forall i \in [1, .., q], \exists ! \ i_k \in \{i_1, \cdots, i_{q_1}\} / y_i^1 = y_{i_k}^1, Cl(i) =_{def} i_k$.

Il existe $\alpha = \frac{2^n!}{(2^n - q_1)!}$ valeurs x^4 tels que $x^4 \sim y^1$ (le nombre de x^4 vérifiant cette propriété est en fait entièrement déterminée par le nombre q_1 de valeurs distincts). On en déduit donc :

$$\begin{array}{rl} |Z| & \geq |\{x^4 | x^4 \sim y^1\}| - |\{x^4 | x^4 \sim y^1 \wedge x^4 \oplus y^0 \notin I^{\neq}\}| \\ & \quad - |\{x^4 | x^4 \sim y^1 \wedge x^4 \oplus x^3 \notin I^{\neq}\}| \\ & \geq |\{x^4 | x^4 \sim y^1\}| - \displaystyle\sum_{i \neq j} |\{x^4 | x^4 \sim y^1 \wedge x_i^4 \oplus x_j^4 = y_i^0 \oplus y_j^0\}| \\ & \quad - \displaystyle\sum_{i \neq j} |\{x^4 | x^4 \sim y^1 \wedge x_i^4 \oplus x_j^4 = x_i^3 \oplus x_j^3\}| \end{array}$$

Étant donné $i \neq j$, on peut donner une borne supérieure à la taille de l'ensemble S_{ij} défini par $S_{ij} = \{x^4 | x^4 \sim y^1 \wedge x_i^4 \oplus x_j^4 = y_i^0 \oplus y_j^0\}$. Cette borne vaut $\frac{2\alpha}{2^n}$.
En effet :

- Si $y_i^1 = y_j^1$ alors $y_i^0 \oplus y_j^0 \neq 0$ car sinon (y_i^1, y_i^0) sera égal à $(y_j^1, y_j^0))$ alors que $x^4 \sim y^1$ implique que $x_i^4 = x_j^4$ alors que $x_i^4 \oplus x_j^4$ ne peut pas être égal à $y_i^0 \oplus y_j^0$. Donc, $|S_{ij}| = 0$.
- $y_i^1 \neq y_j^1$, alors si $x^4 \in S_{ij}$, $x_{Cl(j)}^4$ est entièrement déterminé par $x_{Cl(i)}^4$ puisque $x_{Cl(j)}^4 = x_{Cl(i)}^4 \oplus y_i^0 \oplus y_j^0$. Donc $|S_{ij}|$ contient au plus $2^n(2^n - 2) \cdots (2^n - q_1) = \frac{\alpha}{2^n - 1} \leq \frac{2\alpha}{2^n}$ éléments.

De la même manière, en utilisant le fait que x_i^3 est différent de x_j^3, on peut donner une borne supérieure à la taille de l'ensemble $\{x^4 | x^4 \sim y^1 \wedge x_i^4 \oplus x_j^4 = x_i^3 \oplus x_j^3\}|$. Cette borne

vaut $\frac{2\alpha}{2^n}$.
On a donc :

$$|Z| \geq \alpha \left[1 - \frac{q(q-1)}{2}\frac{2}{2^n} - \frac{q(q-1)}{2}\frac{2}{2^n}\right] \text{ i.e. } |Z| \geq \frac{2^n!}{(2^n-q_1)!}\left[1 - \frac{2q^2}{2^n}\right]$$

On a également :

$$\sum_{x^4} \Pr\left[\begin{array}{c}(c_3^*(x^2) \oplus x^3 = x^4) \wedge \\ (c_4^*(x^3) \oplus x^4 = y^0) \wedge (c_5^*(x^4) = y^1)\end{array}\right]$$
$$\geq \sum_{x^4 \in Z} \Pr\left[\begin{array}{c}(c_3^*(x^2) \oplus x^3 = x^4) \wedge \\ (c_4^*(x^3) \oplus x^4 = y^0) \wedge (c_5^*(x^4) = y^1)\end{array}\right]$$

Or, pour tout x^4 appartenant à Z, on a $\Pr[c_5^*(x^4) = y^1] = \frac{(2^n-q_1)!}{2^n!} = \frac{1}{\alpha}$ et $\Pr[(c_3^*(x^2) = x^4 \oplus x^3)] = \frac{(2^n-q)!}{2^n!}$ d'après la propriété 3 de la section 1.1 de cette partie car x_i^2 et $x_i^4 \oplus x_i^3$ sont bien deux à deux distincts. On a également, pour les mêmes raisons, $\Pr[(c_3^*(x^2) = x^4 \oplus x^3)] = \frac{(2^n-q)!}{2^n!}$. On obtient donc les inégalités suivantes :
$\sum_{x^4} \Pr[(c_3^*(x^2) \oplus x^3 = x^4) \wedge (c_4^*(x^3) \oplus x^4 = y^0) \wedge (c_5^*(x^4) = y^1)]$

$$\begin{aligned}
&\geq |Z| \cdot \left(\frac{(2^n-q)!}{2^n!}\right)^2 \cdot \frac{1}{\alpha} \\
&\geq \alpha \cdot \left(1 - \frac{2q^2}{2^n}\right)\left(\frac{(2^n-q)!}{2^n!}\right)^2 \cdot \frac{1}{\alpha} \\
&\geq \left(1 - \frac{2q^2}{2^n}\right)\left(\frac{(2^n-q)!}{2^n!}\right)^2
\end{aligned}$$

L'inégalité (1) devient donc :

$$\Pr[X \overset{c}{\mapsto} Y] \geq \sum_{x^2, x^3 \in I^{\neq}} \Pr\left[\begin{array}{c}(c_1^*(x^1) \oplus x^0 = x^2) \\ \wedge \\ (c_2^*(x^0) \oplus x^2 = x^3)\end{array}\right] \cdot \left(1 - \frac{2q^2}{2^n}\right)\left(\frac{(2^n-q)!}{2^n!}\right)^2 \quad (i)$$

Il ne reste alors plus qu'à établir une borne inférieure pour la quantité

$$\begin{aligned}
B &= \sum_{x^2, x^3 \in I^{\neq}} \Pr[(c_1^*(x^1) \oplus x^0 = x^2) \wedge (c_2^*(x^0) \oplus x^2 = x^3)] \\
&= \Pr[x^2 \in I^{\neq} \wedge x^3 \in I^{\neq}] \\
&\quad \cdot \Pr[(c_1^*(x^1) \oplus x^0) \in I^{\neq} \wedge (c_2^*(x^0) \oplus x^2) \in I^{\neq} | x^2 \in I^{\neq}]
\end{aligned}$$

Or $\Pr[(c_1^*(x^1) \oplus x^0) \in I^{\neq}] \geq 1 - \sum_{i \neq j} \Pr[c_1^*(x_i^1) \oplus c_1^*(x_j^1) = x_i^0 \oplus x_j^0]$.
De plus, il est facile d'établir que pour tous indices i et j distincts, $\Pr[c_1^*(x_i^1) \oplus c_1^*(x_j^1) = x_i^0 \oplus x_j^0] \leq \frac{1}{2^n-1} \leq \frac{2}{2^n}$ car $(x_i^1, x_i^0) \neq (x_j^1, x_j^0)$ et $\Pr[c^*(x) \oplus c^*(x') = \delta] \leq \frac{2}{2^n}$ pour une valeur δ de I_n donnée. On a donc $\Pr[(c_1^*(x^1) \oplus x^0) \in I^{\neq}] \geq 1 - \frac{q(q-1)}{2} \cdot \frac{2}{2^n} \geq 1 - \frac{q^2}{2^n}$.
Pour les mêmes raisons, on a $\Pr[(c_2^*(x^0) \oplus x^2) \in I^{\neq} | x^2 \in I^{\neq}] \geq 1 - \frac{q^2}{2^n}$.
Donc

$$B \geq \left(1 - \frac{q^2}{2^n}\right)^2 \geq 1 - \frac{2q^2}{2^n} \ (ii).$$

On obtient donc en combinant (i) et (ii) :

$$\Pr[X \overset{c}{\mapsto} Y] \geq \left(1 - \frac{2q^2}{2^n}\right)^2 \left(\frac{(2^n-q)!}{2^n!}\right)^2$$

$$\text{Or, } \left(\frac{(2^n-q)!}{2^n!}\right)^2 \geq \frac{1}{2^{2nq}} \text{ et } \left(1 - \frac{2q^2}{2^n}\right)^2 \geq 1 - \frac{4q^2}{2^n}, \text{ donc}$$

$$\Pr[X \overset{c}{\mapsto} Y] \geq \left(1 - \frac{4q^2}{2^n}\right) \cdot \frac{1}{2^{2nq}}$$

On peut à présent appliquer le lemme précédent avec $\epsilon = \frac{4q^2}{2^n}$, on obtient alors :

$$\begin{array}{r}
\mathrm{Adv}^q_{\mathcal{A}}(c, c^*) \leq \frac{4q^2}{2^n} + \frac{q(q-1)}{2 \cdot 2^{2n}} \\
\mathrm{Adv}^q_{\mathcal{A}}(c, c^*) \leq \frac{9}{2} \cdot \frac{q^2}{2^n}
\end{array}$$

□

3.3 Résultats Concernant le Schéma R

Nous venons de voir des résultats concernant des preuves de sécurité pour des distingueurs pseudo-aléatoires ou super-pseudo-aléatoires concernant la permutation $f = \psi_L(c_1^*, c_2^*, ..., c_r^*)$ pour des r valant 3, 4 ou 5. Nous allons à présent nous intéresser au cas d'un schéma du type R sur r étages, $f = \psi_R(c_1^*, c_2^*, ..., c_r^*)$ représentant une permutation de $2n$ bits déduite de r permutations aléatoires parfaites de n bits vers n bits notées $c_1^*, c_2^*, ..., c_r^*$ en le comparant à une fonction aléatoire parfaite f^* de l'ensemble I_{2n}.

3.3.1 Version sur Trois Tours du Schéma R

Nous allons tout d'abord démontrer le théorème suivant :

Théorème 29 *Soit n un entier, c_1^*, c_2^*, c_3^* trois permutations aléatoires parfaites de I_n dans I_n indépendantes et soit f^* une fonction aléatoire de distribution uniforme sur l'ensemble I_{2n}. On note $f = \psi_R(c_1^*, c_2^*, c_3^*)$ la permutation aléatoire associée à trois étages d'un schéma de type R. Pour tout distingueur adaptatif $\mathcal{A}$ faisant appel à q requêtes, on a :*

$$Adv^q_{\mathcal{A}}(f, f^*) \leq 3q^2 2^{-n}$$

Preuve :

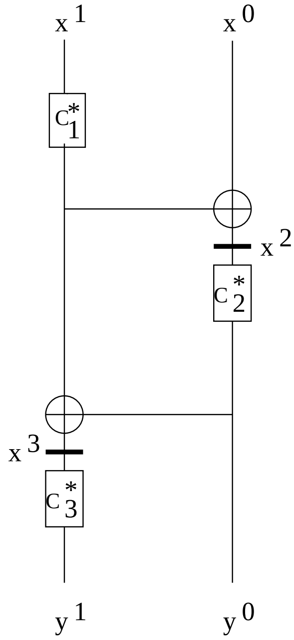


FIG. 3.5 – Trois Étages d'un Schéma R

On cherche à comparer ici la permutation $f = \psi'_R(c_1^*, c_2^*, c_3^*)$ de la figure précédente (qui possède exactement les mêmes propriétés dans le modèle de sécurité à la Luby et Rackoff que la version complète de $\psi_R(c_1^*, c_2^*, c_3^*)$) à une fonction aléatoire parfaite f^*.

On introduit les notations suivantes :

- On note $X = (X_i)_{i\in[1..q]} = (x_i^1, x_i^0)$ le q-uplet d'entrées de la fonction f où chaque X_i appartient à I_{2n}.
- On note également $Y = (Y_i)_{i\in[1..q]} = (y_i^1, y_i^0)_{i\in[1..q]}$ le q-uplet correspondant de sorties de la fonction f.
- Pour chaque calcul de $(y_i^1, y_i^0) = \psi'_R(c_1^*, c_2^*, c_3^*)(x_i^1, x_i^0)$, on note x_i^2 et x_i^3 les valeurs intermédiaires dont la position est précisée sur la figure 5 définies par :
$$x_i^2 = c_1^*(x_i^1) \oplus x_i^0$$
$$x_i^3 = c_1^*(x_i^1) \oplus c_2^*(x_i^2) = y_i^0 \oplus c_1^*(x_i^1)$$
- On note $(x_i^0)_{i\in[1..q]}$, $(x_i^1)_{i\in[1..q]}$, $(x_i^2)_{i\in[1..q]}$, $(x_i^3)_{i\in[1..q]}$, $(y_i^0)_{i\in[1..q]}$ et $(y_i^1)_{i\in[1..q]}$ les q-uplets de mots de n bits correspondants aux séries de valeurs liées à x^0, x^1, x^2, x^3, y^0, y^1 respectivement.
- On définit finalement l'ensemble $\mathcal{X}$ comme l'ensemble des q-uplets X d'éléments de I_{2n} distincts deux à deux (c'est à dire tels que pour tout i dans $[1..q]$ et tout j différent de i dans $[1..q]$, $x_i^1 \neq x_j^1$ ou $x_i^0 \neq x_j^0$) et l'ensemble $\mathcal{Y}$ comme l'ensemble des q-uplets Y de I_{2n} tel que les valeurs de I_n du q-uplet y^1 induit soient deux à deux distinctes et que le q-uplet induit y^0 soit également constitué de valeurs de I_n deux à deux distinctes : $\mathcal{Y} = \{(Y_1, \cdots, Y_q) \in (I_{2n})^q \ / \ y^1 \in I^{\neq},\ y^0 \in I^{\neq}\}$.

Nous cherchons alors une borne inférieure sur la taille de $\mathcal{Y}$ et sur la probabilité $\Pr[X \overset{f}{\mapsto} Y]$, probabilité de transition associée à un q-uplet X de $\mathcal{X}$ et à un q-uplet Y de $\mathcal{Y}$ afin d'être dans les conditions du théorème 11 de Patarin. On cherche à montrer qu'il existe deux nombres réels ϵ_1 et ϵ_2 qui satisfont ces hypothèses.

Calculons tout d'abord $|\mathcal{Y}|$:

$$\begin{array}{rcl} |\mathcal{Y}| & = & |I^{2n}|^q \cdot (1 - \Pr[y^1 \notin I^{\neq} \vee y^0 \notin I^{\neq}]) \\ & \geq & |I^{2n}|^q (1 - \sum_{i,j \in [1..q], i \neq j} \Pr[y_i^1 = y_j^1] - \sum_{i,j \in [1..q], i \neq j} \Pr[y_i^0 = y_j^0]) \\ & \geq & |I^{2n}|^q (1 - \frac{q(q-1)}{2} \cdot 2^{-n} - \frac{q(q-1)}{2} \cdot 2^{-n}) \\ & \geq & |I^{2n}|^q (1 - \frac{q(q-1)}{2^n}) \end{array}$$

On peut donc prendre $\epsilon_1 = \frac{q(q-1)}{2^n}$.
À présent, pour tout q-uplet X de l'ensemble $\mathcal{X}$ et pour tout q-uplet Y de l'ensemble $\mathcal{Y}$, on cherche à établir une borne inférieure pour la probabilité $\Pr[X \overset{f}{\mapsto} Y]$:

$$\begin{array}{rl} \Pr[X \overset{f}{\mapsto} Y] & = \sum_{x^2, x^3 \in I} \Pr[(c_1^*(x^1) \oplus x^0 = x^2) \wedge (c_1^*(x^1) \oplus y^0 = x^3) \\ & \quad \wedge (c_2^*(x^2) = y^0) \wedge (c_3^*(x^3) = y^1)] \\ & \geq \sum_{x^2, x^3 \in I^{\neq}} \Pr[(c_1^*(x^1) \oplus x^0 = x^2) \wedge (c_1^*(x^1) \oplus y^0 = x^3)] \\ & \quad \cdot \Pr[(c_2^*(x^2) = y^0) \wedge (c_3^*(x^3) = y^1)] \quad (1) \end{array}$$

On cherche tout d'abord à calculer pour tout q-uplet x^2 de $I^{\neq}$ et pour tout q-uplet x^3 de $I^{\neq}$, $\Pr[(c_2^*(x^2) = y^0) \wedge (c_3^*(x^3) = y^1)] = \Pr[(c_2^*(x^2) = y^0)] \cdot \Pr[(c_3^*(x^3) = y^1)]$. Puisque x^2 et y^0 appartiennent tous les deux à $I^{\neq}$, on peut appliquer la propriété 3 de la section 1.1 de cette partie concernant les permutations aléatoires, on a : $\Pr[(c_2^*(x^2) = y^0)] = \frac{(2^n-q)!}{2^n!}$.
Pour les mêmes raisons, on a également $\Pr[(c_3^*(x^3) = y^1)] = \frac{(2^n-q)!}{2^n!}$.
On en déduit donc que :

$$\Pr[(c_2^*(x^2) = y^0) \wedge (c_3^*(x^3) = y^1)] = \left(\frac{(2^n-q)!}{2^n!}\right)^2 \geq \frac{1}{2^{2nq}}$$

car on a bien $\left(\frac{(2^n-q)!}{2^n!}\right)^2 \geq \frac{1}{2^{2nq}}$.
L'inégalité (1) devient donc :

$$\Pr[X \overset{f}{\mapsto} Y] \geq \frac{1}{2^{2nq}} \cdot \sum_{x^2, x^3 \in I^{\neq}} \Pr[(c_1^*(x^1) \oplus x^0 = x^2) \wedge (c_1^*(x^1) \oplus y^0 = x^3)] \quad (i)$$

Il ne reste alors plus qu'à estimer

$$B = \sum_{x^2, x^3 \in I^{\neq}} \Pr[(c_1^*(x^1) \oplus x^0 = x^2) \wedge (c_1^*(x^1) \oplus y^0 = x^3)]$$

On a :

$$\begin{array}{rcl} B & = & \Pr[(c_1^*(x^1) \oplus x^0) \in I^{\neq} \wedge (c_1^*(x^1) \oplus y^0) \in I^{\neq}] \\ & = & 1 - \Pr[(c_1^*(x^1) \oplus x^0) \notin I^{\neq} \vee (c_1^*(x^1) \oplus y^0) \notin I^{\neq}] \\ & \geq & 1 - \sum_{i,j,i \neq j} Pr[c_1^*(x_i^1) \oplus x_i^0 = c_1^*(x_j^1) \oplus x_j^0] \\ & & \quad - \sum_{i,j,i \neq j} Pr[c_1^*(x_i^1) \oplus y_i^0 = c_1^*(x_j^1) \oplus y_j^0] \end{array}$$

On cherche donc à calculer $\Pr[c_1^*(x_i^1)\oplus x_i^0 = c_1^*(x_j^1)\oplus x_j^0]$ et $\Pr[c_1^*(x_i^1)\oplus c_1^*(x_j^1) = x_i^0\oplus x_j^0]$. Grâce à la propriété 4 de la section 1.1 de cette partie et si $x_i^1 \neq x_j^1$, étant donné une différence δ fixée (ici égale à $x_i^0 \oplus x_j^0$), on a $\Pr[c_1^*(x_i^1) \oplus c_1^*(x_j^1) = \delta] \leq \frac{2}{2^n}$. D'autre part, si $x_i^1 = x_j^1$, alors $x_i^0 \neq x_j^0$, et donc $\Pr[c_1^*(x_i^1) \oplus x_i^0 = c_1^*(x_j^1) \oplus x_j^0] = 0$. Donc dans tous les cas, $\Pr[c_1^*(x_i^1)\oplus x_i^0 = c_1^*(x_j^1)\oplus x_j^0] \leq \frac{2}{2^n}$. Il ne reste plus qu'à appliquer cette propriété aux $\frac{q(q-1)}{2}$ paires d'indices (i,j) de $[1..q]$ avec $i \neq j$, on obtient alors $\sum_{i,j} \Pr[c_1^*(x_i^1) \oplus x_i^0 = c_1^*(x_j^1) \oplus x_j^0] \leq \frac{q(q-1)}{2^n}$. Pour les mêmes raisons, on a : $\sum_{i,j,i\neq j} \Pr[c_1^*(x_i^1) \oplus y_i^0 = c_1^*(x_j^1) \oplus y_j^0] \leq \frac{q(q-1)}{2^n}$. On obtient donc :

$$B \geq 1 - \frac{2q(q-1)}{2^n} \quad (ii)$$

En combinant les inégalités (i) et (ii), on obtient :

$$\Pr[X \overset{f}{\mapsto} Y] \geq \left(1 - \frac{2q(q-1)}{2^n}\right) \cdot \frac{1}{2^{2nq}}$$

Or $Pr[X \overset{f^*}{\mapsto} Y] = \frac{1}{2^{2nq}}$. On est donc bien dans les hypothèses du théorème 11 en prenant $\epsilon_1 = \frac{q(q-1)}{|I^n|}$ et $\epsilon_2 = \frac{2q(q-1)}{|I^n|}$. On obtient donc :

$$\mathrm{Adv}_{\mathcal{A}}^q(f, f^*) \leq \frac{3q(q-1)}{2^n}$$
$$\mathrm{Adv}_{\mathcal{A}}^q(f, f^*) \leq \frac{3q^2}{2^n}$$

□

3.3.2 Version sur 4 Étages du Schéma R : $\psi_R(c_1^*, c_2^*, c_3^*, c_4^*)$ n'est Pas une Permutation Super-Pseudo-Aléatoire

On se contentera ici d'étudier la permutation $f = \psi'_R(c_1^*, c_2^*, c_3^*, c_4^*)$, déduite de la permutation $\psi_R(c_1^*, c_2^*, c_3^*, c_4^*)$ par omission du X-or final. Cela ne change en rien les propriétés induites dans le distingueur super-pseudo-aléatoire. On représente la fonction f sur quatre étages en ajoutant un tour à la figure précédente.

Le but de ce paragraphe est de montrer qu'à l'aide de seulement deux textes clairs choisis et de deux chiffrés choisis, il est possible de distinguer f d'une permutation aléatoire parfaite c^* avec un très grande probabilité. Pour cela, on chiffre deux textes clairs de $2n$ bits (a, b) et (a, b'), ici représentés sous la forme (x^1, x^0), leurs parties gauches étant égales. On note (c, d) et (c', d') les deux chiffrés correspondants également représentés sous la forme (y^1, y^0). Il suffit alors d'échanger les parties gauches de ces deux chiffrés pour obtenir deux nouveaux chiffrés (c', d) et (c, d') qu'on soumet alors au déchiffrement (f^{-1}). On obtient deux nouveaux textes clairs (α, β) et (α', β') qui sont tels que $\alpha = \alpha'$. Cette propriété particulière ne peut se produire qu'extrêmement rarement si la fonction de chiffrement est une permutation aléatoire parfaite c^*. Elle permet donc de distinguer f d'une permutation aléatoire parfaite de I_{2n} avec une probabilité proche de 1.

3.3.3 Version sur 5 Étages du Schéma R : $\psi_R(c_1^*, c_2^*, c_3^*, c_4^*, c_5^*)$ est une Permutation Super-Pseudo-Aléatoire

Le théorème présenté dans cette section est déduit directement de l'étude menée à la section 3.2.4 concernant la version à 5 étages d'un schéma du type L. En effet, puisque $\psi'_R(c_1, c_2, c_3, c_4, c_5)$ et $\psi'_L(c_1^{-1}, c_2^{-1}, c_3^{-1}, c_4^{-1}, c_5^{-1})$ sont des permutations inverses l'une de l'autre, tout distingueur super-pseudo-aléatoire pour la fonction $\psi_R(c_1^*, c_2^*, c_3^*, c_4^*, c_5^*)$ peut être construit à partir d'un distingueur super-pseudo-aléatoire de $\psi_L(c_1^*, c_2^*, c_3^*, c_4^*, c_5^*)$ et ce avec le même nombre de requêtes de chiffrement et de déchiffrement et le même avantage. Ainsi, on a :

Théorème 30 *Soit n un entier, c_1^*, c_2^*, c_3^*, c_4^*, c_5^* cinq permutations aléatoires parfaites de I_n dans I_n indépendantes et soit c^* une permutation aléatoire de distribution uniforme sur l'ensemble I_{2n}. On note $c = \psi_R(c_1^*, c_2^*, c_3^*, c_4^*, c_5^*)$ la permutation aléatoire associée à cinq étages d'un schéma de type R. Pour tout distingueur super-pseudo-aléatoire $\mathcal{A}$ faisant appel à q requêtes, on a :*

$$Adv_{\mathcal{A}}^q(c, c^*) \leq \frac{9}{2} \cdot \frac{q^2}{2^n}$$

Conclusion

Nous avons abordé dans cette thèse deux des principaux aspects de la cryptologie symétrique. Certes, les cryptanalyses présentées dans la deuxième partie sont non pas des cryptanalyses génériques, mais des attaques dédiées. Notons cependant que celles-ci ont permis aux différents auteurs de "réparer" de façon efficace leurs algorithmes. Je pense notamment ici au changement des boîtes S de la partie non linéaire de Crypton [Lim99] et aux nouveaux paramètres choisis pour SFLASH proposés au comité NESSIE en octobre 2001 [Nes01b]. La cryptanalyse de l'AES répond également au "défi" proposé par les auteurs (voir page 67) et utilise une faiblesse de l'AES ignorée jusqu'ici même si cette attaque, comme celle sur la version à huit étages de Crypton, est loin de mettre en danger réel l'algorithme.

La troisième partie de cette thèse, de facture plus classique et concernant des sujets qui ont pu paraître plus difficiles d'accès, reste cependant indispensable à la compréhension globale de ce qu'est la cryptographie symétrique. L'apport de nouvelles preuves de sécurité est une voie toujours intéressante - même si celle-ci peut parfois paraître rébarbative - et indispensable pour comprendre quels sont les schémas et les modes de chiffrement les plus sûrs.

La piste de recherche la plus intéressante et la plus stimulante reste l'étude cryptanalytique de l'AES, défi lancé au monde de la cryptographie. De nouveaux résultats, trop récents pour avoir été intégrés à cette thèse, portant sur les équations quadratiques des boîtes S de l'AES [CP02] [MR02] et la résolution des systèmes qui en découlent, sont des axes de recherche très intéressants, échos directs du travail de construction de SFLASH présenté dans le troisième chapitre de la deuxième partie de cette thèse. Peut-on également combiner ces nouvelles équations avec les différentes cryptanalyses présentées dans le deuxième chapitre de la deuxième partie ?

Il reste encore beaucoup de sueur de cryptographe, de CPU et de mémoire d'ordinateur à dépenser.

Table des figures

Index

Bibliographie

[ABK98] R.J. Anderson, E. Biham, et L.R. Knudsen. « Serpent : A Proposal for the Advanced Encryption Standard ». In *The First Advanced Encryption Standard Candidate Conference*. N.I.S.T., téléchargeable à l'adresse http ://www.cl.cam.ac.uk/ ~rja14/serpent.html, 1998.

[AES98] AES. « Advanced Encryption Standard ». N.I.S.T., http ://www.nist.gov/aes, 1998.

[AO97] K. Aoki et K. Ohta. « Strict Evaluation for the Maximum Average of Differential Probability and the Maximum Average of Linear Probability ». *IEICE Transactions on Fundamentals* vol E80-A, pages 2–8, 1997.

[AV96] W. Aiello et R. Venkatesan. « Foiling Birthday Attacks in Length-Doubling Transformations ». In *Advances in Cryptology -EUROCRYPT'96*, Saragosse, Espagne, pages 307–321. Lecture Notes in Computer Science 1070, Springer-Verlag, 1996.

[BAB96] I. Ben-Aroya et E. Biham. « Differential Cryptanalysis of Lucifer ». *Journal of Cryptology* vol 9, pages 21–34, 1996.

[BF00] S. Babbage et L. Frisch. « On MISTY1 Higher Order Differential Cryptanalysis ». In *Information Security and Cryptology, ICISC'00*, Seoul, Corée, pages 23–37. Lectures Notes in Computer Science 2015, Springer-Verlag, 2000.

[BGG+99] O. Baudron, H. Gilbert, L. Granboulan, H. Handschuh, A. Joux, P. Nguyen, F. Noilhan, D. Pointcheval, T. Pornin, G. Poupard, J. Stern, et S. Vaudenay. « Report on the AES Candidates ». In *The Second Advanced Encryption Standard Candidate Conference*. N.I.S.T., 1999.

[Bih00] E. Biham. « Cryptanalysis of Patarin's 2 Round Public Key System with S-Boxes ». In *Advances in Cryptology -EUROCRYPT'2000*, Bruges, Belgique, pages 408–417. Lecture Notes in Computer Science 1807, Springer-Verlag, 2000.

[BKR94] M. Bellare, J. Kilian, et P. Rogaway. « The Security of Cipher Block Chaining ». In *Advances in Cryptology -CRYPTO'94*, Santa Barbara, Californie, États-Unis, pages 341–355. Lecture Notes in Computer Science 839, Springer-Verlag, 1994.

[BR00] P. Barreto et V. Rijmen. « The Anubis Block Cipher ». In *First Open Nessie Workshop*, Leuven, Belgique. I.S.T., téléchargeable à l'adresse http ://www.cryptonessie.org/, 2000.

[BS91] E. Biham et A. Shamir. « Differential Cryptanalysis of DES-like Cryptosystems ». *Journal of Cryptology*, vol 4, no. 1, 1991.

[BS93] E. Biham et A. Shamir. *Differential Cryptanalysis of the Data Encryption Standard*. Springer-Verlag, New York, 1993.

[BS01] A. Biryukov et A. Shamir. « Structural Cryptanalysis of SASAS ». In *Advances in Cryptology -EUROCRYPT'2001*, Innsbruck, Austria, pages 394–405. Lecture Notes in Computer Science 1807, Springer-Verlag, 2001.

[CBD+98] D. Coppersmith, C. Burwick, E. D'Avignon, R. Gennaro, S. Halevi, C. Jutla, S. Matyas Jr., L. O'Connor, M. Peyravian, D.Safford, et N. Zunic. « Mars - a Candidate Ciphe for AES ». In *The First Advanced Encryption Standard Candidate Conference*. N.I.S.T., 1998.

[CCD99] A. Canteaut, P. Charpin, et H. Dobbertin. « A New Characterization of Almost Bent Functions ». In *Fast Software Encryption'99*, Rome, Italie, pages 186–200. Lectures Notes in Computer Science 1636, Springer-Verlag, 1999.

[CCD00a] A. Canteaut, P. Charpin, et H. Dobbertin. « Binary m-sequences with Three-valued Crosscorrelation : A Proof of Welch Conjecture ». *IEEE Transactions on Information Theory*, vol 46, no. 1, 2000.

[CCD00b] A. Canteaut, P. Charpin, et H. Dobbertin. « Weight Divisibility of Cyclic Codes, Highly Nonlinear Functions on $GF(2^m)$ and Crosscorrelation of Maximum-length Sequences ». *SIAM Journal on Discrete Mathematics*, vol 13, no. 1, 2000.

[CCZ98] C. Carlet, P. Charpin, et V. Zinoviev. « Codes, bent functions and permutations suitable for DES-like cryptosystems ». *Designs, Codes and Cryptography* vol 15, pages 125–126, 1998.

[CG91] A. Corfdir et H. Gilbert. « A Known Plaintext Attack of FEAL-4 and FEAL-6 ». In *Advances in Cryptology -CRYPTO'91*, Santa Barbara, Californie, États-Unis, pages 172–181. Lectures Notes in Computer Science 576, Springer-Verlag, 1991.

[CGMT02] N. Courtois, L. Goubin, W. Meier, et J. Tacier. « Solving Underdefined Systems of Multivariate Quadratic Equations ». In *Public Key Cryptography 2002*, Paris, France, pages 211–227. Lectures Notes in Computer Science 2274, Springer-Verlag, 2002.

[Cou01] N. Courtois. « *La Sécurité des Primitives Cryptographiques Basées sur des Problèmes Algébriques Multivariables : MQ, IP, MinRank, HFE* ». Thèse de doctorat, Université Paris VI, France, Septembre 2001.

[CP02] N. Courtois et J. Pieprzyk. « Cryptanalysis of Block Ciphers with Overdefined Systems of Equations ». In *Advances in Cryptology -Asiacrypt'02*, Queenstown, Nouvelle Zélande. à paraître, Springer-Verlag, 2002.

[CV94] F. Chabaud et S. Vaudenay. « Links Between Differential and Linear Cryptanalysis ». In *Advances in Cryptology -EUROCRYPT'94*, Pérouse, Italie, pages 256–265. Lecture Notes in Computer Science 950, Springer-Verlag, 1994.

[CV02] A. Canteaut et M. Videau. « Degree of Composition of Highly Functions and Applications to Higher Order Differential Cryptanalysis ». In *Advances in Cryptology -EUROCRYPT'2002*, Amsterdam, Hollande, pages 518–533. Lecture Notes in Computer Science 2332, Springer-Verlag, 2002.

[CW79] L. Carter et M. Wegman. « Universal Classes of Hash Functions ». *Journal of Computer and System Sciences* vol 18, pages 143–154, 1979.

[CW81] L. Carter et M. Wegman. « New Hash Functions and their Use in Authentification and Set Equality ». *Journal of Computer and System Sciences* vol 22, pages 265–279, 1981.

[CW82] D. Coppersmith et S. Winograd. « On the Asymptotic Complexity of Matrix Multiplication ». *SIAM Journal on Computing* vol 11 nN° 3, pages 472–492, Août 1982.

[Dae95] J. Daemen. « *Cipher and Hash Function Design Strategies Based on Linear and Differential Cryptanalysis* ». Thèse de doctorat, Katholieke Universiteit Leuven, Belgique, Mars 1995.

[DBRP99] C. D'Halluin, G. Bijnens, V. Rijmen, et B. Preneel. « Attack on Six Rounds of Crypton ». In *Fast Software Encryption'99*, Rome, Italie, pages 46–59. Lectures Notes in Computer Science 1636, Springer-Verlag, 1999.

[DH76] W. Diffie et M.E. Hellman. « New Directions in Cryptography ». *IEEE Trans. Inform. Theory* vol IT-22, no. 6, pages 644–654, 1976.

[DKR97] J. Daemen, L.R. Knudsen, et V. Rijmen. « The Block Cipher Square ». In *Fast Software Encryption'97*, Haifa, Israël, pages 149–165. Lectures Notes in Computer Science 1267, Springer-Verlag, 1997.

[Dob99] H. Dobbertin. « Almost Perfect nonlinear Power functions on $GF(2^n)$: The Welch Case ». *IEEE Transactions on Information Theory*, vol 45 nN° 4, Mai 1999.

[DR98] J. Daemen et V. Rijmen. « AES Proposal : Rijndael ». In *The First Advanced Encryption Standard Candidate Conference*. N.I.S.T., téléchargeable à l'adresse : www.esat.kuleuven.ac.be/~rijmen/rijndael/, 1998.

[DR99] J. Daemen et V. Rijmen. « AES Proposal : Rijndael ». In *The Second Advanced Encryption Standard Candidate Conference*. N.I.S.T., téléchargeable à l'adresse : http ://csrc.nist.gov/encryption/aes/, 1999.

[DR02] J. Daemen et V. Rijmen. *The Design of Rijndael*. Springer-Verlag, 2002.

[FKS+00] N. Ferguson, J. Kelsey, B. Schneier, M. Stay, D. Wagner, et D. Whiting. « Improved Cryptanalysis of Rijndael ». In *Fast Software Encryption'00*, New York, États Unis, pages 213–230. Lectures Notes in Computer Science 1978, Springer-Verlag, 2000.

[GGH+98] Henri Gilbert, Marc Girault, Philippe Hoogvorst, Fabrice Noilhan, Thomas Pornin, Guillaume Poupard, Jacques Stern, et Serge Vaudenay. « Decorrelated Fast Cipher : an AES Candidate (Extended Abstract) ». In *The First Advanced Encryption Standard Candidate Conference*. N.I.S.T., 1998.

[Gil97] H. Gilbert. « *Cryptanalyse Statistique des Algorithmes de Chiffrement et Sécurité des Schémas d'Authentification* ». Thèse de doctorat, Université de Paris-Sud, 1997.

[GM00] H. Gilbert et M. Minier. « A Collision Attack on 7 Rounds of Rijndael ». In *The Third Advanced Encryption Standard Candidate Conference*. N.I.S.T., 2000.

[GM01] H. Gilbert et M. Minier. « New Results on the Pseudo-Randomness of some Blockcipher Constructions ». In *Fast Software Encryption'01*, Yokohama, Japon. Lectures Notes in Computer Science, Springer-Verlag, à paraître, 2001.

[GM02] H. Gilbert et M. Minier. « Cryptanalysis of SFLASH ». In *Advances in Cryptology -EUROCRYPT'2002*, Amsterdam, Hollande, pages 288–298. Lecture Notes in Computer Science 2332, Springer-Verlag, 2002.

[HG97] H. Handschuh et H. Gilbert. « χ^2 Cryptanalysis of SEAL Encryption Algorithm ». In *Fast Software Encryption'97*, Haifa, Israël, pages 1–12. Lectures Notes in Computer Science 1267, Springer-Verlag, 1997.

[HM97] C. Harpes et J.L. Massey. « Partitioning Cryptanalysis ». In *Fast Software Encryption'97*, Haifa, Israël, pages 13–26. Lectures Notes in Computer Science 1267, Springer-Verlag, 1997.

[JK97] T. Jacobsen et L.R. Knudsen. « The Interpolation Attack on Block Ciphers ». In *Fast Software Encryption'97*, Haifa, Israël, pages 28–40. Lectures Notes in Computer Science 1267, Springer-Verlag, 1997.

[Jun01] P. Junod. « On the Complexity of Matsui's Attack ». In *Selected Areas in Cryptography - SAC'01*, Toronto, Canada, pages 199–211. Lecture Notes in Computer Science 2259, Springer-Verlag, 2001.

[Kas71] T. Kasami. « The Weight Enumerators for Several Classes of Subcodes of the Second Order Binary Reed-Muller Codes ». *Information and Control* vol 18, pages 369–394, 1971.

[KAS99] KASUMI. « Specification of the 3GPP confidentiality and Integrity algorithm KASUMI ». 1999.

[KMA+98] M. Kanda, S. Moriai, K. Aoki, H. Ueda, M. Ohkubo, Y. Takashima, K. Ohta, et T. Matsumoto. « E2 : A Candidate Cipher for AES ». In *The First Advanced Encryption Standard Candidate Conference*. N.I.S.T., téléchargeable à l'adresse http ://info.isl.ntt.co.jp/e2/, 1998.

[Knu95] L. Knudsen. « Truncated and Higher Order Differentials ». In *Fast Software Encryption'95*, Leuven, Belgium, pages 196–211. Lectures Notes in Computer Science 1008, Springer-Verlag, 1995.

[KS99] A. Kipnis et A. Shamir. « Cryptanalysis of the HFE Public Key Cryptosystem by Relinearization ». In *Advances in Cryptology -Crypto'99*, Santa Barbara, Californie, États-Unis, pages 19–31. Lecture Notes in Computer Science 1666, Springer-Verlag, 1999.

[KSS97] Y. Kaneko, F. Sano, et K. Sakurai. « On Provable Security against Differential and Linear Cryptanalysis in Generalized Feistel Ciphers with Multiple

Random Functions ». In *Selected Areas in Cryptography - SAC'97*, Ottawa, Ontario, Canada. Springer-Verlag, 1997.

[KW02] L. Knudsen et D. Wagner. « Integral Cryptanalysis ». In *Fast Software Encryption'02*, Leuven, Belgique, page à paraitre. Lectures Notes in Computer Science, Springer-Verlag, 2002.

[Lim98] C.H. Lim. « CRYPTON : A New 128-bit Block Cipher ». In *The First Advanced Encryption Standard Candidate Conference*. N.I.S.T., 1998.

[Lim99] C.H. Lim. « A Revised Version of CRYPTON Version 1.0 ». In *Fast Software Encryption'99*, Rome, Italie, pages 31–46. Lectures Notes in Computer Science 1636, Springer-Verlag, 1999.

[LM90] X. Lai et J.L. Massey. « A Proposal for a New Block Encryption Standard ». In *Advances in Cryptology -EUROCRYPT'90*, Aarhus, Denemark, pages 389–402. Lecture Notes in Computer Science 473, Springer-Verlag, 1990.

[LM91] X. Lai et J.L. Massey. « Markov Ciphers and Differential Cryptanalysis ». In *Advances in Cryptology -EUROCRYPT'91*, Brighton, Royaume Uni, pages 17–38. Lecture Notes in Computer Science 547, Springer-Verlag, 1991.

[LR88] M. Luby et C. Rackoff. « How to Construct Pseudorandom Permutations from Pseudorandom Functions ». *SIAM Journal on Computing* vol 17, no. 2, pages 373–386, 1988.

[Luc99] S. Lucks. « On Security of the 128-Bit Block Cipher DEAL ». In *Fast Software Encryption'99*, Rome, Italie, pages 60–70. Lectures Notes in Computer Science 1636, Springer-Verlag, 1999.

[Luc00] S. Lucks. « Attacking Seven Rounds of Rijndael Under 192-bit and 256-bit Keys ». In *The Third Advanced Encryption Standard Candidate Conference*. N.I.S.T., 2000.

[Luc01] S. Lucks. « The Saturation Attack - a Bait for Twofish ». In *Fast Software Encryption'01*, Yokohama, Japon, page à paraître. Lectures Notes in Computer Science, Springer-Verlag, 2001.

[Mas94] J. Massey. « SAFER K-64 : a Byte-oriented Block-ciphering Algorithm ». In *Fast Software Encryption'94*, Cambridge, Royaume Uni, pages 1–17. Lectures Notes in Computer Science 809, Springer-Verlag, 1994.

[Mat93] M. Matsui. « Linear Cryptanalysis Method for DES Cipher ». In *Advances in Cryptology -EUROCRYPT'93*, Lofthus, Norvége, pages 386–397. Lecture Notes in Computer Science 765, Springer-Verlag, 1993.

[Mat94] M. Matsui. « The First Experimental Cryptanalysis of the Data Encryption Standard ». In *Advances in Cryptology -CRYPTO'94*, Santa Barbara, Californie, États-Unis, pages 1–11. Lecture Notes in Computer Science 839, Springer-Verlag, 1994.

[Mat97] M. Matsui. « New Block Encryption Algorithm MISTY ». In *Fast Software Encryption'97*, Haifa, Israël, pages 54–68. Lectures Notes in Computer Science 1267, Springer-Verlag, 1997.

[Mau92] U. Maurer. « A Simplified and Generalized Treatment of Luby-Rackoff Pseudorandom Permutation Generators ». In *Advances in Cryptology - EUROCRYPT'92*, Balatonfüred, Hongrie, pages 239–253. Lecture Notes in Computer Science 658, Springer-Verlag, 1992.

[MG00] M. Minier et H. Gilbert. « Stochastic Cryptanalysis of Crypton ». In *Fast Software Encryption'00*, New York, États Unis, pages 121–133. Lectures Notes in Computer Science 1978, Springer-Verlag, 2000.

[MI88] M. Mastsumoto et H. Imai. « Public Quadratic Polynomial-tuples for Efficient Signature-Verification and Message Encryption ». In *Advances in Cryptology -EUROCRYPT'88*, Davos, Suisse, pages 419–453. Lecture Notes in Computer Science 330, Springer-Verlag, 1988.

[MKK98] J.L. Massey, G.H. Khachatrian, et M. K. Kuregian. « SAFER + ». In *The First Advanced Encryption Standard Candidate Conference*. N.I.S.T., 1998.

[MKK00] J.L. Massey, G.H. Khachatrian, et M. K. Kuregian. « SAFER ++ ». In *First Open Nessie Workshop*, Leuven, Belgique. I.S.T., téléchargeable à l'adresse http ://www.cryptonessie.org/, 2000.

[MR02] S. Murphy et M. Robshaw. « Essential Algebraic Structure Within the AES ». In *Advances in Cryptology -CRYPTO'02*, Santa Barbara, Californie, États-Unis, pages 1–16. Lectures Notes in Computer Science 2442, Springer-Verlag, 2002.

[MT99] M. Matsui et T. Tokita. « Cryptanalysis of a Reduced Version of the Block Cipher E2 ». In *Fast Software Encryption'99*, Rome, Italie, pages 71–80. Lectures Notes in Computer Science 1636, Springer-Verlag, 1999.

[MV00] S. Moriai et S. Vaudenay. « Comparison of Randomness Provided by Several Schemes for Block Ciphers ». In *The Third Advanced Encryption Standard Candidate Conference*. N.I.S.T., 2000.

[Nat77] National Bureau of Standards (États Unis). « Data Encryption Standard ». *Federal Information Processing Standard*, 1977. Publication 46.

[NES01a] NESSIE. « NESSIE Phase I : Selection of Primitives ». téléchargeable sur http ://www.cryptonessie.org/, 2001.

[NES01b] NESSIE. « Security Evaluation of NESSIE First Phase ». téléchargeable sur http ://www.cryptonessie.org/, 2001.

[NK95] K. Nyberg et L.R. Knudsen. « Provable Security against Differential Attack ». *Journal of Cryptology* vol 8, no. 1, pages 27–38, 1995.

[Nyb91] K. Nyberg. « Perfect Nonlinear S-boxes ». In *Advances in Cryptology - EUROCRYPT'91*, Brighton, Royaume Uni, pages 378–385. Lecture Notes in Computer Science 547, Springer-Verlag, 1991.

[Nyb94] K. Nyberg. « Linear Approximation of Block Ciphers ». In *Advances in Cryptology -EUROCRYPT'94*, Pérouse, Italie, pages 439–444. Lecture Notes in Computer Science 950, Springer-Verlag, 1994.

[OK98] Richard Outerbridge et Lars Knudsen. « DEAL - A 128-bit Block Cipher ». In *The First Advanced Encryption Standard Candidate Conference*. N.I.S.T., 1998.

[Pat91] J. Patarin. « *Etudes des Générateurs de Permutations Pseudo-aléatoires Basés sur le Schéma du DES* ». Thèse de doctorat, Université Paris VI, 1991.

[Pat95] J. Patarin. « Cryptanalysis of the Matsumoto and Imai Public Key Scheme of Eurocrypt'88 ». In *Advances in Cryptology -Crypto'95*, Santa Barbara, Californie, États-Unis, pages 248–261. Lecture Notes in Computer Science 963, Springer-Verlag, 1995.

[Pat96] J. Patarin. « Hidden Fields Equations (HFE) and Isomorphisms of Polynomials (IP) : two new families of Asymmetric Algorithms ». In *Advances in Cryptology -EUROCRYPT'96*, Saragosse, Espagne, pages 33–48. Lecture Notes in Computer Science 1070, Springer-Verlag, 1996.

[Pat00] J. Patarin. « *La Cryptographie Multivariable* ». Habilitation à diriger les recherches, Université de Paris VII, 2000.

[PCG00] J. Patarin, N. Courtois, et L. Goubin. « SFLASH ». In *First Open Nessie Workshop*, Leuven, Belgique. I.S.T., téléchargeable à l'adresse http ://www.cryptonessie.org/, 2000.

[PGC98] J. Patarin, L. Goubin, et N. Courtois. « C^{*-+} and HM : Variations around two Schemes of T. Matsumoto and H. Imai ». In *Advances in Cryptology -ASIACRYPT'98*, Pékin, Chine, pages 35–49. Lecture Notes in Computer Science 1514, Springer-Verlag, 1998.

[RRSY98] R. Rivest, M. Robshaw, R. Sidney, et Y. Yin. « The RC6 Block Cipher ». In *The First Advanced Encryption Standard Candidate Conference*. N.I.S.T., 1998.

[RSA78] R.L. Rivest, A. Shamir, et L.M. Adleman. « A Method for Obtaining Digital Signatures and Public-key Cryptosystems ». *Communications of ACM* vol 21, no. 3, pages 120–126, 1978.

[Sha49] C.E. Shannon. « Communication Theory of Secrecy Systems ». *Bell System Technical Journal* vol 28, pages 656–715, 1949.

[SKW$^+$98] B. Schneier, J. Kelsey, D. Whiting, D. Wagner, C. Hall, et N. Ferguson. « Twofish : A New Block Cipher ». In *The First Advanced Encryption Standard Candidate Conference*. N.I.S.T., 1998.

[Sug96] M. Sugita. « Pseudorandomness of a Block Cipher MISTY ». ISEC96-9 IEICE, 1996.

[Sug97] M. Sugita. « Pseudorandomness of a Block Cipher with Recursive Structures ». ISEC97-9 IEICE, 1997.

[Sug98] M. Sugita. « Higher Order Differential Attack of Block Cipher MISTY 1,2 ». ISEC98-1 IEICE, 1998.

[SV94] C. P. Schnorr et S. Vaudenay. « Black Box Cryptanalysis of Hash Networks Based on Multipermutations ». In *Advances in Cryptology -EUROCRYPT'94*, Pérouse, Italie, pages 47–57. Lecture Notes in Computer Science 950, Springer-Verlag, 1994.

[SV98] J. Stern et S. Vaudenay. « CS-Cipher ». In *Fast Software Encryption'98*, Paris, France, pages 189–205. Lectures Notes in Computer Science 1372, Springer-Verlag, 1998.

[SZ97] K. Sakurai et Y. Zheng. « On Non-Pseudorandomness from Block Ciphers with Provable Immunity Against Linear Cryptanalysis ». *IEICE Trans. Fundamentals*, vol vol. E80-A nN° 1, Janvier 1997.

[THK99] H. Tanaka, K. Hisamatsu, et T. Kaneko. « Strength of MISTY1 Without FL Function for Higher Differential Attack ». In *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes*, Honolulu, Hawai, États-Unis, pages 221–230. Lecture Notes in Computer Science 1719, Springer-Verlag, 1999.

[Vau95] S. Vaudenay. « *La Sécurité des Primitives Cryptographiques* ». Thèse de doctorat, Université Paris VII, France, Avril 1995.

[Vau98a] S. Vaudenay. « Feistel Ciphers with L_2-Decorrelation ». In *Selected Areas in Cryptography - SAC'98*, Kingston, Ontario, Canada, pages 1–14. Lectures Notes in Computer Science 1556, Springer-Verlag, 1998.

[Vau98b] S. Vaudenay. « Provable Security for Block Ciphers by Decorrelation ». In *STACS'98*, Paris, France, pages 249–275. Lectures Notes in Computer Science 1373, Springer-Verlag, 1998.

[Vau98c] S. Vaudenay. « Provable Security for Block Ciphers by Decorrelation ». Technical Report LIENS-98-8, École Normale Supérieure, téléchargeable à ftp ://ftp.ens.fr/pub/reports/liens/liens-98-8.A4.ps.Z, 1998.

[Vau99a] S. Vaudenay. « Adaptative-Attack Norm for Decorrelation and Super-Pseudorandomness ». Technical Report LIENS-99-2, École Normale Supérieure, téléchargeable à ftp ://ftp.ens.fr/pub/reports/liens/liens-99-2.A4.ps.Z, 1999.

[Vau99b] S. Vaudenay. « On Provable Security for Conventional Cryptography ». In *Information Security and Cryptology - ICISC'99*, Séoul, Corée, pages 1–16. Lectures Notes in Computer Science 1787, Springer-Verlag, 1999.

[Vau99c] S. Vaudenay. « Resistance Against General Iterated Attacks ». In *Advances in Cryptology -EUROCRYPT'99*, Prague, République Tchéque, pages 255–271. Lecture Notes in Computer Science 1592, Springer-Verlag, 1999.

[Vau99d] S. Vaudenay. « *Vers une Théorie du Chiffrement Symétrique* ». Habilitation à diriger les recherches, Université de Paris VII, 1999.

[YLD99] D. Ye, K. Lam, et Z. Dai. « A Known Plaintext Attack of FEAL-4 and FEAL-6 ». In *Advances in Cryptology -CRYPTO'99*, Santa Barbara, Californie, États-Unis, pages 315–325. Lectures Notes in Computer Science 1666, Springer-Verlag, 1999.

[ZMI90] Y. Zheng, T. Matsumoto, et H. Imai. « On the construction of Block Ciphers Provably Secure and Not relying on Any Unproved Hypotheses ». In

Advances in Cryptology -CRYPTO'89, Santa Barbara, Californie, États-Unis, pages 461–480. Lecture Notes in Computer Science 435, Springer-Verlag, 1990.