La cryptanalyse différentielle et ses généralisations

Céline Blondeau

Thèse de doctorat de l'université Pierre et Marie Curie

Spécialité Informatique

edite de Paris

présentée par

Céline Blondeau

pour obtenir le grade de

Docteur de l'université Pierre et Marie Curie

La cryptanalyse différentielle et ses généralisations

soutenue le 7 novembre 2011,

devant le jury composé de

Directeur de thèse	
Pascale Charpin	INRIA Paris-Rocquencourt
Rapporteurs	
Kaisa Nyberg	Aalto University, Finlande
Henri Gilbert	ANSSI
Examinateurs	
Thierry Berger	Université de Limoges
Anne Canteaut	INRIA Paris-Rocquencourt
Gohar Kyureghyan	Otto-von-Guericke University of Magdeburg, Allemagne
Michèle Soria	Université Pierre et Marie Curie
Jean-Pierre Tillich	INRIA Paris-Rocquencourt

Céline Blondeau

La cryptanalyse différentielle et ses généralisations

INRIA équipe-projet SECRET
Domaine de Voluceau
78153 Le Chesnay

Remerciements

Je voudrais d'abord exprimer toute ma gratitude à Pascale Charpin, ma directrice de thèse, qui m'a accompagnée pendant les trois ans et demi que j'ai passés au sein de l'équipe SECRET qui a d'abord accepté, de diriger mon stage de master, puis, d'encadrer ma thèse qui sans elle, j'en suis persuadée ne serait pas ce qu'elle est. Je tiens à la remercier pour toutes ces heures pendant lesquelles nous avons travaillé ensemble, pour tous les conseils qu'elle m'a prodigués et le temps qu'elle a bien voulu consacrer à la relecture du manuscrit de ma thèse.

Je suis aussi très sincèrement reconnaissante à Henri Gilbert et Kaisa Nyberg qui ont accepté de rapporter cette thèse. Merci à Kaisa de l'avoir relue alors que celle-ci est en français. Merci à Henri pour ses commentaires détaillés et judicieux qui m'ont permis d'améliorer le rendu du manuscrit. Merci à Kaisa qui m'a accueillie au sein de son laboratoire en mai de cette année pour deux semaines qui m'ont en outre permis de découvrir ce magnifique pays qu'est la Finlande et qui maintenant m'a intégré dans son équipe. Puisse la Finlande encore me plaire après deux hivers !

Je tiens à remercier les autres membres du jury, Thierry Berger, Anne Canteaut, Gohar Kyureghyan, Michèle Soria et Jean-Pierre Tillich. Un grand merci à Anne, la directrice de l'équipe SECRET, qui m'a admise au sein de son équipe avec laquelle j'ai partagé de nombreuses discussions, scientifiques ou non, qui s'est toujours montrée disponible et dont les conseils m'ont été précieux lors de rédaction d'articles.

Merci à Jean-Pierre pour ses recommandations et ses lumières scientifiques.

Cette thèse n'aurait pas vu le jour, si je n'avais pas auparavant effectué mon master à l'université de Limoges. Aussi je tiens à remercier les professeurs qui m'ont suivie pendant cette année d'étude, en particulier Thierry qui de plus a accepté de faire partie du jury.

Merci à Gohar membre de ce jury, alors que la thèse et sa soutenance sont en français, ainsi qu'à Michèle qui a eu de même la gentillesse d'y participer.

Durant ces trois années, j'ai eu l'occasion d'échanger avec Nicolas Sendrier et Daniel Augot les autres permanents de l'équipe (Daniel au début). Merci aussi à eux.

Toute ma gratitude à ceux qui m'ont permis d'enseigner, Julien Brajard, Louis Goubin et Christophe Vermaelen et à Matthieu Finiasz qui a toujours été de bon conseils.

Je voudrais, dire un grand merci aussi aux personnes avec lesquelles j'ai eu l'occasion de travailler : Benoît Gérard pour nos nombreuses collaborations ainsi que Pascale, Anne et Jean-Pierre une nouvelle fois, mais aussi Mohammed Ahmed Abdelraheem, Maria Naya-Plasencia, Marion Videau et Erik Zenner.

Alors que je préparais cette thèse, j'ai eu l'opportunité de croiser beaucoup de monde au sein de l'équipe : des stagiaires, des doctorants, des post-doctorants et des visiteurs. Tous ont contribué à la dynamique du projet. En espérant n'oublier personne, je remercie par ordre alphabétique : Alexander, Andrea, Anne C, Anne M., Assia, Ayca, Ayoub,

Baudoin, Bhaskar, Benoît, Cédric F., Cédric L., Claire, Chrysanthi, Christina, Christelle, Christophe, Denise, Daniel, Grégory, Jean-Christophe,Jean-Pierre, Mamdouh, Maria, Marion, Mathieu, Matthieu, Maxime, Nicolas, Pascale, Rafael, Stéphane J., Stéphane M., Sumanta, Sunandan, Valentin, Valérie, Vincent et Yann.

Je tiens aussi à faire des dédicaces particulières aux personnes qui ont fait que ces trois années ont été agréables et sont passées relativement vite.

Je remercie tout particulièrement Benoît, avec qui j'ai toujours vécu de bons moments lors des conférences, ou dans la salle café quand nous discutions de nos articles. Je garde un vif souvenir de nos collaborations qui, je le souhaite, se renouvelleront dans l'avenir.

Je tiens à faire une dédicace spéciale à tous les membres du bureau 1 avec qui j'ai passé de longues heures : Ayoub, Christina, Grégory, Maria. Merci à Ayoub pour son écoute, à Grégory qui a toujours pris le temps de répondre à mes questions "informatique" et à Christina pour sa présence et sa générosité.

Je remercie Andrea, qui a facilité mon installation en Finlande en me cédant son appartement et dont les conseils m'ont été très utiles. Merci à Maria pour m'avoir hébergé alors que je n'habitais déjà plus à Paris.

De ces trois années, je retiens aussi le voyage au Texas avec Christina et la visite divertissante de ses villes. Ce voyage qui était un préliminaire à la conférence ISIT me laisse d'heureux souvenirs en compagnie de Alexander, Antonia, Christina, Jean-Pierre, Matthieu et Stéphane.

Une petite dédicace aussi aux nouveaux arrivants dans l'équipe SECRET, Marion et Valentin avec qui j'aurais aimé passer plus de temps. Bonne chance à eux.

J'ai de même une petite pensée pour toutes les personnes avec qui j'ai eu l'occasion de courir : Alexandre, Ayoub, Baudoin, Cécilia, Christina, Marion, Mathieu, Julien, Phillipe et Stéphane.

Je tiens aussi à remercier, l'équipe Lospumas, en particulier, Andrea, Fab, Gaetan, Lore, Maria, Stéphane et Yann. Les soirées passées ensemble aux blindtests, nos différents week end à Helsinki, Zurick, Luxembourg et nos vacances à Grenoble sont autant d'agréables souvenirs.

Cela fait deux mois maintenant que j'habite en Finlande, je tiens donc aussi à remercier les membres de l'équipe cryptographie : Billy, Hadi, Kaisa, Kimmo, Risto ainsi que les camarades de la salle café Siert, Tero, Roland et les autres.

Avant de conclure ces remerciements par mes proches, je tenais à faire une dédicace spéciale à Grace qui m'a beaucoup appris et qui a toujours été là dans les moments difficiles. Merci pour tous les moments passés ensemble en cours, en vacances, en soirées, ou même au téléphone ! Merci aussi à Florine et Stéphane pour mon premier voyage aux USA.

Enfin, j'ai une pensée émue pour ma famille, mes grand parents, mes parents Christine et Régis, qui sont toujours à mon écoute et m'encouragent de leur confiance. Merci à mon frère Mathieu et à ma sœur Juliette qui m'ont depuis le début soutenue dans mon projet.

Overview

In this document, I present my work in the field of *symmetric cryptography* done during the period 2008-2011, where I did my research as Phd-student in the team SECRET at INRIA.

Our results have mainly concern with *the analysis and the design of block ciphers.* Since the beginning of 90's, there exist a lot of statistical attacks against block ciphers. In the first part of our work, we focus on the generalizations of the so-called *differential cryptanalysis.* The second part is devoted to some design criteria for block ciphers.

In the first part, our main interest was the determination of the complexity of statistical attacks. We notably made an extensive study on the data complexity and the success probability of most of the statistical attacks on block ciphers. Our results have been presented in a poster session at EUROCRYPT 2009 [BG09a] and in the international conference *Workshop on coding and cryptography* [BG09b]. A complete version was published in the journal *Designs, Codes and Cryptography* [BGT11]. Among the statistical attacks, the differential cryptanalysis and its generalizations have a crucial role because of their importance for the security of block ciphers. During our cryptanalysis of PRESENT we checked the hypotheses which are currently done in a differential cryptanalysis. The results and observations stemming from our experiments have been published in the *Workshop on Tools for cryptanalysis* [BG10]. Most recently, we proposed to use "many differentials" in such a cryptanalysis. We then described more carefully the so-called *multiple differential cryptanalysis* and began an overall study of its complexity. A part of our results has been presented at the international conference *Fast Software Encryption-FSE-2011* [BG11]. Our results in this context have been obtained in collaboration with Benoît Gérard and Jean-Pierre Tillich.

The second part is dedicated to the study of the S-boxes of block ciphers. The most important criterion concerning the resistance of a block cipher against differential attacks is called the *differential uniformity* of its S-boxes. In this part, we introduce the notion of *differential spectrum* of power functions over a finite field and we explain why we have here a more general criterion which may be of great interest. Some of the results in this second part have been presented in *IEEE International Symposium on Information Theory- ISIT-2010* [BCC10a] and in *The 10th International Conference on Finite Fields and their Applications Fq10-2011* [BCC11a] and published in the *International Journal of Information and Coding Theory* [BCC10b] and in *IEEE Transactions on Information Theory* [BCC11b]. In these papers, written with Anne Canteaut and Pascale Charpin, we notably describe the differential spectra of several classes of power functions.

Part I

Statistical attacks are the most important attacks against block ciphers. The aim, of such an attack is to recover information on the secret key. There exist a lot of statistical attacks. In Chapter 1 we recall the basic knowledge concerning statistical attacks against block ciphers. Chapters 2 and 3 form a survey of the main statistical attacks on block ciphers.

Differential cryptanalysis was introduced in the 90's. This statistical attack exploits a non-uniform distribution of some differential to recover information on the key. There exists a lot of generalizations of this attack, namely *truncated differential cryptanalysis, impossible differential cryptanalysis, higher-order differential cryptanalysis,* for instance. Chapter 2 is a survey of the generalizations of differential cryptanalysis.

There are statistical attacks which are more or less related to differential cryptanalysis. In Chapter 3, we present some of these attacks that can be used against block ciphers. We describe, in particular, linear cryptanalysis and its generalizations, integral cryptanalysis, boomerang attack or related-key attack.
In Chapters 2 and 3, we explain the connections between the different attacks. For each attack, we describe the corresponding algorithm and study the distribution of the random variables involved in the attack.

In this thesis, we concentrate our attention on differential cryptanalysis. To analyze the security of a block cipher against differential attack, we need to be able to compute the probabilities of the *best differentials*. We present here our algorithm to find the best differential trails. Using the best differential trails we obtain an estimation of the probabilities of the differentials. Generally, this computation is done under some general assumptions like the wrong-key randomization hypothesis. In Chapter 4, we present our experiments on the lightweight block cipher PRESENT, to verify some of the current hypotheses for differential cryptanalysis. Our results, from these experimentations, show that classical assumptions are not far to be true in the case of reduced version of PRESENT. This analysis have been taken into account when computing in Section 5.5 the formula of the success probability for differential cryptanalysis.

The complexities of a statistical attack rely on the data complexity, the success probability, the time complexity and the memory complexity. In order to study the data complexity and the success probability of a statistical attack, we need to know the distributions of the random variables involved in the attack. Actually, in a lot of statistical attacks against block ciphers, as presented in Chapters 2 and 3, the variables follow a binomial distribution. In Chapter 5, we use a general framework to analyze the data complexity and the success probability of all attacks presented in Chapter 2 and 3. We recall previous works concerning differential cryptanalysis and linear cryptanalysis. We explain why they cannot be applied for all statistical attacks. In Section 5.3.2, we present an algorithm which uses a dichotomic search in order to find the data complexity of a statistical attack for a fixed success probability. This algorithm gives an exact value of the data complexity, but finding a general formula for the data complexity is more complicated as the binomial distribution is hard to invert. In Section 5.3, by using an approximation

of the binomial distribution, we propose a formula of the data complexity of statistical attacks when the random variables follow a binomial distribution. In [*Sel08*], a formula for the success probability is given. This formula uses a normal approximation of the binomial distribution. In Section 5.4, we present a formula for the success probability of a statistical attack which does not involve any approximation of binomial distribution.

The so-called *truncated differential cryptanalysis* uses a set of differentials which are related through certain properties, to find information of the key of a given block cipher. On the contrary, for some attacks, we need some differentials that are not necessary related. We call this attack *multiple differential cryptanalysis*. When computing the data complexity and the success probability of multiple differential cryptanalysis we have been confronted to the fact that the random variables do not follow a binomial distribution anymore. Therefore, after studying the distributions of the random variables (in Section 6.2), we derive (in Section 6.3) a formula for the data complexity and the success probability of multiple differential cryptanalysis. Our results, presented in this chapter, are validated by our experimentations on SMALLPRESENT (Section 6.4). Using this framework, in Section 6.5 we propose an attack on PRESENT which improves the best differential attack on this block cipher.

Part II

For block ciphers, the resistance to differential cryptanalysis is quantified by the so-called *differential uniformity* of the Substitution-box (S-box) used in the cipher [*NK92*].

Most notably, finding appropriate S-boxes which guarantee that the cipher using them resists differential attacks is a major topic for the last twenty years. Power functions, *i.e.*, monomial functions, form a class of suitable candidates since they usually have a lower implementation cost in hardware. Also, their particular algebraic structure makes the determination of their differential properties easier. However, there are only few power functions for which we can prove that they have a low differential uniformity.
Chapter 7 is dedicated to the presentation of some known results for Boolean functions, vectorial functions (especially power functions) and almost perfect non-linear (APN) functions.

In Chapter 8, we introduce the notion of the differential spectrum of power function that measures the resistance of the cipher to differential cryptanalysis.

The differential spectrum is the same for all APN functions. For non-APN functions, we can have a large deviation between the differential spectrum of two functions with same differential uniformity. We begin by explaining our motivation for the study of the full differential spectra of power permutations (in Section 8.1.2). Further, we study power functions differentially 4- and 6-uniform. We have done an exhaustive search, for fields with reasonable size, for all power functions differentially 4- and 6-uniform. The main results of our experiments are presented, and discussed, in Section 8.2. They indicate that the number of families which are differentially 4- and 6-uniform is relatively small.

In Section 8.3, we study the differential spectrum of the function

$$x \mapsto x^{2^{2k}+2^k+1} \quad \text{on the field } \mathbb{F}_{2^{4k}}.$$

Carl Bracken and Gregor Leander proved in [*BL10*] that these power functions are all differentially 4-uniform. We provide here the whole differential spectrum of these functions Other classes of power functions are sometimes differentially 4-uniform. In Section 8.4, we study the differential spectrum of power functions with a quadratic or a Kasami exponent. These families of functions have the particularity that the differential uniformity is always a power of 2 and that the differential spectrum is two-valued.
Another class of differentially 4-uniform functions is the class of inverse functions on the fields $\mathbb{F}_{2^n}$ with n even. Our experiments lead us to the conjecture that, up to equivalence, all differentially 4-uniform power functions $F(x) = x^d$ on $\mathbb{F}_{2^n}$ belong to one of the classes :

- $d = 2^{2k} + 2^k + 1$ and $n = 4k$,
- $d = 2^{t+1}$, $gcd(t, n) = 2$ and $gcd(2t, n) = 2$,
- $d = 2^{2t} - 2^t + 1$, $n \neq 3t$, $gcd(t, n) = 2$, and $n \equiv 2 \mod 4$,
- $d = 2^n - 2$ and n even.

If this conjecture is true, we have determined in this thesis the differential spectra of all differentially 4-uniform power functions.

Our study of differentially 6-uniform power functions, led us to the following observation : almost all these functions are of the form $G_t = x \mapsto x^{2^t-1}$. In Section 8.6, we present an extensive study of the differential spectrum of this family of functions. In particular, in Section 8.6.3, we point out a link between the differential spectra of G_t and G_s where $s = n - t + 1$.

In Section 8.6.5, we give the differential spectrum of $G_3(x) = x^7$ which is differentially 6-uniform. The last section in this chapter is dedicated to the study of the differential spectra of other particular exponents in this family.

Introduction générale

Le travail de recherche présenté dans cette thèse se situe en cryptographie symétrique. En particulier, nous nous intéressons à l'analyse et à la conception des systèmes de chiffrement par blocs.

Le début des années 1990 a vu l'avènement d'un certain nombre d'attaques statistiques pour les systèmes de chiffrement par bloc. Durant cette thèse, je me suis intéressée aux généralisations de la cryptanalyse différentielle.

La première partie de ce manuscrit est dédiée à la présentation d'un certain nombre d'attaques statistiques sur les systèmes de chiffrement par bloc. Dans le chapitre 5, nous proposons une étude générale qui permet de calculer la complexité en donnée et la probabilité de succès d'un certain nombre d'attaques statistiques des systèmes de chiffrements par bloc. Ces résultats ont été présentés lors d'une session poster à *EUROCRYPT-2009* [BG09a] et dans la conférence internationale *Workshop on coding and cryptography WCC-2009* [BG09b]. Une version complète de ces résultats comprenant l'analyse de la probabilité de succès a été publiée dans le journal *Designs Codes and Cryptography* [BGT11]. Le fil conducteur de cette partie reste l'analyse de la cryptanalyse différentielle et de ses généralisations. Des travaux plus récents nous ont permis en utilisant plusieurs différentielles de généraliser la cryptanalyse différentielle et la cryptanalyse différentielle tronquée. Dans ces travaux, nous étudions la complexité d'une attaque différentielle multiple. La majeure partie de ces résultats a été présenté dans la conférence internationale *Fast sofware encryption-FSE-2011* [BG11]. Ces travaux sur la cryptanalyse différentielle n'auraient pas été complets sans une analyse des hypothèses communément utilisées pour calculer la complexité d'une attaque différentielle multiple. Les résultats expérimentaux concernant ces travaux ont été quant à eux présentés au *"Workshop on Tools for Cryptanalysis"* [BG10]. Les résultats obtenus dans ce domaine sont le travail d'une collaboration avec Benoît Gérard et Jean-Pierre Tillich.

La seconde partie de cette thèse est dédiée à l'étude des critères sur les boîtes-S qui permettent de prémunir les systèmes de chiffrement par bloc contre les attaques différentielles. À la suite d'une étude approfondie de la résistance des boîtes-S de ces systèmes de chiffrement par bloc, nous avons introduit un nouveau critère, plus précis que l'uniformité différentielle, nous permettant de mesurer la vulnérabilité des boîtes-S aux attaques différentielles. Ainsi, avec Anne Canteaut et Pascale Charpin, nous avons introduit la notion de spectre différentiel et étudié le spectre différentiel de différentes classes de fonctions puissances. La plupart des résultats présentés dans cette section ont été soit présentés dans les conférences internationales *IEEE International Symposium on Information Theory-ISIT-2010* [BCC10a] et *The 10th International Conference on Finite Fields and their*

Applications Fq10-2011 [BCC11a] ou publiés dans les journaux *International Journal of Information and Coding Theory* [BCC10b] et *IEEE Transactions on Information Theory* [BCC11b].

Bibliographie

[ABNP+11] Mohamed Ahmed Abdelraheem, Céline Blondeau, María Naya-Plasencia, Marion Videau, and Erik Zenner. Cryptanalysis of ARMADILLO2. In D.H. Lee and X. Wang, editors, *Asiacrypt 2011*, volume 7073 of *Lecture Notes in Computer Science*, pages 308–326. Springer, 2011.

[BCC10a] Céline Blondeau, Anne Canteaut, and Pascale Charpin. Differential Properties of Power Functions. In *Proceedings of the 2010 IEEE International Symposium on Information Theory, ISIT 10*, pages 2478–2482, 2010.

[BCC10b] Céline Blondeau, Anne Canteaut, and Pascale Charpin. Differential Properties of Power Functions. *Int. J. Inform. and Coding Theory*, 1(2) :149–170, 2010.

[BCC11a] Céline Blondeau, Anne Canteaut, and Pascale Charpin. Differential Properties of $x \mapsto x^{2^t-1}$, July 2011. The 10th International Conference on Finite Fields and Applications - Fq10.

[BCC11b] Céline Blondeau, Anne Canteaut, and Pascale Charpin. Differential Properties of $x \mapsto x^{2^t-1}$. *IEEE Trans. Inform. Theory*, 57, December 2011.

[BG09a] Céline Blondeau and Benoît Gérard. On the Data Complexity of Statistical Attacks Against Block Ciphers, 2009. *EUROCRYPT 2009 Poster-session.*

[BG09b] *Céline Blondeau and Benoît Gérard. On the Data Complexity of Statistical Attacks Against Block Ciphers. In Alexander Kholosha, Eirik Rosnes, and Matthew G. Parker, editors,* Workshop on Coding and Cryptography - WCC 2009*, pages 469–488, 2009.*

[BG10] *Céline Blondeau and Benoît Gérard. Links Between Theoretical and Effective Differential Probabilities : Experiments on PRESENT. In* TOOLS'10*, 2010.* `http://eprint.iacr.org/2010/261`.

[BG11] *Céline Blondeau and Benoît Gérard. Multiple Differential Cryptanalysis : Theory and Practice. In Antoine Joux, editor,* Fast Software Encryption, FSE 2011*, volume 6733 of* Lecture Notes in Computer Science*, pages 35–54. Springer, 2011.*

[BGT11] *Céline Blondeau, Benoît Gérard, and Jean-Pierre Tillich. Accurate estimates of the data complexity and success probability for various cryptanalyses.* Designs, Codes and Cryptography*, 59(1-3) :3–34, 2011.*

[Blo08] *Céline Blondeau.* La cryptanalyse différentielle tronquée*. Rapport de stage de master, Université de Limoges, septembre 2008.*

Première partie

Les attaques différentielles

Chapitre 1

Introduction

La cryptographie est traditionnellement utilisée pour dissimuler des messages aux yeux de certains utilisateurs. Cette utilisation a aujourd'hui un intérêt d'autant plus grand que les communications via internet circulent dans des réseaux dont on ne peut garantir la fiabilité et la confidentialité. Désormais, la cryptographie sert non seulement à préserver la confidentialité des données mais aussi à garantir leur intégrité et leur authenticité.

Soit Alice et Bob, deux interlocuteurs qui veulent s'échanger des messages chiffrés. Il existe deux grandes familles de cryptographie, la *cryptographie asymétrique* et *la cryptographie symétrique*.

La cryptographie asymétrique est aussi appelée *cryptographie à clé publique*. Dans ce modèle, Alice choisit une clé publique et la clé privée correspondante et diffuse la clé publique. Bob peut alors, à l'aide de cette clé publique, chiffrer les messages de son choix et les envoyer à Alice qui est la seule à pouvoir les déchiffrer (car elle est la seule à posséder la clé privée).

La cryptographie à clé publique consiste alors à construire un système de chiffrement pour lequel il est "dur" en connaissant la clé publique de retrouver la clé privée.

A l'opposé de la cryptographie asymétrique, il y a la cryptographie symétrique. Celle-ci est aussi appelée *cryptographie à clé secrète*. Dans ce modèle, Alice et Bob partagent la même clé. C'est-à-dire que la même clé est utilisée pour le chiffrement et le déchiffrement. Dans ce modèle, chaque couple d'interlocuteurs a alors besoin de posséder son propre jeu de clé pour pouvoir s'échanger des messages chiffrés.

Dans cette thèse, nous nous intéressons uniquement à la cryptographie symétrique. Dans cette famille, nous pouvons distinguer plusieurs constructions différentes.

1.1 La cryptographie symétrique

Dans la famille de la cryptographie symétrique, nous pouvons distinguer plusieurs méthodes de chiffrement, d'intégrité ou d'authentification. Nous détaillons ici brièvement certaines de ces constructions.

Les systèmes de chiffrement à flot

Les systèmes de *chiffrement à flot* sont aussi appelés *chiffrement à la volée*. Dans un système de chiffrement à flot, la clé utilisée est une suite chiffrante que l'on additionne bit à bit au message clair. La sécurité des systèmes de chiffrement à flot repose alors sur le comportement aléatoire de la suite chiffrante. Les systèmes de chiffrement à flot sont des systèmes de chiffrement rapides et bien adaptés pour les implémentations matérielles courantes.

Les systèmes de chiffrement par bloc

Parallèlement aux systèmes de chiffrement à flot, où le chiffrement se fait à la volée, on définit ce que l'on appelle *système de chiffrement par bloc*.

Pour chiffrer un message en utilisant un système de chiffrement par bloc, on découpe ce message en blocs [1] et on applique une fonction de chiffrement E à chacun de ces blocs. Cette fonction de chiffrement est paramétrée par une clé secrète K. Un mode opératoire nous permet alors de définir le lien entre le clair et le chiffré d'un bloc avec ceux des autres blocs. Dans la suite de cette thèse, on étudie plus particulièrement cette famille de systèmes de chiffrement symétrique. Ainsi une étude plus détaillée sur les systèmes de chiffrement par bloc est donnée dans les chapitres suivants.

Les fonctions de hachage

Une fonction de hachage est une fonction qui prend en entrée une chaîne de longueur arbitraire finie et qui retourne un haché de longueur fixe [2]. Soit h une fonction de hachage :

$h : \{0,1\}^* \rightarrow \{0,1\}^m$ où m est la taille du haché et $\{0,1\}^*$ signifie ensemble des chaînes binaires de taille quelconque.

Les fonctions de hachage ne sont pas des systèmes de chiffrement puisqu'elles ne nécessitent pas l'utilisation d'une clé. Et pourtant, on les classe naturellement dans la famille des algorithmes symétriques car la plupart du temps, elles sont construites à partir de primitives dérivées de systèmes de chiffrement par bloc ou de systèmes de chiffrement à flot.

Code d'authentification de message

Ce type d'application cryptographique est plus connue sous le nom anglais *"message authentification code"* ou encore sous l'abréviation MAC. Les codes d'authentification permettent d'assurer l'intégrité du message ainsi que de l'authentifier. Lors de l'envoi du message on y ajoute un code d'authentification qui est l'empreinte du message obtenue grâce à un haché paramétré par une clé.

Dans cette thèse on s'intéresse plus particulièrement aux systèmes de chiffrement par bloc, et en particulier à la primitive utilisée pour chiffrer un bloc.

1. Dans la pratique, les blocs sont de taille comprise entre 64 et 256 bits.
2. Pour les fonctions de hachage classiques le haché est compris entre 128 et 512 bits

1.2 Les systèmes de chiffrement par bloc

1.2.1 Définition

Chiffrement par bloc

Définition 1.1. *Soit X un message clair de taille m. Soit K la clé utilisée pour chiffrer ce message. Nous notons Ω le nombre de bits de cette clé. Un* **système de chiffrement par bloc** *E est une fonction qui dépend de la clé et du message :*

$$\begin{array}{rccc} E: & \mathbb{F}_2^\Omega \times \mathbb{F}_2^m & \to & \mathbb{F}_2^m \\ & (K, X) & \mapsto & Y = E(K, X). \end{array}$$

La clé K utilisée pour chiffrer les messages est appelée **clé maître***. Lorsque celle-ci est fixée, on note E_K le système de chiffrement paramétré par la clé K :*

$$\begin{array}{rccc} E_K: & \mathbb{F}_2^m & \to & \mathbb{F}_2^m \\ & X & \mapsto & E_K(X) \stackrel{\text{def}}{=} E(K, X) \end{array}$$

Afin de pouvoir déchiffrer de façon unique les messages chiffrés obtenus, on demande à la fonction de chiffrement E_K d'être une bijection de l'espace $\mathbb{F}_2^m$.

Définition 1.2. *Soit K une clé maître. Soit E_K un système de chiffrement paramétré par K. La fonction de déchiffrement notée D_K est la fonction réciproque de E_K :*

$$\begin{array}{rccc} D_K: & \mathbb{F}_2^m & \to & \mathbb{F}_2^m \\ & Y & \mapsto & D_K(Y) \end{array}$$

Par la définition même de la fonction de déchiffrement, si X est un message de $\mathbb{F}_2^m$ et D_K est la fonction de déchiffrement associée à la fonction de chiffrement E_K, on a la propriété suivante :

$$D_K(E_K(X)) = X.$$

Système de chiffrement itératif

Les systèmes de chiffrement par bloc actuels sont *itératifs*. C'est-à-dire qu'une même fonction est utilisée plusieurs fois de façon itérative avec une clé différente à chaque tour. Chaque clé est appelée *clé de tour*. Il est courant d'utiliser un algorithme de *cadencement de clé* pour dériver les clés de tour à partir de la clé maître.

Définition 1.3. Un algorithme de cadencement de clé *relié au système de chiffrement itératif est une fonction qui permet à partir d'une clé maître K de dériver des clés de taille égale ou inférieure. Ces clés sont appelées* **clés de tours***. On note par K_i la clé du tour numéro i.*

La structure de l'algorithme de cadencement de clé est détaillée dans la section 1.3.3. Les fonctions que l'on appelle fonctions de tour sont des fonctions paramétrées par les clés de tour K_i (et donc indirectement par la clé maître K).

Définition 1.4. *Soit m le nombre de bits du message clair. Soit K_i une clé de tour dérivée d'une clé maître K. Une fonction de tour au tour i ($1 \leq i \leq r$) paramétrée par la clé de tour K_i est une fonction définie de la façon suivante :*

$$\begin{array}{rccc} F_{K_i} : & \mathbb{F}_2^m & \to & \mathbb{F}_2^m \\ & X & \mapsto & F_{K_i}(X). \end{array}$$

Un système de chiffrement itératif est alors la composition de ses fonctions de tour.

Définition 1.5. *Soit K une clé maître. Soient $(K_1, \cdots K_r)$ les clés de tour dérivées de la clé maître par un algorithme de cadencement de clés. Soient $(F_{K_i})_i$ les fonctions de tour paramétrées par les clés de tour. Le système de chiffrement itératif E_K utilisant la fonction de tour F est défini par*

$$E_K = F_{K_r} \circ F_{K_{r-1}} \circ \cdots \circ F_{K_2} \circ F_{K_1}.$$

Dans les cryptanalyses, nous avons besoin d'étudier un nombre réduit de tours du système de chiffrement. Par abus de notation, nous notons par F_K^j la composition de j tours de la fonction de tour. On a

$$F_K^j = F_{K_j} \circ \cdots \circ F_{K_1},$$

où $(K_j, \cdots K_1)$ sont les clés de tour dérivées de la clé maître K.

Dans les systèmes de chiffrement par bloc itératifs, les deux principales familles sont : les primitives utilisant les schémas de type *Feistel* et les primitives utilisant les schémas de type *substitution-permutation*. La suite de cette section est dédiée à la présentation de ces deux types de systèmes de chiffrement.

1.2.2 Chiffrement de type Feistel et ses généralisations

Les schémas de Feistel ont été introduits par Horst Feistel au début des années 70, pour la conception de système de chiffrement LUCIFER. Ce système de chiffrement a servi de base à la conception du système de chiffrement très connu *"data encryption standard"* (DES) [DES77] qui a été standardisé en 1977. Par la suite de nombreux algorithmes ont repris cette structure qui a pris le nom de son auteur.
Dans un schéma de Feistel simple, le message de taille m (m pair), est divisé en deux parties de taille $m/2$. Nous notons par $X_0^{(g)}$ et $X_0^{(d)}$ les deux parties du message clair. Le message clair est alors la concaténation de $X_0^{(g)}$ et $X_0^{(d)}$ [3] :

$$X = X_0^{(g)} || X_0^{(d)}.$$

De la même façon, nous notons $X_i^{(g)}$ et $X_i^{(d)}$ les deux parties de l'état interne après i tours du système de chiffrement. La fonction de tour d'un schéma de Feistel classique (voir figure 1.1) est alors définie de la façon suivante :

3. les notations (g) et (d) sont utilisées pour symboliser les parties gauche et droite.

Définition 1.6. *Un* **chiffrement de Feistel** *est un système de chiffrement par bloc itératif opérant sur des bloc de taille m, m pair. Au tour i, la fonction de tour, paramétrée par la clé K_i, est définie par*

$$\begin{aligned} F_{K_i} : \ \mathbb{F}_2^{m/2} \times \mathbb{F}_2^{m/2} &\rightarrow \mathbb{F}_2^{m/2} \times \mathbb{F}_2^{m/2} \\ (X_{i-1}^{(g)}, X_{i-1}^{(d)}) &\mapsto (X_i^{(g)}, X_i^{(d)}) \end{aligned}$$

où

$$X_i^{(g)} = X_{i-1}^{(d)},$$
$$X_i^{(d)} = X_{i-1}^{(g)} \oplus f(X_{i-1}^{(d)}, K_i)$$

et f est une fonction interne non nécessairement inversible.

Dans un schéma de Feistel pour déchiffrer il suffit d'utiliser le même processus en inversant l'ordre des clés de tour. En effet pour les schémas de Feistel sur r tours, on a la propriété simple que $X_{i-1}^{(d)} = X_i^{(g)}$ et $X_{i-1}^{(g)} = X_i^{(d)} \oplus f(X_i^{(g)}, K_i)$.

Ainsi dans un schéma de Feistel on ne demande pas nécessairement à la fonction interne f d'être une permutation.

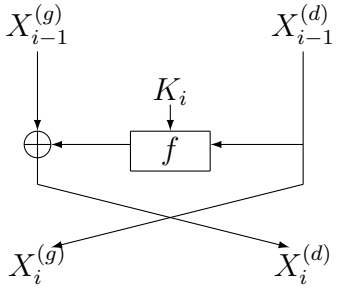

FIGURE 1.1 – Fonction de tour d'un schéma de Feistel

Il y a de nombreuses généralisations des schémas de Feistel.

Le schéma de Feistel généralisé introduit par Kaisa Nyberg en 1994 [Nyb94](*"Generalized Feistel Network"*) fait partie de cette grande famille. Il divise le message en parties plus petites. Chaque partie du message constitue une branche de la fonction de tour du Feistel.

Définition 1.7. *Soit $m = 2\lambda s$. Soit X l'état interne du système de chiffrement. On note par $X = (x^{(1)}, \cdots, x^{(2\lambda)})$, le découpage de X en mots de taille s. Soit λ fonctions internes définies par $f_j : \mathbb{F}_2^s \rightarrow \mathbb{F}_2^s$ pour $(1 \leq j \leq \lambda)$. Soit X l'entrée de taille $2s\lambda$ divisée en 2λ blocs de s bits. La sortie de la* **fonction de tour d'un schéma de Feistel généralisé** *est notée Y et est définie pour $j = 1 \cdots 2\lambda$ par*

$$\begin{aligned} z^{(j)} &= X^{(j)} \oplus f_j(X^{(2s-j)} \oplus K_i{}^{(j)}) \text{ pour } j = 1 \cdots \lambda \\ z^{(j)} &= X^{(j)} \text{ pour } j = \lambda + 1, ..., 2\lambda \\ Y^{(j)} &= z^{(j-1)} \text{ pour } j = 2, ..., 2\lambda \\ Y^{(1)} &= z^{(2\lambda)} \end{aligned}$$

Le cas particulier d'un schéma de Feistel généralisé divisé en 8 branches est décrit dans la figure 1.2.

Un autre exemple de généralisation des schémas de Feistel consiste à utiliser des branches de tailles différentes. C'est le cas par exemple du système de chiffrement Misty

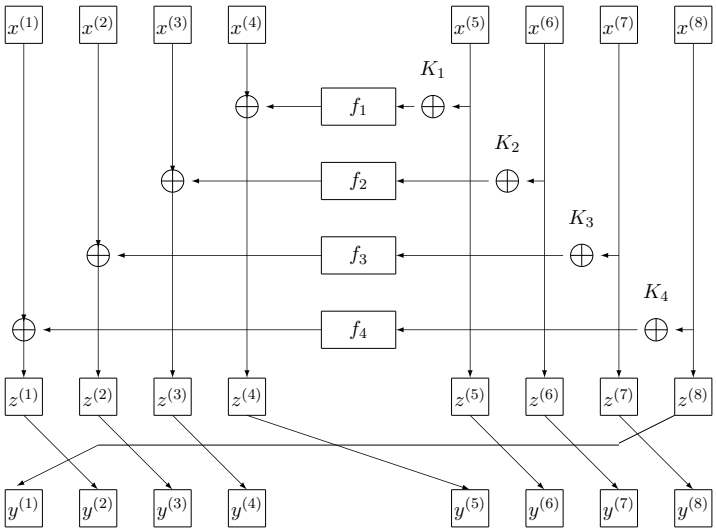

FIGURE 1.2 – Fonction de tour d'un schéma de Feistel généralisé comme introduit par Kaisa Nyberg [Nyb94] (cas particulier où le message est divisé en 8)

[Mat97] qui comporte un schéma de Feistel interne de taille 16 bits divisé en 7+9 bits et une fonction interne différente suivant le nombre de bits.

Il existe de nombreuses autres généralisations des schémas de Feistel. Ceux-ci comportent principalement l'avantage d'utiliser une primitive similaire pour le chiffrement et le déchiffrement.

1.2.3 Chiffrement de type substitution-permutation

Une autre grande famille de systèmes de chiffrement itératif que nous allons étudier sont les schémas de *type substitution-permutation* (SPN : *"Substitution Permutation Network"*).

Définition 1.8. *Un système de chiffrement par bloc itératif est dit de type* **substitution-permutation** *si la fonction de tour peut se décomposer en trois grandes étapes : une étape dite d'ajout de clé, une étape de substitution qui est non-linéaire et une étape de permutation qui est linéaire.*

Afin de permettre le déchiffrement la fonction de tour doit être une bijection.

La figure 1.3 représente les trois étapes importantes de la fonction de tour d'un système de chiffrement de type substitution-permutation.

Cette définition très large nous permet de faire rentrer un grand nombre de systèmes de chiffrement dans la catégorie des SPN.

L'exemple le plus connu de système de chiffrement de ce type est "Rijndael". Plusieurs versions ont été standardisées par le NIST en 2000 [DR99] sous le nom de *"Advanced encryption standard"* (AES). Une description de cet algorithme est donnée dans la section 1.4.2.

Outre l'AES, dans cette thèse nous nous sommes particulièrement intéressés au système de chiffrement PRESENT [BKL+07]. Tout au long de ce manuscrit nous faisons référence à ce système de chiffrement de type substitution-permutation. Une description de cet algorithme est faite dans la section 1.4.1.

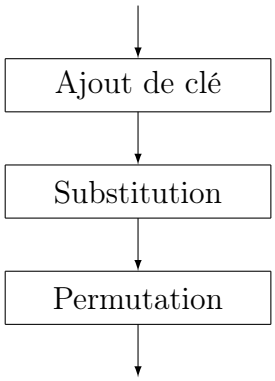

FIGURE 1.3 – Fonction de tour d'un SPN

1.3 Les différentes primitives

Les systèmes de chiffrements par bloc comportent en général deux parties. Une partie de *confusion* et une partie de *diffusion*.

1.3.1 La partie de confusion

La partie de *confusion* du système de chiffrement est la seule partie non-linéaire du système de chiffrement par bloc. Dans un système de chiffrement de type substitution-permutation elle correspond à la partie de substitution. Dans cette thèse on s'intéresse aux systèmes de chiffrement pour lesquels l'état interne de la partie de substitution est divisé en mots de petite taille [4]. Une application non-linéaire est appliquée en parallèle à chacun de ces mots. Cette application non-linéaire qui peut être la même ou être différente pour chacun des mots du système de chiffrement est appelée *boîte-S*. Dans les systèmes de chiffrement de type substitution-permutation la fonction de tour doit être bijective afin de pouvoir déchiffrer. Ainsi les boîtes-S qui composent ce système de chiffrement sont toujours bijectives. Dans les systèmes de chiffrement de type Feistel nous n'avons pas besoin de cette contrainte de bijectivité. Pour le DES par exemple les boîtes-S sont de 6 bits vers 4. Mais pour des soucis d'implémentation la plupart des systèmes de chiffrement comportent des boîtes-S inversibles.

Il existe plusieurs façons de définir les boîtes-S. On peut, par exemple, comme dans le cas de l'AES (section 1.4.2), la définir en identifiant l'espace vectoriel $\mathbb{F}_2^n$ au corps fini $\mathbb{F}_{2^n}$ et définir une permutation sur ce corps fini. L'autre méthode tout aussi utilisée pour définir une boîte-S est de donner l'image point par point de la fonction (c'est le cas par exemple de la définition de la boîte-S de PRESENT donnée dans le tableau 1.1).

Dans la première partie de cette thèse on s'intéresse très peu à la façon dont sont construites les boîtes-S. On admettra donc que l'on possède une table avec les correspondances entre les entrées et les sorties. La deuxième partie de cette thèse est entièrement consacrée à l'étude de certaines propriétés de ces fonctions (voir partie II).

4. de l'ordre de 4 ou 8 bits pour les systèmes de chiffrement actuels.

1.3.2 La partie de diffusion

Dans un système de chiffrement de type substitution-permutation cette partie de *diffusion* correspond à la permutation. Il existe plusieurs manières de diffuser l'information entre les tours.

Dans le système de chiffrement PRESENT (voir la description de PRESENT dans la section 1.4.1), la permutation effectuée est une permutation dite *"bit à bit"*.

Un autre exemple de permutation que l'on peut citer est la permutation dite *"mot à mot"* qui mélange des ensembles de bits de taille fixe. C'est ce type de permutation qui est faite dans la fonction "ShiftRows" et "MixColumns" de l'AES (voir la description de l'AES dans la section 1.4.2).

Pour les schémas de Feistel, les deux types de permutations citées précédemment sont utilisés dans la partie linéaire de la fonction interne.

1.3.3 Le cadencement de clé

Dans les systèmes de chiffrement par bloc itératifs, pour chaque tour, la fonction de tour F est paramétrée par une clé de tour K_i $(1 \leq i \leq r)$. Les concepteurs des systèmes de chiffrement actuels aimeraient que les clés de tours soient indépendantes entre elles pour pouvoir obtenir des critères de sécurité pour leur système de chiffrement contre les attaques connues. Or cela nécessiterait que la clé maître soit la concaténation de toutes les clés de tours. En pratique cela n'est pas possible car la taille de la clé maître serait alors beaucoup trop grande. Dans les algorithmes de chiffrement par bloc la taille actuelle des clés maîtres ne dépasse pas 256 bits [5]. Les clés de tour sont alors dérivées de la clé maître par un algorithme de *cadencement de clé*.

Dans cette thèse on ne s'intéresse qu'aux systèmes de chiffrements par bloc où la clé de tour est ajoutée par un "ou" exclusif. Ce type de système de chiffrement est appelé *"key-alternating cipher"* [DR05].

1.4 Quelques systèmes de chiffrement par bloc

A l'heure actuelle, il existe une grande variété de systèmes de chiffrement par bloc. Mon attention durant cette thèse s'est porté plus particulierement sur deux systèmes de chiffrement de type substitution-permutation.
Le premier est l'AES, puisque c'est le standard du NIST actuellement très répandu et très utilisé.

Le second est l'algorithme de chiffrement PRESENT. En effet grâce à sa structure simple à comprendre et les différentes versions réduites existantes, nous avons pu mener un certain nombre d'expérimentations sur ce système de chiffrement. Ce système de chiffrement est très étudié à l'heure actuelle.

5. Les critères de sécurité actuels recommandent des clés de taille supérieure à 80 bits

1.4.1 PRESENT et SMALLPRESENT-[s]

Le système de chiffrement PRESENT

Le système de chiffrement PRESENT [BKL+07] a été introduit à la conférence CHES [6] en 2007. Ce système de chiffrement fait partie de la nouvelle génération des systèmes de chiffrement par bloc qui sont dits à bas coût (*"lightweight block cipher"*). Il chiffre des blocs de 64 bits au moyen d'un algorithme qui suit le modèle d'un schéma de type substitution-permutation. Il existe deux versions qui dépendent de la taille de la clé maître (80 bits ou 128 bits). Ce système de chiffrement qui utilise 32 clés de tours se décompose en 31 tours avec un ajout de clé supplémentaire à la fin du dernier tour. La fonction de tour F_{K_i} au tour i ($1 \leq i \leq 31$) se décompose de la façon suivante :

Ajout de la clé de tour : Cette étape consiste en l'addition bit à bit de la clé de tour et de l'état interne.

La substitution : Pour cette partie de substitution, l'état interne de 64 bits est divisé en mots de 4 bits. Les 16 mots sont modifiés par passage dans une petite boîte-S. La boîte-S de PRESENT, qui est la même pour les 16 mots, effectue une bijection de $\mathbb{F}_2^4$. Dans la suite de cette thèse on utilise le préfixe `0x` pour indiquer une notation hexadécimale (notation en base 16). Par exemple le chiffre 13 est représenté par `0xd` et le chiffre $52 = 3 \cdot 16 + 4$ est représenté par `0x34`. La boîte-S de PRESENT, notée S, est définie dans le tableau 1.1.

x	`0x0` `0x1` `0x2` `0x3` `0x4` `0x5` `0x6` `0x7` `0x8` `0x9` `0xa` `0xb` `0xc` `0xd` `0xe` `0xf`
S(x)	`0xc` `0x5` `0x6` `0xb` `0x9` `0x0` `0xa` `0xd` `0x3` `0xe` `0xf` `0x8` `0x4` `0x7` `0x1` `0x2`

TABLE 1.1 – La boîte-S de PRESENT

La permutation : La permutation est une permutation bit à bit. Elle consiste à changer la place des bits de l'état interne. Dans le tableau 1.2 décrivant la permutation, la valeur i correspond à la position du bit et $P(i)$ correspond à sa position après la permutation ($0 \leq i \leq 63$).

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$P(i)$	0	16	32	48	1	17	33	49	2	18	34	50	3	19	35	51
i	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
$P(i)$	4	20	36	52	5	21	37	53	6	22	38	54	7	23	39	55
i	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
$P(i)$	8	24	40	56	9	25	41	57	10	26	42	58	11	27	43	59
i	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
$P(i)$	12	28	44	60	13	29	45	61	14	30	46	62	15	31	47	63

TABLE 1.2 – La permutation de PRESENT

La fonction de tour de PRESENT est représentée sur la figure 1.4.

6. CHES : *"Cryptographic Hardware and Embedded Systems"*

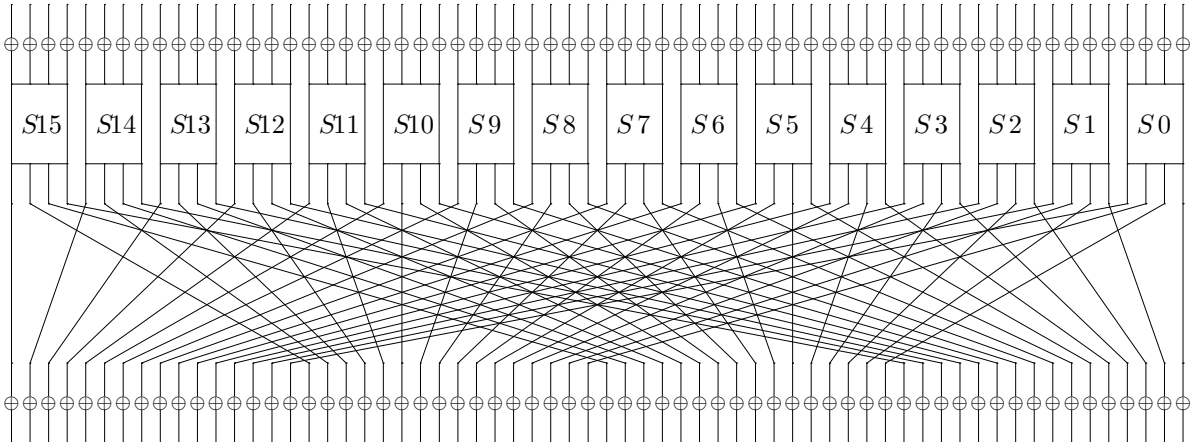

FIGURE 1.4 – Fonction de tour de PRESENT

Soit X l'état interne après $i-1$ tours du système de chiffrement. Soit K_i la clé du tour i. Les différentes étapes de la fonction de tour s'enchaînent de la façon suivante :

- Addition de la clé de tour : $Y = X \oplus K_i$.
- Substitution : Soit $Y = (Y^{(63)}, \cdots, Y^{(0)})$ le découpage bit à bit de Y. On applique la boîte-S à chacun des mots de 4 bits. Pour $0 \leq j \leq 15$ on a :

$$[Z^{(4*j+3)}, Z^{(4*j+2)}, Z^{(4*j+1)}, Z^{(4*j)}] = S[Y^{(4*j+3)}, Y^{(4*j+2)}, Y^{(4*j+1)}, Y^{(4*j)}].$$

- Permutation : l'état à la sortie de la fonction de tour est alors $P(Z)$ où P est la permutation définie dans le tableau 1.2.

Les versions réduites : SMALLPRESENT-[s]

Les systèmes de chiffrement SMALLPRESENT-[s] ($0 < s \leq 16$) sont des versions réduites de PRESENT. Les spécifications sont données dans [Lea10]. La boîte-S utilisée pour le système de chiffrement est la même que celle de PRESENT (tableau 1.1). La valeur de s nous donne le nombre de boîtes-S utilisées pour le chiffrement. Ainsi comme la boîte-S transforme des mots de 4 bits, la taille du message clair est $4s$ bits. La permutation utilisée est une mise à l'échelle de celle utilisée pour PRESENT. Les fonctions de tour de SMALLPRESENT-[4] et SMALLPRESENT-[8] sont représentées sur la figure 1.5. Ces deux versions sont utilisées dans les chapitres suivants pour mener des expérimentations. SMALLPRESENT-[4] chiffre des message de 16 bits et SMALLPRESENT-[8] chiffre des messages de 32 bits.

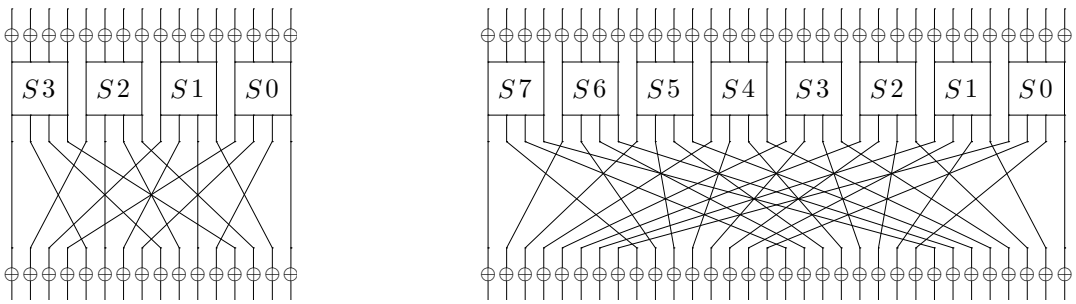

FIGURE 1.5 – Fonction de tour de SMALLPRESENT-[4] et SMALLPRESENT-[8].

Les algorithmes de cadencement de clé

Comme nous l'avons défini dans la définition 1.3, un algorithme de cadencement de clé sert à dériver des clés de tours à partir d'une clé maître K. Dans la version de PRESENT présentée en 2007, deux algorithmes de cadencement de clé sont décrits. Un pour des clés de 80 bits et un pour des clés de 128 bits. Nous décrivons ici l'algorithme de cadencement de clé pour une clé maître de 80 bits.

Le registre de clé K est représenté bit à bit par $K^{(79)}K^{(78)}\cdots K^{(0)}$. Au tour i, les 64 bits de la clé de tour $K_i = {K_i}^{(63)}{K_i}^{(62)}\cdots {K_i}^{(0)}$ correspondent aux 64 bits les plus à gauche du registre K. Ainsi au tour i nous avons $K_i = {K_i}^{(63)}{K_i}^{(62)}\cdots {K_i}^{(0)} = K^{(79)}K^{(78)}\cdots K^{(16)}$. Par la suite le registre est mis à jour de la façon suivante :

1. $[K^{(79)}K^{(78)}\cdots K^{(61)}K^{(60)}\cdots K^{(0)}] = [K^{(18)}K^{(17)}\cdots K^{(0)}K^{(79)}\cdots K^{(20)}K^{(19)}]$ (rotation de 19 bits),
2. $[K^{(79)}K^{(78)}K^{(77)}K^{(76)}] = S[K^{(79)}K^{(78)}K^{(77)}K^{(76)}]$ (passage des 4 bits de poids fort dans la boîte-S),
3. $[K^{(19)}K^{(18)}K^{(17)}K^{(16)}K^{(15)}] = [K^{(19)}K^{(18)}K^{(17)}K^{(16)}K^{(15)}] \oplus \mathit{compteur}$ (ajout d'un compteur).

La valeur de *compteur* est donnée par le numéro du tour.

L'algorithme de cadencement de clé pour la clé de 128 bits est décrit dans [BKL+07].

Lorsqu'il introduit SMALLPRESENT-[s], Gregor Leander propose d'utiliser les clés de 80 bits pour toutes les versions réduites. Pour nos expérimentations (voir chapitre 4, chapitre 5 et chapitre 6) nous avons trouvé intéressant de mettre aussi à l'échelle la clé maître et l'algorithme de cadencement de clé, c'est-à-dire d'avoir des tailles de clé maître du même ordre de grandeur que le nombre de bits du message. Durant cette thèse on s'est intéressé plus particulièrement à deux des versions réduites de PRESENT : SMALLPRESENT-[4] et SMALLPRESENT-[8]. Pour cette raison dans le tableau 1.3 nous décrivons l'algorithme de cadencement de clé que nous avons utilisé pour des clés maîtres de 20 bits et 40 bits.

Clé de 20 bits :

$[K^{(19)}K^{(18)}\cdots K^{(1)}K^{(0)}]$	=	$[K^{(6)}K^{(5)}\cdots K^{(8)}K^{(7)}]$	rotation de 7 bits
$[K^{(19)}K^{(18)}K^{(17)}K^{(16)}]$	=	$S[K^{(19)}K^{(18)}K^{(17)}K^{(16)}]$	passage dans la boîte-S
$[K^{(7)}K^{(6)}K^{(5)}K^{(4)}K^{(3)}]$	=	$[K^{(7)}K^{(6)}K^{(5)}K^{(4)}K^{(3)}] \oplus \mathit{compteur}$	ajout du compteur

Clé de 40 bits :

$[K^{(39)}K^{(38)}\cdots K^{(1)}K^{(0)}]$	=	$[K^{(10)}K^{(9)}\ldots K^{(12)}K^{(11)}]$	rotation de 11 bits
$[K^{(39)}K^{(38)}K^{(37)}K^{(36)}]$	=	$S[K^{(39)}K^{(38)}K^{(37)}K^{(36)}]$	passage dans la boîte-S
$[K^{(11)}K^{(10)}K^{(9)}K^{(8)}K^{(7)}]$	=	$[K^{(11)}K^{(10)}K^{(9)}K^{(8)}K^{(7)}] \oplus \mathit{compteur}$	ajout du compteur

TABLE 1.3 – Algorithmes de cadencement de clé pour SMALLPRESENT-[4] et SMALLPRESENT-[8].

1.4.2 Rijndael

Rijndael est un standard de chiffrement par bloc. Il a été inventé par Vincent Rijmen et Joan Daemen. En 2000 certaines versions ont été choisies par le NIST pour devenir l'*"Advanced Encryption Standard"* (AES). Dans la norme NIST FIPS 197 il permet de chiffrer des blocs de 128 bits au moyen d'une clé maître de taille variable : 128, 192 ou 256 bits. Le nombre d'itérations de la fonction de tour dépend de la taille de la clé : le standard précise que pour la clé de 128 bits on applique 10 fois la fonction de tour (sans le dernier "MixColumns"), alors qu'il faut 14 tours si on utilise la clé de 256 bits Un mot de 128 bits est découpé en 16 mots de 8 bits et est représenté sous forme d'une matrice 4×4 d'octets.

La fonction de tour est décomposée en 4 opérations simples sur la matrice :

SubBytes : C'est une opération de substitution qui consiste à appliquer parallèlement à chaque octet de l'entrée une boîte-S. Ici la boîte-S est une permutation de l'espace vectoriel $\mathbb{F}_2^8$. Elle identifie chaque mot de 8 bits à un élément du corps

$$\mathbb{F}_{2^8} = \mathbb{F}_2[X]/\langle X^8 + X^4 + X^3 + X + 1\rangle$$

par l'isomorphisme suivant :

$$(x_0, x_1, ..., x_7) \in \mathbb{F}_2^8 \mapsto \bigoplus_{i=0}^{7} x_i X^i.$$

Les opérations effectuées sur les polynômes sont définies modulo le polynôme irréductible $X^8 + X^4 + X^3 + X + 1$. Cette boite S est composée de la fonction inverse dans le corps fini $\mathbb{F}_{2^8}$:

$$x \in \mathbb{F}_{2^8} \mapsto x^{254}$$

avec la fonction affine dans l'espace vectoriel $\mathbb{F}_2^8$ (bijective) décrite ci dessous :

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

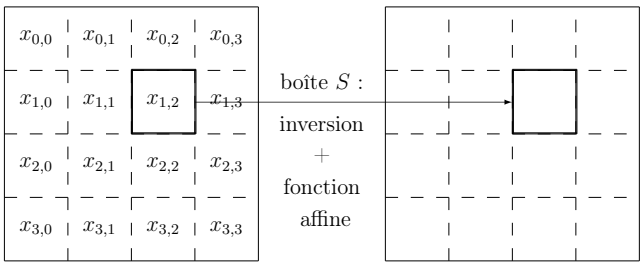

FIGURE 1.6 – Fonction SubBytes

ShiftRows : C'est une rotation des lignes de la matrice : la ligne i, $0 \leq i \leq 3$ est décalée de i octets vers la gauche.

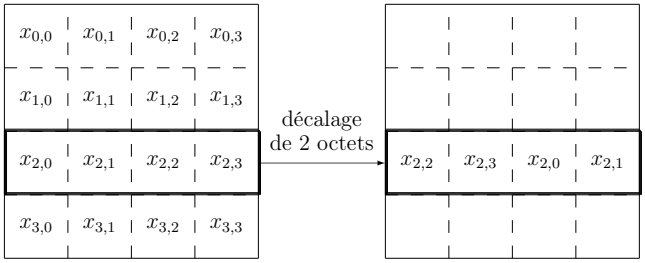

FIGURE 1.7 – Fonction ShiftRows

MixColumns : C'est une transformation linéaire appliquée en parallèle aux 4 colonnes de la matrice. Chaque colonne subit alors la transformation suivante :

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} \alpha & \alpha+1 & 1 & 1 \\ 1 & \alpha & \alpha+1 & 1 \\ 1 & 1 & \alpha & \alpha+1 \\ \alpha+1 & 1 & 1 & \alpha \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

où α est racine de $X^8 + X^4 + X^3 + X + 1$ et $x_0, ..., x_3, y_0, ..., y_3$ sont des octets en entrée et en sortie respectivement.

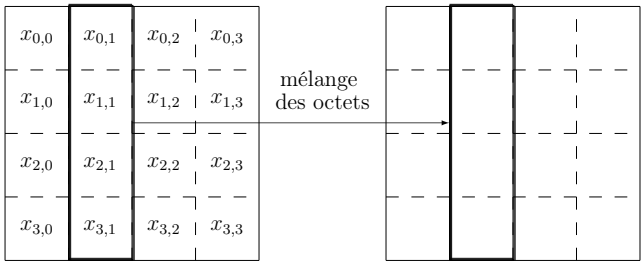

FIGURE 1.8 – Fonction MixColumns

AddRoundKey : C'est l'insertion de la clé de tour par un ou exclusif bit à bit.

1.5 Les attaques statistiques

1.5.1 Introduction

Attaquer un système de chiffrement par bloc consiste en général à retrouver de l'information sur la clé ayant servi à chiffrer. Si nous excluons les attaques physiques, il existe deux grandes familles d'attaques structurelles sur les systèmes de chiffrement par bloc : les *attaques algébriques* et les *attaques statistiques*.

L'approche algébrique consiste à essayer de reconstruire le système de chiffrement comme un polynôme qui dépend de la clé. On peut aussi décrire l'algorithme par un système d'équations qui dépend du message clair, du chiffré correspondant ainsi que des

bits de la clé. Cette méthode qui n'est pas très efficace pour les systèmes de chiffrement par bloc actuels peut être alliée à d'autres méthodes pour retrouver de l'information sur la clé d'un système de chiffrement comportant plus de tours. Un mélange entre l'observation d'un phénomène statistique combiné avec une attaque algébrique peut être utilisé. On obtient alors des équations algébriques probabilistes. Si on prend l'exemple de l'attaque différentielle [7] qui sera décrite dans la section 2.1 on peut la combiner avec une attaque algébrique. C'est le cas par exemple de l'attaque différentielle algébrique introduite à FSE en 2009 par Martin Albrecht et Carlos Cid [AC09].

L'approche statistique consiste, à partir de l'observation d'un comportement non-aléatoire du système de chiffrement par bloc, à retrouver de l'information sur la clé. C'est ce type d'attaque qui est détaillé dans les parties suivantes.

Dans un premier temps avant de chercher à retrouver de l'information sur la clé nous avons besoin de définir la notion de distingueur.

Définition 1.9. *On appelle* **distingueur** *un algorithme qui cherche par un jeu de questions/réponses à distinguer un système de chiffrement particulier d'une permutation idéale c'est à dire ayant le comportement d'une permutation tirée au hasard selon la distribution uniforme.*

Les attaques statistiques peuvent alors être divisées en deux parties. Les attaques dites *"faibles"* qui consistent à observer un comportement non-aléatoire d'un système de chiffrement sans donner d'information sur la clé utilisée (ces attaques sont appelées *"distinguishing attacks"*). Et les attaques plus *"fortes"* qui consistent à retrouver de l'information sur la clé utilisée pour chiffrer. Dans beaucoup d'attaques statistiques la première étape consiste à construire un distingueur avant de pouvoir récupérer de l'information sur la clé.

Afin de pouvoir distinguer le système de chiffrement d'une permutation aléatoire nous récupérons de l'information venant de différents couples de messages clairs/chiffrés obtenus avec la même clé. La manière dont ces couples sont obtenus détermine le contexte dans lequel les attaques peuvent s'appliquer. Ainsi on distingue différents types d'attaques. On peut citer, par exemple les *attaques à clairs connus* et les *attaques à clairs choisis*. Le premier type d'attaque nécessite juste de pouvoir récupérer les messages clairs avec les chiffrés correspondants. Dans les attaques à clairs choisis l'attaquant doit pouvoir demander les chiffrés correspondant aux messages clairs de son choix.

1.5.2 Les attaques statistiques

Il existe un grand nombre d'attaques statistiques. Les chapitres 2 et 3 détaillent certaines de ces attaques sur les systèmes de chiffrement par bloc. La présentation qui est faite ici se veut générale.

Supposons que nous avons réussi à extraire une propriété particulière $\mathcal{P}$ pour le système de chiffrement que nous sommes en train d'étudier. Une attaque statistique que j'appelle "classique" consiste à tester une partie des clés pour savoir laquelle est la bonne [8].

7. La cryptanalyse différentielle fait partie de la famille des attaques statistiques

8. La plupart des attaques statistiques que nous nous présentons dans cette thèse peuvent être qualifiées de ce que j'appelle "classique". La seule qui ne l'est pas est l'attaque linéaire de type 1 (voir section 3.2)

Dans cette thèse on se limite aux attaques statistiques qui consistent à faire la distinction entre deux types de clés : on suppose en effet que la propriété étudiée a un comportement non-idéal pour la clé testée correspondant à la clé utilisée pour chiffrer et qu'elle a un comportement idéal si la clé testée n'est pas celle qui a été utilisée pour chiffrer. Cette hypothèse est appelée *hypothèse de répartition aléatoire par fausse clé*[9].

Nous allons donc faire la distinction entre les clés utilisées.

Notation 1.1. *Dans le contexte où la clé maître est fixée mais inconnue de l'attaquant, nous notons par K^* la clé maître utilisée pour chiffrer. On appelle* sous-clé ou clé candidate *un ensemble de bits de taille n que nous testons. Une clé candidate est notée k. Si cette clé candidate est celle qui correspond à la clé maître K^*, nous la notons k^*. Dans le cas contraire nous gardons la notation k.*

Dans les cryptanalyses statistiques usuelles on se limite à l'étude d'un test d'hypothèses binaire. Les deux hypothèses sont les suivantes :

$$\begin{cases} H_* : \text{Le candidat testé correspond à la clé maître } K^*. \\ H \ : \text{Le candidat testé ne correspond pas à la clé maître.} \end{cases} \tag{1.1}$$

Ainsi dans les attaques statistiques de ce type nous devons comparer deux distributions de probabilités : si le candidat testé correspond la bonne sous clé, le système de chiffrement présente un biais statistique par rapport à une permutation idéale. Nous noterons par p_* la probabilité de l'événement correspondant à la bonne clé.

On suppose que pour toutes les autres sous clés testées la propriété $\mathcal{P}$ étudiée sur le système de chiffrement a un comportement idéal. Nous notons par p la probabilité de l'événement pour toutes les autres clés. Cette seconde probabilité est égale à la probabilité que la propriété apparaisse dans le cas idéal.

Cette modélisation qui consiste à dire que "toutes les variables aléatoires correspondantes aux mauvais candidats ont le même comportement" repose sur l'hypothèse de répartition aléatoire par fausse clé.

Dans une attaque statistique classique, afin de distinguer le bon candidat des autres l'attaquant a à sa disposition des couples clairs/chiffrés. Un *échantillon* est alors composé d'un certain nombre de ces couples clairs/chiffrés. La taille de l'échantillon dépend du type de cryptanalyse. Nous la détaillons dans les chapitres suivants. On peut toutefois retenir qu'un échantillon peut être égal à un couple clair/chiffré (voir le cas de la cryptanalyse linéaire dans la section 3.2), peut être égal à deux couples clairs/chiffrés (voir le cas de la cryptanalyse différentielle dans la section 2.1) ou encore bien plus (voir par exemple le cas de la cryptanalyse différentielle tronquée dans la section 2.2).

Nous notons par N le nombre d'échantillons dont l'attaquant dispose. À partir de ces échantillons, l'attaquant est capable de générer N variables aléatoires binaires $\mathbf{X}_1; \mathbf{X}_2; \cdots; \mathbf{X}_N$ qui sont définies par

$$\mathbf{X}_i = \begin{cases} 1 & \text{si la propriété } \mathcal{P} \text{ est observée pour l'échantillon } i\,; \\ 0 & \text{sinon.} \end{cases} \tag{1.2}$$

Pour résumer nous avons les notations suivantes :

9. ou *"Wrong-key randomisation hypothesis"* en anglais

Notation 1.2. *Soit H et H_* les hypothèses définies en (1.1). Soit $\mathbf{X}_i$ la variable aléatoire correspondant à l'échantillon numéro i. Les probabilités étudiées sont les suivantes :*

- *Si la clé testée est celle utilisée pour chiffrer, le phénomène apparaît avec probabilité p_*. Ainsi*
$$p_* \stackrel{\text{def}}{=} P\left[\mathbf{X}_i = 1 | H_*\right].$$
- *Si la clé testée n'est pas celle utilisée pour chiffrer, le phénomène apparaît avec probabilité p. Ainsi*
$$p \stackrel{\text{def}}{=} P\left[\mathbf{X}_i = 1 | H\right].$$

Supposons que nous ayons à notre disposition un certain nombre de couples de messages clairs/chiffrés. Ce nombre de messages noté N_{DC} est proportionnel au nombre d'échantillons N. Soit E_{K^*} le système de chiffrement par bloc sur lequel on a observé un biais statistique.

Une attaque statistique se décompose alors en les trois phases suivantes :

Distillation : Dans un premier temps, à partir des N échantillons à disposition, on extrait de l'information sur les clés candidates k que l'on teste.

Analyse : À partir de cette observation, on calcule la chance de chaque clé candidate k et on génère la liste $\mathcal{L}$ des ℓ candidats les plus probables.

Recherche exhaustive : Pour chaque sous clé dans la liste ordonnée $\mathcal{L}$ on teste toutes les clés maîtres correspondantes jusqu'à ce que la bonne soit trouvée.

Ces trois phases, communes à toutes les attaques statistiques, sont légèrement différentes selon le type d'attaque statistique. Une description plus précise de ces phases est faite lors de la présentation de chaque attaque statistique dans les chapitres suivants. On peut cependant décrire plus en détail les attaques statistiques appelées *attaques sur le dernier tour*.

1.5.3 Les attaques sur le dernier tour

Dans les attaques statistiques sur le dernier tour on retrouve en général de l'information sur certaines des clés de tour. Ces clés de tour sont en général des clés du ou des derniers tours du système de chiffrement. Pour simplifier les explications nous supposons ici que l'on cherche à retrouver de l'information sur la clé du dernier tour. On note par k^* la clé candidate correspondant à la clé maître K^* et par k toutes les autres clés (ceci conformément aux notations de la section précédente). En général on n'attaque pas l'algorithme de chiffrement en entier mais une version réduite de celui-ci. Supposons que nous ayons une propriété particulière $\mathcal{P}$ sur r tours du système de chiffrement qui arrive avec probabilité p_*. Alors on peut en général attaquer $r+1$ tours de la façon suivante :

1. On récupère les messages chiffrés que l'on obtient après $r+1$ tours du système de chiffrement.
2. Pour chaque sous-clé candidate, on déchiffre partiellement le dernier tour.
3. Si la clé utilisée pour déchiffrer est la même que celle utilisée pour chiffrer on retrouve le distingueur observé sur r tours du système de chiffrement. Dans le cas contraire, c'est-à-dire si la clé utilisée pour déchiffrer est différente de celle utilisée pour chiffrer, on peut supposer que l'on observe un comportement tout autre, proche de l'aléatoire.

Il semble raisonnable de supposer que pour les mauvais candidats les variables aléatoires correspondantes ont un comportement presque idéal. Puisque dans le cas contraire cela pourrait signifier que l'on peut faire une attaque sur plus de tours.

Ce type d'attaque est résumé dans la figure 1.9.

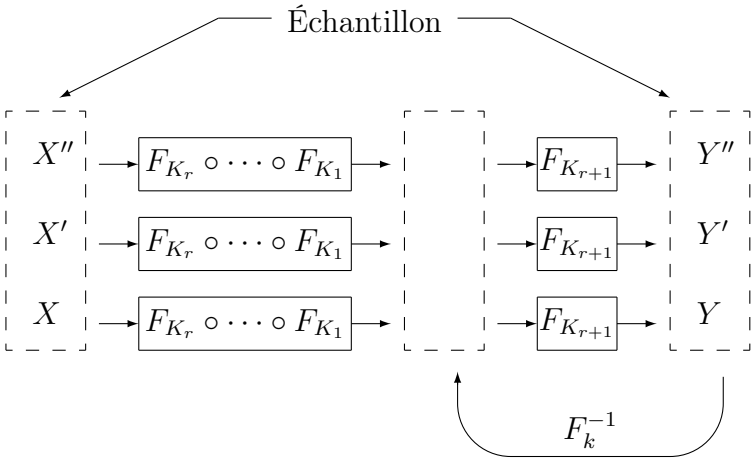

FIGURE 1.9 – Attaque sur le dernier tour

1.5.4 Complexité d'une attaque statistique

Certaines quantités ont besoin d'être calculées afin de pouvoir déterminer la puissance d'une attaque statistique. Dans cette section, nous présentons les principaux outils d'analyse des attaques statistiques. Nous utilisons les mêmes notations que dans les sections précédentes.

Définition 1.10. *Dans les attaques statistiques, nous avons besoin de définir les quantités suivantes :*

La complexité en données : *C'est le nombre de couples de messages clairs/chiffrés dont on a besoin pour obtenir de l'information sur la clé. Ce nombre de messages est noté N_{DC}.*

La probabilité de succès : *C'est la probabilité que la clé candidate k^* soit dans la liste $\mathcal{L}$ des sous clés que l'on garde lors de la phase d'analyse. On note P_S la probabilité de succès de l'attaque.*

$$P_S = P\left[k^* \in \mathcal{L}\right].$$

La complexité en temps : *C'est le nombre d'opérations effectuées par l'algorithme pour retrouver de l'information sur la clé.*

La complexité en mémoire : *C'est la place mémoire dont a besoin l'algorithme pour stocker les quantités nécessaires à l'attaque.*

Il existe plusieurs façons de définir la taille de la liste des clés gardées. Dans cette thèse, on en utilise deux. La première façon consiste à fixer un seuil et à accepter tous les

candidats dont la statistique étudiée a une valeur supérieure au seuil. La seconde consiste pour une taille de liste fixée ℓ, à accepter les ℓ clés candidates les plus probables. Dans la première méthode nous n'avons pas un contrôle direct sur la taille de la liste contrairement à la seconde méthode. Dans le chapitre 5, les deux méthodes sont utilisées afin de calculer la complexité en données et la probabilité de succès d'une attaque statistique.

Les trois premières quantités de la définition précédente sont reliées entre elles. Par exemple il est assez facile de voir que pour une complexité en données fixée, si on augmente la taille de la liste alors la probabilité que la clé candidate k^* soit dans la liste augmente et ainsi la probabilité de succès augmente.

Dans la plupart de ces attaques afin de déterminer les candidats les plus probables on utilise souvent un compteur pour chaque sous clé testée. Ainsi la complexité en mémoire d'une attaque statistique est souvent reliée au stockage de ces compteurs. Pour l'étude de la complexité en temps des attaques statistiques, la phase d'analyse est souvent négligeable par rapport aux deux autres phases. La taille de la liste $\mathcal{L}$ détermine en général laquelle des deux étapes parmi la phase d'analyse ou la phase de recherche exhaustive est la plus coûteuse.

Dans les chapitres 2 et 3 nous décrivons un grand nombre d'attaques statistiques connues des systèmes de chiffrement par bloc. Une étude détaillée des complexités sera faite pour certaines cryptanalyses statistiques dans les chapitres suivants. Avec Benoît Gérard et Jean-Pierre Tillich, nous nous sommes intéressés à la complexité d'un certain nombre d'attaques statistiques. Ce travail détaillé dans le chapitre 5 nous a permis de trouver une formule générale de la complexité en données et de la probabilité de succès d'une certaine classe d'attaques statistiques.
Dans le chapitre 6 nous présentons une autre attaque statistique appelée *cryptanalyse différentielle multiple*. Avec Benoît Gérard nous avons étudié le cas particulier de cette attaque, et nous avons notamment extrait une formule pour la complexité en temps, en données, en mémoire ainsi qu'une formule pour calculer la probabilité de succès de l'attaque.

1.5.5 Les variables aléatoires étudiées

Pour chacune des attaques statistiques que nous présentons dans cette thèse (voir chapitre 2 et chapitre 3), nous avons besoin d'étudier la distribution d'un certain nombre de variables aléatoires afin d'obtenir un distingueur optimal pour chaque type d'attaque mais aussi dans le but d'étudier la complexité en données et la probabilité de succès [10]. Dans cette section nous faisons une description préliminaire générale des variables aléatoires que nous utilisons par la suite.

Supposons que l'attaquant ait en sa possession N échantillons composés d'un certain nombre de couples clair/chiffré. Suivant le type d'attaque statistique, la propriété $\mathcal{P}$ est observée pour un échantillon composé de un ou plusieurs couples clair/chiffré. Pour l'échantillon numéro i, dans la section précédente nous avons défini la variable aléatoire $\mathbf{X}_i$ qui prend les valeurs 0 ou 1 selon que la propriété est observée pour l'échantillon en question

10. Le calcul de la complexité en données et de la probabilité de succès de ces attaques est fait dans le chapitre 5.

ou pas. Dans la réalité, pour des attaques sur le dernier tour [11] pour chaque échantillon on inverse le dernier tour avec toutes les clés candidates possibles et pour chaque échantillon on regarde si la propriété $\mathcal{P}$ est vérifiée. Les variables aléatoires simples que nous étudions dépendent alors des clés candidates. Nous notons par $C_{i,k}$ les variables aléatoires simples que nous étudions. Elles sont définies par

$$C_{i,k} = \begin{cases} 1 & \text{si la propriété } \mathcal{P} \text{ est observée pour l'échantillon} \\ & \text{numéro } i \text{ et pour la clé candidate } k\,; \\ 0 & \text{sinon.} \end{cases} \tag{1.3}$$

Dans les cryptanalyses statistiques sur les derniers (ou premiers) tours on veut distinguer le candidat correspondant à la clé maître des autres. La manière la plus classique consiste alors, pour une clé candidate fixée, à sommer les variables aléatoires simples. Les variables aléatoires que l'on obtient sont notées C_k et sont définies par

$$C_k \stackrel{\text{def}}{=} \sum_{i=1}^{N} C_{i,k}. \tag{1.4}$$

Dans la plupart des attaques statistiques c'est la distribution de ces variables aléatoires qui est étudiée afin de déterminer la complexité en données et la probabilité de succès de l'attaque. Le chapitre 5 est dédié à l'étude de ces quantités dans le cas particulier où les variables C_k suivent des lois binomiales.
L'étude de la distribution de ces variables aléatoires dans le cas d'une généralisation de la cryptanalyse différentielle que nous avons appelée "cryptanalyse différentielle multiple" (dans ce cas les variables aléatoires C_k ne suivent pas une loi binomiale) conduit à une formule de la complexité en données et de la probabilité de succès (voir chapitre 6) pour cette attaque.

11. Cela marche de la même façon pour des attaques sur le premier tour

Chapitre 2

La cryptanalyse différentielle et ses généralisations

Depuis l'avènement de la cryptanalyse différentielle au début des années 90 beaucoup de variantes de cette cryptanalyse ont été introduites. La cryptanalyse différentielle, la cryptanalyse différentielle tronquée, la cryptanalyse différentielle impossible et la cryptanalyse différentielle d'ordre supérieur font partie de ces généralisations. Dans ce chapitre nous présentons ces attaques. Nous détaillons les faiblesses des algorithmes de chiffrement par bloc contre ces différents types d'attaques ainsi que le lien entre ces attaques.

2.1 La cryptanalyse différentielle

2.1.1 Définition d'une attaque différentielle

La cryptanalyse différentielle est une des premières attaques statistiques. Elle a été introduite en 1990 par Eli Biham et Adi Shamir dans le but de casser le DES [BS90, BS91]. Cette attaque statistique sur les systèmes de chiffrement par bloc exploite la mauvaise propagation des différences à l'intérieur du système de chiffrement.

Définition 2.1. *Soit $F_K^r : \mathbb{F}_2^m \to \mathbb{F}_2^m$ r tours d'un système de chiffrement itératif paramétré par une clé K. Une* **différentielle** *sur r tours de ce système de chiffrement est un couple $(a_0, a_r) \in \mathbb{F}_2^m \times \mathbb{F}_2^m$ de différence en entrée et de différence en sortie après r tours.*

Définition 2.2. *Soit $F_K^r : \mathbb{F}_2^m \to \mathbb{F}_2^m$ r tours d'un système de chiffrement itératif paramétré par une clé K. Soit (a_0, a_r) une différentielle sur r tours du système de chiffrement. On définit la* **probabilité d'une différentielle** (a_0, a_r) *par :*

$$P\left[a_0 \to a_r\right] \stackrel{\text{def}}{=} P_{\mathbf{X},\mathbf{K}}\left[F_K^r(X) \oplus F_K^r(X \oplus a_0) = a_r\right],$$

où $P_{\mathbf{X},\mathbf{K}}[\cdot]$ signifie que la probabilité est calculée en moyenne sur tous les messages en entrée et sur toutes les clés possibles.

Il existe plusieurs façons de passer de la différence en entrée à la différence en sortie. On appelle **chemin différentiel** la suite des différences intermédiaires.

Définition 2.3. *Un* **chemin différentiel** *sur r tours d'un système de chiffrement itératif avec fonction de tour $F_K : \mathbb{F}_2^m \to \mathbb{F}_2^m$ est un $(r+1)$-uplet $(\beta_0, \beta_1, \cdots, \beta_{r-1}, \beta_r) \in (\mathbb{F}_2^m)^{(r+1)}$ de différences intermédiaires à chaque tour.*

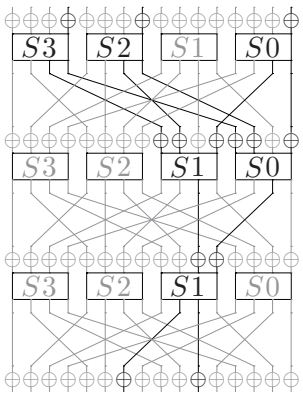

FIGURE 2.1 – Exemple de chemin différentiel sur 3 tours de SMALLPRESENT-[4]

La probabilité d'un chemin différentiel est définie de la façon suivante :

Définition 2.4. *En utilisant les notations de la définition 2.3 on définit la* **probabilité d'un chemin différentiel** $\beta = (\beta_0, \beta_1, \cdots, \beta_{r-1}, \beta_r) \in (\mathbb{F}_2^m)^{(r+1)}$ *par :*

$$P[\beta] \stackrel{\text{def}}{=} P_{\mathbf{X},\mathbf{K}}\left[F_K^i(X) \oplus F_K^i(X \oplus \beta_0) = \beta_i \ \forall i\right].$$

Exemple 2.1. *La figure 2.1 nous donne l'exemple d'un chemin différentiel sur 3 tours de SMALLPRESENT-[4]. Comme la boîte-S est de taille 4, il est pratique d'utiliser la notation hexadécimale pour décrire un message ou une différence. Un nombre en hexadécimal est représenté par la notation* `0x`*. La figure 2.1 représente le chemin différentiel* $(\texttt{0x1101}, \texttt{0x00dd}, \texttt{0x0030}, \texttt{0x0220})$*, c'est-à-dire qu'en entrée du premier tour, les boîtes-S* $S0, S2, S3$ *sont actives*[1] *avec différence en entrée* `0x1`*. Nous supposons ici que la différence en sortie après passage dans les boîtes-S est* `0x3`*. Au second tour, les boîtes actives sont* $S0$ *et* $S1$ *avec différence en entrée* `0xd`*, et ainsi de suite.*

2.1.2 Les primitives utilisées pour résister aux attaques différentielles

Dans un système de chiffrement par bloc itératif, c'est la partie de confusion de la fonction de tour qui joue le plus grand rôle dans la résistance du système de chiffrement contre les attaques différentielles. Cette partie de substitution est composée de boîtes-S qui sont appliquées en parallèle au message divisé en petits blocs.

Dans le contexte de la cryptanalyse différentielle on dit qu'une boîte-S est active si les deux messages en entrée de la boîte-S étudiée possèdent une différence non-nulle.

Dans les attaques différentielles classiques la partie de diffusion joue un rôle dans la diffusion des différences. Cette partie a pour but de toujours garder un nombre suffisant de boîtes-S actives sur plusieurs tours. En revanche cette partie de diffusion ne joue aucun rôle dans le calcul des probabilités du chemin différentiel. C'est-à-dire que pour une différence donnée celle-ci passe cette partie de diffusion avec probabilité 1.

La probabilité d'une différentielle sur un tour du système de chiffrement est donc déterminée par les propriétés des boîtes-S. Afin de prémunir le système de chiffrement

1. On dit qu'une boîte-S est active s'il y a une différence non-nulle en entrée de la boîte-S.

contre les attaques différentielles on demande aux boîtes-S d'être *différentiellement λ-uniforme* avec λ petit.

Définition 2.5. *Soit $f : \mathbb{F}_2^s \to \mathbb{F}_2^t$ une fonction. On note par $\delta(a, b)$ le nombre de x tels que $f(x) + f(x + a) = b$. C'est-à-dire*

$$\delta(a, b) \stackrel{\text{def}}{=} \#\{x | f(x) + f(x + a) = b\}$$

pour (a, b) un couple de différences dans $\mathbb{F}_2^s \backslash \{0\} \times \mathbb{F}_2^t$. Soit λ la valeur maximale des $\delta(a, b)$:

$$\lambda = \max_{a \neq 0,\, b} \delta(a, b).$$

On dit alors que f est **différentiellement λ-uniforme**.

Il est facile de voir que $\delta(a, b)$ est toujours pair et donc que la valeur minimale de λ est 2. Les fonctions qui sont différentiellement 2-uniformes sont dites *"almost perfect non-linear"*(APN). Le but de cette partie n'est pas de rentrer en détail dans l'étude des propriétés de ces boîtes-S. Une étude plus détaillée sera faite dans la seconde partie de ce manuscrit (voir partie II). Ce qu'il faut retenir pour le moment c'est que pour une boîte-S donnée on s'intéresse aux valeurs $\delta(a, b)$ afin de déterminer la probabilité d'une différentielle d'un système de chiffrement. On appelle alors *table des différences* le tableau à deux dimensions qui nous donne les valeurs $\delta(a, b)$ pour toutes les différences en entrée et en sortie de la boîte-S. On obtient alors par exemple le tableau 2.1 pour la boîte-S du système de chiffrement PRESENT (section 1.4.1). On peut y lire par exemple que $\delta(\texttt{0x1}, \texttt{0x3}) = 4$.

	0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7	0x8	0x9	0xa	0xb	0xc	0xd	0xe	0xf
0x0	16	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
0x1	-	-	-	4	-	-	-	4	-	4	-	-	-	4	-	-
0x2	-	-	-	2	-	4	2	-	-	-	2	-	2	2	2	-
0x3	-	2	-	2	2	-	4	2	-	-	2	2	-	-	-	-
0x4	-	-	-	-	-	4	2	2	-	2	2	-	2	-	2	-
0x5	-	2	-	-	2	-	-	-	-	2	2	2	4	2	-	-
0x6	-	-	2	-	-	-	2	-	2	-	-	4	2	-	-	4
0x7	-	4	2	-	-	-	2	-	2	-	-	-	2	-	-	4
0x8	-	-	-	2	-	-	-	2	-	2	-	4	-	2	-	4
0x9	-	-	2	-	4	-	2	-	2	-	-	-	2	-	4	-
0xa	-	-	2	2	-	4	-	-	2	-	2	-	-	2	2	-
0xb	-	2	-	-	2	-	-	-	4	2	2	2	-	2	-	-
0xc	-	-	2	-	-	4	-	2	2	2	2	-	-	-	2	-
0xd	-	2	4	2	2	-	-	2	-	-	2	2	-	-	-	-
0xe	-	-	2	2	-	-	2	2	2	2	-	-	2	2	-	-
0xf	-	4	-	-	4	-	-	-	-	-	-	-	-	-	4	4

TABLE 2.1 – Table des différences de la boîte-S de PRESENT.

La probabilité d'une différentielle (a, b) pour une boîte-S est alors donnée par la valeur $\dfrac{\delta(a, b)}{2^s}$ si la boîte-S est une permutation de $\mathbb{F}_2^s$.

2.1.3 Calcul théorique des probabilités d'une différentielle

Dans la pratique il n'est pas facile de calculer la probabilité exacte d'une différentielle. Dans cette thèse nous faisons la distinction entre la valeur calculée de la probabilité d'une différentielle et la vraie valeur de cette probabilité en moyenne sur les clés et sur les messages. Cette section est dédiée à la présentation des hypothèses communément admises pour calculer la probabilité d'un chemin différentiel puis d'une différentielle.

Pour cela nous définissons les notations suivantes permettant de distinguer les probabilités calculées théoriquement, les probabilités expérimentales et les probabilités exactes ou réelles d'un chemin différentiel et d'une différentielle.

Notation 2.1. *Soit un chemin différentiel $\beta = (\beta_0, \cdots, \beta_r)$ la probabilité théorique de ce chemin est notée $P^t[\beta]$ et la probabilité réelle est notée $P^r[\beta]$. La probabilité dite réelle correspond à la probabilité obtenue en moyenne sur les messages et les clés.*

Sous certaines hypothèses que nous détaillons ci dessous on peut estimer la probabilité d'un chemin différentiel. Pour cela nous avons besoin de définir la notion de chiffrement de Markov qui a été introduite par Xuejia Lai et James Massey dans [LM91].

Définition 2.6. *[LM91] Un système de chiffrement itératif avec fonction de tour F est de* **Markov** *relativement à la cryptanalyse différentielle si la probabilité de la différence en sortie connaissant la différence en entrée est indépendante de la clé utilisée pour chiffrer.*

Proposition 2.1. *[LM91] Supposons que le système de chiffrement E est de Markov. Supposons aussi que les clés de tours sont indépendantes et uniformément distribuées. Alors la séquence des différences $\beta_0, \cdots, \beta_r$ forme une chaîne de Markov. Dans ce cas particulier la probabilité du chemin différentiel $\beta = (\beta_0, \cdots, \beta_r)$ se calcule de la façon suivante :*

$$P^t[\beta] = \prod_{i=1}^{r} P_{\mathbf{X},\mathbf{K}}\left[F(X) \oplus F(X \oplus \beta_{i-1}) = \beta_i,\ \forall i\right].$$

Hypothèse 2.1. *Pour récapituler, les hypothèses communément utilisées pour calculer la probabilité théorique d'un chemin différentiel sont*

- *Le système de chiffrement est de Markov pour la cryptanalyse différentielle.*
- *Les clés de tour sont indépendantes.*
- *Les clés de tour sont uniformément distribuées dans l'espace des clés.*

Ces hypothèses ne sont pas toujours vérifiées ; ainsi, la probabilité théorique d'un chemin différentiel est souvent différente de sa probabilité réelle en moyenne sur les messages et sur les clés. Dans le chapitre 4 nous détaillons la validité de ces hypothèses.

Connaissant tous les chemins qui composent une différentielle ainsi que leur probabilité, il est assez facile de passer de la probabilité des chemins à la probabilité d'une différentielle, puisque cette probabilité est égale à la somme sur les messages des probabilités des chemins qui la composent.

Proposition 2.2. *La probabilité d'une différentielle est égale à la somme des probabilités des chemins qui la composent.*

$$P[a_0 \to a_r] = \sum_{\beta=(a_0,\beta_1,\cdots,\beta_{r-1},a_r)} P[\beta].$$

Preuve : À clé fixée, ce résultat provient du fait que si les messages X et $X \oplus a_0$ suivent un chemin différentiel, ils ne peuvent pas suivre un autre chemin différentiel. Ainsi les événements étudiés sont disjoints. □

Exemple 2.2. *Nous reprenons les notations de l'exemple 2.1. Dans cet exemple nous étudions la différentielle* $(\texttt{0x1101}, \texttt{0x00dd})$ *sur 3 tours de SMALLPRESENT-[4].*

Un des chemins composant la différentielle est donné dans l'exemple 2.1. Le calcul de la probabilité théorique de ce chemin sous les hypothèses décrites précédemment se fait de la façon suivante :

- *Au tour 1 nous avons 3 boîtes-S avec différence en entrée* `0x1` *et différence en sortie* `0x3`.
- *Au tour 2 nous avons 2 boîtes-S avec différence en entrée* `0xd` *et différence en sortie* `0x2`.
- *Au tour 3 nous avons 1 boîte-S avec différence en entrée* `0x3` *et différence en sortie* `0x6`.

D'après la table des différences de PRESENT (tableau 2.1) nous avons que la probabilité pour une boîte-S de passer d'une différence en entrée `0x1` *à une différence en sortie* `0x3` *est* $\frac{4}{2^4} = 2^{-2}$*. Au total nous avons 6 boîtes-S actives (différence en entrée non nulle) avec probabilité de transition* $\frac{4}{2^4} = 2^{-2}$ *(voir tableau 2.1) donc la probabilité théorique de ce chemin différentiel est*

$$P^t\left[(\texttt{0x1101}, \texttt{0x00dd}, \texttt{0x0030}, \texttt{0x220})\right] = (2^{-2})^6 = 2^{-12}.$$

Pour calculer la probabilité de la différentielle, nous avons besoin de calculer la probabilité théorique de tous les chemins la composant (en utilisant la même méthode que celle expliquée ci-dessus). La probabilité de chacun de ces chemins est résumée dans le tableau 2.2.

β_0	β_1	β_2	β_3	$P^t\left[(\beta_0, \beta_1, \beta_2, \beta_3)\right]$
`0x1101`	`0x00dd`	`0x0030`	`0x220`	2^{-12}
`0x1101`	`0x0cdd`	`0x0070`	`0x220`	2^{-16}
`0x1101`	`0x09dd`	`0x0070`	`0x220`	2^{-16}
`0x1101`	`0x0ddd`	`0x0070`	`0x220`	2^{-15}
`0x1101`	`0xd00d`	`0x0090`	`0x220`	2^{-13}

TABLE 2.2 – Chemins différentiels composant la différentielle $(\texttt{0x1101}, \texttt{0x00dd})$ pour le système de chiffrement SMALLPRESENT-[4]

Compte tenu des probabilités théoriques des chemins composant la différentielle

$$(\texttt{0x1101}, \texttt{0x00dd})$$

la probabilité théorique de cette différentielle est

$$2^{-12} + 2^{-13} + 2^{-15} + 2 \cdot 2^{-16} = 2^{-11{,}1926}.$$

Une étude approfondie du calcul de ces probabilités est détaillée dans le chapitre 4. On verra qu'il est assez compliqué voir impossible en général de trouver tous les chemins

qui composent une différentielle. En déterminant la probabilité de certains chemins nous avons une borne inférieure sur la probabilité de la différentielle. Dans le chapitre 4 nous décrivons aussi un algorithme pour trouver les chemins qui composent une différentielle ainsi que leur probabilité.

2.1.4 Comment retrouver de l'information sur la clé

Une attaque différentielle se comporte différemment suivant le type de système de chiffrement par bloc. En effet dans un chiffrement de type substitution-permutation c'est une cryptanalyse de type attaque sur le dernier tour, alors que dans un système de chiffrement de type Feistel, l'attaque est sensiblement différente et les quantités que l'on regarde aussi. Comme la première attaque différentielle a été faite sur le DES qui est un chiffrement de Feistel de 16 tours, nous allons dans un premier temps détailler les attaques différentielles sur ce type de système de chiffrement.

Attaque différentielle sur les systèmes de chiffrement de type Feistel

Dans cette partie nous expliquons le principe de l'attaque différentielle pour les schémas de Feistel classiques (ceux définis dans la section 1.2.2). Le principe de l'attaque pour les généralisations du schéma de Feistel reste sensiblement le même.

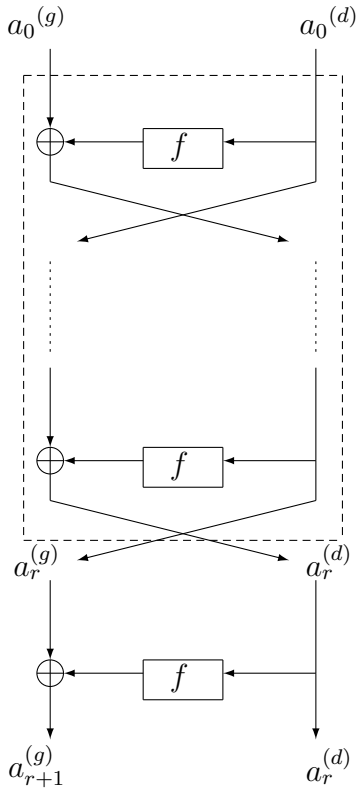

FIGURE 2.2 – Attaque différentielle d'un schéma de Feistel.

Supposons que nous avons une différentielle sur r tours

$$\Big((a_0{}^{(g)}, a_0{}^{(d)}), (a_r^{(g)}, a_r^{(d)})\Big)$$

(voir figure 2.2). Dans une attaque différentielle sur un schéma de Feistel, on récupère les messages chiffrés sur $r+1$ tours. Dans ce type de système de chiffrement on connaît la différence $a_r^{(d)}$ en entrée de la fonction interne f du tour r ainsi que la valeur du message avant ajout de la clé. Si la caractéristique différentielle est bien celle espérée alors on connaît aussi la différence à la sortie de la fonction f. Cette différence est $a_{r+1}^{(g)} \oplus a_r^{(g)}$. Le détail des valeurs connues par l'attaquant est donné dans la figure 2.3.

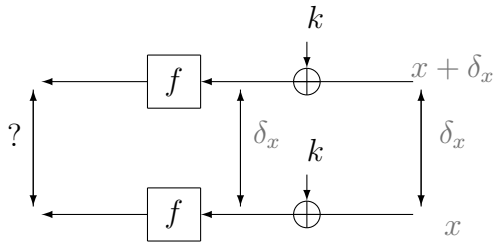

FIGURE 2.3 – Détails de l'étape de recherche de clé pour une différentielle sur un schéma de Feistel. (Les valeurs connues sont en gris)

Dans le cas d'un schéma de Feistel on peut utiliser un crible au tour $r+1$ pour supprimer des mauvaises paires. Ce crible est défini pour une différentielle

$$\Big((a_0{}^{(g)}, a_0{}^{(d)}), (a_r^{(g)}, a_r^{(d)})\Big)$$

de la façon suivante :

$$\Delta_{sieve} = \left\{ (a^{(g)} || a_r^{(d)}) \;\middle|\; P\left[a_r^{(d)} \underset{f}{\rightarrow} a^{(g)} \oplus a_r^{(g)} \right] \neq 0 \right\}. \tag{2.1}$$

C'est-à-dire Δ_{sieve} correspond à toutes les différences possibles après un tour de chiffrement quand la différence en entrée de ce tour est $(a_r^{(g)}, a_r^{(d)})$. Il est d'usage dans les systèmes de type Feistel d'enlever la permutation finale.

Le principe de l'attaque qui consiste à retrouver la clé du dernier tour d'un schéma de Feistel classique est résumé dans l'algorithme 1.

Attaque différentielle sur les systèmes de chiffrement de type substitution-permutation

Les attaques différentielles sur les schémas de type substitution-permutation sont des attaques sur le dernier tour (ou les derniers tours). Elles prennent en compte une caractéristique différentielle sur r tours pour faire une attaque sur $r+1$ tours et ainsi retrouver une partie de la clé du $(r+1)$ème tour.

À la différence des attaques sur les schémas de Feistel on ne connaît pas la valeur du message avant l'entrée dans les boîtes-S mais seulement la valeur du message en sortie.

Dans le cas d'un système de type substitution-permutation un crible est aussi utilisé pour supprimer un certain nombre de mauvaises paires. Ce crible est défini pour une différentielle (a_0, a_r) de la façon suivante :

Algorithme 1 : Cryptanalyse différentielle d'un système de type Feistel (attaque sur le dernier tour).

Entrée : N couples $(X, X' = X \oplus a_0)$ et les chiffrés correspondants $(Y = E_{K^*}(X), Y' = E_{K^*}(X'))$
Sortie : La clé maître K^* utilisée pour chiffrer les échantillons
Initialiser une table C de 2^n compteurs à 0 ;
Pour chaque *couple* (X, X') *tel que* $X \oplus X' = a_0$ **faire**
  Si $E_{K^*}(X) \oplus E_{K^*}(X \oplus a_0) \in \Delta_{sieve}$ **alors**
    Pour chaque *sous clé candidate* k **faire**
      Calculer $d = f(Y^{(r)} \oplus k) \oplus f(Y'^{(r)} \oplus k)$;
      Si $d \oplus a_{r+1}^{(l)} = a_r^{(l)}$ **alors** $C[k] \leftarrow C[k] + 1$;
Générer une liste $\mathcal{L}$ de ℓ candidats ayant les plus grandes valeurs de $C[k]$;
Pour chaque $k \in \mathcal{L}$ **faire**
  Pour chaque *clé maître* K correspondant à la clé k **faire**
    Si $E_K(X) = Y = E_{K^*}(X)$ **alors retourner** K;

$$\Delta_{sieve} = \left\{ a \mid P\left[a_r \underset{F}{\rightarrow} a \right] \neq 0 \right\}. \tag{2.2}$$

L'algorithme 2 résume cette attaque.

Algorithme 2 : Cryptanalyse différentielle d'un système de type substitution-permutation (attaque sur le dernier tour).

Entrée : N couples $(X, X' = X \oplus a_0)$ et les chiffrés correspondants $(Y = E_{K^*}(X), Y' = E_{K^*}(X'))$
Sortie : La clé maître K^* utilisée pour chiffrer les échantillons
Initialiser une table C de 2^n compteurs à 0 ;
Pour chaque *couple* $(X, X \oplus a_0)$ **faire**
  Si $E_{K^*}(X) \oplus E_{K^*}(X \oplus a_0) \in \Delta_{sieve}$ **alors**
    Pour chaque *sous clé candidate* k **faire**
      Calculer $d = F_k^{-1}(E_{K^*}(X)) \oplus F_k^{-1}(E_{K^*}(X'))$;
      Si $d = a_r$ **alors** $C[k] \leftarrow C[k] + 1$;
Générer une liste $\mathcal{L}$ de ℓ candidats ayant les plus grandes valeurs de $C[k]$;
Pour chaque $k \in \mathcal{L}$ **faire**
  Pour chaque clé maître K correspondant à la clé k **faire**
    Si $E_K(X) = Y = E_{K^*}(X)$ **alors retourner** K;

2.1.5 Quantités importantes dans la cryptanalyse différentielle

Dans la cryptanalyse différentielle, un échantillon est composé de deux messages ayant une différence fixée ainsi que des chiffrés correspondants. Nous rappelons les notations communément utilisées dans cette thèse : la *complexité en données* de l'attaque est notée

par N_{DC} alors que le *nombre d'échantillons* est noté par N. Ainsi dans le cas de la cryptanalyse différentielle nous avons $N_{DC} = 2N$.

Le rapport signal sur bruit Pour évaluer le nombre de couples clair/chiffré N_{DC} dont nous avons besoin pour une attaque différentielle, une quantité importante a été introduite par Eli Biham et Adi Shamir [BS90]. Cette quantité s'appelle le rapport signal sur bruit.

Définition 2.7. *Le rapport entre le nombre de bonnes paires et la moyenne du nombre de clés cochées est appelé le* **rapport signal sur bruit**. *Cette quantité est notée S/N.*

Cette quantité peut être calculée assez facilement si on introduit les définitions suivantes. On suppose que l'on veut retrouver n bits de clés. Soit α le nombre moyen de cases cochées pour les paires gardées (c'est-à-dire les paires qui passent le crible) et soit β, le nombre moyen de cases cochées pour toutes les paires. Soit p_* la probabilité de la différentielle lorsque l'on déchiffre avec la bonne clé. Cette probabilité est différente suivant le type de système de chiffrement. Avec ces notations la bonne clé est comptée p_*N fois. Le rapport signal sur bruit devient alors

$$S/N = \frac{N \cdot p_*}{N \cdot \alpha \cdot \beta / 2^n} = \frac{p_* \cdot 2^n}{\alpha \cdot \beta}.$$

L'étude basique du calcul de la complexité en données faite par Eli Biham et Adi Shamir est directement reliée à la valeur du rapport signal sur bruit. Dans la plupart des attaques existantes on suppose que si le rapport signal sur bruit est suffisamment grand (de l'ordre de 13) alors la complexité en données est de $\frac{4}{p_*}$. Le détail de ce calcul est donné par exemple dans la thèse de Henri Gilbert [Gil97] [2]. Durant cette thèse, avec Benoît Gérard [BG10] nous avons mené une étude complète de la complexité en données de certaines attaques statistiques et en particulier celle de la cryptanalyse différentielle. Un autre travail avec Jean-Pierre Tillich [BGT11], nous a permis de trouver une formule générale pour la probabilité de succès d'une attaque statistique. Le détail de cette étude est donné dans le chapitre 5.

2.1.6 Les probabilités utilisées

L'étude classique pour calculer la complexité en données d'une attaque différentielle repose sur le calcul du rapport signal sur bruit [BS90, BS91].

Nous allons dans le chapitre 5 utiliser une autre méthode pour calculer la complexité en données et la probabilité de succès d'une cryptanalyse différentielle. Pour cela nous avons besoin d'étudier les valeurs des probabilités des compteurs définis dans les algorithmes 1 et 2 pour la bonne et les mauvaises sous-clés.

Distribution des variables aléatoires

Définition 2.8. *Soit E_{K^*} un système de chiffrement par bloc itératif avec fonction de tour F. Soit (a_0, a_r) la différentielle étudiée sur r tours du système de chiffrement. Supposons que l'on cherche à retrouver de l'information sur la clé du tour $r+1$. Les variables*

2. Cette étude repose sur une approximation de la distribution des variables aléatoires par une loi de Poisson.

aléatoires utilisées dans le cadre de la cryptanalyse différentielle pour un message fixé et une clé fixée sont définis par

$$C_{X,k} \stackrel{\text{def}}{=} \begin{cases} 1 & si \quad F_k^{-1}(E_{K^*}(X)) \oplus F_k^{-1}(E_{K^*}(X \oplus a_0)) = a_r, \\ 0 & sinon. \end{cases}$$

Ces variables aléatoires suivent des distributions différentes suivant la valeur de la clé.

L'hypothèse communément faite dans la cryptanalyse différentielle est que les variables aléatoires correspondant aux mauvaises clés suivent toutes la même distribution. Cette hypothèse est appelée *hypothèse de répartition aléatoire par mauvaise clé*

Hypothèse 2.2 (Hypothèse de répartition aléatoire par fausse clé)**.** *Soit $E_{K^*} : \mathbb{F}_2^m \to \mathbb{F}_2^m$ un système de chiffrement par bloc paramétré par la clé K^* avec fonction de tour F.*

$$P_{\mathbf{X}}\left[F_k^{-1}(E_{K^*}(X)) \oplus F_k^{-1}(E_{K^*}(X \oplus a_0)) = a_r\right] = \begin{cases} p_* & \text{si } k = k^*, \\ p = \frac{1}{2^m - 1} & \text{pour } k \neq k^*. \end{cases}$$

Sous cette hypothèse on obtient que pour un message fixé les variables aléatoires $C_{X,k}$ suivent une loi de Bernoulli avec probabilité p_* dans le cas où la clé candidate correspond à la bonne sous clé. Dans le cas contraire ces variables suivent une loi de Bernoulli avec probabilité $p = \dfrac{1}{2^m - 1}$.

Les compteurs étudiés dans le cas de la cryptanalyse différentielle correspondent à la somme des $C_{X,k}$:

$$C_k = \sum_X C_{X,k}.$$

Théoriquement le compteur C_k est incrémenté avec probabilité p_* si $k = k^*$ et avec probabilité p dans le cas contraire.

La probabilité p_* correspondant au compteur relié à la bonne clé dépend quant à elle de la valeur de la probabilité de la différentielle étudiée ($P[a_0 \to a_r]$) mais est différente suivant le type de système de chiffrement étudié. Par la suite nous détaillons la valeur de cette probabilité dans le cas d'un système de chiffrement de type Feistel et d'un système de chiffrement de type substitution-permutation.

Détail des probabilités dans le cas d'un chiffrement de Feistel Afin de comprendre le principe dans le cas d'un schéma de Feistel nous avons besoin de revenir sur la description de l'attaque sur le dernier tour donnée dans la section 2.1.4. Pour une paire de messages qui passe le crible l'attaquant fait une hypothèse sur la clé du dernier tour utilisée pour chiffrer. Ainsi il connaît donc la valeur des messages en entrée de la fonction interne et en sortie de cette fonction interne. Soit $a = f(x \oplus k) \oplus f(x \oplus a_r^{(d)} \oplus k)$ la différence en sortie de la fonction interne.

Le compteur est incrémenté si $a \oplus a_r^{(g)} = a_{r+1}^{(g)}$. Deux cas se présentent alors ; en effet la valeur $a_r^{(g)}$ n'est pas connue par l'attaquant mais arrive avec une probabilité supérieure à une autre valeur $a_r^{(g)'}$. Donc en appliquant la fonction interne on peut obtenir ${a_{r+1}^{(g)}}'$ à partir de $a \oplus a_{r+1}^{(g)}$ mais aussi avec d'autres valeurs $a' \oplus {a_{r+1}^{(g)}}'$ Ainsi le compteur correspondant à la bonne clé est incrémenté avec la probabilité suivante :

$$p_* = P\left[a_r^{(d)} \underset{F}{\to} a \oplus a_r^{(g)} = a_{r+1}^{(g)}\right] + P\left[a_r^{(d)} \underset{F}{\to} a' \oplus a_r'^{(g)} = a_{r+1}^{(g)}\right] \tag{2.3}$$

Proposition 2.3. *Soit (a_0, a_r) la différentielle étudiée. On se place dans le contexte d'une attaque sur le dernier tour d'un schéma de Feistel classique comme celle présentée dans l'algorithme 1. Soit p la probabilité qu'un compteur correspondant à une mauvaise clé soit incrémenté*

$$p = P\left[C_{X,k} = 1 | k \neq k^*\right] \approx 2^{-m}.$$

La probabilité que le compteur correspondant à la bonne clé soit incrémenté est :

$$p_* = P\left[a_0 \rightarrow a_r\right] + p. \tag{2.4}$$

Détail des probabilités dans le cas d'un chiffrement de type substitution-permutation Dans un système de type substitution-permutation le compteur correspondant à la bonne clé est incrémenté avec une probabilité différente de celle dans le cas d'un système de type Feistel. Dans ce cas précis, pour une paire qui passe le crible, on déchiffre en testant toutes les clés possibles. Pour la bonne sous-clé le compteur est incrémenté si et seulement si on obtient la différence a_r en déchiffrant. Ce phénomène apparaît avec la même probabilité que la probabilité de la différentielle.

Proposition 2.4. *Considérons une attaque sur le dernier tour d'un schéma de type substitution-permutation comme présentée dans l'algorithme 2. Soit p la probabilité qu'un compteur correspondant à une mauvaise clé soit incrémenté. La probabilité que le compteur correspondant à la bonne clé soit incrémenté est égale à la probabilité de la différentielle.*

$$p_* = P\left[a_0 \rightarrow a_r\right]. \tag{2.5}$$

2.2 La cryptanalyse différentielle tronquée

La cryptanalyse différentielle tronquée [Knu95] est une généralisation de la cryptanalyse différentielle. Dans le cas de la cryptanalyse différentielle tronquée, on ne tire pas de l'information à partir d'une seule différence mais d'un ensemble de différences. Soit A_0 un ensemble de différences en entrée et A_r un ensemble de différences en sortie. Dans le cas de la cryptanalyse différentielle tronquée l'attaquant s'intéresse à la probabilité pour un couple de messages en entrée ayant une différence dans A_0 d'obtenir une différence entre les messages chiffrés dans A_r. La cryptanalyse différentielle tronquée a servi à casser de nombreux systèmes de chiffrement par bloc. Dans la section 2.2.5, nous donnons des exemples de systèmes de chiffrement ayant montré des faiblesses contre cette attaque. Ces exemples tendent à montrer les différentes variantes de la cryptanalyse différentielle tronquée et de la difficulté de trouver une formalisation générale pour définir cette cryptanalyse. Une première formalisation a été faite dans [MSAK99]. Nous présentons ici une généralisation de celle ci en utilisant les notations communément utilisées dans cette thèse.

2.2.1 Définition

Définition 2.9. *Soit E_K un système de chiffrement par bloc ($E : \mathbb{F}_2^m \rightarrow \mathbb{F}_2^m$). Une* **différentielle tronquée** *de ce système de chiffrement est un couple formé par un ensemble de différences en entrée $A_0 \subset \mathbb{F}_2^m$ et un ensemble de différences en sortie $A_r \subset \mathbb{F}_2^m$.*

Définition 2.10. *Soit E_K un système de chiffrement par bloc itératif comprenant r tours. Soit (A_0, A_r) une différentielle tronquée de ce système de chiffrement. La probabilité de cette différentielle tronquée est définie par*

$$P\left[A_0 \to A_r\right] \stackrel{\text{def}}{=} P_{\mathbf{X},\mathbf{K}}\left[E_K(X) + E_K(X \oplus a) \in A_r | a \in A_0\right].$$

Comme dans le cas de la cryptanalyse différentielle où l'on définit un chemin différentiel, on peut aussi dans le cas de la différentielle tronquée parler de chemin différentiel tronqué.

Définition 2.11. *Soit A_0 et A_r un ensemble de différences en entrée et en sortie de r tours d'un système de chiffrement par bloc itératif. Un chemin différentiel tronqué relatif à (A_0, A_r) est défini comme un ensemble d'ensembles de différence à chaque tour : $(B_0, B_1, \cdots, B_r)$ tel que $B_0 = A_0$ et $B_r = A_r$* [3].

De la même façon que dans le cas de la cryptanalyse différentielle on peut définir la probabilité d'un chemin différentiel tronqué.

Définition 2.12. *Soit E un système de chiffrement par bloc itératif avec fonction de tour F. Soit $B = (B_0, B_1, \cdots, B_r)$ un chemin différentiel tronqué. La probabilité de ce chemin est*

$$P\left[B = (B_0, B_1, \cdots, B_r)\right] \stackrel{\text{def}}{=} P_{\mathbf{X},\mathbf{K}}\left[F_K^i(X) + F_K^i(X + a) \in B_i \; \forall i = 1 \cdots r | a \in B_0\right].$$

où F_K^i est la notation communément utilisée pour $F_{K_i} \circ \cdots \circ F_{K_1}$ où $K_1 \cdots K_r$ sont les clés de tours correspondant à la clé maître K.

2.2.2 L'attaque

Dans cette partie nous décrivons l'attaque différentielle tronquée d'un système de chiffrement par bloc. Un certain nombre d'attaques différentielles tronquées ont été faites sur des systèmes de chiffrement de type Feistel (comme par exemple l'attaque sur E2 [MSAK99]). Soit un système de chiffrement avec une différentielle tronquée (A_0, A_r). Le crible pour les schémas de Feistel classiques qui peut être appliqué afin de diminuer la complexité en temps de l'attaque est défini de la façon suivante :

$$\Delta_{sieve} = \left\{ (a^{(g)} || a^{(d)}) | a^{(d)} \in a_r^{(d)} \text{ et } P\left[a_r^{(d)} \underset{f}{\to} \gamma \in a^{(g)} \oplus A_r^{(g)}\right] \neq 0 \right\}$$

L'algorithme 3 décrit le principe de l'attaque différentielle tronquée pour les schémas de Feistel classiques. Il peux être facilement adapté en fonction du système de chiffrement.

Pour les systèmes de chiffrement de type SPN l'algorithme est similaire à l'algorithme utilisé pour la cryptanalyse différentielle.

2.2.3 Les variables aléatoires utilisées dans la cryptanalyse différentielle tronquée

Définition 2.13. *Soit E_{K^*} un système de chiffrement par bloc itératif avec fonction de tour F. Soit (A_0, A_r) la différentielle tronquée étudiée sur r tours du système de chiffrement. Supposons que l'on cherche à retrouver de l'information sur la clé du tour $r+1$. Les*

3. où $B_0 = A_0$ signifie que les ensembles étudiés sont égaux élément par élément.

Algorithme 3 : Cryptanalyse différentielle tronquée d'un schéma de Feistel

Entrée : N_{DC} couples de messages clairs-chiffrés (X, Y) avec $Y = E_{K^*}(X)$
Sortie : La clé K^* utilisée pour chiffrer les messages
Initialiser une table C de 2^n compteurs à 0.
Pour chaque $a_0 \in A_0$ **faire**
  Pour chaque *couple* (X, X') *tel que* $X \oplus X' = a_0$ **faire**
    Si $E_{K^*}(X) \oplus E_{K^*}(X \oplus a_0) \in \Delta_{sieve}$ **alors**
      Pour chaque *sous clé candidate* k **faire**
        Calculer $d = f(Y^{(d)} \oplus k) \oplus f(Y'^{(d)} \oplus k)$;
        Si $d \oplus a_{r+1}^{(g)} \in A_r^{(g)}$ **alors** $C[k] \leftarrow C[k] + 1$;
  Générer une liste $\mathcal{L}$ de ℓ candidats ayant les plus grandes valeurs de $C[k]$;
  Pour chaque $k \in \mathcal{L}$ **faire**
    Pour chaque *clé maître* K correspondant à la clé k **faire**
      Si $E_K(X) = Y = E_{K^*}(X)$ **alors retourner** K;

variables aléatoires utilisées dans le cadre de la cryptanalyse différentielle tronquée pour un message fixé et une clé fixée sont définies pour $a_0 \in A_0$ *par*

$$C_{X,k}^{(a_0)} \stackrel{\text{def}}{=} \begin{cases} 1 & si \quad F_k^{-1}(E_{K^*}(X)) \oplus F_k^{-1}(E_{K^*}(X \oplus a_0)) \in A_r, \\ 0 & sinon. \end{cases}$$

Puis on définit les variables

$$C_{X,k} \stackrel{\text{def}}{=} \sum_{a_0 \in A_0} C_{X,k}^{(a_0)}.$$

Les variables aléatoires $C_{X,k}^{(a_0)}$ suivent des distributions différentes suivant la valeur de la clé. Comme dans la cryptanalyse différentielle, l'hypothèse communément faite dans la cryptanalyse différentielle tronquée consiste à dire que les variables aléatoires correspondant aux mauvaises clés suivent toutes la même distribution.

Hypothèse 2.3. Hypothèse de répartition aléatoire par fausse clé
Soit $\#A_r$ *le cardinal de* A_r*. Soit* $E_{K^*} : \mathbb{F}_2^m \to \mathbb{F}_2^m$ *un système de chiffrement par bloc avec fonction de tour* F*.*

$$P_{\mathbf{X}}\left[F_k^{-1}(E_{K^*}(X)) \oplus F_k^{-1}(E_{K^*}(X \oplus a_0)) \in A_r\right] = \begin{cases} p_* & \text{si } k = k^*, \\ p = \frac{\#A_r}{2^m - 1} & \text{pour } k \neq k^*. \end{cases}$$

Sous cette hypothèse on obtient que pour un message fixé les variables aléatoires $C_{X,k}$ suivent une loi de Bernoulli avec probabilité p_* dans le cas où la clé candidate correspond à la bonne sous clé. Dans le cas contraire ces variables suivent une loi de Bernoulli avec probabilité $p \approx \#A_r 2^{-m}$.

Les compteurs étudiés dans le cas de la cryptanalyse différentielle tronquée correspondent à la somme des variables aléatoires $C_{X,k}$:

$$C_k \stackrel{\text{def}}{=} \sum_X C_{X,k}.$$

Lemme 2.1. *Soit les variables aléatoires définies dans la définition 2.13. Sous l'hypothèse que les variables aléatoires $C_{X,k}^{(a_0)}$ sont indépendantes les variables aléatoires C_k suivent des lois binomiales de paramètres $(N\#A_0, p_*)$ ou $(N\#A_0, p)$.*

Preuve : Nous faisons la preuve dans le cas où la clé candidate n'est pas la bonne clé. La preuve pour le compteur correspondant à la bonne clé se fait de la même façon. Nous avons que les variables aléatoires $C_{X,k}^{(a_0)}$ suivent une distribution de Bernoulli de paramètre p. Sous l'hypothèse d'indépendance de ces variables aléatoires, les variables aléatoires $C_{X,k}$ suivent une loi binomiale de paramètres $\#A_0$ et p. Les variables aléatoires C_k sont la somme des variables aléatoires définies précédemment. Maintenant sous l'hypothèse d'indépendance de ces variables aléatoires, comme elles ont toutes la même probabilité, nous avons que les variables aléatoires C_k suivent des lois binomiales de paramètre $(N\#A_0, p)$. □

2.2.4 Calcul théorique des probabilités

Lien avec la probabilité d'une différentielle La probabilité d'une différentielle tronquée peut être calculée à partir des différentielles qui la composent. En effet elle correspond à la somme des différentielles qui la composent divisée par le nombre de différences en entrée.

Proposition 2.5. *Sous l'hypothèse que toutes les différences en entrée sont uniformément réparties dans l'espace A_0, c'est-à-dire que $P[a_0|a_0 \in A_0] = P[a_1|a_1 \in A_0]$. On obtient que la probabilité d'une différentielle tronquée est égale à la somme des probabilités des différentielles qui la composent divisée par le cardinal de A_0.*

$$\begin{aligned} P_{\mathbf{X},\mathbf{K}}\left[E(X) \oplus E(X \oplus a_0) \in A_r | a_0 \in A_0\right] &= \sum_{a_r \in A_r} P_{\mathbf{X},\mathbf{K}}\left[E(X) \oplus E(X \oplus a_0) = a_r | a_0 \in A_0\right] \\ &= \frac{1}{\#A_0} \sum_{a_0 \in A_0} P_{\mathbf{X},\mathbf{K}}\left[E(X) \oplus E(X \oplus a_0) \in A_r\right] \\ &= \frac{1}{\#A_0} \sum_{a_0 \in A_0} \sum_{a_r \in A_r} P_{\mathbf{X},\mathbf{K}}\left[E(X) \oplus E(X \oplus a_0) = a_r\right]. \end{aligned}$$

Preuve : Pour des raisons de simplicité nous allons démontrer cette proposition dans le cas où les espaces de différences en entrée et en sortie sont réduits à deux éléments. Pour cela nous notons $A_0 = \{a_0, a_1\}$ et $A_r = \{b_0, b_1\}$. Pour simplifier les écritures nous utilisons la notation non standard suivante :

$$P[A_r|A_0] \stackrel{\text{def}}{=} P_{\mathbf{X},\mathbf{K}}\left[E_K(X) \oplus E_K(X \oplus a_0) \in A_r | a_0 \in A_0\right].$$

La première égalité se montre facilement. En effet, comme pour un couple de message fixé, la différence en sortie est unique les événements $P[b_i|A_0]$ pour $i = 0, 1$ sont disjoints donc :

$$\begin{aligned} P[A_r|A_0] &= P[b_0 \cup b_1|A_0] \\ &= P[b_0|A_0] + P[b_1|A_0]. \end{aligned}$$

La seconde égalité est un peu moins facile à prouver

$$\begin{aligned} P\left[A_r|A_0\right] &= P\left[A_r|a_0 \cup a_1\right] \\ &= \frac{P\left[A_r \cap (a_0 \cup a_1)\right]}{P\left[a_0 \cup a_1\right]} \\ &= \frac{P\left[(A_r \cap a_0) \cup (A_r \cap a_1)\right]}{P\left[a_0\right] + P\left[a_1\right]} \\ &= \frac{P\left[A_r|a_0\right] P\left[a_0\right] + P\left[A_r|a_1\right] P\left[a_1\right]}{P\left[a_0\right] + P\left[a_1\right]} \\ &= \frac{P\left[A_r|a_0\right] + P\left[A_r|a_1\right] P\left[a_1\right]}{P\left[a_0\right] + P\left[a_1\right]} \\ &= \frac{P\left[A_r|a_0\right] P\left[a_0\right] + P\left[A_r|a_1\right] P\left[a_0\right]}{2P\left[a_0\right]} \\ &= \frac{1}{2} \sum_{a=\{a_0,a_1\}} P\left[A_r|a\right]. \end{aligned}$$

La troisième égalité se déduit facilement à partir des deux premières. □

Pour calculer la probabilité théorique d'une différentielle tronquée on n'est pas obligé de passer par la probabilité théorique des différentielles qui la composent. En général on préfère calculer la probabilité d'une différentielle tronquée quand on ne peut pas avoir une bonne estimation de la probabilité des différentielles. En utilisant les propriétés des chaînes de Markov pour la différentielle tronquée nous pouvons avoir une estimation de la probabilité d'une différentielle tronquée.

Chiffrement de Markov

Si le système de chiffrement est de Markov pour la cryptanalyse différentielle (définition 4.2), la probabilité d'un chemin différentiel peut être obtenue en multipliant les probabilités de transition de chaque tour (voir proposition 2.1). Dans ce paragraphe nous allons montrer que même si le système de chiffrement est de Markov pour la cryptanalyse différentielle il n'est pas toujours vrai qu'il soit de Markov pour la différentielle tronquée c'est-à-dire que l'on ne peut pas toujours estimer la probabilité d'un chemin différentiel tronqué en multipliant les probabilités de transition de chaque tour.
Si on reprend la définition d'un chiffrement de Markov donné dans le cas de la cryptanalyse différentielle on obtient la définition suivante :

Définition 2.14. *Un système de chiffrement itératif est de* **Markov relativement à la cryptanalyse différentielle tronquée** *si la probabilité que la différence en sortie soit dans un espace de sortie, connaissant la différence en entrée dans un espace de différence en entrée, est indépendante de la clé utilisée.*

Proposition 2.6. *Soit $E = F^r$ un système de chiffrement par bloc itératif. Supposons que E est un chiffrement de Markov pour la différentielle tronquée. Alors pour tout chemin différentiel tronqué $(B_0, B_1, \cdots, B_r)$ on a*

$$P_{\mathbf{X},\mathbf{K}}\left[F_K^i(X) \oplus F_K^i(X \oplus a) \in B_i | a \in B_0\right] = \prod_{i=1}^{r} P_{\mathbf{X},\mathbf{K}}\left[F_K(X) + F_K(X \oplus a) \in B_i | a \in B_{i-1}\right].$$

Remarque 2.1. *Un système de chiffrement peut être de Markov pour la cryptanalyse différentielle et ne pas l'être pour la cryptanalyse différentielle tronquée. Supposons que nous ayons un chemin différentiel tronqué sur deux tours (B_0, B_1, B_2) d'un système de chiffrement avec fonction de tour F. On note par F_K^2 la composition de deux fois la fonction de tour. Soit les probabilités*

$$\begin{aligned} p_0 &= P_{\mathbf{X},\mathbf{K}}\left[F_K(X) \oplus F_K(X \oplus a) \in B_1 \text{ et } F_K^2(X) \oplus F_K^2(X \oplus a) \in B_2 | a \in B_0\right] \\ p_1 &= P_{\mathbf{X},\mathbf{K}}\left[F_K(X) \oplus F_K(X \oplus b) \in B_2 | b \in B_1\right] \cdot P_{\mathbf{X},\mathbf{K}}\left[F_K(X) \oplus F_K(X \oplus a) \in B1 | a \in B_0\right] \\ p_2 &= \sum_{b \in B_1} P_{\mathbf{X},\mathbf{K}}\left[F_K(X) \oplus F_K(X \oplus b) \in B_2\right] \cdot P_{\mathbf{X},\mathbf{K}}\left[F_K(X) \oplus F_K(X \oplus a) = b | a \in B_0\right]. \end{aligned}$$

Alors on a que la probabilité du chemin différentiel tronqué est égale à p_0. Si le système de chiffrement est de Markov pour la cryptanalyse différentielle tronquée alors $p_1 = p_0$ et s'il est de Markov pour la cryptanalyse différentielle alors $p_2 = p_0$. En général p_1 est différent de p_2. Ce qui est encore plus compliqué est que p_1 n'est ni une majoration ni une minoration de p_2.

2.2.5 Attaques existantes

La plupart des attaques différentielles tronquées sont effectuées sur des systèmes de chiffrement *"orientés mot"*.

Souvent les mots manipulés sont des mots de la taille des boîtes-S composant le système de chiffrement (Par exemple l'AES est un système de chiffrement "orienté mot" contrairement à PRESENT).

Dans ce type de chiffrement par bloc, il est facile de déterminer des chemins différentiels tronqués. Contrairement à la cryptanalyse différentielle ces chemins différentiels dépendent en général de la partie de diffusion de la fonction de tour. Les attaques différentielles tronquées peuvent être très différentes suivant le système de chiffrement étudié.

Ainsi la première attaque différentielle tronquée faite par Lars R. Knudsen pour cryptanalyser le DES [Knu95] prend en compte un chemin différentiel tronqué sur 4 tours du DES avec probabilité 1. La sortie de ce chemin différentiel tronqué nous indique que pour une différence en entrée fixée, 2 boîtes-S du dernier tour ne sont pas actives.

Dans E2 et Camellia [MSAK99, LHL+02], les attaques différentielles tronquées reposent quant à elles sur le fait qu'une boîte-S soit active ou non active. Ce type d'attaque différentielle ne dépend pas de la boîte-S utilisée dans le système de chiffrement mais plutôt d'une mauvaise diffusion des différences à travers les tours le composant. Ici cette idée que les boîtes-S ne jouent pas un rôle important dans la cryptanalyse différentielle tronquée est entretenue par le fait que la différence relative entre E2 et Camellia est dans le nombre de couches non linéaires par tour. En effet, dans E2 la structure de la fonction de tour est décomposée en 4 parties : 2 passages dans les boîtes-S, une partie linéaire qui mixe les octets entre eux et une addition de clé. Dans Camellia la permutation est sensiblement la même mais pour gagner en temps d'exécution les concepteurs ont choisi d'enlever une des deux couches non-linéaires. La complexité des attaques différentielles tronquées entre les deux systèmes de chiffrement reste la même.

Pour SKIPJACK [KRW99], les différentes attaques différentielles tronquées peuvent casser jusqu'à 30 tours (des 32 tours) du système de chiffrement. Elles sont très différentes de celle de E2 et CAMELLIA puisqu'elles étudient la propagation d'une différence à

travers les tours. C'est-à-dire que la différence en sortie est déterminée par la valeur en entrée mais est valable quelle que soit cette valeur en entrée. L'attaque différentielle tronquée sur SAFER [KB96] est sensiblement du même type que celle sur SKIPJACK.

La cryptanalyse *"stochastique"* de CRYPTON [MG00] peut elle aussi être vue comme une cryptanalyse différentielle tronquée. La particularité ici est que la probabilité de la différentielle tronquée est calculée à l'aide de matrices de transition et repose sur des propriétés de chaîne de Markov pour la cryptanalyse différentielle tronquée. Cette différentielle tronquée a aussi la particularité que la troncature n'est pas faite sur tous les mots de la taille des boîtes-S mais seulement sur une partie de ces mots. Ici l'ensemble des différences en entrée et l'ensemble des différences en sortie sont composés seulement de 16 éléments alors que les boîtes-S sont définies sur 8 bits (au lieu des 2^8 éléments que comporte une boîte-S).

Il existe d'autres attaques différentielles tronquées. Certaines attaques dites différentielles peuvent aussi se classer dans la catégorie des attaques différentielles tronquées.

2.2.6 Lien avec les autres cryptanalyses

Contrairement à ce que l'on peut penser il y a de grosses différences entre une cryptanalyse différentielle et une cryptanalyse différentielle tronquée. Ainsi un système de chiffrement peut résister à la cryptanalyse différentielle car les boîtes-S qui le composent ont de bonnes propriétés différentielles. En revanche il peut comporter des faiblesses pour la cryptanalyse différentielle tronquée si la permutation *"orientée mot"* possède de mauvaises propriétés de diffusion.

On peut remarquer que les systèmes de chiffrement sur lesquels il existe des attaques différentielles tronquées sont aussi souvent vulnérables aux attaques différentielles impossibles (section 2.3). Par exemple il existe des attaque différentielles impossibles sur SKIPJACK [BBS99], SAFER [BEA08], CRYPTON[CKK+01], et E2 [SKU+00]. La cryptanalyse différentielle tronquée ne s'applique pas très bien aux systèmes de chiffrement à flot. Il existe tout de même une différentielle tronquée sur le système de chiffrement à flot SALSA [AFK+08] mais ce système de chiffrement à flot est proche d'un système de chiffrement par bloc. Les attaques différentielles tronquées servent aussi faire des attaques sur les fonctions de hachage basées sur des systèmes de chiffrement par bloc.

2.3 La cryptanalyse différentielle impossible

2.3.1 Définition

La cryptanalyse différentielle impossible à été introduite par Eli Biham, Alex Biryukov et Adi Shamir en 1999 pour cryptanalyser Skipjack [BBS99]. L'idée principale de la cryptanalyse différentielle impossible est de trouver une différentielle (a_1, a_r) qui ne peut jamais arriver, c'est-à-dire une différentielle telle que $P[a_1 \to a_r] = 0$. Plus généralement on pourrait parler de chemin différentiel tronqué impossible puisque cette idée de différentielle impossible s'applique souvent dans ce cas comme le montre l'exemple suivant. On a alors une différentielle tronquée (A_1, A_r) qui est impossible : c'est-à-dire que $P[A_1 \to A_r] = 0$.

Pour la recherche de chemins différentiels impossibles l'attaquant cherche un chemin différentiel tronqué avec probabilité 1 pour un certain nombre de tours du système de

chiffrement et un chemin différentiel tronqué avec probabilité 1 pour un certain nombre de tours de l'inverse de la fonction qui n'est pas en adéquation avec la sortie de la première.

Exemple 2.3. *L'attaque différentielle impossible peut s'effectuer sur un grand nombre de systèmes de chiffrement par bloc itératif* "orientés mots"*. L'exemple le plus connu d'attaque différentielle impossible est celui donné sur 4 tours de l'AES par [CKK+01]. Le motif de cette attaque est représenté sur la figure 2.4.*

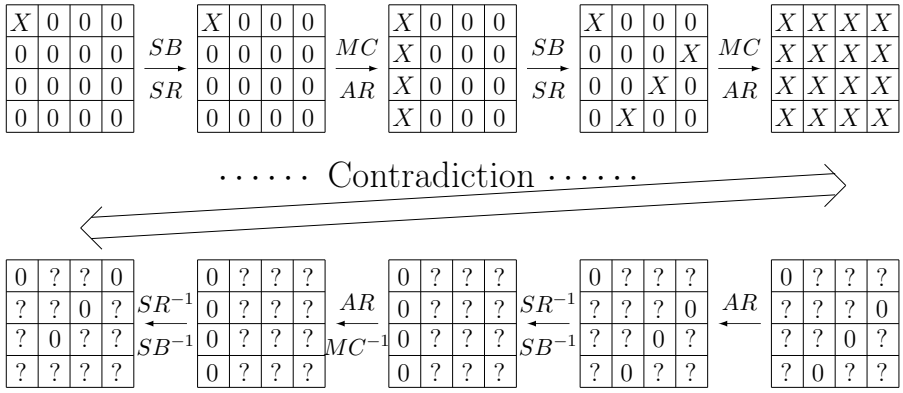

FIGURE 2.4 – Différentielle impossible sur 4 tours de l'AES où X symbolise une différence non nulle qui peut être différente et "?" n'importe quelle différence.

Ce motif a été utilisé par la suite dans de nombreuses attaques contre l'AES.

2.3.2 L'attaque en elle même

L'attaque différentielle impossible est différente de l'attaque différentielle dans le sens où ce n'est pas une attaque sur le dernier tour. En effet dans la cryptanalyse différentielle impossible la différentielle regardée se situe au milieu du système de chiffrement et l'attaquant doit faire des suppositions sur la clé des premiers et derniers tours.

Dans cette section nous donnons l'exemple d'une attaque différentielle impossible d'un système de chiffrement de type SPN ou l'on cherche à retrouver la clé du premier et du dernier tour. Nous supposons que nous avons trouvé une différentielle impossible (A_1, A_r) où A_1 et A_r sont des ensembles de différence après un tour du système de chiffrement et après r tours [4]. Afin de pouvoir détailler l'algorithme utilisé pour l'attaque nous introduisons les notations suivantes pour la clé du premier tour : les clés du premier tour sont notées par h et la clé du premier tour correspondant à la clé utilisée pour chiffrer est notée h^*. Pour les clés du dernier tour on utilise les notations communément utilisées jusque ici (c'est-à-dire k et k^*). On a besoin ici de définir l'ensemble des différences en entrée que l'on doit utiliser pour obtenir une différence dans A_1 après un tour ainsi que le crible utilisé pour supprimer des paires de messages chiffrés :

$$\begin{aligned} A_0 &= \left\{ a \,\middle|\, \exists b \in A_1; P\left[a \underset{F}{\rightarrow} b\right] \neq 0 \right\} \\ \Delta_{sieve} &= \left\{ b \,\middle|\, \exists a \in A_r; P\left[a \underset{F}{\rightarrow} b\right] \neq 0 \right\} \end{aligned}$$

4. On suppose ici que c'est une différentielle tronquée. Ce qui est souvent le cas dans les attaques impossibles.

L'attaque différentielle impossible sur un système de chiffrement de type substitution permutation utilisant ce cadre est décrite dans l'algorithme 4.

Algorithme 4 : Cryptanalyse différentielle impossible d'un système de type substitution-permutation.

Entrée : N_{DC} couples de messages clairs-chiffrés (X, Y) avec $Y = E_{K^*}(X)$.
Sortie : La clé maître K^* utilisée pour chiffrer les échantillons.
Pour chaque $a_0 \in A_0$ **faire**
  Pour chaque *couple* (X, X') *tel que* $X \oplus X' = a_0$ **faire**
    Si $E_{K^*}(X) \oplus E_{K^*}(X \oplus a_0) \in \Delta_{sieve}$ **alors**
      Pour chaque *sous clé du dernier tour* k **faire**
        Si $F_k^{-1}(Y) \oplus F_k^{-1}(Y') \in A_r$ **alors**
          Pour chaque *sous clé du premier tour* h **faire**
            Si $F_h(X) \oplus F_h(X') \in A_1$ **alors**
              Rejeter la sous clé h pour la clé k.
Pour *les clés* (h, k) *restantes* **faire**
  Pour chaque clé maître K correspondant aux sous clés (h, k) **faire**
    Si $E_K(X) = Y = E_{K^*}(X)$ **alors retourner** K;

2.3.3 Les variables aléatoires utilisées dans les attaques différentielles impossibles

Lors de l'attaque comme présentée dans l'algorithme 4, nous n'avons pas besoin de stocker des compteurs pour chaque clé regardée. Il suffit d'avoir une liste de toutes les clés et de les supprimer au fur et à mesure des tests effectués. Pour une étude classique de la complexité en donnée (voir chapitre 5) des attaques statistique nous avons en revanche besoin de connaître la distribution des variables aléatoires utilisées dans les attaques différentielles impossibles.

Définition 2.15. *Soit k une clé candidate pour le dernier tour et h une clé candidate pour le premier tour. Dans le cas de la cryptanalyse différentielle impossible comme définie dans la section 2.3.2, les variables aléatoires étudiées sont*

$$C_{X,(h,k)} = \begin{cases} 1 & si \quad F_h(X) \oplus F_h(X') \in A_1 \\ & \quad et\ F_k^{-1}(Y) \oplus F_k^{-1}(Y') \in A_r \\ 0 & sinon. \end{cases}$$

Les compteurs que nous regardons alors sont $C_{(h,k)} = \sum_X C_{X,(h,k)}$. En utilisant cette définition la liste des clés gardées pour la recherche exhaustive de la clé maître correspond aux clés (h, k) telles que $C_{(h,k)} = 0$.

2.3.4 Lien avec les autres cryptanalyses

Du point de vu de l'attaque en elle même la cryptanalyse différentielle impossible est sensiblement différente d'une cryptanalyse différentielle classique par le fait que l'atta-

quant cherche en même temps à retrouver la clé des premiers et derniers tours mais aussi par le fait que dans l'attaque différentielle impossible usuelle les tests effectués sont là non pas pour incrémenter le compteur correspondant à la bonne clé mais pour éliminer les mauvaises clés.

En ce qui concerne la recherche de la caractéristique différentielle impossible il s'avère que celle ci est assez proche de la recherche de chemin différentiel tronqué. D'ailleurs on peut remarquer que les systèmes de chiffrement qui sont sensibles aux attaques différentielles tronquées sont souvent sensibles aux attaques différentielles impossibles.

Dans la cryptanalyse différentielle impossible on cherche par le milieu deux chemins qui arrivent avec probabilité 1. Or lors d'un passage d'une différence dans les boîtes--S une probabilité arrive. Pour contourner cette probabilité reliée aux propriétés de la boîte-S nous considérons des mots de la taille de la boîte-S. La méthode la plus classique consiste à dire que s'il y a une différence non nulle en entrée d'une boîte-S alors celle ci subsiste à la sortie. C'est une des raisons pour laquelle les systèmes de chiffrement qui comportent des faiblesses contre les attaques différentielles tronquées comportent aussi des faiblesses contre les attaques différentielles impossibles. Ces deux attaques reposent sur des propriétés de mauvaise diffusion dans les systèmes de chiffrements par bloc *"orientés mots"*.

2.4 La cryptanalyse différentielle d'ordre supérieur

2.4.1 Définition

La cryptanalyse différentielle d'ordre supérieur sur les systèmes de chiffrement par bloc a été introduite par Xuejia Lai en 1994 [Lai94] puis par Lars R. Knudsen en 1995 dans le but de cryptanalyser le DES [Knu95].

Dans la cryptanalyse différentielle classique l'étude porte sur l'analyse de la différence entre deux messages clairs et deux messages chiffrés. Pour la cryptanalyse différentielle d'ordre supérieur le principe reste le même mais au lieu d'étudier la dérivée à l'ordre un de la fonction, on s'intéresse aux dérivées d'ordre supérieur.

Définition 2.16. *[Lai94] Soit f une fonction de $\mathbb{F}_2^w$ dans $\mathbb{F}_2^z$. Soit $a_1, a_2, ..a_v$ des vecteurs indépendants de $\mathbb{F}_2^w$. La dérivée d'ordre v de f relativement à $(a_1, a_2, \cdots, a_v)$ est la fonction définie par :*

$$\begin{array}{rccc} D_{\langle a_1,\cdots,a_v\rangle} f : & \mathbb{F}_2^w & \rightarrow & \mathbb{F}_2^z \\ & x & \mapsto & \sum_{a\in\langle a_1\cdots a_v\rangle} f(x+a) \end{array}$$

où $\langle a_1, \cdots, a_v\rangle$ symbolise le sous espace vectoriel engendré par les vecteurs $a_1, a_2, \cdots, a_v$.

Par exemple la dérivée à l'ordre deux suivant les valeurs a et b d'une fonction f au point x est notée $D_aD_bf(x)$ et vaut :

$$\begin{aligned} D_aD_bf(x) &= D_a(f(x) \oplus f(x \oplus b)) \\ &= f(x) \oplus f(x \oplus b) \oplus f(x \oplus a) \oplus f(x \oplus a \oplus b). \end{aligned}$$

Dans cet exemple on voit que l'on a besoin de connecter l'image par la fonction f aux points $x, x \oplus a, x \oplus b, x \oplus a \oplus b$. Ainsi les messages clairs doivent former un espace affine de taille 2^v si on étudie une différentielle d'ordre v.

Comme pour l'étude de la cryptanalyse différentielle la dérivée d'ordre supérieur est définie de façon probabiliste. Mais dans la plupart des attaques différentielles d'ordre supérieur existantes, les attaquants utilisent des différentielles d'ordre supérieur avec probabilité égale à 1. On parle alors de *cryptanalyse différentielle d'ordre supérieur déterministe*.

Soit $deg(f)$ le degré algébrique d'une fonction f. Le fait de dériver une fois la fonction f fait baisser son degré d'au moins un. Ainsi on a la propriété suivante :

Proposition 2.7. *Soit f une fonction de $\mathbb{F}_2^w$ dans $\mathbb{F}_2^z$. Soit V un sous espace de $\mathbb{F}_2^v$ de dimension $deg(f)+1$ on a*

$$D_V f(x) = 0 \quad \textit{pour tout } x \in \mathbb{F}_2^v.$$

L'étude de la résistance d'un système de chiffrement aux attaques différentielles d'ordre supérieur est souvent directement reliée à l'étude du degré du système de chiffrement.

2.4.2 L'attaque

Les attaques différentielles d'ordre supérieur ont pour l'instant été principalement appliquées sur des systèmes de chiffrement ayant une structure de schéma de Feistel comme par exemple Misty [Mat97]. Pour cela l'algorithme 5 décrit l'attaque dans le but de retrouver la clé des deux derniers tours d'un schéma de Feistel en supposant que l'on ait trouvé une différentielle d'ordre supérieur sur les r tours précédents.

On verra dans le chapitre 5, que dans le cas général d'une attaque différentielle d'ordre supérieur déterministe (c'est-à-dire qui arrive avec probabilité égale à 1) il suffit souvent d'utiliser un seul échantillon.

2.4.3 Les variables aléatoires utilisées dans les attaques différentielles d'ordre supérieur

Comme dans toutes attaques statistiques, afin d'évaluer la complexité en donnée nous avons besoin de définir les variables aléatoires que nous étudions

Définition 2.17. *Soit E_K un système de chiffrement avec clé maître K et fonction de tour F. Dans le cas de la cryptanalyse différentielle d'ordre supérieur comme définie dans la partie précédente, les variables aléatoires étudiées sont :*

$$C_{X,k} = \begin{cases} 1 & si \quad D_V F_k^{-1}(E_K(X)) = 0 \\ 0 & sinon. \end{cases}$$

Les compteurs que nous regardons alors correspondent à la somme des variables aléatoires simples :

$$C_k = \sum_X C_{X,k}.$$

Algorithme 5 : Cryptanalyse différentielle d'ordre v sur $r+2$ tours d'un schéma de Feistel.

Entrée : $N_{DC} = 2^v N$. Les N échantillons sont composés de 2^v messages clairs $X_i + V = (X_{i,1}, \cdots X_{i,2^v})$ où V est un espace vectoriel avec $dim(V) = v$ et i correspond au numéro de l'échantillon $1 \leq i \leq N$ et les messages chiffrés correspondants : $(Y_{i,1}, \cdots Y_{i,2^v})$

Sortie : La clé maître K^* utilisée pour chiffrer les échantillons

Initialiser une table C à 0 ;

Pour chaque *structure* **faire**

  Pour chaque *message dans une structure* **faire**

    Pour chaque *sous clé du dernier tour* k_{r+2} *correspondant à* K_{r+2} **faire**

      $(z_{i,j}^{(g)}, z_{i,j}^{(d)}) \leftarrow (Y_{i,j}^{(d)} \oplus f(Y_{i,j}^{(g)} \oplus k_{r+2}), Y_{i,j}^{(g)})$;

      Pour chaque *sous clé de l'avant dernier tour* k_{r+1} *correspondant à* K_{r+1} **faire**

        Calculer $t_{i,j} = f(z_{i,j}^{(g)} \oplus k_{r+1})$;

        Si $\sum_j t_{i,j} \oplus z_{i,j}^{(d)} = 0$ **alors**

          $C[k_{r+1}||k_{r+2}] + +$;

Générer une liste $\mathcal{L}$ de ℓ candidats ayant les plus grandes valeurs de compteur $C[k_{r+1}, k_{r+2}]$;

Pour chaque *candidat* $\left(k_{r+1}, k_{r+2}\right) \in \mathcal{L}$ **faire**

  Pour chaque clé maître K correspondant à la clé k **faire**

    Si $E_K(X) = Y = E_{K^*}(X)$ **alors retourner** K;

2.4.4 Lien entre la cryptanalyse différentielle d'ordre supérieur et d'autres attaques

Il existe différentes variantes de la cryptanalyse différentielle d'ordre supérieur. Dans la section 3.4, nous détaillons le lien avec les attaques intégrales. Mais il existe aussi un lien avec les "zero-sum" appliquées principalement aux fonctions de hachage. En 2009, Jean Phillipe Aumasson et Willi Meier ont trouvés des distingueurs sur certaines fonctions de hachage de la compétition SHA-3 [AM09]. Ces distingueurs, qu'ils ont appelés *"zero-sum distinguisher"*, utilisent des propriétés des dérivées d'ordre supérieur de la fonction étudiée. Ce type d'attaque avait déjà été introduit en partie par Lars Knudsen et Vincent Rijmen dans le cas des cryptanalyses à clés fixées [KR07]. La différence principale de ces attaques avec les attaques différentielles d'ordre supérieur est qu'elles ne peuvent être utilisées que dans le cas où la permutation ne dépend d'aucun paramètre secret car dans ce cas précis on peut utiliser les propriétés des différentielles d'ordre supérieur en commençant par le milieu du système de chiffrement (c'est-à-dire un état intermédiaire). Une étude plus approfondie de ce type de distingueur a été faite par Christina Boura et Anne Canteaut [BC10]. Ces travaux ont permis d'appliquer ce nouveau type d'attaque à un certain nombre de fonctions de hachage.

Chapitre 3

Autres attaques statistiques

Dans le chapitre précédent, nous avons détaillé un certain nombre d'attaques statistiques relatives à la cryptanalyse différentielle. Le nombre d'attaques statistiques sur les systèmes de chiffrement par bloc est élevé ; nous avons présenté celles dont le nom comporte le mot "différentiel" dans le chapitre précédent. Dans ce chapitre nous présentons d'autres attaques statistiques qui ont soit un lien avec la cryptanalyse différentielle, soit un lien avec la cryptanalyse linéaire que nous détaillons ici.
La liste des attaques statistiques que nous présentons ici ne se veut pas exhaustive. Elle illustre la quantité d'attaques qui sont utilisées sur les systèmes de chiffrement par bloc et parfois sur des fonctions de hachage.

3.1 Les attaques "boomerang"

L'attaque boomerang a été introduite en 1999 par David Wagner [Wag99] et a été utilisée par la suite pour attaquer un certain nombre de systèmes de chiffrement par bloc. C'est une généralisation de la cryptanalyse différentielle mais qui ne s'applique pas tout à fait au même contexte puisque l'attaque boomerang est une attaque à messages clairs et chiffrés choisis. C'est-à-dire que l'attaquant doit pouvoir obtenir les chiffrés des messages clairs de son choix mais il doit aussi pouvoir obtenir les messages clairs correspondant à des chiffrés de son choix.

3.1.1 Description

Dans la plupart des attaques boomerang la sous clé que l'on cherche à retrouver est celle du premier tour. Ainsi nous allons décrire l'attaque dans ce cas.

Soit E un système de chiffrement que l'on décompose en trois parties E_0, E_1 et la fonction de tour F : $E = E_1 \circ E_0 \circ F$.

Supposons que l'on ait extrait une différentielle $a \to \tilde{a}$ pour la fonction E_0, et une différentielle $b \to \tilde{b}$ pour la fonction E_1^{-1}avec les probabilités suivantes :

$$p_* = P\left[a \underset{E_0}{\to} \tilde{a}\right] \qquad q_* = P\left[b \underset{E_1^{-1}}{\to} \tilde{b}\right]$$

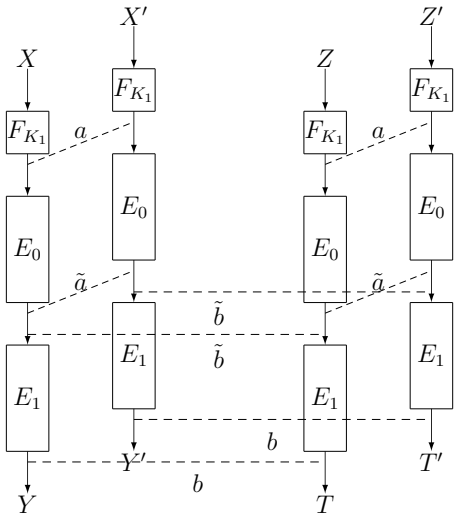

FIGURE 3.1 – Schéma descriptif de l'attaque boomerang

Pour attaquer le système de chiffrement, nous considerons un ensemble de quatre messages clairs X, X', Z, Z' et leur messages chiffrés correspondant Y, Y', T, T'. Dans une attaque boomerang nous voulons que la paire (X, X') suive la différentielle $a \to \tilde{a}$ sur la partie E_0 et que les couples X, Z et X', Z' suivent la différentielle $b \to \tilde{b}$ sur E_1^{-1}. Alors on espère que la paire (Z, Z') suive la différentielle $\tilde{a} \to a$ sur E_0^{-1}. Un schéma descriptif est donné dans la figure 3.1.

Soit A_0 l'ensemble défini par $A_0 = \{a_0 \Big| P\left[a_0 \underset{F}{\to} a\right] \neq 0\}$. L'attaque boomerang utilisant ces différences est résumée dans l'algorithme 6.

3.1.2 Les variables aléatoires étudiées

Soit $C_{X,k}$ et C_k les variables aléatoire définies dans la section 1.5.5. Dans le cas des attaques boomerang les variables aléatoires simples $C_{X,k}$ valent :

$$C_{X,k} = \begin{cases} 1 & \text{si} \quad F_k(X) \oplus F_k(X') = a \text{ et} \\ & \quad F_k\left(E_{K^*}^{-1}(E_{K^*}(X) \oplus b)\right) \oplus F_k\left(E_{K^*}^{-1}(E_{K^*}(X') \oplus b)\right) = a \\ 0 & \text{sinon.} \end{cases}$$

Comme dans beaucoup d'attaques statistiques, les variables aléatoires C_k permettant de générer la liste des clés gardées correspondent à la somme de ces variables aléatoires simples.

Dans le cas où la clé candidate testée est égale à la bonne sous clé k^*, la variable aléatoire C_{k^*} suit une loi binomiale de paramètre N et $p_*^2 q_*^2$.

3.1.3 Lien avec les autres attaques

Naturellement l'attaque boomerang est une généralisation de la cryptanalyse différentielle puisqu'elle utilise des propriétés différentielles sur des versions réduites du système de chiffrement (E_0) et (E_1). L'attaque boomerang est aussi beaucoup utilisée pour les

Algorithme 6 : Attaque boomerang

Entrée : N échantillons composés de deux messages clairs et de deux messages chiffrés
Sortie : La clé maître K^* utilisée pour chiffrer les échantillons
Initialiser une table C de 2^n compteurs à 0 ;
Pour chaque $a_0 \in A_0$ **faire**
  Pour chaque *couple* (X,X')*avec* $X' = X \oplus a_0$ **faire**
    Récupérer les chiffrés correspondants $Y = E_{K^*}(X)$ et $Y' = E_{K^*}(X')$;
    Soit $T = Y \oplus b$ et $T' = Y' \oplus b$;
    Récupérer les clairs correspondants $Z = E^{-1}_{K^*}(T)$ et $Z' = E_{K^*}(T')$;
    Si $Z \oplus Z' \in A_0$ **alors**
      Pour chaque *sous clé candidate* k **faire**
        Calculer $d = F_k(X) \oplus F_k(X')$;
        Calculer $d' = F_k(Z) \oplus F_k(Z')$;
        Si $d = a$ et $d' = a$ **alors** $C[k] \leftarrow C[k] + 1$;
Générer une liste $\mathcal{L}$ de ℓ candidats ayant les plus grandes valeurs de $C[k]$;
Pour chaque $k \in \mathcal{L}$ **faire**
  Pour chaque clé maître K correspondant à la clé k **faire**
    Si $E_K(X) = Y = E_{K^*}(X)$ **alors retourner** K;

fonctions de hachage : on peut citer par exemple l'attaque boomerang sur SHA-1 faite par Antoine Joux et Thomas Perrin [JP07].

3.2 La cryptanalyse linéaire

La cryptanalyse linéaire qui fait partie de la famille des attaques statistiques est assez différente des attaques vues jusqu'ici, puisque qu'elle relève de l'étude des propriétés de non linéarité du système de chiffrement.

3.2.1 La cryptanalyse linéaire

La cryptanalyse linéaire a été introduite par Mitsuru Matsui à Eurocrypt en 1993 [Mat93].

Définition 3.1. *Soit* $(\pi, \kappa, \gamma) \in \mathbb{F}_2^m \times \mathbb{F}_2^\Omega \times \mathbb{F}_2^m$ *un triplet où* π *est appelé* **masque d'entrée**, κ **masque de clé** *et* γ **masque de sortie**[1]. *Soit* F *la fonction de tour d'un système de chiffrement par bloc paramétré par une clé* K. *Une* **approximation linéaire** *sur* r *tours, relative à ce triplet, est*

$$(X, K) \to \langle \pi, X \rangle \oplus \langle \kappa, K \rangle = \langle \gamma, F_K^r(X) \rangle,$$

où $\langle a, b \rangle$ *désigne le produit scalaire dans les corps de caractéristique 2.*

1. On rappelle que Ω désigne le nombre de bits de la clé maitre

Définition 3.2. *Soit (π, κ, γ), un triplet définissant une approximation linéaire. La* **probabilité de cette approximation** *est :*

$$P_{\mathbf{X},\mathbf{K}}\left[\langle \pi, X\rangle \oplus \langle \kappa, K\rangle = \langle \gamma, F_K^r(X)\rangle\right] = \frac{1}{2} + \varepsilon. \tag{3.1}$$

La variable ε est appelé le **biais** *de l'approximation linéaire. C'est un réel positif ou négatif de valeur absolue plus petite que* 0.5

Il est bien évident que les approximations linéaires avec un biais nul ne sont pas intéressantes pour une cryptanalyse linéaire classique car cela correspond au cas de la distribution uniforme [2].

C'est la connaissance de l'évaluation de l'équation linéaire en certains points qui va nous permettre de tirer de l'information sur la clé utilisée pour chiffrer. Dans les attaques linéaires on n'a pas besoin de choisir les couples clair/chiffré utilisés pour récupérer cette information. On dit alors que la cryptanalyse linéaire est une attaque à clairs connus [3]. Dans la pratique, lorsque l'on effectue une cryptanalyse linéaire, la clé que l'on cherche à retrouver est fixée. Ainsi si on se place à clé fixée la probabilité est

$$P_{\mathbf{X}}\left[\langle \pi, X\rangle \oplus \langle \kappa, K\rangle = \langle \gamma, F_K^r(X)\rangle\right] = \frac{1}{2} + \varepsilon_K. \tag{3.2}$$

Dans la pratique le biais peut être différent suivant la clé utilisée pour chiffrer. En théorie on fait l'hypothèse que le biais est le même pour toutes les clés. Cette hypothèse est appelé *hypothèse d'indépendance à clé fixée.*

Hypothèse 3.1. Hypothèse d'indépendance à clé fixée *Soit (π, κ, γ) un triplet définissant une approximation linéaire. Dans les attaques linéaires classiques on suppose que toutes les clés possèdent le même biais pour une approximation linéaire donnée, c'est-à-dire :*

$$P_{\mathbf{X}}\left[\langle \pi, X\rangle \oplus \langle \kappa, K\rangle = \langle \gamma, E_K(X)\rangle\right] = P_{\mathbf{X},\mathbf{K}}\left[\langle \pi, X\rangle \oplus \langle \kappa, K\rangle = \langle \gamma, E_K(X)\rangle\right]$$

3.2.2 Attaque linéaire de type 1 et de type 2

Dans son premier papier, Mitsuru Matsui présente plusieurs méthodes pour effectuer des cryptanalyses linéaires. Ses méthodes sont décrites dans les algorithmes 1 et 2 de [Mat93]. Par la suite dans la littérature les cryptographes ont repris ces attaques en les nommant attaque de type 1 et de type 2 pour faire référence aux deux premiers algorithmes proposés par Mitsuru Matsui. L'attaque de type 3 que nous ne présenterons pas ici est une combinaison des deux précédentes attaques.

Attaque de type 1

Dans cette attaque on cherche à tirer de l'information à partir des chiffrés directement (c'est-à-dire $E_K = F_K^r$). Cette attaque retrouve en général la valeur de 1 bit de la clé maître. Le reste de l'attaque se fait par une recherche exhaustive des bits de la clé.

2. On peut quand même en tirer certaine informations. Cela a fait l'objet de travaux récents.

3. Contrairement à la cryptanalyse différentielle et ses généralisations qui sont des attaques à clairs choisis

Pour une clé maître K^* fixée, dans une attaque de type 1 on s'intéresse à la valeur suivante :

$$P_{\mathbf{X}}\left[\langle \pi, X\rangle \oplus \langle \gamma, E_{K^*}(X)\rangle\right] = \frac{1}{2} + (-1)^{\langle \kappa, K^*\rangle}\varepsilon$$

Supposons que l'on ait à notre disposition N couples clairs chiffrés (X_i, Y_i) avec $Y = E_{K^*}(X)$, dans une attaque de type 1 on compte le nombre de paires (X, Y) qui vérifient l'équation $\langle \pi, X\rangle \oplus \langle \gamma, Y\rangle = 0$. Nous notons par C la variable aléatoire correspondant à la somme des variables aléatoire simple C_X :

$$C_X \stackrel{\text{def}}{=} \begin{cases} 1 & \text{si} \quad \langle \pi, X\rangle \oplus \langle \gamma, Y\rangle = 0 \\ 0 & \text{sinon.} \end{cases} \qquad \text{et} \quad C = \sum_X C_X.$$

Contrairement aux attaques que nous avons présentées jusqu'ici, dans la cryptanalyse linéaire de type 1, on ne cherche pas à retrouver la clé du dernier tour mais de l'information sur un bit de la clé maître. Ainsi comme on étudie la distribution d'un compteur pour une clé fixée, on n'utilise pas la notation classique C_k définie dans la section 1.5.5. Cependant l'analyse de la complexité en donnée faite dans le chapitre 5 reste valable.

À partir de ces compteurs on définit le biais empirique d'une approximation linéaire pour la clé utilisée pour chiffrer par $\hat{\varepsilon} = \frac{C}{N} - \frac{1}{2}$. On s'intéresse alors aux signes du biais empirique $\hat{\varepsilon}$ et du biais théorique ε : si les deux biais ont le même signe alors $\langle \kappa, K^*\rangle = 0$ sinon $\langle \kappa, K^*\rangle = 1$. L'algorithme 7 décrit cette attaque.

Algorithme 7 : Cryptanalyse linéaire de type 1 pour un triplet (π, κ, γ)

Entrée : N couples clair/chiffré (X, Y) avec $y = E_K^*(x_i)$.
Sortie : La clé maître K^* utilisée pour chiffrer les échantillons
Initialiser un compteur C à 0;
Pour chaque *message clair* X **faire**
  Si $\langle \pi, X\rangle \oplus \langle \gamma, Y\rangle$ **alors** $C \leftarrow C + 1$;
Si $\varepsilon\hat{\varepsilon} > 0$ **alors**
  $\langle \kappa, K^*\rangle = 1$.
Sinon
  $\langle \kappa, K^*\rangle = 0$.
Faire une recherche exhaustive de la clé maître.

Attaque de type 2

L'attaque de type 2, porte aussi le nom d'attaque linéaire sur le dernier tour. En effet dans cette attaque on va chercher à retrouver de l'information non pas sur la clé maître directement mais sur la clé du dernier tour [4]. On rappelle les notations qui sont utilisées dans les chapitres précédents. Soit K^* la clé utilisée pour chiffrer. La clé du dernier tour correspondant à la clé maître est notée k^* alors que toutes les autres clés candidates sont notées k. Soit un système de chiffrement E comportant $r+1$ tours $E_K = F_K^{r+1}$. Soit ε le biais théorique d'une approximation linéaire sur r tours du système de chiffrement :

$$P_{\mathbf{X},\mathbf{K}}\left[\langle \pi, X\rangle \oplus \langle \kappa, K\rangle = \langle \gamma, F_K^r(X)\rangle\right] = \frac{1}{2} + \varepsilon.$$

4. La même méthode peut être appliquée pour retrouver la clé des premiers tours et des derniers tours combinés

Supposons que nous ayons N couples de messages clairs, messages chiffrés $(X, Y = E_K^*(X))$. Pour chaque clé du dernier tour on calcule les compteurs

$$C_k = \sum_{i=1}^{N} \langle \pi, X_i \rangle \oplus \langle \gamma, F^{-1}(Y_i) \rangle \oplus 1.$$

Afin de distinguer la bonne sous clé des autres on fait l'hypothèse que la valeur $\left|\frac{C_k}{N} - \frac{1}{2}\right|$ pour $k \neq k^*$ est proche de 0. Cette hypothèse est appelée *hypothèse de répartition aléatoire par fausse clé.*

Hypothèse 3.2. Répartition aléatoire par fausse clé *Soit K^* la clé maître utilisée pour chiffrer. Pour toutes les sous clés k ne correspondant pas à la clé maître on a*

$$P_{\mathbf{X}}\left[\langle \pi, X \rangle \oplus \langle \kappa, K \rangle = \langle \gamma, F_k^{-1}\left(E_K^*(X)\right) \rangle\right] = \frac{1}{2}.$$

La bonne sous clé se distingue alors par le fait que $\left|\frac{C_{k^*}}{N} - \frac{1}{2}\right| \approx \varepsilon$.
L'algorithme 8 décrit cette attaque.

Algorithme 8 : Cryptanalyse linéaire de type 2 pour un triplet (π, κ, γ) définissant une approximation linéaire sur r tours. Cas où $E_{K^*} = F_{K^*}^{r+1}$

Entrée : N clairs X et les chiffrés correspondant $Y = E_K^*(X)$
Sortie : La clé maître K^* utilisée pour chiffrer les échantillons
Initialiser une table $C[k]$ à 0;
Pour chaque *message clair* X **faire**
  Pour chaque *sous clé* k **faire**
    Si $\langle \pi, X \rangle \oplus \langle \gamma, F_k^{-1}(Y_i) \rangle = 0$ **alors**
      $C[k] \leftarrow C[k] + 1$;
Générer une liste $\mathcal{L}$ de ℓ candidats ayant les plus grandes valeurs de $C[k]$;
Pour chaque $k \in \mathcal{L}$ **faire**
  Pour chaque clé maître K correspondant à la clé k **faire**
    Si $E_K(X) = Y = E_{K^*}(X)$ **alors retourner** K;

3.2.3 Distribution des variables aléatoires

La complexité en données, la probabilité de succès et le gain d'une attaque ont été beaucoup étudiés dans le cas de la cryptanalyse linéaire, en particulier par Pascal Junod [Jun01] et par Ali Aydin Selçuk [Sel08]. Ces résultats reposent sur l'étude de la distribution des compteurs utilisés dans ces cryptanalyses.
Si on reprend le cas particulier de la cryptanalyse linéaire de type 2, l'attaque consiste alors à distinguer deux distributions. Dans le cas de la bonne sous clé la variable aléatoire C_{k^*} suit une loi binomiale de paramètres N et p_* avec $p_* = \frac{1}{2} + \varepsilon$ alors que pour toutes les autres clés les variables aléatoires C_k suivent une loi binomiale de paramètres N et p avec $p = \frac{1}{2}$. Contrairement au cas de la cryptanalyse différentielle où l'étude de la complexité en données et de la probabilité de succès peut être difficile, ici cette étude est simplifiée

par l'utilisation d'une approximation gaussienne de la loi binomiale. Cette approximation gaussienne peut se faire grâce au jeu de paramètres que l'on manipule ici. Ainsi dans [Jun01], une étude du gain est donnée en utilisant cette approximation gaussienne et dans [Sel08] une formule de la probabilité de succès est aussi extraite de cette approximation.

3.2.4 La cryptanalyse linéaire multiple et multidimensionnelle

Les attaques linéaires ont été améliorées suivant deux directions. Ainsi on peut distinguer la cryptanalyse linéaire multiple de la cryptanalyse linéaire multidimensionnelle.

La cryptanalyse linéaire multiple a été introduite dans un second article de Mitsuru Matsui [Mat94] qui utilisait deux approximations linéaires sur le DES puis a été reprise dans de nombreux articles notament [JR94, JV03, BCQ04]. Dans l'analyse de la distribution des variables aléatoires on considère que les approximations sont statistiquement indépendantes [5]. Les complexités de ce type d'attaque ont déjà été beaucoup étudiées en particulier par Pascal Junod et Serge Vaudenay dans [JV03] et par Alex Biryukov, Christophe de Cannière et Mickaël Quisquater dans [BCQ04].

L'introduction de la cryptanalyse différentielle multidimensionnelle est quant à elle plus récente puisqu'elle a été introduite dans de nombreux articles écrits par Joo Yeon Cho, Miia Hermelin et Kaisa Nyberg [CHN08, HCN09, HN10]. L'objectif principal de cette approche est de s'affranchir de l'hypothèse faite dans la cryptanalyse linéaire multiple que les approximations linéaires sont indépendantes. Ainsi dans le cas de la cryptanalyse linéaire multidimensionnelle, les compteurs sont définis pour chaque approximation. On obtient alors des vecteurs de 0 et de 1 selon que l'approximation est vérifiée pour une clé ou pas. L'étude théorique repose alors sur la distribution de ces vecteurs.

3.3 La cryptanalyse différentielle-linéaire

3.3.1 Définition

La cryptanalyse "différentielle-linéaire" a été introduite en 1994 par Susan K. Langford et Martin E. Hellman dans [LH94] dans le but de monter une attaque sur 8 tours du DES qui améliore la meilleure attaque différentielle et la meilleure attaque linéaire. Cette attaque comme son nom l'indique est un mélange de l'attaque linéaire et de l'attaque différentielle classique. Les différentes attaques différentielles-linéaires combinent la connaissance d'une différentielle sur les premiers tours du système de chiffrement avec une approximation linéaire sur la fin du système de chiffrement.

Afin de décrire cette attaque nous reprenons les notations définies dans la section 2.1 et la section 3.2 concernant la cryptanalyse différentielle et la cryptanalyse linéaire.

Définition 3.3. *Soit* $(a_0, \kappa, \gamma) \in \mathbb{F}_2^m \times \mathbb{F}_2^\Omega \times \mathbb{F}_2^m$ *un triplet, où* a_0 *est une différence entre deux messages clairs,* γ *est un masque en sortie et* κ *un masque sur la clé. Soit* F *la fonction de tour d'un système de chiffrement par bloc paramétré par la clé* K*. Une*

5. Ce qui n'est pas le cas dans la pratique.

approximation différentielle-linéaire sur r tours est alors définie par

$$X \to \langle \gamma, F_K^r(X) \rangle \oplus \langle \gamma, F_K^r(X \oplus a_0) \rangle = \langle \kappa, K \rangle.$$

Pour une clé fixée, la probabilité d'une approximation différentielle-linéaire est alors définie par

$$P_{\mathbf{X}} \left[\langle \gamma, F_K^r(X) \rangle \oplus \langle \kappa, K \rangle = \langle \gamma, F_K^r(X \oplus a_0) \rangle \right] = \frac{1}{2} + \varepsilon_K,$$

où sous l'hypothèse d'indépendance à clé fixée (voir hypothèse 3.1) le biais ε_K est le même pour toutes les clés.

3.3.2 L'attaque

L'attaque en elle même étant assez simple à comprendre puisque c'est une combinaison d'attaques déjà connues, nous allons juste donner l'algorithme la décrivant (voir l'algorithme 9) dans le cas d'une attaque sur le dernier tour (c'est-à-dire le cas où $E_{K^*} = F_{K^*}^{r+1}$).

Algorithme 9 : Cryptanalyse "différentielle-linéaire" sur le dernier tour utilisant le triplet (a_0, κ, γ) définissant une approximation différentielle-linéaire sur r tours. $(E_{K^*} = F_{K^*}^{r+1})$

Entrée : N couples $(X, X' = X \oplus a_0)$ et les chiffrés correspondants $(Y = E_{K^*}(X), Y' = E_{K^*}(X \oplus a_0))$
Sortie : La clé maître K^* utilisée pour chiffrer les échantillons
Initialiser une table $C[k]$ à 0;
Pour chaque *couple* $(X, X \oplus a_0)$ **faire**
  Pour chaque *sous clé* k **faire**
    Si $\langle \gamma, F_k^{-1}(Y) \rangle \oplus \langle \gamma, F_k^{-1}(Y') \rangle = 0$ **alors**
      $C[k] \leftarrow C[k] + 1$;
Générer une liste $\mathcal{L}$ de ℓ candidats ayant les plus grandes valeurs de $\left| C[k] - \frac{N}{2} \right|$;
Pour chaque $k \in \mathcal{L}$ **faire**
  Pour chaque clé maître K correspondant à la clé k **faire**
    Si $E_K(X) = Y = E_{K^*}(X)$ **alors retourner** K;

Dans le cas de la cryptanalyse "différentielle-linéaire", on ne peut pas utiliser de crible pour supprimer un certain nombre de paires car la caractéristique différentielle n'apparaît que sur les premiers tours du système de chiffrement. De la même façon on ne peut pas utiliser un masque sur l'entrée car la caractéristique linéaire n'apparaît que sur la seconde partie du système de chiffrement.

3.3.3 Les variables aléatoires

Les variables aléatoires simples (voir section 1.5.5) utilisées pour les attaques différentielles-linéaires sont définies par

$$C_{X,k} = \begin{cases} 1 & \text{si} \quad \left\langle \gamma, F_k^{-1}\big(E_{K^*}(X)\big) \right\rangle \oplus \left\langle \gamma, F_k^{-1}\big(E_{K^*}(X \oplus a_0)\big) \right\rangle = 0 \ ; \\ 0 & \text{sinon} \ . \end{cases}$$

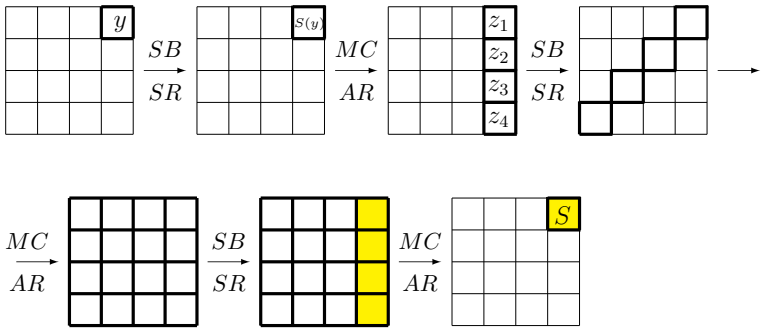

FIGURE 3.2 – Distingueur pour une attaque intégrale sur 3 tours de l'AES

Les variables aléatoire étudiées C_k correspondent à la somme de ces variables aléatoires simples et suivent une loi binomiale de paramètre $(N, \frac{1}{2} + \varepsilon)$ dans le cas où $k = k^*$ ou une loi binomiale de paramètre $(N, \frac{1}{2})$ dans les autres cas. Comme pour la cryptanalyse linéaire de type 2, on obtient alors que la distribution de ces variables aléatoires peut être approchée par une loi gaussienne.

3.4 L'attaque par saturation ou attaque intégrale

La cryptanalyse intégrale a été introduite par Joan Daemen, Vincent Rijmen et Lars R. Knudsen dans [DKR97] [6]. Suite au papier de Stefan Lucks en 2001 [Luc01] elle prend le nom d'*attaque par saturation*. Puis, à FSE en 2002, Lars R. Knudsen et David Wagner [KW02] la renomment *attaque intégrale*.

3.4.1 Description

L'attaque intégrale s'applique relativement bien aux systèmes de chiffrement "orientés mots". Il s'agit de "saturer" un ou plusieurs mots, 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 pour créer un distingueur sur la sortie.

De façon générale, soit Λ l'espace des différences en entrée que l'on veut saturer (c'est-à-dire faire prendre toutes les valeurs). Pour un message fixé en entrée, on fait varier tous les bits correspondant à Λ. Dans une attaque intégrale on s'attend à ce que la somme de tous les chiffrés correspondant à ces $\#\Lambda$ messages clairs soit égale à 0 sur un sous espace.

Exemple 3.1. *Dans cet exemple nous présentons un distingueur bien connu sur 3 tours de l'AES. Ce distingueur permet de monter une attaque sur 6 tours de l'AES. Le distingueur est représenté sur la figure 3.2. Soit un ensemble Λ qui contient* 256 *mots à chiffrer, où les mots sont tous égaux sur* 120 *bits et prennent toutes les valeurs possibles sur l'octet en haut à droite. Si on regarde l'évolution des octets actifs après 3 tours on observe que la somme des* 256 *sorties est égale à 0 sur un octet. Ce distingueur s'applique à toutes les*

6. Cette attaque a été mise en place sur les système de chiffrement SQUARE. En référence à ce papier certains la nomment "square attaque".

boîtes-S en entrée.

Pour simplifier la compréhension de l'attaque qui permet de retrouver une partie de la clé d'un système de chiffrement dans le cadre d'une attaque intégrale, l'algorithme 10 se restreint à la recherche de la clé du dernier tour. Cet algorithme peut évidemment se généraliser facilement pour retrouver la clé du premier ou des premiers tours et/ou la clé des derniers tours.

Algorithme 10 : Attaque intégrale (attaque sur le dernier tour).

Entrée : N échantillons où un échantillon est composé de $\#\Lambda$ messages $X \oplus a$ avec a parcourant Λ et les chiffrés correspondants $Y_a = E_{K^*}(X \oplus a)$.
Sortie : La clé maître K^* utilisée pour chiffrer les échantillons
Initialiser une table C de 2^n compteurs à 0 ;
Pour chaque *échantillon* **faire**
  Pour chaque *sous clé candidate* k **faire**
    $d = 0$;
    Pour chaque $a \in \Lambda$ **faire**
      Calculer $d = d \oplus F_k^{-1}(Y_a)$;
    Si d est égal à 0 sur la partie étudiée **alors** $C[k] \leftarrow C[k] + 1$;
Générer une liste $\mathcal{L}$ de ℓ candidats ayant les plus grandes valeurs de $C[k]$;
Pour chaque $k \in \mathcal{L}$ **faire**
  Pour chaque *clé maître* K correspondant à la clé k **faire**
    Si $E_K(X) = Y = E_{K^*}(X)$ **alors retourner** K;

3.4.2 Les variables aléatoires

Soit Λ un espace de différences en entrée que l'on fait varier et M la fonction qui permet de tronquer la sortie sur les bits qui nous intéressent. Les variables aléatoires utilisées dans une cryptanalyse intégrales sont définies par

$$C_{X,k} = \begin{cases} 1 & \text{si} \quad \bigoplus_{a \in \Lambda} M\Big(F_k^{-1}(E_{K^*}(X \oplus a)\Big) = 0, \\ 0 & \text{sinon.} \end{cases}$$

Comme pour la cryptanalyse différentielle d'ordre supérieur déterministe, les variables aléatoires C_{X,k^*} suivent des lois de Bernoulli avec probabilité 1 alors que pour les autres candidats les variables aléatoires $C_{X,k}$ suivent des lois de Bernoulli de paramètre $\frac{1}{2^{m-\#M}}$ où $\#M$ est la taille de la troncature. Les variables aléatoires qui nous intéressent pour former la liste des clés gardées correspondent à la somme de ces variables aléatoires simples :

$$C_k = \sum_X C_{X,k}.$$

3.4.3 Lien avec les autres cryptanalyses

Lien avec la cryptanalyse différentielle d'ordre supérieur La cryptanalyse intégrale peut être vue comme une attaque différentielle d'ordre supérieur (voir section 2.4).

La différence entre ces deux types d'attaques vient principalement de la manière de trouver un distingueur sur la fonction.
Dans la cryptanalyse d'ordre supérieur on s'intéresse au degré algébrique du système de chiffrement. Cette propriété est directement reliée au degré algébrique des boîtes-S composant le système de chiffrement.
Dans la cryptanalyse intégrale des systèmes de chiffrement orientés mot on s'intéresse plus aux propriétés de diffusion du système de chiffrement, puisque quand on sature une boîte-S en entrée alors la sortie est aussi saturée (si la boîte-S est une permutation).

Lien avec la cryptanalyse linéaire multiple La cryptanalyse intégrale a aussi été appliquée sur des systèmes de chiffrement qui n'étaient pas orientés mots. On peut citer par exemple l'attaque par saturation sur PRESENT faite par Baudoin Collard et François-Xavier Standaert [CS09]. La meilleure attaque linéaire multiple sur PRESENT peut cryptanalyser 26 tours ([Cho10]) et l'attaque par saturation [CS09] est assez proche puisque elle permet de retrouver de l'information sur la clé si le système de chiffrement est réduit à 24 tours [7]. La question s'est alors posée de savoir pourquoi PRESENT était vulnérable "autant" à l'attaque par saturation qu'a l'attaque linéaire multiple. Une étude faite par Gregor Leander en 2011 [Lea11] montre le lien entre ces deux types d'attaques. L'auteur introduit alors une nouvelle famille d'attaques incluant la cryptanalyse linéaire multiple et les attaques par saturation.

3.5 Les attaques à clés liées

Je ne pouvais pas faire une étude des attaques statiques sans parler rapidement des attaques à clés liées [8]. Ces attaques sont moins puissantes que les attaques précédentes car elle supposent que l'on a en sa possession un certain nombre de messages clairs et les chiffrés correspondants pour différentes clés.
La famille des attaques à clés liées est très large. L'idée est, connaissant le lien entre plusieurs clés de retrouver une de ces clés. Les plus connues sont les attaques différentielles à clés liées [Bih94]. Dans les attaques différentielles à clés liées, on suppose que l'on a en possession d'un certain nombre de paires de messages clairs avec une différence donnée a_0 et les chiffrés correspondants par deux clés différentes liées, K et $K + \gamma_0$. Dans cette section nous détaillons le principe de l'attaque différentielle à clés liées.

3.5.1 Attaque différentielle à clés liées

L'attaque différentielle à clés liées a été introduite par Eli Biham [Bih94]. Elle est utilisée pour montrer des faiblesses d'un certain nombre de systèmes de chiffrement et est relativement efficace sur certaines des versions de l'AES. La dernière attaque différentielle à clés liées peut casser 10 des 14 tours de l'AES avec clés de 256 bits [BDK+10].

Définition 3.4. *Soit E_K un système de chiffrement par bloc qui chiffre des messages de m bits avec un clé maître K de taille Ω. On appelle* **caractéristique différentielle à clés liées** *le triplet (a_0, a_r, γ_0) où $a_0 \in \mathbb{F}_2^m$ correspond à une différence entre deux messages en*

7. Au lieu des 31 de la description de PRESENT
8. *"related key differential attacks"*

entrée, $a_r \in \mathbb{F}_2^m$ *correspond à une différence entre les deux messages chiffrés avec des clés différentes et* $\gamma_0 \in \mathbb{F}_2^\Omega$ *correspond à la différence entre les deux clés que l'on utilise.*

L'algorithme 11 décrit cette attaque dans le cas où l'on a trouvé une caractéristique différentielle à clés liées de la forme $(a_0, a_r\gamma_0)$.

Algorithme 11 : Cryptanalyse différentielle à clés liées.

Entrée : N couples $(X, X' = X \oplus a_0)$ et les chiffrés correspondants respectivement par les clés K et $K' = K \oplus \gamma_0$: $Y = E_K(X)$ et $Y' = E_{K'}(X')$
Sortie : La clé maître K
Initialiser une table C de 2^{2n} compteurs à 0 ;
Pour chaque *couple* $(X, X \oplus a_0)$ **faire**
  Pour chaque *couple de sous clés candidates* (k, k') **faire**
    Calculer $d = F_k^{-1}(Y) \oplus F_{k'}^{-1}(Y')$;
    Si $d = a_r$ **alors** $C[(k, k')] \leftarrow C[(k, k')] + 1$;
Générer une liste $\mathcal{L}$ des ℓ candidats ayant les plus grandes valeurs de $C[(k, k')]$;
Pour chaque $(k, k') \in \mathcal{L}$ **faire**
  Rechercher la clé maître correspondante.

3.5.2 Les variables aléatoires

Dans une attaque différentielle à clés liées avec paramètres (a_0, a_r, γ_0), les variables aléatoires simples que nous étudions sont les suivantes :

$$C_{X,(k,k')} = \begin{cases} 1 & \text{si} \quad F_k^{-1}(E_K(X)) \oplus F_{k'}^{-1}(E_{K\oplus\gamma_0}(X \oplus a_0)) = a_r \\ 0 & \text{sinon.} \end{cases} \tag{3.3}$$

Les variables aléatoires qui nous intéressent dans les attaques différentielles à clés liées correspondent à la somme des variables aléatoires simples :

$$C_{(k,k')} = \sum_X C_{X,(k,k')}.$$

Ces variables aléatoires suivent des lois binomiales avec des paramètres différents si les candidats (k, k') correspondent aux clés maîtres (K, K') ou non. L'ordre de grandeur des paramètres étudiés est le même que dans les attaques différentielles classiques.

3.5.3 Lien avec d'autres attaques

Il existe d'autres versions des attaques à clés liées. On peut citer par exemple les attaques boomerang à clés liées. Le principe est alors un mélange entre les attaques différentielles à clés liées et les attaques boomerang (voir section 3.1).

Application aux fonctions de hachage

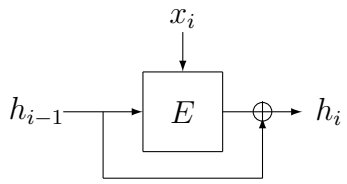

FIGURE 3.3 – Construction Davis-Meyer

Les attaques différentielles à clés liées sur les systèmes de chiffrement par bloc peuvent aussi être utilisées sur les fonctions de hachage utilisant des systèmes de chiffrement par bloc comme primitives. Par exemple dans le cas des fonctions de hachage utilisant le construction *Davis-Meyer* la fonction de compression prend en entrée la valeur de chaînage et la clé du système de chiffrement par bloc est alors composée d'un bloc de message que l'on veut hacher (voir figure 3.3).

Dans les attaques par collision, on cherche alors deux valeurs de chaînages h et $h' = h + a_0$ et deux messages X et $X' = X + \gamma_0$ qui donnent une différence nulle après passage dans la fonction de compression c'est-à-dire une différence $a_r = a_0$ à la sortie du système de chiffrement par bloc.

Chapitre 4

Hypothèses utilisées dans la cryptanalyse différentielle

Dans le chapitre 2, nous avons détaillé les attaques différentielles et leurs généralisations. En particulier nous avons expliqué que la probabilité de succès d'une attaque différentielle repose sur la probabilité de la différentielle utilisée pour attaquer le système de chiffrement. L'attaquant doit trouver une bonne estimation de cette probabilité.

Afin de calculer la probabilité d'un chemin différentiel ou d'une différentielle, beaucoup d'hypothèses sont communément utilisées. L'idée de ce chapitre est de vérifier la validité de ces hypothèses sur des versions réduites du système de chiffrement PRESENT (voir section 1.4.1) appelées SMALLPRESENT. Les expérimentations que nous faisons dans ce chapitre sont faites sur SMALLPRESENT-[4] [1] et sur SMALLPRESENT-[8] [2].

4.1 Les chemins différentiels

Nous allons d'abord vérifier les hypothèses sur les chemins composant les différentielles.

Définition 4.1. *Un* **chemin différentiel** *sur r tours d'un système de chiffrement itératif avec fonction de tour $F_K : \mathbb{F}_2^m \to \mathbb{F}_2^m$ est un $(r+1)$-uplet, $(\beta_0, \beta_1, \cdots, \beta_{r-1}, \beta_r) \in (\mathbb{F}_2^m)^{(r+1)}$, de différences intermédiaires à chaque tour.*

La **probabilité d'un chemin différentiel** $\beta = (\beta_0, \beta_1, \cdots, \beta_{r-1}, \beta_r) \in (\mathbb{F}_2^m)^{(r+1)}$ *est :*

$$P[\beta] \stackrel{\text{def}}{=} P_{\mathbf{X},\mathbf{K}}\left[F_K^i(X) \oplus F_K^i(X \oplus \beta_0) = \beta_i,\ \forall i\right].$$

Calculer la valeur exacte de la probabilité d'un chemin différentiel n'est pas possible pour un système de chiffrement par bloc. En effet, ce calcul nécessite de chiffrer tous les messages avec toutes les clés. Par exemple on aurait une complexité de $\mathcal{O}(2^{128+128})$ dans le cas de l'AES avec une clé de 128 bits ou encore une complexité de $\mathcal{O}(2^{64+80})$ dans le cas de PRESENT.

Dans ce chapitre, on note par $P^t[\beta]$ la probabilité théorique d'un chemin différentiel et par $P^r[\beta]$ la probabilité réelle obtenue par une moyenne sur les messages et sur les clés.

1. SMALLPRESENT-[4] permet de chiffrer des messages de 16 bits.
2. SMALLPRESENT-[8] permet de chiffrer des messages de 32 bits.

4.1.1 Chiffrement de Markov

Pour calculer la probabilité théorique d'un chemin différentiel, une des hypothèses communément utilisée est que le système de chiffrement est de Markov (voir définition 4.2). Cette hypothèse consiste à dire que les différences à chaque tour sont indépendantes. La probabilité d'un chemin différentiel s'obtient en multipliant les probabilités de chaque tour.

Définition 4.2. *[LM91] Un système de chiffrement itératif avec fonction de tour F est de* **Markov** *relativement à la cryptanalyse différentielle si la probabilité de la différence en sortie, connaissant la différence en entrée, est indépendante de la clé utilisée pour chiffrer.*

Soit $\beta = (\beta_0, \beta_1, \cdots, \beta_r)$ un chemin différentiel. Sous l'hypothèse que le système de chiffrement est de Markov, la probabilité théorique du chemin différentiel se calcule de la façon suivante :

Proposition 4.1. *[LM91] Supposons que le système de chiffrement itératif avec fonction de tour F est de Markov. Supposons aussi que les clés de tours sont indépendantes et uniformément distribuées. Alors la séquence des différences $\beta_0, \cdots, \beta_r$ forme une chaîne de Markov. Dans ce cas particulier la probabilité du chemin différentiel $\beta = (\beta_0, \cdots, \beta_r)$ se calcule de la façon suivante :*

$$P^t[\beta] = \prod_{i=1}^{r} P_{\mathbf{X},\mathbf{K}}\left[F_K(X) \oplus F_K(X \oplus \beta_{i-1}) = \beta_i,\ \forall i\right].$$

L'hypothèse que le système de chiffrement est de Markov n'est pas vraie en général (les expérimentations que nous avons faites sur SMALLPRESENT-[4] et que nous détaillons ci-après le montrent). En effet, lors de nos expérimentations sur SMALLPRESENT-[4] nous avons remarqué une dépendance des clés pour certains des chemins.

La probabilité d'un chemin différentiel peut être influencée par la clé utilisée pour chiffrer. Pour illustrer ce phénomène, étant donné un chemin différentiel, nous introduisons des compteurs pour chaque clé :

$$\begin{aligned} T_K &\stackrel{\text{def}}{=} \frac{1}{2}\#\{X \in \mathbb{F}_2^m | F_K^i(X) \oplus F_K^i(X + \beta_0) = \beta_i \quad \forall\ 1 \leq i \leq r\}, \\ T[j] &\stackrel{\text{def}}{=} \#\{K | T_K = j\}. \end{aligned} \tag{4.1}$$

Soit Ω le nombre de bits de la clé maître. La valeur exacte de la probabilité réelle d'un chemin différentiel est alors :

$$P^r[\beta] = 2^{-m-1-\Omega} \sum_{K \in \mathbb{F}_2^\Omega} T_K = 2^{-m-1-\Omega} \sum_j T[j] \cdot j.$$

Nous donnons ici un exemple de chemin différentiel sur SMALLPRESENT-[4] où l'on observe une dépendance de la clé qui donne $P^t[\beta] \neq P^r[\beta]$.

Exemple 4.1. *Nous avons trouvé un chemin différentiel sur 3 tours de SMALLPRESENT-[4] tel que la probabilité théorique est différente de la probabilité réelle.*
Soit $\beta =$ (`0x1101`, `0xdd`, `0x30`, `0x220`) *ce chemin différentiel. Ce chemin différentiel était*

déjà étudié dans la section 2.1. Ainsi la figure 2.1 représentait ce chemin et l'exemple 2.2 donnait le calcul de la probabilité théorique de ce chemin qui est de $P^t[\beta] = 2^{-12}$. *Nous nous sommes intéressés à la probabilité du chemin différentiel pour une clé fixée. Le tableau 4.1 nous donne le nombre de couples de messages clairs qui suivent ce chemin pour une clé fixée. On observe alors sur cet exemple que la probabilité de ce chemin différentiel* $\frac{j}{2^{16}}$ *n'est pas la même pour toutes les clés. Comme cet exemple est petit nous ne distinguons que 3 classes de clés mais dans la réalité ce nombre de classes peut être beaucoup plus important. En utilisant les résultats du tableau 4.1 nous pouvons calculer la probabilité*

j	0	8	16
$T[j]$	131072	524288	393216

Les valeurs $T[j]$ pour le chemin
$\beta = (\texttt{0x1101}, \texttt{0xdd}, \texttt{0x30}, \texttt{0x220})$
$T[j] = \#\{K | T_K = j\}$

TABLE 4.1 – Expérimentations sur SMALLPRESENT-[4]

expérimentale de ce chemin différentiel :

$$\frac{0 \times 131072 + 8 \times 524288 + 16 \times 393216}{2^{20} \times 2^{16}} = 2^{-12.68}.$$

Ce résultat illustre que la probabilité théorique d'un chemin différentiel peut être différente de la probabilité expérimentale pour une clé ou en moyenne sur les clés. Le problème est que pour un système de chiffrement classique nous ne pouvons pas faire une recherche exhaustive pour trouver la probabilité exacte d'un chemin différentiel. Quand il construit une attaque différentielle, l'attaquant ne connaît pas la clé maître utilisée pour chiffrer donc il ne peut pas non plus connaître la probabilité exacte d'un chemin différentiel.
Une autre remarque que l'on peut faire sur cet exemple est que le chemin étudié est impossible pour certaines clés. L'existence de chemins différentiels impossibles est assez classique pour les systèmes de chiffrement par bloc itératifs . L'existence de chemins impossibles sur un petit nombre de tours induit forcément des chemins impossibles sur plus que 3 tours du système de chiffrement. Si le nombre de tours n'est pas trop grand on peut trouver ces chemins impossibles en écrivant les équations dépendant des bits des messages clairs, de la clé et des chiffrés. Malheureusement la complexité de ce système, qui est non linéaire après passage dans les boîtes-S, explose très vite.

4.1.2 L'algorithme "branch and bound" pour trouver les chemins différentiels

Dans le but de trouver les meilleurs chemins différentiels, nous utilisons un algorithme récursif. Cet algorithme connu dans le cas de la cryptanalyse linéaire [BCQ04] peut s'appliquer de la même façon pour la cryptanalyse différentielle. Soit $B_{\text{proba}-\text{chemin}}$ une borne sur la probabilité des chemins que nous voulons garder. L'algorithme consiste à construire un arbre de toutes les différences possibles et à couper les branches pour lesquelles on sait que la probabilité après r tours sera plus grande que la borne donnée. Les algorithmes de ce type sont appelés algorithmes *"branch and bound"*.

La racine de l'arbre que l'on construit est la différence en entrée et les feuilles sont les différences après r tours. Au niveau i, chaque noeud contient la différence après i tours

ainsi que la probabilité du chemin différentiel défini à partir de la racine de l'arbre. Dans la réalité on ne peut pas stocker tous les chemins différentiels. L'astuce consiste alors, à couper les branches de l'arbre avant la fin, si l'on sait que la probabilité du chemin sera supérieure à la borne que l'on s'est fixée. Pour faire cela nous construisons une table des meilleures probabilités pour chaque tour : $[q_1, q_2, \cdots, q_r]$. (q_i est la probabilité du meilleur chemin sur i tours). Imaginons que nous voulons trouver les chemins sur r tours tels que la probabilité de ces chemins soit supérieure à $B_{\text{proba-chemin}}$. Au niveau i nous gardons seulement les chemins avec probabilité p_* tele que $p_* \cdot q_{r-i} \geq B_{\text{proba-chemin}}$. L'algorithme 12 résume ce principe.

Algorithme 12 : Recherche automatique de chemins différentiels

Entrée : Un système de chiffrement, une différence en entrée δ_0, une borne $B_{\text{proba-chemin}}$ sur la probabilité d'un chemin

Sortie : Chemins avec différence en entrée δ_0, avec probabilité plus grande que $B_{\text{proba-chemin}}$, et leurs probabilités

Add-Trail(β ,i)
Si $i = ROUND$ **alors**
  Si $P_\beta \leq B_{\text{proba-chemin}}$ **alors**
    afficher $\beta = (\delta_0, \beta_1, \cdots, \beta_{i+1}, \cdots, \beta_{ROUND})$
  fin si
fin si
Sinon
  Pour chaque β_{i+1} **faire**
    Si $Pr(\beta_i \mapsto \beta_{i+1}) \neq 0$ *and* $P_\beta \cdot Pr(\beta_i \mapsto \beta_{i+1}) \leq B_{\text{proba-chemin}}$ **alors**
      $P_\beta = P_\beta \cdot Pr(\beta_i \mapsto \beta_{i+1})$
      Add-Trail($\beta = (\delta_0, \beta_1, \cdots, \beta_{i+1}, 0, \cdots, 0)$,$i+1$)
    fin si
  fin pour
fin si
Add-Trail($\beta = (\delta_0, 0, \cdots, 0)$,0)

Cet algorithme simple peut être adapté suivant les usages que l'on veut en faire. Pour augmenter la rapidité de la recherche de chemin on peut par exemple imposer des contraintes sur la recherche. Les meilleurs chemins différentiels étant souvent ceux avec peu de boîtes-S actives, on peut, par exemple, limiter le nombre de boîtes-S actives par tour.

Cet algorithme peut aussi être adapté pour trouver des différentielles tronquées (voir section 2.2) ou encore d'autres types de chemins pour d'autres types d'attaques statistiques.

4.1.3 Expériences

Nos expériences sur SMALLPRESENT-[4] nous ont permis d'observer qu'en général la probabilité théorique des chemins différentiels correspondait à la probabilité des chemins différentiels prise en moyenne sur les clés. Pour cela nous avons fait des expérimentations en utilisant différents algorithmes de cadencement de clé.

Dans les figures 4.1, 4.2 et 4.3, nous avons calculé les différences entre $\log(P^t[\beta])$ et $\log(P^r[\beta])$ pour 500 chemins différentiels aléatoires pour 5 tours de SMALLPRESENT-[4].

- Dans la *figure 4.1*, nous supposons que les clés de tour sont obtenues par l'algorithme de cadencement de clés défini dans la section 1.4.1 à partir d'une clé maître de 20 bits. Pour nos expérimentations, nous avons calculé les probabilités moyennes à partir des 2^{20} clés maîtres possibles.
- Dans la *figure 4.2*, nous supposons que toutes les clés de tour sont identiques, c'est-à-dire égales à la clé maître. Nous avons calculé la moyenne des probabilités en utilisant toutes les 2^{16} clés possibles afin d'obtenir la valeur $P^r[\beta]$.
- Dans la *figure 4.3*, nous supposons que les clés de tour sont obtenues à partir d'une clé maître de 80 bits en utilisant l'algorithme de cadencement de clé défini dans la section 1.4.1. Dans ce cas précis, nous ne pouvons pas calculer la moyenne sur les 2^{80} clés possibles. La valeur obtenue pour $P^r[\beta]$ est calculée à partir d'une moyenne sur 2^{20} clés.

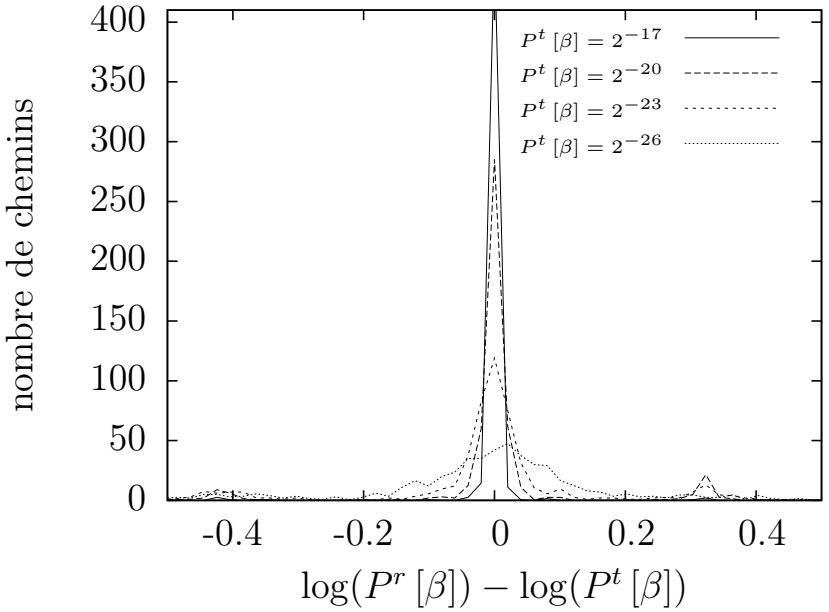

FIGURE 4.1 – Nombre de chemins en fonction du $\log(P^r[\beta]) - \log(P^t[\beta])$: cas de la clé maître de 20 bits.

On remarque ici, au travers de ces expériences, que le phénomène de dépendance d'un chemin différentiel vis à vis des clés n'est pas le même suivant l'algorithme de cadencement de clé. En effet dans la figure 4.2 quand la clé maître est utilisée comme clé de tour pour les 5 tours, la dépendance de la clé est plus importante que dans la figure 4.3 où l'algorithme de cadencement de clé avec une clé maître de 80-bit est utilisé. Nous pouvons remarquer que sur 5 tours de SMALLPRESENT-[4] seulement $16 * 6 = 96$ bits de clés sont utilisés ce qui fait que les bits de la clé maître sont utilisés en moyenne un peu plus d'une fois. Dans ce cas on est proche d'avoir des clés indépendantes (condition nécessaire pour l'utilisation des chaînes de Markov).

L'algorithme de cadencement de clé pour une clé maître de 20 bits semble le plus approprié ici (voir section 1.4.1). En effet on rappelle que PRESENT est un système de chiffrement qui chiffre des messages de 64 bits avec une clé maître de 80, c'est-à-dire que

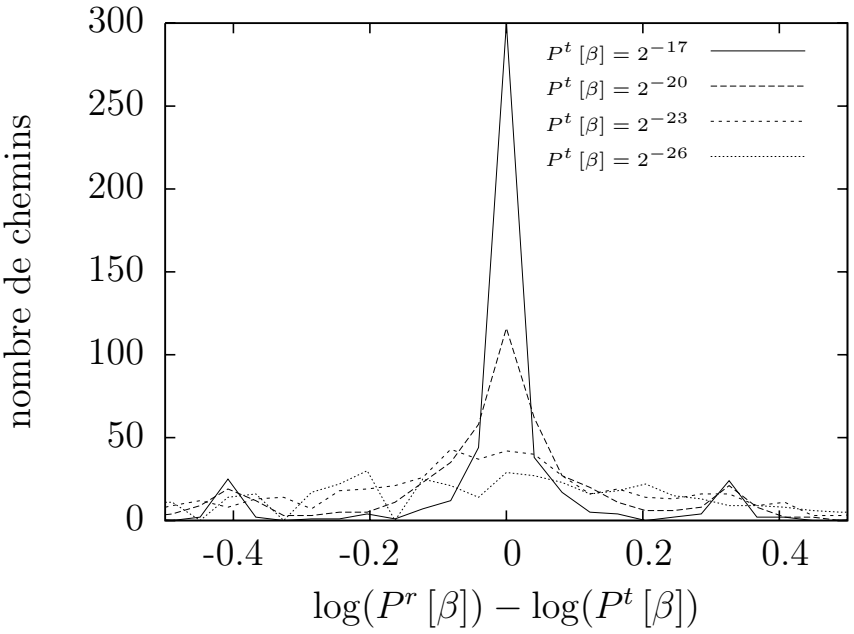

FIGURE 4.2 – Nombre de chemins en fonction de $\log(P^r[\beta]) - \log(P^t[\beta])$: cas où toutes les clés sont identiques.

$\frac{64}{80} = \frac{4}{5}$ des bits de la clé maître sont utilisés à chaque tour. Le rapport est le même si on prend une clé maître de 20 bits pour un système de chiffrement de 16 bits.

Pour cet algorithme de cadencement de clé, les expériences montrent qu'en moyenne le nombre de paires de messages qui satisfont un chemin différentiel est proche de la probabilité théorique.
Nous pouvons observer que ce comportement est de plus en plus mauvais quand la probabilité du chemin diminue

4.2 Les différentielles : somme de chemins

4.2.1 Théorie

Les meilleures différentielles (celles avec grandes probabilités) comportent des chemins avec de grandes probabilités. La probabilité d'une différentielle se calcule aisément à partir de la probabilité des chemins qui la composent.

Lemme 4.1. *Soit (a_0, a_r) une différentielle sur r tours d'un système de chiffrement. Soit p_* la probabilité de la différentielle*

$$p_* = P\left[a_0 \underset{F^r}{\rightarrow} a_r\right].$$

La probabilité de la différentielle est égale à la somme des probabilités des chemins qui la composent.

$$p_* = \sum_{\beta=(\delta_0,\beta_1,\ldots,\beta_{r-1},\delta_r)} p_\beta.$$

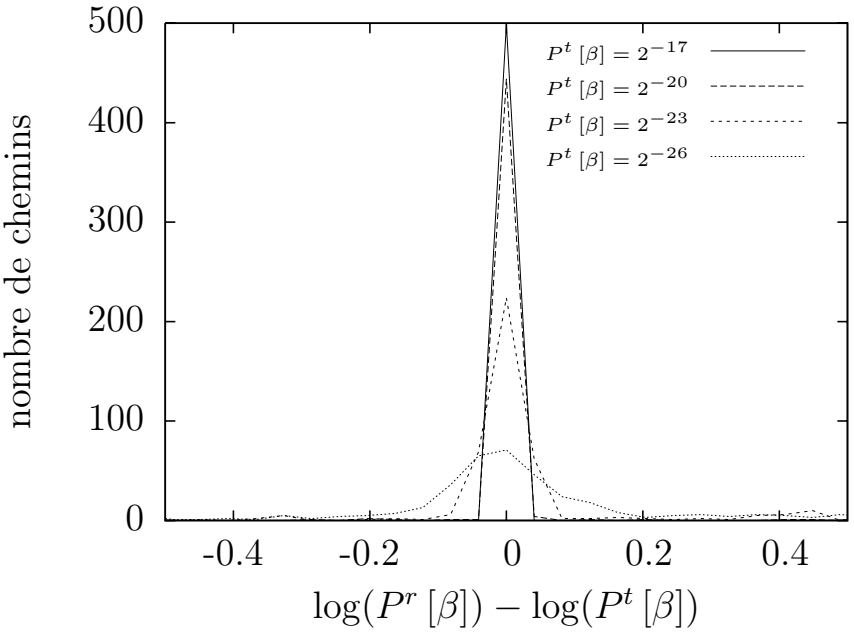

FIGURE 4.3 – Nombre de chemins en fonction de $\log(P^r[\beta]) - \log(P^t[\beta])$: cas de la clé maître de 80-bits.

Dans la section 4.1.1, nous avons montré que dans certains cas la probabilité d'un chemin différentiel pouvait en pratique être différente de la probabilité théorique. Cependant, il semble (dans le cas de PRESENT) y avoir certaines symétries qui font qu'en général il n'y pas de grande différence entre la probabilité théorique d'une différentielle (obtenue en sommant tous les chemins) et la probabilité estimée de celle-ci en prenant la moyenne sur toutes les clés.

Malheureusement, pour certains systèmes de chiffrement où le nombre de chemins est important, plus le nombre de tours augmente moins il est facile de trouver tous les chemins différentiels et donc il devient impossible de calculer la probabilité d'une différentielle. Dans les cryptanalyses courantes, on utilise souvent une borne inférieure de la probabilité d'une différentielle. Cette borne est obtenue en considérant les chemins différentiels les plus probables ou les plus facilement calculables.

4.2.2 Expériences

Pour illustrer la section précédente, nous avons fait des expériences sur 9 tours de SMALLPRESENT-[8] .

Pour une différentielle (a_0, a_r), nous avons calculé la somme de la probabilité théorique d'un certain nombre de chemins composant la différentielle (a_0, a_r). Dans nos expériences sur SMALLPRESENT-[8], nous avons calculé la probabilité effective d'une différentielle en effectuant une moyenne sur 250 clés et sur tous les messages clairs. Sur la figure 4.4 nous avons dessiné la différence entre la valeur théorique et la valeur expérimentale pour 3 différentielles. Nous remarquons que, naturellement, plus on prend de chemins plus la probabilité obtenue est proche de celle de la différentielle.

En regardant les résultats de la figure 4.4, nous pouvons nous demander à partir de combien de chemins, nous avons une bonne estimation de la probabilité de la différentielle.

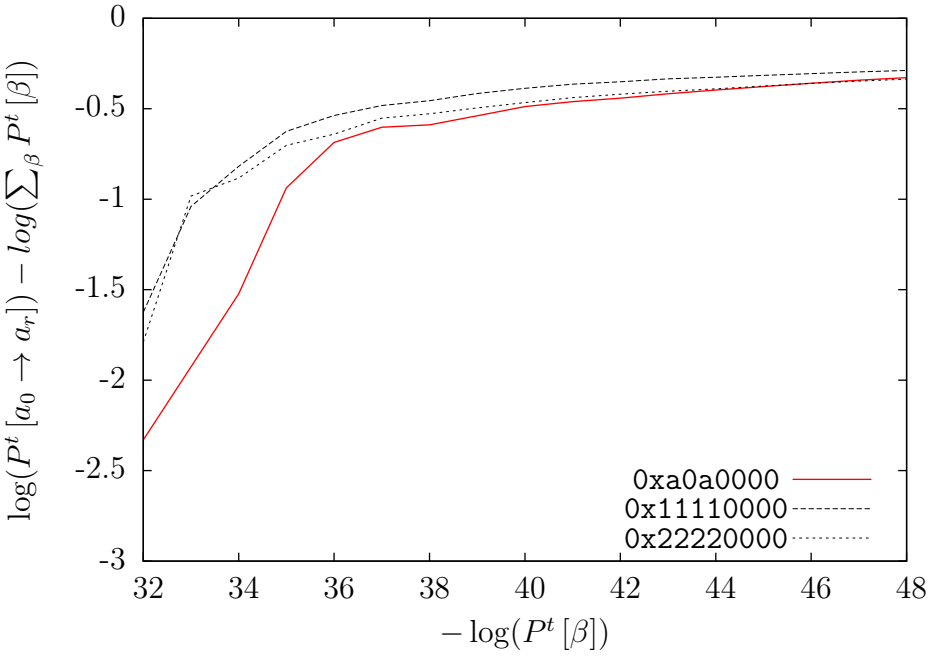

FIGURE 4.4 – Convergence de la somme des chemins vers la différentielle

Dans le cadre de nos expérimentations sur SMALLPRESENT-[4] (voir [BG10]), il avait suffi de prendre des chemins avec probabilité supérieure à 2^{-24} pour avoir une bonne estimation de la probabilité de la différentielle. Ici sur SMALLPRESENT-[8] on voit que prendre tous les chemins différentiels avec probabilité supérieure à 2^{-48} nous donne une erreur de 0.2 dans le logarithme de la probabilité de la différentielle.
Dans le chapitre 6, nous faisons une attaque sur PRESENT. Afin d'estimer la probabilité de nos différentielles nous avons cherché avec l'algorithme de "branch and bound" tous les chemins différentiels avec probabilité supérieure à 2^{-90} ayant 3 boîtes-S actives. Nous savons très bien que prendre seulement ces chemins ne donne pas une bonne estimation de la probabilité de la différentielle mais un calcul exact demanderait beaucoup trop de ressources.

4.3 Les différentielles : dépendance de la clé

Dans la section 4.1.1, nous avons vu que la probabilité d'un chemin différentiel pouvait être dépendante de la clé. Suivant cette remarque il est logique de dire que la probabilité d'une différentielle est elle aussi dépendante de la clé. Dans leur article en 2005, Vincent Rijmen et Joan Daemen [DR05] ont dit qu'en règle générale, pour une différentielle fixée, le nombre de clés pour lesquelles la différentielle avait une certaine probabilité suivait une répartition binomiale.
Dans cette section nous avons voulu vérifier expérimentalement dans le cas de PRESENT si cette hypothèse était vérifiée.

Nous considérons ici une différentielle fixée (a_0, a_r). Pour une clé fixée K nous notons par D_K le nombre de paires de messages clairs avec différence a_0 qui ont une différence

a_r entre leur chiffrés.

$$D_K \stackrel{\text{def}}{=} \frac{1}{2}\#\{X|F_K^r(X) \oplus F_K^r(X \oplus a_0) = a_r\}.$$

Nous nous intéressons à la distribution des

$$D[j] \stackrel{\text{def}}{=} \#\{K|D_K = j\}.$$

Vincent Rijmen et Joan Daemen ont montré que :

La distribution des variables $D[j] \stackrel{\text{def}}{=} \#\{K|D_K = j\}$ est proche d'une loi binomiale.

Ce résultat vient d'une approximation de la loi hypergéométrique de la distribution des variables $D[j]$ par une loi binomiale. Dans le chapitre 5, nous avons utilisé ce résultat afin d'avoir une bonne approximation de la probabilité de succès dans le cas de la cryptanalyse différentielle.

Cependant nous avons voulu vérifier si, dans le cas de PRESENT, les variables aléatoires $D[j]$ suivent bien une distribution binomiale.

4.3.1 Expérimentation

Nous avons fait des expérimentations sur 5 tours de SMALLPRESENT-[4] pour vérifier si les variables $D[j]$ ont bien une distribution binomiale. Ces expérimentations ont été faites à l'aide de l'algorithme de cadencement de clé pour une clé maître de 20 bits. Dans ces expérimentations nous avons calculé la répartition des variables D_K. Sur la figure 4.5 nous remarquons que les variables D_K suivent bien une distribution binomiale.

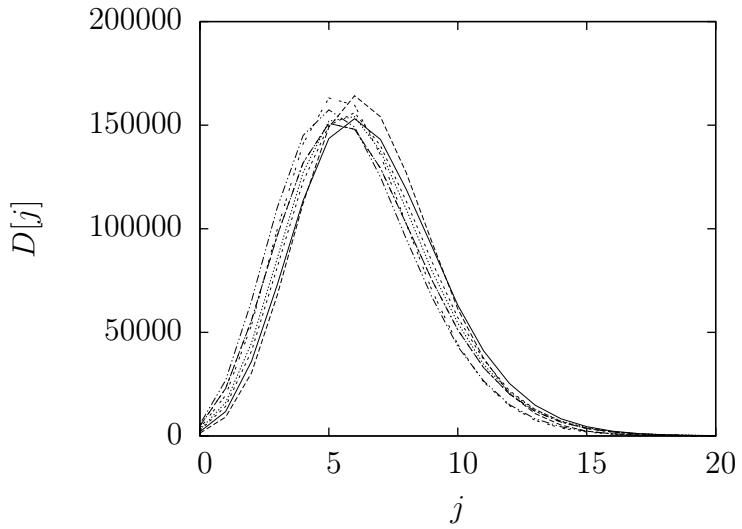

FIGURE 4.5 – Distribution des variables $D[j]$ pour 8 différentielles sur 5 tours de SMALLPRESENT-[4]

Cette observation pourra être prise en compte au moment de calculer la probabilité de succès d'une attaque différentielle. Dans le chapitre 5, nous donnons une formule générale de la probabilité de succès d'une attaque statistique simple. La dernière section de ce chapitre (voir section 5.5) est dédiée à la présentation de la formule de la probabilité de succès d'une attaque différentielle. Cette formule tient compte du fait que pour une différentielle fixée, cette différentielle n'a pas la même probabilité suivant la clé maître utilisée.

4.4 L'hypothèse de répartition aléatoire par fausse clé

4.4.1 Théorie

Dans beaucoup d'attaques statistiques, afin d'étudier la complexité de l'attaque, il est souvent supposé que le phénomène observé a un comportement uniforme pour toutes les clés candidates qui ne correspondent pas à la clé maître. Dans la section 2.1, nous avons écrit cette hypothèse pour le cas de la cryptanalyse différentielle. Nous rappelons ici cette hypothèse.

Hypothèse 4.1 (Hypothèse de répartition aléatoire par mauvaise clé)**.** *Soit $E_{K^*} : \mathbb{F}_2^m \to \mathbb{F}_2^m$ un système de chiffrement par bloc paramétré par la clé K^* avec fonction de tour F. On suppose que*

$$P_{\mathbf{X}}\left[F_k^{-1}(E_{K^*}(X)) \oplus F_k^{-1}(E_{K^*}(X \oplus a_0)) = a_r\right] = \begin{cases} p_* & \text{si } k = k^*, \\ p = \frac{1}{2^m - 1} & \text{pour } k \neq k^*. \end{cases}$$

4.4.2 Expériences

Pour faire ces expériences, nous avons monté des attaques sur différents nombres de tours de SMALLPRESENT-[8]. Dans chacune de nos expériences nous avons utilisé la différentielle

$$(a_0, a_r) = (\texttt{0x7}, \texttt{0x0a0a0000}).$$

Cette différentielle est celle avec la meilleure probabilité sur 7 tours. Pour toutes les paires qui passent le crible, nous avons déchiffré sur un tour et regardé le nombre de messages pour lesquels le compteur $C_{X,k}$ est incrémenté. Ici nous définissons le compteur suivant :

$$W_k = \frac{1}{2}\#\{F_k^{-1}(E_{K^*}(X)) \oplus F_k^{-1}(E_{K^*}(X \oplus a_0)) = a_r\}.$$

Pour une différentielle et une clé maître fixée, nous nous sommes intéressés aux valeurs

$$W[j] = \#\{k | W_k = j\}.$$

Nous avons mené des expérimentations pour différentes clés maîtres aléatoires et pour un nombre de tours variable. D'après nos expérimentations, la clé maître influence peu le résultat. Le résultat de nos expérimentations est représenté sur la figure 4.6.

On peut y voir qu'au bout de 7 tours la distribution des mauvaises clés est plus éparse que lorsque l'on regarde sur plus de tours. De plus lorsque que l'on regarde la probabilité moyenne de la différentielle pour les mauvaises clés, on se rend compte qu'au bout de 7 tours celle-ci n'est pas uniforme. Par contre pour plus de tours on retrouve qu'en moyenne, pour les mauvaises clés la probabilité de la différentielle est proche de $2^{-m} = 2^{-32}$.

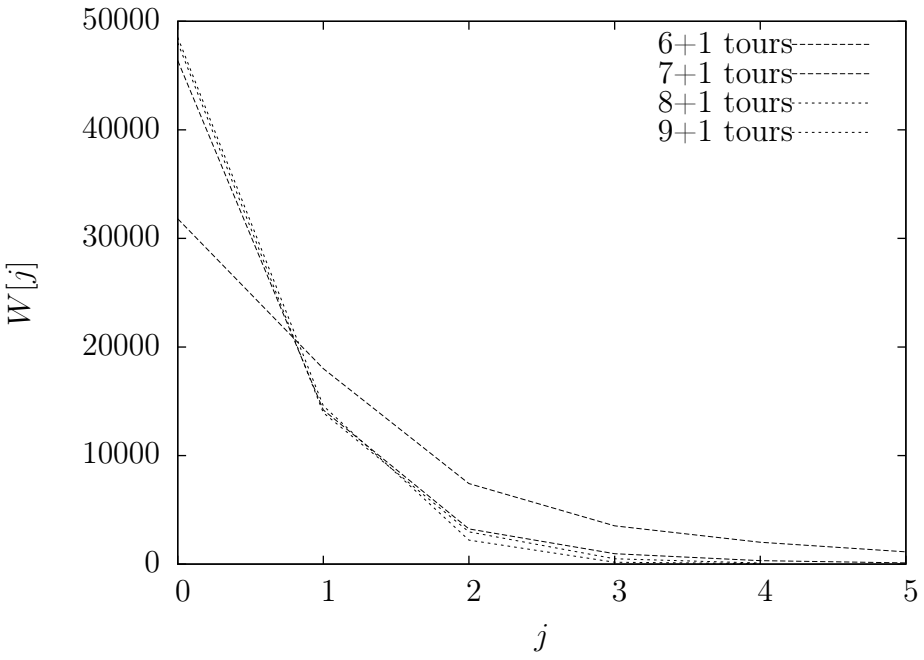

FIGURE 4.6 – Répartition des mauvais candidats. Expérimentation sur SMALLPRESENT-[8]

Le fait que pour les mauvaises clés la distributions des $W[j]$ semble être binomiale, n'influence pas vraiment la probabilité de succès de l'attaque puisque dans chaque attaque on teste toutes les clés candidates. Ce qui est important ici c'est que si on effectue la moyenne sur les clés, la probabilité de la différentielle pour les mauvais candidats est proche de la valeur théorique 2^{-m}.

Les résultats des expérimentations que nous avons présentées dans ce chapitre montrent que les hypothèses faites jusqu'à présent dans les attaques différentielles sont pas tout à fait vérifiées (au moins dans le cas de PRESENT). Toutefois, le fait de faire ses hypothèses n'influence pas vraiment les compléxités des attaques.

Chapitre 5

Complexité en données et probabilité de succès des attaques statistiques simples

Dans les chapitres 2 et 3 nous avons détaillé un certain nombre de cryptanalyses statistiques. Pour évaluer la puissance d'une attaque statistique nous avons besoin d'étudier les complexités en temps, en mémoire, en données ainsi que la probabilité de succès. Il est facile à partir des algorithmes donnés dans les chapitres 2 et 3, d'en déduire la complexité en temps et en mémoire de chacune des attaques statistiques présentées. Ce chapitre est donc dédié à l'étude de la complexité en données et de la probabilité de succès d'un certain nombre d'attaques statistiques comme la cryptanalyse linéaire, la cryptanalyse différentielle, la cryptanalyse différentielle tronquée, la cryptanalyse différentielle d'ordre supérieur et les autres attaques présentées dans les chapitres 2 et 3.

5.1 Introduction

5.1.1 Les variables aléatoires étudiées

Les attaques statistiques sur les systèmes de chiffrement par bloc sont des attaques qui consistent à distinguer plusieurs distributions de probabilités. Dans la plupart des attaques actuelles [1] on se limite à distinguer deux distributions. Les attaques statistiques que nous avons présentées dans les chapitres 2 et 3 comportent un certain nombre de similitudes. En effet le calcul de la complexité de l'étape de distillation (qui consiste à extraire de l'information pour chaque échantillon et pour chaque candidat k testé) se résume souvent à étudier la distribution des variables aléatoires [2]. Nous notons par $C_{X,k}$ la variable aléatoire pour la clé k et le message X. Dans les attaques décrites dans les chapitres 2 et 3 nous avons décrit le cas particulier des variables aléatoires pour chaque type d'attaque.

Par exemple dans le cas de la cryptanalyse différentielle classique (voir section 2.1) les

1. En tout cas celles présentées dans les chapitres 2 et 3
2. Par abus de langage on identifie les termes compteur et variable aléatoire.

variables aléatoires simples sont définies par

$$C_{X,k} = \begin{cases} 1 & \text{si } F_k^{-1}(E_K^*(X)) \oplus F_k^{-1}(E_K^*(X \oplus a_0)) = a_r, \\ 0 & \text{sinon.} \end{cases}$$

Dans les attaques étudiées ici ces variable aléatoires suivent des lois de Bernoulli de paramètre p_* si $k = k^*$ et p sinon.

Afin de déterminer les candidats les plus probables, nous étudions la distribution de la somme de ces variables aléatoires simples.

$$C_k = \sum_X C_{X,k} \qquad \text{et} \qquad C_{k^*} = \sum_X C_{X,k^*} \tag{5.1}$$

Définition 5.1. *Nous appelons* **attaque statistique simple** *les attaques statistiques où les compteurs étudiés afin de déterminer les clés les plus probables suivent des lois binomiales avec seulement deux distributions de probabilités.*

- *On note par C_{k^*} la variable aléatoire correspondant à la bonne sous clé. Soit N le nombre d'échantillons utilisés pour faire une attaque, cette variable aléatoire suit une loi binomiale de paramètre (N, p_*).*
- *On note par C_k $(k \neq k^*)$ les variables aléatoires qui correspondent aux candidats qui ne sont pas la bonne sous clé. Ces variables aléatoires suivent une loi binomiale de paramètre (N, p).*

Remarque 5.1. *Les attaques présentées dans les chapitres 2 et 3 sont des attaques statistiques simples. On verra plus en détail dans le chapitre 6 un exemple d'attaque statistique (la cryptanalyse différentielle multiple) qui ne fait pas partie des attaques statistiques simples. L'étude de la complexité en données et de la probabilité de succès de cette attaque est alors différente et demande une analyse différente.*

5.1.2 Complexité des attaques statistiques simples

Dans ce chapitre nous voulons introduire une étude générale de la complexité en données et de la probabilité de succès des attaques statistiques simples. Des travaux ont déjà été menés pour des cas particulier comme la cryptanalyse linéaire. L'étude générale de la complexité en données et de la probabilité de succès pour l'ensemble des attaques statistiques simples est quant à elle compliquée car les probabilités étudiées sont très différentes suivant le type d'attaque. En effet dans le cas de la cryptanalyse linéaire nous avons $p = 1/2$ et $p_* = p + \varepsilon$. Ainsi l'étude classique de la complexité en données relève d'une approximation gaussienne de la loi binomiale. Malheureusement cette approximation ne peut pas être utilisée dans le cas de la cryptanalyse différentielle car dans ce cas les probabilités étudiées sont de l'ordre de grandeur suivant : $p = 2^m$ et p_* est beaucoup plus grand que p. Dans ce cas là on sait qu'une approximation par la loi de Poisson de la loi binomiale peut être utilisée [Gil97]. L'ordre de grandeur des paramètres pour un certain nombre d'attaques statistiques simples est donné dans le tableau 5.1. Ce tableau illustre bien la variabilité des paramètres étudiés en fonction du type d'attaque.

5.1.3 Les travaux déjà effectués

Comme dit précédemment, des travaux portant sur l'étude de la complexité en données notamment de la cryptanalyse linéaire et différentielle ont déjà été effectués auparavant.

Type de cryptanalyse	p	p_*
Cryptanalyse différentielle	2^{-m}	λp avec λ grand
Cryptanalyse différentielle tronquée	2^{-t}	λp avec λ petit
Cryptanalyse différentielle impossible	2^{-t}	0
Cryptanalyse différentielle d'ordre supérieure(déterministe)	2^{-t}	1
Cryptanalyse linéaire	$1/2$	$1/2 + \varepsilon$
Cryptanalyse différentielle-linéaire	$1/2$	$1/2 + \varepsilon$

TABLE 5.1 – Ordre de grandeur des probabilités pour certaines cryptanalyses statistiques.

Ces travaux reposent essentiellement sur l'utilisation d'une approximation de la loi binomiale par une loi de Poisson pour la cryptanalyse différentielle [BS90] ou par une loi normale pour la cryptanalyse linéaire [Jun01, BJV04, Sel08]. Les sections 5.3.3 et 5.3.4 détaillent le cas particulier de ces deux cryptanalyses. Dans ce chapitre pour calculer la complexité en données dans le cas général d'une attaque statistique simple nous avons besoin d'utiliser une approximation de la loi binomiale qui marche quels que soit les paramètres de la cryptanalyse.

5.2 Estimation de la loi binomiale

Les variables aléatoires utilisées dans la plupart des attaques statistiques présentées dans les chapitres 2 et 3 suivent des lois binomiales. Comme la complexité en données des attaques statistiques est souvent très grande, cette loi est souvent difficile à manipuler. En conséquence, le calcul de la complexité en données et de la probabilité de succès de nombreuses attaques statistiques se base souvent sur une approximation de celle-ci par une loi normale ou de Poisson. Cette section est donc consacrée à la présentation d'une autre estimation des queues de la loi binomiale.

Définition 5.2. *Soit* $\mathbf{X}$*, une variable aléatoire qui suit une loi binomiale de paramètres* N *et* p*. La fonction densité est définie par*

$$P\left[\mathbf{X} = \lfloor \tau N \rfloor\right] = \binom{N}{\lfloor \tau N \rfloor} p^{\lfloor \tau N \rfloor} (1-p)^{N - \lfloor \tau N \rfloor},$$

où $0 \leq \tau \leq 1$*.*

5.2.1 Estimation

Une quantité, nommée divergence de Kullback-Leibler, joue un rôle important dans notre approximation des queues de la loi binomiale.

Définition 5.3. La divergence de Kullback-Leibler [CT91]
Soit $\mathcal{P}$ *et* $\mathcal{Q}$ *deux distributions de probabilités qui suivent une loi de Bernoulli avec paramètres respectifs* p *et* q*. La divergence de Kullback-Leibler entre ces deux distributions est définie par*

$$D\left(p||q\right) \stackrel{\text{def}}{=} p \ln\left(\frac{p}{q}\right) + (1-p)\ln\left(\frac{1-p}{1-q}\right). \tag{5.2}$$

Nous utilisons la convention $0 \ln \frac{0}{p} = 0$ *et* $p \ln \frac{p}{0} = \infty$*.*

À partir de cette définition, nous obtenons une approximation de la fonction densité de la loi binomiale, en utilisant l'approximation de Stirling pour les coefficients binomiaux :

Lemme 5.1. Approximation de Stirling appliquée aux coefficients binomiaux [AS64] *Soit N un nombre entier. Soit T un nombre entier compris entre 0 et $N/2$. Une bonne approximation du coefficient binomial entre N et T est :*

$$\binom{N}{T} = \sqrt{\frac{N}{2\pi T(1-T)}} \left(\frac{N}{N-T}\right)^{(N-T)} \left(\frac{T}{N}\right)^{T} \left[1 - \frac{1}{12T} + \mathcal{O}\left(\frac{1}{N} + \frac{1}{T^2}\right)\right].$$

Cette approximation des coefficients binomiaux nous permet de donner une bonne approximation de la fonction de densité de la loi binomiale.

Lemme 5.2. *Soit C_k une variable aléatoire qui suit une loi binomiale de paramètre (N, p). Soit τ un nombre réel tel que $0 \leq \tau \leq 1$ (τ est appelé le seuil relatif). Nous avons :*

$$P\left[C_k = \lfloor \tau N \rfloor\right] = \sqrt{\frac{1}{2\pi N(1-\tau)\tau}}\, e^{-ND(\tau||p)} \left[1 + \mathcal{O}\left(\frac{1}{\tau N}\right)\right]. \tag{5.3}$$

où $D(\tau||p)$ est la divergence de Kullback-Leibler définie en (5.2)

Preuve : Nous rappelons que la fonction densité de la loi binomiale est :

$$P(C_k = \lfloor \tau N \rfloor) = \binom{N}{\lfloor \tau N \rfloor} p^{\lfloor \tau N \rfloor} (1-p)^{N - \lfloor \tau N \rfloor}.$$

Nous écrivons

$$p^{\tau N}(1-p)^{N-\tau N} = e^{\tau N \ln(p) + (N - \tau N)\ln(1-p)}.$$

Nous utilisons l'approximation de Stirling donnée dans le lemme 5.1 pour $T = \lfloor N\tau \rfloor$:

$$\binom{N}{\lfloor N\tau \rfloor} = \sqrt{\frac{1}{2\pi N\tau(1-\tau)}} e^{-N\tau\ln(\tau) - (1-\tau)\ln(1-\tau)} \left[1 + \mathcal{O}\left(\frac{1}{\tau N}\right)\right]$$

En combinant ces résultats nous obtenons

$$P(C_k = \lfloor \tau N \rfloor) = \sqrt{\frac{1}{2\pi\tau(1-\tau)}} \cdot e^{-N\left[\tau \ln\left(\frac{\tau}{p}\right) + (1-\tau)\ln\left(\frac{1-\tau}{1-p}N\right)\right]} \left[1 + \mathcal{O}\left(\frac{1}{\tau N}\right)\right].$$

□

À partir de notre approximation de la fonction densité de la loi binomiale, nous en déduisons une approximation de la fonction de répartition. Tout d'abord nous introduisons un nouveau lemme qui nous donne un encadrement de la somme de termes consécutifs de la densité de la loi binomiale.

Lemme 5.3. *Soit C_k une variable aléatoire qui suit une loi binomiale de paramètres N et p. Soit A et B deux entiers tels que $0 \leq A \leq B \leq N$. Soit γ_+ et γ_- deux réels définis par*

$$\gamma_+ \stackrel{\text{def}}{=} \frac{1-p}{p} \max\left(\frac{B}{N-B+1}, \frac{A+1}{N-A}\right),$$

$$\gamma_- \stackrel{\text{def}}{=} \frac{1-p}{p} \min\left(\frac{B}{N-B+1}, \frac{A+1}{N-A}\right).$$

Nous avons :

$$P\left[C_k = B\right] \frac{1-\gamma_-^{B-A+1}}{1-\gamma_-} \leq \sum_{i=A}^{B} P\left[C_k = i\right] \leq P\left[C_k = B\right] \frac{1-\gamma_+^{B-A+1}}{1-\gamma_+}, \quad (5.4)$$

$$P\left[C_k = A\right] \frac{1-1/\gamma_+^{B-A+1}}{1-1/\gamma_+} \leq \sum_{i=A}^{B} P\left[C_k = i\right] \leq P\left[C_k = A\right] \frac{1-1/\gamma_-^{B-A+1}}{1-1/\gamma_-}. \quad (5.5)$$

Preuve : Le lien entre deux valeurs consécutives de la fonction de densité de la loi binomiale est le suivant :

$$P\left[C_k = i-1\right] = \frac{1-p}{p} \frac{i}{N-i+1} P\left[C_k = i\right], \text{ pour } 0 < i \leq N.$$

Ce qui conduit à :

$$\sum_{i=A}^{B} P\left[C_k = i\right] = P\left[C_k = B\right] \left[1 + \frac{(1-p)B}{p(N-B+1)} + \cdots + \frac{(1-p)^{B-A} B \cdots (A+1)}{p^{B-A}(N-B+1)\cdots(N-A)}\right].$$

Chaque terme de la somme est majoré par γ_+^i et est minoré par γ_-^i. Nous en déduisons que :

$$P\left[C_k = B\right] \sum_{i=0}^{B-A} \gamma_-^i \leq \sum_{i=A}^{B} P\left[C_k = i\right] \leq P\left[C_k = B\right] \sum_{i=0}^{B-A} \gamma_+^i.$$

Ceci prouve (5.4). Les mêmes arguments sont utilisés pour prouver (5.5).

□

À partir de cette approximation de la fonction densité de la loi binomiale nous pouvons déduire une approximation de la fonction de répartition de cette loi. Le théorème suivant est connu dans un autre contexte [AG89]. Sa preuve découle des deux lemmes précédents.

Théorème 5.1. *Soit p_* et p deux nombres réels tel que $0 < p < p_* < 1$ et soit τ un seuil relatif tel que $p < \tau < p_*$. Soit C_k et C_{k^*} deux variables aléatoires qui suivent une loi binomiale avec paramètres respectifs (N, p) et (N, p_*). Alors,*

$$P\left[C_k \geq \tau N\right] \underset{N\to\infty}{\sim} \frac{(1-p)\sqrt{\tau}}{(\tau-p)\sqrt{2\pi N(1-\tau)}} e^{-ND(\tau||p)}, \quad (5.6)$$

et

$$P\left[C_{k^*} \leq \tau N\right] \underset{N\to\infty}{\sim} \frac{p_*\sqrt{1-\tau}}{(p_*-\tau)\sqrt{2\pi N\tau}} e^{-ND(\tau||p_*)}. \quad (5.7)$$

Preuve : Ce théorème se prouve facilement à partir des lemmes 5.2 et 5.3. Nous nous concentrons sur la preuve de (5.6), l'assertion (5.7) se prouve de la même façon. Dans un premier temps, remplaçons le A du lemme 5.3 par $\lceil \tau N \rceil$ de telle sorte que

$$P\left[C_k \geq \tau N\right] = \sum_{i=A}^{N} P\left[C_k = i\right] = \underbrace{\sum_{i=A}^{B} P\left[C_k = i\right]}_{(1)} + \underbrace{\sum_{i=B+1}^{N} P\left[C_k = i\right]}_{(2)}.$$

Nous appliquons le lemme 5.3 à la somme (1) en choisissant B tel que :

– la somme (2) est négligeable par rapport à (1),
– et γ_+ et γ_- soient proches.

Dans ce cas particulier nous avons

$$\sum_{i=A}^{B} P\left[C_k = i\right] \sim P\left[C_k = \lceil N\tau \rceil\right] \frac{1}{1 - 1/\gamma_-}. \tag{5.8}$$

Sous l'hypothèse que γ_- est proche de γ_+ nous pouvons prendre $\gamma_- = \dfrac{1-p}{p}\dfrac{A+1}{N-A}$ (dans le cas contraire il suffit de faire l'analyse avec γ_+). Dans ce cas particulier nous avons

$$\begin{aligned} \frac{1}{1-1/\gamma_-} &= \frac{(1-p)(A+1)}{(1-p)-p(N-A)} = \frac{(1-p)(A+1)}{N(\tau-p)\left[1+\frac{1-p}{N}\right]} \\ &\sim \frac{(1-p)\tau}{(\tau-p)}. \end{aligned}$$

L'équation (5.6) se déduit donc à partir de cette observation conjuguée avec les résultats de (5.8) et du lemme 5.2. Une étude similaire peux être effectuée pour prouver l'expression (5.7). □

5.2.2 Comparaison avec les autres approximations

Afin d'illustrer les résultats donnés dans le théorème 5.1, avec Benoît Gérard, nous avons fait des expérimentations pour un grand nombre de paramètres. Nous avons pu observer que cette approximation est assez précise pour des valeurs de paramètre vraiment très différentes, que p soit petit ou non ou que τ soit proche de p ou non. Cette approximation de la fonction de répartition comporte alors une grosse différence avec les approximations précédentes de la loi binomiale. En effet l'approximation de Poisson ou l'approximation gaussienne ne sont valables que pour certaines tranches de paramètre. Par exemple dans le cas de la cryptanalyse différentielle la probabilité p est vraiment petite. Dans ce cas, l'approximation de Poisson est juste mais elle n'est plus valide dans le cas de la cryptanalyse linéaire où p est égal à $1/2$. Dans le cas de la cryptanalyse linéaire, l'approximation gaussienne est bonne. Elle a été utilisée dans beaucoup d'études sur la complexité en données ou la probabilité de succès d'une cryptanalyse linéaire ([Jun01, Jun03, JV03, BJV04, BV08, Sel08]). Cependant cette approximation gaussienne n'est pas bonne dans le cas de la cryptanalyse différentielle.

Explication du tableau 5.2

Pour illustrer les propos du paragraphe précédent, nous avons fait des expérimentations afin de comparer les différentes approximations des queues de binomiale. Nous avons choisi de prendre un nombre d'échantillons assez petit ($N = 2^{23}$) afin de pouvoir calculer les valeurs exactes des queues de la loi binomiale. Puis nous avons comparé ce résultat avec les approximations de Gauss, de Poisson et avec notre approximation donnée par le théorème 5.1. Nous avons fait ces calculs pour différentes valeurs de p et p_*. Comme le nombre d'échantillons n'est pas très grand, nous n'avons pas pu prendre des paramètres de cryptanalyses réelles mais nous avons essayé d'avoir des paramètres du même ordre de grandeur que ceux utilisés dans les attaques différentielles, linéaires ou différentielles

tronquées. L'idée principale qui ressort de ces expérimentations est celle décrite ci-dessus, c'est-à-dire que pour la cryptanalyse linéaire l'approximation gaussienne est bonne, que dans le cas de la cryptanalyse différentielle l'approximation de Poisson est bonne mais que dans d'autres cas on ne sait pas quelle approximation utiliser. Notre approximation quant à elle est valable pour un grand nombre de valeurs pour les paramètres p_* et p.

		Binomiale	Poisson	Gauss	Théorème 5.1
Crypt. Linéaire : $p = 0.5$ $p_* = 0.5 + 2^{-10}$	$P[C_k \geq \tau N]$	$8.12 \cdot 10^{-5}$	$3.84 \cdot 10^{-3}$	$8.12 \cdot 10^{-5}$	$8.62 \cdot 10^{-5}$
	$P[C_k \leq \tau N]$	$2.97 \cdot 10^{-2}$	$9.14 \cdot 10^{-2}$	$2.97 \cdot 10^{-2}$	$3.58 \cdot 10^{-2}$
Crypt. Différentielle : $p = 2^{-27}$ $p_* = 2^{-20}$	$P[C_k \geq \tau N]$	$2.03 \cdot 10^{-3}$	$2.03 \cdot 10^{-3}$	$8.84 \cdot 10^{-5}$	$1.97 \cdot 10^{-3}$
	$P[C_k \leq \tau N]$	$3.27 \cdot 10^{-3}$	$3.27 \cdot 10^{-3}$	$6.66 \cdot 10^{-3}$	$3.33 \cdot 10^{-3}$
Crypt. Diff. Tronquée(1) : $p = 2^{-4}$ $p_* = 1.01 \cdot 2^{-4}$	$P[C_k \geq \tau N]$	$9.29 \cdot 10^{-5}$	$1.46 \cdot 10^{-4}$	$9.23 \cdot 10^{-5}$	$9.90 \cdot 10^{-5}$
	$P[C_k \leq \tau N]$	$9.80 \cdot 10^{-5}$	$1.55 \cdot 10^{-4}$	$9.89 \cdot 10^{-5}$	$1.04 \cdot 10^{-4}$
Crypt. Diff. Tronquée(2) : $p = 2^{-15}$ $p_* = 1.5 \cdot 2^{-15}$	$P[C_k \geq \tau N]$	$5.05 \cdot 10^{-5}$	$5.06 \cdot 10^{-5}$	$3.17 \cdot 10^{-5}$	$5.34 \cdot 10^{-5}$
	$P[C_k \leq \tau N]$	$4.37 \cdot 10^{-4}$	$4.38 \cdot 10^{-4}$	$5.45 \cdot 10^{-4}$	$4.67 \cdot 10^{-4}$

TABLE 5.2 – Comparaison entre les différentes approximations des queues de binomiales. Ces valeurs ont été calculées pour $N = 2^{23}$ et $\tau = \frac{p_* + p}{2}$. Les paramètres utilisés ici sont ceux d'une attaque linéaire, d'une attaque différentielle, ou encore deux cas particuliers d'attaque différentielle tronquée

5.3 Complexité en données

Afin de calculer la complexité en données nous avons besoin de définir le contexte dans lequel nous allons nous placer.

5.3.1 Le test d'hypothèses

Pour calculer la complexité en données d'une attaque statistique simple nous avons utilisé un test d'hypothèses statistiques. La problématique dans les attaques statistiques simples est de pouvoir distinguer les compteurs C_k correspondant aux mauvaises clés du compteur C_{k^*} correspondant à la bonne clé dans le cas où les compteurs suivent des lois binomiales. Dans le contexte du test d'hypothèses consistant à distinguer entre les deux distributions il nous faut donc définir un seuil T et comparer les compteurs à ce seuil. Ainsi si $C_k \geq T$ alors on ajoute la clé k à la liste $\mathcal{L}$ des clés admissibles ; dans le cas contraire on rejette le candidat :

$$\text{Si } C_k \geq T \text{ alors } k \in \mathcal{L} \text{ sinon } k \notin \mathcal{L}.$$

Les hypothèses que l'on regarde alors sont
- H_* : Le candidat suggéré correspond à la bonne sous clé k^* ;
- H : Le candidat suggéré n'est pas la bonne sous clé, $k \neq k^*$.

Supposons que nous avons une attaque qui utilise N échantillons. L'attaquant cherche à savoir si la clé qu'il a sélectionnée est la bonne ou pas. Dans ce contexte l'attaquant peut faire deux types d'erreur. Il peut accepter une mauvaise clé ou rejeter la bonne clé. Ces deux types d'erreurs sont communément appelées erreur de première espèce et erreur de seconde espèce ou encore erreur de *non détection* et de *fausse alarme*.

Définition 5.4. *Dans le contexte d'un test d'hypothèses comme défini précédemment, les probabilités d'erreurs étudiées sont :*

La probabilité de non-détection : *Comme son nom l'indique cette erreur correspond au cas où l'attaquant décide de rejeter la bonne clé. On note par α la probabilité de non-détection.*

$$\alpha = P\left[k^* \notin \mathcal{L}\right].$$

La probabilité de fausse alarme : *Cette erreur correspond au cas où l'attaquant décide d'accepter un mauvais candidat. On note par β la probabilité de fausse alarme.*

$$\beta = P\left[k \in \mathcal{L} | k \neq k^*\right].$$

Pour l'instant dans la description du test d'hypothèses nous avons supposé que la statistique étudiée correspondait à la somme de variables aléatoires suivant une loi de Bernoulli (ce qui est le cas des attaques présentées dans les chapitres 2 et 3). Nous allons justifier l'utilisation de cette statistique.

Justification de l'utilisation de cette statistique

Un résultat bien connu permettant de distinguer les deux distributions regardées est le test de Neyman-Pearson.

Lemme 5.4. *[CT91]**Test de Neyman-Pearson** :*
Supposons que le but de notre analyse consiste à distinguer entre deux hypothèses $k = k^*$ et $k \neq k^*$ à l'aide de N variables aléatoires $(C_{X,k})_X$. Si nous utilisons un test de la forme

$$\frac{P\left[C_{X_1 k}, \ldots, C_{X_N,k} | k = k^*\right]}{P\left[C_{X_1 k}, \ldots, C_{X_N,k} | k \neq k^*\right]} \geq t$$

qui nous donne des probabilités d'erreur α et β, alors aucun autre test ne peut améliorer en même temps la probabilité de non-détection et de fausse alarme.

Il est bien connu en théorie des probabilités que la statistique C_k qui consiste à prendre la somme des variables aléatoires qui suivent des lois de Bernoulli indépendantes de paramètre p est une statistique exhaustive [3]. Ainsi utiliser le test de Neyman-Pearson pour distinguer les deux hypothèses est équivalent à comparer la somme des variables aléatoires regardées à un seuil fixé. Les variables aléatoires regardées sont alors définies par :

$$C_k = \sum_X C_{X,k}. \tag{5.9}$$

3. Le terme statistique exhaustive intuitivement signifie que la statistique contient l'ensemble de l'information sur les paramètres de la loi de probabilité.

Nous allons donc par la suite utiliser cette statistique pour calculer la complexité en données des attaques statistiques. La liste des clés gardées par l'attaquant est alors définie de la façon suivante :

$$\mathcal{L} \stackrel{\text{def}}{=} \{k\ ,\ C_k \geq T\}\,.$$

5.3.2 Méthode générale pour calculer la complexité en données d'une attaque statistique

Les probabilités de non-détection et de fausse alarme (voir définition 5.4) jouent un rôle important dans la complexité d'une cryptanalyse statistique.

- Pour une cryptanalyse, la taille ℓ de la liste $\mathcal{L}$ des clés gardées augmente si on diminue le seuil T puisque l'on accepte plus de candidats. La complexité en temps de l'attaque étant souvent reliée à la taille de la liste $\mathcal{L}$, celle-ci varie sensiblement en fonction de β.
- De la même façon si on augmente la valeur du seuil, on accepte moins de candidats. On a alors un plus grand risque de passer à côté de la bonne clé (dans ce cas l'attaque échoue). La probabilité de non-détection α est alors directement reliée à la probabilité de succès de l'attaque.

D'où l'intérêt pour le cryptanalyste de fixer la probabilité de non-détection et de fausse alarme pour étudier les complexités d'une attaque donnée.

Étant données des probabilités d'erreurs fixées, cette section est dédiée au calcul du nombre d'échantillons nécessaire à la cryptanalyse. Le contexte dans lequel nous nous plaçons dans cette section est le suivant :

Contexte. *Soit C_k (resp. C_{k^*}) une variable aléatoire qui suit une loi binomiale avec paramètres N et p (resp. p_*). Dans le contexte du test d'hypothèses défini précédemment, les probabilités de non-détection et de fausse alarme, pour un seuil donné T, sont définies par $P(C_{k^*} < T)$ et $P(C_k \geq T)$.*

Soit b_{nd} et b_{fa} [4] deux nombres réels donnés ($0 < b_{\text{nd}}, b_{\text{fa}} < 1$). Le problème consiste à trouver un nombre d'échantillons N et un seuil T tels que les probabilités d'erreurs soient plus petites que les bornes données b_{nd} et b_{fa}. Ceci est équivalent à trouver une solution (N, T) au système d'inéquations suivant :

$$\left\{ \begin{array}{rcl} P\left[C_{k^*} < T\right] & \leq & b_{\text{nd}}, \\ P\left[C_k \geq T\right] & \leq & b_{\text{fa}}. \end{array} \right. \tag{5.10}$$

En pratique, résoudre ce système d'inéquations est assez compliqué.

La continuité, approximation de la loi binomiale

Les quantités regardées ne sont pas continues puisque la loi binomiale est discrète. Donc, nous avons besoin de trouver une approximation des probabilités d'erreurs qui prennent en entrée des paramètres N et τ où le seuil relatif τ est la quantité égale à $\frac{T}{N}$ (nous avons $0 \leq \tau \leq 1$). Une bonne approximation des probabilités d'erreur peut être donnée par exemple par les formules du théorème 5.1. Comme il existe d'autres approximations

4. b_{nd} correspond à une borne pour la probabilité de non détection et b_{fa} à une borne pour la probabilité de fausse alarme

de la loi binomiale, nous allons dans un cas général définir deux fonctions $G_{\mathrm{nd}}(N,\tau)$ et $G_{\mathrm{fa}}(N,\tau)$ qui sont des estimations des probabilités de non-détection et de fausse alarme. Dans le but de trouver une solution au système (5.10), on demande à ces estimations des probabilités de non-détection et de fausse alarme d'être des fonctions décroissantes en fonction des deux variables N et τ. De cette façon, le problème consistant à trouver le nombre d'échantillons nécessaires à la cryptanalyse pour des probabilités d'erreur fixées se résume à trouver N et τ tels que

$$\begin{cases} G_{\mathrm{nd}}(N,\tau) & \leq & b_{\mathrm{nd}}, \\ G_{\mathrm{fa}}(N,\tau) & \leq & b_{\mathrm{fa}}. \end{cases} \tag{5.11}$$

Description de l'algorithme

Pour un seuil relatif τ donné, nous définissons les quantités $N_{\mathrm{nd}}(\tau)$ et $N_{\mathrm{fa}}(\tau)$ comme les valeurs vérifiant les équations :

$$G_{\mathrm{nd}}(N_{\mathrm{nd}}(\tau),\tau) = b_{\mathrm{nd}} \text{ et } G_{\mathrm{fa}}(N_{\mathrm{fa}}(\tau),\tau) = b_{\mathrm{fa}}. \tag{5.12}$$

Dans le cas où l'une des valeurs $N_{\mathrm{nd}}(\tau)$ ou $N_{\mathrm{fa}}(\tau)$ est plus grande que l'autre, nous devons changer la valeur du seuil relatif dans le but de rapprocher les quantités $N_{\mathrm{nd}}(\tau)$ et $N_{\mathrm{fa}}(\tau)$ l'une de l'autre. D'un autre côté, pour un nombre d'échantillons fixé, si on diminue la valeur du seuil relatif τ alors on accepte plus de candidats. Dans ce cas la probabilité de non-détection diminue alors que la probabilité de fausse alarme augmente. La méthode que nous avons utilisée pour calculer la complexité en données d'une attaque statistique est basée sur l'observation faite dans le lemme suivant.

Lemme 5.5. *Soit $G_{nd}(N,\tau)$ et $G_{fa}(N,\tau)$ deux fonctions qui dépendent des paramètres N et τ, définies sur $[0,+\infty[\times [p,p_*]$, avec les propriétés suivantes :*

- *pour un seuil τ, ces deux fonctions sont décroissantes en N,*
- *pour N fixé, $G_{nd}(N,\tau)$ est croissante en τ,*
- *pour N fixé, $G_{fa}(N,\tau)$ est décroissante en τ,*
- $\lim\limits_{N\to 0} G_{nd}(N,\tau) \geq 1$, $\lim\limits_{N\to 0} G_{fa}(N,\tau) \geq 1$,
- $\lim\limits_{N\to\infty} G_{nd}(N,\tau) = \lim\limits_{N\to\infty} G_{fa}(N,\tau) = 0$.

Soient b_{nd} et b_{fa} les bornes sur les probabilités d'erreurs $0 \leq b_{nd}, b_{fa} \leq 1$. Soit le seuil relatif τ compris entre p et p_*. Soient les quantités $N_{nd}(\tau)$ et $N_{fa}(\tau)$ définies par (5.12) telles que

$$G_{nd}(N_{nd}(\tau),\tau) \;=\; b_{nd} \quad et \quad G_{fa}(N_{fa}(\tau),\tau) \;=\; b_{fa}.$$

Nous introduisons une nouvelle quantité $N(\tau)$ qui est égale au maximum de $N_{nd}(\tau)$ et de $N_{fa}(\tau)$:

$$N(\tau) = \max(N_{nd}(\tau), N_{fa}(\tau)).$$

Cette quantité représente la valeur minimale de N telles que le couple (N,τ) soit solution de (5.11). Alors, pour $p \leq w \leq p_*$,

(i) si $N_{nd}(w) > N_{fa}(w)$, alors, pour tout $\tau > w$, $N(\tau) > N(w)$,

(ii) si $N_{nd}(w) < N_{fa}(w)$, alors, pour tout $\tau < w$, $N(\tau) > N(w)$.

Preuve : Nous allons prouver la propriété (i). La propriété (ii) se montre de la même façon. Nous voulons prouver que si $N_{\mathrm{nd}}(w) > N_{\mathrm{fa}}(w)$, alors, pour tout $\tau > w$, $N(\tau) > N(w)$. Puisque $N_{\mathrm{nd}}(w) > N_{\mathrm{fa}}(w)$, nous avons $G_{\mathrm{nd}}(N(w), w) = b_{\mathrm{nd}}$ et $G_{\mathrm{fa}}(N(w), w) < b_{\mathrm{fa}}$. En utilisant les propriétés de croissance et de décroissance des fonctions G_{nd} et G_{fa}, nous pouvons dire que pour $\tau > w$, nous avons $G_{\mathrm{nd}}(N(\tau), \tau) > b_{\mathrm{nd}}$ et $G_{\mathrm{fa}}(N(w), \tau) < b_{\mathrm{fa}}$. Puisque les fonctions étudiées sont décroissantes en N, nous en déduisons que $N(\tau) > N(w)$. □

À partir de ces observations nous en déduisons l'algorithme 13, ci après, qui est basé sur une recherche dichotomique en τ.

Algorithme 13 : Calcul de la valeur exacte du nombre d'échantillons nécessaire à une cryptanalyse statistique simple et du seuil relatif correspondant.

Entrée : Des probabilités d'erreurs $(b_{\mathrm{nd}}, b_{\mathrm{fa}})$ et les probabilités (p_*, p) relatives à notre attaque.

Sortie : N et τ : le nombre minimum d'échantillons et le seuil relatif correspondant nécessaire pour atteindre des probabilités d'erreur plus petites ou égales à $(b_{\mathrm{nd}}, b_{\mathrm{fa}})$.

Initialiser τ_{min} à p et τ_{max} à p_*;

Faire

  Changer la valeur de τ à $\dfrac{\tau_{min} + \tau_{max}}{2}$;

  Calculer $N_{\mathrm{nd}}(\tau)$ vérifiant $\forall N > N_{\mathrm{nd}}(\tau), G_{\mathrm{nd}}(N, \tau) \leq b_{\mathrm{nd}}$;

  Calculer $N_{\mathrm{fa}}(\tau)$ vérifiant $\forall N > N_{\mathrm{fa}}(\tau), G_{\mathrm{fa}}(N, \tau) \leq b_{\mathrm{fa}}$;

  Si $N_{nd}(\tau) > N_{fa}(\tau)$ **alors**

    $\tau_{\max} = \tau$;

  Sinon

    $\tau_{min} = \tau$;

tant que $N_{nd}(\tau) \neq N_{fa}(\tau)$;

Retourner $N = N_{nd}(\tau) = N_{fa}(\tau)$ *et* τ

Détail d'implémentation

Supposons que nous ayons choisi d'utiliser pour G_{fa} et G_{nd} les approximations définies dans le théorème 5.1. Un moyen simple de calculer les valeurs de $N_{\mathrm{fa}}(\tau)$ et $N_{\mathrm{nd}}(\tau)$ à chaque étape de l'algorithme 13 consiste à faire une recherche dichotomique sur N. Cette méthode peut être améliorée en fixant certaines quantités. En effet si on fixe la probabilité de non-détection à b_{nd}, l'équation (5.7) peut se récrire :

$$N \sim \frac{1}{D\left(\tau || p_*\right)} \ln\left(\frac{p_* \sqrt{1-\tau}}{b_{\mathrm{nd}}(p_* - \tau)\sqrt{2\pi N \tau}}\right).$$

Nous pouvons utiliser une méthode du point fixe pour calculer N. Le détail des conditions d'utilisation de cette méthode est présenté dans la section 5.3.5. Cette méthode peut être

utilisée en commençant l'itération au point $\frac{1}{D(\tau||p_*)}$. La même méthode peut être utilisée pour le calcul de $N_{\text{fa}}(\tau)$ en prenant la fonction (provenant de l'équation (5.7))

$$f(N) = \frac{1}{D(\tau||p)} \ln\left(\frac{(1-p)\sqrt{\tau}}{b_{\text{fa}}(\tau-p)\sqrt{2\pi N(1-\tau)}}\right)$$

et en partant du point $\frac{1}{D(\tau||p)}$.

Il est aussi possible de calculer N en partant de la loi binomiale directement ou d'une approximation assez fidèle de celle-ci. La loi binomiale n'étant pas continue, elle ne satisfait pas les conditions du lemme 5.5 et donc une solution peut ou ne peut pas exister. Donc si on utilise la loi binomiale elle même on autorise une certaine marge d'erreur sur les probabilités de non détection et de fausse alarme que l'on veut atteindre. L'autre problème soulevé par cette méthode est le temps mis par l'algorithme pour calculer les valeurs intermédiaires de N. Nous allons cependant voir comment calculer de manière efficace la fonction de répartition de la loi binomiale.

Calcul efficace des queues de la loi binomiale

Pour calculer efficacement les queues de la loi binomiale nous utilisons l'approximation de Stirling donnée dans le lemme 5.1. Si C_{k^*} suit une loi binomiale avec paramètres N et p_*, nous avons

$$P[C_{k^*} = T-1] = \frac{1-p_*}{p_*} \cdot \frac{T}{N-T+1} \cdot P[C_{k^*} = T].$$

Ce qui nous donne :

$$\begin{aligned} P[C_{k^*} < T] &= P[C_{k^*} = T] \cdot \left[\frac{(1-p_*)\cdot T}{p_* \cdot (N-T+1)} + \frac{(1-p_*)^2 \cdot T(T-1)}{p_*^2 \cdot (N-T+1)(N-T+2)} + \cdots\right] \\ &= P[C_{k^*} = T] \cdot \sum_{i=1}^{T} \left(\frac{1-p_*}{p_*}\right)^i \frac{T!}{(T-i)!}\frac{(N-T)!}{(N-T+i)!} \\ &= \binom{N}{T} p_*^T (1-p_*)^{N-T} \cdot \sum_{i=1}^{T} \left(\frac{1-p_*}{p_*}\right)^i \frac{T!}{(T-i)!}\frac{(N-T)!}{(N-T+i)!}. \end{aligned}$$

En utilisant l'approximation de Stirling donnée dans le lemme 5.1 nous obtenons :

$$P[C_{k^*} < T] \simeq \frac{2^{-ND\left(\frac{T}{N}||p_*\right)}}{\sqrt{2\pi(1-\frac{T}{N})T}} \cdot \sum_{i=1}^{T} \left(\frac{1-p_*}{p_*}\right)^i \frac{T!}{(T-i)!}\frac{(N-T)!}{(N-T+i)!}. \tag{5.13}$$

L'astuce nécessaire pour pouvoir calculer les queues de la loi binomiale consiste à remarquer que le terme dominant est le dernier terme. En conséquence, nous commençons avec ce terme et nous ajoutons les suivants jusqu'à avoir atteint une certaine précision. Cette estimation est précise quand N et T sont assez grand. Quand T est petit [5] nous pouvons utiliser la formule exacte de la loi binomiale. La même méthode peut être utilisée pour calculer l'erreur de fausse alarme.

5. C'est le cas dans les attaques différentielles classiques

5.3.3 Travaux relatifs dans le cas de la cryptanalyse linéaire

Beaucoup de travaux concernant l'étude des complexités des attaques linéaires ont été fait auparavant. Certaines de ces études avaient déjà exhibées le terme polynomial ou le terme exponentiel des formules données dans le théorème 5.1. Nous allons donc revenir rapidement sur ces méthodes.

Le comportement exponentiel des queues de la loi binomiale

Les queues de la loi binomiale sont bien connues pour décroître exponentiellement en N. Le bon terme exponentiel (c'est-à-dire celui donné par le théorème 5.1) a déjà été donné dans plusieurs articles. Par exemple, dans [BJV04, BV08], le but des auteurs était de déterminer une formule asymptotique pour le meilleur distingueur, c'est-à-dire le distingueur qui maximise la quantité $|1-\alpha-\beta|$ où α et β sont les probabilités d'erreurs définies dans la définition 5.4. De leur étude sur la complexité en données ils déduisent le résultat suivant :

$$\max(\alpha,\beta) \doteq 2^{-N\cdot C_{her}(p_*,p)} \tag{5.14}$$

où $f(N) \doteq g(N)$ signifie que $f(N) = g(N)e^{o(N)}$ et C_{her} est l'information de Chernoff [CT91]. Dans le cas général où $p_*, p \notin \{0,1\}$, ce type de distingueur a une région d'acceptation de la forme décrite par le lemme 5.4 avec t égal à 1. Dans ce contexte, la valeur du seuil relatif τ satisfait l'égalité $D(\tau||p_*) = D(\tau||p)$. Cependant, dans notre contexte la valeur de la divergence de Kullback-Leibler est égale à l'information de Chernoff $C_{her}(p_*,p)$ multipliée par $\ln(2)$ (voir [CT91, section 12.9]). Donc le terme exponentiel des équations (5.6) et (5.7) est le même que celui donné par l'équation (5.14) :

$$\alpha \doteq e^{-ND(\tau||p_*)} \doteq 2^{-NC_{her}(p_*,p)} \quad \text{et} \quad \beta \doteq e^{-ND(\tau||p)} \doteq 2^{-NC_{her}(p_*,p)}.$$

Dans les cas où $p_* = 0$ ou $p_* = 1$ [6] le seuil relatif τ est égal à p_* et la probabilité de non détection est alors égale à 0. Dans ce cas on a

$$\max(\alpha,\beta) = \beta \doteq e^{-ND(p_*||p)} \doteq 2^{-NC_{her}(p_*,p)}.$$

Cette égalité se déduit directement de la définition de la divergence de Kullback-Leibler. Le comportement exponentiel $N\cdot D(\tau||p)$ est pris en compte dans le théorème 5.1. Mais dans ce théorème nous avons aussi en complément un terme polynomial $\frac{(1-p)\sqrt{\tau}}{(\tau-p)\sqrt{2\pi N(1-\tau)}}$ qui n'est pas négligeable. Prendre en compte seulement le terme exponentiel comme dans l'équation (5.14) est trop grossier dans certains cas comme le montre le calcul de la complexité en données faite avec ces approximations pour différentes cryptanalyses dans le tableau 5.3.

Le comportement polynomial des queues de la loi binomiale

Dans [BJV04], un terme polynomial est pris en considération. Cependant ce terme polynomial est seulement bon pour les tranches de paramètres où une approximation

6. C'est le cas par exemple de la cryptanalyse différentielle impossible ou de la cryptanalyse différentielle d'ordre supérieur.

gaussienne des queues de la loi binomiale peut être utilisée. Dans ce cas le nombre d'échantillons est :

$$N \approx \frac{2 \cdot \Phi^{-1}(\frac{\alpha+\beta}{2})^2}{D\left(p_* || p\right)}, \tag{5.15}$$

où Φ^{-1} est l'inverse de la fonction de réparation d'une variable aléatoire qui suit une loi gaussienne. Par exemple cette formule donne une estimation plutôt mauvaise dans le cas de la cryptanalyse différentielle. En général cette formule est trop optimiste comme le montrent les résultats expérimentaux donnés dans le tableau 5.3.

		$\log_2(N)$	(5.6) & (5.7)	[BJV04]	[BV08]
Linéaire	$p_* = 0.5 + 1.49 \cdot 2^{-24}$ $p = 0.5$ $b_{\text{nd}} = 0.1$ $b_{\text{fa}} = 0.1$	47.57	47.88	47.57	49.58
Linéaire	$p_* = 0.5 + 1.49 \cdot 2^{-24}$ $p = 0.5$ $b_{\text{nd}} = 0.001$ $b_{\text{fa}} = 0.001$	50.10	50.13	50.10	51.17
Différentielle	$p_* = 1.87 \cdot 2^{-56}$ $p = 2^{-64}$ $b_{\text{nd}} = 0.1$ $b_{\text{fa}} = 0.1$	56.30	56.77	54.44	57.71
Différentielle	$p_* = 1.87 \cdot 2^{-56}$ $p = 2^{-64}$ $b_{\text{nd}} = 0.001$ $b_{\text{fa}} = 0.001$	58.30	58.50	56.98	59.29
Différentielle tronquée	$p_* = 1.18 \cdot 2^{-16}$ $p = 2^{-16}$ $b_{\text{nd}} = 0.001$ $b_{\text{fa}} = 0.001$	26.32	26.35	26.28	27.39

TABLE 5.3 – Comparaison des estimations de $\log_2(N)$ en utilisant l'algorithme 13 avec la loi binomiale (valeur exacte), les estimations (5.6) & (5.7) et les estimations de [BJV04, BV08]

.

5.3.4 Travaux relatifs dans le cas de la cryptanalyse différentielle

Pour la cryptanalyse différentielle on ne peut pas utiliser l'approximation gaussienne de la loi binomiale. Dans [BS90], la loi binomiale est approchée par une loi de Poisson. Le détail du calcul de la complexité en données est présenté dans la thèse d'Henri Gilbert [Gil97]. Nous donnons ici le détail de ce calcul.
Pour la cryptanalyse différentielle il est supposé que p_* est suffisant loin de p. Ainsi à

partir du moment où pour une clé fixée, la différence a_r après avoir déchiffré le dernier tour apparaît au moins une fois alors cette clé est un candidat potentiel. Ainsi dans la cryptanalyse différentielle, il est classique de fixer le seuil T à 1. Dans ce cas précis on ne peut donc pas vraiment contrôler la probabilité de fausse alarme on va donc s'intéresser au calcul de la complexité en données pour une probabilité de non-détection très faible de l'ordre de $0,01$ (c'est-à-dire une probabilité de succès proche de 99%). Si on écrit la formule de la probabilité de non-détection en utilisant l'approximation binomiale on a

$$G_{\text{nd}} = \sum_{i=0}^{0} e^{-Np_*} \frac{(Np_*)^i}{i!} = e^{-Np_*}.$$

Pour une probabilité de succès supérieure à 99%, on obtient donc une complexité en données :

$$N \approx \frac{4.6}{p_*}.$$

5.3.5 Le comportement asymptotique de la complexité en données

Le but ici est de trouver un critère simple pour comparer la puissance de deux attaques statistiques. Les paramètres de l'attaque que nous allons utiliser afin de calculer la complexité en données sont les probabilités p_* et p :

- p_* : probabilité que le phénomène observé arrive pour la bonne clé k^*.
- p : probabilité que le phénomène apparaisse pour une autre clé $k \neq k^*$.

Comme détaillé dans la section 5.3.2, ce calcul consiste à résoudre un système d'inéquations en N, τ. Au vu de sa complexité, ce système est difficile à résoudre. Nous avons donc décidé de fixer le seuil relatif τ afin de trouver une formule asymptotique de la complexité en données.

Une première approximation

Comme nous avons dit dans la section 5.3.1, la probabilité de fausse alarme joue un rôle important dans le calcul de la complexité en temps d'une attaque statistique. Sous l'hypothèse que la complexité en données ne varie pas beaucoup en fonction de la probabilité de fausse alarme, nous fixons le seuil relatif à p_*. Cette hypothèse nous donne une probabilité de non-détection de l'ordre de $1/2$. Nous pouvons alors utiliser la formule (5.7) donnée dans le théorème 5.1 pour obtenir une approximation relativement précise du nombre d'échantillons N nécessaire à une cryptanalyse statistique simple.

Théorème 5.2. *Soit p_* (resp. p) la probabilité que le phénomène observé arrive pour la bonne sous clé k^* (resp. pour les mauvaises sous clés). Pour un seuil relatif τ fixé à p_*, une bonne approximation du nombre d'échantillons N nécessaire pour distinguer les deux distributions des compteurs avec une probabilité de fausse alarme plus petite ou égale à b_{fa} est :*

$$N' \stackrel{\text{def}}{=} -\frac{1}{D\left(p_* || p\right)} \left[\ln\left(\frac{\nu b_{fa}}{\sqrt{D\left(p_* || p\right)}} \right) + 0.5 \ln\left(-\ln(\nu b_{fa}) \right) \right], \tag{5.16}$$

pour

$$\nu \stackrel{\text{def}}{=} \frac{(p_* - p)\sqrt{2\pi(1-p_*)}}{(1-p)\sqrt{p_*}} \quad et \quad \theta \stackrel{\text{def}}{=} \left[1 + \frac{1}{2\ln(\nu b_{fa})} \ln\left(-\frac{\ln(\nu b_{fa})}{D\left(p_*||p\right)}\right)\right]^{-1}. \tag{5.17}$$

L'erreur de cette approximation est encadrée par :

$$N' \leq N_\infty \leq N' \left[1 + \frac{(\theta - 1)\ln(\theta)}{\ln(N')}\right],$$

où N_∞ est la valeur obtenue par l'algorithme 13 en utilisant les approximations des probabilités d'erreurs données par les équations (5.6) et (5.7).

Preuve : Le seuil relatif τ est fixé à p_*. De cette façon on sait que la probabilité de non-détection est proche de $\frac{1}{2}$. Nous voulons contrôler la probabilité de fausse alarme b_{fa}. L'équation (5.6) nous donne

$$N \approx -\frac{\ln(\nu b_{\text{fa}}\sqrt{N})}{D\left(p_*||p\right)} \quad \text{où} \quad \nu \stackrel{\text{def}}{=} \frac{(p_* - p)\sqrt{2\pi(1-p_*)}}{(1-p)\sqrt{p_*}}. \tag{5.18}$$

La formule (5.18) nous suggère d'utiliser la fonction contractante f suivante :

$$f(x) \stackrel{\text{def}}{=} -\frac{\ln(\nu b_{\text{fa}}\sqrt{x})}{D\left(p_*||p\right)}.$$

Nous appliquons cette fonction itérativement en commençant par le terme $N_0 = \dfrac{\ln(\nu b_{\text{fa}})}{D\left(p_*||p\right)}$. Cela nous donne une séquence $(N_i)_{i\geq 0}$ avec $N_{i+1} = f(N_i)$. La limite de cette fonction N_∞ peut être vue comme le nombre d'échantillons nécessaires à la cryptanalyse. Puisque f est décroissante, la suite $(N_i)_{i\geq 0}$ forme une suite alternée. Les termes consécutifs de cette suite satisfont : $N_{2i-1} \leq N_\infty \leq N_{2i}$. La fonction f peut se récrire

$$f(x) = a - b\ln(x) \text{ avec } a \stackrel{\text{def}}{=} -\frac{\ln(\nu b_{\text{fa}})}{D\left(p_*||p\right)} \text{ et } b \stackrel{\text{def}}{=} \frac{1}{2D\left(p_*||p\right)}.$$

Notre choix du premier terme N_0 se justifie par le fait qu'il est égal à a. Nous voulons maintenant montrer que le second terme N_1 nous donne une bonne approximation de N_∞.

$$N_1 = f(N_0) = a - b\ln(N_1) - \frac{1}{D\left(p_*||p\right)}\left[\ln\left(\frac{\nu b_{\text{fa}}}{\sqrt{D\left(p_*||p\right)}}\right) + 0.5\ln\left(-\ln(\nu b_{\text{fa}})\right)\right],$$

Comme $N_1 \leq N_\infty \leq N_2$, on va exprimer N_2 en fonction de N_1

$$\begin{aligned} N_2 &= f(N_1) \\ &= N_0 - b\ln(N_0) + b\ln\left(N_0/N_1\right) \\ &= N_1 + b\ln\left(N_0/N_1\right). \end{aligned}$$

Soit θ comme défini dans le théorème :

$$\theta \stackrel{\text{def}}{=} \left[1 + \frac{1}{2\ln(\nu b_{\text{fa}})} \ln\left(-\frac{\ln(\nu b_{\text{fa}})}{D\left(p_*||p\right)}\right)\right]^{-1}.$$

Alors

$$\begin{aligned}\frac{N_1}{N_0} &= 1+\frac{b\ln(a)}{a} = 1+\frac{\ln(a)}{2\ln(\nu b_{\text{fa}})} \\ &= \left[1+\frac{1}{2\ln(\nu b_{\text{fa}})}\ln\left(-\frac{\ln(\nu b_{\text{fa}})}{D\left(p_*||p\right)}\right)\right] = \theta^{-1}.\end{aligned}$$

L'encadrement de N_∞ devient :

$$N_1 \leq N_\infty \leq N_1\left[1+\frac{b\ln(\theta)}{N_1}\right].$$

Dans le but de prouver que N_1 est une bonne approximation de N_∞, nous nous concentrons sur $b\ln(\theta)/N_1$ et nous le comparons à 1. Puisque $N_1/b = a/b - \ln(a)$, nous allons chercher une borne pour a/b. Nous avons $\theta N_1 = N_0$, ce qui implique que $a/b = \theta\ln(a)/(\theta-1)$. Puisque f est une fonction décroissante, $N_0 > N_1$ nous donne $N_1/b \geq \ln(N_2)/(\theta-1)$. Pour conclure nous avons $N_2 \leq N_1\left[1+\frac{(\theta-1)\ln(\theta)}{\ln(N_1)}\right]$ et

$$N_1 \leq N_\infty \leq N_1\left[1+\frac{(\theta-1)\ln(\theta)}{\ln(N_1)}\right]$$

où N_1 est égal à la valeur N' donnée dans le théorème. □
Cette approximation du nombre d'échantillons nécessaire est assez précise : en effet nous avons estimé la complexité en données de certaines attaques connues (voir tableau 5.4) et observé que θ est toujours compris entre 1 et 7. De plus, pour $b_{\text{fa}} = 2^{-32}$, les valeurs observées de θ sont plus petites que 2. Dans le tableau 5.4 nous avons comparé la valeur de N' avec la vraie valeur de N. Ces expérimentations montrent que N' est proche de la vraie valeur de N et valident expérimentalement le fait que θ est compris entre 1 et 7.

Le comportement asymptotique

La formule donnée dans le théorème 5.2, pour calculer le nombre d'échantillons nécessaire à une cryptanalyse reste cependant assez compliquée. À partir de cette formule nous déduisons une formule plus simple qui nous donne le comportement asymptotique d'une cryptanalyse statistique simple.

Lemme 5.6. *En utilisant les notations du théorème 5.2,* $\ln(2\sqrt{\pi D\left(p_*||p\right)})$ *est une bonne approximation de* $\ln(\nu)$ *où* ν *est donné par (5.18).*

Preuve : Ceci se fait facilement à partir du développement limité de $D\left(p_*||p\right)$. □

À partir de ce lemme, on déduit donc une bonne approximation de N'.

Lemme 5.7. *En utilisant les notations du théorème 5.2, une bonne approximation de* N' *est :*

$$N'' \stackrel{\text{def}}{=} -\frac{\ln(2\sqrt{\pi}b_{fa})}{D\left(p_*||p\right)}. \tag{5.19}$$

Preuve : Dans l'expression de N' donnée dans le théorème 5.2 il est facile de voir que le terme $0.5 \ln(-\ln(\nu b_{\text{fa}}))$ est négligeable par rapport au terme $\ln\left(\frac{\nu b_{\text{fa}}}{\sqrt{D(p_*||p)}}\right)$. Ainsi une bonne approximation de N' devient :

$$-\frac{1}{D(p_*||p)}\left[\ln(\nu) - 1/2\ln(D(p_*||p)) + \ln(b_{\text{fa}})\right].$$

De plus, par le lemme 5.6, nous avons qu'une bonne approximation de $\ln(\nu)$ est $\ln(2\sqrt{\pi D(p_*||p)})$. Ainsi on obtient qu'une bonne approximation de N' est donnée par :

$$-\frac{1}{D(p_*||p)}\left[\ln(2\sqrt{\pi}) + \ln(b_{\text{fa}})\right].$$

□

N'' est essentiellement une fonction décroissante en fonction de la divergence de Kullback-Leibler entre p_* et p. On peut donc en conclure que comparer la complexité en données de deux attaques statistiques peut se résumer à la comparaison entre les divergences de Kullback-Leibler correspondantes.

5.3.6 Les résultats expérimentaux

Nous avons fait un certain nombre d'expérimentations dans le but de prouver que l'estimation N' donnant le nombre d'échantillons nécessaire à l'attaque est plutôt bonne et devient encore meilleure quand la probabilité de fausse alarme tend vers 0 (voir tableau 5.4).

La valeur N' est une approximation. Pour avoir une valeur plus précise de la complexité en données d'une attaque statistique et afin de pouvoir la comparer avec la complexité en données d'autres attaques, l'algorithme 13 peut être utilisé.

Rappelons que la quantité N' donne le nombre d'échantillons et non le nombre de messages clairs nécessaire à la cryptanalyse. Dans le cas de la cryptanalyse linéaire ces deux quantités sont égales. Mais dans le cas de la cryptanalyse différentielle le nombre de messages clairs est égal au double du nombre d'échantillons. À partir du nombre d'échantillons il est assez facile d'en déduire la complexité en données en multipliant cette quantité par un facteur dépendant du type de cryptanalyse.

Les résultats donnés dans le tableau 5.4 montrent que N'' donne une estimation moins précise de la complexité en données d'une attaque statistique. Néanmoins cette estimation reflète bien le comportement asymptotique de la complexité en données d'une attaque statistique (c'est-à-dire en $1/D(p_*||p)$).

5.3.7 Comportement asymptotique pour certaines cryptanalyses statistiques simples

Nous avons vu dans le lemme 5.7 que le nombre d'échantillons d'une cryptanalyse statistique simple dépend essentiellement de la divergence de Kullback-Leibler. Elle est

$b_{\text{fa}} = 2^{-8}$

	p	p_*	$\log_2(N)$	$\log_2(N')$	$\log_2(N'')$	θ
L	0.5	$0.5 + 1.19 \cdot 2^{-21}$	42.32	42.00 (-0.32)	42.60	6.48
DL	0.5	$0.5 + 1.73 \cdot 2^{-6}$	11.26	11.15 (-0.11)	11.52	2.28
D	2^{-64}	$1.87 \cdot 2^{-56}$	54.57	54.68 ($+0.11$)	54.82	6.14
D'	2^{-32}	$1.53 \cdot 2^{-27}$	27.14	26.80 (-0.34)	26.94	3.56
TD	2^{-16}	$1.18 \cdot 2^{-16}$	23.85	23.66 (-0.19)	24.13	3.87

$b_{\text{fa}} = 2^{-16}$

	p	p_*	$\log_2(N)$	$\log_2(N')$	$\log_2(N'')$	θ
L	0.5	$0.5 + 1.19 \cdot 2^{-21}$	43.62	43.54 (-0.08)	43.79	2.89
DL	0.5	$0.5 + 1.73 \cdot 2^{-6}$	12.54	12.52 (-0.02)	12.71	1.53
D	2^{-64}	$1.87 \cdot 2^{-56}$	55.85	55.94 ($+0.09$)	56.02	3.14
D'	2^{-32}	$1.53 \cdot 2^{-27}$	28.27	28.05 (-0.22)	28.14	2.09
TD	2^{-16}	$1.18 \cdot 2^{-16}$	25.15	25.11 (-0.04)	25.33	2.07

$b_{\text{fa}} = 2^{-32}$

	p	p_*	$\log_2(N)$	$\log_2(N')$	$\log_2(N'')$	θ
L	0.5	$0.5 + 1.19 \cdot 2^{-21}$	44.78	44.76 (-0.02)	44.88	1.42
DL	0.5	$0.5 + 1.73 \cdot 2^{-6}$	13.70	13.69 (-0.01)	13.80	1.25
D	2^{-64}	$1.87 \cdot 2^{-56}$	56.98	57.06 ($+0.08$)	57.11	2.00
D'	2^{-32}	$1.53 \cdot 2^{-27}$	29.13	29.17 ($+0.04$)	29.23	1.51
TD	2^{-16}	$1.18 \cdot 2^{-16}$	26.31	26.30 (-0.01)	26.42	1.48

TABLE 5.4 – Comparaison entre les deux estimations N' et N'' et la vraie valeur de la complexité en données pour différentes valeurs de β, p et p_* avec une probabilité de non détection proche de 0.5. La colonne de θ est mise ici pour illustrer le fait que dans le théorème 5.2 on a $1 \leq \theta \leq 7$.

– **L** : Attaque linéaire sur le DES retrouvant 26 bits de clés [Mat94].
– **DL** : Attaque différentielle-linéaire sur le DES [LH94].
– **D** : Attaque différentielle du DES [BS93].
– **D'/TD** :Autres attaques pour des paramètres d'ordre de grandeur d'une cryptanalyse différentielle/différentielle tronquée.

dominée par $D(p_*||p)^{-1}$. Dans cette section nous allons estimer la quantité $D(p_*||p)^{-1}$ pour en extraire le comportement asymptotique de la complexité en données de plusieurs attaques statistiques. Nous allons voir que nous retrouvons alors les résultats connus de la complexité en données pour certaines cryptanalyses comme la cryptanalyse linéaire. Le tableau 5.5 donne le comportement asymptotique de la complexité en données dans le cas des cryptanalyses linéaire, différentielle-linéaire, différentielle, différentielle tronquée, différentielle d'ordre supérieur, différentielle impossible. Nous présentons ici le détail des calculs.

Expansion de Taylor de la divergence de Kullback-Leibler

Pour extraire les résultats dont nous avons besoin, nous introduisons quelques calculs intermédiaires qui sont essentiellement des développement limités de la divergence de Kullback-Leibler. Dans un premier temps, rappelons la définition de cette quantité :

$$D(p_*||p) = p_* \ln\left(\frac{p_*}{p}\right) + (1-p_*)\ln\left(\frac{1-p_*}{1-p}\right).$$

Une première expansion de cette divergence nous donne le résultat suivant :

Lemme 5.8. *Soit $0 < a < b < 1$ tel que $\mathcal{O}\left(\frac{b-a}{1-a}\right) = \mathcal{O}(b-a)$. Alors,*

$$D(b||a) = b\left[\ln\left(\frac{b}{a}\right) - \frac{b-a}{b} + \frac{(b-a)^2}{2b(1-b)} + \frac{(a-b)^3}{3b(1-b)^2}\right] + \mathcal{O}\left((b-a)^4\right)$$

Preuve : En utilisant un développement limité, nous avons :

$$\begin{aligned}(1-b)\ln\left(\frac{1-b}{1-a}\right) &= -(1-b)\ln\left(1+\frac{b-a}{1-b}\right)\\ &= a-b+\frac{(a-b)^2}{2(1-b)} + \frac{(a-b)^3}{3(1-b)} + \mathcal{O}\left((b-a)^4\right).\end{aligned}$$

En conséquence,

$$\begin{aligned}D(b||a) &= b\ln\left(\frac{b}{a}\right) + (1-b)\ln\left(\frac{1-b}{1-a}\right)\\ &= b\ln\left(\frac{b}{a}\right) + a - b + \frac{(a-b)^2}{2(1-b)} + \frac{(a-b)^3}{3(1-b)^2} + \mathcal{O}\left((a-b)^4\right).\end{aligned}$$

□

Lemme 5.9. *Soit $\varepsilon > 0$ un nombre réel tel que $\mathcal{O}\left(\frac{\varepsilon}{a}\right) = \mathcal{O}\left(\frac{\varepsilon}{1-a}\right) = \mathcal{O}(\varepsilon)$. Alors,*

$$D(a+\varepsilon||a) = \frac{\varepsilon^2}{2a(1-a)} + \frac{\varepsilon^3(1-2a)}{3a^2(1-a)^2} + \mathcal{O}\left(\varepsilon^4\right).$$

Preuve : En utilisant le lemme 5.8, nous avons

$$D(a+\varepsilon||a) = (a+\varepsilon)\left[\ln\left(1+\frac{\varepsilon}{a}\right) - \frac{\varepsilon}{a+\varepsilon} + \frac{\varepsilon^2}{2(a+\varepsilon)(1-a-\varepsilon)} - \frac{\varepsilon^3}{3(a+\varepsilon)(1-a-\varepsilon)^2}\right] + \mathcal{O}\left(\varepsilon^4\right).$$

Puisque $\varepsilon/a = \mathcal{O}(\varepsilon)$, nous pouvons faire le développement limité du logarithme pour avoir :

$$\begin{aligned} D(a+\varepsilon||a) &= (a+\varepsilon)\left[\frac{\varepsilon}{a} - \frac{\varepsilon^2}{2a^2} + \frac{\varepsilon^3}{3a^3} - \frac{\varepsilon}{a+\varepsilon} + \frac{\varepsilon^2}{2(a+\varepsilon)(1-a-\varepsilon)}\right. \\ &\quad \left. - \frac{\varepsilon^3}{3(a+\varepsilon)(1-a-\varepsilon)^2} + \mathcal{O}\left(\frac{\varepsilon^4}{a^4}\right)\right] + \mathcal{O}\left(\varepsilon^4\right) \\ &= (a+\varepsilon)\left[\frac{\varepsilon^2\, a}{2a^2(a+\varepsilon)(1-a-\varepsilon)} + \frac{\varepsilon^3(1-2a)}{3a^2(1-a)^2(a+\varepsilon)} + \mathcal{O}\left(\frac{\varepsilon^4}{a^4}\right)\right] + \mathcal{O}\left(\varepsilon^4\right) \\ &= \frac{\varepsilon^2}{2a(1-a)} + \frac{\varepsilon^3(1-2a)}{3a^2(1-a)^2} + \mathcal{O}\left(\varepsilon^4\right). \end{aligned}$$

□

La cryptanalyse linéaire

Nous rappelons que dans le cas de la cryptanalyse linéaire les variables aléatoires étudiées suivent des lois binomiales de paramètres N et $p = 1/2$ dans le cas d'une mauvaise sous clé ou de paramètres N et $p_* = p + \varepsilon$ pour le bon candidat (ε, le biais, est petit). Un résultat bien connu dû à Matsui [Mat93] est que la complexité en données d'une attaque linéaire est de l'ordre de ε^{-2}. Or

Lemme 5.10. *Pour un jeu de paramètres donné par la cryptanalyse linéaire nous avons*

$$D(p+\varepsilon||p) = 2\varepsilon^2 + \mathcal{O}\left(\varepsilon^3\right).$$

Preuve : Cela découle directement du lemme 5.9 avec $a = p = \frac{1}{2}$. □

Ainsi nous retrouvons bien le résultat connu sur le comportement asymptotique de la complexité en données d'une cryptanalyse linéaire.

La cryptanalyse différentielle

Dans le cas de la cryptanalyse différentielle p_* et p sont tous les deux assez petits mais la différence $p_* - p$ est dominée par p_*. Dans le cas de la cryptanalyse différentielle, le rapport p_*/p est grand, et l'on obtient l'expansion suivante de la divergence de Kullback-Leibler :

Lemme 5.11. *Pour un jeu de paramètres donné par la cryptanalyse différentielle nous avons*

$$D(p_*||p) = p_* \ln\left(\frac{p_*}{p}\right) + p_* + \mathcal{O}\left(p_*^2\right).$$

Preuve : Cela découle directement du lemme 5.8. Comme le rapport p_*/p peut être grand, la première partie en $\ln\left(\frac{p_*}{p}\right)$ ne peut pas être simplifiée. □

Ainsi dans le cas de la cryptanalyse différentielle le comportement asymptotique du nombre d'échantillons est

$$\frac{1}{p_*\left[\ln(p_*/p) + 1\right]}.$$

La valeur asymptotique de la complexité en données que l'on obtient est différente de la complexité en données asymptotique donnée par Eli Biham et Adi Shamir [BS91] qui est $\frac{1}{p_*}$. Cette nouvelle formule prend en compte le rapport p_*/p. Ce qui est logique, car intuitivement, plus ce rapport est grand plus la complexité en données est petite.

La cryptanalyse différentielle linéaire

Cette attaque présentée dans la section 3.3 possède le même type de paramètres que pour la cryptanalyse linéaire (c'est-à-dire $p = 1/2$ et $p_* = p + \varepsilon$). Ainsi un nombre d'échantillons nécessaire à l'attaque est le même que pour la cryptanalyse linéaire. Comme dans le cas de la cryptanalyse différentielle un échantillon est composé de deux messages.

La cryptanalyse différentielle tronquée

Il existe plusieurs types de cryptanalyses différentielles tronquées. Dans certaines cryptanalyses différentielles tronquées les paramètres p_* et p sont du même ordre de grandeur que pour le cas de la cryptanalyse différentielle classique. Ce cas ne nous intéresse pas ici puisque le comportement asymptotique du nombre d'échantillons sera alors le même que celui de la cryptanalyse différentielle. Le type de cryptanalyse différentielle tronquée que nous étudions ici correspond au cas où p_* et p sont petits et $p_* = p + \varepsilon$ où ε est petit. Dans ce cas on a :

Lemme 5.12. *Pour un jeu de paramètre donné par la cryptanalyse différentielle tronquée nous avons*

$$D\left(p_* || p\right) \approx \frac{\varepsilon^2}{2p_*}.$$

Preuve : Cela découle directement du lemme 5.9. □

La cryptanalyse différentielle impossible

Ce cas est un peu particulier. En effet dans notre analyse nous avons toujours supposé que p_* est plus grand que p. Or ici $p_* = 0$. En réalité le cas $p_* \leq p$ a été traité par [Tez10]. Dans cet article, il est montré que la preuve de la formule de la complexité en données peut aussi être faite dans le cas où $p_* < p$. Par convention

$$D\left(0 || p\right) = -\ln(1-p) = p + \mathcal{O}\left(p^2\right).$$

Ainsi le nombre d'échantillons pour une cryptanalyse différentielle impossible est $\dfrac{1}{p}$.

La cryptanalyse différentielle d'ordre supérieur

Dans cette attaque, présentée dans la section 2.4, la probabilité p_* est égale à 1. Et

$$D\left(1 || p\right) = \ln(\frac{1}{p}) \geq 1.$$

En fait dans les attaques différentielles d'ordre v, un seul échantillon est souvent suffisant à une cryptanalyse mais un échantillon est composé de 2^v messages clairs donc la complexité en données est de 2^v.

Attaque	Comportement asymptotique du nombre d'échantillons	Comportement asymptotique du nombre messages clairs
Linéaire	$\frac{1}{2(p_* - p)^2}$	$\frac{1}{2(p_* - p)^2}$
Différentielle	$\frac{1}{p_* \ln(p_*/p) + p_*}$	$\frac{2}{p_* \ln(p_*/p) + p_*}$
Différentielle-linéaire	$\frac{1}{2(p_* - p)^2}$	$\frac{1}{(p_* - p)^2}$
Différentielle tronquée	$\frac{p}{(p_* - p)^2}$	$\frac{p \cdot \gamma}{(p_* - p)^2}$, $1 < \gamma < 2$
Différentielle impossible	$\frac{1}{p}$	$\frac{2}{p}$
Différentielle d'ordre v	$-\frac{1}{\ln p}$	$-\frac{2^v}{\ln p}$

TABLE 5.5 – Comportement asymptotique de la complexité en données de certaines attaques statistiques.

5.4 Probabilité de succès

Pour le calcul de la complexité en données d'une attaque statistique simple, nous avons eu besoin de fixer la probabilité de succès $(1 - \alpha)$ afin d'en déduire une formule asymptotique simple. Dans cette section, nous allons utiliser une autre méthode pour calculer la probabilité de succès d'une attaque pour une complexité données. Dans l'approche faite dans la section 5.3, en fixant la probabilité de fausse alarme nous ne savons pas exactement quelle est la taille de la liste des clés gardées. Dans l'approche que nous allons utiliser ici nous allons fixer la taille de cette liste.

5.4.1 Les statistiques d'ordre

Dans le modèle *"taille de liste fixée"*, la problématique n'est pas de décider si une clé candidate est probable ou pas ou pas mais de distinguer les clés candidates les plus probables parmi l'ensemble des clés candidates. Soit 2^n le nombre total de toutes les clés candidates possibles : la bonne sous clé k^* plus les $2^n - 1$ sous clés incorrectes que nous notons $k_1, \ldots, k_{2^n-1}$. La liste $\mathcal{L}$ des clés candidates les plus probables est de taille fixée ℓ. Dans cette liste sont gardées ℓ clés candidates les plus probables (la bonne sous clé peut y être ou non). La cryptanalyse réussit si la bonne sous clé fait partie de la liste des clés gardées.

Définition 5.5. *La* **probabilité de succès** *d'une attaque statistique simple est égale à la probabilité que la sous clé k^* fasse partie de la liste des clés gardées.*

$$P_S \stackrel{\text{def}}{=} P\left[k^* \in \mathcal{L}\right].$$

L'étude que nous présentons ici nous conduit à une formule simple qui est une bonne estimation de la probabilité de succès. Cette formule est une fonction du nombre d'échantillons N nécessaire à la cryptanalyse, du nombre total des clés regardées 2^n et de la taille de la liste ℓ des clés gardées. Cette section repose sur des arguments venant de la théorie des statistiques d'ordre que nous présentons ici.

Notation 5.1. *Les variables aléatoires correspondant aux compteurs C_{k_i} sont notées par $(\psi_i)_{0\leq i<2^n-1}$. La phase d'analyse consiste à trier les clés et à garder les ℓ candidats les plus probables.*

Nous définissons donc les variables aléatoires triées : la $i$ème plus grande valeur des variables ψ_i est notée : Ψ_i.

ψ_0 est la variable aléatoire correspondant à la clé k^*.

Dans le modèle des statistiques d'ordre nous nous intéressons à la distribution de Ψ_ℓ puisque si nous gardons une liste de taille ℓ la bonne clé est dans la liste si et seulement si $\psi_0 \geq \Psi_\ell$. La probabilité de succès de l'attaque est alors

$$P_S = P\left[\Psi_\ell \leq \psi_0\right] = \sum_{i=0}^{N} P\left[\psi_0 = i\right] \cdot P\left[\Psi_\ell \leq i\right].$$

Soit G la fonction de répartition des variables $\psi_i (i \neq 0)$:

$$G(x) = P\left[\psi_1 \leq x\right] = \cdots = P\left[\psi_{N-1} \leq x\right].$$

Il est bien connu (voir [DN03]) que $G(\Psi_\ell)$ suit une loi bêta avec paramètres $N-\ell-1$ et $\ell-1$. Nous notons par h la fonction densité de la loi bêta et par g_* la fonction $g_*(x) = P\left[\psi_0 = \lfloor x \rfloor\right]$. Nous pouvons alors écrire,

$$\begin{aligned} P_S &= \sum_{i=0}^{N} g_*(i) \cdot P\left[\Psi_\ell < i\right] \\ &= \sum_{i=0}^{N} g_*(i) \cdot P\left[G(\Psi_\ell) < G(i)\right] \\ &= \sum_{i=0}^{N} g_*(i) \cdot \int_0^{G(i)} h(t)\, dt. \end{aligned} \tag{5.20}$$

A l'aide des définitions présentées dans cette section nous avons trouvé une formule simple pour calculer la probabilité de succès d'une attaque statistique.

5.4.2 La formule de la probabilité de succès

En 2008, Selçuk dans [Sel08], a utilisé une approximation gaussienne de la loi binomiale pour exhiber une formule pour la probabilité de succès d'une attaque statistique. Il s'est en particulier intéressé au cas de la cryptanalyse linéaire et de la cryptanalyse différentielle. L'approximation gaussienne étant bonne dans le cas de la cryptanalyse linéaire, son approximation de la probabilité de succès est plutôt bonne dans ce cas. En revanche, dans le cas de la cryptanalyse différentielle, comme l'approximation normale de la loi binomiale n'est pas bonne, la formule exhibée pour la probabilité de succès n'est pas très bonne comme Ali Aydin Selçuk le dit lui même. Ici nous exhibons une formule générale de la probabilité de succès d'une cryptanalyse statistique, laquelle ne dépend pas d'une approximation de la loi binomiale. Nous utilisons une approximation seulement pour le calcul du terme d'erreur. Pour trouver sa formule de probabilité de succès, Selçuk suppose que la distribution de la ℓème statistique d'ordre (ℓ est la taille de la liste des clés gardées) tend vers une loi normale. Dans notre analyse, nous utilisons directement le fait que la ℓ-ème statistique d'ordre suit une loi bêta.

Le résultat que nous avons trouvé est basé sur le fait que la loi bêta est concentrée autour du point

$$t_0 \stackrel{\text{def}}{=} \frac{2^n - \ell - 1}{2^n - 2}.$$

Nous avons besoin de définir quelques notions avant de donner le résultat principal de cette section.

Définition 5.6. *Soit G la fonction de répartition de la loi binomiale avec paramètre (N, p). Cette fonction est définie par la formule suivante*

$$G(x) \stackrel{\text{def}}{=} \sum_{i \leq x} \binom{N}{i} p^i (1-p)^{N-i}.$$

Nous définissons l'inverse de cette fonction G^{-1} par

$$G^{-1}(x) = \min\{t | G(t) \geq x\}.$$

Remarque 5.2. *Comme la loi binomiale n'est pas continue il est facile de voir que $G^{-1}(G(x))$ peut être différent de x. Soit g la densité de la loi binomiale de paramètre (N, p). La définition de G^{-1} implique que*

$$\sum_{i=0}^{G^{-1}(x)} g(x) \geq x \quad \text{et} \quad \sum_{i=0}^{G^{-1}(x)-1} g(x) < x.$$

En conséquence, nous pouvons borner le terme d'erreur par

$$G(G^{-1}(x)) - x < g(G^{-1}(x)). \tag{5.21}$$

La preuve du théoreme suivant est donnée dans la section 5.4.3.

Théorème 5.3. *Soit P_S la probabilité de succès d'une attaque statistique qui garde ℓ clés sur un total de 2^n. Soit N le nombre d'échantillons que nous avons à notre disposition.*

La probabilité que le compteur correspondant à la bonne clé soit à la $i$ème place dans la liste des clés gardées est notée par $g_*(i)$:

$$g_*(i) = \binom{N}{i} {p_*}^i (1-p_*)^{N-i}.$$

Soit G la fonction de répartition des compteurs correspondants aux autres clés. Nous notons par G^{-1} l'inverse de cette fonction définie dans la définition 5.6. Soit $t_0 \stackrel{\text{def}}{=} 1 - \frac{\ell-1}{2^n-2}$, le point de concentration de la loi bêta. Et soit

$$B \stackrel{\text{def}}{=} G^{-1}(t_0), \tag{5.22}$$

$$\delta \stackrel{\text{def}}{=} \sum_{i=0}^{B-1} g_*(i), \tag{5.23}$$

$$\theta \stackrel{\text{def}}{=} \frac{p}{p_*} \frac{p_*(N+1) - B}{B - p(N+1)}. \tag{5.24}$$

Si $t_0 \geq \frac{3}{4}$ alors

$$P_S = 1 - \delta + \mathcal{O}\left(\delta(1+\theta)\sqrt{\frac{\ln(\ell/\delta^2)}{\ell}} + \frac{1}{\ell^2} + \frac{1}{n}\right).$$

Les hypothèses faites au début du théorème précédent sont nécessaires afin de prouver le théorème. Pour les cryptanalyses usuelles, ces conditions sont toujours respectées. Le paragraphe suivant explique plus en détail ce que signifie chacune de ces conditions.

Discussion sur les conditions du théorème 5.3

Les valeurs prises par θ : Pour les tranches de paramètres que nous utilisons dans le cas des cryptanalyses statistiques, θ est petit. Il est difficile d'obtenir la vraie valeur de cette constante. Celle-ci dépend de l'ordre de grandeur des paramètres p et p_*. Nous avons calculé l'ordre de grandeur de cette valeur pour les cryptanalyses citées dans les chapitres 2 et 3. Nous avons remarqué que la valeur de θ est la plus grande quand on se place dans le cas de la cryptanalyse linéaire ($p = 1/2$ et $p_* = p + \varepsilon$). Nous détaillons donc rapidement le calcul permettant de nous donner l'ordre de grandeur de cette valeur.

Dans le cas de la cryptanalyse linéaire, l'approximation gaussienne de la loi binomiale est très bonne. Nous allons donc l'utiliser pour obtenir l'ordre de grandeur de θ. Soit Φ la fonction de répartition de la loi normale :

$$\Phi(x) \stackrel{\text{def}}{=} \int_x^\infty \frac{e^{-u^2/2}}{\sqrt{2\pi}} du.$$

En utilisant les notations du théorème 5.3 et par l'équation (5.22) il peut être vérifié que

$$B \approx pN + \Phi^{-1}(\lambda)\sqrt{Np(1-p)}.$$

À partir de la définition de δ donnée dans l'équation (5.23), on a $\Phi^{-1}(\lambda) \underset{\lambda \to 0^+}{\sim} \sqrt{-2\ln(\lambda)}$. De la même façon

$$B \approx p_*N - \Phi^{-1}(\delta)\sqrt{Np_*(1-p_*)}.$$

On a aussi $\Phi^{-1}(\delta) \underset{\lambda \to 0^+}{\sim} \sqrt{-2\ln(\delta)}$. En rassemblant toutes ces remarques nous obtenons

$$\theta \approx \frac{p}{p_*} \frac{y\sqrt{Np_*(1-p_*)}}{x\sqrt{Np(1-p)}} \approx \sqrt{\frac{-\ln(\delta)}{-\ln(\lambda)}}.$$

Pour obtenir cette formule nous avons aussi utilisé le fait que $p_* \approx p$ (ce qui est vrai dans le cas de la cryptanalyse linéaire).

À propos de δ : Nous pouvons remarquer que δ peut être vu comme une approximation de $1 - P_S$ et donc que cette valeur est souvent de l'ordre de 0.05.

À propos de t_0 : Pour ne pas avoir une complexité en temps trop grande nous avons besoin que $1 - t_0 \approx \frac{\ell}{2^n}$ reste petit, par exemple 10^{-5}. Dans le cas où $1 - t_0 = 10^{-5}$ et $\delta = 0.05$ on obtient alors $\theta \approx 0.5$.

Condition non restrictive : Dans le théorème 5.3 nous avons supposé que

$$\frac{\ell - 1}{2^n - 2} \leq \frac{1}{4}. \tag{5.25}$$

Cette inégalité signifie que nous gardons au plus 1/4 des clés. Dans la plupart des attaques statistiques connues on accepte mois d'un quart des clés donc cette condition n'est pas restrictive.

Expression du terme d'erreur dans la formule de la probabilité de succès

Dans le théorème 5.3 nous avons exhibé le terme d'erreur de notre formule par rapport à la vraie valeur de la probabilité de succès. Ce terme d'erreur est égal à

$$P_S - \sum_{i=G^{-1}(1-\frac{\ell-1}{2^n-2})}^{N} g_*(i).$$

Ce terme d'erreur décroît quand 2^n et ℓ tendent vers l'infini mais est aussi décroissant avec δ. Rappelons que $\delta \approx 1 - P_S$, donc, le terme d'erreur induit par notre formule décroît quand la probabilité de succès augmente.

Lien avec la complexité en données

Dans la section 5.3 nous avons utilisé une autre méthode pour calculer la complexité en données d'une attaque statistique. En utilisant des outils venant du modèle *test d'hypothèses* (voir section 5.3.1) nous avons extrait une formule de la complexité en données qui dépend de la probabilité de non-détection α et de la probabilité de fausse alarme β. Cette dernière correspond à la probabilité d'accepter un mauvais candidat dans la liste $\mathcal{L}$ des clés gardées. Dans ce cas il semble naturel de prendre $\beta = \ell/2^n$. D'autre part, α correspond à la probabilité de rejeter la bonne sous-clé et α peut être choisi de telle sorte que $\alpha = 1 - P_S$. Si nous utilisons l'équation (5.10) pour exprimer α en fonction de β, nous obtenons

$$\alpha = \sum_{i=0}^{G^{-1}(1-\beta)-1} g_*(i).$$

En utilisant les valeurs suggérées pour les probabilités d'erreurs α et β, nous obtenons

$$P_S = 1 - \sum_{i=0}^{G^{-1}(1-\ell/n)-1} g_*(i)$$

ce qui correspond au résultat donné par le théorème 5.3.

5.4.3 Preuve de la formule de la probabilité de succès

Le théorème 5.3 est difficile et long à prouver. Pour cette raison nous dédions toute une section à sa preuve. Dans un premier temps, nous donnons une idée de la preuve :

Idée de la preuve du théorème 5.3

L'idée principale consiste à décomposer la somme

$$\sum_{i=0}^{N} g_*(i) \int_0^{G(i)} h(t)\, dt$$

en effectuant un encadrement autour de $G^{-1}(t_0)$ où t_0 est défini par

$$t_0 \stackrel{\text{def}}{=} \frac{2^n - \ell - 1}{2^n - 2}.$$

Soit $\varepsilon > 0$ un réel, nous avons

$$\begin{aligned} P_S &= \sum_{i=0}^{N} g_*(i) \int_0^{G(i)} h(t)\, dt \\ &= \underbrace{\sum_{i=0}^{G^{-1}(t_0-\varepsilon)-1} g_*(i) \int_0^{G(i)} h(t)\, dt}_{A} + \underbrace{\sum_{i=G^{-1}(t_0-\varepsilon)}^{G^{-1}(t_0)-1} g_*(i) \int_0^{G(i)} h(t)\, dt}_{B} \\ &+ \underbrace{\sum_{i=G^{-1}(t_0)}^{N} g_*(i) \int_0^{G(i)} h(t)\, dt}_{C}. \end{aligned} \tag{5.26}$$

Le troisième terme de la somme (5.26) (celui noté C) est :

$$\sum_{i=G^{-1}(t_0)}^{N} g_*(i) \int_0^{G(i)} h(t)\, dt = \sum_{i=G^{-1}(t_0)}^{N} g_*(i) - \sum_{i=G^{-1}(t_0)}^{N} g_*(i) \int_{G(i)}^{1} h(t)\, dt.$$

Au regard de la valeur de C, nous montrons que la probabilité de succès de l'attaque est essentiellement concentrée en

$$\sum_{i=G^{-1}(t_0)}^{N} g_*(i)$$

et que les autres termes de (5.26) sont négligeables.

$$P_S - \sum_{i=G^{-1}(t_0)}^{N} g_*(i) = \underbrace{\sum_{i=0}^{G^{-1}(t_0-\varepsilon)-1} g_*(i) \int_0^{G(i)} h(t)\, dt}_{S_1} + \underbrace{\sum_{i=G^{-1}(t_0-\varepsilon)}^{G^{-1}(t_0)-1} g_*(i) \int_0^{G(i)} h(t)\, dt}_{S_2} - \underbrace{\sum_{i=G^{-1}(t_0)}^{N} g_*(i) \int_{G(i)}^{1} h(t)\, dt}_{S_3}$$

Le premier argument pour prouver que ces termes sont négligeables est que la loi bêta est concentrée autour de t_0. Ce qui signifie que les intégrales avec des domaines suffisamment loin de t_0 sont négligeables. C'est le cas de l'intégrale définie par S_1, mais aussi pour une partie de la somme S_3 que nous notons S_5 .

$$S_3 = \underbrace{\sum_{i=G^{-1}(t_0)}^{G^{-1}(t_0+\varepsilon)-1} g_*(i) \int_{G(i)}^{1} h(t)\, dt}_{S_4} + \underbrace{\sum_{i=G^{-1}(t_0+\varepsilon)}^{N} g_*(i) \int_{G(i)}^{1} h(t)\, dt}_{S_5}.$$

Pour résumer, nous avons maintenant un terme d'erreur $S_1 + S_2 - S_4 - S_5$ avec S_1 et S_5 négligeables grâce aux propriétés de la loi bêta.

Par la suite nous donnons le détail de la preuve qui montre que les termes S_2 et S_4 sont négligeables. Cette preuve qui repose sur deux lemmes est longue et fastidieuse. On peut aisément ignorer cette partie pour lire directement la fin de la preuve du théorème.

Les quantités S_2 et S_4 sont négligeables.

Concentrons nous sur $S_2 - S_4$. On a

$$\begin{aligned} |S_2 - S_4| &\leq \max(S_2, S_4) \\ &\leq \max\left(\sum_{i=G^{-1}(t_0-\varepsilon)}^{G^{-1}(t_0)-1} g_*(i), \sum_{i=G^{-1}(t_0)}^{G^{-1}(t_0+\varepsilon)-1} g_*(i) \right). \end{aligned}$$

L'argument ici est que la somme tend vers 0 et est négligeable par rapport à δ. Les lemmes suivants justifient les arguments donnés précédemment. Avant d'avoir une estimation des queues de la loi bêta nous introduisons un lemme intermédiaire.

Lemme 5.13. *Soit $f(t)$ une fonction définie sur $]0,1[$ qui est 4 fois différentiable. Supposons que cette fonction atteint sa valeur minimale 0 au point $t_0 \in]\frac{1}{2}, 1[$ et que $f''(t_0) > 0$. Soit λ un nombre réel positif. Alors, pour $\varepsilon \in (0, 1 - t_0)$, on a*

$$\int_{t_0}^{t_0+\varepsilon} e^{-\lambda f(t)}\, dt = \int_0^{\phi(t_0+\varepsilon)} \left[\frac{1}{\sqrt{2\tau f''(t_0)}} - \frac{1}{3}\frac{f'''(t_0)}{f''^2(t_0)} + A_{t_0}\sqrt{\tau} + o\left(\sqrt{\tau}\right) \right] e^{-\lambda\tau} d\tau$$

et

$$\int_{t_0-\varepsilon}^{t_0} e^{-\lambda f(t)}\,dt = \int_0^{f(t_0-\varepsilon)} \left[\frac{1}{\sqrt{2\tau f''(t_0)}} + \frac{1}{3}\frac{f'''(t_0)}{f''^2(t_0)} + A_{t_0}\sqrt{\tau} + o\left(\sqrt{\tau}\right)\right] e^{-\lambda\tau} d\tau.$$

où $A_{t_0} \stackrel{\text{def}}{=} \dfrac{\sqrt{2}}{24 f''(t_0)^{5/2}}\left(\dfrac{5f^{(3)}(t_0)^2}{f''(t_0)} - 3f^{(4)}(t_0)\right)$.

Preuve : En remplaçant τ par $f(t)$ dans $\int_{t_0}^{t_0\pm\varepsilon} e^{-\lambda f(t)}dt$ nous obtenons :

$$\int_{t_0}^{t_0\pm\varepsilon} e^{-\lambda f(t)}dt = \int_0^{f(t_0\pm\varepsilon)} I(\tau)e^{-\lambda\tau}d\tau$$

avec $I(\tau) = \left.\dfrac{1}{f'(t)}\right|_{t=f^{-1}(\tau)}$. Dans un premier temps nous exprimons $t - t_0$ comme une fonction de τ en utilisant l'expansion suivante de f.

$$f(t) = \frac{f''(t_0)}{2}(t-t_0)^2 + \frac{f^{(3)}(t_0)}{6}(t-t_0)^3 + \frac{f^{(4)}(t_0)}{24}(t-t_0)^4 + o\left((t-t_0)^4\right).$$

Sans perte de généralité, nous supposons que $t > t_0$ et nous en déduisons donc le comportement asymptotique de $t - t_0$.

$$(t-t_0)^2 = \frac{2f(t)}{f''(t_0)}\left[1 + \frac{1}{3}\frac{f^{(3)}(t_0)}{f''(t_0)}(t-t_0) + \frac{1}{12}\frac{f^{(4)}(t_0)}{f''(t_0)}(t-t_0)^2 + o\left((t-t_0)^2\right)\right]^{-1} \quad (5.27)$$

Ce qui nous donne $t - t_0 = \sqrt{\dfrac{2\tau}{\phi''(t_0)}}\,[1 + \mathcal{O}(\sqrt{\tau})]$.

En remettant cette quantité dans l'équation (5.27) cela nous donne :

$$t - t_0 = \sqrt{\frac{2\tau}{f''(t_0)}}\left[1 - \frac{\sqrt{2}}{6}\frac{f^{(3)}(t_0)}{f''(t_0)^{3/2}}\sqrt{\tau} + o(\sqrt{\tau})\right].$$

En allant itérant une nouvelle fois on obtient :

$$\begin{aligned}(t-t_0)^2 &= \frac{2\tau}{f''(t_0)}\left[1 + \frac{\sqrt{2}}{3}\frac{f^{(3)}(t_0)}{f''(t_0)^{3/2}}\left[1 - \frac{\sqrt{2}}{6}\frac{f^{(3)}(t_0)}{f''(t_0)^{3/2}}\sqrt{\tau}\right]\sqrt{\tau} + \frac{1}{6}\frac{f^{(4)}(t_0)}{f''(t_0)^2}\tau + o(\tau)\right]^{-1} \\ t - t_0 &= \sqrt{\frac{2\tau}{f''(t_0)}}\left[1 + \frac{\sqrt{2}}{3}\frac{f^{(3)}(t_0)}{f''(t_0)^{3/2}}\sqrt{\tau} + \left(\frac{1}{6}\frac{f^{(4)}(t_0)}{f''(t_0)^2} - \frac{1}{9}\frac{f^{(3)}(t_0)^2}{f''(t_0)^3}\right)\tau + o(\tau)\right]^{-1/2}.\end{aligned}$$

Nous obtenons finalement :

$$t - t_0 = \sqrt{\frac{2\tau}{f''(t_0)}}\left[1 - \frac{\sqrt{2}}{6}\frac{f^{(3)}(t_0)}{f''(t_0)^{3/2}}\sqrt{\tau} - \left(\frac{1}{12}\frac{f^{(4)}(t_0)}{f''(t_0)^2} - \frac{5}{36}\frac{f^{(3)}(t_0)^2}{f''(t_0)^3}\right)\tau + o(\tau)\right]. \quad (5.28)$$

Nous pouvons utiliser la même méthode pour le cas où $t < t_0$. Nous obtenons alors :

$$t - t_0 = -\sqrt{\frac{2\tau}{f''(t_0)}}\left[1 + \frac{\sqrt{2}}{6}\frac{f^{(3)}(t_0)}{f''(t_0)^2}\sqrt{\tau} - \left(\frac{1}{12}\frac{f^{(4)}(t_0)}{f''(t_0)^2} - \frac{5}{36}\frac{f^{(3)}(t_0)^2}{f''(t_0)^3}\right)\tau + o(\tau)\right]. \quad (5.29)$$

À partir de l'expression de $t - t_0$ en fonction de τ nous pouvons calculer $I(\tau)$. Nous utilisons le développement limité suivant :

$$f'(t) = f''(t_0)(t-t_0) + \frac{f^{(3)}(t_0)}{2}(t-t_0)^2 + \frac{f^{(4)}(t_0)}{6}(t-t_0)^3 + o\left((t-t_0)^3\right).$$

Ce qui nous donne l'expression suivante pour $\frac{1}{f'(t)}$:

$$\begin{aligned}
\frac{1}{f'(t)} &= \frac{1}{f''(t_0)(t-t_0) + \frac{1}{2}f^{(3)}(t_0)(t-t_0)^2 + \frac{1}{6}f^{(4)}(t_0)(t-t_0)^3 + o\left((t-t_0)^3\right)} \\
&= \frac{1}{f''(t_0)(t-t_0)}\left[1 + \frac{1}{2}\frac{f^{(3)}(t_0)}{f''(t_0)}(t-t_0) + \frac{1}{6}\frac{f^{(4)}(t_0)}{f''(t_0)}(t-t_0)^2 + o\left((t-t_0)^2\right)\right]^{-1} \\
&= \frac{1}{f''(t_0)(t-t_0)}\left[1 - \frac{f^{(3)}(t_0)}{2f''(t_0)}(t-t_0) + \left(3\frac{f^{(3)}(t_0)^2}{f''(t_0)} - 2f^{(4)}(t_0)\right)\frac{(t-t_0)^2}{12f''(t_0)} + o\left((t-t_0)^2\right)\right] \\
&= \frac{1}{f''(t_0)(t-t_0)} - \frac{f^{(3)}(t_0)}{2f''(t_0)^2} + \left(3\frac{f^{(3)}(t_0)^2}{f''(t_0)} - 2f^{(4)}(t_0)\right)\frac{t-t_0}{12f''(t_0)^2} + o\left(t-t_0\right). \qquad (5.30)
\end{aligned}$$

Nous allons maintenant remplacer la valeur de $t - t_0$ donnée par l'équation (5.28) dans la formule précédente. Le premier terme de (5.30) s'écrit alors :

$$\begin{aligned}
\frac{1}{f''(t_0)(t-t_0)} &= \frac{1}{\sqrt{2f''(t)\tau}}\left[1 - \frac{\sqrt{2}}{6}\frac{f^{(3)}(t_0)}{f''(t_0)^{3/2}}\sqrt{\tau} - \left(\frac{1}{12}\frac{f^{(4)}(t_0)}{f''(t_0)^2} - \frac{5}{36}\frac{f^{(3)}(t_0)^2}{f''(t_0)^3}\right)\tau + o\left(\tau\right)\right]^{-1} \\
&= \frac{1}{\sqrt{2f''(t)\tau}} + \frac{f^{(3)}(t_0)}{6f''(t_0)^2} + \left(f^{(4)}(t_0) - \frac{f^{(3)}(t_0)^2}{f''(t_0)}\right)\frac{\sqrt{2}}{24f''(t_0)^{5/2}}\sqrt{\tau} + o\left(\sqrt{\tau}\right).
\end{aligned}$$

Le troisième terme de (5.30) devient :

$$\left(3\frac{f^{(3)}(t_0)^2}{f''(t_0)} - 2f^{(4)}(t_0)\right)\frac{t-t_0}{12f''(t_0)^2} = \left(6\frac{f^{(3)}(t_0)^2}{f''(t_0)} - 4f^{(4)}(t_0)\right)\frac{\sqrt{2}}{24f''(t_0)^{5/2}}\sqrt{\tau} + o\left(\tau\right).$$

Comme $I(\tau) = \left.\frac{1}{f'(t)}\right|_{t=f^{-1}(\tau)}$, en rassemblant ces deux équations nous obtenons :

$$I(\tau) = \frac{1}{\sqrt{2f''(t_0)\tau}} - \frac{f^{(3)}(t_0)}{3f''(t_0)^2} + \frac{\sqrt{2}}{24f''(t_0)^{5/2}}\left(5\frac{f^{(3)}(t_0)^2}{f'(t_0)} - 3f^{(4)}(t_0)\right)\sqrt{\tau} + o\left(\tau\right).$$

Dans le cas où $t < t_0$, en utilisant l'équation (5.29), nous obtenons

$$-I(\tau) = \frac{1}{\sqrt{2f''(t_0)\tau}} + \frac{f^{(3)}(t_0)}{3f''(t_0)^2} + \frac{\sqrt{2}}{24f''(t_0)^{5/2}}\left(5\frac{f^{(3)}(t_0)^2}{f''(t_0)} - 3f^{(4)}(t_0)\right)\sqrt{\tau} + o\left(\tau\right).$$

□

Lemme 5.14. *Soit h la fonction de densité de la loi bêta avec paramètres $(2^n-\ell-1, \ell-1)$:*

$$h(t) \stackrel{\text{def}}{=} (2^n-1)\cdot\binom{2^n-2}{\ell-1}\cdot t^{2^n-\ell-1}(1-t)^{\ell-1}.$$

Le maximum de h est atteint au point

$$t_0 \stackrel{\text{def}}{=} \frac{2^n-\ell-1}{2^n-2}.$$

Soit $\varepsilon \stackrel{\text{def}}{=} z\cdot\dfrac{\sqrt{\ell-1}}{2^n-2}$. Si $z = o\left(\sqrt{\ell}\right)$ et $\ell \in [1, 2^n/2]$, nous avons :

$$\int_{t_0-\varepsilon}^{t_0+\varepsilon} h(t)\,dt = 1+\mathcal{O}\left(\frac{1}{\ell^2}+\frac{1}{2^n}+\frac{e^{-z^2/2}}{z}\right).$$

Preuve : Nous appliquons, d'abord, l'approximation de Stirling au coefficient binomial :

$$\binom{2^n-2}{\ell-1} = \sqrt{\frac{1}{2\pi}}\left(\frac{2^n-2}{2^n-\ell-1}\right)^{2^n-\ell-1/2}\left(\frac{2^n-2}{\ell-1}\right)^{\ell-1/2}\left[1-\frac{1}{12(\ell-1)}+\mathcal{O}\left(\frac{1}{2^n}+\frac{1}{\ell^2}\right)\right].$$

Nous simplifions l'expression :

$$\left(\frac{2^n-2}{2^n-\ell-1}\right)^{2^n-\ell-1}\left(\frac{2^n-2}{\ell-1}\right)^{\ell-1} t^{2^n-\ell-1}(1-t)^{\ell-1} \;=\; e^{-(2^n-2)D(t_0||t)}.$$

Cela nous conduit à définir une nouvelle fonction $\tilde{h}$

$$\tilde{h}(t) = C_{2^n,\ell}\cdot e^{-(2^n-2)D(t_0||t)} \quad \text{avec} \quad C_{2^n,\ell} = (2^n-1)\cdot\sqrt{\frac{2^n-2}{2\pi(\ell-1)(2^n-\ell-1)}}.$$

Alors

$$h(t) = \tilde{h}(t)\cdot\left[1-\frac{1}{12(\ell-1)}+\mathcal{O}\left(\frac{1}{2^n}+\frac{1}{\ell^2}\right)\right].$$

La structure de $\tilde{h}$ suggère d'utiliser le lemme 5.13 avec $\lambda = 2^n-2$ et $f(t) = D\,(t_0||t)$. Alors,

$$\begin{aligned}
f''(t_0) = \frac{1}{t_0}+\frac{1}{1-t_0} &= \frac{1}{t_0(1-t_0)} > 0,\\
f^{(3)}(t_0) = \frac{2}{(1-t_0)^2}-\frac{2}{t_0^2} &= 2\frac{2t_0-1}{t_0^2(1-t_0)^2},\\
f^{(4)}(t_0) = \frac{6}{(1-t_0)^3}+\frac{6}{t_0^3} &= 6\frac{3t_0^2-3t_0+1}{t_0^3(1-t_0)^3},\\
\text{et } A_{t_0} = \frac{13t_0^2-13t_0+1}{6\sqrt{2t_0(1-t_0)}}&.
\end{aligned}$$

Puisque $f''(t_0) > 0$ and $f(t_0) = f'(t_0) = 0$, nous pouvons appliquer le lemme 5.13 sous les deux contraintes $z = o\left(\sqrt{\ell}\right)$ et $\ell < 2^n/2$. La première contrainte vient du fait que ε doit

être petit vis à vis de t_0. Cette condition est vérifiée par notre choix final de z. La seconde contrainte vient de la restriction sur t_0 et signifie que nous gardons au plus une clé sur deux ce qui est actuellement le cas pour les cryptanalyses statistiques. Par intégration par parties il est facile de calculer les trois intégrales suivantes. Soit a un nombre réel, nous avons :

- $\int_0^a e^{-t} \cdot t^{-1/2}\, dt = \sqrt{\pi} - e^{-a}a^{-1/2} + \mathcal{O}\left(e^{-a}a^{-3/2}\right)$.
- $\int_0^a e^{-t}\, dt = 1 - e^{-a}$.
- $\int_0^a e^{-t} \cdot t^{1/2}\, dt = \frac{\sqrt{\pi}}{2} - e^{-a}\sqrt{a} + \mathcal{O}\left(e^{-a}a^{-1/2}\right)$.

En appliquant ce résultat au lemme 5.13, nous avons :

$$\begin{aligned}\int_{t_0}^{t_0+\varepsilon} e^{-\lambda f(t)}\, dt &= \sqrt{\frac{\pi}{2\lambda f''(t_0)}} + \mathcal{O}\left(\frac{e^{-\lambda f(t_0+\varepsilon)}}{\lambda\sqrt{f''(t_0)f(t_0+\varepsilon)}}\right)\\ &- \frac{1}{3\lambda}\frac{f^{(3)}(t_0)}{f''^2(t_0)} + \mathcal{O}\left(\frac{1}{\lambda}\frac{f^{(3)}(t_0)}{f''^2(t_0)}e^{-\lambda f(t_0+\varepsilon)}\right)\\ &+ \frac{A_{t_0}}{2\lambda}\sqrt{\frac{\pi}{\lambda}} + \mathcal{O}\left(A_{t_0}\frac{e^{-\lambda f(t_0+\varepsilon)}}{\lambda}\sqrt{f(t_0+\varepsilon)}\right)\end{aligned}$$

et,

$$\begin{aligned}\int_{t_0-\varepsilon}^{t_0} e^{-\lambda f(t)}\, dt &= \sqrt{\frac{\pi}{2\lambda f''(t_0)}} + \mathcal{O}\left(\frac{e^{-\lambda f(t_0-\varepsilon)}}{\lambda\sqrt{f''(t_0)f(t_0-\varepsilon)}}\right)\\ &+ \frac{1}{3\lambda}\frac{f^{(3)}(t_0)}{f''^2(t_0)} + \mathcal{O}\left(\frac{1}{\lambda}\frac{f^{(3)}(t_0)}{f''^2(t_0)}e^{-\lambda f(t_0-\varepsilon)}\right)\\ &+ \frac{A_{t_0}}{2\lambda}\sqrt{\frac{\pi}{\lambda}} + \mathcal{O}\left(A_{t_0}\frac{e^{-\lambda f(t_0-\varepsilon)}}{\lambda}\sqrt{f(t_0-\varepsilon)}\right).\end{aligned}$$

Additionner les deux intégrales nous donne :

$$\begin{aligned}\int_{t_0-\varepsilon}^{t_0+\varepsilon} e^{-\lambda f(t)}\, dt &= \sqrt{\frac{2\pi}{\lambda f''(t_0)}} + \mathcal{O}\left(\frac{e^{-\lambda(f(t_0-\varepsilon)} + e^{-\lambda f(t_0+\varepsilon))}}{\lambda\varepsilon f''(t_0)}\right)\\ &+ \mathcal{O}\left(\frac{1}{3\lambda}\frac{f^{(3)}(t_0)}{f''^2(t_0)}\left(e^{-\lambda f(t_0-\varepsilon)} + e^{-\lambda f(t_0+\varepsilon)}\right)\right)\\ &+ \frac{A_{t_0}}{\lambda}\sqrt{\frac{\pi}{\lambda}} + \mathcal{O}\left(\frac{A_{t_0}\varepsilon}{\lambda}\sqrt{f''(t_0)}\left(e^{-\lambda f(t_0-\varepsilon)} + e^{-\lambda f(t_0+\varepsilon)}\right)\right)\\ &= \sqrt{\frac{2\pi}{\lambda f''(t_0)}}\cdot\left[1 + \sqrt{f''(t_0)}\frac{A_{t_0}}{\lambda}\right]\\ &+ \mathcal{O}\left(\frac{1}{\lambda}\left[e^{-\lambda f(t_0-\varepsilon)} + e^{-\lambda f(t_0+\varepsilon)}\right]\left[\frac{1}{\varepsilon f''(t_0)} + \frac{f^{(3)}(t_0)}{3\phi''^2(t_0)} + A_{t_0}\varepsilon\sqrt{f''(t_0)}\right]\right).\end{aligned}$$

Nous remplaçons maintenant λ et les dérivées de f par leur vraie valeur :

$$\begin{aligned}
\int_{t_0-\varepsilon}^{t_0+\varepsilon} \tilde{h}(t)\,dt &= \int_{t_0-\varepsilon}^{t_0+\varepsilon} C_{2^n,\ell} e^{-(2^n-2)D(t_0||t)}\,dt \\
&= C_{2^n,\ell}\cdot\sqrt{\frac{2\pi t_0(1-t_0)}{2^n-2}}\cdot\left[1+\frac{13t_0^2-13t_0+1}{12(2^n-2)t_0(1-t_0)}\right]+R \\
&= \frac{2^n-1}{2^n-2}\cdot\left[1+\frac{13t_0^2-13t_0+1}{12(2^n-2)t_0(1-t_0)}\right]+R \\
&= \left[1+\frac{1}{2^n-2}\right]\cdot\left[1+\frac{13t_0^2-13t_0+1}{12(2^n-2)t_0(1-t_0)}\right]+R \\
&= 1+\frac{13t_0^2-13t_0+1}{12(\ell-1)t_0}+\mathcal{O}\left(\frac{1}{2^n}\right)+R,
\end{aligned}$$

avec

$$R=\mathcal{O}\left(\frac{C_{2^n,\ell}}{2^n}\left(e^{-\lambda f(t_0-\varepsilon)}+e^{-\lambda f(t_0+\varepsilon)}\right)\left[\frac{t_0(1-t_0)}{\varepsilon}+\frac{2}{3}(2t_0-1)+\frac{13t_0^2-13t_0+1}{12t_0(1-t_0)}\cdot\varepsilon\right]\right).$$

Dans R, la somme entre les crochets est dominée par le premier terme. Ce terme

$$\frac{t_0(1-t_0)}{\varepsilon}\approx\frac{(1-\ell/2^n)\ell/2^n}{z\sqrt{\ell}/2^n}$$

est de l'ordre $\sqrt{\ell}/z$. Nous obtenons alors

$$\begin{aligned}
R &= \mathcal{O}\left(\frac{\sqrt{\ell}C_{2^n,\ell}}{z\cdot 2^n}\left[e^{-(2^n-2)D(t_0||t_0-\varepsilon)}+e^{-(2^n-2)D(t_0||t_0+\varepsilon)}\right]\right) \\
&= \mathcal{O}\left(\frac{\sqrt{\ell}}{z\cdot 2^n}C_{2^n,\ell}e^{-(2^n-2)D(t_0||t_0-\varepsilon)}\left[1+e^{-(2^n-2)[D(t_0||t_0-\varepsilon)-D(t_0||t_0+\varepsilon)]}\right]\right).
\end{aligned}$$

En utilisant le lemme 5.9 nous avons

$$(2^n-2)D\left(t_0||t_0-\varepsilon\right)\approx\frac{2^n\varepsilon^2}{2(t_0-\varepsilon)(1-t_0=\varepsilon)}\approx\frac{z^2\ell/2^n}{2}.$$

Et $\frac{\sqrt{\ell}}{2^n}C_{2^n,\ell}\approx\frac{1}{\sqrt{2\pi}}=\mathcal{O}\left(1\right)$. On en déduit donc que :

$$R=\mathcal{O}\left(\frac{e^{-z^2/2}}{z}\left[1+e^{-(2^n-2)[D(t_0||t_0-\varepsilon)-D(t_0||t_0+\varepsilon)]}\right]\right).$$

En utilisant le même développement limité que le précédent nous avons

$$\begin{aligned}
D\left(t_0||t_0-\varepsilon\right)-D\left(t_0||t_0+\varepsilon\right) &= \frac{\varepsilon^2}{2(t_0-\varepsilon)(1-t_0+\varepsilon)}+\frac{\varepsilon^3(1-2t_0+2\varepsilon)}{3(t_0-\varepsilon)^2(1-t_0+\varepsilon)^2} \\
&- \frac{\varepsilon^2}{2(t_0+\varepsilon)(1-t_0-\varepsilon)}+\frac{\varepsilon^3(1-2t_0-2\varepsilon)}{3(t_0+\varepsilon)^2(1-t_0-\varepsilon)^2}+\mathcal{O}\left(\varepsilon^4\right) \\
&= \frac{2\varepsilon^3(1-2t_0)}{3t_0^2(1-t_0)^2}+\mathcal{O}\left(\varepsilon^4\right).
\end{aligned}$$

Or $\frac{t_0(1-t_0)}{\varepsilon}\approx\frac{\sqrt{\ell}}{z}$, donc $D\left(t_0||t_0-\varepsilon\right)-D\left(t_0||t_0+\varepsilon\right)\approx\frac{2\varepsilon(1-2t_0)z^2}{3\ell}$.

Et $(2^n-2)\left[D\left(t_0||t_0-\varepsilon\right)-D\left(t_0||t_0+\varepsilon\right)\right] \approx \dfrac{2\sqrt{\ell-1}(1-2t_0)z^3}{3\ell} \approx \dfrac{2z^3}{3\sqrt{\ell}}$.

Alors,

$$e^{-(2^n-2)[D(t_0||t_0-\varepsilon)-D(t_0||t_0+\varepsilon)]} = \mathcal{O}\left(e^{-\frac{2z^3}{3\sqrt{\ell}}}\right).$$

Et en conséquence $R = \mathcal{O}\left(\frac{e^{-z^2/2}}{z}\right)$.
Pour conclure cette preuve nous avons

$$\begin{aligned}
\int_{t_0-\varepsilon}^{t_0+\varepsilon} h(t)\,dt &= \left[1+\frac{13t_0^2-13t_0+1}{12(\ell-1)t_0}+\mathcal{O}\left(\frac{1}{2^n}+\frac{e^{-z^2/2}}{z}\right)\right] \\
&\cdot \left[1-\frac{1}{12(\ell-1)}+\mathcal{O}\left(\frac{1}{2^n}+\frac{1}{\ell^2}\right)\right] \\
&= 1-(1-t_0)\frac{13t_0-1}{12t_0}\cdot\frac{1}{\ell-1}+\mathcal{O}\left(\frac{1}{\ell^2}+\frac{1}{2^n}+\frac{e^{-z^2/2}}{z}\right) \\
&= 1+\mathcal{O}\left(\frac{1}{\ell^2}+\frac{1}{2^n}+\frac{e^{-z^2/2}}{z}\right)
\end{aligned}$$

□

La seconde partie de la preuve du théorème 5.3 consiste à exprimer S_2 comme une fonction de $\delta \approx 1-P_S$. Cette preuve peut être faite de la même façon pour S_4.

Lemme 5.15. *Soit* $\delta = \sum_{i=0}^{G^{-1}(t_0)-1} g_*(i)$. *Soit* $\varepsilon = z\frac{\sqrt{\ell-1}}{2^n}$ *pour une certaine valeur de* z *où* $z = o(\sqrt{\ell})$ *quand* ℓ *tend vers l'infini. Si* $\lambda \leq \frac{1}{4}$,*alors*

$$S_2 = \sum_{i=G^{-1}(t_0-\varepsilon)}^{G^{-1}(t_0)-1} g_*(i) = \mathcal{O}\left(\frac{z\theta\delta}{\sqrt{\ell-1}}\right).$$

Preuve : Dans un premier temps, afin de simplifier les formules, nous notons par B et B_ε les valeurs $B \stackrel{\text{def}}{=} F^{-1}(t_0)$ et $B_\varepsilon \stackrel{\text{def}}{=} F^{-1}(t_0-\varepsilon)$. Dans le cas où $B = B_\varepsilon$, il n'y a aucun terme dans la somme. Le lemme est alors prouvé. À partir de maintenant nous supposons que $B \geq B_\varepsilon + 1$.

La preuve de ce lemme 5.15 est basée sur le lemme 5.3. Nous utilisons donc le coefficient suivant

$$\gamma \stackrel{\text{def}}{=} \frac{(1-p)\cdot B}{p\cdot(N-B+1)}. \tag{5.31}$$

Afin de prouver ce lemme, dans un premier temps nous allons prouver que

$$(B-B_\varepsilon)(\gamma-1) = \mathcal{O}\left(\frac{z}{\sqrt{\ell-1}}\right) \tag{5.32}$$

Afin de prouver cette équation nous remarquons en utilisant le lemme 5.3 que

$$\sum_{i=B+1}^{N} f(i) = \theta\left(\frac{f(B)}{1-1/\gamma}\right) = \theta\left(\gamma\frac{f(B)}{\gamma-1}\right). \tag{5.33}$$

Et,

$$(\gamma_-^{B-B_\varepsilon} - 1)\frac{f(B)}{\gamma_- - 1} \leq \sum_{i=B_\varepsilon+1}^{B} f(i) \tag{5.34}$$

où γ_- est défini par

$$\gamma_- \stackrel{\text{def}}{=} \frac{1-p}{p} \min\left(\frac{B}{N-B+1}, \frac{B_\varepsilon+2}{N-B_\varepsilon-1}\right)$$

Comme nous avons supposé que $\frac{\ell-1}{2^n-2} \leq \frac{1}{4}$, nous avons $B > Np$. Pour ℓ suffisament grand $B_\epsilon > Np$. En conséquence pour ℓ suffisament large nous avons $\gamma_- = \gamma$. À partir de l'hypothèse que nous avons faite sur ε, nous savons que $\sum_{i=B_\varepsilon+1}^{B} f(i) = o\left(\sum_{i=B+1}^{N} f(i)\right)$ quand ℓ tend vers l'infini. Ceci est possible si $\gamma_-^{B-B_\varepsilon} - 1$ tend vers 0 quand ℓ tend vers l'infini. Ceci implique que $\gamma_-^{B-B_\varepsilon} - 1 \sim (B - B_\varepsilon)(\gamma_- - 1)$ quand ℓ tend vers l'infini. La même remarque peut être faite en remplaçant γ_- par γ (puisque γ_- coïncide avec γ pour ℓ suffisamment large). En mettant toutes ces remarques ensemble et en utilisant (5.34) et (5.33) nous obtenons

$$\begin{aligned}(B - B_\varepsilon)(\gamma - 1) &\underset{\ell\to\infty}{\sim} \gamma_-^{B-B_\varepsilon} - 1 \\ &= \mathcal{O}\left(\sum_{i=B_\varepsilon+1}^{B} f(i)\frac{\gamma-1}{f(B)}\right) \\ &= \mathcal{O}\left(\frac{\sum_{i=B_\varepsilon+1}^{B} f(i)}{\sum_{i=B+1}^{N} f(i)}\right)\end{aligned}$$

car $\gamma = \mathcal{O}(1)$. Alors, nous pouvons exprimer les sommes apparaissant dans cette fraction comme fonction de ε et t_0 :

$$\sum_{i=B_\varepsilon+1}^{B} f(i) = F(F^{-1}(t_0)) - F(F^{-1}(t_0 - \varepsilon)) = \varepsilon\left[1 + \mathcal{O}\left(\frac{f(B_\varepsilon)}{\varepsilon}\right)\right],$$

et

$$\sum_{i=B+1}^{N} f(i) = 1 - F(F^{-1}(t_0)) = (1 - t_0)\left[1 + \mathcal{O}\left(\frac{f(B)}{1-t_0}\right)\right].$$

Finalement, nous obtenons

$$(B - B_\varepsilon)(\gamma - 1) = O\left(\frac{\varepsilon}{1-t_0}\left[1 + \mathcal{O}\left(\frac{f(B_\varepsilon)}{\varepsilon}\right)\right]\right).$$

Nous pouvons vérifier que $\mathcal{O}\left(\frac{f(B_\varepsilon)}{\varepsilon}\right) = \mathcal{O}(1)$. En remplaçant les valeurs de ε et t_0 par leur vraie valeur nous obtenons

$$(B - B_\varepsilon)(\gamma - 1) = \mathcal{O}\left(\frac{z}{\sqrt{\ell - 1}}\right).$$

Maintenant que nous avons prouvé l'assertion (5.32) nous pouvons nous concentrer de nouveau sur la preuve du lemme 5.15. Nous pouvons une nouvelle fois utiliser le lemme 5.3 pour obtenir les expressions suivantes

$$\sum_{i=0}^{B-1} f_0(i) = \mathcal{O}\left(\frac{f_0(B)}{1-\gamma_0}\right) \tag{5.35}$$

$$\sum_{i=B_\varepsilon}^{B} f_0(i) = \mathcal{O}\left(\frac{(1-\gamma_0^{B-B_\varepsilon})f_0(B)}{1-\gamma_0}\right) \tag{5.36}$$

Nous avons

$$\begin{aligned} 1-\gamma_0^{B-B_\epsilon} &= \mathcal{O}\left((B-B_\epsilon)(1-\gamma_0)\right) \\ &= \mathcal{O}\left(\theta(\gamma-1)(B-B_\varepsilon)\right) \\ &= \mathcal{O}\left(\theta\frac{z}{\sqrt{\ell-1}}\right), \end{aligned}$$

où $\theta = \mathcal{O}\left(\frac{1-\gamma_0}{1-\gamma}\right)$ En remettant ce résultat dans l'équation (5.36) et en utilisant le fait que $\delta \stackrel{\text{def}}{=} \sum_{i=0}^{B-1} f_0(i)$ nous obtenons :

$$\sum_{i=B_\varepsilon}^{B} f_0(i) = \mathcal{O}\left(\theta\frac{z}{\sqrt{\ell-1}\delta}\right)$$

et nous avons prouvé le lemme 5.15 □

Preuve du théorème

À partir des lemmes que nous venons de prouver nous pouvons revenir sur la preuve du théorème 5.3.

Preuve : Rappelons que nous voulons borner supérieurement l'erreur suivante :

$$P_S - \sum_{i=G^{-1}(t_0)}^{N} g_*(i) = S_1 + S_2 - S_4 - S_5.$$

Nous commençons par borner S_1 et S_5 :

$$\begin{aligned} S_1 &= \sum_{i=0}^{G^{-1}(t_0-\varepsilon)-1} g_*(i)\int_0^{G(i)} h(t)\,dt \le \int_0^{t_0-\varepsilon} h(t)\,dt, \\ S_5 &= \sum_{i=G^{-1}(t_0+\varepsilon)}^{N} g_*(i)\int_{G(i)}^{1} h(t)\,dt \le \int_{t_0+\varepsilon}^{1} h(t)\,dt. \end{aligned}$$

Or, $|S_1 - S_5| \le S_1 + S_5 \le 1 - \int_{t_0-\varepsilon}^{t_0+\varepsilon} g(t)\,dt$. Donc, en utilisant le lemme 5.14 nous avons :

$$|S_1 - S_5| = \mathcal{O}\left(\frac{1}{\ell^2} + \frac{1}{2^n} + \frac{e^{-z^2/2}}{z}\right). \tag{5.37}$$

Pour montrer que S_2 est négligeable nous utilisons le lemme 5.15. Une preuve peut être faite de la même manière pour montrer S_4 est négligeable. Nous avons alors

$$|S_2 - S_4| = \mathcal{O}\left(\delta\frac{z}{\sqrt{\ell}}\right). \tag{5.38}$$

En ajoutant les équations (5.37) et (5.38), nous obtenons

$$P_S - \sum_{i=G^{-1}(t_0)}^{N} g_*(i) = \mathcal{O}\left(\frac{1}{\ell^2} + \frac{1}{2^n} + \frac{e^{-z^2/2}}{z} + \delta\frac{z}{\sqrt{\ell}}\right).$$

La dernière étape de la preuve consiste à choisir une valeur particulière pour z. En prenant z de la forme $z = \sqrt{\ln\left(\frac{\ell}{\delta^2}\right)}$ nous obtenons

$$P_S - \sum_{i=G^{-1}(t_0)}^{N} g_*(i) = \mathcal{O}\left(\delta\sqrt{\frac{\ln(\ell/\delta^2)}{\ell}} + \frac{1}{\ell^2} + \frac{1}{2^n}\right).$$

Le choix de z que nous avons pris vérifie bien la condition réclamée dans les lemmes qui était $\frac{e^{-z^2/2}}{z} = \mathcal{O}\left(\delta\frac{z}{\sqrt{\ell}}\right)$.

□

5.4.4 Lien avec les formules existantes

Le calcul de la probabilité de succès d'une attaque différentielle ou linéaire avait déjà été fait par Selçuk [Sel08]. Cette étude reposait sur l'hypothèse que la distribution des statistiques d'ordre étudiée convergeait vers une loi normale.

Description des travaux de Selçuk

Nous rappelons ici un des théorèmes principaux de l'article [Sel08].

Théorème 5.4. *Soit ϕ_* la fonction densité de la loi gaussienne de moyenne Np_* et de variance $Np_*(1-p_*)$. Soit Φ^{-1} l'inverse de la fonction de répartition de la loi normale de paramètre Np et $Np(1-p)$. Alors une bonne approximation de la probabilité de succès est*

$$P_S \approx \int_{\Phi^{-1}(1-\ell/2^n)}^{\infty} \phi_*(x)\,dx. \tag{5.39}$$

On peut remarquer que la formule donnée dans le théorème 5.3 et la formule de la probabilité de succès donnée par Selçuk sont très similaires. En effet les fonctions Φ et ϕ_* sont des approximations de G et g_*. Ainsi la formule de Selçuk est bonne lorsque l'approximation gaussienne l'est. Le problème de l'utilisation de l'approximation gaussienne est soulevé par Selçuk lui-même lors de l'étude de la complexité en données d'une attaque différentielle. Les résultats expérimentaux montrent que son approximation de la probabilité de succès est bonne dans le cas de la cryptanalyse linéaire mais est très loin de la réalité pour la cryptanalyse différentielle.

5.4.5 Résultats expérimentaux

Nous avons fait des expérimentations afin de comparer notre formule de la probabilité de succès avec celle donnée par Selçuk (voir théorème 5.4). Nous avons aussi comparé cette formule avec la vraie valeur de la probabilité de succès. Cette valeur a pu être calculée en utilisant une astuce simple qui permet de calculer la densité de la loi bêta avec une grande précision. Dans le cas de la cryptanalyse linéaire comme l'approximation gaussienne est bonne, notre formule de la probabilité de succès donne le même résultat que la formule de la probabilité de succès donnée par Selçuk. Cependant dans le cas de la cryptanalyse différentielle, la formule donnée par Selçuk est trop optimiste alors que notre formule donnée par le théorème 5.3 est proche de la vraie valeur de la probabilité de succès. Les résultats de certaines des expérimentations que nous avons faites sont donnés dans le tableau 5.6 et illustrent bien le phénomène décrit ci dessus.

Type de cryptanalyse	Probabilités	Paramètres $N = 2^{48}$ $2^n = 2^{20}$	P_S	Notre estimation de P_S (5.22)	Estimation de [Sel08] de P_S (5.39)
Linéaire	$p = 0.5$ $p_* = p + 1.49 \cdot 2^{-24}$	$\ell = 2^{15}$	0.8681	0.8681	0.8681
Linéaire	$p = 0.5$ $p_* = p + 1.49 \cdot 2^{-24}$	$\ell = 2^{10}$	0.4533	0.4533	0.4533
Différentielle	$p = 2^{-64}$ $p_* = 2^{-47.2}$	$\ell = 2^{15}$	0.8257	0.8247	0.9050
Différentielle	$p = 2^{-64}$ $p_* = 2^{-47.2}$	$\ell = 2^{10}$	0.8250	0.8247	0.9050

TABLE 5.6 – Comparaison entre les équations (5.22) et (5.39) avec la vraie valeur de la probabilité de succès.

5.4.6 Lien entre la probabilité de succès et la complexité en données

Nous avons mené d'autre expérimentations dans le but de montrer que quand nous choisissons N de la forme

$$N = -c \cdot \frac{\ln(2\sqrt{\pi} \cdot \ell / 2^n)}{D\left(p_* || p\right)},$$

(ce choix est guidé par la formule donnée dans le théorème 5.2) alors la probabilité de succès de l'attaque dépend essentiellement de la valeur de c et est indépendante du type de cryptanalyse. Pour illustrer ce propos nous avons calculé dans le tableau 5.7 plusieurs valeurs de probabilité de succès pour un nombre total de clés fixé à $2^n = 2^{30}$, pour différentes valeurs de taille de liste ℓ et pour différents types de cryptanalyses. Ces probabilités de succès ont été calculées à l'aide de la formule donnée dans le théorème 5.3. Les valeurs calculées dans le tableau 5.7 pour plusieurs valeurs de c montrent que la probabilité de succès dépend essentiellement de la valeur de c et est indépendante du type de cryptanalyse.

	$c=1$			$c=1.5$			$c=2$		
Paramètres	ℓ			ℓ			ℓ		
	2^{10}	2^{25}	2^{40}	2^{10}	2^{25}	2^{40}	2^{10}	2^{25}	2^{40}
$p=0.5$ $p_*=p+1.49\cdot 2^{-24}$	0.5855	0.5922	0.6012	0.9799	0.9606	0.9169	0.9998	0.9988	0.9902
$p=0.5$ $p_*=p+1.23\cdot 2^{-11}$	0.5856	0.5924	0.6013	0.9800	0.9606	0.9170	0.9998	0.9988	0.9903
$p=2^{-30}$ $p_*=1.2\cdot 2^{-30}$	0.5802	0.5847	0.6117	0.9766	0.9580	0.9105	0.9998	0.9985	0.9880
$p=2^{-40}$ $p_*=1.2\cdot 2^{-40}$	0.5802	0.5847	0.5981	0.9766	0.9580	0.9105	0.9998	0.9985	0.9880
$p=2^{-64}$ $p_*=2^{-60}$	0.5496	0.5976	0.5300	0.9078	0.8936	0.8054	0.9928	0.9783	0.9292
$p=2^{-32}$ $p_*=2^{-29}$	0.6421	0.7058	0.6817	0.9381	0.8936	0.8875	0.9959	0.9880	0.9832

TABLE 5.7 – Probabilité de succès pour différents paramètres avec $2^n = 2^{60}$ et $N = -c \cdot \frac{\ln(2\sqrt{\pi}\cdot\ell/2^n)}{D(p_*||p)}$.

5.5 Amélioration de la formule de la probabilité de succès dans le cas de la cryptanalyse différentielle

Dans la section précédente nous avons vu que la formule de la probabilité de succès que nous donnons dans le théorème 5.3 est bonne et assez proche de la vraie formule lorsque l'on utilise la loi binomiale. Cependant dans le cas de la cryptanalyse différentielle nous avons remarqué expérimentalement que pour une différentielle fixée, les clés suivaient aussi une distribution binomiale (voir section 4.3). Cette hypothèse n'a pas été prise en compte dans notre analyse générale de la probabilité de succès.

Dans cette section nous présentons les résultats d'une attaque expérimentale que nous avons fait sur SMALLPRESENT-[8]. En utilisant la remarque que nous avons faite dans la section 4.3, nous donnons une formule de la probabilité de succès plus précise dans le cas de la cryptanalyse différentielle.

5.5.1 Amélioration de la formule de la probabilité de succès dans le cas de la cryptanalyse différentielle

L'observation faite dans la section 4.3 que la répartition des clés suit une distribution binomiale peut être prise en compte dans le calcul de la probabilité de succès d'une attaque différentielle.

On commence par rappeler que dans la section précédente nous avons montré que la probabilité de succès (voir théorème 5.3) est proche de

$$1 - \sum_{i=0}^{G^{-1}(t_0)-1} g_*(i),$$

où G est la fonction de répartition d'une loi binomiale de paramètres (N, p_*) et g_* est la densité d'une loi binomiale de paramètres (N, p_*). Nous notons ici par $P_s(p_*)$ cette quantité. À partir de maintenant comme la distribution des différences est différente suivant les clés nous notons par $P_s(q)$ la quantité suivante

$$P_s(q) = 1 - \sum_{i=0}^{G^{-1}(t_0)-1} g_q(i),$$

où g_q est la densité d'une loi binomiale avec paramètres N et q.

Proposition 5.1. *En utilisant les notations précédentes, la probabilité de succès d'une attaque différentielle utilisant une seule différentielle avec probabilité p_* est :*

$$P_S = \sum_{i=0}^{2^{m-1}} P_S\left(\frac{i}{2^{m-1}}\right) \cdot \left[p_*^i (1-p_*)^{2^{m-1}-i} \binom{2^{m-1}}{i}\right]. \tag{5.40}$$

Dans la section suivante, nous validons expérimentalement cette nouvelle formule. Et nous montrons que la formule donnée dans la proposition 5.1 est plus exacte que la formule donnée dans le théorème 5.3.

5.5.2 Cryptanalyse différentielle de SMALLPRESENT-[8]

Nous présentons ici une cryptanalyse différentielle simple sur SMALLPRESENT-[8]. Dans cette cryptanalyse nous utilisons la différentielle suivante sur 7 tours du système de chiffrement :

$$(a_0, a_7) = (\texttt{0x7}, \texttt{0x2a2a0000}).$$

Cette différentielle arrive avec probabilité

$$p_* = 2^{-24.885}.$$

Afin de faire une attaque rapide, nous n'inversons qu'un seul tour de clé. Comme au dernier tour nous avons 4 boîtes-S actives nous avons 2^{16} clés candidates. Dans notre attaque, nous avons choisi de ne garder que $\ell = 2^{10}$ candidats (voir algorithme 2).
Afin de mesurer la probabilité de succès expérimentale nous avons fait 200 expérimentations. Les résultats expérimentaux sont donnés dans la figure 5.1. Sur ce même graphique nous avons aussi dessiné la courbe de la probabilité de succès obtenue par la formule donnée dans le théorème 5.3 et la courbe de la nouvelle formule de la probabilité de succès que nous avons obtenue après avoir pris en compte la distribution binomiale des clés.

Dans le cas de la cryptanalyse différentielle les formules théoriques de la probabilité de succès ne donnent pas toujours de bons résultats. Ceci peut s'expliquer facilement pas le fait que comme p_* et p sont éloignées, $G(t_0)$ (où $t_0 = 1 - \frac{\ell-1}{2^n-2}$) est souvent égal à 0 et donc les formules de la probabilité de succès sont plus ou moins indépendantes de la taille de la liste des clés gardées.
Cependant dans ce graphique on remarque que la courbe de la probabilité de succès obtenue grâce à la proposition 5.1 est plus proche des résultats expérimentaux.

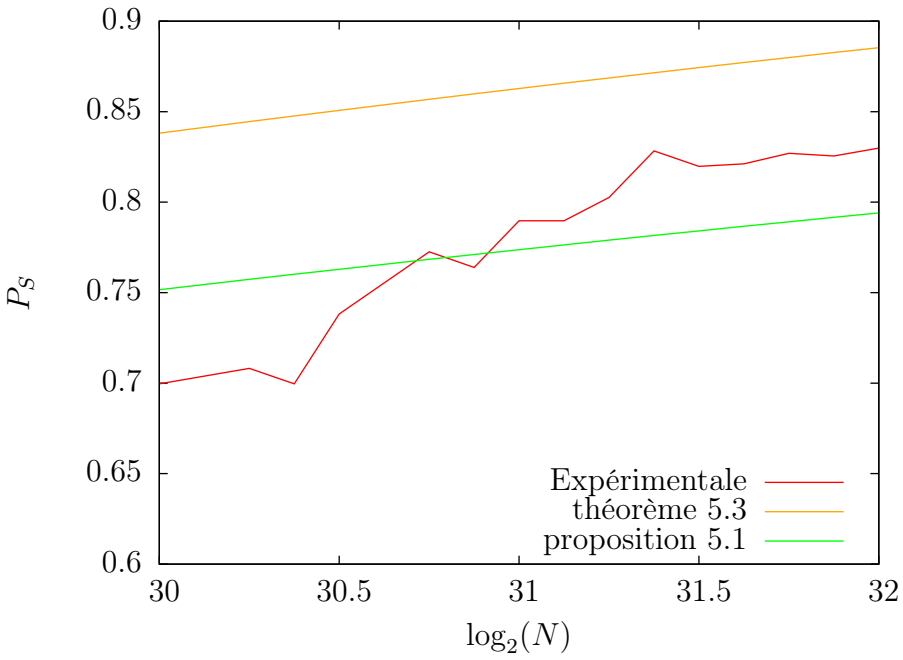

FIGURE 5.1 – Probabilité de succès de l'attaque différentielle sur SMALLPRESENT-[8] spécifiée dans la section 5.5.2

Dans ce chapitre, nous avons étudié la complexité en données et la probabilité de succès d'une attaque statistique dans laquelle les variables aléatoires suivent une loi binomiale. Un certain nombre des généralisations de la cryptanalyse différentielle rentrent dans ce contexte. On peut cependant réfléchir à d'autres types de cryptanalyses qui généraliseraient la cryptanalyse différentielle et qui ne rentrent pas dans ce contexte.
Dans le chapitre suivant nous présentons une nouvelle généralisation de la cryptanalyse différentielle tronquée. Dans cette attaque, les variables aléatoires étudiées ne suivent pas des distributions binomiales. Une autre étude de la complexité en données et de la probabilité de succès est alors nécessaire.

Chapitre 6

La cryptanalyse différentielle multiple

Dans le chapitre 2 nous avons vu qu'il existait un certain nombre de variantes de la cryptanalyse différentielle. Certaines de ces cryptanalyses tirent de l'information à partir de plusieurs différentielles. Nous présentons ici une généralisation de ces attaques. Ce chapitre a pour objet nos travaux avec Benoît Gérard dont les principaux résultats ont été présentés à FSE 2011 [BG11].

Nous avons introduit la cryptanalyse différentielle multiple dans le but d'avoir une attaque plus performante que l'attaque différentielle classique ou que l'attaque différentielle tronquée.

L'idée de la cryptanalyse différentielle multiple consiste à exploiter l'information fournie par plusieurs différentielles n'ayant pas forcément de lien direct entre elles. Dans la cryptanalyse différentielle classique, celle introduite par Eli Biham et Adi Shamir, l'attaquant exploite de l'information venant de plusieurs différentielles ayant la même différence en sortie. Dans les attaques différentielles tronquées classiques l'attaquant exploite des différentielles telles que pour chaque différence en entrée étudiée l'ensemble des différences en sortie est le même. La notion de différentielle multiple que nous introduisons dans ce chapitre regroupe ces deux types d'attaques.

Dans ce chapitre nous abordons le problème en prenant les différentielles qui ont les meilleures probabilités.

Contrairement aux autres généralisations de la cryptanalyse différentielle décrites dans le chapitre 2, les variables aléatoires étudiées ici ne suivent pas une loi binomiale. Ainsi l'étude que nous avons faite dans le chapitre 5 pour calculer la complexité en données et la probabilité de succès d'une attaque statistique simple ne s'applique pas ici. Dans ce chapitre après avoir présenté la cryptanalyse différentielle multiple, nous étudions la distribution des variables aléatoires qui sont impliquées dans l'attaque afin de pouvoir calculer la complexité en données et la probabilité de succès d'une attaque différentielle multiple. Comme nous l'avons fait régulièrement dans les chapitres précédents, nous testons cette attaque sur une version réduite de PRESENT.

6.1 La cryptanalyse différentielle multiple

Dans un premier temps nous posons les notations de ce chapitre.

6.1.1 Contexte

Dans la cryptanalyse différentielle multiple, étudiée avec Benoît Gérard, nous cherchons à exploiter de l'information à partir d'un certain nombre de différences.

Nous notons par A l'ensemble des différentielles que l'attaquant cherche à exploiter.

$$A \stackrel{\text{def}}{=} \{(a_0, a_r) \in \mathbb{F}_2^m \times \mathbb{F}_2^m\},$$

où a_r est une différence après r tours.

Une façon naturelle d'ordonner ces différentielles consiste à regrouper toutes les différentielles qui ont la même différence en entrée. Nous notons par A_0 l'ensemble des différences en entrée qui sont comprises dans A :

$$A_0 \stackrel{\text{def}}{=} \{a_0,\ \exists a_r, (a_0, a_r) \in A\}.$$

Soit $\#A_0$ le nombre de différences en entrée ; nous indexons les éléments de A_0 :

$$A_0 = \{{a_0}^{(1)}, \ldots, {a_0}^{(\#A_0)}\}.$$

Donc, pour une différence en entrée fixée ${a_0}^{(i)} \in A_0$ ($i \in \{1..\#A_0\}$), nous définissons l'ensemble des différences en sortie correspondantes $A_r^{(i)}$ par :

$$A_r^{(i)} \stackrel{\text{def}}{=} \{a_r \mid ({a_0}^{(i)}, a_r) \in A\}.$$

L'ensemble des différentielles A qui sont impliquées dans l'attaque différentielle multiple peut alors s'exprimer

$$A = \left\{ \left({a_0}^{(i)}, a_r^{(i,j)}\right) \;\middle|\; i = 1 \ldots \#A_0 \text{ and } j = 1 \ldots \#A_r^{(i)} \right\}.$$

On ne peut pas parler de différentielles sans les associer à leurs probabilités. Soit F la fonction de tour d'un système de chiffrement avec clé maître K, F_K^r correspond à r tours de ce système de chiffrement Ainsi nous notons par $p_*^{(i,j)}$ la probabilité de la différentielle $({a_0}^{(i)}, a_r^{(i,j)})$:

$$p_*^{(i,j)} = P_{\mathbf{X},\mathbf{K}}\left[F_K^r(X) + F_K^r(X + {a_0}^{(i)}) = a_r^{(i,j)}\right].$$

La probabilité théorique d'une différentielle peut être obtenue grâce à un algorithme "branch and bound" (voir algorithme 12)

6.1.2 L'algorithme décrivant l'attaque

L'attaque différentielle multiple est assez similaire à l'attaque différentielle classique (voir algorithme 2). Nous présentons l'algorithme dans le cas d'une attaque sur le dernier tour d'un système de chiffrement de type substitution-permutation. L'attaque consiste alors à déchiffrer partiellement les N_{DC} messages chiffrés en utilisant toutes les clés possibles du dernier tour et à compter le nombre d'occurrences des différentielles dans A. En d'autres termes, nous comptons le nombre de paires de messages clairs avec différence en entrée ${a_0}^{(i)} \in A_0$ qui conduisent à une différence dans $A_r^{(i)}$ après r tours. Comme dans l'attaque différentielle classique, dans le but de réduire la complexité en temps de l'attaque, un crible est utilisé pour supprimer certaines paires de messages. La particularité

ici est que ce crible dépend de la différence en entrée. Ainsi pour chaque différence en entrée ${a_0}^{(i)}$ nous définissons un crible $\Delta_{sieve}^{(i)}$. Ce crible consiste en toutes les différences possibles après un tour sachant que les différences précédentes appartenaient à $A_r^{(i)}$:

$$\forall\, i \quad \Delta_{sieve}^{(i)} = \bigcup_{a_r \in A_r^{(i)}} \left\{ a \mid P\left[a_r \underset{F}{\rightarrow} a \right] \right\}.$$

L'attaque différentielle multiple qui consiste à ajouter le nombre d'occurrences de chaque différentielle est décrite dans l'algorithme 14.

Algorithme 14 : Cryptanalyse différentielle multiple d'un système de chiffrement de type substitution-permutation

Entrée : N_{DC} couples de clairs/chiffrés (X_i, Y_i) avec $Y_i = E_{K^*}(X_i)$
Sortie : La clé K^* utilisée pour chiffrer les messages
Initialiser une table C de 2^n compteurs à 0.
Pour chaque ${a_0}^{(i)} \in A_0$ **faire**
  Pour chaque *paire de messages* (X_a, X_b) *tel que* $X_b = X_a \oplus {a_0}^{(i)}$ **faire**
    Si $Y_a \oplus Y_b \in \Delta_{sieve}^{(i)}$ **alors**
      Pour chaque sous clé candidate k **faire**
        Calculer $\delta = F_k^{-1}(Y_a) \oplus F_k^{-1}(Y_b)$;
        Si $\delta \in A_r^{(i)}$ **alors**
          $C[k] \leftarrow C[k] + 1$;
Générer une liste $\mathcal{L}$ des ℓ candidats avec la plus grande valeur de $C[k]$;
Pour chaque $k \in \mathcal{L}$ **faire**
  Pour chaque clé maître possible K correspondant à k **faire**
    Si $E_K(X) = Y = E_{K^*}(X)$ **alors retourner** K;

Cet algorithme va nous servir de support pour calculer la complexité en temps et en mémoire d'une attaque différentielle multiple. Dans le chapitre 2, nous n'avons pas pris le temps de détailler la complexité en temps des attaques différentielles et différentielles tronquées. Ces deux cryptanalyses sont des cas particulier de la cryptanalyse différentielle multiple, la section suivante est dédiée à l'étude de ces deux complexités.

6.1.3 La complexité en temps et en mémoire

Afin de comparer une attaque statistique à une autre en plus de la probabilité de succès et de la complexité en données, il est aussi intéressant de comparer la complexité en temps et en mémoire. Dans cette section nous détaillons donc brièvement le calcul de ces complexités dans le cas de l'attaque présentée dans l'algorithme 14. La complexité en temps d'une attaque statistique peut se découper en trois phases importantes [1] :

Phase de distillation : Pour chaque paire de différences en sortie qui passe le crible, l'attaquant doit inverser partiellement la fonction de tour pour toutes les clés candidates possibles.

1. Ces trois phases sont décrites dans le chapitre 1 dans la section 1.5

Phase d'analyse : Pour les variables aléatoires étudiées, les trier afin de garder une liste des candidats les plus probables.

Phase de recherche exhaustive : Pour toutes le clés candidates qui sont dans la liste il faut tester toutes les clés maîtres correspondantes afin de trouver la bonne.

Complexité en temps de la phase de distillation

Nous analysons dans un premier temps la complexité en temps de la phase de distillation. Pour ceci nous avons besoin d'introduire un certain nombre de notations. Nous notons par S_r (resp. S_{r+1}) le cardinal maximum des différences possibles en sortie (resp. le cardinal maximum du crible) :

$$S_r \stackrel{\text{def}}{=} \max_i \{\#A_r^{(i)}\} \text{ et } S_{r+1} \stackrel{\text{def}}{=} \max_i \{\#\Delta_{sieve}^{(i)}\}.$$

Pour calculer la complexité en données nous nous plaçons dans le pire des cas, c'est-à-dire que nous allons supposer que tous les cribles ont la même taille, égale à S_{r+1}.

Soit m le nombre de bits de la sortie de la fonction de chiffrement. Nous notons par p_{sieve} la probabilité maximale sur toutes les différences en entrée A_0 qu'une paire passe le crible :

$$p_{\text{sieve}} = 2^{-m} S_{r+1}.$$

Quand l'ensemble des différences en sortie n'est pas réduit à un élément, nous avons besoin de vérifier si une différence appartient à un certain ensemble. En supposant que l'ensemble A est trié, cette étape peut se faire à l'aide d'un recherche dichotomique. La complexité en temps de cette vérification est alors en $\mathcal{O}(\log(\#A))$.

Dans l'algorithme 14, le nombre de paires à tester est

$$N = \#A_0 N_{DC}/2.$$

Pour chaque paire nous devons vérifier les paires qui passent le crible. Ceci peut se faire avec une complexité de $N \log(S_{r+1})$.

Cependant, on peut réduire la complexité en temps de cette étape lorsque les cribles $\Delta_{sieve}^{(i)}$ se ressemblent.

L'inversion partielle de la fonction de tour doit être faite pour toutes les paires qui passent le crible et pour toutes les clés candidates possibles. Donc, la complexité de la phase de distillation est en

$$\mathcal{O}\left(2^n N p_{\text{sieve}}\right) \text{ déchiffrements/chiffrements}$$

et

$$\mathcal{O}\left(2^n N 2^{-m} S_{r+1} \log(1 + S_r)\right) \text{ comparaisons}$$

Complexité en temps de la phase d'analyse

L'étape consistant à extraire la liste $\mathcal{L}$ des ℓ candidats les plus probables peut être faite en temps linéaire, en fonction du nombre 2^n de clés testées.

Complexité en temps de la phase de recherche exhaustive

La dernière partie de l'algorithme correspond à une recherche exhaustive des bits restant de la clé maître. Cette étape qui peut être coûteuse nécessite

$$\mathcal{O}\left(\ell \cdot \frac{2^{\Omega}}{2^{n}}\right) \text{ applications de la fonction de chiffrement}$$

où Ω correspond au nombre de bits de la clé maître.

Il est difficile de prédire laquelle de ces étapes est la plus coûteuse. En effet suivant les paramètres de l'attaque (notamment le nombre de clés testées et la taille de la liste des clés gardées) la complexité en temps de la phase de distillation peut être plus ou moins importante que la complexité en temps de la phase de recherche exhaustive.

Le tableau 6.1 donne le détail de la complexité en temps. Les termes correspondant aux étapes avec une complexité en temps négligeable sont omis ici,

Chiffrement	Dechiffrement partiel	Comparaisons
$\mathcal{O}\left(\ell 2^{\Omega-n}\right)$	$\mathcal{O}\left(2^{n} N p_{\text{sieve}}\right)$	$\mathcal{O}\left(2^{n} N 2^{-m} S_{r+1} \log(1+S_r)\right)$

TABLE 6.1 – Complexité en temps d'une attaque différentielle multiple. Les quantités S_r (resp. S_{r+1}) correspondent au nombre maximum de différences pour une différence en entrée dans A_0 après r tours (resp. $(r+1)$ tours).

La complexité en mémoire d'une attaque est essentiellement due au stockage des compteurs, des paires de messages qui passent le crible et au stockage des cribles $\Delta^{(i)}_{sieve}$.

6.2 La statistique étudiée

L'étude de la complexité en données et de la probabilité de succès d'une attaque différentielle multiple comme définie précédemment nécessite la connaissance de la distribution des variables aléatoires utilisées. Cette section est dédiée à la définition des variables aléatoires impliquées et à l'étude de leur distribution.

6.2.1 Les variables aléatoires simples

Dans cette section, nous rappelons que E_K est un système de chiffrement avec fonction de tour F. Nous supposons que le système de chiffrement est composé de $r+1$ tours et que l'on cherche à retrouver de l'information sur la clé du $r+1$ème tour.

Définition 6.1. *Nous notons par $C_{{a_0}^{(i)},X,k}$ les variables aléatoires simples pour une différence en entrée ${a_0}^{(i)}$ fixée et l'ensemble des différences en sortie $A_r^{(i)}$ correspondant. Pour un message clair donné X et pour une clé candidate donnée k nous avons*

$$C_{{a_0}^{(i)},X,k} \stackrel{\text{def}}{=} \begin{cases} 1 & si\ F_k^{-1}\big(E_{K^*}(X)\big) \oplus F_k^{-1}\big(E_{K^*}(x \oplus {a_0}^{(i)})\big) \in A_r^{(i)}, \\ 0 & sinon. \end{cases}$$

Définition 6.2. *En utilisant les notations de la définition 6.1, nous obtenons les variables* $C_{X,k}$ *ou* $C_{a_0{}^{(i)},X,k}$ *selon que l'on somme sur les messages clairs ou sur les différences en entrée :*

$$C_{X,k} = \sum_{i=1}^{\#A_0} C_{a_0{}^{(i)},X,k},$$

$$C_{a_0{}^{(i)},k} = \frac{1}{2}\sum_{x} C_{a_0{}^{(i)},X,k}.$$

Les variables aléatoires qui nous intéressent dans le cas de notre cryptanalyse sont alors les suivantes :

Définition 6.3. *En utilisant les notations de la définition 6.1 et de la définition 6.2, les variables aléatoires que nous regardons dans le cas de notre cryptanalyse sont*

$$C_k = \frac{1}{2}\sum_{X} C_{X,k}, \tag{6.1}$$

ou de manière équivalente,

$$C_k = \sum_{i=1}^{\#A_0} C_{a_0{}^{(i)},k}. \tag{6.2}$$

Afin de déterminer la distribution des variables aléatoires définies dans la définition 6.2, nous commençons par une petite discussion sur la meilleure façon de sommer.

6.2.2 Distribution des variables aléatoires simples

On note par $p^{(i,j)}$ la probabilité suivante

$$p^{(i,j)} = P_{\mathbf{X}}\left[F_k^{-1}(E_{K^*}(X)) \oplus F_k^{-1}(E_{K^*}(X \oplus a_0{}^{(i)})) = a_r^{(i,j)} \middle| k \neq k^*\right]$$

En utilisant l'hypothèse de répartition aléatoire par fausse clé définie dans le cas de la cryptanalyse différentielle (voir hypothèse 2.1), nous obtenons que les variables aléatoires simples $C_{a_0{}^{(i)},X,k}$ (voir définition 6.2) suivent une loi de Bernoulli de paramètre

$$\begin{cases} p_*^{(i)} \stackrel{\text{def}}{=} \sum_{j=1}^{\#A_r^{(i)}} p_*^{(i,j)} & \text{si } k = k^*, \\ \quad et & \\ p^{(i)} \stackrel{\text{def}}{=} \sum_{j=1}^{\#A_r^{(i)}} p^{(i,j)} \approx \#A_r^{(i)}\, 2^{-m} & \text{sinon.} \end{cases}$$

Il existe deux moyens de sommer ces variables aléatoires simples. On peut sommer sur les messages clairs. A ce moment là on obtient les variables aléatoires $C_{a_0{}^{(i)},k}$. Ces variables aléatoires sont alors une somme de variables de Bernoulli avec même paramètre. Sous une hypothèse d'indépendance, ces variables aléatoires suivent une loi binomiale de paramètres $N_{DC}/2$ et $p^{(i)}$ ou $p_*^{(i)}$ suivant la clé candidate.
On rappelle que les variables aléatoires que nous utilisons dans notre cryptanalyse correspondent à la somme des variables aléatoires simples $C_{a_0{}^{(i)},X,k}$. Le problème qui se pose

alors est que l'on ne peut pas définir aisément la distribution de la somme de variables binomiales qui n'ont pas la même probabilité. Pour cette raison, pour trouver la distribution des variables aléatoires C_k, nous avons choisi de sommer d'abord sur les différences en entrée avant de sommer sur les messages clairs. Ce choix est purement théorique et ne change rien au principe de l'attaque ni à la puissance de celle-ci. De cette façon on obtient que les variables aléatoires $C_x(k)$ sont identiquement distribuées.

6.2.3 Approximation par une loi de Poisson

Les variables aléatoires $C_{{a_0}^{(i)},X,k}$ sont des variables aléatoires indépendantes qui suivent des lois de Bernoulli avec paramètres différents. La somme des ces variables aléatoires ne suit alors pas une loi binomiale. Cependant grâce à un résultat de Le Cam [Cam60] nous pouvons dire que la distribution des variables aléatoires $C_{X,k}$ est proche d'une loi de Poisson.

Théorème 6.1. *[Cam60] Soit $C_{{a_0}^{(i)},X,k}$ un ensemble de $\#A_0$ variables aléatoires indépendantes qui suivent des lois de Bernoulli de paramètre $p^{(i)}$. Soient*

$$C_{X,k} \stackrel{\text{def}}{=} \sum_{i=1}^{\#A_0} C_{{a_0}^{(i)},X,k}$$

et

$$\lambda = \sum_{i=1}^{\#A_0} p^{(i)}.$$

Alors, pour tout ensemble $A \subset \{0, 1, \ldots, \#A_0\}$, nous avons

$$\left| P\left[C_{X,k} \in A\right] - \sum_{a \in A} \frac{\lambda^a e^{-\lambda}}{a!} \right| < \sum_{i=1}^{\#A_0} {p^{(i)}}^2.$$

Il est facile de vérifier que dans notre contexte les variables aléatoires $C_{{a_0}^{(i)},X,k}$ sont indépendantes puisque pour une clé fixée et un message fixé la connaissance des valeurs $F_K^r(X) \oplus F_K^r(X \oplus {a_0}^{(i)})$ ne nous donne pas d'information sur les $F_K^r(X) \oplus F_K^r(X \oplus \delta)$ si $\delta \neq {a_0}^{(i)}$.

À partir de ce théorème nous pouvons donc déduire que la distribution des variables aléatoires $C_{X,k}$ est proche d'une distribution de Poisson avec paramètre $\sum_{i=1}^{\#A_0} p^{(i)}$.

Lemme 6.1. *En utilisant le théorème 6.1 nous avons que*

- *La distribution des variables aléatoires C_{X,k^*} est proche d'une loi de Poisson de paramètre $\sum_{i=1}^{\#A_0} p_*^{(i)}$.*
- *La distribution des variables aléatoires $C_{X,k}$ pour $k \neq k^*$ est proche d'une loi de Poisson de paramètre $\sum_{i=1}^{\#A_0} p^{(i)}$.*

Comme la loi de Poisson est stable par addition, si l'on somme des variables aléatoires indépendantes, nous voulons pouvoir en déduire que $\sum_x C_{X,k}$ suit une loi de Poisson avec paramètre $\frac{N_{DC}}{2} \cdot \sum_{i=0}^{\#A_0} p^{(i)}$. Si l'on regarde plus en détail, cela n'est pas le cas puisque

chaque différentielle est comptée deux fois et en conséquence les variables aléatoires $C_{X,k}$ sont dépendantes. Dans le cas de la cryptanalyse différentielle classique (une seule différence en entrée) il est facile de voir que pour avoir des variables aléatoires indépendantes il suffit de sommer sur la moitié des messages. Quand le nombre de différences augmente, pour pouvoir sommer sur seulement la moitié des messages il faut alors que l'ensemble des différences en entrée possède une certaine propriété.
En effet s'il existe un ensemble $\mathcal{X}$ de cardinalité $N_{DC}/2$ tel que pour tout X et X' dans X il n'existe pas de différence en entrée ${a_0}^{(i)}$ tel que $x = x' \oplus {a_0}^{(i)}$, alors nous avons que $\sum_{x\in\mathcal{X}} C_{X,k} = \frac{1}{2}\sum_x C_{X,k} = C_k$. Un ensemble de différence en entrée qui vérifie cette condition est dit *admissible*.

Définition 6.4. *L'ensemble des différences en entrée A_0 est dit* **admissible** *s'il est tel qu'il existe un ensemble $\mathcal{X}$ de $N_{DC}/2$ messages clairs qui vérifient :*

$$\forall {a_0}^{(i)} \in A_0, \forall x \in \mathcal{X}, x \oplus {a_0}^{(i)} \notin \mathcal{X}. \tag{6.3}$$

Pour la suite de notre analyse on suppose que l'ensemble des différences en entrée A_0 est *admissible*. Si on est en possession d'un certain nombre de différences en entrée il est facile de voir si A_0 est *admissible* ou non. L'explication sur la façon de vérifier qu'un ensemble est *admissible* est donnée dans la section 6.2.4.

Pour la suite de notre analyse, afin d'étudier la distribution des variables aléatoires C_k nous avons besoin de supposer que les variables aléatoires $(C_{X,k})_{x\in\mathcal{X}}$ sont indépendantes.

Hypothèse 6.1. *Soit A_0 un ensemble de différences en entrée admissible comme défini dans la définition 6.4. Soit $\mathcal{X}$ un ensemble de taille $N_{DC}/2$ qui vérifie l'équation (6.3). Alors, les variables aléatoires $(C_{x,k})_{x\in\mathcal{X}}$ sont indépendantes.*

Des expérimentations montrent que l'hypothèse précédente est souvent vraie.

En conséquence en supposant l'hypothèse 6.1 vérifiée nous obtenons que la distribution des variables aléatoires (C_k) est proche d'une loi de Poisson.

Lemme 6.2. *D'après le lemme précédent, si l'on se place sous l'hypothèse 6.1 nous avons que*

- *La distribution des variables aléatoires C_{k^*} est proche d'une loi de Poisson de paramètre $\frac{N_{DC}}{2}\sum_{i=1}^{\#A_0} p_*^{(i)}$.*
- *La distribution des variables aléatoires C_k pour $k \neq k^*$ est proche d'une loi de Poisson de paramètre $\frac{N_{DC}}{2}\sum_{i=1}^{\#A_0} p^{(i)}$.*

Nous introduisons ici les quantités suivantes qui vont jouer un rôle important dans la suite de notre analyse de la cryptanalyse différentielle multiple.

$$p_* \stackrel{\text{def}}{=} \frac{\sum_i p_*^{(i)}}{\#A_0} \quad \text{et} \quad p \stackrel{\text{def}}{=} \frac{\sum_i p^{(i)}}{\#A_0} \approx \frac{\#A \cdot 2^{-m}}{\#A_0}.$$

Ces notations sont utilisées dans la section 6.2.5

6.2.4 Comment vérifier qu'un ensemble de différences en entrée A_0 est *admissible*

Dans la section précédente, afin de pouvoir sommer des variables indépendantes, nous avons supposé que l'ensemble des différences en entrée vérifiait certaines propriétés . Pour

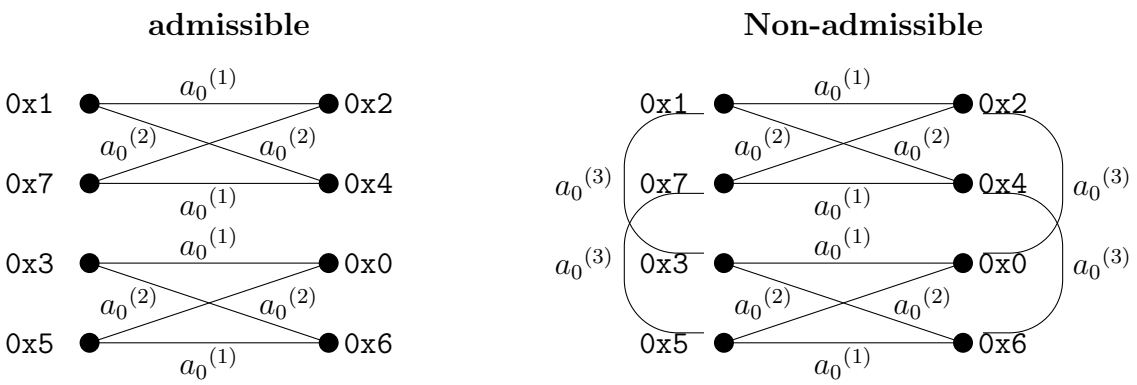

FIGURE 6.1 – Exemple de graphe : 1 biparti et l'autre non. ${a_0}^{(1)} = \texttt{0x3}$, ${a_0}^{(2)} = \texttt{0x5}$ et ${a_0}^{(3)} = \texttt{0x2}$. $\{{a_0}^{(1)}, {a_0}^{(2)}\}$ est admissible. $\{{a_0}^{(1)}, {a_0}^{(2)}, {a_0}^{(3)}\}$ n'est pas admissible.

autant, plus l'ensemble des différences en entrée est grand, plus il est difficile de vérifier facilement si cet ensemble est *admissible*. Nous présentons ici une méthode efficace pour vérifier si l'ensemble des différences en entrée est *admissible*.

Nous rappelons ici la définition d'un ensemble *admissible*. Un ensemble A_0 est admissible s'il existe un ensemble $\mathcal{X}$ de $N_{DC}/2$ messages clairs tel que $\forall {a_0}^{(i)} \in A_0$, $\forall x \in \mathcal{X}, x \oplus {a_0}^{(i)} \notin \mathcal{X}$.
Cette condition s'exprime par l'existence d'un graphe biparti [2]. Les sommets de ce graphe représentent l'ensemble des messages X et les arêtes l'ensemble des différences en entrée. La figure 6.1 montre sous forme de représentation d'un graphe deux ensembles de différences : un *admissible* (le graphe est biparti) l'autre non (le graphe n'est pas biparti).

L'existence d'un tel graphe est équivalente à la non-existence de cycles de poids impairs dans le graphe, c'est-à-dire que la somme d'un nombre impair de ${a_0}^{(i)}$ n'est jamais égale à 0.

Nous pouvons tester efficacement cette condition si nous redéfinissons le problème dans un contexte de théorie des codes. Soit M la matrice définie de la façon suivante : chaque colonne de M correspond à la décomposition binaire des différences dans A_0. Dire que chaque combinaison d'un nombre impair de colonnes est non-nulle est équivalent à dire que le dual du code engendré par la matrice M n'a que des mots de poids de Hamming pair. Cette condition est équivalente au fait que le code contient le vecteur tout à un. Elle peut se vérifier en un temps polynomial en utilisant une élimination gaussienne. Il suffit en effet de calculer la forme systématique du code, c'est-à-dire d'exprimer M sous forme systématique : $M' = (I||U)$ où I est la matrice identité.

Vérifier que A_0 est admissible peut alors se faire en vérifiant que

$$(1 \dots 1) \cdot U = (1 \dots 1).$$

6.2.5 Approximation des queues de la distribution des variables aléatoires C_k

Dans la section 6.2.3 nous avons montré que les variables aléatoires C_k avaient une distribution proche d'une loi de Poisson. D'après le théorème de Le Cam (voir théo-

2. En théorie des graphes, un graphe est dit biparti s'il existe une partition de son ensemble de sommets en deux sous-ensembles U et V telle que chaque arête ait une extrémité dans U et l'autre dans V.

rème 6.1), la borne d'erreur que nous avons en utilisant une approximation de Poisson est relativement petite (de l'ordre de 10^{-1}). Cet ordre de grandeur sur l'erreur reste cependant important si l'on regarde les queues de la fonction de répartition de la distribution des variables aléatoires C_k. Cette hypothèse, que l'erreur pour l'approximation des queues des variables aléatoires C_k est assez grande a été vérifiée expérimentalement. Une discussion sur ces expérimentations est faite dans la section 6.4.
Pour avoir une bonne estimation de la distribution des queues des variables aléatoires C_k nous avons utilisé un autre résultat qui nous donne une bonne estimation pour les queues de la distribution de ces variables aléatoires.

Théorème 6.2. *[Gal68, chapitre 5] Soit $C_k = \sum_x C_{x,k}$ une somme de M variables aléatoires discrètes, indépendantes et uniformément distribuées. Soit $s \mapsto \mu(s)$ le logarithme de la fonction génératrice des moments de chaque $C_{x,k}$. On note par μ' et μ'' les dérivées d'ordre 1 et 2 de μ. Alors, pour $s > 0$,*

$$P\left[C_k \geq \mu'(s)M\right] = e^{M[\mu(s)-s\mu'(s)]}\left[\frac{1}{|s|\sqrt{\pi 2M\mu''(s)}} + o\left(\frac{1}{\sqrt{M}}\right)\right].$$

En utilisant le résultat de ce théorème et en calculant le logarithme de la fonction génératrice des moments dans le cas particulier où les variables aléatoires $C_{a_0{}^{(i)},X,k}$ suivent des lois binomiales nous déduisons le théorème suivant.

Théorème 6.3. *Soit $C_k = \sum_x C_{X,k}$ une somme de $\frac{N_{DC}}{2}$ variables aléatoires discrètes, indépendantes et identiquement distribuées. Nous définissons les fonctions $G_-(\tau, q)$ et $G_+(\tau, q)$ pour τ et q des nombres réels dans $[0, 1]$ avec $\tau \neq q$ par :*

$$G_-(\tau, q) \stackrel{\text{def}}{=} e^{-ND(\tau||q)} \cdot \left[\frac{q\sqrt{(1-\tau)}}{(q-\tau)\sqrt{2\pi\tau N}} + \frac{1}{\sqrt{8\pi\tau N}}\right], \tag{6.4}$$

$$G_+(\tau, q) \stackrel{\text{def}}{=} e^{-ND(\tau||q)} \cdot \left[\frac{(1-q)\sqrt{\tau}}{(\tau-q)\sqrt{2\pi N(1-\tau)}} + \frac{1}{\sqrt{8\pi\tau N}}\right]. \tag{6.5}$$

Alors, les queues de la fonction de répartition des variables aléatoires C_k sont égales à :

$$\begin{aligned} P\left[C_k \leq \tau N\right] &= G_-(\tau, p)\left[1 + \mathcal{O}\left(\frac{p-\tau}{p}\right)\right], \\ P\left[C_k \geq \tau N\right] &= G_+(\tau, p)\left[1 + \mathcal{O}\left(\frac{p-\tau}{p}\right)\right]. \end{aligned}$$

Preuve : La preuve de ce théorème est compliquée et nécessite l'introduction de lemmes intermédiaires. La section 6.2.6 est dédiée à la preuve de ce théorème. □

6.2.6 Preuve du théorème 6.3

Afin de pouvoir utiliser les résultats du théorème 6.2 nous avons calculé le logarithme de la fonction génératrice des moments des $C_{X,k}$. Nous avons que

$$C_{X,k} = \sum_{i=1}^{\#A_0} C_{a_0{}^{(i)}X,k}.$$

où les variables aléatoires $C_{a_0{}^{(i)}X,k}$ suivent des lois de Bernoulli de paramètre $p^{(i)}$ ou $p_*^{(i)}$ suivant la valeur de k. Pour plus de simplicité, dans la suite de cette section nous notons par q_i la valeur de $p^{(i)}$ ou de $p_*^{(i)}$. Pour simplifier les notations nous notons par d le cardinal de l'espace des différences en entrée A_0 : $d = \#A_0$. Afin de prouver le théorème 6.3 nous introduisons les notations suivantes :

$$\bar{q} \stackrel{\text{def}}{=} \frac{\sum_{i=1}^{d} q_i}{d}, \quad m_2 \stackrel{\text{def}}{=} \frac{\sum_i q_i^2}{d}, \quad s_0 \stackrel{\text{def}}{=} \ln\left(\frac{\tau(1-\bar{q})}{\bar{q}(1-\tau)}\right) \tag{6.6}$$

Les autres notations utilisées sont celles du théorème 6.2.
Le logarithme de la fonction génératrice des moments des $C_{X,k}$ dans notre contexte vaut :

$$\mu(s) \;=\; \sum_{i=1}^{d} \ln\left(1 - q_i + q_i e^s\right).$$

Pour $s \notin \cup_i \{\log\left(\frac{1-q_i}{q_i}\right)\}$, ses dérivées sont alors égales à

$$\mu'(s) \;=\; \sum_{i=1}^{d} \frac{q_i e^s}{1 - q_i + q_i e^s}, \tag{6.7}$$

$$\mu''(s) \;=\; \sum_{i=1}^{d} \frac{q_i e^s (1-q_i)}{(1 - q_i + q_i e^s)^2}. \tag{6.8}$$

La fonction μ' est continue pour un voisinage de s_0. Soit s_r la valeur réelle telle que $\mu'(s_r) = d\tau$. Le but de cette preuve consiste à montrer que s_r est proche de s_0. D'après (6.8), on a

$$e^s = \frac{\mu'(s)}{\sum_{i=1}^{d} \frac{q_i}{1-q_i+q_i e^s}}$$

On note f la fonction suivante :

$$f(s) \stackrel{\text{def}}{=} \ln(d\tau) - \ln\left(\sum_{i=1}^{d} \frac{q_i}{1 - q_i + q_i e^s}\right).$$

On a $f(s_r) = s_r$.

Nous pouvons tout d'abord remarquer que $\forall\ s$ on a $\mu''(s) = \mu'(s)(1 - f'(s))$. En utilisant ce résultat sur s_r et le théorème 6.2 nous arrivons à la formule suivante :

$$P\left[C_k \geq d\tau N_{DC}/2\right] = e^{N_{DC}/2[\mu(s_r) - s_r d\tau]} \left[\frac{1}{|s_r|\sqrt{2\pi d\tau N_{DC}/2(1 - f'(s_r))}} + o\left(\frac{1}{\sqrt{N_{DC}/2}}\right)\right]. \tag{6.9}$$

Nous avons besoin de quantifier l'erreur faite en remplaçant s_0 par s_r dans l'équation (6.9) mais nous avons besoin dans un premier temps de calculer $f(s_0) - s_0$.

Lemme 6.3. *En utilisant les notations précédentes, on a*

$$f(s_0) - s_0 = \frac{\tau - \bar{q}}{\bar{q}^2} \cdot (\bar{q}^2 - m_2) + \mathsf{o}\left(\frac{\tau - \bar{q}}{\bar{q}^2} \cdot (\bar{q}^2 - m_2)\right).$$

Preuve : Nous pouvons d'abord extraire s_0 à partir de la formule.

$$\begin{aligned} f(s_0) &= \ln(d\tau) - \ln\left(\sum_{i=1}^{d} \frac{q_i}{1-q_i+q_i e^{s_0}}\right) \\ &= \ln\left(\frac{d\tau}{(1-\tau)\bar{q}}\right) - \ln\left(\sum_{i=1}^{d} \frac{q_i}{(1-q_i)(1-\tau)\bar{q}+q_i(1-\bar{q})\tau}\right) \\ &= \ln\left(\frac{(1-\bar{q})\tau}{(1-\tau)\bar{q}}\right) - \ln\left(\frac{1-\bar{q}}{d}\sum_{i=1}^{d} \frac{q_i}{(1-q_i)(1-\tau)\bar{q}+q_i(1-\bar{q})\tau}\right) \\ &= s_0 - \ln\left(\frac{1}{d}\sum_{i=1}^{d} q_i \cdot \frac{1-\bar{q}}{(1-q_i)(1-\tau)\bar{q}+q_i(1-\bar{q})\tau}\right). \end{aligned}$$

Nous obtenons alors

$$\begin{aligned} f(s_0)-s_0 &= -\ln\left(\frac{1}{d}\sum_{i=1}^{d} q_i \cdot \frac{1-\bar{q}}{q_i(\tau-\bar{q})+\bar{q}(1-\tau)}\right) = -\ln\left(\frac{1}{d}\sum_{i=1}^{d} \frac{q_i(1-\bar{q})}{\bar{q}(1-\tau)} \cdot \frac{1}{1+\frac{q_i(\tau-\bar{q})}{\bar{q}(1-\tau)}}\right) \\ &= -\ln\left(\frac{1}{d}\sum_{i=1}^{d} \frac{q_i}{\bar{q}}(1-\bar{q})[1+\tau+\mathrm{o}(\tau)]\left[1-\frac{q_i(\tau-\bar{q})}{\bar{q}(1-\tau)}+\mathrm{o}\left(\frac{q_i(\tau-\bar{q})}{\bar{q}}\right)\right]\right) \\ &= -\ln\left(\sum_{i=1}^{d} \frac{q_i}{d\bar{q}} + \sum_{i=1}^{d} \frac{\tau-\bar{q}}{d\bar{q}}\left[q_i - \frac{q_i^2}{\bar{q}}\right] + \mathrm{o}\left(\frac{\tau-\bar{q}}{d\bar{q}}\left[q_i-\frac{q_i^2}{\bar{q}}\right]\right)\right) \\ &= -\ln\left(1+\frac{\tau-\bar{q}}{\bar{q}^2}\cdot(\bar{q}^2-m_2)+\mathrm{o}\left(\frac{\tau-\bar{q}}{\bar{q}^2}\cdot(\bar{q}^2-m_2)\right)\right) \\ &= \frac{\tau-\bar{q}}{\bar{q}^2}\cdot(\bar{q}^2-m_2)+\mathrm{o}\left(\frac{\tau-\bar{q}}{\bar{q}^2}\cdot(\bar{q}^2-m_2)\right). \end{aligned}$$

□

Lemme 6.4. *En utilisant les notations précédentes nous avons*

$$s_r = s_0 + \mathcal{O}\left(\frac{\tau-\bar{q}}{\bar{q}^2}\cdot(\bar{q}^2-m_2)\right) \quad \text{et} \quad f'(s_0) = \tau\frac{m_2}{\bar{q}^2} + \mathrm{o}\left(\tau\frac{m_2}{\bar{q}^2}\right).$$

Preuve : Le développement en série de Taylor de f est

$$f(s_r) = f(s_0) + (s_r - s_0)f'(s_0) + \mathcal{O}\left(f''(s_0)(s_r-s_0)^2\right).$$

Donc, comme $f(s_r) = s_r$, nous avons $s_r = s_0 + \mathcal{O}\left(\frac{f(s_0)-s_0}{1-f'(s_0)}\right)$.

Par définition, $f'(s_0) = \sum_{i=1}^{d} \frac{q_i^2 e^{s_0}}{(1-q_i+q_i e^{s_0})^2} \cdot \left[\sum_{i=1}^{d} \frac{q_i}{1-q_i+q_i e^{s_0}}\right]^{-1}$ et $e^{s_0} = \tau/\bar{q} + \mathrm{o}(\tau/\bar{q})$. Donc,

$$\begin{aligned} f'(s_0) &= \left[\sum_{i=1}^{d} q_i^2 e^{s_0}(1+\mathrm{o}(1))\right] \cdot \left[\sum_{i=1}^{d} q_i(1-\mathrm{o}(1))\right]^{-1} \\ &= \left[\frac{d\tau}{\bar{q}}m_2 + \mathrm{o}\left(\frac{d\tau}{\bar{q}}m_2\right)\right] \cdot (d\bar{q})^{-1}\Big[1+\mathrm{o}(1)\Big]. \end{aligned}$$

Ce résultat donne $f'(s_0) = \tau\frac{m_2}{\bar{q}^2} + \mathtt{o}\left(\tau\frac{m_2}{\bar{q}^2}\right)$. En utilisant le fait que $\frac{1}{1-f'(s_0)} = \mathcal{O}(1)$ et le lemme 6.3, nous obtenons que

$$s_r = s_0 + \mathcal{O}\left(\frac{f(s_0)-s_0}{1-f'(s_0)}\right) = s_0 + \mathcal{O}\left(\frac{\tau-\bar{q}}{\bar{q}^2}\cdot(\bar{q}^2-m_2)\right).$$

□

Lemme 6.5. *En utilisant les notations précédentes nous avons*

$$\begin{aligned}\mu(s_r) &= d\ln\left(\frac{1-\bar{q}}{1-\tau}\right) + \mathcal{O}\left(d\frac{(\tau-\bar{q})}{\bar{q}^2}\cdot(\bar{q}^2-m_2)\max(\tau-\bar{q},\tau)\right),\\ 1-f'(s_r) &= 1-\tau+\mathcal{O}\left(\frac{\max(\tau-\bar{q},\tau)}{\bar{q}^2}(\bar{q}^2-m_2)\right).\end{aligned}$$

Preuve : En utilisant le lemme 6.4 nous avons

$$e^{s_r} = e^{s_0}\times e^{\mathcal{O}\left(\frac{\tau-\bar{q}}{\bar{q}^2}\cdot(\bar{q}^2-m_2)\right)} = e^{s_0}\left[1+\mathcal{O}\left(\frac{\tau-\bar{q}}{\bar{q}^2}\cdot(\bar{q}^2-m_2)\right)\right].$$

Donc,

$$\begin{aligned}\mu(s_r) = \sum_{i=1}^{d}\ln\left(1-q_i+q_ie^{s_r}\right) &= \sum_{i=1}^{d}\ln\left(1-q_i+q_ie^{s_0}\left[1+\mathcal{O}\left(\frac{\tau-\bar{q}}{\bar{q}^2}\cdot(\bar{q}^2-m_2)\right)\right]\right)\\ &= \sum_{i=1}^{d}\ln\left(\left[1-q_i+q_ie^{s_0}\right]\left[1+\mathcal{O}\left(\frac{q_i\,\tau}{\bar{q}}\frac{\tau-\bar{q}}{\bar{q}^2}\cdot(\bar{q}^2-m_2)\right)\right]\right)\\ &= \sum_{i=1}^{d}\ln\left(1-q_i+q_ie^{s_0}\right)+\mathcal{O}\left(d\tau\cdot\frac{\tau-\bar{q}}{\bar{q}^2}\cdot(\bar{q}^2-m_2)\right).\end{aligned}$$

Et finalement, $\mu(s_r) = \mu(s_0) + \mathcal{O}\left(d\tau\cdot\frac{\tau-\bar{q}}{\bar{q}^2}\cdot(\bar{q}^2-m_2)\right)$. Cependant,

$$\begin{aligned}\mu(s_0) &= \sum_{i=1}^{d}\ln\left(1-q_i+q_ie^{s_0}\right)\\ &= \sum_{i=1}^{d}\ln\left((1-q_i)\bar{q}(1-\tau)+q_i(1-\bar{q})\tau\right)-d\ln\left(\bar{q}(1-\tau)\right)\\ &= \sum_{i=1}^{d}\ln\left(\bar{q}-q_i\bar{q}-\bar{q}\tau+q_i\tau\right)-d\ln\left(\bar{q}(1-\tau)\right).\end{aligned}$$

Et,

$$\begin{aligned}
\mu(s_0) &= \sum_{i=1}^{d} \ln\left(\frac{\bar{q}(1-\tau)+q_i(\tau-\bar{q})}{(1-\bar{q})\bar{q}}\right) - d\ln\left(\frac{1-\tau}{1-\bar{q}}\right) \\
&= d\ln\left(\frac{1-\bar{q}}{1-\tau}\right) + \sum_{i=1}^{d} \ln\left(1+\frac{(q_i-\bar{q})(\tau-\bar{q})}{(1-\bar{q})\bar{q}}\right) \\
&= d\ln\left(\frac{1-\bar{q}}{1-\tau}\right) + \sum_{i=1}^{d} \frac{(q_i-\bar{q})(\tau-\bar{q})}{(1-\bar{q})\bar{q}} + \mathcal{O}\left(d\frac{(\tau-\bar{q})^2}{\bar{q}^2}\cdot(\bar{q}^2-m_2)\right) \\
&= d\ln\left(\frac{1-\bar{q}}{1-\tau}\right) + \mathcal{O}\left(d\frac{(\tau-\bar{q})^2}{\bar{q}^2}\cdot(\bar{q}^2-m_2)\right).
\end{aligned}$$

Donc $\mu(s_r) = d\ln\left(\frac{1-\bar{q}}{1-\tau}\right) + \mathcal{O}\left(d\frac{(\tau-\bar{q})}{\bar{q}^2}\cdot(\bar{q}^2-m_2)\max(\tau-\bar{q},\tau)\right)$. La seconde partie du lemme est donnée par un développement en série de Taylor de $f'(s_r)$:

$$f'(s_r) = f'(s_0) + \mathcal{O}(s_0-s_r) = \tau\frac{m_2}{\bar{q}^2} + \mathcal{O}\left(\frac{\tau-\bar{q}}{\bar{q}^2}(\bar{q}^2-m_2)\right).$$

Donc

$$\begin{aligned}
1-f'(s_r) &= (1-\tau)\left[1+\mathcal{O}\left(\frac{\tau(\bar{q}^2-m_2)}{\bar{q}^2(1-\tau)}\right)\right] + \mathcal{O}\left(\frac{\tau-\bar{q}}{\bar{q}^2}(\bar{q}^2-m_2)\right) \\
&= (1-\tau) + \mathcal{O}\left(\frac{\max(\tau-\bar{q},\tau)}{\bar{q}^2}(\bar{q}^2-m_2)\right).
\end{aligned}$$

□

Preuve du théorème 6.3

A partir de l'équation (6.9), du lemme 6.4 et du lemme 6.5, nous donnons une preuve du théorème 6.3. Dans cette partie nous supposons que τ est plus grand que $\bar{q}$. Nous considérons d'abord le terme exponentiel.

$$\begin{aligned}
e^{N_{DC}/2[\mu(s_r)-s_r d\tau]} &= \exp\left[N_{DC}/2\, d\ln\left(\frac{1-\bar{q}}{1-\tau}\right) - N_{DC}/2\,\ln\left(\frac{\tau(1-\bar{q})}{\bar{q}(1-\tau)}\right)d\tau\right. \\
&\qquad \left. + \mathcal{O}\left(N_{DC}/2\, d\tau\frac{\tau-\bar{q}}{\bar{q}^2}(\bar{q}^2-m_2)\right)\right] \\
&= e^{-ND(\tau\|\bar{q})}\left[1+\mathcal{O}\left(N\tau\frac{\tau-\bar{q}}{\bar{q}^2}(\bar{q}^2-m_2)\right)\right]. \qquad (6.10)
\end{aligned}$$

Puis nous considérons le terme polynomial

$$\begin{aligned}
\left[|s_r|\sqrt{2\pi\tau N(1-f'(s_r))}\right]^{-1} &= \left[|s_0| + \mathcal{O}\left(\frac{\tau-\bar{q}}{\bar{q}^2}(\bar{q}^2-m_2)\right)\right]^{-1} \\
&\times \left[\sqrt{2\pi\tau N}\sqrt{1-\tau+\mathcal{O}\left(\frac{\tau}{\bar{q}^2}(\bar{q}^2-m_2)\right)}\right]^{-1} \\
&= \left[s_0\sqrt{2\pi\tau N(1-\tau)}\left[1+\mathcal{O}\left(\frac{\tau}{\bar{q}^2}(\bar{q}^2-m_2)\right)\right]\right]^{-1}.
\end{aligned}$$

Comme nous avons supposé que τ est plus grand que $\bar{q}$, nous avons que s_0 est positif, et donc

$$s_0 \stackrel{\text{def}}{=} -\ln\left(\frac{\bar{q}(1-\tau)}{\tau(1-\bar{q})}\right) = -\ln\left(1-\frac{\tau-\bar{q}}{\tau(1-\bar{q})}\right) = \frac{\tau-\bar{q}}{\tau(1-\bar{q})}\left[1+\frac{\tau-\bar{q}}{2\tau(1-\bar{q})}+\mathrm{o}\left(\frac{\tau-\bar{q}}{\tau}\right)\right].$$

Donc

$$\begin{aligned}\left[s_r\sqrt{2\pi\tau N(1-f'(s_r))}\right]^{-1} &= \frac{1+\mathcal{O}\left(\frac{\tau}{\bar{q}^2}(\bar{q}^2-m_2)\right)}{s_0\sqrt{2\pi\tau N(1-\tau)}} \\ &= \frac{\tau(1-\bar{q})}{(\tau-\bar{q})\sqrt{2\pi\tau N(1-\tau)}}\cdot\left[1+\frac{(\tau-\bar{q})}{2\tau(1-\bar{q})}+\mathrm{o}\left(\frac{\tau-\bar{q}}{\tau}\right)\right] \\ &= \frac{\sqrt{\tau}(1-\bar{q})}{(\tau-\bar{q})\sqrt{2\pi N(1-\tau)}}+\frac{1}{\sqrt{8\pi T}}+\mathrm{o}\left(\frac{1}{\sqrt{T}}\right). \qquad (6.11)\end{aligned}$$

Pour conclure, nous remplaçons les termes (6.10) et (6.11) dans (6.9) et nous obtenons

$$\begin{aligned}P\left[C_k \geq \tau N\right] &= \left[\frac{\sqrt{\tau}(1-\bar{q})}{(\tau-\bar{q})\sqrt{2\pi N(1-\tau)}}+\frac{1}{\sqrt{8\pi T}}+\mathrm{o}\left(\frac{1}{\sqrt{T}}+\frac{1}{\sqrt{N_{DC}/2}}\right)\right] \\ &\times\ e^{-ND(\tau||\bar{q})}\cdot\left[1+\mathcal{O}\left(N\tau\frac{\tau-\bar{q}}{\bar{q}^2}(\bar{q}^2-m_2)\right)\right] \\ &= e^{-ND(\tau||\bar{q})}\left[\frac{\sqrt{\tau}(1-\bar{q})}{(\tau-\bar{q})\sqrt{2\pi N(1-\tau)}}+\frac{1}{\sqrt{8\pi T}}+\mathrm{o}\left(\frac{1}{\sqrt{T}}\right)\right].\end{aligned}$$

La formule pour $P\left[C_k \leq \tau N\right]$ s'obtient de la même façon en supposant cette fois ci que $\tau \leq \bar{q}$.

6.2.7 Distribution des variables aléatoires C_k

En combinant les résultats du théorème 6.1 avec ceux du théorème 6.3 nous proposons une définition pour la fonction de répartition de variables aléatoires C_k

Définition 6.5. *Soit $G_{Poisson}(\tau, q)$ la fonction de répartition de la loi de Poisson avec paramètre qN. Soit $G_-(\tau, q)$ et $G_+(\tau, q)$ les deux queues de distributions définies dans le théorème 6.3. Nous définissons la fonction $G(\tau, q)$ par*

$$G(\tau,q) \stackrel{\text{def}}{=} \begin{cases} G_-(\tau,q) & si\ \tau < q-3\cdot\sqrt{q/N}, \\ 1-G_+(\tau,q) & si\ \tau > q+3\cdot\sqrt{q/N}, \\ G_{Poisson}(\tau,q) & sinon. \end{cases}$$

Proposition 6.1. *Soit $G(\tau, q)$ la fonction définie dans la définition 6.5. Soit p_* et p les paramètres de l'attaque :*

$$p_* = \frac{\sum_{i,j} p_*^{(i,j)}}{\#A_0} \text{ et } p \approx \frac{\#A}{2^m \#A_0}$$

Nous proposons les estimations suivantes de la fonction de répartition des variables aléatoires C_k et C_{k^*} :

– *La fonction de répartition correspondant à la variable aléatoire C_{k^*} est proche de*

$$G_*(\tau) \stackrel{\text{def}}{=} G(\tau, p_*).$$

– *La fonction de répartition correspondant à la variable aléatoire C_k est proche de*

$$G(\tau) \stackrel{\text{def}}{=} G(\tau, p).$$

6.3 Complexité en données et probabilité de succès

6.3.1 La complexité en données

Pour le calcul de la complexité en données d'une attaque différentielle multiple, nous supposons que l'ensemble A_0 est admissible et que l'hypothèse 6.1 est vérifiée. L'expression de la fonction de répartition des variables aléatoires est déterminée par la proposition 6.1. Pour le calcul de la complexité en données nous utilisons le contexte du test d'hypothèses (voir section 5.3.1) Dans la section 5.3.5, afin de calculer la complexité en données de l'attaque nous avons fixé le seuil τ à p_*, c'est-à-dire que nous calculons une approximation de la complexité en données pour une probabilité de succès proche de 50%. Il s'avère que dans le cas où p_* est suffisant loin de p ($p_* > p+3\cdot\sqrt{p/N}$), on se place dans le contexte où la fonction de répartition des variables aléatoire C_k est proche de la fonction de répartition définie dans la proposition 6.1. La fonction $1-G_+(\tau, q)$ est similaire à celle que nous avons utilisée dans la section 5.3. En utilisant la même méthode que dans la section 5.3, nous obtenons une estimation de la complexité en données.

Théorème 6.4. *Soit ℓ la taille de la liste des clés gardées. Soit 2^n le nombre de clés candidates qui sont testées dans notre attaque. Pour une probabilité de succès proche de 0.5, la complexité en données d'une attaque différentielle multiple quand les variables aléatoires étudiées correspondent à la somme des variables aléatoires simples est :*

$$N = -2 \cdot \frac{\ln(2\sqrt{\pi}\ell/2^n)}{\#A_0 D(p_*||p)}.$$

Preuve : Dans la section 5.3, nous avons étudié la complexité en données d'une attaque statistique simple. Dans la preuve du théorème 5.2, afin de déterminer la complexité en données d'une attaque statistique simple, nous avons approché la queue de la loi binomiale par

$$e^{-ND(\tau||p)} \frac{(1-p)\sqrt{\tau}}{(\tau-p)\sqrt{2\pi N(1-\tau)}}. \tag{6.12}$$

Dans le théorème 6.3 nous avons montré que la queue de la fonction de distribution des variables aléatoires C_k (voir définition 6.3) est égale à

$$P\left[C_k \geq \tau N\right] \underset{N\to\infty}{\sim} e^{-ND(\tau||q)} \cdot \left[\frac{(1-q)\sqrt{\tau}}{(\tau-q)\sqrt{2\pi N(1-\tau)}} + \frac{1}{\sqrt{8\pi\tau N}}\right]. \tag{6.13}$$

Nous utilisons donc ici la même méthode pour déterminer le nombre d'échantillons d'une attaque différentielle multiple dans le cas où la statistique étudiée correspond à la somme des variables aléatoires. Pour les mêmes raisons que celles spécifiées dans la

preuve du théorème 5.2, nous pouvons fixer le seuil relatif τ à p_* [3]. Sous cette condition, le nombre d'échantillons N peut alors être trouvé en résolvant l'équation en N

$$e^{-ND(\tau||p)} \cdot \left[\frac{(1-p)\sqrt{\tau}}{(\tau-p)\sqrt{2\pi N(1-\tau)}} + \frac{1}{\sqrt{8\pi\tau N}} \right] = \frac{\ell}{2^n}.$$

Dans les attaques différentielles multiples on a $p_* \geq p + 3\sqrt{p/N}$.

De la même façon que pour la preuve du théorème 5.2 une bonne estimation de N peut être trouvée en utilisant une méthode du point fixe pour l'équation :

$$e^{-ND(p_*||p)} \cdot \left[\frac{(1-p)\sqrt{p_*}}{(p_*-p)\sqrt{2\pi N(1-p_*)}} + \frac{1}{\sqrt{8\pi\tau N}} \right] = \frac{\ell}{2^n}.$$

Comme pour la preuve de théorème 5.2, on obtient que N est alors proche de

$$-\frac{1}{D(p_*||p)} \left[\ln\left(\frac{\nu\ell/2^n}{\sqrt{D(p_*||p)}} \right) + 0.5 \ln(-\ln(\nu\ell/2^n)) \right] \tag{6.14}$$

$$\text{où } \nu \stackrel{\text{def}}{=} \frac{(p_*-p)\sqrt{8\pi(1-p_*)p_*}}{2p_*(1-p) + (p_*-p)\sqrt{1-p_*}}.$$

Le lemme 5.6 nous dit que $\ln(2\sqrt{\pi D(p_*||p)})$ est une bonne estimation de $\ln(\nu)$. Ce qui implique que N est proche de

$$-\frac{\ln(2\sqrt{\pi}\frac{\ell}{2^n})}{D(p_*||p)}.$$

On complète la preuve en remarquant que le nombre de messages clairs $N_{DC} = \frac{2N}{\#A_0}$. □

6.3.2 La probabilité de succès

Dans le théorème 6.5 ci-après, nous donnons une formule de la probabilité de succès d'une attaque différentielle multiple dans le cas où la statistique étudiée correspond à la somme des variables aléatoires simples. Ce résultat est prouvé en utilisant des arguments similaires à ceux utilisés dans la preuve du théorème 5.3.

Théorème 6.5. *Soit $G_*(x)$ (resp. $G(x)$) l'estimation de la fonction de répartition des variables aléatoires C_{k^*} (resp. C_k) comme définie dans la proposition 6.1. La probabilité de succès, P_S, d'une cryptanalyse différentielle multiple où la statistique étudiée correspond à la somme des variables aléatoires simples est donnée par*

$$P_S \approx 1 - G_* \left[G^{-1}\left(1 - \frac{\ell-1}{2^n-2}\right) - 1 \right] \tag{6.15}$$

où la pseudo-inverse de G est définie par $G^{-1}(y) = \min\{x | G(x) \geq y\}$.

3. On rappelle ici que cette condition nous donne une probabilité de succès proche de 0.5.

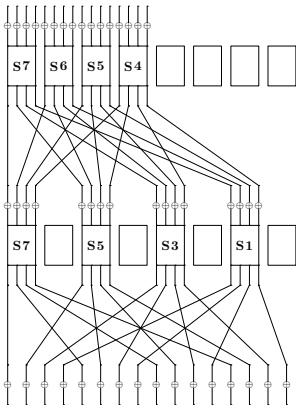

FIGURE 6.2 – Diffusion sur 2 tours de SMALLPRESENT-[8].

6.4 Validation expérimentale

Dans cette section nous validons expérimentalement les résultats théoriques présentés dans la section 6.3. Dans le but de valider expérimentalement la formule de la probabilité de succès donnée dans le théorème 6.5 nous faisons une attaque différentielle multiple sur 11 tours de la version réduite de PRESENT appelé SMALLPRESENT-[8] . Une description de PRESENT et SMALLPRESENT-[8] est faite dans la section 1.4.1 [4].

6.4.1 Description de l'attaque

L'attaque que nous présentons ici utilise des caractéristiques différentielles sur 9 tours de SMALLPRESENT-[8] et a pour but de retrouver de l'information sur les clés des deux derniers tours. Cela donne alors une attaque sur 11 tours.

Dans le but d'estimer empiriquement la probabilité de succès, nous avons itéré 250 fois cette attaque pour l'algorithme de cadencement de clé utilisant des clés de 80 bits ou des clés de 40 (voir section 1.4.1). Pour limiter la complexité en temps de l'attaque nous avons limité le nombre de bit de clés à retrouver à 32 bits (pour les 2 derniers tours). Dans ce but nous avons pris des différences après 9 tours de la forme `0x????0000`. Cette structure nous permet dans le cas de SMALLPRESENT-[8] de retrouver 16 bits des deux dernières clés. La figure 6.2 montre la diffusion sur 2 tours du système de chiffrement quand la différence en sortie est `0x????0000`.

Il apparaît clairement que les meilleures différentielles s'obtiennent quand il n'y a qu'une boîte-S active au premier tour. Pour cette attaque expérimentale nous avons décidé de réduire l'ensemble des différences en entrée à une seule boîte-S.

L'ensemble des différences en entrée que nous avons pris est alors le suivant :

$$A_0 = \{\texttt{0x3}, \texttt{0x5}, \texttt{0x7}, \texttt{0xB}, \texttt{0xD}, \texttt{0xF}\}.$$

Il est assez facile de vérifier que cet ensemble est admissible car nous pouvons par exemple séparer l'ensemble des messages clairs en les messages pairs et les messages impairs. Notre

4. On rappelle que SMALLPRESENT-[8] est la version pour chiffrer des messages de 32 bits.

attaque utilise 55 différentielles sur 9 tours de SMALLPRESENT-[8]. La probabilité de chaque différentielle pour les deux algorithmes de cadencement de clés (celui avec 40 bits et celui avec 80 bits) a été estimée par une moyenne sur 250 clés. Les 55 différentielles ainsi que l'estimation de leurs probabilités sont données dans le tableau A.1 qui se trouve en section A.1. Pour chacune de nos attaques, nous avons décidé de garder une liste $\mathcal{L}$ de taille $\ell = 2^{12}$ clés les plus probables pour les deux derniers tours.

6.4.2 Analyse des résultats expérimentaux

Soit $H_*(x)$ et $H(x)$ les estimations de la fonction de répartition des variables aléatoires C_{k^*} et C_k. La probabilité de succès théorique d'une attaque est

$$P_S = 1 - H_* \left[H^{-1} \left(1 - \frac{\ell - 1}{2^n - 2} \right) - 1 \right],$$

H et H_* variant en fonction des variables aléatoires étudiées. Dans ce chapitre nous avons montré que les variables aléatoires C_k suivaient une distribution hybride, c'est-à-dire que les fonctions H et H_* étaient égales aux fonctions G et G_* définies dans la proposition 6.1. Cependant pour valider cette théorie vous avons comparé notre formule de la probabilité de succès avec d'autres formules utilisant d'autres fonctions de répartition.

Dans les figures 6.3 et 6.4, nous comparons la probabilité de succès expérimentale obtenue grâce à une moyenne sur 250 expérimentations avec différentes formules de probabilités de succès théorique. Comme nous l'avons dit dans la section 5.4, la formule de la probabilité de succès donnée par Ali Aydin Selçuk est sensiblement la même que la notre. La différence réside principalement en l'estimation de la distribution des variables aléatoires.

Ainsi dans les figures 6.4 et 6.4 nous représentons la probabilité de succès dans le cas où les fonctions G et G_* sont les fonctions de répartition :

- d'une loi Normale (Formule proche de celle de Ali Aydin Selçuk [Sel08]) ;
- d'une loi de Poisson ;
- de la loi hybride définie par sa fonction de répartie dans la proposition 6.1.

6.4.3 Commentaires sur les figures 6.3 et 6.4

Si on analyse les figures 6.3 et 6.4, on remarque qu'il est clair que l'utilisation de l'approximation normale pour analyser la probabilité de succès d'une attaque différentielle multiple n'est pas bonne. D'ailleurs cette remarque a déjà été faite par Ali Aydin Selçuk dans son article [Sel08] pour le cas de la cryptanalyse différentielle simple.
Dans la cryptanalyse différentielle simple, la loi de Poisson est une bonne approximation de la distribution des variables aléatoires C_{k^*} et C_k. Pour autant comme le montrent les résultats, dans le cas de la cryptanalyse différentielle multiple cette approximation est moins bonne que l'approximation hybride que nous définissons dans la définition 6.5.

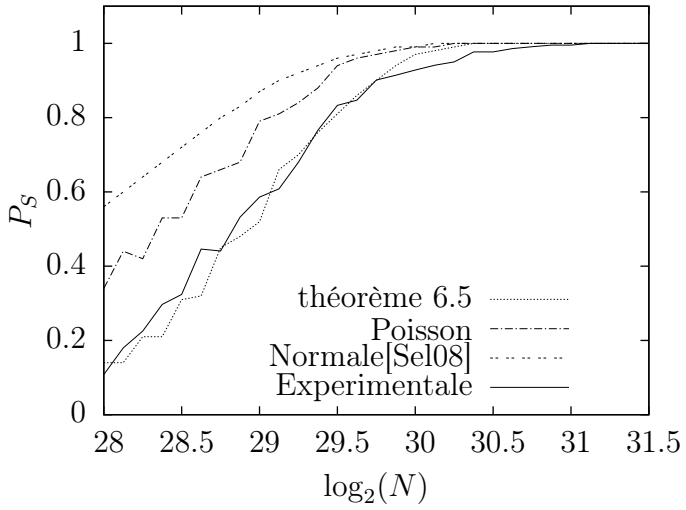

FIGURE 6.3 – Comparaison des différentes formules de la probabilité de succès avec la valeur expérimentale de celle-ci. (Expériences faites avec l'algorithme de cadencement de clé de 40 bits))

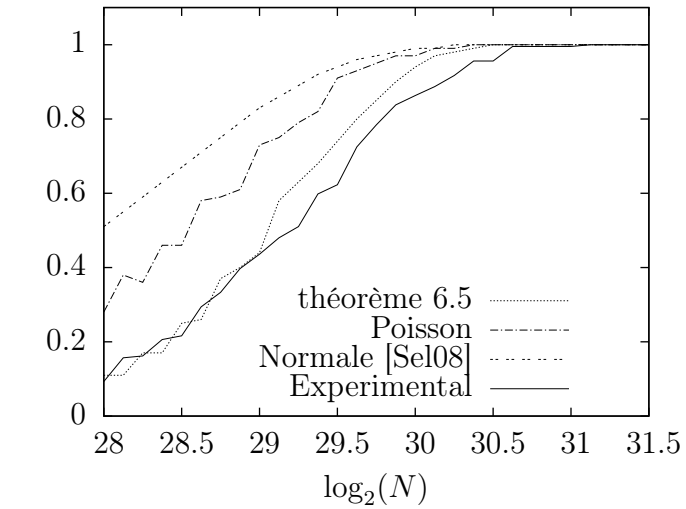

FIGURE 6.4 – Comparaison des différentes formules de la probabilité de succès avec la valeur expérimentale de celle-ci. (Expériences faites avec l'algorithme de cadencement de clé de 80 bits))

6.4.4 Validation de la formule de la probabilité de succès

Quand le rapport $\frac{\ell-1}{2^n-2}$ est petit, la précision du calcul de $G^{-1}\left(1-\frac{\ell-1}{2^n-2}\right)$ dépend essentiellement d'une bonne approximation de l'estimation de la queue de distribution des variables aléatoires. Or la queue de la loi de Poisson ne constitue pas une bonne approximation de la distribution des variables aléatoires. D'où l'importance de notre approche hybride qui est la plus pertinente ici. Nos résultats expérimentaux justifient l'utilisation de cette approche. En effet, comme dans la figure 6.3, la courbe représentant la probabilité de succès expérimentale et celle représentant la probabilité de succès théorique obtenue en utilisant notre approximation de la probabilité de succès donnée dans le théorème 6.5 sont proches, on peut estimer que notre approximation de la distribution des variables aléatoires est correcte.

6.4.5 Validation de la formule de la complexité en données

En utilisant les mêmes expérimentations, nous pouvons aussi confirmer la pertinence du théorème 6.4. En effet, dans le chapitre 5 nous avons conjecturé (voir section 5.4.6) que prendre N de la forme

$$N = -2 \cdot c \cdot \frac{\ln(2\sqrt{\pi}\ell/2^n)}{\#A_0 D\left(p_* || p\right)} \tag{6.16}$$

conduit à une probabilité de succès de 50% pour $c = 1$, 80% pour $c = 1.5$ et de 90% pour $c = 2$. Dans la tableau 6.2, nous donnons les valeurs de la probabilité de succès empirique pour ces différentes valeurs de N. Ces calculs ont été faits pour des clés maîtres de 40 bits et de 80 bits.

TABLE 6.2 – Probabilité de succès empirique calculée à partir de N donné par (6.16).

	$c = 1.0$		$c = 1.5$		$c = 2.0$	
taille de la clé	40-bit	80-bit	40-bit	80-bit	40-bit	80-bit
N	$2^{28.92}$	$2^{29.06}$	$2^{29.50}$	$2^{29.65}$	$2^{29.92}$	$2^{30.06}$
P_S	0.55	0.47	0.83	0.75	0.92	0.88

6.5 Attaque sur 18 tours de PRESENT

Dans les sections précédentes nous avons décrit le principe de la cryptanalyse différentielle multiple et étudié les différentes complexités de celle-ci. Comme il est précisé dans la partie expérimentale (voir section 6.4) la formule de la probabilité de succès de l'attaque est bonne lorsque l'on a une bonne estimation de la probabilité des différentielles. Nous avons donc voulu appliquer cette approche en effectuant une cryptanalyse d'une version réduite du système de chiffrement PRESENT (voir section 1.4.1). Depuis 2008 il y a eu un certain nombre de cryptanalyses de ce système de chiffrement. Parmi ces attaques on peut notamment citer l'attaque faite par Meiqin Wang dans [Wan08]. Cette cryptanalyse sur 16 tours permet de retrouver de l'information sur la clé des 2 derniers tours (c'est le même principe que celui utilisé pour notre attaque expérimentale sur SMALLPRESENT-[8]). Pour faire cette attaque Meiqin Wang utilise de l'information venant de 24 différentielles.

Afin d'estimer la probabilité de chacune de ces différentielles sur 4 tours, Meiqin Wang étudie la probabilité du meilleur chemin sur 4 tours avec la même différence en entrée et en sortie pour pouvoir les itérer facilement. En utilisant cette méthode, elle obtient une borne inférieure sur la probabilité des différentielles qu'elle utilise. Nous détaillons certains problèmes que nous avons relevés dans l'attaque de Meiqin Wang.

Quelques remarques sur l'attaque faite par Meiqin Wang

Une des différentielles sur 14 tours de PRESENT utilisées par Meiqin Wang dans son attaque est la suivante :

$$(d_0, d_{14}) = (\texttt{0x0700000000000700}, \texttt{0x0000000900000009}).$$

La probabilité de cette différentielle est obtenue en itérant 3 fois un chemin différentiel sur 4 tours et en ajoutant 1 tour au début et à la fin. En utilisant notre algorithme de "branch and bound", nous avons quelques remarques à propos de ce chemin :

- Le chemin utilisé sur 4 tours n'est pas le meilleur chemin sur 4 tours. En effet le meilleur chemin sur 4 tours a une probabilité 2^{-12}. Le chemin choisi par Meiqin Wang est le suivant :

  $$(\texttt{0x4004}, \texttt{0x900000009}, \texttt{0x10100000000}, \texttt{0x200000000000500}, \texttt{0x4004}).$$

  La probabilité théorique de ce chemin est 2^{-18}.
  Le choix de Meiqin Wang pour ce chemin peut être justifié par le fait que nous avons remarqué que ce chemin est le meilleur chemin qui peut être itéré(c'est-à-dire les différences en entrée et en sortie après 4 tours sont les mêmes)
- En utilisant notre algorithme de recherche automatique de chemins, nous avons vu qu'il existe beaucoup de chemins sur 14 tours avec probabilité 2^{-62}. Bien sûr nous n'avons pas pu faire une recherche exhaustive sur les $2^{64} - 1$ différences en entrée. Mais en supposant que les meilleurs chemins différentiels ont peu de boîtes-S actives nous pouvons conjecturer que 2^{-62} semble être la meilleure probabilité pour un chemin sur 14 tours. En utilisant cette conjecture, nous pouvons dire que les chemins différentiels utilisés par Meiqin Wang ont la meilleure probabilité possible.
- En utilisant l'algorithme de "branch and bound", nous avons pu déterminer tous les chemins avec différence en entrée d_0 et différence en sortie d_{14} qui ont une probabilité théorique supérieure à 2^{-73}. En sommant la probabilité de ces chemins (voir section 4.2) nous avons pu observer que la probabilité de la différentielle (d_0, d_{14}) est supérieure à $2^{-57.53}$. Au regard de ce résultat la complexité de l'attaque faite par Meiqin Wang est plus petite que la complexité qu'elle nous donne.

Dans cette section nous présentons une attaque sur 18 tours de PRESENT. Pour effectuer cette attaque nous avons calculé la probabilité de chaque différentielle à l'aide de l'algorithme "branch and bound" décrit dans la section 4.1.2. En prenant la somme des chemins ayant une probabilité supérieure à 2^{-90} et un nombre maximal de boîtes-S actives par tour égal à 3, nous avons trouvé une borne inférieure sur la probabilité d'un certain nombre de différentielles sur 16 tours. Parmi les meilleures différentielles trouvées, nous avons gardé les différentielles pour lesquelles l'ensemble de différences en entrée est "admissible". Les différentielles que nous avons utilisées sont détaillées dans le tableau 6.3.

a_0	a_r	$\log(P[a_0 \to a_r])$
0x1001	0x4040400400000000	-62.21
0x1001	0x40400000000	-62.58
0x1001	0x4000404000000000	-62.84
0x1001	0x4040400000000	-62.84
0x100100000000	0x4040404000000000	-62.97
0x4004	0x4040404000000000	-62.99
0x10010000	0x4040404000000000	-63.13
0x400c	0x4040404000000000	-63.16
0xc004	0x4040404000000000	-63.16
0xc00c	0x4040404000000000	-63.16
0x2002	0x4040404000000000	-63.17
0x1008	0x4040404000000000	-63.21
0x100e	0x4040404000000000	-63.21
0x101	0x4040404000000000	-63.29
0x11	0x4040404000000000	-63.29
0x100100000000	0x40400000000	-63.35
0x200a	0x4040404000000000	-63.37
0xa002	0x4040404000000000	-63.37
0xa00a	0x4040404000000000	-63.37
0x4004	0x40400000000	-63.39
0x1001	0x400400000000	-63.40
0x2004	0x4040404000000000	-63.45
0x4002	0x4040404000000000	-63.45

TABLE 6.3 – Les différentielles utilisées dans l'attaque différentielle multiple sur PRESENT.

Pour la cryptanalyse que nous avons effectuée, nous avons utilisé 23 différentielles sur 16 tours. Ces différentielles peuvent être regroupées par rapport à $\#A_0 = 16$ différences en entrée

Pour chacune des différences en entrée, l'ensemble des différences en sortie varie. Ainsi l'ensemble des différences en sortie est inclus dans l'ensemble

$$A_r \in \{\texttt{0x0?0?0?0?00000000}\}.$$

Il s'avère que dans notre cas le nombre de différences en sortie pour chaque différence en entrée varie. Il est le plus important pour la différence en entrée `0x1001`. Pour cette entrée on a $|A_r| = 5$. En utilisant cet ensemble de différence en sortie, nous obtenons des cribles similaires pour chaque différence en entrée après 18 tours du système de chiffrement. La taille de chacun de ces cribles est proche de 2^{32}.

$$\#\Delta_{sieve}^{(i)} \approx 2^{32}.$$

Dans le cas de nos expérimentations sur SMALLPRESENT-[8], il nous avait été possible d'obtenir une bonne estimation de la probabilité des différentielles [5]. La probabilité de la meilleure différentielle que nous avons trouvée est $(\texttt{0x1001}, \texttt{0x4040404000000000})$ sur 16 tours est bornée par $2^{-62.21}$. En utilisant seulement cette différentielle, une attaque différentielle classique utilisant les 2^{64} messages clairs aurait une probabilité de réussite de 59% pour une taille de liste $\ell = 2^{41}$. En utilisant plusieurs différentielles nous augmentons sensiblement la probabilité de succès de l'attaque.

A partir des 23 différentielles qui sont utilisées dans notre cryptanalyse nous obtenons des probabilités

$$p_* = 2^{-62.59} \quad \text{et} \quad p = 2^{-63.47}.$$

Le nombre de boîte-S actives est égal à 4 pour le tour 17 et à 8 pour le tour 18. Comme pour chaque boîtes-S le nombre de bits de clés impliqués est égal à 4. Cela nous fait en théorie 48 bits de clés à retrouver. En regardant de plus près l'algorithme de cadencement de clés pour la clé maître de 80 bits, nous avons remarqué que 6 bits sont partagés sur les deux derniers tours. Notre attaque nous permet donc au maximum de retrouver 42 bits de clés (on a 2^{42} candidats possibles pour la clé des deux derniers tours.)
Afin de diminuer la complexité en temps de l'attaque, nous pouvons utiliser la même méthode que celle donnée par Meiqin Wang dans son papier : nous pouvons décomposer le crible en ajoutant un crible intermédiaire après avoir partiellement déchiffré un tour. Les cribles $\Delta_{sieve}^{(i)'}$, correspondent à l'ensemble des différences possibles après $r+1$ tours. La taille maximale de ces cribles est 2^{12}.

Le tableau 6.4 nous donne les complexités de l'attaque pour différentes valeurs de la complexité en données et pour différentes valeurs de taille de liste.

6.5.1 Conclusion

L'attaque que nous avons présentée n'est pas la meilleure attaque sur PRESENT. Cette attaque a pour but d'illustrer le fait que la cryptanalyse différentielle multiple améliore la cryptanalyse différentielle classique. Le tableau 6.5 donne la liste des attaques statistiques effectuées sur PRESENT.

5. Dans la section 6.4, nous expliquons que nous les avons obtenues par une moyenne sur tous les clairs pour un certain nombre de clés

N_{DC}	ℓ	P_S	complexité en temps
2^{62}	$\ell = 2^{41}$	73%	$2^{79.00}$
2^{64}	$\ell = 2^{39}$	77%	$2^{76.00}$
2^{64}	$\ell = 2^{41}$	98%	$2^{79.00}$

TABLE 6.4 – Paramètres d'attaque différentielle multiple sur 18 tours de PRESENT. Pour une complexité en mémoire de 2^{42}(clé de 80 bits)

TABLE 6.5 – Résumé des attaques sur PRESENT.

#rounds	version	Cryptanalyse	N_{DC}	temps	mémoire	référence
8	128	intégrale	$2^{24.3}$	$2^{100.1}$	$2^{77.0}$	[ZRHD08]
16	80	différentielle	$2^{64.0}$	$2^{64.0}$	$2^{32.0}$	[Wan08]
17	128	clés liées	2^{63}	$2^{104.0}$	$2^{53.0}$	[ÖVTcK09]
18	**80**	**différentielle multiple**	2^{64}	2^{79}	2^{42}	**section 6.5**
19	128	différentielle algébrique	$2^{62.0}$	$2^{113.0}$	n/r	[AC09]
24	80	linéaire	$2^{63.5}$	$2^{40.0}$	$2^{40.0}$	[Ohk09]
24	80	saturation	$2^{57.0}$	$2^{57.0}$	$2^{32.0}$	[CS09]
25	128	linéaire	$2^{64.0}$	$2^{96.7}$	$2^{40.0}$	[NSZW09]
26	80	linéaire multiple	$2^{64.0}$	$2^{72.0}$	$2^{32.0}$	[Cho10]

Deuxième partie

Propriétés des boîtes-S

Chapitre 7

Introduction

Au début de la partie précédente nous avons défini les systèmes de chiffrement par bloc. Cette partie est dédiée à l'étude de certaines propriétés de la partie de substitution des algorithmes de chiffrement par bloc (voir section 1.3.1). Nous rappelons ici que cette partie de substitution, qui est non-linéaire, est composée de petites fonctions appliquées en parallèle à l'état interne du système de chiffrement. Ces fonctions sont appelées boîtes-S. Afin d'avoir un système de chiffrement qui résiste aux attaques statistiques présentées dans les chapitres 2 et 3, les boîtes-S doivent avoir de bonnes propriétés. Par exemple *l'uniformité différentielle* des boîtes-S (voir définition 2.5) permet de mesurer la vulnérabilité des boîtes-S contre les attaques différentielles (voir section 2.1). Chaque bit de sortie de la boîte-S peut s'écrire comme combinaison des bits de l'entrée. Ainsi certaines propriétés des boîtes-S peuvent être étudiées en analysant les propriétés des *fonctions booléennes* (fonction qui prend plusieurs bits d'entrée et a un bit en sortie).
Les boîtes-S peuvent aussi être représentées par un polynôme sur le corps fini $\mathbb{F}_{2^n}$. Pour faire le parallèle avec les fonctions booléennes, les fonctions alors obtenues sont appelées *fonctions vectorielles*. Ce chapitre d'introduction est dédié à la présentation d'un certain nombre de propriétés des fonctions booléennes et des fonctions vectorielles.
Le chapitre 8 est quant à lui consacré à certaines nombre de propriétés obtenues avec Anne Canteaut et Pascale Charpin.

Soit k un entier positif. Dans ce chapitre et le suivant nous manipulons des éléments du corps fini $\mathbb{F}_{2^n}$ de taille 2^n. Nous rappelons la définition de la trace que nous utilisons dans cette partie.

Définition 7.1. *Soit α un élément du corps fini $\mathbb{F}_{2^n}$. La trace de α sur le sous corps $\mathbb{F}_{2^k}$ où k divise n est notée $\mathbf{Tr}_k^n$ et est définie par*

$$\mathbf{Tr}_k^n(\alpha) = \alpha + \alpha^{2^k} + ... + \alpha^{2^{k(n/k-1)}}.$$

Lorsque que $k = 1$, c'est-à-dire lorsque le sous corps en question est $\mathbb{F}_2$, on note $\mathbf{Tr}$ l'application trace de $\mathbb{F}_{2^n}$ dans $\mathbb{F}_2$. On appelle cette trace la "trace absolue" *sur $\mathbb{F}_{2^n}$ Celle-ci vaut alors :*

$$\mathbf{Tr}(\alpha) = \alpha + \alpha^2 + ... + \alpha^{2^{(n-1)}}.$$

7.1 Les fonctions booléennes

7.1.1 Définition

Définition 7.2. *On appelle* **fonction booléenne à** n **variables** *toute application f de $\mathbb{F}_2^n$ dans $\mathbb{F}_2$. L'ensemble des fonctions booléennes à n variables est noté $\mathcal{B}_n$.*

Comme il existe une bijection entre l'espace vectoriel $\mathbb{F}_2^n$ et le corps $\mathbb{F}_{2^n}$, les vecteurs de $\mathbb{F}_2^n$ peuvent être identifiés aux les éléments du corps fini $\mathbb{F}_{2^n}$.

Définition 7.3. *Soit f une fonction booléenne à n variables. On appelle* **support** *de f l'ensemble des vecteurs de $\mathbb{F}_2^n$ qui ont une image non nulle, c'est-à-dire*

$$\mathrm{supp}(f) = \{x | f(x) \neq 0\}.$$

Définition 7.4. *Soit f une fonction booléenne, le* **poids de Hamming de f** *est égal au cardinal du support de f :*

$$wt(f) = \#supp(x).$$

Une fonction booléenne peut être définie par sa *table de vérité* : la table de vérité d'une fonction booléenne $f \in \mathcal{B}_n$ est l'ensemble des couples $(x, f(x))$ où x parcourt $\mathbb{F}_2^n$.

Exemple 7.1. *Soit f une fonction booléenne à trois variables définie par sa table de vérité :*

x	*(000)*	*(001)*	*(010)*	*(011)*	*(100)*	*(101)*	*(110)*	*(111)*
$f(x)$	*0*	*1*	*0*	*0*	*0*	*1*	*1*	*0*

Le support de f est $\mathrm{supp}(f) = \big\{(001), (101), (110)\big\}$ *et le poids de f est $wt(f) = 3$.*

Définition 7.5. *Une fonction booléenne est dite* **équilibrée** *si son image possède autant de 0 que de 1. C'est-à-dire $f \in \mathcal{B}_n$ est équilibrée si et seulement si*

$$wt(f) = 2^{n-1}.$$

Définition 7.6. *Soit f une fonction booléenne à n variables. La* **transformée de Mobius** *de f est définie par*

$$\begin{array}{rccl} f^\circ : & \mathbb{F}_2^n & \to & \mathbb{F}_2 \\ & u & \mapsto & \bigoplus_{v \preceq u} f(v) \end{array}$$

où $v \preceq u$ signifie $v_i \leq u_i \ \forall i \in \{1..n\}$.

Exemple 7.2. *Soit f la fonction définie dans l'exemple 7.1. La transformée de Mobius f° de f satisfait*

$$f^\circ(1,0,1) = f(0,0,0) \oplus f(0,0,1) \oplus f(1,0,0) \oplus f(1,0,1) = 0.$$

A partir de la définition de la transformée de Mobius on peut calculer la forme algébrique normale d'une fonction booléenne.

Définition 7.7. *Soit f une fonction booléenne à n variables.* **La forme algébrique normale***(ANF[1]) de f est définie par*

$$\bigoplus_{u=(u_1,..u_n)} f^\circ(u)\ x_1^{u_1}...x_n^{u_n},$$

où $u_i \in \{0,1\}$ et u parcourt $\mathbb{F}_2^n$.

Exemple 7.3. *Soit f la fonction définie dans l'exemple 7.1. La forme algébrique normale de f est :*

$$f(x_1,x_2,x_3) = x_3 \oplus x_2x_3 \oplus x_1x_2 \oplus x_1x_2x_3.$$

À partir de la forme algébrique normale d'une fonction booléenne on peut définir le degré algébrique de cette fonction.

Définition 7.8. *Le* **degré algébrique** *d'une fonction booléenne f à n variables est le degré de sa forme algébrique normale. C'est-à-dire si f est définie par*

$$\bigoplus_{u=(u_1,\cdots,u_n)} f^\circ(u)\ x_1^{u_1}...x_n^{u_n},$$

alors

$$deg(f) = \max_{u\in\mathbb{F}_2^n}\{wt(u)|f^\circ(u) \neq 0\}.$$

On dit que f est **affine** *si $deg(f) = 1$ et que f est* **constante** *si $deg(f) = 0$.*

7.1.2 Spectre de Walsh

En cryptographie nous avons besoin de calculer la distance des fonctions booléennes aux fonctions affines. Pour cela nous étudions la transformée de Walsh de la somme de f avec une fonction affine.

Définition 7.9. *Soit $u \in \mathbb{F}_{2^n}$, nous notons par φ_u la* **fonction booléenne affine** *définie par*

$$\begin{array}{rccc} \varphi_u : & \mathbb{F}_{2^n} & \to & \mathbb{F}_2 \\ & x & \mapsto & \mathbf{Tr}(ux) \end{array}$$

Nous définissons alors la *Transformée de Walsh* d'une fonction booléenne f comme la corrélation de $(-1)^f$ avec la fonction signe d'une fonction linéaire. Nous parlerons plus commodément de corrélation centrée de f avec une fonction linéaire.

Définition 7.10. *Soit $f \in \mathcal{B}_n$ une fonction booléenne. Le* **coefficient de Walsh** *de f au point $u \in \mathbb{F}_2^n$ est noté $\mathcal{F}(f+\varphi_u)$. Il correspond à la quantité*

$$\mathcal{F}(f+\varphi_u) = \sum_{x\in\mathbb{F}_2^n} (-1)^{f(x)+u\cdot x}$$

L'ensemble

$$\{\mathcal{F}(f+\varphi_u), u \in \mathbb{F}_2^n\},$$

est appelé **spectre de Walsh** *de la fonction f.*

1. ANF est l'abréviation du terme anglais *"Algebraic normal form"*

La *non-linéarité* d'une fonction booléenne mesure la distance entre la fonction booléenne et l'ensemble des fonctions affines.

Définition 7.11. *Soit $f \in \mathcal{B}_n$ une fonction booléenne. La* **non-linéarité** *de f notée $\mathcal{NL}(f)$ est la distance de Hamming entre la fonction f et l'ensemble des fonctions affines. Elle est définie par la valeur*

$$\mathcal{NL}(f) = 2^{n-1} - \frac{1}{2}\mathcal{L}(f) \quad où \quad \mathcal{L}(f) = \max_{u \in \mathbb{F}_{2^n}} |\mathcal{F}(f + \varphi_u)| \ .$$

La plus petite valeur pour $\mathcal{L}(f)$ est $2^{\frac{n}{2}}$. Cette valeur est atteinte par les fonctions dites *courbes*. Les fonctions courbes font partie d'une famille plus grande, appelée *fonctions plateaux*.

Définition 7.12. *[ZZ99][CCCF00] Soit $f \in \mathcal{B}_n$ une fonction booléenne. La fonction f est dite* **plateau** *si ses coefficients de Walsh prennent au plus trois valeurs, $\{0, \pm\mathcal{L}(f)\}$. On a alors $\mathcal{L}(f) = 2^s$ avec $s \geq n/2$.*
Les fonctions **courbes** *sont les fonctions plateaux qui vérifient $s = n/2$. Les fonctions courbes existent si et seulement si n est pair. Dans ce cas particulier ses coefficients de Walsh ne prennent que deux valeurs $\pm 2^{\frac{n}{2}}$.*
De plus, f est dite **plateau optimal** *si $s = (n+1)/2$ pour n impair ou si $s = (n+2)/2$ pour n pair.*

Dans le contexte des fonctions booléennes, la relation de Parseval est la suivante

$$\sum_{a \in \mathbb{F}_{2^n}} \mathcal{F}^2(f + \varphi_a) = 2^{2n}.$$

En appliquant cette relation il est facile de voir que le s défini dans la définition précédente ne peut pas être inférieur à $\frac{n}{2}$.

Avant de donner un exemple de fonction plateau. Nous donnons deux formules simples que nous utilisons à maintes reprises dans ce chapitre et dans le suivant.

Lemme 7.1. *Soient t et n deux entiers positifs. Nous avons*

$$\mathrm{pgcd}(2^t - 1, 2^n - 1) = 2^{\mathrm{pgcd}(t,n)} - 1 \tag{7.1}$$

A partir de l'égalité précédente on peut montrer [McE87][lemme 11.1] :

$$\mathrm{pgcd}(2^t + 1, 2^n - 1) = \begin{cases} 1, & si \quad \mathrm{pgcd}(t,n) = \mathrm{pgcd}(2t,n) \\ 2^{\mathrm{pgcd}(t,n)} + 1, & si \quad 2\mathrm{pgcd}(t,n) = \mathrm{pgcd}(2t,n). \end{cases} \tag{7.2}$$

Exemple 7.4. *Soit $f \in \mathcal{B}_n$ la fonction booléenne définie par $f(x) = \mathbf{Tr}(x^{2^t+1})$ et soit $k = \mathrm{pgcd}(2t, n)$. A partir de l'égalité (7.2), on peut calculer la transformée de Walsh des fonctions $f + \varphi_u$ pour $u \in \mathbb{F}_{2^n}$.*

$\mathcal{F}(f + \varphi_u)$	*nombre de u*
0	$2^n - 2^{n-k}$
$2^{(n+k)/2}$	$2^{n-k-1} + 2^{(n-k-2)/2}$
$-2^{(n+k)/2}$	$2^{n-k-1} - 2^{(n-k-2)/2}$.

Ainsi la fonction $f : x \mapsto \mathbf{Tr}(x^{2^t+1})$ est une fonction plateau qui n'est jamais courbe.

Il est aussi intéressant d'étudier les moments d'ordre supérieur de la transformée de Walsh d'une fonction booléenne. Nous donnons ici la définition du *moment d'ordre quatre normalisé*.

Définition 7.13. *Le* **moment d'ordre quatre normalisé** *d'une fonction booléenne $f \in \mathcal{B}_n$ est défini par :*

$$\nu(f) = 2^{-n} \sum_{u \in \mathbb{F}_{2^n}} \mathcal{F}^4(f + \varphi_u) \ .$$

Le moment d'ordre quatre normalisé de f est relié au moment d'ordre deux de la transformée de Walsh des dérivées de la fonction f.

Définition 7.14. *Soit $f : \mathbb{F}_{2^n} \to \mathbb{F}_2$ une fonction booléenne. La* **dérivée au point** *a de f est définie par*

$$\mathbb{D}_a f(x) = f(x) + f(x + a).$$

Remarque 7.1. *On vérifie facilement que*

$$\nu(f) = \sum_{a \in \mathbb{F}_{2^n}} \mathcal{F}^2(\mathbb{D}_a f). \tag{7.3}$$

C'est sous cette forme que le moment d'ordre quatre normalisé a été introduit dans [ZZ95] comme un critère cryptographique sous le nom de "sum of square indicator". Cette quantité a été particulièrement étudiée dans [CCCF00, CCCF01, ZZ99].

Le théorème suivant donne une borne sur la valeur du moment d'ordre quatre normalisé. La preuve de ce théorème se trouve dans [CCCF00] et [ZZ99].

Théorème 7.1. *Toute fonction booléenne $f \in \mathcal{B}_n$ vérifie $\nu(f) \leq 2^n \mathcal{L}^2(f)$. L'égalité est vraie si et seulement si f est plateau. Dans ce cas on a*

$$\mathcal{L}(f) = 2^s \text{ et } \nu(f) = 2^{n+2s}, \text{ pour } \frac{n}{2} \leq s \leq n. \tag{7.4}$$

7.2 Fonctions vectorielles

Au lieu d'étudier chaque bit de sortie des boîtes-S d'un système individuellement, on peut aussi regarder la fonction qui à une entrée de $\mathbb{F}_2^n$ donne une sortie dans $\mathbb{F}_2^m$ avec m plus grand que 1. Les fonctions de ce type, sont appelées *fonctions vectorielles*.

7.2.1 Définition

Définition 7.15. *On appelle* **fonction vectorielle à n entrées et m sorties** *une application de $\mathbb{F}_2^n$ dans $\mathbb{F}_2^m$. L'ensemble des fonctions vectorielles à n entrées et m sorties est noté $\mathcal{B}_n^m$.*

Par la suite une fonction vectorielle est caractérisée par une lettre majuscule afin de faire la distinction avec les fonctions booléennes que nous notons en minuscule.

Définition 7.16. *Soit F une fonction vectorielle $F : \mathbb{F}_{2^n} \mapsto \mathbb{F}_{2^m}$. Les* **composantes** *de F sont des fonctions booléennes. Elles sont notées f_λ ($f_\lambda \in \mathcal{B}_n$) et sont définies pour tout $\lambda \in \mathbb{F}_{2^m}$ par*

$$\begin{array}{rccl} f_\lambda : & \mathbb{F}_{2^n} & \to & \mathbb{F}_2 \\ & x & \mapsto & \mathbf{Tr}_1^m(\lambda F(x)). \end{array}$$

Toute fonction de $\mathbb{F}_{2^n}$ dans $\mathbb{F}_{2^n}$ peut s'exprimer comme un polynôme univarié de $\mathbb{F}_{2^n}[X]$. Avec cette représentation, on définit le degré d'une fonction vectorielle comme suit.

Définition 7.17. *Soit F une fonction de $\mathbb{F}_{2^n}$ dans $\mathbb{F}_{2^n}$ représentée par le polynôme univarié P dans $\mathbb{F}_{2^n}[X]$. Le* **degré** *de F est alors le degré maximal du poids de Hamming des exposants du polynôme P :*

$$\deg\left(\sum_{i=0}^{2^n-1} \lambda_i X^i\right) = \max\{wt(i) \mid \lambda_i \neq 0\},$$

où $\lambda_i \in \mathbb{F}_{2^n}$

En cryptographie, les fonctions de petit degré présentent des faiblesses contre les attaques algébriques.

7.2.2 Différentiabilité

Les boîtes-S des systèmes de chiffrement par bloc sont définies à partir de fonctions vectorielles. Dans la section 2.1 nous avons vu que la résistance des systèmes de chiffrement par bloc aux attaques différentielles était reliée à certaines propriétés des dérivées des fonctions vectorielles. Nous rappelons ici quelques définitions.

Définition 7.18. *Soit $F \in \mathcal{B}_n^m$ une fonction de $\mathbb{F}_{2^n}$ dans $\mathbb{F}_{2^m}$. Soit a un élément du corps $\mathbb{F}_{2^n}$. La* **dérivée de** *F par rapport à a est notée $\mathbb{D}_a F$ et est définie par*

$$\begin{array}{rccl} \mathbb{D}_a F & : \mathbb{F}_{2^n} & \to & \mathbb{F}_{2^m} \\ & x & \mapsto & F(x) + F(x+a) \end{array}$$

En cryptographie on s'intéresse à la répartition de l'image de la dérivée d'une fonction vectorielle

Définition 7.19. *Soit $F \in \mathcal{B}_n^m$ une fonction de $\mathbb{F}_{2^n}$ dans $\mathbb{F}_{2^m}$. Soient $a \in \mathbb{F}_{2^n}$ et $b \in \mathbb{F}_{2^m}$. On définit la quantité $\delta(a,b)$ par*

$$\begin{array}{rcl} \delta(a,b) & = & \#\{x \in \mathbb{F}_{2^n} | \mathbb{D}_a F(x) = b\} \\ & = & \#\{x \in \mathbb{F}_{2^n} | F(x) + F(x+a) = b\} \end{array}$$

Comme on est en caractéristique deux, si x vérifie $\mathbb{D}_a F(x) = b$ alors on a $\mathbb{D}_a F(x+a) = b$ et donc pour tout a et tout b, la quantité $\delta(a,b)$ est pair. Évidemment la dérivée par rapport à 0 n'a pas d'intérêt car pour toute fonction F et pour tout x nous avons $\mathbb{D}_0 F(x) = 0$.
En cryptographie le maximum des $\delta(a,b)$ pour a non nul, définit *l'uniformité différentielle* d'une fonction F.

Définition 7.20. *Soit $F \in \mathcal{B}_n^m$ une fonction de $\mathbb{F}_{2^n}$ dans $\mathbb{F}_{2^m}$. On note par $\delta(F)$ le maximum des $\delta(a,b)$ pour a non nul :*

$$\delta(F) = \max_{a \neq 0, b} \delta(a,b).$$

On dit alors que F est **différentiellement $\delta(F)$-uniforme**.

La quantité $\delta(F)$ est paire et supérieure ou égale à 2. Les fonctions atteignant la valeur minimale, c'est-à-dire les fonctions différentiellement 2-uniformes sont dites *APN* (*"almost perfect non-linear"*).

Dans beaucoup de systèmes de chiffrement par bloc, les boîtes-S sont "carrées" c'est-à-dire que le nombre de bits en entrée et en sortie de la boîte-S est le même. Cette propriété n'est pas toujours vraie notamment pour les systèmes de chiffrement de type Feistel pour lesquels la fonction de tour n'a pas besoin d'être inversible [2]. Par la suite on s'intéresse aux fonctions F qui sont dans $\mathcal{B}_n^n$, c'est-à-dire aux fonctions de $\mathbb{F}_{2^n}$ dans $\mathbb{F}_{2^n}$. Dans ces fonctions on peut distinguer les *permutations* des *non-permutations*. On rappelle qu'une fonction $F \in \mathcal{B}_n^m$ n'est pas une permutation si il existe x_1 et x_2 tels que $F(x_1) = F(x_2)$. Dans le cas où F n'est pas une permutation il est facile de voir qu'il existe a tel que $\delta(a,0) = 0$.

Proposition 7.1. *Soit $F \in \mathcal{B}_n^n$ une fonction de $\mathbb{F}_{2^n}$. Alors F est une permutation du corps $\mathbb{F}_{2^n}$ si et seulement si, pour tout a non nul, $\delta(a,0) = 0$.*

7.2.3 Non-linéarité

La *non-linéarité* d'une fonction vectorielle F de $\mathbb{F}_{2^n}$ dans $\mathbb{F}_{2^n}$ est définie à partir de la non-linéarité de ses fonctions composantes f_λ pour tout $\lambda \in \mathbb{F}_{2^n} \backslash \{0\}$(voir définition 7.16).

Définition 7.21. *Soit F une fonction de $\mathbb{F}_{2^n}$ dans lui-même avec fonctions composantes f_λ, $\lambda \in \mathbb{F}_{2^n}$. La* **non-linéarité** *de F est reliée à la non-linéarité de ses fonctions composantes f_λ. Celle-ci est égale à*

$$\mathcal{NL}(F) = 2^{n-1} - \frac{\Lambda(F)}{2} \text{ où } \Lambda(F) = \max_{\lambda \in \mathbb{F}_{2^n}^*} \mathcal{L}(f_\lambda).$$

La non-linéarité de la fonction F est une mesure qui permet de calculer sa vulnérabilité aux attaques linéaires (voir section 3.2). Les fonctions qui ont une non-linéarité maximale sont dites *presque courbes*. Cette propriété existe seulement pour n impair lorsque l'on considère les fonctions de $\mathcal{B}_n^n$.

Définition 7.22. *Soit F une fonction de $\mathbb{F}_{2^n}$ dans lui même avec composantes f_λ, $\lambda \in \mathbb{F}_{2^n}$. Alors,*

$$\Lambda(F) \geq 2^{\frac{n+1}{2}} .$$

Les fonctions F qui satisfont

$$\Lambda(F) = 2^{\frac{n+1}{2}}$$

2. On peut par exemple citer le DES [DES77] qui utilise des boîtes-S de 6 bits vers 4 bits

sont dites **presque courbes**[3]*. De plus si F est presque courbe, alors pour tout $a \in \mathbb{F}_{2^n}$ pour tout λ non nul*

$$\{\mathcal{F}(f_\lambda + \varphi_a),\ \lambda \in \mathbb{F}_{2^n}^*, a \in \mathbb{F}_{2^n}\} = \{0, \pm 2^{\frac{n+1}{2}}\}, \tag{7.5}$$

c'est-à-dire, toutes les composantes f_λ, $\lambda \neq 0$, sont "plateau optimal"(ou "semi bent").

7.2.4 Fonctions puissances

Dans sa généralité, l'étude des fonctions vectorielles et de leurs propriétés cryptographiques est complexe. Nous devons identifier des classes particulières. Les *monômes* sont faciles à implémenter et sont utilisés dans beaucoup de boîtes-S des système de chiffrement par bloc. On peut citer par exemple la boîte-S de l'AES (voir section 1.4.2) qui utilise la fonction $x \mapsto x^{-1}$ dans le corps $\mathbb{F}_{2^n}$.

Définition 7.23. *Soit F un polynôme sur le corps fini $\mathbb{F}_{2^n}$. Soit d un entier tel que $1 \leq d \leq 2^n - 2$. F est une* **fonction puissance** *ou encore un* **monôme** *si $F(x) = x^d$. Dans ce cas on note par F_d cette fonction*

$$\begin{array}{rccc} F_d: & \mathbb{F}_{2^n} & \to & \mathbb{F}_{2^n} \\ & x & \mapsto & x^d. \end{array}$$

Proposition 7.2. *Soit F_d une fonction puissance sur le corps $\mathbb{F}_{2^n}$:*

$$F_d(x) = x^d.$$

F_d est une permutation si et seulement $\text{pgcd}(d, 2^n - 1) = 1$.

Pour les fonctions puissances il existe des classes d'équivalence qui préservent l'uniformité différentielle. Les lemmes suivants détaillent certaines de ces équivalences.

Lemme 7.2. *Soit F_d une permutation puissance $F(x) = x^d$ du corps $\mathbb{F}_{2^n}$. Soit $d' = 2^i d \mod 2^n - 1$ un exposant dans la classe cyclotomique de d. Alors $\delta_d(a, b) = \delta_{d'}(a, b^{2^i})$*

Preuve : Supposons que $\delta_d(a, b) = \lambda \neq 0$ alors il existe λ racines du polynôme

$$X^d + (X + a)^d = b, \tag{7.6}$$

dans le corps fini $\mathbb{F}_{2^n}$.

En élevant cette équation à la puissance 2^i on obtient

$$X^{2^i d} + (X + a)^{2^i d} = b^{2^i}.$$

Ainsi si x est racine de (7.6), alors x est racine de

$$x^{d'} + (x + a)^{d'} = b^{2^i}.$$

Et on a $\delta_d(a, b) = \delta_{d'}(a, b^{2^i})$.

□

3. Le terme anglais est *"almost bent"* connu sous l'abréviation **AB**

Lemme 7.3. *Soit F une permutation $F_d(x) = x^d$. Soit F^{-1} la réciproque de F. Alors*

$$\delta_F(a,b) = \delta_{F^{-1}}(b,a).$$

Preuve : Supposons que $\delta_{F^{-1}}(a,b) = \lambda \neq 0$ alors il existe λ racines du polynôme

$$F^{-1}(X) + F^{-1}(X+a) = b. \tag{7.7}$$

Notons x une de ces racines. Alors on a

$$\begin{array}{rccc} & F^{-1}(x+a) & = & b + F^{-1}(x) \\ \Leftrightarrow & x+a & = & F(b+F^{-1}(x)) \\ \Leftrightarrow & a & = & F(b+F^{-1}(x)) + x \end{array}$$

En posant $y = F^{-1}(x)$, l'équation précédente devient $a = F(b+y)+F(y)$. Ainsi on obtient

$$\delta_{F^{-1}}(a,b) = \delta_F(b,a).$$

□

Remarque 7.2. *Le lemme précédent est vrai en particulier pour les monômes de permutation. Soit $F_d(x) = x^d$ une permutation de $\mathbb{F}_{2^n}$. On a*

$$\delta_d(a,b) = \delta_{d^{-1}}(b,a).$$

7.2.5 Dérivée en un point des fonctions puissances

Nous pouvons remarquer que pour les fonctions puissances, analyser la dérivée en un point suffit pour étudier les propriétés différentielles de la fonction.
En effet dans le cas des fonctions puissances, $F(x) = x^d$, les propriétés différentielles peuvent être analysées plus facilement puisque, pour tout $a \in \mathbb{F}_{2^n} \backslash \{0\}$, l'équation $(x+a)^d + x^d = b$ peut se récrire

$$a^d \left(\left(\frac{x}{a} + 1 \right)^d + \left(\frac{x}{a} \right)^d \right) = b,$$

ce qui implique que $\delta(a,b) = \delta(1, b/a^d)$ pour tout $a \neq 0$. Alors, si $F : x \mapsto x^d$ est une fonction puissance, les propriétés différentielles de F sont déterminées par les valeurs de $\delta(1,b)$, quand b parcourt le corps $\mathbb{F}_{2^n}$.

À partir de maintenant, nous écrivons $\delta(b)$ pour parler de $\delta(1,b)$.

Définition 7.24. *Soit F un monôme. L'uniformité différentielle de F se détermine par la quantité*

$$\delta(F) = \max_{b \in \mathbb{F}_{2^n}} \delta(b).$$

On dit alors que F est **différentiellement $\delta(F)$-uniforme.**

7.2.6 Remarques sur $\delta(0)$

Dans le but d'évaluer la différentiabilité d'une fonction puissance $F_d(x) = x^d$ définie sur le corps $\mathbb{F}_{2^n}$, nous pouvons commencer par étudier le cas particulier de certaines valeurs de $\delta(b)$. En particulier nous nous intéressons au cas particulier où $b = 0$. Dans cette partie, nous notons $\mathcal{S}_d(b)$ l'ensemble formé par les solutions de l'équation

$$(x+1)^d + x^d = b. \tag{7.8}$$

Dans le cas où $b = 0$, nous avons le résultat suivant :

Lemme 7.4. *Soit d un exposant tel que $\mathrm{pgcd}(d, 2^n - 1) = s$ alors $\delta(0) = s - 1$. En particulier $s = 1$ si et seulement si $\delta(0) = 0$.*

Ce lemme signifie que $\delta(0) = 0$ si et seulement si F_d est une permutation.
Une conséquence immédiate de ce lemme pour certaines valeurs de l'exposant d est la suivante :

Lemme 7.5. *Soit $d \geq 3$ tel que $d = \mathrm{pgcd}(d, 2^n - 1)$. Alors $\delta(0) = \delta(F_d) = d - 1$.*

Preuve : Puisque $d = \mathrm{pgcd}(d, 2^n - 1)$, à partir du lemme 7.4 nous avons $\delta(0) = d - 1$. De plus le polynôme $x^d + (x+1)^d + b$ est de degré $d - 1$ pour tout b. Ainsi on a que $\delta(b) \leq d - 1$ et on en déduit donc que $\delta(F_d) = d - 1$. □

Exemple 7.5. *Soit $d = 11$, d'après le lemme précédent on a $\delta(F_d) = \delta(0) = 10$ pour tout n tel que 11 divise $2^n - 1$. Comme 11 divise $1023 = 2^{10} - 1$, on obtient que cette propriété est vérifiée si 10 divise n.*

7.2.7 Les monômes APN

Dans cette thèse nous nous sommes intéressés aux monômes différentiellement 4- et 6-uniformes. Les résultats que nous avons trouvés sur ces fonctions sont détaillés dans le chapitre 8. Cependant cette thèse n'aurait pas été complète sans citer les résultats connus sur les monômes APN qui restent ceux qui résistent le mieux aux attaques différentielles. Dans son habilitation Anne Canteaut [Can06] avait déjà regroupé tous les résultats connus sur les monômes APN.

Le tableau 7.1 donne la liste des exposants d tel que la fonction $F(x) = x^d$ est APN sur le corps $\mathbb{F}_{2^n}$ dans le cas où n est impair. Nous rappelons que dans ce cas toutes les fonctions sont des permutations.
Dans le cas où n est pair, les monômes APN ne sont jamais des permutations. Le tableau 7.2.7 nous donne la liste des exposants d tels que $F(x) = x^d$ est APN.

Dans le chapitre suivant, nous remarquons que si $\mathrm{pgcd}(t, n) = s$, les fonctions avec exposant quadratique ou de Kasami sont différentiellement 2^s - uniformes. De plus dans le théorème 8.3 et le théorème 8.4, nous montrons que pour tout b, $\delta(a, b)$ est égal à 0 ou à 2^s.

Nom	Exposant	Conditions	Références
fonction quadratique Q_t	2^t+1	$1 \leq t \leq m$ $\text{pgcd}(t,n)=1$	[Gol68, Nyb94]
fonction de Kasami K_t	$2^{2t}-2^t+1$	$2 \leq t \leq m$ $\text{pgcd}(t,n)=1$	[Kas71]
fonction de Welsh	2^m+3		[Dob99a, CCD00]
fonction de Niho	$2^m+2^{\frac{m}{2}}-1$ $2^m+2^{\frac{3m+1}{2}}-1$	m pair m impair	[Dob99b, HX01]
fonction inverse	$2^{2m}-1$		[Nyb94, BD93]
fonction de Dobbertin	$2^{4g}+2^{3g}+2^{2g}+2^g-1$	$n=5g$	[Dob00]

TABLE 7.1 – Fonctions puissances $F(x)=x^d$ APN connues sur le corps $\mathbb{F}_{2^n}$ avec $n=2m+1$.

Nom	Exposant	Conditions	Références
fonction quadratique Q_t	2^t+1	$1 \leq t \leq m$ $\text{pgcd}(t,n)=1$	[Gol68, Nyb94]
fonction de Kasami K_t	2^{2t-2^t+1}	$2 \leq t \leq m$ $\text{pgcd}(t,n)=1$	[Kas71]
fonction de Dobbertin	$2^{4g}+2^{3g}+2^{2g}+2^g-1$	$n=5g$	[Dob00]

TABLE 7.2 – Fonctions puissances $F(x)=x^d$ APN connues sur le corps $\mathbb{F}_{2^n}$ avec $n=2m$.

Chapitre 8

Spectre différentiel des monômes

L'uniformité différentielle d'une fonction vectorielle a été introduite dans le but d'évaluer la résistance des systèmes de chiffrement par bloc aux attaques différentielles (chapitre 2). Ce chapitre est fondé sur un ensemble d'observations sur le spectre différentiel des fonctions. Par exemple, nous avons remarqué que deux fonctions F et G qui ont la même uniformité différentielle peuvent avoir des comportements différents par rapport aux attaques différentielles en fonction de la répartition des valeurs de leurs tables de différences [1].

Durant cette thèse, nous nous somme donc intéressés à cette répartition des différences pour certaines classes particulières de fonctions. Les fonctions monômes étant très utilisées pour définir les boîtes-S des systèmes de chiffrement par bloc, nous avons restreint notre analyse à cet ensemble de fonctions.
Pour les monômes nous avons précisé la notion de spectre différentiel et étudié l'intérêt de celui-ci. Dans le cheminement de cette étude du spectre différentiel, nous avons en particulier, étudié les fonctions avec uniformité différentielle égale à 4 ou 6.
Dans ce chapitre, après avoir défini l'intérêt de l'étude du spectre différentiel, nous présentons les résultats de notre recherche exhaustive du spectre différentiel des fonctions différentiellement 4- et 6-uniformes pour des petites tailles de corps. Les résultats de nos expériences nous ont permis d'identifier un certain nombre de classes de monômes qui étaient différentiellement 4- ou 6- uniformes. Les sections suivantes de ce chapitre sont dédiées à l'étude du spectre différentiel des monômes différentiellement 4- ou 6- uniformes.

Dans ce chapitre nous utilisons les mêmes notations que celles du chapitre 7. Soit d un entier ; nous étudions les monômes sur $\mathbb{F}_{2^n}$:

$$\begin{array}{rccc} F: & \mathbb{F}_{2^n} & \to & \mathbb{F}_{2^n} \\ & x & \mapsto x^d. & \end{array}$$

On rappelle que F est une permutation si et seulement si $\text{pgcd}(d, 2^n - 1) = 1$.

Dans la suite de ce chapitre, nous utilisons indifféremment le terme de *monôme* et de *fonction puissance*. Le terme *permutation puissance* désigne quant à lui, une fonction puissance bijective.

1. La définition de $\delta(a, b)$ est donnée dans la définition 7.19.

8.1 Spectre différentiel

Pour mesurer la résistance d'un système de chiffrement aux attaques différentielles, un certain nombre de propriétés des fonctions puissances ont été étudiées. Notamment on peut mesurer la résistance d'un système de chiffrement aux attaques différentielles en calculant l'uniformité différentielle des boîtes-S composant le système de chiffrement. Ainsi, soit F une fonction vectorielle comme définie dans le chapitre précédent ; pour vérifier la résistance du système de chiffrement contre les attaques différentielles on s'intéresse au maximum des $\delta(a,b)$. Dans ce chapitre nous nous intéressons aussi à la répartition des $\delta(a,b)$. En particulier nous illustrons le fait que cette répartition peut être différente pour deux fonctions qui ont la même uniformité différentielle.

8.1.1 Définition

En introduction on a souligné qu'on voulait étudier la répartition de $\delta(a,b)$ pour tout $a \in \mathbb{F}_{2^n}^*$ et pour tout $b \in \mathbb{F}_{2^n}$, c'est-à-dire que l'on veut s'intéresser aux valeurs $\#\{b \in \mathbb{F}_{2^n} | \delta(a,b) = i\}$. Or pour les fonctions puissances, nous avons vu à la fin du chapitre précédent qu'il suffit d'étudier la dérivée par rapport à 1 :

$$\#\{b \in \mathbb{F}_{2^n} | \delta(a,b) = i\} = \#\{b \in \mathbb{F}_{2^n} | \delta(1,b) = i\} \quad \forall a \neq 0.$$

Par la suite $\delta(1,b)$ est notée $\delta(b)$.

On définit alors le *spectre différentiel* d'une fonction puissance de la façon suivante.

Définition 8.1. *Soit $F(x) = x^d$ une fonction puissance sur le corps $\mathbb{F}_{2^n}$. Nous définissons la quantité ω_i, pour i pair, comme le nombre de b tels que l'équation $F(x) + F(x+1) = b$ a i racines :*

$$\omega_i = \#\{b \in \mathbb{F}_{2^n} | \delta(b) = i\}.$$

Le spectre différentiel de F est alors l'ensemble des ω_i :

$$\mathbb{S}_F = \{\omega_0, \omega_2, ..., \omega_{\delta(F)}\}.$$

Nous donnons d'abord des propriétés élémentaires des ω_i.

Lemme 8.1. *Soit ω_i les valeurs définies dans la définition 8.1. Pour une fonction puissance F nous avons les propriétés suivantes.*

$$\begin{cases} \sum_{i=0}^{\delta(F)} \omega_i = 2^n, \\ \sum_{i=0}^{\delta(F)} i \times \omega_i = 2^n. \end{cases}$$

Preuve : En sommant tous les ω_i, on obtient tous les $b \in \mathbb{F}_{2^n}$ et ceci une et une seule fois. Et en sommant tous les $i\omega_i$ on obtient toutes les racines des différentes équations $x^d + (x+1)^d = b$. Chaque élément du corps étant racine d'une et d'une seule équation, on en déduit la deuxième égalité. □

À titre d'exemple nous donnons le spectre différentiel de la fonction inverse. Le lemme suivant est basé sur un résultat de Kaisa Nyberg [Nyb94].

Lemme 8.2. *Soit $F(x) = x^{2^n-2}$ définie sur le corps $\mathbb{F}_{2^n}$. Le spectre différentiel de F est :*

– *Si n est impair, on a* $\mathbb{S}_F = \{\omega_0 = 2^{n-1}, \omega_2 = 2^{n-1}\}$.
– *Si n est pair, on a* $\mathbb{S}_F = \{\omega_1 = 2^{n-1}+1, \omega_2 = 2^{n-1}-2, \omega_4 = 1\}$.

Preuve : Dans cette preuve on note F, la fonction définie par $F(x) = x^{-1}$ avec la convention que $F(0) = 0$.

L'étude du spectre différentiel repose sur l'étude du nombre de solutions de l'équation :

$$x^{-1} + (x+1)^{-1} = b \tag{8.1}$$

Deux cas se présentent à nous.

1. Si $b = 1$ alors 0 et 1 sont solutions de (8.1). Pour $x \neq \{0, 1\}$ l'équation (8.1) peut se récrire
$$\frac{1}{x} + \frac{1}{x+1} = 1 \qquad \Leftrightarrow \qquad x^2 + x + 1 = 0 \tag{8.2}$$
Or $x^2+x+1 = 0$ a deux solutions dans $\mathbb{F}_{2^n}$ si et seulement si $\mathbf{Tr}(1) = 0$, c'est-à-dire si et seulement si n est pair.

   Donc si $b = 1$ on a 4 solutions dans le cas où n est pair et 2 solutions dans le cas où n est impair.
2. Si $b \neq 1$ alors une solution x de (8.1) est telle que $x \neq \{0, 1\}$ et doit vérifier $b = \dfrac{1}{x+1} + \dfrac{1}{x}$. Cette condition se réecrit $bx^2 + bx + 1 = 0$. Cette équation a 0 ou 2 solutions dans $\mathbb{F}_{2^n}$ selon que $\mathbf{Tr}(\frac{1}{b}) = 0$ ou 1.

En utilisant le lemme 8.1, on prouve la seconde partie du lemme. Cela revient à résoudre le système

$$\begin{cases} \omega_0 + \omega_2 + \omega_4 = 2^n \\ 2\omega_2 + 4\omega_4 = 2^n. \end{cases}$$

Ainsi dans le cas où n est impair, comme $\omega_4 = 0$, le spectre différentiel est $\{2^{n-1}, 2^{n-1}\}$. Dans le cas où n est pair, comme $\omega_4 = 1$, le spectre différentiel est $\{2^{n-1}+1, 2^{n-1}-2, 1\}$. □

Dans [CHZ07] Pascale Charpin, Tor Helleseth et Victor Zinoviev ont étudié d'autres critères sur la fonction inverse. Ils ont notamment étudié le spectre de Walsh de la dérivée de la fonction inverse.

Toutes les fonctions APN ont le même spectre différentiel $\{2^{n-1}, 2^{n-1}\}$. Mais les fonctions telles que $\delta(F) > 2$ présentent une grande variété de spectres.

Lemme 8.3. *Soit $F_d(x) = x^d$ et $F_e(x) = x^e$ deux fonctions puissances. S'il existe k tel que $e = 2^k d \bmod 2^n - 1$ ou si $\mathrm{pgcd}(2^n - 1, d) = 1$ et $e = d^{-1} \mod 2^n - 1$ alors F_d et F_e ont le même spectre différentiel.*

Dans la partie suivante nous donnons des arguments pour expliquer ce qui a motivé l'introduction du spectre différentiel : comment pour deux fonctions qui ont la même uniformité différentielle, le spectre différentiel peut influer sur la résistance du système de chiffrement aux attaques différentielles.

8.1.2 Intérêt de l'étude du spectre différentiel

Par le passé, Lars R. Knudsen et Kaisa Nyberg [NK92] a montré que connaissant l'uniformité différentielle de la fonction de tour d'un système de chiffrement de type Feistel on pouvait en déduire une borne sur la probabilité d'un chemin différentiel. Ce résultat nous dit que plus l'uniformité différentielle de la fonction de tour est petite, plus le système de chiffrement résiste aux attaques différentielles. D'où l'importance pour un système de chiffrement par bloc d'étudier la différentiabilité des boîtes-S le composant.

Lors de la conception d'un système de chiffrement par bloc, le concepteur a alors tout intérêt à choisir des boîtes-S qui sont APN. Or dans le cas où n (taille de la boîte-S) est pair, la seule permutation APN connue à l'heure actuelle est celle découverte par Dillon [BDMW10] dans le cas où $n = 6$. La forme algébrique normale de cette fonction est très complexe. En revanche, il a été prouvé qu'il n'existe pas de permutation puissance APN dans le cas où n est pair (une preuve récente peut être trouvée dans [BCCLC06]).

Pour des raisons d'implémentation dans la plupart des systèmes de chiffrement actuels les boîtes-S sont de taille 4 ou 8 bits [2]. Pour λ fixé, il existe un certain nombre de fonctions qui sont différentiellement λ-uniformes. Pour résister aux autres types de cryptanalyses, il existe d'autres critères qui permettent de choisir les meilleures boîtes-S. La non-linéarité, par exemple, (définition 7.21) de la fonction donne un critère de résistance du système de chiffrement contre les attaques linéaires (section 3.2).

Dans cette section, nous donnons l'exemple de deux systèmes de chiffrement identiques utilisant des boîtes-S différentes, ayant la même uniformité différentielle, et ne possédant pas le même potentiel de résistance à la cryptanalyse Ceci est dû aux différences entre leurs spectres différentiels.
Nous rappelons que les boîtes-S APN ont toutes le même spectre différentiel ($\omega_0 = 2^{n-1}, \quad \omega_2 = 2^{n-1}$). Par la suite, nous illustrons ce phénomène en utilisant des boîtes-S qui sont différentiellement 4-uniformes.

Exemple 8.1. *Dans cet exemple nous nous plaçons dans le cas où $n = 6$. D'après le tableau 8.1 qui est commenté dans la section 8.2, il existe un certain nombre de permutations puissances qui sont différentiellement 4-uniformes. Cette table illustre le fait que le spectre différentiel peut être très différent pour deux fonctions qui ont la même uniformité différentielle. Pour illustrer l'importance du spectre différentiel, nous allons prendre deux fonctions qui ont des spectres différentiels éloignés. Nous prenons par exemple les fonctions $G(x) = x^{31}$ et $H(x) = x^5$. La première fonction est dans la classe de la fonction inverse et donc son spectre différentiel est :*

$$\omega_0(G) = 33, \quad \omega_2(G) = 30, \quad \omega_4(G) = 1.$$

La fonction H quant à elle, est un monôme avec exposant quadratique [3]. La formule générale du spectre différentiel des fonctions avec exposant quadratique est donnée dans la section 8.4.2. D'après le tableau 8.1, pour $n = 6$, le spectre différentiel de H est :

$$\omega_0(H) = 48, \quad \omega_2(H) = 0, \quad \omega_4(H) = 16.$$

2. Voir par exemple les boîtes-S des systèmes de chiffrement PRESENT (section 1.4.1) et AES (section 1.4.2)

3. fonctions avec exposant $d = 2^i + 1$

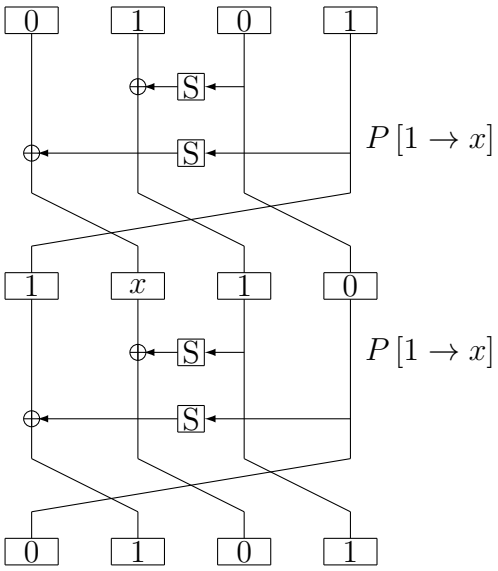

FIGURE 8.1 – Propagation d'une différence sur un Feistel généralisé. Illustre l'intérêt de l'étude du spectre différentiel

Le but est d'illustrer que la résistance d'un système de chiffrement à la cryptanalyse différentielle est différente selon que les boîtes-S de celui-ci sont définies par la fonction G ou par la fonction H.
Prenons l'exemple simple du chiffrement "jouet" défini par un schéma de Feistel généralisé avec deux boîtes-S (voir définition 1.7). La probabilité de la différentielle $(0,1,0,1) \to (0,1,0,1)$ *sur deux tours du système de chiffrement (voir figure 8.1) est définie par*

$$\sum_x P\left[1 \underset{S}{\to} x\right] P\left[1 \underset{S}{\to} x\right]$$

Dans le cas où la boîte-S est définie par la fonction H ($S = H$), tous les chemins différentiels avec différence en entrée $a = (0,1,0,1)$ et différence en sortie $b = (0,1,0,1)$ arrivent avec probabilité $2^{-8} = \left(\frac{4}{2^6}\right)^2$. Et le nombre de chemins suivant cette différentielle est exactement égal à $\omega_4 = 16$. On en déduit donc que la probabilité théorique de la différentielle est égale à $16 \times 2^{-8} = 2^{-4}$.

Dans le cas où $S = G$ un seul chemin avec différence en entrée $a = (0,1,0,1)$ et différence en sortie $b = (0,1,0,1)$ arrive avec probabilité $2^{-8} = \left(\frac{4}{2^6}\right)^2$ (c'est le cas où $x = 1$) tous les autres chemins arrivent avec probabilité $2^{-10} = \left(\frac{2}{2^6}\right)^2$. On en déduit donc que la probabilité théorique de la différentielle est égale à $30 \times 2^{-10} + 2^{-8} = 2^{-4.91}$.

Cet exemple simple sur deux tours d'un schéma de Feistel généralisé à quatre branches illustre bien l'intérêt de l'étude du spectre différentiel des boîtes-S puisque dans cet exemple plus ω_4 est petit plus la probabilité de la différentielle est aussi petite.

Au regard de cette discussion, la fonction inverse est celle qui possède la meilleure résistance contre la cryptanalyse différentielle (parmi les monômes de permutation différentiellement 4-uniformes dans le cas où n est pair) puisque pour la fonction inverse $\omega_4 = 1$. Le spectre différentiel de cette fonction est proche du spectre différentiel d'une fonction APN.

Définition 8.2. *Soit F une fonction puissance. Si F vérifie les conditions suivantes :*

$$\delta(b) \leq 2 \text{ pour tout } b \neq \{0,1\},$$

alors le spectre différentiel de F est proche du spectre différentiel d'une fonction APN. Par la suite, on dit que F est **localement-APN**.

Par rapport aux travaux de Kaisa Nyberg, nous dirons que dans le cas des schémas de Feistel, l'étude de l'uniformité différentielle nous donne une borne sur la probabilité du meilleur chemin différentiel alors que l'étude du spectre différentiel nous donne des indications sur le nombre de chemins composant une différentielle ainsi que leurs probabilités. Ce résultat donne alors une borne sur la probabilité d'une différentielle.

8.2 Fonctions puissances différentiellement 4- et 6- uniformes

Depuis les travaux de Lars R. Knudsen et Kaisa Nyberg [NK92], les fonctions APN ont été beaucoup étudiées. Ainsi on sait par exemple qu'il n'existe pas de monôme de permutation APN quand n est pair [BCCLC06]. Or, pour la conception de primitive cryptographique, nous avons besoin de fonctions bijectives, faciles à implémenter et différentiellement λ-uniformes avec λ petit (voir section 2.1). Dans cette thèse, nous nous sommes concentrées sur les fonctions puissances différentiellement 4- ou 6-uniformes. On rappelle ici, que les monômes APN ont tous le même spectre différentiel : $\{2^{n-1}, 2^{n-1}\}$. Dans l'exemple 8.1, nous avons illustré l'importance de l'étude du spectre différentiel quand $\delta(F) > 2$. Dans cette partie nous avons calculé le spectre différentiel de tous les monômes différentiellement 4- ou 6-uniformes sur le corps $\mathbb{F}_{2^n}$ avec $n < 27$. Lors de cette recherche nous avons identifié un certain nombre de classes de fonctions différentiellement 4- ou 6- uniformes. Cette section est dédiée à nos résultats expérimentaux. Nous divisons notre étude en quatre sous-sections : les permutations et non-permutations différentiellement 4- ou 6-uniformes. Dans chaque section nous identifions différentes classes de fonctions et nous commentons un résumé des résultats obtenus dans les sections suivantes. Certains résultats, qui ne sont pas prouvés, sont des questions ouvertes ou des conjectures.

Nous nous intéressons dans un premier temps aux permutations puissances différentiellement 4-uniformes.

8.2.1 Permutation puissances différentiellements 4-uniformes

Le tableau 8.1 présente tous les monômes $F(x) = x^d$ qui sont différentiellement 4-uniformes dans le corps $\mathbb{F}_{2^n}$, pour n compris entre 6 et 26. En accord avec le lemme 8.3, nous avons calculé le spectre différentiel des fonctions x^d où l'exposant d est le représentant de la classe cyclotomique modulo $2^n - 1$.

Permutations puissances $x \mapsto x^d$ sur le corps $\mathbb{F}_{2^n}$, différentiellement 4-uniformes					
n	exposant/inverse	ω_0	ω_2	ω_4	Type
6	5/13	48	0	16	Quadratique/Kasami
6	31/31	33	30	1	Inverse
7	19/47	85	22	21	
8	127/127	129	126	1	Inverse
9	45/125	292	184	36	
10	5/205	768	0	256	Quadratique
10	13/79	768	0	256	Kasami
10	17/181	768	0	256	Quadratique
10	29/247	573	390	61	
10	103/149	588	360	76	
10	223/367	603	330	91	
10	511/511	513	510	1	Inverse
11	79/183	1156	760	132	
11	109/695	1189	694	165	
11	251/367	1255	562	231	
11	463/703	1222	628	198	
12	73/731	2496	1152	448	Bracken et Leander[BL10]
12	2047/2047	2049	2046	1	Inverse
13	303/947	4603	3082	507	
14	5/3277	12288	0	4096	Quadratique
14	13/1339	12288	0	4096	Kasami
14	17/2893	12288	0	4096	Quadratique
14	65/2773	12288	0	4096	Quadratique
14	205/241	12288	0	4096	Kasami
14	319/979	12288	0	4046	Kasami (4033)
14	8191/8191	8193	8190	1	Inverse
16	32767/32767	32769	32766	1	Inverse
18	5/52429	196608	0	65536	Quadratique
18	13/20165	196608	0	65536	Kasami
18	17/46261	196608	0	65536	Quadratique
18	241/12101	196608	0	65536	Kasami
18	257/43861	196608	0	65536	Quadratique
18	1279/12605	196608	0	65536	Kasami (65281)
18	131071/131071	131073	131070	1	Inverse
20	1057/306539	651264	270336	126976	Bracken et Leander [BL10]
20	524287/524287	524289	524286	1	Inverse
22	5/838861	3145728	0	1048576	Quadratique
22	13/322639	3145728	0	1048576	Kasami
22	17/740173	3145728	0	1048576	Quadratique
22	65/709813	3145728	0	1048576	Quadratique

Permutations puissances $x \mapsto x^d$ sur le corps $\mathbb{F}_{2^n}$, différentiellement 4-uniformes.					
n	exposant/inverse	ω_0	ω_2	ω_4	Type
22	241/87019	3145728	0	1048576	Kasami
22	257/734419	3145728	0	1048576	Quadratique
22	1025/699733	3145728	0	1048576	Quadratique
22	3277/16639	3145728	0	1048576	Kasami (65281)
22	4033/246739	3145728	0	1048576	Kasami
22	5119/49981	3145728	0	1048576	Kasami (1047553)
22	2097151/2097151	2097153	2097150	1	Inverse
24	8388607/8388607	8388609	8388606	1	Inverse
26	5/13421773	50331648	0	16777216	Quadratique
26	13/5162299	50331648	0	16777216	Kasami
26	17/11842741	50331648	0	16777216	Quadratique
26	65/11356885	50331648	0	16777216	Quadratique
26	241/1396651	50331648	0	16777216	Kasami
26	257/11750611	50331648	0	16777216	Quadratique
26	1025/11719501	50331648	0	16777216	Quadratique
26	4033/848653	50331648	0	16777216	Kasami
26	4097/11187541	50331648	0	16777216	Quadratique
26	20479/3208147	50331648	0	16777216	Kasami
26	52429/65281	50331648	0	16777216	Kasami
26	66559/3951439	50331648	0	16777216	Kasami
26	33554431/33554431	33554433	33554430	1	Inverse

TABLE 8.1 – Permutations puissances $x \mapsto x^d$ sur le corps $\mathbb{F}_{2^n}$, différentiellement 4-uniformes

Nous décrivons les différentes classes de permutations puissances différentiellement 4-uniformes qui apparaissent dans le tableau 8.1. Pour certaines des ces fonctions le spectre différentiel est déjà connu. Pour d'autres, une étude plus complète du spectre différentiel est faite dans les sections suivantes.

La fonction inverse

La fonction inverse est la fonction avec exposant $2^n - 2$ (celui-ci est dans la même classe cyclotomique que l'exposant $2^{n-1} - 1$). Il est bien connu que dans le cas où n est pair la fonction inverse est différentiellement 4-uniformes [Nyb94]. Le spectre différentiel de cette fonction est donné dans le lemme 8.2.

L'exposant $2^{2k} + 2^k + 1$ pour $n = 4k$

En 2010 Carl Bracken et Gregor Leander ([BL10]) ont étudié les monômes $F(x) = x^d$ avec exposant $d = 2^{2k} + 2^k + 1$ dans le corps $\mathbb{F}_{2^{4k}}$. Dans leur article ils ont prouvé que cette fonction est différentiellement 4-uniformes. Nous rappelons ici leur résultat.

Théorème 8.1. *[BL10] Soit $n = 4k$ et $F(x) = x^{2^{2k}+2^k+1}$ la fonction puissance définie sur le corps $\mathbb{F}_{2^n}$. F est différentiellement 4-uniformes. De plus cette fonction est une permutation si et seulement si k est impair.*

Dans la section 8.3, à l'aide des résultats liant le spectre de Walsh avec le spectre différentiel, nous déduisons le spectre différentiel de ces fonctions (voir théorème 8.2).

Exposant quadratique ou de Kasami

Nous rappelons ici que l'on appelle *exposant quadratique*, tout exposant de la forme $2^t + 1$.

$$F : x \mapsto x^{2^t+1}$$

Et que l'on appelle *exposant de Kasami*, tout exposant de la forme $2^{2t} - 2^t + 1$.

$$F : x \mapsto x^{2^{2t}-2^t+1}$$

Les fonctions avec exposant quadratique ou avec exposant de Kasami sont APN si et seulement si $\text{pgcd}(t, n) = 1$. Dans les sections 8.4.2 et 8.4.3, nous étudions le spectre différentiel des fonctions avec exposant quadratique ou exposant de Kasami. Nous remarquons dans le tableau 8.1 qu'un certain nombre de ces fonctions sont différentiellement 4-uniformes. L'exposant de Kasami n'étant pas toujours le plus petit de sa classe nous donnons ici le lien entre les exposants donnés dans la table et l'exposant de Kasami correspondant :

Pour $n = 12$ (resp. $n = 20$), x^{73} (resp. x^{1057}) sont dans la même classe cyclotomique que x^d avec $d = 2^{2t} - 2^t + 1$ et $n = 4t$. Pour $n = 14$, nous remarquons que 319 est dans la classe de l'exposant de Kasami $319 \cdot 2^6 = 2^{12} - 2^6 + 1$. De la même façon, pour $n = 18$, 1279 est dans la même classe que l'exposant de Kasami $1279 \cdot 2^8 = 2^{16} - 2^8 + 1$.

Conjectures

En analysant les résultats du tableau 8.1, il est aussi important de remarquer qu'il n'existe pas de permutation puissance différentiellement-4-uniformes pour n impair entre 15 et 25.

On peut donc faire la conjecture suivante.

Conjecture 8.1. *Soit F une permutation puissance du corps $\mathbb{F}_{2^n}$. Si n est impair et $n \geq 15$ alors F n'est pas différentiellement 4-uniformes.*

De la même façon, dans le cas où n est pair, il semble que pour $n > 12$ toutes les permutations puissances différentiellement 4-uniformes font partie d'une des famille citées ci-dessus.

Conjecture 8.2. *Soit F une permutation puissance du corps $\mathbb{F}_{2^n}$ différentiellement 4-uniformes. Si n est pair et $n \geq 12$ alors d est équivalent soit à un exposant quadratique, soit à un exposant de Kasami, soit à l'exposant $2^{n-1} - 1$, soit si $n = 4k$ à l"exposant $2^{2k} + 2^k + 1$.*

8.2.2 Non-permutations puissances différentiellement 4-uniformes

D'un point de vue expérimental, nous avons aussi cherché les monômes différentiellement 4-uniformes qui ne sont pas des permutations. Le tableau 8.2 donne la liste de toutes les fonctions puissances $F(x) = x^d$ sur le corps $\mathbb{F}_{2^n}$, non-permutations, différentiellement 4-uniformes pour n compris entre 8 et 26. On peut d'abord remarquer que le corpus considéré est "pauvre". Nous détaillons cependant les différentes classes de non-permutations puissances différentiellement 4-uniformes.

n	s	$\text{pgcd}(s, 2^n - 1)$	ω_0	ω_2	ω_4	Type
8	5	5	192	0	64	Quadratique
	21	3	152	80	24	Bracken et Leander[BL10]
	95	5	156	72	28	
	111	3	140	104	12	
12	5	5	3072	0	1024	Quadratique
14	69	3	9200	6176	1008	
	81	3	9200	6176	1008	
16	5	5	49152	0	16384	Quadratique
	65	5	49152	0	16384	Quadratique
	273	3	40448	17408	7680	Bracken et Leander[BL10]
20	5	5	786432	0	262144	Quadratique
24	5	5	12582912	0	4194304	Quadratique
	1025	5	12582912	0	4194304	Quadratique
	4161	3	10452992	4259840	2064384	Bracken et Leander[BL10]

TABLE 8.2 – Non-permutations puissances $x \mapsto x^s$ sur le corps $\mathbb{F}_{2^n}$ différentiellement 4-uniformes

L'exposant quadratique

Certaines des non-permutations puissances avec exposant quadratique

$$F : x \mapsto x^{2^t+1}$$

sont différentiellement 4-uniformes. Dans la section 8.4.2, nous donnons leur spectre différentiel.

L'exposant $2^{2k} + 2^k + 1$ pour $n = 4k$

Soit $n = 4k$, le théorème 8.2 montre que les fonctions puissances avec exposant $2^{2k} + 2^k + 1$ sont différentiellement 4-uniformes. Dans le cas où k est pair, ce ne sont pas des permutations.

n impair

Le résultat suivant montre que dans le cas où n est impair, il n'existe pas de fonctions puissances différentiellement 4-uniformes qui ne sont pas des permutations.

Proposition 8.1. *Soit F une fonction puissance sur le corps $\mathbb{F}_{2^n}$ avec n impair. Si F est différentiellement 4-uniformes alors F est une permutation.*

Preuve : Raisonnons par l'absurde. Supposons que F est différentiellement 4-uniformes et que F n'est pas une permutation. D'après le lemme 7.4, nous avons que $\delta(0) = \text{pgcd}(d, 2^n - 1) - 1$. Comme F n'est pas une permutation, $\delta(0) \neq 0$ et comme F est différentiellement 4-uniformes, $\delta(0) \leq 4$. C'est-à-dire $\delta(0) \in \{2, 4\}$.

Supposons que $\delta(0) = 2$. Cela signifie que 3 divise $2^n - 1$. Ceci n'est pas possible car n est impair ce qui implique que $\text{pgcd}(2^2 - 1, 2^n - 1) = 2^{\text{pgcd}(2,n)} - 1 = 1$ et donc que $\delta(0) \neq 2$.

Supposons que $\delta(0) = 4$. Cela signifie que 5 divise $2^n - 1$. Or $\text{pgcd}(2^2 + 1, 2^n - 1) = 1$ (voir lemme 7.1). Donc $\delta(0) \neq 4$. On peut donc conclure que $\delta(0) \geq 6$ et que donc qu'il n'existe pas de non-permutation puissance différentiellement 4-uniforme dans le cas où n est impair.

□

La proposition précédente est analogue au résultat bien connu suivant :

Proposition 8.2. *Soit F un fonction puissance de $\mathbb{F}_{2^n}$ avec n impair. Si F est APN alors F est une permutation.*

Conjectures

Dans le tableau 8.2, on remarque que si $n \equiv 2 \mod 4$ et $18 \leq n \leq 26$, il n'existe pas de non-permutation puissance différentiellement 4-uniforme. Nous faisons, donc, la conjecture suivante :

Conjecture 8.3. *Soit $\mathbb{F}_{2^n}$ le corps de taille 2^n. Si $n \equiv 2 \mod 4$ il n'existe pas de non-permutation puissance différentiellement 4-uniforme pour $n \geq 18$.*

Dans le cas où $n \equiv 0 \mod 4$ on fait la conjecture suivante.

Conjecture 8.4. *Soit F une non-permutation puissance sur le corps $\mathbb{F}_{2^n}$ avec $n \equiv 0 \mod 4$ et $n \geq 12$. Si F est différentiellement 4-uniforme alors d est équivalent soit à un exposant quadratique soit à l'exposant $2^{2k} + 2^k + 1$ pour $n = 4k$.*

8.2.3 Permutation puissance différentiellement 6-uniformes

Le nombre de permutations puissances différentiellement 6-uniformes est assez important en particulier pour les corps $\mathbb{F}_{2^n}$ avec $n \leq 14$.

Dans nos expérimentations, nous avons remarqué que parmi les permutations puissances différentiellement 6-uniformes, une classe d'exposants se dégageait nettement. Cette classe correspond aux fonctions $F(x) = x^d$ avec $d = 2^t - 1$ pour certaines valeurs de $2 \leq t \leq n - 1$. Dans le tableau 8.3, pour $7 \leq n \leq 26$, nous avons calculé le nombre de permutations puissances différentiellement 6-uniformes qui ont un exposant de la forme $d = 2^t - 1$.

On peut premièrement remarquer que quand n grandit la plupart des permutations puissances différentiellement 6-uniformes sont équivalentes aux fonctions $F(x) = x^d$ avec $d = 2^t - 1$ pour $2 \leq t \leq n - 1$.

Une étude approfondie du spectre des fonctions puissances avec exposant $d = 2^t - 1$ est présentée dans la section 8.6.7.

On peut aussi remarquer que pour $n = 18$ et $n = 24$ il n'existe pas de permutations puissances différentiellement 6-uniformes. Ainsi on peut formuler la conjecture suivante.

Conjecture 8.5. *Soit $\mathbb{F}_{2^n}$ le corps à 2^n éléments. Pour $n \equiv 0 \mod 6$ et $n \geq 18$, il n'existe pas de permutation puissance différentiellement 6-uniforme.*

n	*exposant*	nombre de fonctions
7	2^t-1	1
	Autre	1
8	2^t-1	1
	Autre	**0**
9	2^t-1	**0**
	Autre	2
10	2^t-1	2
	Autre	11
11	2^t-1	6
	Autre	46
12	2^t-1	**0**
	Autre	2
13	2^t-1	5
	Autre	130
14	2^t-1	2
	Autre	24
15	2^t-1	**0**
	Autre	6
16	2^t-1	2
	Autre	**0**

n	*exposant*	nombre de fonctions
17	2^t-1	12
	Autre	**0**
18	2^t-1	**0**
	Autre	**0**
19	2^t-1	8
	Autre	**0**
20	2^t-1	2
	Autre	**0**
21	2^t-1	**0**
	Autre	2
22	2^t-1	2
	Autre	**0**
23	2^t-1	6
	Autre	**0**
24	2^t-1	**0**
	Autre	**0**
25	2^t-1	6
	Autre	**0**
26	2^t-1	2
	Autre	**0**

TABLE 8.3 – Monômes de permutation différentiellement 6-uniformeS sur le corps $\mathbb{F}_{2^n}$. Résumé du nombre de fonctions pour un exposant d de la forme particulière $d = 2^t - 1$ ou avec une autre valeur.

n	s	$\text{pgcd}(s, 2^n - 1)$	ω_0	ω_2	ω_4	ω_6	Type
6	7	7	35	27	1	1	$2^t - 1$
8	25	5	172	48	28	8	
	63	3	156	86	0	14	$2^t - 1$
12	7	7	2401	1518	1	176	$2^t - 1$
14	75	3	9858	4958	1470	98	
	105	3	9487	5693	1113	91	
15	7	7	19046	12391	0	1331	$2^t - 1$
16	63	3	38116	24746	0	2674	$2^t - 1$
	4915	5	38988	21024	4828	696	
	16383	3	38116	24746	0	2674	$2^t - 1$
18	7	7	153167	97929	1	11047	$2^t - 1$
22	255	3	2446578	1573013	0	174713	$2^t - 1$
24	7	7	9788205	6289212	1	699798	$2^t - 1$
26	262143	3	39142742	25171967	0	2794155	$2^t - 1$
	16777215	3	39142742	25171967	0	2794155	$2^t - 1$

TABLE 8.4 – Non-permutations puissances $x \mapsto x^s$ sur le corps $\mathbb{F}_{2^n}$ différentiellement 6-uniformes.

8.2.4 Non-permutations puissances différentiellement 6-uniformes

Le tableau 8.4 donne la liste des fonctions puissances $F = x^d$ sur le corps $\mathbb{F}_{2^n}$ pour $6 \leq n \leq 26$ qui ne sont pas des permutations, et qui sont différentiellement 6-uniformes.

On peut remarquer qu'à part quelques exceptions pour des corps de petite taille, tous les exposants cités dans le tableau sont de la forme $2^t - 1$. Ainsi, comme dans le cas des permutations différentiellement 6-uniformes, on peut faire la conjecture suivante :

Conjecture 8.6. *Soit $F = x^d$ une fonction puissance, non-permutation du corps $\mathbb{F}_{2^n}$ avec $n \geq 18$. Si F est différentiellement 6-uniforme alors d est équivalent à un exposant de la forme $2^t - 1$.*

8.2.5 Récapitulatif

Les résultats de nos expérimentations détaillées dans les sections précédentes nous ont permis de remarquer que le nombre de fonctions différentiellement 4- ou 6-uniformes est assez petit. De plus quand la taille de corps grandit tous les monômes observés ont des exposants de la forme

- $d = 2^t + 1$
- $d = 2^{2t} - 2^t + 1$
- $d = 2^{2m} + 2^m + 1$ avec $n = 4m$
- $d = 2^t - 1$

Dans le tableau 8.5 nous résumons les résultats de nos expérimentations.

Les sections suivantes de ce chapitre sont dédiées à l'étude de ces quatre classes de fonctions. Ainsi dans la section 8.3 nous étudions le spectre différentiel de la fonction

puissance avec exposant $d = 2^{2m} + 2^m + 1$ avec $n = 4m$. La section 8.4 est dédiée à l'étude des fonctions avec exposant quadratique $d = 2^t + 1$ ou de Kasami $d = 2^{2t} - 2^t + 1$. Et enfin dans la section 8.6 nous étudions le spectre différentiel des fonctions avec exposant $d = 2^t - 1$.

	Exposant	permutation 4-uniforme	non-permutation 4-uniforme	permutation 6-uniforme	non-permutation 6-uniforme
n pair	Inverse $2^{n-1}-1$	toujours	$\emptyset$	$\emptyset$	$\emptyset$
	Quadratique 2^t+1	$\gcd(t,n)=2$ $\gcd(t,n)=\gcd(t,2n)$	$\gcd(t,n)=2$ $\gcd(t,n)\neq\gcd(t,2n)$	$\emptyset$	$\emptyset$
	Kasami $2^{2t}-2^t+1$	$n\neq 3t$ $\gcd(n,t)=2$ $n\equiv 2 \mod 4$	$\emptyset$	$\emptyset$	$\emptyset$
	$n=4k$ $2^{2k}+2^k+1$	k impair	k pair	$\emptyset$	$\emptyset$
	2^t-1 $(t\neq\{2,n-1\})$	$\emptyset$	$\emptyset$	$\gcd(t,n)=1$ autres conditions voir section 8.6	$\gcd(t,n)\neq 1$ autres conditions voir section 8.6
	Remarques	Conjecture : Liste complète pour $n\geq 12$	Conjecture : $n\equiv 2 \mod 4$ Inexistence pour $n\geq 18$ $n\equiv 0 \mod 4$ Liste complète pour $n\geq 16$	Conjecture : Liste complète pour $n\geq 18$	Conjecture : Liste complète pour $n\geq 18$
n impair	2^t-1 $(t\neq\{2,n-1\})$	$\emptyset$	$\emptyset$	$\gcd(t,n)=1$ autres conditions voir section 8.6	$\gcd(t,n)\neq 1$ autres conditions voir section 8.6
	Remarques	Conjecture : Inexistence pour $n\geq 15$	Inexistence prouvée voir **??**		Conjecture : Liste complète pour $n\geq 17$

TABLE 8.5 – Resumé sur les monômes différentiellement 4-et 6-uniformes

8.3 Monômes avec exposant $2^{2k}+2^k+1$

Dans [BL10], Carl Bracken et Gregor Leander ont montré que les fonctions $F(x) = x^{2^{2k}+2^k+1}$ sur le corps $\mathbb{F}_{2^{4k}}$ sont différentiellement 4-uniformes. Nous avons étudié le spectre différentiel de ces fonctions. Les résultats présentés dans cette section ont été présentés dans [BCC10a]. Dans cette section nous donnons tout d'abord quelques résultats préliminaires qui relient le spectre différentiel d'une fonction différentiellement 4-uniforme avec le spectre de Walsh de la fonction booléenne associée.

8.3.1 Lien entre le spectre différentiel d'une fonction différentiellement 4-uniformes et le spectre de Walsh

Le résultat principal de cette section donne le lien entre le spectre différentiel d'une fonction vectorielle différentiellement 4-uniforme et le spectre de Walsh de la fonction booléenne associée.

Soit la fonction puissance $F(x) = x^d$ dans cette section nous notons par f_d sa fonction booléenne associée :

$$\begin{array}{rccc} f_d: & \mathbb{F}_{2^n} & \to & \mathbb{F}_2 \\ & x & \mapsto & \mathbf{Tr}(x^d) \end{array} \tag{8.3}$$

Nous rappelons (voir définition 7.13) la définition du moment d'ordre quatre normalisé pour les fonctions booléennes :

$$\nu(f) = 2^{-n} \sum_{u \in \mathbb{F}_{2^n}} \mathcal{F}^4(f + \varphi_u) \ . \tag{8.4}$$

Proposition 8.3. *[BCCLC06] Soit $F(x)$ une permutation puissance du corps $\mathbb{F}_{2^n}$. Soit $f_d = \mathbf{Tr}(x^d)$ la fonction booléenne associée à F. Alors*

$$\begin{aligned} \nu(f) &= 2^n \#\{(x,y) \in \mathbb{F}_{2^n} \times \mathbb{F}_{2^n} | x^d + (x+1)^d = y^d + (y+1)^d\} \\ &= 2^{2n+1} + 2^n \#\big\{(x,y) \in \mathbb{F}_{2^n} \times \mathbb{F}_{2^n} | x \neq y \neq y+1 \text{ et } x^d + (x+1)^d = y^d + (y+1)^d\big\} \end{aligned}$$

À partir de la proposition précédente, nous pouvons déduire la proposition suivante :

Proposition 8.4. *Soit $F(x) = x^d$ une permutation puissance du corps $\mathbb{F}_{2^n}$. Soit $f_d(x)$ la fonction booléenne associée. On a*

$$2^{-n}\nu(f_d) = \sum_{x \in \mathbb{F}_{2^n}} \delta(x^d + (x+1)^d).$$

Donc $\delta(F) \geq 2^{-2n}\nu(f_d)$.

Preuve : Soit $\Lambda = 2^{-n}\nu(f_d)$. A partir de la proposition 8.3, nous avons

$$\Lambda = \#\{(x,y) \in \mathbb{F}_{2^n} \times \mathbb{F}_{2^n} | x^d + (x+1)^d = y^d + (y+1)^d\}.$$

Ce qui implique que

$$\begin{aligned} \Lambda &= \sum_{x \in \mathbb{F}_{2^n}} \#\{y \in \mathbb{F}_{2^n} | y^d + (y+1)^d = b \text{ avec } b = x^d + (x+1)^d\} \\ &= \sum_{x \in \mathbb{F}_{2^n}} \big(\delta(x^d + (x+1)^d\big) \leq 2^n \delta(F) \end{aligned}$$

□

Une conséquence directe de la proposition 8.4 est que le spectre différentiel d'une permutation puissance différentiellement 4-uniforme est déterminé par le moment d'ordre quatre normalisé de la fonction booléenne associée.

Lemme 8.4. *Soit $F(x) = x^d$ une permutation puissance de $\mathbb{F}_{2^n}$. Soit $f_d = \mathbf{Tr}(x^d)$ sa fonction booléenne associée (voir (8.3)). Si $\delta(F) = 4$ alors le spectre différentiel de F est le suivant.*

$$\omega_4 = \frac{\nu(f_d)}{2^{n+3}} - 2^{n-2}, \omega_2 = 2^{n-1} - 2\ \omega_4 \quad et \quad \omega_0 = 2^{n-1} + \omega_4. \tag{8.5}$$

Donc, $\nu(f) = 2^{n+3}\kappa$ avec $2^{n-2} < \kappa \leq 2^{n-1}$. En particulier, si $\kappa = 2^{n-1}$ alors $\omega_2 = 0$.

Preuve : D'après la proposition 8.4 on a

$$2^{-n}\nu(f) = \sum_{b\in\mathbb{F}_{2^n}} \delta(b)^2 = \sum_{i=0}^{2^n} i^2\omega_i,$$

ce qui implique que

$$2^{-n}\nu(f) = 2^2\omega_2 + 2^4\omega_4 \text{ avec } 2\omega_2 + 4\omega_4 = 2^n.$$

En remplaçant ω_2 par $(2^{n-1} - 2\omega_4)$, nous obtenons $\omega_4 = \nu(f)/2^{n+3} - 2^{n-2}$.

De ce résultat nous déduisons que $\nu(f) = 2^{n+3}\kappa$ avec $\kappa > 0$. Comme $0 < \omega_4 \leq 2^{n-2}$, nous devons avoir $2^{n-2} < \kappa \leq 2^{n-1}$. En particulier $\omega_2 = 0$ si et seulement si $\kappa = 2^{n-1}$. □

8.3.2 Spectre différentiel

A partir des résultats précédents qui nous donnent le lien entre le spectre différentiel d'une fonction différentiellement 4-uniforme et le spectre de Walsh de la fonction booléenne associée, nous pouvons extraire le spectre différentiel de la fonction $F(x) = x^d$ sur le corps $\mathbb{F}_{2^n}$ avec $d = 2^{2k} + 2^k + 1$.

Théorème 8.2. *Soit $F(x)$ la fonction puissance définie sur le corps $\mathbb{F}_{2^{4k}}$ par $F(x) = x^d$ avec $d = 2^{2k} + 2^k + 1$. Le spectre différentiel de cette fonction est le suivant :*

$$\omega_4 = 2^{3k-3}(2^k - 1), \qquad \omega_2 = 2^{3k-2}(2^k + 1) \qquad et \qquad \omega_0 = 5.2^{4k-3} - 2^{3k-3}.$$

Preuve : Carl Bracken et Gregor Leander ont prouvé que $\delta(F) = 4$ (voir théorème 8.1). Si on veut utiliser le lemme précédent il suffit de calculer le moment d'ordre quatre normalisé de la fonction booléenne associée. D'après la définition même du moment d'ordre quatre normalisé pour calculer celui-ci, il suffit de connaître le spectre de Walsh. Or Hans Dobbertin a montré [Dob98] que le spectre de Walsh de la fonction $f_d = \mathbf{Tr}(x^d)$ avec $d = 2^{2k} + 2^k + 1$ est le suivant :

$\mathcal{F}(f_d + \varphi_u)$	nombre u
-2^{2k+1}	$(2^{n-2} - 2^{3(k-1)})/3 - 2^{2k-2}$
-2^{2k}	$(2^{n-1} + 2^{3k-1})/3$
0	$2^{n-1} - 2^{3k-2}$
2^{2k}	$(2^{n-1} + 2^{3k-1})/3$
-2^{2k+1}	$(2^{n-2} - 2^{3(k-1)})/3 + 2^{2k-2}$.

D'après (8.4), nous avons

$$\begin{aligned} 2^n\nu(f_d) &= 2^{4(2k+1)}\ \frac{(2^{n-1}-2^{3(k-1)+1})}{3}+2^{8k}\frac{(2^n+2^{3k})}{3} \\ &= 2^{8k}\ \frac{(2^{k+3}-2^{3k+2}+2^n+2^{3k})}{3} \\ &= 2^{8k}\ \frac{(9.2^n-3.2^{3k})}{3}=2^{11.k}(3.2^k-1). \end{aligned}$$

De sorte que le moment d'ordre quatre normalisé est $\nu(f_d)=2^{7k}(3.2^k-1)$.

Ainsi en appliquant le lemme 8.4, on obtient

$$\omega_4 = 2^{3k-3}(2^{k+1}+2^k-1)-2^{4k-2}=2^{3k-3}(2^k-1)$$

et

$$\omega_2 = 2^{4k-1}-2^{3k-2}(2^k-1)=2^{4k-2}+2^{3k-2}=2^{3k-2}(2^k+1).$$

□

8.4 Fonctions avec exposant quadratique ou de Kasami

En analysant les résultats du tableau 8.1, nous avons remarqué que le spectre différentiel des fonctions puissances avec exposant quadratique ou exposant de Kasami différentiellement 4-uniformes est de la forme suivante $\{2^{n-1}+2^{n-2},0,2^{n-2}\}$. Ce spectre différentiel semble particulier car seuls ω_0 et ω_4 sont non-nuls.

Dans cette section nous nous sommes donc intéressés au cas particulier des fonctions puissances où $\delta(a,b)$ prend seulement 2 valeurs, c'est-à-dire, $\delta(a,b)\in\{0,\kappa\}$ pour tout $(a,b)\in\mathbb{F}_{2^n}^*\times\mathbb{F}_{2^n}$ et pour $\kappa\geq 2$.

Définition 8.3. *Soit F une fonction puissance définie sur le corps $\mathbb{F}_{2^n}$. On dit que F est* **différentiellement 2-valuées** *si $\delta(b)\in\{0,\delta(F)\}$. C'est-à-dire si pour tout $i\neq\{0,\delta(F)\}$, $\omega_i=0$.*

Exemple 8.2. *Les fonctions puissances APN sont différentiellement 2-valuées ($\delta(b)\in\{0,2\}$).*

Remarque 8.1. *Soit F une fonction linéaire sur le corps $\mathbb{F}_{2^n}$ alors F est différentiellement 2-valuées avec spectre différentiel $\omega_0=2^n-1,\quad \omega_{2^n}=1$.*

Dans la suite de cette section, on étudie les fonctions différentiellement 2-valuées non-linéaires.

8.4.1 Fonctions puissances différentiellement 2-valuées

Soit F une fonction puissance différentiellement 2-valuées. Nous remarquons que dans ce cas $\delta(F)$ est une puissance de 2.

Lemme 8.5. *Soit F une fonction puissance du corps $\mathbb{F}_{2^n}$ Supposons que F est différentiellement 2-valuées. Alors il existe s ($1\leq s\leq n$) telle que $\delta(F)=2^s$.*

Preuve : Cette propriété est simple à montrer dans le cas où F est une fonction puissance. En effet, d'après le lemme 8.1, nous avons

$$\begin{cases} \omega_0 + \omega_\kappa = 2^n \\ \kappa\omega_\kappa = 2^n. \end{cases}$$

Donc κ divise 2^n. □

Proposition 8.5. *Soit $F(x) = x^d$ une fonction puissance du corps $\mathbb{F}_{2^n}$ tel que $\delta(F) = 2^s$. Soit $f_d(x) = \mathbf{Tr}(x^d)$ la fonction booléenne associée. Si F est différentiellement 2-valuées alors $\nu(f_d) = 2^{2n+s}$.*

Preuve : D'après la proposition 8.4 nous avons

$$\begin{aligned} \nu(f) &= 2^n \sum_x \delta(x^d + (x+1)^d) \\ &= 2^n \cdot 2^n \cdot 2^s. \end{aligned}$$

□

À partir de ce résultat, nous déduisons que certains monômes, pour lesquels la fonction booléenne associée est plateau, (voir définition 7.12) sont différentiellement 2-valués.

Proposition 8.6. *Soit d un entier tel que $\mathrm{pgcd}(d, 2^n - 1) = 1$. Soit $F(x) = x^d$ une permutation et $f_d(x) = \mathbf{Tr}(x^d)$. Supposons que f_d est une fonction plateau avec spectre de Walsh $\{0, \pm 2^{(n+k)/2}\}$. Alors $\delta(F) \geq 2^k$ avec égalité si et seulement si $\delta(b) \in \{0, 2^k\}$ pour tout b. De plus, si tout $\delta(b)$ non-nul est supérieur ou égal à 2^k alors $\delta(b) \in \{0, 2^k\}$ pour tout $b \in \mathbb{F}_{2^n}$.*

Preuve : Puisque f_d est plateau à partir de la définition 7.12 et du théorème 7.1 nous avons $\nu(f_d) = 2^{2n+k}$. En utilisant la proposition 8.4, nous obtenons

$$2^{n+k} = \sum_{x \in \mathbb{F}_{2^n}} \delta\left(x^d + (x+1)^d\right).$$

Donc $\delta(F) \geq 2^k$ et l'égalité est vraie si et seulement si $\delta\left(x^d + (x+1)^d\right)$ est égal à 2^k pour tout x. D'autre part, il est clairement impossible d'avoir $\delta(b) \geq 2^k$, pour tout $\delta(b)$ non-nul, sauf si $\delta(b) \in \{0, 2^k\}$ pour tout b. □

A l'aide de ces résultats préliminaires nous pouvons déduire le spectre différentiel des fonctions puissances avec exposant quadratique ou exposant de Kasami.

8.4.2 L'exposant quadratique

Dans cette section, nous considérons le cas particulier des fonctions $Q_t(x) = x^{2^t+1}$ avec $1 \leq t \leq n-1$ sur le corps $\mathbb{F}_{2^n}$. La valeur $d = 2^t + 1$ est appelée *exposant quadratique.*

Théorème 8.3. *Soit Q_t une fonction du corps $\mathbb{F}_{2^n}$ définie par $Q_t(x) = x^{2^t+1}$.*
Soit $s = \text{pgcd}(t, n)$. Considérons l'équation

$$Q_t(x) + Q_t(x + a) = b. \tag{8.6}$$

pour tout a, b dans $\mathbb{F}_{2^n}$. Si (8.6) a au moins une solution x, alors l'ensemble des solutions est $x + a\mathbb{F}_{2^s}$. C'est-à-dire que les fonctions puissances avec exposant quadratique sont différentiellement 2-valuées avec $\delta(Q_t) = 2^s$ et $\delta(b) \in \{0, 2^s\}$.

Preuve : Supposons que nous avons une paire (a, b) telle que l'équation (8.6) a au moins une solution x. Alors nous avons

$$x^{2^t+1} + (x + a)^{2^t+1} = x^{2^t}a + a^{2^t}x + a^{2^t+1} = b. \tag{8.7}$$

L'équation $x^{2^t}a+a^{2^t}x+a^{2^t+1}+b = 0$ est affine sur le corps $\mathbb{F}_{2^n}$; donc le nombre de solutions de l'équation (8.6) est soit égal à 0 soit égal au nombre de solutions de $x^{2^t}a + a^{2^t}x$ (partie linéaire de l'équation). Or $x^{2^t}a + a^{2^t}x = ax(x^{2^t-1} + a^{2^t-1})$. Donc l'espace des solutions de cette équation linéaire est $a\mathbb{F}_{2^s}$ et nous concluons que l'équation (8.6) a pour solution l'ensemble $x + a\mathbb{F}_{2^s}$. □

Grâce à ce résultat, nous déduisons le spectre différentiel des fonctions Q_t.

Corollaire 8.1. *Soit Q_t une fonction puissance avec exposant quadratique, définie dans le théorème 8.3. Le spectre différentiel de Q_t est*

$$\omega_0 = 2^n - 2^{n-s} \quad \text{et} \quad \omega_{2^s} = 2^{n-s} \text{ avec } s = \text{pgcd}(n, t).$$

8.4.3 L'exposant de Kasami

Dans cette section, nous nous intéressons aux fonctions puissances ayant un exposant de Kasami. Nous notons par K_t ces fonctions :

$$\begin{array}{rccl} K_t : & \mathbb{F}_{2^n} & \to & \mathbb{F}_{2^n} \\ & x & \mapsto & x^{2^{2t}-2^t+1} \text{ où } 2 \le t \le n/2. \end{array}$$

La quantité $2^{2t} - 2^t + 1$ est appelée *exposant de Kasami*. Dans un premier temps nous rappelons le spectre de Walsh d'une fonction avec exposant de Kasami.

Proposition 8.7. *[Kas71] Soit la fonction K_t avec exposant de Kasami. Soit f la fonction booléenne associée : $f(x) = \mathbf{Tr}(K_t(x))$. Soit $s = \text{pgcd}(n, t) = \text{pgcd}(n, 2t)$. Le spectre de Walsh de f est*

$$\{0, \pm 2^{(n+s)/2}\}.$$

Si l'on veut utiliser la proposition 8.6, afin de prouver que les monômes avec exposant de Kasami sont différentiellement 2-valués nous devons montrer que $\forall b$ si $\delta(b) \neq 0$ alors $\delta(b) \geq 2^k$. Pour cela nous commençons par rappeler un résultat qui relie l'exposant de Kasami à l'exposant quadratique.

Lemme 8.6. *Soit t un entier positif, nous avons*

$$2^{3t} + 1 = (2^t + 1)(2^{2t} - 2^t + 1). \tag{8.8}$$

De plus, si $3t \geq n$, nous pouvons remarquer que :

- *Si $3t = n + k$ avec $k \geq 0$, alors la fonction Q_{3t} correspond à la fonction Q_k.*
- *Si $3t = n$, alors K_t est dans la classe de inverse de Q_t.*

Preuve :
- L'équation (8.8) se démontre facilement en développant le deuxième terme de l'égalité.
- Dans le cas où $3t = n + k$ nous avons $2^{3t} + 1 \equiv 2^n 2^k + 1 \equiv 2^k + 1 \mod (2^n - 1)$.
- Dans le cas où $3t = n$ nous avons $(2^{2t} - 2^t + 1)(2^t + 1) \equiv 2^n + 1 \equiv 2 \mod (2^n - 1)$. Donc $(2^{2t} - 2^t + 1)(2^t + 1)(2^{n-1}) \equiv 1 \mod (2^n - 1)$.

□

Dans la section précédente nous avons montré que la fonction Q_t est différentiellement 2-valuée. A l'aide du lemme précédent, nous montrons que les fonctions K_t sont différentiellement 2-valuées.

Dans un premier temps nous rappelons un résultat bien connu sur les fonctions avec exposant de Kasami K_t.

Lemme 8.7. *[Kas71, JW93] La fonction $K_t : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ est APN si est seulement si* $\text{pgcd}(t, n) = 1$.

Par la suite, nous nous concentrons donc sur les valeurs de t telles que $s = \text{pgcd}(t, n) > 1$. De plus nous supposons que $n/\text{pgcd}(t, n)$ est impair. Ceci implique que pour tout r impair, $\text{pgcd}(2^{rt} + 1, 2^n - 1) = 1$, c'est-à-dire que $d = 2^{2t} - 2^t + 1$ est premier avec $2^n - 1$ et que K_t est une permutation du corps $\mathbb{F}_{2^n}$.

Théorème 8.4. *Soit $K_t : x \mapsto x^{2^{2t}-2^t+1}$ une fonction définie sur $\mathbb{F}_{2^n}$. Supposons que $n \neq 3t$ et $s = \text{pgcd}(n, t)$ avec n/s impair. Alors $\delta(b) \in \{0, 2^s\}$ pour tout b et donc $\delta(K_t) = 2^s$. Plus précisément, si l'équation $K_t(x) + K_t(x + 1) = b$ a une solution x l'ensemble des solutions est*

$$(y + a\mathbb{F}_{2^s})^{2^t+1} \text{ où } x = y^{2^t+1},\ x + 1 = (y + a)^{2^t+1}.$$

Preuve : Soit $b \in \mathbb{F}_{2^n}$ tel que l'équation suivante

$$x^{2^{2t}-2^t+1} + (x + 1)^{2^{2t}-2^t+1} = b \tag{8.9}$$

a au moins une solution x. Pour les valeurs de t définies dans le théorème, on a $\text{pgcd}(2^t + 1, 2^n - 1) = 1$ (voir équation (7.2)). En conséquence, il existe y et z tels que $x = y^{2^t+1}$ et $x + 1 = z^{2^t+1}$ et il existe a tel que $z = y + a$. Avec ces notations, l'équation (8.9) devient

$$y^{2^{3t}+1} + (y + a)^{2^{3t}+1} = b, \tag{8.10}$$

où $2^{3t} + 1$ est calculé modulo $(2^n - 1)$ (c'est-à-dire que si $3t = n + \ell$ avec $\ell \geq 1$ alors $2^{3t} + 1$ est équivalent à $2^\ell + 1$). Soit k défini par $k = \text{pgcd}(3t, n)$. Comme $s = \text{pgcd}(t, n)$ et 3 est premier, nous avons $k = s$ ou $k = 3s$. A partir du théorème 8.3, nous déduisons que si l'équation (8.10) a une solution y alors l'ensemble des solutions est $y + a\mathbb{F}_{2^k}$.

A partir de maintenant, nous voulons prouver que pour tout $\beta \in \mathbb{F}_{2^s}$, l'élément $(y+\beta a)^{2^t+1}$ est solution de l'équation (8.9). Soit $u = (y+\beta a)^{2^t+1}$ et $v = (y+(\beta+1)a)^{2^t+1}$. Nous avons

$$\begin{aligned} u+v &= (y+\beta a)^{2^t}a + (y+\beta a)a^{2^t} + a^{2^t+1} \\ &= ya^{2^t} + y^{2^t}a + a^{2^t+1} \\ &= y^{2^t+1} + (y+a)^{2^t+1} \\ &= x + (x+1) = 1. \end{aligned}$$

Alors l'égalité

$$\begin{aligned} u^{2^{2t}-2^t+1} + (u+1)^{2^{2t}-2^t+1} &= u^{2^{2t}-2^t+1} + v^{2^{2t}-2^t+1} \\ &= (y+\beta a)^{2^{3t}+1} + (y+(\beta+1)a)^{2^{3t}+1} = b, \end{aligned}$$

prouve que l'équation (8.10) a au moins 2^s solutions. Nous en déduisons que $\delta(b) \geq 2^s$ pour tout $\delta(b)$ non-nul. D'après la proposition 8.7, la fonction booléenne $x \mapsto \mathbf{Tr}(x^{2^{2t}-2^t+1})$ est plateau avec spectre de Walsh $\{0, \pm 2^{(n+s)/2}\}$. En utilisant la proposition 8.6 nous pouvons conclure que $\delta(b) \in \{0, 2^s\}$ pour tout b. □

Remarque 8.2. *La condition n/s impair donnée dans le théorème précédent est nécessaire. En effet il existe un certain nombre de fonctions K_t qui ne sont pas différentiellement 2-valuées. Nous donnons ici l'exemple du spectre différentiel de la fonction $K_2(x) = x^{13}$ dans le corps fini $\mathbb{F}_{2^{12}}$:*

$$\omega_0 = 2389, \quad \omega_2 = 1638, \quad \omega_4 = 1, \text{ et } \omega_{12} = 68.$$

8.4.4 Monômes avec exposant $2^{m+1} + 2^{m-1} - 1$ sur le corps $\mathbb{F}_{2^{2m}}$

En 2009, Yassir Nawaz, Kishan Chand Gupta et Guang Gong dans [NGG09] ont conjecturé que les fonctions puissances $F(x) = x^{2^{m+1}+2^{m-1}-1}$ étaient différentiellement 4-uniformes. Or, nous avons remarqué que ces fonctions étaient dans la classe cyclotomique d'une fonction avec exposant de Kasami. Dans cette section nous donnons donc une preuve que les fonctions $F(x) = x^{2^{m-1}+2^{m-1}-1}$ sont différentiellement 4-uniformes et différentiellement 2-valuées.

Proposition 8.8. *Soit $n = 2m$, avec m impair, et $d = 2^{m+1} + 2^{m-1} - 1$. La fonction $x \mapsto x^d$ définie sur le corps $\mathbb{F}_{2^n}$ est une permutation différentiellement 4-uniforme,et différentiellement 2-valuée.*

Preuve : Pour $m > 3$ l'exposant $d = 2^{m+1} + 2^{m-1} - 1$ est dans la classe cyclotomique d'un exposant de Kasami :

$$\begin{aligned} 2^{m-1} \cdot d &= 2^{2m} + 2^{2m-2} - 2^{m-1} \mod (2^{2m}-1) \\ &= 2^{2(m-1)} - 2^{m-1} + 1. \end{aligned}$$

Donc d est dans la classe cyclotomique de la fonction K_t avec $t = m-1$. Nous avons $2m = 3(m-1)$ si et seulement si $m = 3$.

Soit k tel que $m = 2k + 1$ nous avons :

$$\text{pgcd}(m - 1, 2m) = \text{pgcd}(2k, 4k + 2) = 2\text{pgcd}(k, 2k + 1) = 2.$$

D'après le théorème 8.4, la fonction $F(x) = x^d$ est différentiellement 4-uniforme. Et pour tout b, $\delta(b) \in \{0, 4\}$.

Dans le cas où $m = 3$, $d = 19$ est dans la classe de l'exposant de Kasami 13. Dans ce cas le théorème ne peut pas s'appliquer directement. Mais d est l'inverse de l'exposant quadratique $d = 5$ et nous pouvons aussi prouver que la fonction est différentiellement 4-uniforme, 2-valuées. □

Remarque 8.3. *Comme il est dit dans l'article [NGG09] la fonction puissance avec exposant* $2^{m+1} + 2^{m-1} - 1$ *est hautement non-linéaire. En dehors du cas où* $n = 6$ *cet exposant est le représentant de la classe cyclotomique.*

Remarque 8.4. *Dans le cas où* m *est pair la fonction* $F(x) = x^{2^{m+1}+2^{m-1}-1}$ *est une non-permutation APN sur le corps* $\mathbb{F}_{2^{2m}}$ *car elle est dans la classe de* K_{m-1} *et* $\text{pgcd}(m - 1, 2m) = 1$.

8.4.5 Quelques remarques sur les autres exposants : Quelles sont les fonctions différentiellement 2-valuées ?

Dans le tableau 8.1, nous donnons l'ensemble des permutations puissances différentiellement 4-uniformes. Nous remarquons dans ce tableau que les seules fonctions qui sont différentiellement 4-uniformes et différentiellement 2-valuées sont équivalentes aux fonctions avec exposant quadratique ou de Kasami. La même remarque peut être faite dans le cas général où les fonctions sont différentiellement 2^s-uniformes. Nous proposons donc la conjecture suivante :

Conjecture 8.7. *Toute fonction puissance* x^d *différentiellement 2-valuée est telle que* d *est équivalent à un exposant quadratique ou un exposant de Kasami*[4].

Dans la seconde partie de cette section, nous montrons que pour certaines tailles de corps finis il n'existe pas de monômes différentiellement 2-valués. Ces résultats renforcent la conjecture précédente.

Nous pouvons aussi remarquer que cette conjecture illustre le fait que pour les permutations puissances, il y a une décorrélation entre le spectre différentiel et le spectre de Walsh de la fonction booléenne associée. C'est-à-dire que deux fonctions peuvent avoir le même spectre de Walsh et un spectre différentiel différent.

4. Équivalent au sens défini dans le lemme 8.3.

Étude des monômes avec exposant $d = 2^{m+1} + 3$ sur $\mathbb{F}_{2^{2m}}$ avec m impair

Dans cette partie nous allons illustrer ce phénomène en montrant qu'une autre famille de fonction plateau n'est pas différentiellement 2-valuée. Plus précisément, nous donnons ici le spectre différentiel des fonctions puissances étudiées par Thomas Cusick et Hans Dobbertin [CD96].

Théorème 8.5. [CD96] *Soit $n = 2m$ un entier avec m impair. Soit d un entier pouvant prendre les valeurs suivantes :*

$$d = 2^m + 2^{(m+1)/2} + 1\,;$$
$$d = 2^{m+1} + 3.$$

Soit f_d la fonction booléenne définie par $f_d(x) = \mathbf{Tr}(x^d)$ sur le corps $\mathbb{F}_{2^n}$. Alors f_d est plateau avec spectre de Walsh $\{0, \pm 2^{\frac{n+2}{2}}\}$.

Remarque 8.5. *Les fonctions définies dans le théorème précédent sont des permutations.*

Nous rappelons que dans le cas où n est pair, une fonction puissance APN sur $\mathbb{F}_{2^n}$ ne peut pas être bijective. Ce qui signifie que les fonctions F avec d défini comme dans le théorème précédent ne peuvent pas être APN. Or comme la fonction booléenne associée à la fonction F est plateau, la proposition 8.6 implique que la fonction F est :

- soit différentiellement 4-uniforme et différentiellement 2-valuée ;
- soit $\delta(F) \geq 6$.

Le tableau 8.6 retranscrit les résultats des calculs effectués pour $n \equiv 2 \mod 4$ et $10 \leq n \leq 30$. Les résultats montrent que les deux fonctions puissances avec exposant défini comme dans le théorème 8.5 sont différentiellement 8-uniformes. De plus ces deux fonctions ont le même spectre différentiel.

TABLE 8.6 – Spectre différentiel des permutations puissances étudiées par Cusick et Dobbertin : $F_d : x \mapsto x^d$ sur le corps $\mathbb{F}_{2^n}$ avec $d = 2^m + 2^{(m+1)/2} + 1$ et $d = 2^{m+1} + 3$, pour $n = 2m$, avec m impair.

n	s	inverse	ω_0	ω_2	ω_4	ω_6	ω_8
10	41	25	698	200	76	40	10
	67	107	698	200	76	40	10
14	145	113	11504	2240	2080	448	112
	259	1613	11504	2240	2080	448	112
18	545	481	182496	40320	29248	8064	2016
	1027	26291	182496	40320	29248	8064	2016
22	2113	1985	2909184	675840	440320	135168	33792
	4099	419021	2909184	675840	440320	135168	33792
26	8321	8065	46744064	10250240	7552000	2050048	512512
	16387	6712115	46744064	10250240	7552000	2050048	512512
30	33025	32513	746098688	169164800	116187136	33832960	8458240
	65539	1073676229	746098688	169164800	116187136	33832960	8458240

À partir de ces résultats nous proposons la conjecture suivante.

Conjecture 8.8. *Soit $n = 2m$ avec m impair. Soit $F : x \mapsto x^d$ une permutation puissance définie pour les valeurs suivantes de d :*

- $d = 2^m + 2^{(m+1)/2} + 1$;
- $d = 2^{m+1} + 3$.

Alors, pour ces valeurs de d, F est différentiellement 8-uniformes et toutes les valeurs $0, 2, 4, 6$ et 8 apparaissent dans le spectre différentiel.

Non-existence de fonctions différentiellement 2-valuées pour certaines tailles de corps

Dans la conjecture 8.7, nous avons supposé que seules les fonctions puissances avec exposant quadratique ou de Kasami étaient différentiellement 2-valuées (à équivalence près). La validité de cette conjecture est renforcée par les résultats suivants qui montrent que pour certaines valeurs de n il y a un certain nombre d'exposants d pour lesquels les fonctions puissances correspondantes ne peuvent pas être différentiellement 2-valuées.

Afin de prouver que pour certaines tailles de corps et pour certaines uniformités différentielles il n'existe pas de fonctions puissances différentiellement 2-valuées, nous avons besoin du lemme générique suivant.

Lemme 8.8. *Soit p un nombre premier impair, pour tout w, nous avons $2^{p^w} \equiv 2 \mod p$.*

Preuve : Soit i un entier. Comme pour p impair $\text{pgcd}(2^{p^{i-1}}, p) = 1$ d'après le petit théorème de Fermat nous avons :

$$\left(2^{p^{i-1}}\right)^{p-1} \equiv 1 \bmod p.$$

Or $\left(2^{p^{i-1}}\right)^{p-1} = 2^{p^i - p^{i-1}}$, donc

$$2^{p^i} \equiv 2^{p^{i-1}} \bmod p \tag{8.11}$$

Pour $i = 1$, nous avons $2^p \equiv 2 \mod p$. Par récurrence sur l'équation (8.11), on en déduit que , pour tout w,

$$2^{p^w} \equiv 2 \bmod p.$$

□

Proposition 8.9. *Soient p un nombre premier et $n = p^w$ une puissance de p pour une valeur particulière de $w \geq 1$. Soit $F : x \mapsto x^d$ une permutation puissance non-linéaire sur le corps $\mathbb{F}_{2^n}$ différentiellement 2-valuée avec $\delta(F) = 2^s$. Alors, $p > 2$ et p divise $(2^{s-1} - 1)$. Et l'on a notamment,*

- *pour tout p, $\delta(F) \neq 4$;*
- *pour tout $p \neq 3$, $\delta(F) \neq 8$;*
- *pour tout $p \neq 7$, $\delta(F) \neq 16$.*

Ainsi si $p \neq 3, 7$, on a $\delta(F) \geq 2^5$

Preuve : Nous définissons l'ensemble $\mathcal{E} = \{b \in \mathbb{F}_{2^n},\ \delta(b) \neq 0\}$. Soit F une fonction différentiellement 2-valuée avec $\delta(F) = 2^s$. Comme

$$\sum_{b \in \mathbb{F}_{2^n}} \delta(b) = 2^n,$$

on a $\#\mathcal{E} = 2^{n-s}$. Mais, pour tout b, $\delta(b) = \delta(b^2)$. En conséquence, l'ensemble $\mathcal{E}$ correspond à une union de classes cyclotomiques modulo (2^n-1). De plus $b = 1$ est inclus dans l'espace $\mathcal{E}$ puisque $\delta(1) \geq 2$.

Quand $n = p^w$, la taille des classes cyclotomiques est divisible par p excepté pour la classe de 1 et de 0. On en déduit donc qu'il existe λ tel que $\#\mathcal{E} = 1 + p\lambda$ (car $0 \notin \mathcal{E}$). Ce résultat implique que $p\lambda = 2^{n-s} - 1$.

Nous pouvons remarquer que $\lambda \geq 1$ puisque si $\lambda = 0$ on a $\delta(F) = 2^n$. Ceci est impossible car nous avons supposé que F était non-linéaire. Nous obtenons donc

$$2^{p^v-s} - 1 \equiv 0 \bmod p. \tag{8.12}$$

Ceci implique donc que $p > 2$. D'après le lemme 8.8 nous avons alors que $2^{p^v} \equiv 2^s \equiv 2 \bmod p$, c'est-à-dire que p divise $2^{s-1} - 1$. □

Remarque 8.6. *La proposition précédente dit que si n est une puissance de 2, il n'existe pas de monôme de permutation non-linéaire différentiellement 2-valué.*

Proposition 8.10. *Soient $p > 2$ un nombre premier, w un entier positif et $n = 2p^w$. Soit $F : x \mapsto x^d$ une permutation puissance non-linéaire du corps $\mathbb{F}_{2^n}$ différentiellement 2-valuée. Soit s tel que $\delta(F) = 2^s$. Ces permutations existent si et seulement si p divise soit $(2^{s-2} - 1)$ soit $(3 \cdot 2^{s-2} - 1)$. Plus précisément, on a que*

- *pour tout $p \neq 5$, $\delta(F) \neq 8$;*
- *pour tout $p \notin \{3, 11\}$, $\delta(F) \neq 16$;*
- *pour tout $p \notin \{7, 23\}$, $\delta(F) \neq 32$;*
- *pour tout $p \notin \{3, 5, 47\}$, $\delta(F) \neq 64$.*

Preuve : La preuve est similaire à celle de la proposition précédente. Ici comme $n = 2p^v$, en plus des classes d'ordre divisibles par 1 et p on a la classe d'ordre 2. Soit b' un représentant de cette classe. Deux cas se présentent alors : soit $\delta(b') = 0$ soit $\delta(b') = 2^s$.

Dans le cas où $\delta(b') = 2^s$ avec les notations de la preuve précédente on a $\#\mathcal{E} = 1+2+p\lambda$. Alors $2^{2p^v} \equiv 3 \cdot 2^s \bmod p$. De plus par le lemme 8.8 $2^{2p^v} \equiv 4 \bmod p$ donc p divise $3 \cdot 2^{s-2} - 1$.

Dans le cas où $\delta(b') = 0$, avec les notations de la preuve précédente on a $\#\mathcal{E} = 1 + p\lambda$. Alors $2^{2p^m} \equiv 2^s \bmod p$. De plus par le lemme 8.8 $2^{2p^v} \equiv 4 \bmod p$ donc p divise $2^{s-2} - 1$. □

8.5 Résumé sur les fonctions différentiellement 4-uniformes

Le tableau 8.7 résume les spectres différentiels des familles infinies de monômes différentiellement 4-uniformes.

Nom	Exposant	Conditions	Spectre		
			ω_0	ω_2	ω_4
quadratique	2^t+1	$\text{pgcd}(t,n)=2$ $\text{pgcd}(2t,n)=2$	2^n-2^{n-2}	0	2^{n-2}
Kasami	$2^{2t}-2^t+1$	$n\equiv 2 \mod 4$ $n\neq 3t$ $\text{pgcd}(t,n)=2$	2^n-2^{n-2}	0	2^{n-2}
[BL10]	$2^{2k}+2^k+1$	$n=4k$	$5\cdot 2^{4k-3}-2^{3k-3}$	$2^{3k-2}(2^k+1)$	$2^{3k-3}(2^k-1)$
Inverse	$2^{n-1}-1$	n pair	$2^{n-1}+1$	$2^{n-1}-2$	1

TABLE 8.7 – Spectre différentiel des monômes différentiellement 4-uniformes

Dans la section suivante nous continuons, d'explorer les monômes avec petite uniformité différentielle. Dans nos simulations nous avons remarqué que la plupart des fonctions différentiellement 6-uniformes ont un exposant de la forme $d=2^t-1$. Parmi les fonctions avec exposant de cette forme on retrouve naturellement la fonction inverse (qui est APN si n est impair) et localement-APN sinon. Dans cette famille d'exposant on retrouve aussi la fonction quadratique $x\mapsto x^3$ qui est APN.

Nous donnons donc maintenant des propriétés générales des monômes $x\mapsto x^{2^t-1}$. En outre nous montrons qu'un certain nombre de ces fonctions ont une petite uniformité différentielle.

8.6 Les exposants 2^t-1

Dans les sections 8.2.3 et 8.2.4 nous avons vu que pour des tailles de corps suffisamment grandes, la plupart des fonctions puissances différentiellement 6-uniformes ont un exposant équivalent à un exposant de la forme 2^t-1

La seconde partie de ma thèse étant dédiée à l'étude du spectre différentiel des monômes avec petite uniformité différentielle, nous nous sommes donc intéressés aux fonctions $F(x)=x^{2^t-1}$. Les résultats présentés dans cette section ont été publiés dans [BCC11].

Soit $\mathbb{F}_{2^n}$ le corps de taille 2^n. Soit $2\leq t\leq n-1$; dans la suite de cette section nous notons par G_t les fonctions

$$\begin{array}{rccl} G_t: & \mathbb{F}_{2^n} & \to & \mathbb{F}_{2^n} \\ & x & \mapsto & x^{2^t-1} \end{array} \quad ,2\leq t\leq n-1 \tag{8.13}$$

Dans cette section nous étudions les fonctions G_t pour tout t. Pourtant nous remarquons que G_t est une permutation si et seulement si $\text{pgcd}(2^t-1,2^n-1)=1$, c'est-à-dire si et seulement si $\text{pgcd}(t,n)=1$.

8.6.1 Lien avec les polynômes linéaires

Dans un premier temps nous donnons quelques propriétés générales sur la famille des fonctions G_t.

Théorème 8.6. *Soit $G_t(x) = x^{2^t-1}$ la fonction puissance définie par (8.13). Nous avons,*

$$G_t(x+1) + G_t(x) + 1 = \frac{(x^{2^{t-1}} + x)^2}{x^2 + x}. \tag{8.14}$$

Donc, pour tout $b \in \mathbb{F}_{2^n} \setminus \{1\}$, $\delta(b)$ est égal au nombre de racines dans $\mathbb{F}_{2^n} \setminus \mathbb{F}_2$ du polynôme linéaire

$$P_b(x) = x^{2^t} + bx^2 + (b+1)x \ .$$

De plus

$$\begin{aligned} \delta(0) &= 2^{\mathrm{pgcd}(t,n)} - 2, \\ \delta(1) &= 2^{\mathrm{pgcd}(t-1,n)}. \end{aligned}$$

Et pour tout $b \in \mathbb{F}_{2^n} \setminus \mathbb{F}_2$, il existe r avec $1 \leq r \leq \min(t, n-t+1)$ tel que

$$\delta(b) = 2^r - 2.$$

Preuve : Dans un premier temps nous regardons le cas particulier où $b = 1$. On s'intéresse alors aux racines du polynôme $1 + x^{2^t-1} + (1+x)^{2^t-1}$. En multipliant ce polynôme par $x^2 + x$ on obtient le polynôme linéarisé suivant :

$$(x + x^2)(1 + x^{2^t-1} + (1+x)^{2^t-1}) = x + x^2 + x^{2^t} + x^{2^t+1} + x(1+x)^{2^t} = x^2 + x^{2^t}.$$

On en déduit donc que $\delta(1)$ correspond au nombre de racines du polynôme $P_1(x) = (x^{2^{t-1}} + x)^2$.

Maintenant considérons le cas où $b \neq 1$. Alors $x \in \mathbb{F}_{2^n} \setminus \mathbb{F}_2$ est solution de $(x+1)^d + x^d = b$ si et seulement si il est solution de $(x^{2^{t-1}} + x)^2 = (b+1)x(x+1)$. De manière équivalente $x\mathbb{F}_{2^n} \setminus \mathbb{F}_2$ est solution de $(x+1)^d + x^d = b$ si et seulement si c'est une racine du polynôme linéarisé

$$P_b(x) = x^{2^t} + bx^2 + (b+1)x.$$

Les racines $x = 0$ et $x = 1$ de (8.14) sont comptées dans $\delta(1)$. Or $P_b(0) = P_b(1) = 0$ pour tout b. Nous obtenons donc que pour $b \neq 1$, le nombre de racines de P_b dans le corps $\mathbb{F}_{2^n}$ est égal à $(\delta(b)+2)$. Comme l'ensemble des racines d'un polynôme linéarisé est un espace vectoriel, nous déduisons que

$$\forall b \in \mathbb{F}_{2^n} \setminus \{1\}, \ \ \delta(b) = 2^r - 2 \text{ avec } r \leq t.$$

De plus, en élevant P_b à la puissance 2^{n-t}, nous obtenons que toute racine de P_b est aussi une racine de $b^{2^{n-t}} x^{2^{n-t+1}} + (b^{2^{n-t}} + 1)x^{2^{n-t}} + x$. Ce qui implique que $\delta(b) = 2^r - 2$ avec $r \leq n - t + 1$. Dans le cas particulier où $b = 0$, $P_0(x) = x^{2^t} + x$, ce qui implique que $\delta(0) = 2^{\mathrm{pgcd}(t,n)} - 2$. □

Remarque 8.7. *La fonction inverse $x \mapsto x^{2^n-2}$ est dans la même classe cyclotomique que la fonction $G_{n-1} : x \mapsto x^{2^{n-1}-1}$ définie sur le corps $\mathbb{F}_{2^n}$. Ainsi en appliquant le théorème précédent on retrouve la formule bien connue du spectre différentiel de la fonction inverse (voir lemme 8.2). En effet si on applique le théorème précédent dans le cas où $t = n-1$, cela nous donne $\delta(0) = 0$ et $\delta(1) = 2$ quand n est impair et $\delta(1) = 4$ quand n est pair. Pour tout $b \notin \mathbb{F}_2$, $\delta(b) \in \{0, 2\}$.*

On obtient donc

– *si n est impair,* $\delta(G_{n-1}) = 2$ *et* $\omega_0 = 2^{n-1}, \omega_2 = 2^{n-1}$;
– *si n est pair,* $\delta(G_{n-1}) = 4$ *et* $\omega_0 = 2^{n-1} + 1$, $\omega_2 = 2^{n-1} - 2$, $\omega_4 = 1$.

L'application du théorème 8.6 nous permet de faire la remarque suivante qui nous dit que $\delta(G_t)$ prend un nombre limité de valeurs.

Remarque 8.8. *Soit $G_t(x) = x^{2^t-1}$ une fonction puissance définie par (8.13). Alors, il existe $2 \leq r \leq n$ tel que $G_t(x)$ est différentiellement (2^r-2)-uniforme ou différentiellement 2^r-uniforme. De plus, si $\delta(G_t) = 2^r$, alors cette valeur apparaît une seule fois dans le spectre différentiel, c'est-à-dire, $\omega_{2^r} = 1$. Cette valeur correspond à $\delta(1)$, ce qui implique, d'après le théorème, que $\delta(G_t) = 2^{\mathrm{pgcd}(t-1,n)}$.*

8.6.2 D'autres formulations équivalentes

Le but de cette section est d'étudier le spectre différentiel des fonctions $x \mapsto x^{2^t-1}$. Dans le théorème 8.6, nous avons montré qu'étudier le spectre différentiel de cette famille de fonctions est équivalent à trouver les racines d'un polynôme linéaire. Dans cette section nous donnons d'autres méthodes équivalentes pour résoudre ce problème.

Dans un premier temps nous allons voir le lien entre le nombre de racines de $P_{t,b}$:

$$P_{t,b} = x^{2^t} + bx^2 + (b+1)x \ .$$

et celui de son application adjointe. Ce lien nous permet dans la section 8.6.3 d'établir un lien entre le spectre différentiel de G_t et celui de G_{n+1-t}.

Dans un second temps nous remarquons que trouver les racines du polynôme $P_{t,b}$ est équivalent à résoudre un système de deux équations linéaires de degré plus petit que celui de $P_{t,b}$. Ce lien nous permet en particulier dans la section 8.6.5 de déterminer le spectre différentiel de la fonction $G_3(x) = x^7$.

Nous commençons par introduire quelques notations que nous allons utiliser dans cette partie.

Pour tout sous espace E de $\mathbb{F}_{2^n}$ (où le corps $\mathbb{F}_{2^n}$ est identifié à l'espace vectoriel $\mathbb{F}_2^n$), nous définissons le *dual* de E de la manière suivante :

$$E^{\perp} = \{\ x \mid \mathbf{Tr}(xy) = 0,\ \forall\ y \in E\ \}.$$

Nous notons par $Im(F)$ l'espace image d'une fonction F et par $Ker(F)$ le noyau d'une l'application linéaire F.

Lien avec le nombre de racines de l'application adjointe

Lemme 8.9. *Soit $t, s \geq 2$ et $s = n - t + 1$. Considérons l'application linéaire*

$$P_{t,b}(x) = x^{2^t} + bx^2 + (b+1)x, \quad b \in \mathbb{F}_{2^n}.$$

Alors le dual de $\mathcal{I}m(P_{t,b})$ est l'ensemble des α satisfaisant $P^*_{t,b}(\alpha) = 0$ où

$$P^*_{t,b}(x) = x^{2^s} + (b+1)^2x^2 + bx.$$

Dans la littérature, $P^*_{t,b}$ est appelé application adjointe *de $P_{t,b}$.*

Preuve : Par définition, $\mathcal{I}m(P_{t,b})^\perp$ consiste en l'ensemble des α tels que $\mathbf{Tr}(\alpha P_{t,b}(x)) = 0$ pour tout $x \in \mathbb{F}_{2^n}$. Nous avons

$$\begin{aligned}\mathbf{Tr}(\alpha P_{t,b}(x)) &= \mathbf{Tr}(\alpha x^{2^t}) + \mathbf{Tr}(b\alpha x^2) + \mathbf{Tr}(\alpha(b+1)x) \\ &= \mathbf{Tr}(\alpha^{2^{n-t+1}} x^2) + \mathbf{Tr}(b\alpha x^2) + \mathbf{Tr}(\alpha^2(b+1)^2 x^2) \\ &= \mathbf{Tr}(x^2(\alpha^{2^s} + \alpha^2(b+1)^2 + \alpha b)).\end{aligned}$$

Donc, α appartient au dual de l'image de $P_{t,b}$ si et seulement si $\alpha^{2^s} + \alpha^2(b+1)^2 + \alpha b = 0$, c'est-à-dire, si et seulement si α est une racine de $P^*_{t,b}$. □

Système d'équations linéaires

Le théorème suivant nous donne une information nouvelle nous permettant de trouver une borne supérieure sur l'uniformité différentielle des fonctions G_t.

Théorème 8.7. *Soit $t, s \geq 2$ et $s = n - t + 1$. Soit $P_{t,b}$ et $P^*_{t,b}$ comme défini dans le lemme 8.9. Alors*

$$\dim Ker(P_{t,b}) = \dim Ker(P^*_{t,b}).$$

Donc, la dimension de $Ker(P_{t,b})$ peut être déterminée en résolvant l'équation

$$x^{2^s} + (b+1)^2 x^2 + bx = 0, \quad où\ s = n - t + 1.$$

Preuve : Soit κ la dimension de l'espace image de $P_{t,b}$. Il est bien connu que $n = \kappa + \dim Ker(P_{t,b})$. D'autre part, le lemme 8.9 dit que $\alpha \in \mathcal{I}m(P_{t,b})^\perp$ est dans le dual de l'image de $P_{t,b}$ si et seulement si $P^*_{t,b}(\alpha) = 0$. Nous déduisons que

$$n - \kappa = \dim Ker(P^*_{t,b}) = \dim Ker(P_{t,b}).$$

□

Le théorème suivant nous dit qu'étudier le spectre différentiel des fonctions $x \mapsto x^{2^t-1}$ est équivalent à la résolution d'un système d'équations linéaires.

Théorème 8.8. *Pour tout $t \geq 2$, nous définissons l'équation suivante :*

$$E_b \ : \ x^{2^t} + bx^2 + (b+1)x = 0, \ b \in \mathbb{F}_{2^n}.$$

Soit N_b le nombre de solutions de E_b dans $\mathbb{F}_{2^n} \setminus \mathbb{F}_2$. Soit M_b le nombre de solutions dans $\mathbb{F}^*_{2^n}$ du système

$$\begin{cases} y^{2^{t-1}} + \cdots + y^2 + y(b+1) &= 0 \\ \mathbf{Tr}(y) &= 0 \end{cases}$$

Alors $N_b = 2 \times M_b$.

Preuve : Remarquons que

$$\begin{aligned}x^{2^t} + bx^2 + (b+1)x &= x^{2^t} + x + b(x^2 + x) \\ &= (x^2+x)^{2^{t-1}} + (x^2+x)^{2^{t-2}} + \cdots + (x^2+x) + b(x^2+x) \\ &= y^{2^{t-1}} + y^{2^{t-2}} + \cdots y^2 + y(b+1), \text{ avec } y = x^2 + x.\end{aligned}$$

Nous nous intéressons au nombre de solutions de E_b qui ne sont pas dans $\mathbb{F}_2$. Ce problème est équivalent au calcul du nombre de solutions non-nulles y de

$$y^{2^{t-1}} + y^{2^{t-2}} + \cdots y^2 + y(b+1) = 0$$

pour lesquelles l'équation $x^2 + x + y = 0$ a 2 solutions. Cette dernière condition est vraie si et seulement si $\mathbf{Tr}(y) = 0$. On a alors 2 solutions x_1 et $x_2 = x_1 + 1$ qui vérifient $x_i^2 + x_i = y$. □

Remarque 8.9. *Dans le théorème 8.8, la quantité b peut prendre toutes les valeurs du corps $\mathbb{F}_{2^n}$. Or dans le théorème 8.6, P_b est défini pour $b \neq 1$. Nous faisons ici le lien entre M_b et $\delta(b)$.*

Pour tout $b \neq 1$, nous avons $N_b = \delta(b)$. Si $b = 1$, $P_1(x) = x^{2^t} + x^2$ et le nombre de racines de P_1 dans $\mathbb{F}_{2^n}$ est égal à

$$N_1 + 2 = 2^{\mathrm{pgcd}(t-1,n)} = \delta(1).$$

Donc $M_1 = \delta(1)/2 - 1$.

8.6.3 Une propriété de symétrie

Soit $G_t(x)$ la fonction définie par $G_t(x) = x^{2^t-1}$ sur le corps $\mathbb{F}_{2^n}$. Dans cette partie, nous établissons que les spectres différentiels de G_t et G_s, où $t, s \geq 2$ et $s = n - t + 1$, sont reliés entre eux.

Le résultat principal de cette section est le théorème suivant. La preuve de ce théorème est longue et très technique. Cette preuve est détaillée dans la section 8.6.4.

Théorème 8.9. *Pour tout μ avec $2 \leq \mu \leq n-1$, nous définissons*

$$S_\mu^i = \{\ b \mid \dim Ker(P_{\mu,b}) = i\ \} \text{ avec } 1 \leq i \leq \mu\ .$$

Alors pour tout $s, t \geq 2$ avec $t = n - s + 1$ et pour tout i, nous avons $\#S_s^i = \#S_t^i$.

Le théorème 8.9 n'est pas très explicite en lui-même. Dans le corollaire suivant nous donnons une version plus explicite de ce théorème. C'est-à-dire nous donnons la méthode pour calculer le spectre différentiel de G_{n-t+1} à partir de celui de G_t.

Corollaire 8.2. *Nous notons par $\delta_\mu(b)$, $b \in \mathbb{F}_{2^n}$, la quantité $\delta(b)$ correspondante à la fonction $G_\mu : x \mapsto x^{2^\mu-1}$ ($\mu = s$ ou $\mu = t$). Alors, pour tout $s, t \geq 2$ avec $t = n - s + 1$, nous avons*

$$\begin{aligned} \delta_s(0) &= \delta_t(1) - 2 = 2^{\mathrm{pgcd}(t-1,n)} - 2 \\ \delta_s(1) &= \delta_t(0) + 2 = 2^{\mathrm{pgcd}(t,n)}. \end{aligned}$$

et nous avons l'égalité entre les deux "multi-ensembles"[5] suivants :

$$\{\delta_s(b), b \in \mathbb{F}_{2^n} \setminus \mathbb{F}_2\} = \{\delta_t(b), b \in \mathbb{F}_{2^n} \setminus \mathbb{F}_2\}. \tag{8.15}$$

5. Le terme "multi-ensemble" correspond à un ensemble où les répétitions sont possibles

De plus G_t et G_s ont le même spectre différentiel si et seulement si

$$\text{pgcd}(s, n) = \text{pgcd}(t, n) = 1\,^{6}.$$

Dans tous les cas, G_t est localement-APN[7] si et seulement si G_s est localement-APN.

Preuve : Puisque $s = n - t + 1$, nous avons

$$\text{pgcd}(s, n) = \text{pgcd}(t - 1, n) \text{ et } \text{pgcd}(s - 1, n) = \text{pgcd}(t, n).$$

Alors, en appliquant le théorème 8.6, nous obtenons

$$\begin{cases} \delta_s(0) = 2^{\text{pgcd}(s,n)} - 2 = 2^{\text{pgcd}(t-1,n)} - 2 = \delta_t(1) - 2 \\ \delta_s(1) = 2^{\text{pgcd}(s-1,n)} = 2^{\text{pgcd}(t,n)} = \delta_t(0) + 2, \end{cases}$$

c'est-à-dire

$$\{\dim Ker(P_{t,0}), \dim Ker(P_{t,1})\} = \{\dim Ker(P_{s,0}), \dim Ker(P_{s,1})\}\ .$$

A partir du théorème 8.9, nous déduisons que

$$\#\{\ b \in \mathbb{F}_{2^n} \setminus \mathbb{F}_2 \mid \dim Ker(P_{t,b}) = i\ \} = \#\{\ b \in \mathbb{F}_{2^n} \setminus \mathbb{F}_2 \mid \dim Ker(P_{s,b}) = i\ \}\ .$$

L'égalité (8.15) est alors une conséquence directe du théorème 8.6, puisque

$$\{\delta_\mu(b), b \in \mathbb{F}_{2^n} \setminus \mathbb{F}_2\} = \{2^{\kappa(b)} - 2,\ \kappa(b) = \dim Ker(P_{\mu,b}),\ b \in \mathbb{F}_{2^n} \setminus \mathbb{F}_2\}\ .$$

Remarquons que $\delta_s(0) = \delta_t(0)$ si et seulement si $\delta_s(1) = \delta_t(1)$. Nous déduisons alors que G_t et G_s ont le même spectre différentiel si et seulement si $\delta_s(0) = \delta_t(0)$. Or

$$\delta_s(0) = 2^{\text{pgcd}(s,n)} - 2 \quad \text{et} \quad \delta_t(0) = 2^{\text{pgcd}(t,n)} - 2.$$

La condition précédente est vérifiée si et seulement si $\text{pgcd}(t, n) = \text{pgcd}(s, n) = 1$. Cette condition n'est jamais vérifiée quand n est pair car dans ce cas soit s soit t est pair, c'est-à-dire que 2 divise $\text{pgcd}(t, n)$ ou $\text{pgcd}(s, n)$.

□

Corollaire 8.3. *Soit n et $t < n$ tels que $G_t : x \mapsto x^{2^t-1}$ est APN sur le corps $\mathbb{F}_{2^n}$. Soit $s = n - t + 1$. Alors,*

- *si n est impair, G_t et G_s sont des permutations APN ;*
- *si n est pair, G_t n'est pas une permutation et G_s est une permutation différentiellement 4-uniformes, localement-APN avec spectre différentiel :*

$$\omega_4 = 1, \quad \omega_2 = 2^{n-1} - 2 \quad \text{et} \quad \omega_0 = 2^{n-1} + 1.$$

6. Ce cas ne peut apparaître que pour n impair
7. Une fonction est localement-APN si $\forall b \neq 0, 1$ on a $\delta(b) \leq 2$(voir définition 8.2)

Preuve : Supposons G_t est APN, alors $\forall b \in \mathbb{F}_{2^n}$, nous avons $\delta_t(b) \in \{0, 2\}$. En particulier nous avons $\delta_t(1) \leq 2$ et $\delta_t(0) \leq 2$. En utilisant les résultats du théorème 8.6, nous déduisons donc que $\text{pgcd}(n, t-1) = 1$ et $\text{pgcd}(n, t) \in \{1, 2\}$.

Si n est impair, nous avons $\text{pgcd}(n, t) = 1$ c'est-à-dire que $\delta_t(0) = 0$. En utilisant le corollaire 8.2, nous obtenons : $\delta_s(0) = 0$, $\delta_s(1) = 2$ et $\delta_s(b) \in \{0, 2\}$ pour tout $b \in \mathbb{F}_{2^n}$.

Nous venons de montrer que si G_t est APN alors G_t et G_s sont des permutations APN.

Dans le cas où n est pair, il est bien connu qu'une fonction puissance APN ne peut pas être une permutation. Comme G_t est APN d'après le théorème 8.6, nous avons $\delta_t(0) = 2$ et $\text{pgcd}(n, t) = 2$. D'après le corollaire 8.2, nous en déduisons que $\delta_s(0) = 0$ (car $\delta_t(1) \neq 0$ implique que $\delta_t(1) = 2$) et $\delta_s(1) = 4$. Le calcul du spectre différentiel complet de G_s est une conséquence directe du corollaire 8.2. □

Exemple 8.3. *Dans cet exemple nous illustrons un lien entre la fonction inverse et la fonction avec exposant quadratique $x \mapsto x^3$.*

Pour $t = 2$, nous avons $G_t(x) = x^3$. Il est bien connu que cette fonction est APN quelle que soit la parité de n(c'est une permutation si et seulement si n est impair). Soit $s = n - t + 1 = n - 1$; on a que G_s est dans la classe de la fonction inverse. C'est-à-dire que le spectre différentiel de G_s est le même que celui de la fonction inverse. En appliquant le corollaire 8.3 on retrouve le résultat bien connu montré par Kaisa Nyberg [Nyb94] qui dit que la fonction inverse est une permutation APN quand n est impair et qu'elle est différentiellement 4-uniformes localement-APN quand n est pair.

En utilisant le corollaire 8.2, nous pouvons extraire le même type de résultat pour les fonctions différentiellement 4-uniformes.

Corollaire 8.4. *Soit n et $t < n$ deux entiers tels que la fonction $G_t : x \mapsto x^{2^t-1}$ sur le corps $\mathbb{F}_{2^n}$ est différentiellement 4-uniformes. Alors, n est pair et G_t est une permutation avec le spectre différentiel suivant :*

$$\omega_4 = 1\ , \quad \omega_2 = 2^{n-1} - 2 \quad \text{et} \quad \omega_0 = 2^{n-1} + 1.$$

De plus, pour $s = n - t + 1$, G_s est APN (non-permutation).

Preuve : Comme $\delta(G_t) = 4$, à partir du théorème 8.6 nous avons que $\delta_t(b) \neq 4$ pour tout $b \neq 1$ et que seul $\delta_t(1) = 4$. C'est-à-dire que $\text{pgcd}(n, t-1) = 2$ et que $\omega_4 = 1$. Nous remarquons alors que n doit être pair pour satisfaire $\text{pgcd}(n, t-1) = 2$. De plus comme $\text{pgcd}(n, t-1)$ et $\text{pgcd}(n, t)$ ne peuvent pas être tous les deux égaux à 2, nous déduisons aussi que G_t est une permutation localement-APN. En utilisant le corollaire 8.2, nous avons que $\delta_s(0) = 2$ et que $\delta_s(1) = 2$, ce qui implique que G_s est APN. □

8.6.4 Preuve du théorème 8.9 sur la propriété de symétrie

Afin de prouver le théorème 8.9 nous avons besoin d'introduire quelques lemmes intermédiaires.

Lemme 8.10. *Soit $s, t \geq 2$ avec $t = n - s + 1$. Soit π une permutation de $\mathbb{F}_{2^n}^* \times \mathbb{F}_{2^n}$ définie par*

$$\pi(a, b) = \left(a^{2^s}, \frac{ab}{a^{2^s}} + 1 \right) .$$

Alors, pour tout (a, b) *dans* $\mathbb{F}_{2^n}^* \times \mathbb{F}_{2^n}$*, l'élément* $(\alpha, \beta) = \pi(a, b)$ *satisfait* $P_{s,\beta}^*(\alpha) = P_{t,b}^*(a)$.

Preuve : Dans un premier temps, nous montrons que π est une permutation de $\mathbb{F}_{2^n}^* \times \mathbb{F}_{2^n}$.

En effet, $\pi\left(\mathbb{F}_{2^n}^* \times \mathbb{F}_{2^n}\right) \subset \mathbb{F}_{2^n}^* \times \mathbb{F}_{2^n}$. Soit p la fonction définie par

$$p(\alpha, \beta) = \left(\alpha^{2^{n-s}}, \frac{\alpha(\beta+1))}{\alpha^{2^{n-s}}}\right) .$$

Alors p est l'inverse de π. En effet, $(\alpha^{2^{n-s}})^{2^s} = \alpha$ et il peut être calculé facilement que

$$\pi(p(\alpha, \beta)) = \left(\alpha, \frac{\alpha^{2^{n-s}}\alpha(\beta+1)}{\alpha\alpha^{2^{n-s}}} + 1\right) = (\alpha, \beta).$$

Nous en déduisons que π est une permutation et que p est l'inverse de la fonction π.

En utilisant les égalités $(\beta+1)^2 = \frac{a^2b^2}{a^{2^{s+1}}}$ et $s + t = n + 1$, nous en déduisons que

$$\begin{aligned} P_{s,\beta}^*(\alpha) &= (a^{2^s})^{2^t} + (a^{2^s})^2(\beta+1)^2 + (a^{2^s})\beta \\ &= a^2 + a^2b^2 + ab + a^{2^s} \\ &= P_{t,b}^*(a) . \end{aligned}$$

□

Lemme 8.11. *Soit* $s, t \geq 2$ *avec* $t = n - s + 1$*. Soit* $b \in \mathbb{F}_{2^n}$ *et soit* $a \in \mathbb{F}_{2^n}^*$ *tels que* $P_{t,b}^*(a) = 0$*. Alors* $\dim Ker(P_{t,b}^*) = \dim Ker(P_{s,\beta}^*)$*, où* $\beta = 1 + ab/a^{2^s}$*.*

Preuve : Dans un premier temps nous rappelons l'expression des polynômes $P_{t,b}^*$ et $P_{s,\beta}^*$:

$$P_{t,b}^*(x) = x^{2^s} + x^2(b+1)^2 + xb \quad \text{et} \quad P_{s,\beta}^*(x) = x^{2^t} + x^2(\beta+1)^2 + x\beta.$$

D'après le théorème 8.7 nous avons $\dim Ker(P_{t,b}) = \dim Ker(P_{t,b}^*)$ et $\{0, 1\}$ est inclus dans le noyau de $P_{t,b}$. Nous en déduisons que pour tout $b \notin \mathbb{F}_2$ il existe $\gamma \in \mathbb{F}_{2^n} \setminus \{0, 1\}$ tel que $P_{t,b}^*(\gamma) = 0$. En effet, $P_{t,b}^*(1) = b^2 + b = 0$ si et seulement si $b \in \mathbb{F}_2$.

Nous allons dans un premier temps, traiter le cas où $a = 1$. Ce cas apparaît seulement si b est égal à 0 ou à 1. Pour $\beta = b + 1$, d'après le lemme 8.10 nous avons $P_{s,\beta}^*(1) = 0$ puisque $\pi(1, b) = (1, b + 1)$.

- Pour $b = 0$ on a $P_{t,0}^*(x) = x^{2^s} + x^2 = P_{s,1}(x)$.
- Pour $b = 1$ on a $P_{t,1}^*(x) = x^{2^s} + x = P_{s,0}(x)$. Alors, nous pouvons conclure que pour $a = 1$, si b est tel que $P_{t,b}^*(1) = 0$ et $\beta = b + 1$ alors d'après le théorème 8.7 nous avons $\dim Ker(P_{t,b}^*) = \dim Ker(P_{s,\beta}) = \dim Ker(P_{s,b+1}^*)$.

À partir de maintenant nous supposons que $a \notin \mathbb{F}_2$. Avec $x = ay$, l'équation $P_{t,b}^*(x) = 0$ est équivalente à

$$a^{2^s}y^{2^s} + a^2y^2(b+1)^2 + ayb = 0$$

c'est-à-dire à

$$a^{2^s}\left(y^{2^s} + \frac{a^2(b+1)^2}{a^{2^s}}y^2 + y\frac{ab}{a^{2^s}}\right) = 0.$$

Comme

$$\frac{a^2(b+1)^2}{a^{2^s}} + 1 = \frac{ab}{a^{2^s}}$$

car c'est equivalent à

$$a^{2^s} + a^2(b+1)^2 + ab = 0, \ i.e., \ P^*_{t,b}(a) = 0,$$

nous avons

$$\beta = \frac{a^2(b+1)^2}{a^{2^s}} \text{ et } \beta + 1 = \frac{ab}{a^{2^s}}.$$

En remplaçant dans l'équation nous obtenons que $P^*_{t,b}(x) = 0$ est équivalent à

$$P_{s,\beta}(y) = y^{2^s} + \beta y^2 + (\beta+1)y = 0.$$

En conséquence $\dim Ker(P_{s,\beta}) = \dim Ker(P^*_{t,b})$. En utilisant le théorème 8.7, comme $\dim Ker(P_{s,\beta}) = \dim Ker(P^*_{s,\beta})$ nous pouvous conclure que $\dim Ker(P^*_{t,b}) = \dim Ker(P^*_{s,\beta})$. □

À partir de ces résultats nous pouvons revenir sur la preuve du théorème 8.9.

Preuve : **Preuve du théorème 8.9**

Nous rappelons la notation suivante

$$S^i_\mu = \{ \ b \in \mathbb{F}_{2^n} \mid \dim Ker(P_{\mu,b}) = i \ \} \text{ où } \mu = t \text{ ou } s \ .$$

Le problème consiste alors à montrer que pour tout i,

$$\#S^i_t = \#S^i_s.$$

Pour tout $2 \leq \mu \leq n-1$ et pour tout $1 \leq i \leq \mu$, nous définissons

$$\mathcal{E}^i_\mu = \{(a,b) \in \mathbb{F}^*_{2^n} \times \mathbb{F}_{2^n} \mid P^*_{\mu,b}(a) = 0 \text{ et } \dim Ker(P_{\mu,b}) = i \ \} \ .$$

À partir du théorème 8.7, nous savons que $\dim Ker(P_{\mu,b}) = \dim Ker(P^*_{\mu,b})$. Alors,

$$\mathcal{E}^i_\mu = \{(a,b) \in \mathbb{F}^*_{2^n} \times \ \mathbb{F}_{2^n} | \ P^*_{\mu,b}(a) = 0 \text{ et } \dim Ker(P^*_{\mu,b}) = i \ \}.$$

Pour tout $b \in S^i_\mu$, on a $\dim Ker(P^*_{\mu,b}) = i$ avec $a = 0 \in Ker(P^*_{\mu,b})$ donc on a $2^i - 1$ valeurs $a \neq 0$ dans $Ker(P^*_{\mu,b})$. Pour un b fixé, on a donc $2^i - 1$ paires (a,b) dans $\mathcal{E}^i_\mu$ qui vérifient

$$\#\mathcal{E}^i_\mu = (2^i - 1)\#S^i_\mu \ . \tag{8.16}$$

Nous allons utiliser le lemme 8.10. Nous rappelons que π est la permutation de $\mathbb{F}^*_{2^n} \times \mathbb{F}_{2^n}$ définie par

$$\pi(a,b) = \left(a^{2^s}, \frac{ab}{a^{2^s}} + 1\right).$$

Nous avons donc,

$$\begin{aligned}
\mathcal{E}^i_t &= \{(a,b) \in \mathbb{F}^*_{2^n} \times \mathbb{F}_{2^n} \mid P^*_{t,b}(a) = 0 \text{ et } \dim Ker(P^*_{t,b}) = i \ \}, \\
\mathcal{E}^i_s &= \{(\alpha,\beta) \in \mathbb{F}^*_{2^n} \times \mathbb{F}_{2^n} \mid P^*_{s,\beta}(\alpha) = 0 \text{ et } \dim Ker(P^*_{s,\beta}) = i \ \} \\
&= \{(\alpha,\beta) = \pi(a,b), (a,b) \in \mathcal{E}^i_t\} \ .
\end{aligned}$$

En effet, comme π est une permutation, tout couple (α, β) est identifié à un seul couple (a, b). À partir du lemme 8.10, nous avons $P^*_{s,\beta}(\alpha) = P^*_{t,b}(a)$. De plus d'après le lemme 8.11, nous avons $\dim Ker(P^*_{t,b}) = \dim Ker(P^*_{s,\beta})$, où β est calculé à partir de a et b, pour tout a tel que $P^*_{t,b}(a) = 0$.

En d'autres termes, pour toute paire $(a, b) \in \mathcal{E}^i_t$ il correspond une unique paire $(\alpha, \beta) \in \mathcal{E}^i_s$. Nous obtenons finalement que $\#\mathcal{E}^i_s = \#\mathcal{E}^i_t$. D'après l'équation (8.16), nous complétons la preuve en remarquant que

$$\#S^i_s = \#S^i_t.$$

□

8.6.5 La fonction $x \mapsto x^7$

Dans cette section nous nous intéressons à l'étude du spectre différentiel de la fonction $G_3 : x \mapsto x^7$. Dans cette partie nous montrons que cette fonction est différentiellement 6-uniformes et nous donnons le spectre différentiel complet de cette fonction.

Définition 8.4. *Nous notons par $K(1)$ la somme de Kloosterman définie par*

$$K(1) = \sum_{x \in \mathbb{F}_{2^n}} (-1)^{\mathbf{Tr}(x^{-1}+x)}, \tag{8.17}$$

où par convention $\mathbf{Tr}(x^{-1}) = \mathbf{Tr}(x^{2^n-2}) = 1$ pour $x = 0$. Le calcul de cette somme pour $x = 0$ peut se faire en fixant la convention suivante : $(-1)^{\mathbf{Tr}(x^{-1})} = 1$ pour $x = 0$.

Proposition 8.11. *[Car69, Formula (6.8)] Soit $K(1)$ la somme de Kloosterman définie par (8.17). Alors,*

$$K(1) = 1 + \frac{(-1)^{n-1}}{2^{n-1}} \sum_{i=0}^{\lfloor \frac{n}{2} \rfloor} (-1)^i \binom{n}{2i} 7^i.$$

Le théorème 8.10 est le résultat principal de cette section. La preuve de ce théoreme est longue et nécessite l'introduction d'un certain nombre de résultats préliminaires. La section 8.6.6 est dédiée à la présentation de ces résultats préliminaires ainsi qu'à la preuve du théorème. Les résultats suivants sont des conséquences de ce théorème.

Théorème 8.10. *Soit $G_3 : x \mapsto x^7$ sur le corps $\mathbb{F}_{2^n}$ avec $n \geq 4$.*

- *si n est impair, le spectre différentiel de G_3 est*

$$\omega_6 = \tfrac{2^{n-2}+1}{6} - \tfrac{K(1)}{8}, \quad \omega_4 = 0, \quad \omega_2 = 2^{n-1} - 3\omega_6, \qquad \omega_0 = 2^{n-1} + 2\omega_6.$$

- *si n est pair, le spectre différentiel de G_3 est*

$$\omega_6 = \tfrac{2^{n-2}-4}{6} + \tfrac{K(1)}{8}, \quad \omega_4 = 1, \quad \omega_2 = 2^{n-1} - 3\omega_6 - 2, \quad \omega_0 = 2^{n-1} + 2\omega_6 + 1.$$

où $K(1)$ est la somme de Kloosterman définie dans la définition 8.4. En particulier, G_3 est différentiellement 6-uniformes pour tout $n \geq 6$.

En combinant les résultats du théorème 8.10 et du corollaire 8.2, nous en déduisons le spectre différentiel de $G_{n-2} : x \mapsto x^{2^{n-2}-1}$ sur le corps $\mathbb{F}_{2^n}$.

Corollaire 8.5. *Soit la fonction $G_{n-2} : x \mapsto x^{2^{n-2}-1}$ définie sur le corps $\mathbb{F}_{2^n}$ pour $n \geq 6$. Nous avons :*

- *si* $\text{pgcd}(n, 3) = 1$*, alors G_{n-2} est différentiellement 6-uniformes et pour tout $b \in \mathbb{F}_{2^n}$, $\delta(b) \in \{0, 2, 6\}$. De plus, son spectre différentiel est donné par :*

$$\begin{aligned} \omega_6 &= \begin{cases} \frac{2^{n-2}+1}{6} - \frac{K(1)}{8} & \text{si } n \text{ est impair} \\ \frac{2^{n-2}-4}{6} + \frac{K(1)}{8} & \text{si } n \text{ est pair;} \end{cases} \\ \omega_2 &= 2^{n-1} - 3\omega_6; \\ \omega_0 &= 2^{n-1} + 2\omega_6 \, . \end{aligned}$$

- *Si 3 divise n, alors G_{n-2} est différentiellement 8-uniformes et pour tout $b \in \mathbb{F}_{2^n}$, $\delta(b) \in \{0, 2, 6, 8\}$. De plus, son spectre différentiel est donné par :*

$$\begin{aligned} \omega_8 &= 1; \\ \omega_6 &= \begin{cases} \frac{2^{n-2}-5}{6} - \frac{K(1)}{8} & \text{si } n \text{ est impair} \\ \frac{2^{n-2}-10}{6} + \frac{K(1)}{8} & \text{si } n \text{ est pair;} \end{cases} \\ \omega_2 &= 2^{n-1} - 3\omega_6 - 4; \\ \omega_0 &= 2^{n-1} + 2\omega_6 + 3 \, . \end{aligned}$$

Preuve : Nous appliquons le corollaire 8.2, pour $t = 3$ et $s = n - 2$ où $3 = n - (n - 2) + 1$.

On note par $\delta_3(b)$ la valeur de $\delta(b)$ pour la fonction G_3 et par $\delta_{n-2}(b)$ la valeur de $\delta(b)$ pour la fonction G_{n-2}. D'après le corollaire 8.2, nous avons :

- Si $\text{pgcd}(3, n) = 1$ alors $\delta_3(0) = 0$ et $\delta_{n-2}(1) = 2$.
- Sinon, $\delta_3(0) = 6$ et $\delta_{n-2}(1) = 8$.

De plus

- Si n pair, $\delta_3(1) = 4$ et $\delta_{n-2}(0) = 2$
- Si n impair, $\delta_3(1) = 2$ et $\delta_{n-2}(0) = 0$

Toujours d'après le corollaire 8.2 nous avons que les spectres différentiels de ces fonctions sont égaux "à $\delta(0)$ et $\delta(1)$ près". Dans cette preuve nous notons par $(\lambda_0, \lambda_2, \lambda_4, \lambda_6)$ (resp. $(\mu_0, \mu_2, \mu_4, \mu_6)$) le spectre différentiel de G_3 sur le corps $\mathbb{F}_{2^n}$ quand n est impair (resp. n pair).

D'après le théorème 8.10, pour n impair, nous avons

$$\lambda_6 = \tfrac{2^{n-2}+1}{6} - \tfrac{K(1)}{8}, \quad \lambda_4 = 0, \quad \lambda_2 = 2^{n-1} - 3\lambda_6, \qquad \lambda_0 = 2^{n-1} + 2\lambda_6, \tag{8.18}$$

et pour n pair,

$$\mu_6 = \tfrac{2^{n-2}-4}{6} + \tfrac{K(1)}{8}, \quad \mu_4 = 1, \quad \mu_2 = 2^{n-1} - 3\mu_6 - 2, \quad \mu_0 = 2^{n-1} + 2\mu_6 + 1. \tag{8.19}$$

Nous séparons notre analyse suivant quatre cas.

1. **Si** $\text{pgcd}(3, n) = 1$ **et** n **est impair**, nous avons

$$\big(\delta_3(0), \delta_3(1)\big) = \big(0, 2\big) \quad \text{et} \quad \big(\delta_{n-2}(0), \delta_{n-2}(1)\big) = \big(0, 2\big).$$

Nous obtenons

$$\omega_i = \lambda_i \quad \forall \, i.$$

C'est-à-dire,

$$\omega_6 = \tfrac{2^{n-2}+1}{6} - \tfrac{K(1)}{8}, \quad \omega_4 = 0, \quad \omega_2 = 2^{n-1} - 3\omega_6, \quad \omega_0 = 2^{n-1} + 2\omega_6,$$

et pour $i \geq 8$, $\omega_i = 0$.

2. **Si** $\text{pgcd}(3, n) = 1$ **et** n **pair**, nous avons

$$\big(\delta_3(0), \delta_3(1)\big) = \big(0, 4\big) \text{ et } \big(\delta_{n-2}(0), \delta_{n-2}(1)\big) = \big(2, 2\big).$$

Nous obtenons,

$$\omega_6 = \mu_6, \quad \omega_4 = \mu_4 - 1, \quad \omega_2 = \mu_2 + 2, \quad \omega_0 = \mu_0 - 1.$$

C'est-à-dire,

$$\omega_6 = \tfrac{2^{n-2}-4}{6} + \tfrac{K(1)}{8}, \quad \omega_4 = 0, \quad \omega_2 = 2^{n-1} - 3\omega_6, \quad \omega_0 = 2^{n-1} + 2\omega_6,$$

et pour $i \geq 8$, $\omega_i = 0$.

3. **Si** $\text{pgcd}(3, n) = 3$ **et** n **impair**, nous avons

$$\big(\delta_3(0), \delta_3(1)\big) = \big(6, 2\big) \quad \text{et} \quad \big(\delta_{n-2}(0), \delta_{n-2}(1)\big) = \big(0, 8\big).$$

Nous obtenons,

$$\omega_8 = 1, \quad \omega_6 = \lambda_6 - 1, \quad \omega_4 = \lambda_4, \quad \omega_2 = \lambda_2 - 1, \quad \omega_0 = \omega_0 + 1.$$

C'est-à-dire,

$\omega_8 = 1, \quad \omega_4 = 0,$

$\omega_6 = \tfrac{2^{n-2}+1}{6} - \tfrac{K(1)}{8} - 1,$

$\omega_2 = 2^{n-1} - 3\lambda_6 - 1 = 2^{n-1} - 3(\omega_6 + 1) - 1 = 2^{n-1} - 3\omega_6 - 4,$

$\omega_0 = 2^{n-1} - 2\lambda_6 + 1 = 2^{n-1} + 2(\omega_6 + 1) + 1 = 2^{n-1} + 2\omega_6 + 3,$

et pour $i \geq 8$, $\omega_i = 0$.

4. **Si** $\text{pgcd}(3, n) = 3$ **et** n **pair**, nous avons

$$\big(\delta_3(0), \delta_3(1)\big) = \big(6, 4\big) \quad \text{et} \quad \big(\delta_{n-2}(0), \delta_{n-2}(1)\big) = \big(2, 8\big).$$

Nous obtenons,

$$\omega_8 = \omega_6 = \mu_6 - 1, \quad \omega_4 = \mu_4 - 1, \quad \omega_2 = \mu_2 + 1, \quad \omega_0 = \mu_0.$$
1,

C'est-à-dire,

$\omega_8 = 1, \quad \omega_4 = 0,$

$\omega_6 = \tfrac{2^{n-2}-4}{6} - \tfrac{K(1)}{8} - 1,$

$\omega_2 = 2^{n-1} - 3\mu_6 - 1 = 2^{n-1} - 3(\omega_6 + 1) - 1 = 2^{n-1} - 3\omega_6 - 4,$

$\omega_0 = 2^{n-1} - 2\mu_6 + 1 = 2^{n-1} + 2(\omega_6 + 1) + 1 = 2^{n-1} + 2\omega_6 + 3,$

et pour $i \geq 8$, $\omega_i = 0$.

□

8.6.6 Preuve du théorème 8.10 sur le spectre différentiel de la fonction $x \mapsto x^7$

Lemme 8.12. [BRS67] *Soient $a \in \mathbb{F}_{2^n}$ et $b \in \mathbb{F}_{2^n}^*$, l'équation $x^3 + ax + b = 0$ a une solution unique dans $\mathbb{F}_{2^n}$ si et seulement si $\mathbf{Tr}(a^3/b^2) \neq \mathbf{Tr}(1)$. En particulier, si cette équation a 3 solutions distinctes dans $\mathbb{F}_{2^n}$, alors $\mathbf{Tr}(a^3/b^2) = \mathbf{Tr}(1)$.*

Proposition 8.12. [KHCH96, Appendice] *Soient $a \in \mathbb{F}_{2^n}$, $f_a(x) = x^3 + x + a$ et*

$$M_i = \#\{\ a \in \mathbb{F}_{2^n}^* \mid f_a(x) = 0 \text{ a précisément } i \text{ solutions dans } \mathbb{F}_{2^n}\ \}.$$

Si n est impair, nous avons

$$M_0 = \frac{2^n + 1}{3},\ M_1 = 2^{n-1} - 1,\ M_3 = \frac{2^{n-1} - 1}{3}$$

et si n est pair, nous avons

$$M_0 = \frac{2^n - 1}{3},\ M_1 = 2^{n-1},\ M_3 = \frac{2^{n-1} - 2}{3}.$$

Le résultat suivant est un résultat proche de celui du théorème 8.10.

Théorème 8.11. *Soit*

$$P_b(x) = x^8 + bx^2 + (b+1)x,\ b \in \mathbb{F}_{2^n} \setminus \{1\}$$

Le nombre μ_0 de $b \in \mathbb{F}_{2^n} \setminus \{1\}$ tels que P_b n'a pas de racine dans $\mathbb{F}_{2^n} \setminus \{0, 1\}$ est donné par

$$\mu_0 = \frac{2^n + (-1)^{n+1}}{3} + 2^{n-2} + (-1)^n \frac{K(1)}{4}$$

Preuve : Soit $b \in \mathbb{F}_{2^n} \setminus \{1\}$. D'après le théorème 8.8 nous savons que le nombre N_b de racines de P_b dans $\mathbb{F}_{2^n} \setminus \mathbb{F}_2$ est égal au double du nombre de racines dans $\mathbb{F}_{2^n}^*$ du système suivant où $\beta = b + 1$:

$$\begin{cases} Q_\beta(y) &= y^3 + y + \beta = 0 \\ \mathbf{Tr}(y) &= 0 \end{cases} \tag{8.20}$$

Puisque $\beta \neq 0$, on a $Q_\beta(y) \neq 0$ pour $y \in \mathbb{F}_{2^n}$. Donc, pour tout $\beta \neq 0$, on peut avoir les différents cas suivant :

- Q_β n'a pas de racines dans $\mathbb{F}_{2^n}$. Dans ce cas, $N_b = 0$.
- Q_β a une unique racine $y \in \mathbb{F}_{2^n}$. D'après le lemme 8.12, cette situation arrive si et seulement si $\mathbf{Tr}(\beta^{-1}) \neq \mathbf{Tr}(1)$. Dans ce cas, $N_b = 0$ si $\mathbf{Tr}(y) = 1$ et $N_b = 2$ si $\mathbf{Tr}(y) = 0$.
- Q_β a trois racines $y_1, y_2, y_3 \in \mathbb{F}_{2^n}$. Ces racines sont racines d'un polynôme linéaire de degré 4. Ainsi on a $y_3 = y_1 + y_2$, ce qui implique que $\mathbf{Tr}(y_3) = \mathbf{Tr}(y_1) + \mathbf{Tr}(y_2)$. Cette condition implique qu'il y a au moins un des y_i tel que $\mathbf{Tr}(y_i) = 0$. Dans ce cas on a soit $N_b = 6$ soit $N_b = 2$.

Nous définissons la quantité suivante :

$$B = \#\{\beta \in \mathbb{F}_{2^n}^*,\ \ Q_\beta \text{ a une unique racine } y \in \mathbb{F}_{2^n} \text{ et } \mathbf{Tr}(y) = 1\}.$$

À partir de la discussion précédente nous avons,

$$\mu_0 \ = \ \#\{\beta \in \mathbb{F}_{2^n}^*,\ \ Q_\beta \text{ n'a pas de racines dans } \mathbb{F}_{2^n}\} + B$$

D'après la proposition 8.12 nous avons

$$\mu_0 \ = \ \frac{2^n + (-1)^{n+1}}{3} + B. \tag{8.21}$$

En utilisant le fait que $\beta = y^3 + y$, on obtient

$$B \ = \ \#\{(y^3+y) \in \mathbb{F}_{2^n}^*,\ \ \mathbf{Tr}\left(\frac{1}{y^3+y}\right) \neq \mathbf{Tr}(1) \text{ et } \mathbf{Tr}(y) = 1\}\ ,$$

De plus nous avons

$$\frac{1}{y^3+y} = \frac{1+y^2}{y^3+y} + \frac{y^2+y}{y^3+y} + \frac{y}{y^3+y} = \frac{1}{y} + \frac{1}{y+1} + \frac{1}{y^2+1}.$$

Ce qui implique que

$$\mathbf{Tr}\left(\frac{1}{y^3+y}\right) = \mathbf{Tr}\left(\frac{1}{y}\right).$$

En conséquence

$$B = \#\{(y^3+y) \in \mathbb{F}_{2^n}^*,\ \ \mathbf{Tr}\left(\frac{1}{y}\right) \neq \mathbf{Tr}(1) \text{ et } \mathbf{Tr}(y) = 1\}.$$

Maintenant, nous avons $(y^3 + y) = 0$ si et seulement si $y \in \mathbb{F}_2$. De plus, deux éléments distincts y_1 et y_2 dans $\mathbb{F}_{2^n} \setminus \mathbb{F}_2$ avec $\mathbf{Tr}(y_1^{-1}) \neq \mathbf{Tr}(1)$ et $\mathbf{Tr}(y_2^{-1}) \neq \mathbf{Tr}(1)$ vérifient $(y_1^3 + y_1) \neq (y_2^3 + y_2)$ (dans le cas contraire, Q_β avec $\beta = y_1^3 + y_1$ a au moins 2 racines dans $\mathbb{F}_{2^n}$). Nous en déduisons donc,

$$B = \#\{y \in \mathbb{F}_{2^n} \setminus \mathbb{F}_2,\ \ \mathbf{Tr}\left(\frac{1}{y}\right) \neq \mathbf{Tr}(1) \text{ et } \mathbf{Tr}(y) = 1\}.$$

Si n est impair, nous en déduisons que

$$B = \#\{y \in \mathbb{F}_{2^n} \setminus \mathbb{F}_2,\ \ \mathbf{Tr}\left(\frac{1}{y}\right) = 0 \text{ et } \mathbf{Tr}(y) = 1\}.$$

Si n est pair, nous en déduisons que

$$\begin{aligned} B \ &= \ \#\{y \in \mathbb{F}_{2^n} \setminus \mathbb{F}_2,\ \ \mathbf{Tr}\left(\frac{1}{y}\right) = 1 \text{ et } \mathbf{Tr}(y) = 1\} \\ &= \ \#\{y \in \mathbb{F}_{2^n} \setminus \mathbb{F}_2, \mathbf{Tr}(y) = 1\} - \#\{y \in \mathbb{F}_{2^n} \setminus \mathbb{F}_2,\ \ \mathbf{Tr}\left(\frac{1}{y}\right) = 0 \text{ et } \mathbf{Tr}(y) = 1\} \\ &= \ 2^{n-1} - \#\{y \in \mathbb{F}_{2^n} \setminus \mathbb{F}_2,\ \ \mathbf{Tr}\left(\frac{1}{y}\right) = 0 \text{ et } \mathbf{Tr}(y) = 1\}\ . \end{aligned}$$

D'un autre côté, par la définition de la somme de Kloosterman $K(1)$, nous avons

$$\begin{aligned} K(1)-2 &= \sum_{x\in\mathbb{F}_{2^n}\setminus\mathbb{F}_2} (-1)^{\mathbf{Tr}(x^{-1}+x)} \\ &= -2\#\{x\in\mathbb{F}_{2^n}\setminus\mathbb{F}_2, \mathbf{Tr}(x^{-1}+x)=1\}+2^n-2 \\ &= -4\#\{x\in\mathbb{F}_{2^n}\setminus\mathbb{F}_2, \mathbf{Tr}(x^{-1})=0 \text{ et } \mathbf{Tr}(x)=1\}+2^n-2. \end{aligned}$$

Alors,

$$\#\{x\in\mathbb{F}_{2^n}\setminus\mathbb{F}_2, \mathbf{Tr}(x^{-1})=0 \text{ et } \mathbf{Tr}(x)=1\} = 2^{n-2}-\frac{K(1)}{4}.$$

Nous en déduisons alors, que pour tout n,

$$B = 2^{n-2}+(-1)^n\frac{K(1)}{4}.$$

D'après l'équation 8.21, on en déduit que

$$\mu_0 = \frac{2^n+(-1)^{n+1}}{3}+2^{n-2}+(-1)^n\frac{K(1)}{4}.$$

□

Preuve : **Preuve du théorème 8.10**

D'après le lemme 8.1, afin de trouver le spectre différentiel complet de G_3 nous devons être capable de résoudre le système suivant :

$$\begin{cases} \omega_0+\omega_2+\omega_4+\omega_6 = 2^n \\ 2\omega_2+4\omega_4+6\omega_6 = 2^n \end{cases} \tag{8.22}$$

D'après le théorème 8.6, nous savons que pour tout $b\neq\{0,1\}$, $\delta(b)\in\{0,2,6\}$. De plus dans le théorème 8.11 nous avons calculé la valeur μ_0 (voir définition de μ_0 dans le théorème 8.11) qui correspond exactement à la quantité ω_0. On a donc

$$\omega_0 = \mu_0 = \frac{2^n+(-1)^{n+1}}{3}+2^{n-2}+(-1)^n\frac{K(1)}{4}.$$

Afin de pouvoir déterminer le spectre différentiel complet de la fonction G_3, nous avons besoin de calculer la valeur de $\delta(1)$. D'après le théorème 8.6, on a $\delta(0)=2^{\text{pgcd}(t,n)}$ et $\delta(1)=2^{\text{pgcd}(t-1,n)}$. Comme $t=3$, si n est impair on a $\text{pgcd}(t-1,n)=1$, et si n est pair on a $\text{pgcd}(t-1,n)=1$. Ainsi $\delta(1)=4$ si n est pair et $\delta(1)=2$ si n est impair. Donc, $\omega_4=1$ si n est pair et $\omega_4=0$ sinon. À partir de la seconde équation du système (8.22), nous obtenons

$$\omega_2 = 2^{n-1}-3\omega_6-2\omega_4,$$

et en utilisant la première équation du même système, nous obtenons

$$\begin{aligned} \omega_6 &= 2^n-\omega_0-\omega_2-\omega_4 = 2^{n-1}-\omega_0+\omega_4+3\omega_6\ , \\ \omega_6 &= -2^{n-2}+\frac{\omega_0-\omega_4}{2}\ . \end{aligned}$$

t	$\max_{b \neq \{0,1\}} \delta(b)$	Commentaires
2	2	quadratique
3	6	
$(n+k)/3$	6	$k = 0, 1, 2, 3$ et $n + k \mod 3 = 0$
$n/2$	2	n pair
$(n-1)/2$	6	n impair
$(n+3)/2$	6	n impair
$n/2+1$	2	n pair
$(2n+k)/3$	6	$k = 0, 1, 2, 3$ et $n - k \mod 3 = 0$
$n-2$	6	
$n-1$	2	Inverse

TABLE 8.8 – Uniformité différentielle restreinte des fonctions puissances avec exposant $2^t - 1$.

Nous en déduisons que pour n impair,

$$\begin{aligned} \omega_6 &= -2^{n-2} + \frac{\omega_0}{2} = -2^{n-3} + \frac{2^n+1}{6} - \frac{K(1)}{8} \\ &= \frac{2^{n-2}+1}{6} - \frac{K(1)}{8} \end{aligned}$$

et pour n pair,

$$\begin{aligned} \omega_6 &= -2^{n-2} + \frac{\omega_0 - 1}{2} = -2^{n-3} + \frac{2^n-1}{6} + \frac{K(1)}{8} - \frac{1}{2} \\ &= \frac{2^{n-2}-4}{6} + \frac{K(1)}{8}. \end{aligned}$$

Pour conclure la preuve, d'après l'expression de la somme de Kloosterman, on montre que $\omega_6 \geq 1$ pour tout $n \geq 6$. Ce résultat implique que G_3 est différentiellement 6-uniforme. □

Remarque 8.10. *Pour $n = 5$, la fonction G_3 est l'inverse de la permutation puissance APN avec exposant quadratique $x \mapsto x^9$. Ainsi pour $n = 5$, G_3 est une fonction APN.*

Pour $n = 4$, la fonction G_3 correspond à la fonction inverse sur le corps $\mathbb{F}_{2^4}$. Dans ce cas la fonction G_3 est localement-APN différentiellement 4-uniforme avec $\omega_4 = 1$.

8.6.7 Quelques classes spécifiques

Dans la section A.2 nous avons mis les tables des fonctions $G_t(x) = 2^t - 1$ pour $5 \leq n \leq 32$. Nous avons séparé les résultats suivant que $\max_{b \neq \{0,1\}} \delta(b) \leq \lambda$ pour $\lambda = 2, 6, 14, 30$. Dans cette section, nous allons prouver les propriétés présentées de manière synthétique dans le tableau 8.8.

Exposant $2^{\lfloor n/2 \rfloor} - 1$

Nous étudions la sous famille G_t pour $t = \lfloor n/2 \rfloor$. Dans un premier temps, nous considérons le cas où n est pair. Nous avons que G_t n'est pas une permutation puisque $2^n - 1 = (2^t - 1)(2^t + 1)$.

Théorème 8.12. *Soit n un entier pair, $n > 4$ et $G_t(x) = x^{2^t-1}$ pour $t = \frac{n}{2}$. Alors G_t est quasi-APN (voir définition 8.2). Plus précisément*

$$\delta(G_t) = 2^t - 2 \quad et \quad \delta(b) \leq 2, \ \forall\ b \in \mathbb{F}_{2^n} \setminus \mathbb{F}_2.$$

– *Si $n \equiv 0 \bmod 4$, le spectre différentiel de G_t est :*

$$\begin{aligned} \omega_{2^t-2} &= 1, \\ \omega_i &= 0, \ \forall\ i, \ 2 < i < 2^t - 2, \\ \omega_2 &= 2^{n-1} - 2^{t-1} + 1, \\ \omega_0 &= 2^{n-1} + 2^{t-1} - 2. \end{aligned}$$

– *Si $n \equiv 2 \bmod 4$, le spectre différentiel de G_t est :*

$$\begin{aligned} \omega_{2^t-2} &= 1, \\ \omega_i &= 0, \ \forall\ i, \ 4 < i < 2^t - 2, \\ \omega_4 &= 1, \\ \omega_2 &= 2^{n-1} - 2^{t-1} - 1, \\ \omega_0 &= 2^{n-1} + 2^{t-1} - 1. \end{aligned}$$

Preuve : À partir du théorème 8.6, nous obtenons directement $\delta(0) = 2^t - 2$, et $\delta(1) = 2$ si t est pair ou $\delta(1) = 4$ si t est impair.

Maintenant, pour tout $b \notin \mathbb{F}_2$, nous devons déterminer le nombre de racines dans $\mathbb{F}_{2^n}$ de

$$P_b(x) = x^{2^t} + bx^2 + (b+1)x$$

ou, de manière équivalente, le nombre de racine de

$$(P_b(x))^{2^t} = x + b^{2^t} x^{2^{t+1}} + (b+1)^{2^t} x^{2^t}.$$

Si x est une racine de P_b alors $x^{2^t} = bx^2 + (b+1)x$. Donc, l'équation $P_b(x) = 0$ implique

$$\begin{aligned} (P_b(x))^{2^t} &= x + b^{2^t}(x^{2^t})^2 + (b^{2^t} + 1)x^{2^t} \\ &= x + b^{2^t}(bx^2 + (b+1)x)^2 + (b^{2^t} + 1)(bx^2 + (b+1)x) \\ &= b^{2^t+2}x^4 + (b^{2^t+2} + b^{2^t+1} + b^{2^t} + b)x^2 + (b^{2^t+1} + b^{2^t} + b)x. \end{aligned}$$

On obtient alors un polynôme linéaire de degré 4. Deux racines évidentes de ce polynôme sont 0 et 1. Donc ce polynôme a τ racines où τ est égal à 4 ou 2. Et, pour tout $b \notin \mathbb{F}_2$, $\delta(b) \leq 2$ puisque $\delta(b) \leq \tau - 2$. Nous en déduisons que G_t est quasi-APN. Le spectre différentiel complet peut alors être obtenu en utilisant le lemme 8.1. Dans le cas où t est pair, nous avons $\omega_i = 0$ pour tout $i \notin \{0, 2, 2^t - 2\}$, sinon nous avons $\omega_i = 0$ pour tout

$i \notin \{0, 2, 4, 2^t - 2\}$, de plus si t est impair $\omega_{2^t-2} = \omega_4 = 1$. En utilisant le lemme 8.1, pour t pair nous obtenons :

$$\begin{cases} 2^n &= \omega_0 + \omega_2 + 1 \\ 2^n &= 2\omega_2 + (2^t - 2). \end{cases}$$

Donc $\omega_2 = 2^{n-1} - 2^{t-1} + 1$ et nous concluons avec $\omega_0 = 2^n - \omega_2 - 1$. La même preuve peut être faite pour t impair en résolvant le système suivant :

$$\begin{cases} 2^n = \omega_0 + \omega_2 + 2 \\ 2^n = 2\omega_2 + 2^t + 2. \end{cases}$$

□

À partir de ce résultat, nous déduisons un résultat général sur le nombre de racines des polynômes linéaires correspondants.

Corollaire 8.6. *Soit $n = 2t$. Soit $b \in \mathbb{F}_{2^n}$ fixé. Soit les polynômes définis sur le corps $\mathbb{F}_{2^n}$ par*

$$x^{2^t} + bx^2 + (b+1)x \quad et \quad x^{2^{t+1}} + bx^2 + (b+1)x.$$

Alors, pour tout $b \in \mathbb{F}_{2^n} \setminus \mathbb{F}_2$, ces polynômes ont 2 ou 4 racines dans le corps $\mathbb{F}_{2^n}$.

Théorème 8.13. *Soit n un entier pair $n > 4$ et $G_{t+1}(x) = x^{2^{t+1}-1}$ pour $t = \frac{n}{2}$. Alors, G_{t+1} est localement-APN, différentiellement 2^t-uniformes et son spectre différentiel est*

$$\begin{aligned} \omega_{2^t} &= 1 \\ \omega_i &= 0, \ \forall\ i,\ 2 < i < 2^t \\ \omega_2 &= 2^{n-1} - 2^{t-1} \\ \omega_0 &= 2^{n-1} + 2^{t-1} - 1\ . \end{aligned}$$

De plus, G_{t+1} est une permutation si et seulement si $n \equiv 0 \bmod 4$.

Preuve : Dans un premier temps, puisque $n = 2t$, nous avons $\text{pgcd}(t+1, n) = 1$ si t est pair (c'est-à-dire, $n \equiv 0 \bmod 4$) et $\text{pgcd}(t+1, n) = 2$ si t est impair (c'est-à-dire, $n \equiv 2 \bmod 4$).

Soit $(\omega'_i)_{0 \le i \le 2^n}$ (resp. $(\omega_i)_{0 \le i \le 2^n}$) le spectre différentiel de G_t (resp. G_{t+1}) sur le corps $\mathbb{F}_{2^n}$. Nous obtenons alors le lien suivant entre les deux spectres :

– Pour $n \equiv 0 \bmod 4$, nous avons

$$(\delta_t(0), \delta_t(1)) = (2^t - 2, 2) \text{ et } (\delta_s(0), \delta_s(1)) = (0, 2^t).$$

Nous déduisons,

$$\omega_0 = \omega'_0 + 1,\ \ \omega_2 = \omega'_2 - 1,\ \ \omega_{2^t-2} = \omega'_{2^t-2} - 1 \text{ et } \omega_{2^t} = 1.$$

– Pour $n \equiv 2 \bmod 4$, nous avons

$$(\delta_t(0), \delta_t(1)) = (2^t - 2, 4) \text{ et } (\delta_s(0), \delta_s(1)) = (2, 2^t).$$

Nous déduisons ,

$$\omega_2 = \omega'_2 + 1,\ \ \omega_4 = \omega'_4 - 1,\ \ \omega_{2^t-2} = \omega'_{2^t-2} - 1 \text{ et } \omega_{2^t} = 1.$$

Le spectre différentiel G_{t+1} se déduit directement en combinant les deux formules précédentes avec les valeurs de ω'_i calculées dans le théorème 8.12. □

Dans le cas où n est impair, le théorème suivant nous donne des informations sur le spectre différentiel de G_t, avec $t = \frac{n-1}{2}$.

Théorème 8.14. *Soit n un entier impair, $n > 3$. Soit $G_t(x) = x^{2^t-1}$ avec $t = (n-1)/2$. de spectre différentiel $\{\omega_i\}_i$. Alors, G_t est une permutation. Pour tout $b \in \mathbb{F}_{2^n} \setminus \mathbb{F}_2$ nous avons $\delta(b) \in \{0, 2, 6\}$. De plus*

- *si $n \equiv 0 \bmod 3$, alors $\delta(G_t) = 8$, $\omega_8 = 1$ et $\omega_i = 0$ pour tout $i \notin \{0, 2, 6, 8\}$.*
- *si $n \not\equiv 0 \bmod 3$, alors $\delta(G_t) \leq 6$ et $\omega_i = 0$ pour tout $i \notin \{0, 2, 6\}$*

Preuve : À partir du théorème 8.6, nous avons $\delta(0) = 0$. De plus, si 3 divise n alors $\delta(1) = 8$ sinon $\delta(1) = 2$. En effet, puisque

$$\mathrm{pgcd}(t-1, n) = \mathrm{pgcd}(\frac{n-3}{2}, n) = \mathrm{pgcd}(n-3, n) = \mathrm{pgcd}(3, n),$$

donc

$$\delta(1) = \begin{cases} 8 & \text{si} \quad n \equiv 0 \mod 3; \\ 2 & \text{sinon.} \end{cases}$$

À partir de maintenant, nous supposons que $b \notin \mathbb{F}_2$, Nous devons alors déterminer le nombre de solutions dans $\mathbb{F}_{2^n}$ de

$$P_b(x) = x^{2^t} + bx^2 + (b+1)x,$$

ou, de façon équivalente, le nombre de racines de

$$(P_b(x))^{2^{t+1}} = x + b^{2^{t+1}} x^{2^{t+2}} + (b+1)^{2^{t+1}} x^{2^{t+1}}.$$

Soient $c = b^{2^{t+1}}$ et $Q_b(x) = (P_b(x))^{2^{t+1}}$. Si x est une racine P_b alors $x^{2^t} = bx^2 + (b+1)x$. Donc, l'équation $P_b(x) = 0$ implique

$$\begin{aligned} Q_b(x) &= x + c(x^{2^t})^4 + (c+1)(x^{2^t})^2 \\ &= x + c(bx^2 + (b+1)x)^4 + (c+1)(bx^2 + (b+1)x)^2 \\ &= cb^4x^8 + (c(b+1)^4 + (c+1)b^2)x^4 + (c+1)(b^2+1)x^2 + x\ . \end{aligned}$$

Puisque Q_b est de degré 8, il a soit 8 soit 4 soit 2 solutions. En d'autres termes, pour tout $b \neq \{0, 1\}$, nous avons $\delta(b) \in \{0, 2, 6\}$. □

Lemme 8.13. *Soit n un entier impair, $n > 3$. Soit $G_t(x) = x^{2^t-1}$ avec $t = (n+3)/2$. Alors, si n n'est pas divisible par 3 G_t est une permutation. De plus $\delta(G_t) \leq 6$*

Preuve : On note $s = \frac{n+3}{2}$ et $t = \frac{n-1}{2}$. Nous appliquons le corollaire 8.2. Par le théorème 8.14, nous avons

$$\delta_s(0) = \delta_t(1) - 2 = \begin{cases} 6 & \text{si} \quad n \equiv 0 \mod 3 \\ 0 & \text{sinon.} \end{cases}$$

et $\delta_s(1) = \delta_t(0) + 2 = 2$. De plus, comme $\forall b \neq \{0, 1\}$, on a $\delta(b) \leq 6$ et le résultat suit. □

Le théorème 8.14 et les résultats expérimentaux donnés dans la section A.2 nous amènent à la conjecture suivante.

Conjecture 8.9. *La fonction $G_t(x) = x^{2^t-1}$ avec $t = (n-1)/2$ a le même spectre différentiel que la fonction $G_3 = x^7$.*

Exposant $2^{\lfloor n/3 \rfloor} - 1$

Dans le tableau A.3, nous pouvons remarquer que pour chaque valeur de $n > 8$, il existe au moins 4 exposants (6 dans le cas où 3 divise n) pour lesquels $\max_{b \neq 0,1} \delta(b) = 6$. Dans la section 8.6.5, nous avons étudié le spectre différentiel des fonctions G_3 et G_{n-2}. Dans cette section, nous donnons le début d'une étude du spectre différentiel des fonctions G_t avec $t = \frac{n+k}{3}$ (où $k = 0, 1, 2, 3$ tel que $n + k \equiv 0 \mod 3$). On peut alors appliquer la même technique que celle de la section précédente pour montrer que pour $b \neq \{0, 1\}$ on a $\delta(b) \leq 6$.

Théorème 8.15. *Soit $\mathbb{F}_{2^n}$ le corps à 2^n éléments. Soit $k = 0, 1, 2, 3$ tel que $n + k \equiv 0 \mod 3$. Soit $G_{\frac{n+k}{3}}$ la fonction puissance définie sur le corps $\mathbb{F}_{2^n}$ par :*

$$G_{\frac{n+k}{3}} : x \mapsto x^{2^{\frac{n+k}{3}}-1}$$

Alors $\forall b \neq 0, 1$ nous avons $\delta(b) = 0, 2, 6$

Preuve : Nous devons déterminer le nombre de racines dans $\mathbb{F}_{2^n}$ du polynôme $P_b(X) = X^{2^{(n+k)/3}} + bX^2 + (b+1)X$. Pour simplifier la preuve nous notons $d = b^{2^{(n+k)/3}}$ et $c = b^{2^{(2n+2k)/3}}$

Soit x une racine de $P_b(X)$. On a

$$P_b(x)^{2^{(n+k)/3}} = x^{2^{(2n+2k)/3}} + dx^{2^{(n+k)/3+1}} + (d+1)x^{2^{(n+k)/3}}$$

et

$$Q_b(x) = P_b(x)^{2^{(2n+2k)/3}} = x^{2^{(3n+3k)/3}} + cx^{2^{(2n+2k)/3+1}} + (c+1)x^{2^{(2n+2k)/3}}$$

En remplaçant dans la deuxième équation nous obtenons :

$$\begin{aligned}
Q_b(x) &= x^{2^k} + c[dx^{2^{(n+k)/3+1}} + (d+1)x^{2^{(n+k)/3}}]^2 + (c+1)[dx^{2^{(n+k)/3+1}} + (d+1)x^{2^{(n+k)/3}}] \\
&= x^{2^k} + cd^2x^{2^{(n+k)/3+2}} + [c(d^2+1) + (c+1)d]x^{2^{(n+k)/3+1}} + (c+1)(d+1)x^{2^{(n+k)/3}} \\
&= x^{2^k} + cd^2[bx^2 + (b+1)x]^4 + (cd^2 + c + cd + d)[bx^2 + (b+1)x]^2 \\
&\quad +(c+1)(d+1)[bx^2 + (b+1)x] \\
&= [cd^2b^4]x^8 + [cd^2(b^4+1) + (cd^2 + c + cd + d)b^2]x^4 + [(cd^2 + c + cd + d)(b^2+1) \\
&\quad +(c+1)(d+1)b]x^2 + [(c+1)(d+1)(b+1)]x + x^{2^k} \\
&= [cd^2b^4]x^8 + [cd^2b^4 + cd^2 + cd^2b^2 + cb^2 + cdb^2 + db^2]x^4 + \\
&\quad [cd^2b^2 + cd^2 + cb^2 + c + cdb^2 + cd + db^2 + d + cdb + cb + db + b]x^2 + \\
&\quad [cdb + cd + cb + c + db + d + b + 1]x + x^{2^k}
\end{aligned}$$

Le résultat suit alors immédiatement en remarquant que Q_b est de degré 8, et qu'il a alors 8, 4 ou 2 solutions. □

		$t = 2^{(n+k)/3}$		$t = 2^{(2n+3-k)/3}$	
		$\delta(0)$	$\delta(1)$	$\delta(0)$	$\delta(1)$
$n = 3l$	$k = 0$	$2^{n/3} - 2$	$2^{\text{pgcd}(n/3,3)}$	$2^{\text{pgcd}(n/3,3)} - 2$	$2^{n/3}$
$n = 6l + 1$	$k = 2$	0	2	0	2
$n = 6l + 2$	$k = 1$	0	4	2	2
$n = 6l + 4$	$k = 2$	2	2	0	4
$n = 6l + 5$	$k = 1$	0	2	0	2
$n = 3l$	$k = 3$	$2^{\text{pgcd}(n/3+1,3)} - 2$	$2^{n/3}$	$2^{n/3} - 2$	$2^{\text{pgcd}(n/3+1,3)}$

TABLE 8.9 – $\delta(0)$ et $\delta(1)$ pour les fonctions $2^t - 1$ avec $t = (n + k)/3$

Dans le tableau 8.3 nous avons mis seulement les permutations puissances qui sont différentiellement 6-uniformes. Pourtant si on exclut $\delta(0)$ et $\delta(1)$, d'après le théorème précédent on a que les fonctions $G_{\frac{n+k}{3}}$ vérifient $\max_{b\neq 0,1} \delta(b) \leq 6$.

Pour cette raison en appendice nous avons calculé le spectre différentiel des fonctions G_t pour des tailles de corps comprises entre 6 et 32(voir section A.2)

Si on étudie le spectre différentiel des fonctions $G_{\frac{n+k}{3}}$ nous remarquons que si $k = 1, 2$ alors le spectre différentiel de $G_{\frac{n+k}{3}}$ est le même que celui de G_3 (en tout cas pour $n < 32$). On peut donc faire la conjecture suivante.

Conjecture 8.10. *Soient n et k tel que $n \equiv k \mod 3$ et $k = 1$ ou 2 alors le spectre différentiel privé de $\delta(0)$ et $\delta(1)$ de $G_{\frac{n+k}{3}}$ est le même que celui de $G_3(x) = x^7$.*

Dans le cas où n est divisible par 3, nous avons vu dans le théorème précédent que les fonctions $G_{\frac{n}{3}}$ et $G_{\frac{n+3}{3}}$ vérifient $\max_{b\neq 0,1} \delta(b) \leq 6$. Par symétrie (au sens du théorème 8.9) les fonctions $G_{\frac{2n-3}{3}}$ et $G_{\frac{2n}{3}}$ vérifient aussi $\max_{b\neq 0,1} \delta(b) \leq 6$. Expérimentalement on peut voir que le spectre différentiel de ces fonctions n'est pas le même que le spectre différentiel de G_3.

Afin de déterminer le spectre différentiel des fonctions $G_{\frac{n+k}{3}}$ nous pouvons commencer par nous intéresser aux valeurs de $\delta(0)$ et $\delta(1)$. Le calcul de ces valeurs a été effectué à l'aide des formules données dans le théorème 8.6. Le tableau 8.9 résume ces valeurs suivant la taille du corps n.

Appendices

A.1 Attaque expérimentale sur SMALLPRESENT-[8]

Dans le tableau A.1 nous donnons toutes les différentielles que nous avons utilisées pour faire notre attaque expérimentale sur SMALLPRESENT-[8]. Pour chaque différentielle nous avons calculé la probabilité théorique en sommant les probabilités des chemins qui composent la différentielle (chemins avec probabilité supérieure à 2^{-48}). Cette probabilité a été obtenue grâce à l'algorithme de "branch and bound" que nous avons décrit dans la section 4.1.2. Les autres colonnes correspondent à la probabilité de la différentielle obtenue par une moyenne sur les 2^{32} messages clairs et 250 clés pour les deux algorithmes de cadencement de clés pour des clés maîtres de 40 bits et de 80 bits (voir section 1.4.1).

TABLE A.1 – Différentielles utilisées pour notre attaque par différentielle multiple sur SMALLPRESENT-[8]

Differential	Theo.	40-bit	80-bit	Differential	Theo.	40-bit	80-bit
0x3 → 0x40400000	$2^{-30.28}$	$2^{-29.80}$	$2^{-29.85}$	0x5 → 0x40400000	$2^{-30.20}$	$2^{-29.76}$	$2^{-29.80}$
0x3 → 0x04040000	$2^{-30.33}$	$2^{-29.80}$	$2^{-29.84}$	0x5 → 0x04040000	$2^{-30.25}$	$2^{-29.87}$	$2^{-29.73}$
0x3 → 0x50500000	$2^{-30.46}$	$2^{-29.96}$	$2^{-30.07}$	0x5 → 0x50500000	$2^{-30.34}$	$2^{-29.87}$	$2^{-29.76}$
0x3 → 0x05050000	$2^{-30.58}$	$2^{-29.98}$	$2^{-29.99}$	0x5 → 0x10100000	$2^{-30.50}$	$2^{-30.06}$	$2^{-30.28}$
0x3 → 0x10100000	$2^{-30.59}$	$2^{-29.90}$	$2^{-30.10}$	0x5 → 0x05050000	$2^{-30.52}$	$2^{-30.02}$	$2^{-30.06}$
0x3 → 0x01010000	$2^{-30.64}$	$2^{-29.94}$	$2^{-30.45}$	0x5 → 0x01010000	$2^{-30.55}$	$2^{-29.96}$	$2^{-29.94}$
0x3 → 0x80800000	$2^{-30.70}$	$2^{-30.17}$	$2^{-30.24}$	0x5 → 0x08080000	$2^{-30.57}$	$2^{-30.01}$	$2^{-29.97}$
0x3 → 0x08080000	$2^{-30.70}$	$2^{-30.10}$	$2^{-30.01}$	0x5 → 0x80800000	$2^{-30.57}$	$2^{-29.98}$	$2^{-30.04}$
0x3 → 0x0a0a0000	$2^{-30.97}$	$2^{-30.27}$	$2^{-30.32}$	0x5 → 0x0a0a0000	$2^{-30.77}$	$2^{-30.08}$	$2^{-30.04}$
0x7 → 0x40400000	$2^{-29.47}$	$2^{-29.20}$	$2^{-29.21}$	0xB → 0x40400000	$2^{-30.21}$	$2^{-29.60}$	$2^{-29.88}$
0x7 → 0x04040000	$2^{-29.54}$	$2^{-29.23}$	$2^{-23.23}$	0xB → 0x04040000	$2^{-30.26}$	$2^{-29.75}$	$2^{-29.92}$
0x7 → 0x50500000	$2^{-29.59}$	$2^{-29.26}$	$2^{-29.30}$	0xB → 0x50500000	$2^{-30.41}$	$2^{-29.96}$	$2^{-29.99}$
0x7 → 0x10100000	$2^{-29.74}$	$2^{-29.33}$	$2^{-29.70}$	0xB → 0x05050000	$2^{-30.59}$	$2^{-29.97}$	$2^{-30.06}$
0x7 → 0x05050000	$2^{-29.76}$	$2^{-29.37}$	$2^{-29.43}$	0xB → 0x08080000	$2^{-30.64}$	$2^{-29.94}$	$2^{-30.02}$
0x7 → 0x01010000	$2^{-29.86}$	$2^{-29.54}$	$2^{-29.56}$	0xB → 0x80800000	$2^{-30.65}$	$2^{-29.95}$	$2^{-30.06}$
0x7 → 0x0a0a0000	$2^{-30.00}$	$2^{-29.63}$	$2^{-29.65}$	0xB → 0x10100000	$2^{-30.73}$	$2^{-30.13}$	$2^{-30.33}$
0x7 → 0x80800000	$2^{-30.19}$	$2^{-29.61}$	$2^{-29.72}$	0xB → 0x01010000	$2^{-30.81}$	$2^{-30.13}$	$2^{-30.18}$
0x7 → 0x08080000	$2^{-30.21}$	$2^{-29.66}$	$2^{-29.66}$	0xB → 0x0a0a0000	$2^{-30.86}$	$2^{-30.09}$	$2^{-30.10}$
0x7 → 0x40500000	$2^{-30.76}$	$2^{-30.22}$	$2^{-30.09}$	0xF → 0x00110000	$2^{-30.60}$	$2^{-29.97}$	$2^{-29.78}$
0xD → 0x05050000	$2^{-29.81}$	$2^{-29.30}$	$2^{-29.39}$	0xF → 0x40400000	$2^{-29.49}$	$2^{-29.26}$	$2^{-29.36}$
0xD → 0x40400000	$2^{-29.82}$	$2^{-29.42}$	$2^{-29.42}$	0xF → 0x04040000	$2^{-29.56}$	$2^{-29.23}$	$2^{-29.31}$
0xD → 0x04040000	$2^{-29.91}$	$2^{-29.50}$	$2^{-29.46}$	0xF → 0x50500000	$2^{-29.80}$	$2^{-29.46}$	$2^{-29.45}$
0xD → 0x10100000	$2^{-30.01}$	$2^{-29.50}$	$2^{-29.83}$	0xF → 0x05050000	$2^{-29.82}$	$2^{-29.39}$	$2^{-29.37}$
0xD → 0x50500000	$2^{-30.08}$	$2^{-29.60}$	$2^{-29.71}$	0xF → 0x80800000	$2^{-29.88}$	$2^{-29.32}$	$2^{-29.37}$
0xD → 0x01010000	$2^{-30.15}$	$2^{-29.52}$	$2^{-30.14}$	0xF → 0x08080000	$2^{-29.88}$	$2^{-29.58}$	$2^{-29.38}$
0xD → 0x0a0a0000	$2^{-30.25}$	$2^{-29.74}$	$2^{-29.78}$	0xF → 0x10100000	$2^{-30.10}$	$2^{-29.69}$	$2^{-29.76}$
0xD → 0x80800000	$2^{-30.39}$	$2^{-29.82}$	$2^{-29.96}$	0xF → 0x01010000	$2^{-30.16}$	$2^{-29.68}$	$2^{-29.94}$
				0xF → 0x0a0a0000	$2^{-30.22}$	$2^{-29.67}$	$2^{-29.80}$

A.2 Spectre différentiel des fonctions $x \mapsto x^{2^t-1}$

Dans cette section, nous donnons le spectre différentiel des fonctions $x \mapsto x^{2^t-1}$. Nous avons séparé les résultats en 4 tableaux.

- Dans le tableau A.2 nous donnons les fonctions $G_t(x) = x^{2^t-1}$ pour lesquelles $\max_{b\neq 0,1} \delta(b) = 2$.
- Dans le tableau A.3 nous donnons les fonctions $G_t(x) = x^{2^t-1}$ pour lesquelles $\max_{b\neq 0,1} \delta(b) = 6$.
- Dans le tableau A.4 nous donnons les fonctions $G_t(x) = x^{2^t-1}$ pour lesquelles $\max_{b\neq 0,1} \delta(b) = 14$.
- Dans le tableau A.5 nous donnons les fonctions $G_t(x) = x^{2^t-1}$ pour lesquelles $\max_{b\neq 0,1} \delta(b) = 30$.

Les spectres différentiels donnés dans ces tableaux sont calculés sans la valeur de $\delta(0)$ et de $\delta(1)$.

TABLE A.2 – Spectre différentiel des fonctions $G_t(x) = x^{2^t-1}$ sur le corps $\mathbb{F}_{2^n}$ avec $\forall b \neq \{0,1\}$, $\delta(b) \leq 2$

Spectre différentiel des fonctions $G_t(x) = x^{2^t-1}$ sur le corps $\mathbb{F}_{2^n}$ avec $\forall b \neq \{0,1\}$, $\delta(b) \leq 2$						
n	t	inverse	pgcd	$\delta_{\max}$	δ_0, δ_1	Spectre différentiel sans $\delta(0), \delta(1)$
5	2	11	1	2	0 , 2	0 [15] 2 [15]
5	3	5	1	2	0 , 2	0 [15] 2 [15]
5	4	15	1	2	0 , 2	0 [15] 2 [15]
6	2	-	3	2	2 , 2	0 [32] 2 [30]
6	3	-	7	6	6 , 4	0 [35] 2 [27]
6	4	-	3	8	2 , 8	0 [35] 2 [27]
6	5	31	1	4	0 , 4	0 [32] 2 [30]
7	2	43	1	2	0 , 2	0 [63] 2 [63]
7	4	9	1	2	0 , 2	0 [63] 2 [63]
7	6	63	1	2	0 , 2	0 [63] 2 [63]
8	2	-	3	2	2 , 2	0 [128] 2 [126]
8	4	-	15	14	14 , 2	0 [134] 2 [120]
8	5	91	1	16	0 , 16	0 [134] 2 [120]
8	7	127	1	4	0 , 4	0 [128] 2 [126]
9	2	171	1	2	0 , 2	0 [255] 2 [255]
9	5	17	1	2	0 , 2	0 [255] 2 [255]
9	8	255	1	2	0 , 2	0 [255] 2 [255]
10	2	-	3	2	2 , 2	0 [512] 2 [510]
10	5	-	31	30	30 , 4	0 [527] 2 [495]
10	6	-	3	32	2 , 32	0 [527] 2 [495]
10	9	511	1	4	0 , 4	0 [512] 2 [510]
11	2	683	1	2	0 , 2	0 [1023] 2 [1023]
11	6	33	1	2	0 , 2	0 [1023] 2 [1023]
11	10	1023	1	2	0 , 2	0 [1023] 2 [1023]
12	2	-	3	2	2 , 2	0 [2048] 2 [2046]
12	6	-	63	62	62 , 2	0 [2078] 2 [2016]
12	7	1387	1	64	0 , 64	0 [2078] 2 [2016]
12	11	2047	1	4	0 , 4	0 [2048] 2 [2046]
13	2	2731	1	2	0 , 2	0 [4095] 2 [4095]
13	7	65	1	2	0 , 2	0 [4095] 2 [4095]
13	12	4095	1	2	0 , 2	0 [4095] 2 [4095]
14	2	-	3	2	2 , 2	0 [8192] 2 [8190]
14	7	-	127	126	126 , 4	0 [8255] 2 [8127]
14	8	-	3	128	2 , 128	0 [8255] 2 [8127]
14	13	8191	1	4	0 , 4	0 [8192] 2 [8190]
15	2	10923	1	2	0 , 2	0 [16383] 2 [16383]
15	8	129	1	2	0 , 2	0 [16383] 2 [16383]
15	14	16383	1	2	0 , 2	0 [16383] 2 [16383]

Spectre différentiel des fonctions $G_t(x) = x^{2^t-1}$ sur le corps $\mathbb{F}_{2^n}$ avec $\forall b \neq \{0,1\}$, $\delta(b) \leq 2$

n	t	inverse	pgcd	$\delta_{\max}$	δ_0, δ_1	Spectre différentiel sans $\delta(0), \delta(1)$
16	2	-	3	2	2 , 2	0 [32768] 2 [32766]
16	8	-	255	254	254 , 2	0 [32894] 2 [32640]
16	9	21931	1	256	0 , 256	0 [32894] 2 [32640]
16	15	32767	1	4	0 , 4	0 [32768] 2 [32766]
17	2	43691	1	2	0 , 2	0 [65535] 2 [65535]
17	9	257	1	2	0 , 2	0 [65535] 2 [65535]
17	16	65535	1	2	0 , 2	0 [65535] 2 [65535]
18	2	-	3	2	2 , 2	0 [131072] 2 [131070]
18	9	-	511	510	510 , 4	0 [131327] 2 [130815]
18	10	-	3	512	2 , 512	0 [131327] 2 [130815]
18	17	131071	1	4	0 , 4	0 [131072] 2 [131070]
19	2	174763	1	2	0 , 2	0 [262143] 2 [262143]
19	10	513	1	2	0 , 2	0 [262143] 2 [262143]
19	18	262143	1	2	0 , 2	0 [262143] 2 [262143]
20	2	-	3	2	2 , 2	0 [524288] 2 [524286]
20	10	-	1023	1022	1022 , 2	0 [524798] 2 [523776]
20	11	349867	1	1024	0 , 1024	0 [524798] 2 [523776]
20	19	524287	1	4	0 , 4	0 [524288] 2 [524286]
21	2	699051	1	2	0 , 2	0 [1048575] 2 [1048575]
21	11	1025	1	2	0 , 2	0 [1048575] 2 [1048575]
21	20	1048575	1	2	0 , 2	0 [1048575] 2 [1048575]
22	2	-	3	2	2 , 2	0 [2097152] 2 [2097150]
22	11	-	2047	2046	2046 , 4	0 [2098175] 2 [2096127]
22	12	-	3	2048	2 , 2048	0 [2098175] 2 [2096127]
22	21	2097151	1	4	0 , 4	0 [2097152] 2 [2097150]
23	2	2796203	1	2	0 , 2	0 [4194303] 2 [4194303]
23	12	2049	1	2	0 , 2	0 [4194303] 2 [4194303]
23	22	4194303	1	2	0 , 2	0 [4194303] 2 [4194303]
24	2	-	3	2	2 , 2	0 [8388608] 2 [8388606]
24	12	-	4095	4094	4094 , 2	0 [8390654] 2 [8386560]
24	13	5593771	1	4096	0 , 4096	0 [8390654] 2 [8386560]
24	23	8388607	1	4	0 , 4	0 [8388608] 2 [8388606]
25	2	11184811	1	2	0 , 2	0 [16777215] 2 [16777215]
25	13	4097	1	2	0 , 2	0 [16777215] 2 [16777215]
25	24	16777215	1	2	0 , 2	0 [16777215] 2 [16777215]
26	2	-	3	2	2 , 2	0 [33554432] 2 [33554430]
26	13	-	8191	8190	8190 , 4	0 [33558527] 2 [33550335]
26	14	-	3	8192	2 , 8192	0 [33558527] 2 [33550335]
26	25	33554431	1	4	0 , 4	0 [33554432] 2 [33554430]
27	2	44739243	1	2	0 , 2	0 [67108863] 2 [67108863]
27	14	8193	1	2	0 , 2	0 [67108863] 2 [67108863]
27	26	67108863	1	2	0 , 2	0 [67108863] 2 [67108863]
28	2	-	3	2	2 , 2	0 [134217728] 2 [134217726]
28	14	-	16383	16382	16382 , 2	0 [134225918] 2 [134209536]
28	15	89483947	1	16384	0 , 16384	0 [134225918] 2 [134209536]

Spectre différentiel des fonctions $G_t(x) = x^{2^t-1}$ sur le corps $\mathbb{F}_{2^n}$ avec $\forall b \neq \{0,1\}$, $\delta(b) \leq 2$						
n	t	inverse	pgcd	$\delta_{\max}$	δ_0, δ_1	Spectre différentiel sans $\delta(0), \delta(1)$
28	27	134217727	1	4	0 , 4	0 [134217728] 2 [134217726]
29	2	178956971	1	2	0 , 2	0 [268435455] 2 [268435455]
29	15	16385	1	2	0 , 2	0 [268435455] 2 [268435455]
29	28	268435455	1	2	0 , 2	0 [268435455] 2 [268435455]
30	2	-	3	2	2 , 2	0 [536870912] 2 [536870910]
30	15	-	32767	32766	32766 , 4	0 [536887295] 2 [536854527]
30	16	-	3	32768	2 , 32768	0 [536887295] 2 [536854527]
30	29	536870911	1	4	0 , 4	0 [536870912] 2 [536870910]
31	2	715827883	1	2	0 , 2	0 [1073741823] 2 [1073741823]
31	16	32769	1	2	0 , 2	0 [1073741823] 2 [1073741823]
31	30	1073741823	1	2	0 , 2	0 [1073741823] 2 [1073741823]
32	2	-	3	2	2 , 2	0 [2147483648] 2 [2147483646]
32	16	-	65535	65534	65534 , 2	0 [2147516414] 2 [2147450880]
32	17	1431677611	1	65536	0 , 65536	0 [2147516414] 2 [2147450880]
32	31	2147483647	1	4	0 , 4	0 [2147483648] 2 [2147483646]

TABLE A.3 – Spectre différentiel des fonctions $G_t(x) = x^{2^t-1}$ sur le corps $\mathbb{F}_{2^n}$ avec $\forall b \neq \{0,1\}$, $\delta(b) \leq 6$

Spectre différentiel des fonctions $G_t(x) = x^{2^t-1}$ sur le corps $\mathbb{F}_{2^n}$ avec $\forall b \neq \{0,1\}$, $\delta(b) \leq 6$						
n	t	inverse	pgcd	$\delta_{\max}$	δ_0, δ_1	Spectre différentiel sans $\delta(0), \delta(1)$
7	3	21	1	6	0 , 2	0 [77] 2 [42] 6 [7]
7	5	21	1	6	0 , 2	0 [77] 2 [42] 6 [7]
8	3	37	1	6	0 , 4	0 [156] 2 [84] 6 [14]
8	6	-	3	6	2 , 2	0 [156] 2 [84] 6 [14]
9	3	-	7	6	6 , 2	0 [300] 2 [189] 6 [21]
9	4	239	1	8	0 , 8	0 [300] 2 [189] 6 [21]
9	6	-	7	6	6 , 2	0 [300] 2 [189] 6 [21]
9	7	85	1	8	0 , 8	0 [300] 2 [189] 6 [21]
10	3	439	1	6	0 , 4	0 [582] 2 [405] 6 [35]
10	4	-	3	6	2 , 2	0 [582] 2 [405] 6 [35]
10	7	73	1	6	0 , 4	0 [582] 2 [405] 6 [35]
10	8	-	3	6	2 , 2	0 [582] 2 [405] 6 [35]
11	3	293	1	6	0 , 2	0 [1177] 2 [792] 6 [77]
11	4	137	1	6	0 , 2	0 [1177] 2 [792] 6 [77]
11	5	991	1	6	0 , 2	0 [1177] 2 [792] 6 [77]
11	7	887	1	6	0 , 2	0 [1177] 2 [792] 6 [77]
11	8	731	1	6	0 , 2	0 [1177] 2 [792] 6 [77]
11	9	341	1	6	0 , 2	0 [1177] 2 [792] 6 [77]
12	3	-	7	6	6 , 4	0 [2401] 2 [1518] 6 [175]
12	4	-	15	14	14 , 8	0 [2365] 2 [1575] 6 [154]
12	5	661	1	16	0 , 16	0 [2362] 2 [1578] 6 [154]
12	8	-	15	14	14 , 2	0 [2362] 2 [1578] 6 [154]
12	9	-	7	16	6 , 16	0 [2365] 2 [1575] 6 [154]
12	10	-	3	8	2 , 8	0 [2401] 2 [1518] 6 [175]

Spectre différentiel des fonctions $G_t(x) = x^{2^t-1}$ sur le corps $\mathbb{F}_{2^n}$ avec $\forall b \neq \{0,1\}$, $\delta(b) \leq 6$

n	t	inverse	pgcd	$\delta_{\max}$	δ_0, δ_1	Spectre différentiel sans $\delta(0), \delta(1)$
13	3	3511	1	6	0 , 2	0 [4823] 2 [3003] 6 [364]
13	4	3823	1	6	0 , 2	0 [4823] 2 [3003] 6 [364]
13	5	2907	1	6	0 , 2	0 [4823] 2 [3003] 6 [364]
13	6	4031	1	6	0 , 2	0 [4823] 2 [3003] 6 [364]
13	8	1189	1	6	0 , 2	0 [4823] 2 [3003] 6 [364]
13	9	273	1	6	0 , 2	0 [4823] 2 [3003] 6 [364]
13	10	585	1	6	0 , 2	0 [4823] 2 [3003] 6 [364]
13	11	1365	1	6	0 , 2	0 [4823] 2 [3003] 6 [364]
14	3	2341	1	6	0 , 4	0 [9578] 2 [6111] 6 [693]
14	5	529	1	6	0 , 4	0 [9578] 2 [6111] 6 [693]
14	10	-	3	6	2 , 2	0 [9578] 2 [6111] 6 [693]
14	12	-	3	6	2 , 2	0 [9578] 2 [6111] 6 [693]
15	3	-	7	6	6 , 2	0 [19046] 2 [12390] 6 [1330]
15	5	-	31	30	30 , 2	0 [19058] 2 [12378] 6 [1330]
15	6	-	7	32	6 , 32	0 [19061] 2 [12375] 6 [1330]
15	7	16255	1	8	0 , 8	0 [19046] 2 [12390] 6 [1330]
15	9	-	7	6	6 , 2	0 [19046] 2 [12390] 6 [1330]
15	10	-	31	30	30 , 8	0 [19061] 2 [12375] 6 [1330]
15	11	14199	1	32	0 , 32	0 [19058] 2 [12378] 6 [1330]
15	13	5461	1	8	0 , 8	0 [19046] 2 [12390] 6 [1330]
16	3	28087	1	6	0 , 4	0 [38116] 2 [24744] 6 [2674]
16	6	-	3	6	2 , 2	0 [38116] 2 [24744] 6 [2674]
16	11	1057	1	6	0 , 4	0 [38116] 2 [24744] 6 [2674]
16	14	-	3	6	2 , 2	0 [38116] 2 [24744] 6 [2674]
17	3	18725	1	6	0 , 2	0 [76483] 2 [49113] 6 [5474]
17	4	61167	1	6	0 , 2	0 [75531] 2 [50541] 6 [4998]
17	5	21141	1	6	0 , 2	0 [75531] 2 [50541] 6 [4998]
17	6	2081	1	6	0 , 2	0 [76483] 2 [49113] 6 [5474]
17	7	9289	1	6	0 , 2	0 [75531] 2 [50541] 6 [4998]
17	8	65279	1	6	0 , 2	0 [76483] 2 [49113] 6 [5474]
17	10	56247	1	6	0 , 2	0 [76483] 2 [49113] 6 [5474]
17	11	63455	1	6	0 , 2	0 [75531] 2 [50541] 6 [4998]
17	12	44395	1	6	0 , 2	0 [76483] 2 [49113] 6 [5474]
17	13	4369	1	6	0 , 2	0 [75531] 2 [50541] 6 [4998]
17	14	46811	1	6	0 , 2	0 [75531] 2 [50541] 6 [4998]
17	15	21845	1	6	0 , 2	0 [76483] 2 [49113] 6 [5474]
18	3	-	7	6	6 , 4	0 [153167] 2 [97929] 6 [11046]
18	6	-	63	62	62 , 2	0 [152564] 2 [98847] 6 [10731]
18	7	113527	1	64	0 , 64	0 [152564] 2 [98847] 6 [10731]
18	12	-	63	62	62 , 2	0 [152564] 2 [98847] 6 [10731]
18	13	38053	1	64	0 , 64	0 [152564] 2 [98847] 6 [10731]
18	16	-	3	8	2 , 8	0 [153167] 2 [97929] 6 [11046]
19	3	224695	1	6	0 , 2	0 [306033] 2 [196308] 6 [21945]
19	7	177515	1	6	0 , 2	0 [306033] 2 [196308] 6 [21945]
19	8	187099	1	6	0 , 2	0 [302309] 2 [201894] 6 [20083]

Spectre différentiel des fonctions $G_t(x) = x^{2^t-1}$ sur le corps $\mathbb{F}_{2^n}$ avec $\forall b \neq \{0,1\}$, $\delta(b) \leq 6$						
n	t	inverse	pgcd	$\delta_{\max}$	δ_0, δ_1	Spectre différentiel sans $\delta(0), \delta(1)$
19	9	261631	1	6	0 , 2	0 [306033] 2 [196308] 6 [21945]
19	11	75045	1	6	0 , 2	0 [306033] 2 [196308] 6 [21945]
19	12	84629	1	6	0 , 2	0 [302309] 2 [201894] 6 [20083]
19	13	4161	1	6	0 , 2	0 [306033] 2 [196308] 6 [21945]
19	17	87381	1	6	0 , 2	0 [306033] 2 [196308] 6 [21945]
20	3	149797	1	6	0 , 4	0 [611368] 2 [393666] 6 [43540]
20	7	8257	1	6	0 , 4	0 [611368] 2 [393666] 6 [43540]
20	14	-	3	6	2 , 2	0 [611368] 2 [393666] 6 [43540]
20	18	-	3	6	2 , 2	0 [611368] 2 [393666] 6 [43540]
21	3	-	7	6	6 , 2	0 [1222640] 2 [787479] 6 [87031]
21	7	-	127	126	126 , 8	0 [1222101] 2 [788319] 6 [86730]
21	8	304293	1	128	0 , 128	0 [1222098] 2 [788322] 6 [86730]
21	10	1047551	1	8	0 , 8	0 [1222640] 2 [787479] 6 [87031]
21	12	-	7	6	6 , 2	0 [1222640] 2 [787479] 6 [87031]
21	14	-	127	126	126 , 2	0 [1222098] 2 [788322] 6 [86730]
21	15	-	7	128	6 , 128	0 [1222101] 2 [788319] 6 [86730]
21	19	349525	1	8	0 , 8	0 [1222640] 2 [787479] 6 [87031]
22	3	1797559	1	6	0 , 4	0 [2446578] 2 [1573011] 6 [174713]
22	8	-	3	6	2 , 2	0 [2446578] 2 [1573011] 6 [174713]
22	15	16513	1	6	0 , 4	0 [2446578] 2 [1573011] 6 [174713]
22	20	-	3	6	2 , 2	0 [2446578] 2 [1573011] 6 [174713]
23	3	1198373	1	6	0 , 2	0 [4894653] 2 [3143778] 6 [350175]
23	8	32897	1	6	0 , 2	0 [4894653] 2 [3143778] 6 [350175]
23	11	4192255	1	6	0 , 2	0 [4894653] 2 [3143778] 6 [350175]
23	13	3595703	1	6	0 , 2	0 [4894653] 2 [3143778] 6 [350175]
23	16	2807211	1	6	0 , 2	0 [4894653] 2 [3143778] 6 [350175]
23	21	1398101	1	6	0 , 2	0 [4894653] 2 [3143778] 6 [350175]
24	3	-	7	6	6 , 4	0 [9788205] 2 [6289212] 6 [699797]
24	8	-	255	254	254 , 2	0 [9781202] 2 [6299778] 6 [696234]
24	9	-	7	256	6 , 256	0 [9781205] 2 [6299775] 6 [696234]
24	16	-	255	254	254 , 8	0 [9781205] 2 [6299775] 6 [696234]
24	17	7199671	1	256	0 , 256	0 [9781202] 2 [6299778] 6 [696234]
24	22	-	3	8	2 , 8	0 [9788205] 2 [6289212] 6 [699797]
25	3	14380471	1	6	0 , 2	0 [19572315] 2 [12584565] 6 [1397550]
25	9	11228587	1	6	0 , 2	0 [19572315] 2 [12584565] 6 [1397550]
25	12	16773119	1	6	0 , 2	0 [19572315] 2 [12584565] 6 [1397550]
25	14	4794661	1	6	0 , 2	0 [19572315] 2 [12584565] 6 [1397550]
25	17	65793	1	6	0 , 2	0 [19572315] 2 [12584565] 6 [1397550]
25	23	5592405	1	6	0 , 2	0 [19572315] 2 [12584565] 6 [1397550]
26	3	9586981	1	6	0 , 4	0 [39142742] 2 [25171965] 6 [2794155]
26	9	131329	1	6	0 , 4	0 [39142742] 2 [25171965] 6 [2794155]
26	18	-	3	6	2 , 2	0 [39142742] 2 [25171965] 6 [2794155]
26	24	-	3	6	2 , 2	0 [39142742] 2 [25171965] 6 [2794155]
27	3	-	7	6	6 , 2	0 [78291786] 2 [50334480] 6 [5591460]
27	9	-	511	510	510 , 2	0 [78272340] 2 [50363775] 6 [5581611]

Spectre différentiel des fonctions $G_t(x) = x^{2^t-1}$ sur le corps $\mathbb{F}_{2^n}$ avec $\forall b \neq \{0,1\}$, $\delta(b) \leq 6$						
n	t	inverse	pgcd	$\delta_{\max}$	δ_0, δ_1	Spectre différentiel sans $\delta(0), \delta(1)$
27	10	57596855	1	512	0 , 512	0 [78272340] 2 [50363775] 6 [5581611]
27	13	67100671	1	8	0 , 8	0 [78291786] 2 [50334480] 6 [5591460]
27	15	-	7	6	6 , 2	0 [78291786] 2 [50334480] 6 [5591460]
27	18	-	511	510	510 , 2	0 [78272340] 2 [50363775] 6 [5581611]
27	19	19211557	1	512	0 , 512	0 [78272340] 2 [50363775] 6 [5581611]
27	25	22369621	1	8	0 , 8	0 [78291786] 2 [50334480] 6 [5591460]
28	3	115043767	1	6	0 , 4	0 [156593648] 2 [100653846] 6 [11187960]
28	10	-	3	6	2 , 2	0 [156593648] 2 [100653846] 6 [11187960]
28	19	262657	1	6	0 , 4	0 [156593648] 2 [100653846] 6 [11187960]
28	26	-	3	6	2 , 2	0 [156593648] 2 [100653846] 6 [11187960]
29	3	76695845	1	6	0 , 2	0 [313184775] 2 [201311475] 6 [22374660]
29	10	524801	1	6	0 , 2	0 [313184775] 2 [201311475] 6 [22374660]
29	14	268419071	1	6	0 , 2	0 [313184775] 2 [201311475] 6 [22374660]
29	16	230092215	1	6	0 , 2	0 [313184775] 2 [201311475] 6 [22374660]
29	20	179132075	1	6	0 , 2	0 [313184775] 2 [201311475] 6 [22374660]
29	27	89478485	1	6	0 , 2	0 [313184775] 2 [201311475] 6 [22374660]
30	3	-	7	6	6 , 4	0 [626346875] 2 [402656967] 6 [44737980]
30	10	-	1023	1022	1022 , 8	0 [626261845] 2 [402784767] 6 [44695210]
30	11	153691429	1	1024	0 , 1024	0 [626261842] 2 [402784770] 6 [44695210]
30	20	-	1023	1022	1022 , 2	0 [626261842] 2 [402784770] 6 [44695210]
30	21	-	7	1024	6 , 1024	0 [626261845] 2 [402784767] 6 [44695210]
30	28	-	3	8	2 , 8	0 [626346875] 2 [402656967] 6 [44737980]
31	3	920350135	1	6	0 , 2	0 [1252676117] 2 [805340382] 6 [89467147]
31	11	716527275	1	6	0 , 2	0 [1252676117] 2 [805340382] 6 [89467147]
31	15	1073709055	1	6	0 , 2	0 [1252676117] 2 [805340382] 6 [89467147]
31	17	306792741	1	6	0 , 2	0 [1252676117] 2 [805340382] 6 [89467147]
31	21	1049601	1	6	0 , 2	0 [1252676117] 2 [805340382] 6 [89467147]
31	29	357913941	1	6	0 , 2	0 [1252676117] 2 [805340382] 6 [89467147]
32	3	613566757	1	6	0 , 4	0 [2505379956] 2 [1610639184] 6 [178948154]
32	11	2098177	1	6	0 , 4	0 [2505379956] 2 [1610639184] 6 [178948154]
32	22	-	3	6	2 , 2	0 [2505379956] 2 [1610639184] 6 [178948154]
32	30	-	3	6	2 , 2	0 [2505379956] 2 [1610639184] 6 [178948154]

TABLE A.4 – Spectre différentiel des fonctions $G_t(x) = x^{2^t-1}$ sur le corps $\mathbb{F}_{2^n}$ avec $\forall b \neq \{0,1\}$, $\delta(b) \leq 14$

Spectre différentiel des fonctions $G_t(x) = x^{2^t-1}$ sur le corps $\mathbb{F}_{2^n}$ avec $\forall b \neq \{0,1\}$, $\delta(b) \leq 14$						
n	t	inverse	pgcd	$\delta_{\max}$	δ_0, δ_1	Spectre différentiel sans $\delta(0), \delta(1)$
14	4	-	3	14	2 , 2	0 [9548] 2 [6216] 6 [588] 14 [30]
14	6	-	3	14	2 , 2	0 [9548] 2 [6216] 6 [588] 14 [30]
14	9	7663	1	14	0 , 4	0 [9548] 2 [6216] 6 [588] 14 [30]
14	11	5851	1	14	0 , 4	0 [9548] 2 [6216] 6 [588] 14 [30]
15	4	2185	1	14	0 , 8	0 [18786] 2 [12810] 6 [1155] 14 [15]
15	12	-	7	14	6 , 2	0 [18786] 2 [12810] 6 [1155] 14 [15]
16	4	-	15	14	14 , 2	0 [37838] 2 [25284] 6 [2352] 14 [60]
16	5	31711	1	16	0 , 16	0 [37838] 2 [25284] 6 [2352] 14 [60]

Spectre différentiel des fonctions $G_t(x) = x^{2^t-1}$ sur le corps $\mathbb{F}_{2^n}$ avec $\forall b \neq \{0,1\}$, $\delta(b) \leq 14$						
n	t	inverse	pgcd	$\delta_{\max}$	δ_0, δ_1	Spectre différentiel sans $\delta(0), \delta(1)$
16	7	10837	1	14	0 , 4	0 [38136] 2 [24954] 6 [2324] 14 [120]
16	10	-	3	14	2 , 2	0 [38136] 2 [24954] 6 [2324] 14 [120]
16	12	-	15	14	14 , 2	0 [37838] 2 [25284] 6 [2352] 14 [60]
16	13	4681	1	16	0 , 16	0 [37838] 2 [25284] 6 [2352] 14 [60]
18	4	-	3	14	2 , 8	0 [151223] 2 [101205] 6 [9534] 14 [180]
18	5	93019	1	14	0 , 4	0 [151058] 2 [101271] 6 [9723] 14 [90]
18	8	-	3	14	2 , 2	0 [151058] 2 [101271] 6 [9723] 14 [90]
18	11	17545	1	14	0 , 4	0 [151058] 2 [101271] 6 [9723] 14 [90]
18	14	-	3	14	2 , 2	0 [151058] 2 [101271] 6 [9723] 14 [90]
18	15	-	7	14	6 , 4	0 [151223] 2 [101205] 6 [9534] 14 [180]
19	4	34953	1	14	0 , 2	0 [303335] 2 [201495] 6 [18886] 14 [570]
19	5	16913	1	14	0 , 2	0 [303335] 2 [201495] 6 [18886] 14 [570]
19	6	257983	1	14	0 , 2	0 [303335] 2 [201495] 6 [18886] 14 [570]
19	14	245231	1	14	0 , 2	0 [303335] 2 [201495] 6 [18886] 14 [570]
19	15	227191	1	14	0 , 2	0 [303335] 2 [201495] 6 [18886] 14 [570]
19	16	37449	1	14	0 , 2	0 [303335] 2 [201495] 6 [18886] 14 [570]
20	4	-	15	14	14 , 2	0 [606074] 2 [403110] 6 [38640] 14 [750]
20	5	-	31	30	30 , 16	0 [605739] 2 [403620] 6 [38465] 14 [750]
20	6	-	3	32	2 , 32	0 [605203] 2 [404571] 6 [37975] 14 [825]
20	8	-	15	14	14 , 2	0 [606074] 2 [403110] 6 [38640] 14 [750]
20	9	174421	1	16	0 , 16	0 [606134] 2 [403320] 6 [38220] 14 [900]
20	12	-	15	14	14 , 2	0 [606134] 2 [403320] 6 [38220] 14 [900]
20	13	516031	1	16	0 , 16	0 [606074] 2 [403110] 6 [38640] 14 [750]
20	15	-	31	30	30 , 4	0 [605203] 2 [404571] 6 [37975] 14 [825]
20	16	-	15	32	14 , 32	0 [605739] 2 [403620] 6 [38465] 14 [750]
20	17	374491	1	16	0 , 16	0 [606074] 2 [403110] 6 [38640] 14 [750]
21	4	978671	1	14	0 , 8	0 [1211640] 2 [808059] 6 [75411] 14 [2040]
21	6	-	7	14	6 , 2	0 [1211640] 2 [808059] 6 [75411] 14 [2040]
21	9	-	7	14	6 , 2	0 [1211398] 2 [806022] 6 [78890] 14 [840]
21	13	744283	1	14	0 , 8	0 [1211398] 2 [806022] 6 [78890] 14 [840]
21	16	33825	1	14	0 , 8	0 [1211640] 2 [808059] 6 [75411] 14 [2040]
21	18	-	7	14	6 , 2	0 [1211640] 2 [808059] 6 [75411] 14 [2040]
22	4	-	3	14	2 , 2	0 [2421454] 2 [1615977] 6 [154231] 14 [2640]
22	5	676501	1	14	0 , 4	0 [2422686] 2 [1614129] 6 [154847] 14 [2640]
22	6	-	3	14	2 , 2	0 [2422488] 2 [1615746] 6 [152768] 14 [3300]
22	7	2080639	1	14	0 , 4	0 [2421454] 2 [1615977] 6 [154231] 14 [2640]
22	9	139537	1	14	0 , 4	0 [2422686] 2 [1614129] 6 [154847] 14 [2640]
22	10	-	3	14	2 , 2	0 [2422488] 2 [1615746] 6 [152768] 14 [3300]
22	13	1957615	1	14	0 , 4	0 [2422488] 2 [1615746] 6 [152768] 14 [3300]
22	14	-	3	14	2 , 2	0 [2422686] 2 [1614129] 6 [154847] 14 [2640]
22	16	-	3	14	2 , 2	0 [2421454] 2 [1615977] 6 [154231] 14 [2640]
22	17	1420651	1	14	0 , 4	0 [2422488] 2 [1615746] 6 [152768] 14 [3300]
22	18	-	3	14	2 , 2	0 [2422686] 2 [1614129] 6 [154847] 14 [2640]
22	19	299593	1	14	0 , 4	0 [2421454] 2 [1615977] 6 [154231] 14 [2640]
23	4	559241	1	14	0 , 2	0 [4841477] 2 [3234582] 6 [307027] 14 [5520]

Spectre différentiel des fonctions $G_t(x) = x^{2^t-1}$ sur le corps $\mathbb{F}_{2^n}$ avec $\forall b \neq \{0,1\}$, $\delta(b) \leq 14$

n	t	inverse	pgcd	$\delta_{\max}$	δ_0, δ_1	Spectre différentiel sans $\delta(0), \delta(1)$
23	5	2976603	1	14	0 , 2	0 [4845295] 2 [3230235] 6 [306866] 14 [6210]
23	6	133153	1	14	0 , 2	0 [4841477] 2 [3234582] 6 [307027] 14 [5520]
23	7	1387093	1	14	0 , 2	0 [4845295] 2 [3230235] 6 [306866] 14 [6210]
23	9	3923439	1	14	0 , 2	0 [4841477] 2 [3234582] 6 [307027] 14 [5520]
23	10	598601	1	14	0 , 2	0 [4845295] 2 [3230235] 6 [306866] 14 [6210]
23	14	270865	1	14	0 , 2	0 [4845295] 2 [3230235] 6 [306866] 14 [6210]
23	15	4161407	1	14	0 , 2	0 [4841477] 2 [3234582] 6 [307027] 14 [5520]
23	17	4061151	1	14	0 , 2	0 [4845295] 2 [3230235] 6 [306866] 14 [6210]
23	18	1217701	1	14	0 , 2	0 [4841477] 2 [3234582] 6 [307027] 14 [5520]
23	19	3635063	1	14	0 , 2	0 [4845295] 2 [3230235] 6 [306866] 14 [6210]
23	20	2995931	1	14	0 , 2	0 [4841477] 2 [3234582] 6 [307027] 14 [5520]
24	4	-	15	14	14 , 8	0 [9681841] 2 [6473481] 6 [609532] 14 [12360]
24	6	-	63	62	62 , 2	0 [9687702] 2 [6463380] 6 [614432] 14 [11700]
24	7	1188937	1	64	0 , 64	0 [9683814] 2 [6471612] 6 [608888] 14 [12900]
24	11	2794837	1	14	0 , 4	0 [9687668] 2 [6465216] 6 [611730] 14 [12600]
24	14	-	3	14	2 , 2	0 [9687668] 2 [6465216] 6 [611730] 14 [12600]
24	18	-	63	62	62 , 2	0 [9683814] 2 [6471612] 6 [608888] 14 [12900]
24	19	7847407	1	64	0 , 64	0 [9687702] 2 [6463380] 6 [614432] 14 [11700]
24	21	-	7	16	6 , 16	0 [9681841] 2 [6473481] 6 [609532] 14 [12360]
25	4	15658735	1	14	0 , 2	0 [19368015] 2 [12940515] 6 [1221150] 14 [24750]
25	5	-	31	30	30 , 2	0 [19377280] 2 [12922125] 6 [1232525] 14 [22500]
25	6	16510911	1	32	0 , 32	0 [19377280] 2 [12922125] 6 [1232525] 14 [22500]
25	7	14531447	1	14	0 , 2	0 [19368015] 2 [12940515] 6 [1221150] 14 [24750]
25	8	16711423	1	14	0 , 2	0 [19368015] 2 [12940515] 6 [1221150] 14 [24750]
25	18	2245769	1	14	0 , 2	0 [19368015] 2 [12940515] 6 [1221150] 14 [24750]
25	19	266305	1	14	0 , 2	0 [19368015] 2 [12940515] 6 [1221150] 14 [24750]
25	20	-	31	30	30 , 2	0 [19377280] 2 [12922125] 6 [1232525] 14 [22500]
25	21	1118481	1	32	0 , 32	0 [19377280] 2 [12922125] 6 [1232525] 14 [22500]
25	22	2396745	1	14	0 , 2	0 [19368015] 2 [12940515] 6 [1221150] 14 [24750]
26	4	-	3	14	2 , 2	0 [38755888] 2 [25853646] 6 [2448628] 14 [50700]
26	7	22721899	1	14	0 , 4	0 [38748062] 2 [25865385] 6 [2444715] 14 [50700]
26	8	-	3	14	2 , 2	0 [38749206] 2 [25859379] 6 [2451722] 14 [48555]
26	10	-	3	14	2 , 2	0 [38755888] 2 [25853646] 6 [2448628] 14 [50700]
26	11	29079415	1	14	0 , 4	0 [38749206] 2 [25859379] 6 [2451722] 14 [48555]
26	12	-	3	14	2 , 2	0 [38748062] 2 [25865385] 6 [2444715] 14 [50700]
26	15	4475017	1	14	0 , 4	0 [38748062] 2 [25865385] 6 [2444715] 14 [50700]
26	16	-	3	14	2 , 2	0 [38749206] 2 [25859379] 6 [2451722] 14 [48555]
26	17	33423103	1	14	0 , 4	0 [38755888] 2 [25853646] 6 [2448628] 14 [50700]
26	19	10832533	1	14	0 , 4	0 [38749206] 2 [25859379] 6 [2451722] 14 [48555]
26	20	-	3	14	2 , 2	0 [38748062] 2 [25865385] 6 [2444715] 14 [50700]
26	23	23967451	1	14	0 , 4	0 [38755888] 2 [25853646] 6 [2448628] 14 [50700]
27	4	8947849	1	14	0 , 8	0 [77496654] 2 [51731568] 6 [4887309] 14 [102195]
27	5	21648021	1	14	0 , 2	0 [77516427] 2 [51691377] 6 [4912992] 14 [96930]
27	7	1056833	1	14	0 , 8	0 [77496654] 2 [51731568] 6 [4887309] 14 [102195]
27	8	47897307	1	14	0 , 2	0 [77516427] 2 [51691377] 6 [4912992] 14 [96930]

Spectre différentiel des fonctions $G_t(x) = x^{2^t-1}$ sur le corps $\mathbb{F}_{2^n}$ avec $\forall b \neq \{0,1\}$, $\delta(b) \leq 14$

n	t	inverse	pgcd	$\delta_{\max}$	δ_0, δ_1	Spectre différentiel sans $\delta(0), \delta(1)$
27	11	2163745	1	14	0 , 2	0 [77516427] 2 [51691377] 6 [4912992] 14 [96930]
27	17	9512009	1	14	0 , 2	0 [77516427] 2 [51691377] 6 [4912992] 14 [96930]
27	20	66052031	1	14	0 , 2	0 [77516427] 2 [51691377] 6 [4912992] 14 [96930]
27	21	-	7	14	6 , 2	0 [77496654] 2 [51731568] 6 [4887309] 14 [102195]
27	23	58161015	1	14	0 , 2	0 [77516427] 2 [51691377] 6 [4912992] 14 [96930]
27	24	-	7	14	6 , 2	0 [77496654] 2 [51731568] 6 [4887309] 14 [102195]
28	4	-	15	14	14 , 2	0 [154986630] 2 [103467336] 6 [9780008] 14 [201480]
28	7	-	127	126	126 , 4	0 [155001911] 2 [103443423] 6 [9789150] 14 [200970]
28	8	-	15	128	14 , 128	0 [155017415] 2 [103410720] 6 [9811074] 14 [196245]
28	9	133955071	1	16	0 , 16	0 [154986630] 2 [103467336] 6 [9780008] 14 [201480]
28	13	44733781	1	16	0 , 16	0 [154996274] 2 [103448310] 6 [9791670] 14 [199200]
28	16	-	15	14	14 , 2	0 [154996274] 2 [103448310] 6 [9791670] 14 [199200]
28	20	-	15	14	14 , 2	0 [154986630] 2 [103467336] 6 [9780008] 14 [201480]
28	21	-	127	126	126 , 16	0 [155017415] 2 [103410720] 6 [9811074] 14 [196245]
28	22	-	3	128	2 , 128	0 [155001911] 2 [103443423] 6 [9789150] 14 [200970]
28	25	19173961	1	16	0 , 16	0 [154986630] 2 [103467336] 6 [9780008] 14 [201480]
29	4	250539759	1	14	0 , 2	0 [309989671] 2 [206904531] 6 [19576508] 14 [400200]
29	8	77898917	1	14	0 , 2	0 [309989671] 2 [206904531] 6 [19576508] 14 [400200]
29	11	35931273	1	14	0 , 2	0 [309989671] 2 [206904531] 6 [19576508] 14 [400200]
29	19	267910655	1	14	0 , 2	0 [309989671] 2 [206904531] 6 [19576508] 14 [400200]
29	22	2113665	1	14	0 , 2	0 [309989671] 2 [206904531] 6 [19576508] 14 [400200]
29	26	191739611	1	14	0 , 2	0 [309989671] 2 [206904531] 6 [19576508] 14 [400200]
30	4	-	3	14	2 , 8	0 [619954315] 2 [413841567] 6 [39148060] 14 [797880]
30	8	-	3	14	2 , 2	0 [619931492] 2 [413867820] 6 [39148620] 14 [793890]
30	14	-	3	14	2 , 2	0 [619931492] 2 [413867820] 6 [39148620] 14 [793890]
30	17	501083887	1	14	0 , 4	0 [619931492] 2 [413867820] 6 [39148620] 14 [793890]
30	23	359323051	1	14	0 , 4	0 [619931492] 2 [413867820] 6 [39148620] 14 [793890]
30	27	-	7	14	6 , 4	0 [619954315] 2 [413841567] 6 [39148060] 14 [797880]
31	4	143165577	1	14	0 , 2	0 [1239931087] 2 [827653407] 6 [78301412] 14 [1597740]
31	8	8421505	1	14	0 , 2	0 [1239931087] 2 [827653407] 6 [78301412] 14 [1597740]
31	10	1072692223	1	14	0 , 2	0 [1239931087] 2 [827653407] 6 [78301412] 14 [1597740]
31	22	1002299119	1	14	0 , 2	0 [1239931087] 2 [827653407] 6 [78301412] 14 [1597740]
31	24	921557943	1	14	0 , 2	0 [1239931087] 2 [827653407] 6 [78301412] 14 [1597740]
31	28	153391689	1	14	0 , 2	0 [1239931087] 2 [827653407] 6 [78301412] 14 [1597740]
32	4	-	15	14	14 , 2	0 [2479823822] 2 [1655355108] 6 [156597504] 14 [3190860]
32	8	-	255	254	254 , 2	0 [2479848702] 2 [1655306688] 6 [156626624] 14 [3185280]

Spectre différentiel des fonctions $G_t(x) = x^{2^t-1}$ sur le corps $\mathbb{F}_{2^n}$ avec $\forall b \neq \{0,1\}$, $\delta(b) \leq 14$						
n	t	inverse	pgcd	$\delta_{\max}$	δ_0, δ_1	Spectre différentiel sans $\delta(0), \delta(1)$
32	9	2008800751	1	256	0 , 256	0 [2479848702] 2 [1655306688] 6 [156626624] 14 [3185280]
32	12	-	15	14	14 , 2	0 [2479823822] 2 [1655355108] 6 [156597504] 14 [3190860]
32	15	715806037	1	14	0 , 4	0 [2479908336] 2 [1655272374] 6 [156573704] 14 [3212880]
32	18	-	3	14	2 , 2	0 [2479908336] 2 [1655272374] 6 [156573704] 14 [3212880]
32	21	2145385471	1	16	0 , 16	0 [2479823822] 2 [1655355108] 6 [156597504] 14 [3190860]
32	24	-	255	254	254 , 2	0 [2479848702] 2 [1655306688] 6 [156626624] 14 [3185280]
32	25	287458441	1	256	0 , 256	0 [2479848702] 2 [1655306688] 6 [156626624] 14 [3185280]
32	29	1533916891	1	16	0 , 16	0 [2479823822] 2 [1655355108] 6 [156597504] 14 [3190860]

TABLE A.5 – Spectre différentiel des fonctions $G_t(x) = x^{2^t-1}$ sur le corps $\mathbb{F}_{2^n}$ avec $\forall b \neq \{0,1\}$, $\delta(b) \leq 30$

Spectre différentiel des fonctions $G_t(x) = x^{2^t-1}$ sur le corps $\mathbb{F}_{2^n}$ avec $\forall b \neq \{0,1\}$, $\delta(b) \leq 30$						
n	t	inverse	pgcd	$\delta_{\max}$	δ_0, δ_1	Spectre différentiel sans $\delta(0), \delta(1)$
21	5	1014751	1	30	0 , 2	0 [1211217] 2 [807828] 6 [76559] 14 [1515] 30 [31]
21	17	69905	1	30	0 , 2	0 [1211217] 2 [807828] 6 [76559] 14 [1515] 30 [31]
24	5	541201	1	30	0 , 16	0 [9700642] 2 [6451422] 6 [609966] 14 [15060] 30 [124]
24	10	-	3	30	2 , 8	0 [9690937] 2 [6461730] 6 [611611] 14 [12750] 30 [186]
24	15	-	7	30	6 , 4	0 [9690937] 2 [6461730] 6 [611611] 14 [12750] 30 [186]
24	20	-	15	30	14 , 2	0 [9700642] 2 [6451422] 6 [609966] 14 [15060] 30 [124]
25	10	-	31	30	30 , 2	0 [19378250] 2 [12920850] 6 [1233050] 14 [22125] 30 [155]
25	11	11982555	1	32	0 , 32	0 [19378250] 2 [12920850] 6 [1233050] 14 [22125] 30 [155]
25	15	-	31	30	30 , 2	0 [19378250] 2 [12920850] 6 [1233050] 14 [22125] 30 [155]
25	16	5548629	1	32	0 , 32	0 [19378250] 2 [12920850] 6 [1233050] 14 [22125] 30 [155]
26	5	32472031	1	30	0 , 4	0 [38743980] 2 [25868076] 6 [2448628] 14 [47775] 30 [403]
26	6	-	3	30	2 , 2	0 [38743980] 2 [25868076] 6 [2448628] 14 [47775] 30 [403]
26	21	1082401	1	30	0 , 4	0 [38743980] 2 [25868076] 6 [2448628] 14 [47775] 30 [403]
26	22	-	3	30	2 , 2	0 [38743980] 2 [25868076] 6 [2448628] 14 [47775] 30 [403]

Spectre différentiel des fonctions $G_t(x) = x^{2^t-1}$ sur le corps $\mathbb{F}_{2^n}$ avec $\forall b \neq \{0,1\}$, $\delta(b) \leq 30$

n	t	inverse	pgcd	$\delta_{\max}$	δ_0, δ_1	Spectre différentiel sans $\delta(0), \delta(1)$
27	6	-	7	30	6 , 2	0 [77525544] 2 [51693849] 6 [4895562] 14 [101655] 30 [1116]
27	12	-	7	30	6 , 2	0 [77525544] 2 [51693849] 6 [4895562] 14 [101655] 30 [1116]
27	16	64945119	1	30	0 , 8	0 [77525544] 2 [51693849] 6 [4895562] 14 [101655] 30 [1116]
27	22	45460843	1	30	0 , 8	0 [77525544] 2 [51693849] 6 [4895562] 14 [101655] 30 [1116]
28	5	95251291	1	30	0 , 16	0 [155060664] 2 [103314309] 6 [9880857] 14 [179190] 30 [434]
28	6	-	3	30	2 , 2	0 [155019036] 2 [103395012] 6 [9835378] 14 [184230] 30 [1798]
28	11	129955807	1	30	0 , 4	0 [155019036] 2 [103395012] 6 [9835378] 14 [184230] 30 [1798]
28	12	-	15	30	14 , 2	0 [155060664] 2 [103314309] 6 [9880857] 14 [179190] 30 [434]
28	17	4261921	1	30	0 , 16	0 [155060664] 2 [103314309] 6 [9880857] 14 [179190] 30 [434]
28	18	-	3	30	2 , 2	0 [155019036] 2 [103395012] 6 [9835378] 14 [184230] 30 [1798]
28	23	38966437	1	30	0 , 4	0 [155019036] 2 [103395012] 6 [9835378] 14 [184230] 30 [1798]
28	24	-	15	30	14 , 2	0 [155060664] 2 [103314309] 6 [9880857] 14 [179190] 30 [434]
29	5	17318417	1	30	0 , 2	0 [310040827] 2 [206754021] 6 [19714548] 14 [360615] 30 [899]
29	6	8521761	1	30	0 , 2	0 [310040827] 2 [206754021] 6 [19714548] 14 [360615] 30 [899]
29	7	266321791	1	30	0 , 2	0 [310040827] 2 [206754021] 6 [19714548] 14 [360615] 30 [899]
29	9	89303381	1	30	0 , 2	0 [310094303] 2 [206689467] 6 [19717796] 14 [368445] 30 [899]
29	12	181841259	1	30	0 , 2	0 [310094303] 2 [206689467] 6 [19717796] 14 [368445] 30 [899]
29	13	38343241	1	30	0 , 2	0 [310094303] 2 [206689467] 6 [19717796] 14 [368445] 30 [899]
29	17	86594197	1	30	0 , 2	0 [310094303] 2 [206689467] 6 [19717796] 14 [368445] 30 [899]
29	18	232504183	1	30	0 , 2	0 [310094303] 2 [206689467] 6 [19717796] 14 [368445] 30 [899]
29	21	190536539	1	30	0 , 2	0 [310094303] 2 [206689467] 6 [19717796] 14 [368445] 30 [899]
29	23	259913695	1	30	0 , 2	0 [310040827] 2 [206754021] 6 [19714548] 14 [360615] 30 [899]

Spectre différentiel des fonctions $G_t(x) = x^{2^t-1}$ sur le corps $\mathbb{F}_{2^n}$ avec $\forall b \neq \{0,1\}$, $\delta(b) \leq 30$

n	t	inverse	pgcd	$\delta_{\max}$	δ_0, δ_1	Spectre différentiel sans $\delta(0), \delta(1)$
29	24	251117039	1	30	0 , 2	0 [310040827] 2 [206754021] 6 [19714548] 14 [360615] 30 [899]
29	25	17895697	1	30	0 , 2	0 [310040827] 2 [206754021] 6 [19714548] 14 [360615] 30 [899]
30	5	-	31	30	30 , 4	0 [620184815] 2 [413429025] 6 [39372305] 14 [751275] 30 [4402]
30	6	-	63	62	62 , 32	0 [620125101] 2 [413500845] 6 [39367160] 14 [745275] 30 [3441]
30	7	177547861	1	64	0 , 64	0 [620200006] 2 [413410980] 6 [39370870] 14 [756525] 30 [3441]
30	9	-	7	30	6 , 4	0 [620148195] 2 [413469807] 6 [39371990] 14 [749040] 30 [2790]
30	12	-	63	62	62 , 2	0 [620174850] 2 [413388690] 6 [39445910] 14 [730140] 30 [2232]
30	13	35787025	1	64	0 , 64	0 [620200006] 2 [413410980] 6 [39370870] 14 [756525] 30 [3441]
30	18	-	63	62	62 , 2	0 [620200006] 2 [413410980] 6 [39370870] 14 [756525] 30 [3441]
30	19	383179483	1	64	0 , 64	0 [620174850] 2 [413388690] 6 [39445910] 14 [730140] 30 [2232]
30	22	-	3	30	2 , 8	0 [620148195] 2 [413469807] 6 [39371990] 14 [749040] 30 [2790]
30	24	-	63	62	62 , 2	0 [620200006] 2 [413410980] 6 [39370870] 14 [756525] 30 [3441]
30	25	-	31	64	30 , 64	0 [620125101] 2 [413500845] 6 [39367160] 14 [745275] 30 [3441]
30	26	-	3	32	2 , 32	0 [620184815] 2 [413429025] 6 [39372305] 14 [751275] 30 [4402]
31	5	1039104991	1	30	0 , 2	0 [1240285479] 2 [827005569] 6 [78681813] 14 [1496370] 30 [14415]
31	6	1056698303	1	30	0 , 2	0 [1240370295] 2 [826844679] 6 [78763188] 14 [1496835] 30 [8649]
31	7	152183881	1	30	0 , 2	0 [1240285479] 2 [827005569] 6 [78681813] 14 [1496370] 30 [14415]
31	9	71442705	1	30	0 , 2	0 [1240285479] 2 [827005569] 6 [78681813] 14 [1496370] 30 [14415]
31	12	346638997	1	30	0 , 2	0 [1240342519] 2 [826901037] 6 [78723477] 14 [1509855] 30 [6758]
31	13	311727269	1	30	0 , 2	0 [1240342519] 2 [826901037] 6 [78723477] 14 [1509855] 30 [6758]
31	14	766949083	1	30	0 , 2	0 [1240342519] 2 [826901037] 6 [78723477] 14 [1509855] 30 [6758]
31	18	762014555	1	30	0 , 2	0 [1240342519] 2 [826901037] 6 [78723477] 14 [1509855] 30 [6758]

Spectre différentiel des fonctions $G_t(x) = x^{2^t-1}$ sur le corps $\mathbb{F}_{2^n}$ avec $\forall b \neq \{0,1\}$, $\delta(b) \leq 30$

n	t	inverse	pgcd	$\delta_{\max}$	δ_0, δ_1	Spectre différentiel sans $\delta(0), \delta(1)$
31	19	727102827	1	30	0 , 2	0 [1240342519] 2 [826901037] 6 [78723477] 14 [1509855] 30 [6758]
31	20	357214549	1	30	0 , 2	0 [1240342519] 2 [826901037] 6 [78723477] 14 [1509855] 30 [6758]
31	23	1065320319	1	30	0 , 2	0 [1240285479] 2 [827005569] 6 [78681813] 14 [1496370] 30 [14415]
31	25	17043521	1	30	0 , 2	0 [1240285479] 2 [827005569] 6 [78681813] 14 [1496370] 30 [14415]
31	26	34636833	1	30	0 , 2	0 [1240370295] 2 [826844679] 6 [78763188] 14 [1496835] 30 [8649]
31	27	930576247	1	30	0 , 2	0 [1240285479] 2 [827005569] 6 [78681813] 14 [1496370] 30 [14415]
32	5	692736661	1	30	0 , 16	0 [2480687262] 2 [1653779148] 6 [157478944] 14 [3007680] 30 [14260]
32	6	-	3	30	2 , 2	0 [2480635380] 2 [1653750768] 6 [157606106] 14 [2963880] 30 [11160]
32	7	1860025207	1	30	0 , 4	0 [2480672648] 2 [1653714450] 6 [157596852] 14 [2971440] 30 [11904]
32	10	-	3	30	2 , 2	0 [2480672648] 2 [1653714450] 6 [157596852] 14 [2971440] 30 [11904]
32	13	34082881	1	30	0 , 16	0 [2480687262] 2 [1653779148] 6 [157478944] 14 [3007680] 30 [14260]
32	14	-	3	30	2 , 2	0 [2480635380] 2 [1653750768] 6 [157606106] 14 [2963880] 30 [11160]
32	19	2113400767	1	30	0 , 4	0 [2480635380] 2 [1653750768] 6 [157606106] 14 [2963880] 30 [11160]
32	20	-	15	30	14 , 2	0 [2480687262] 2 [1653779148] 6 [157478944] 14 [3007680] 30 [14260]
32	23	138682897	1	30	0 , 4	0 [2480672648] 2 [1653714450] 6 [157596852] 14 [2971440] 30 [11904]
32	26	-	3	30	2 , 2	0 [2480672648] 2 [1653714450] 6 [157596852] 14 [2971440] 30 [11904]
32	27	1454746987	1	30	0 , 4	0 [2480635380] 2 [1653750768] 6 [157606106] 14 [2963880] 30 [11160]
32	28	-	15	30	14 , 2	0 [2480687262] 2 [1653779148] 6 [157478944] 14 [3007680] 30 [14260]

Bibliographie

[ABNP+11] Mohamed Ahmed Abdelraheem, Céline Blondeau, María Naya-Plasencia, Marion Videau, and Erik Zenner. Cryptanalysis of ARMADILLO2. In D.H. Lee and X. Wang, editors, *Asiacrypt 2011*, volume 7073 of *Lecture Notes in Computer Science*, pages 308–326. Springer, 2011.

[AC09] Martin Albrecht and Carlos Cid. Algebraic Techniques in Differential Cryptanalysis. In Orr Dunkelman, editor, *FSE*, volume 5665 of *Lecture Notes in Computer Science*, pages 193–208. Springer, 2009.

[AFK+08] Jean-Philippe Aumasson, Simon Fischer, Shahram Khazaei, Willi Meier, and Christian Rechberger. New Features of Latin Dances : Analysis of Salsa, ChaCha, and Rumba. In *Fast Software Encryption ,FSE 2008*, volume 5086 of *Lecture Notes in Computer Science*, pages 470–488. Springer, 2008.

[AG89] R. Arriata and L. Gordon. Tutorial on large deviations for the binomial distribution. *Bulletin of Mathematical Biology*, 51(1) :125–131, 1989.

[AIK+00] Kazumaro Aoki, Tetsuya Ichikawa, Masayuki Kanda, Mitsuru Matsui, Shiho Moriai, Junko Nakajima, and Toshio Tokita. Camellia : A 128-Bit Block Cipher Suitable for Multiple Platforms - Design and Analysis. In Douglas R. Stinson and Stafford E. Tavares, editors, *Selected Areas in Cryptography, SAC 2000*, volume 2012 of *Lecture Notes in Computer Science*, pages 39–56. Springer, 2000.

[AM09] J.-P. Aumasson and W. Meier. Zero-sum distinguishers for reduced Keccak-f and for the core functions of Luffa and Hamsi. Presented at the rump session of Cryptographic Hardware and Embedded Systems - CHES 2009, 2009.

[AS64] Milton Abramowitz and Irene A. StegunM. Handbook of mathematical functions. Courier Dover Publications, 1964.

[ASB08] Bora Aslan, M. Tolga Sakalli, and Ercan Bulus. Classifying 8-Bit to 8-Bit S-Boxes Based on Power Mappings from the Point of DDT and LAT Distributions. In Joachim von zur Gathen, José Luis Imaña, and Çetin Kaya Koç, editors, *Arithmetic of Finite Fields, WAIFI 2008*, volume 5130 of *Lecture Notes in Computer Science*, pages 123–133. Springer, 2008.

[BBS99] Eli Biham, Alex Biryukov, and Adi Shamir. Cryptanalysis of Skipjack Reduced to 31 Rounds Using Impossible Differentials. In *EUROCRYPT '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 12–23. Springer, 1999.

[BC10] Christina Boura and Anne Canteaut. Zero-Sum Distinguishers for Iterated Permutations and Application to Keccak- and Hamsi-256. In Alex Biryukov, Guang Gong, and Douglas R. Stinson, editors, *Selected Areas in Cryptography, SAC 2010*, volume 6544 of *Lecture Notes in Computer Science*, pages 1–17. Springer, 2010.

[BCC10a] Céline Blondeau, Anne Canteaut, and Pascale Charpin. Differential Properties of Power Functions. In *Proceedings of the 2010 IEEE International Symposium on Information Theory, ISIT 10*, pages 2478–2482, 2010.

[BCC10b] Céline Blondeau, Anne Canteaut, and Pascale Charpin. Differential Properties of Power Functions. *Int. J. Inform. and Coding Theory*, 1(2) :149–170, 2010.

[BCC11] Céline Blondeau, Anne Canteaut, and Pascale Charpin. Differential Properties of $x \mapsto x^{2^t-1}$. *IEEE Trans. Inform. Theory*, 57, December 2011.

[BCCLC06] Thierry P. Berger, Anne Canteaut, Pascale Charpin, and Yann Laigle-Chapuy. On Almost Perfect Nonlinear Functions Over $\mathbb{F}_2{}^n$. *IEEE Transactions on Information Theory*, 52(9) :4160–4170, 2006.

[BCL08] Lilya Budaghyan, Claude Carlet, and Gregor Leander. Two Classes of Quadratic APN Binomials Inequivalent to Power Functions. *IEEE Transactions on Information Theory*, 54(9) :4218–4229, 2008.

[BCQ04] Alex Biryukov, Christophe De Cannière, and Michaël Quisquater. On Multiple Linear Approximations. In Matthew K. Franklin, editor, *Advances in Cryptology - CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 1–22. Springer, 2004.

[BD93] Thomas Beth and Cunsheng Ding. On Almost Perfect Nonlinear Permutations. In *EUROCRYPT-93*, volume 765 of *Lecture Notes in Computer Science*, pages 65–76. Springer, 1993.

[BDK02] Eli Biham, Orr Dunkelman, and Nathan Keller. Enhancing Differential-Linear Cryptanalysis. In *ASIACRYPT '02*, volume 2501 of *Lecture Notes in Computer Science*, pages 254–266. Springer, 2002.

[BDK+10] Alex Biryukov, Orr Dunkelman, Nathan Keller, Dmitry Khovratovich, and Adi Shamir. Key Recovery Attacks of Practical Complexity on AES-256 Variants with up to 10 Rounds. In Henri Gilbert, editor, *Advances in Cryptology - EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 299–319. Springer, 2010.

[BDMW10] K. A. Browning, J. F. Dillon, M. T. McQuistan, and A. J. Wolfe. An APN permutation in dimension six. In *Finite Fields : theory and applications*, volume 518 of *Contemp. Math.*, pages 33–42. Amer. Math. Soc., 2010.

[BEA08] Behnam Bahrak, Taraneh Eghlidos, and Mohammad Reza Aref. Impossible Differential Cryptanalysis of Safer++. In Hamid R. Arabnia and Selim Aissi, editors, *Proceedings of the 2008 International Conference on Security & Management, SAM 2008*, pages 10–14. CSREA Press, 2008.

[BF00] Steve Babbage and Laurent Frisch. On MISTY1 Higher Order Differential Cryptanalysis. In Dongho Won, editor, *Information Security and Cryptology - ICISC 2000*, Lecture Notes in Computer Science, pages 22–36. Springer, 2000.

[BG09a] Céline Blondeau and Benoît Gérard. On the Data Complexity of Statistical Attacks Against Block Ciphers, 2009. *EUROCRYPT 2009 Poster-session.*

[BG09b] Céline Blondeau and Benoît Gérard. On the Data Complexity of Statistical Attacks Against Block Ciphers. In Alexander Kholosha, Eirik Rosnes, and Matthew G. Parker, editors, Workshop on Coding and Cryptography - WCC 2009*, pages 469–488, 2009.*

[BG10] Céline Blondeau and Benoît Gérard. Links Between Theoretical and Effective Differential Probabilities : Experiments on PRESENT. In TOOLS'10*, 2010.* `http://eprint.iacr.org/2010/261`.

[BG11] *Céline Blondeau and Benoît Gérard. Multiple Differential Cryptanalysis : Theory and Practice. In Antoine Joux, editor,* Fast Software Encryption, FSE 2011*, volume 6733 of* Lecture Notes in Computer Science*, pages 35–54. Springer, 2011.*

[BGT11] *Céline Blondeau, Benoît Gérard, and Jean-Pierre Tillich. Accurate estimates of the data complexity and success probability for various cryptanalyses.* Designs, Codes and Cryptography*, 59(1-3) :3–34, 2011.*

[Bih94] *Eli Biham. New types of cryptanalytic attacks using related keys. volume 7, pages 229–246, 1994.*

[Bir04] *Alex Biryukov. The Boomerang Attack on 5 and 6-Round Reduced AES. In Hans Dobbertin, Vincent Rijmen, and Aleksandra Sowa, editors,* Advanced Encryption Standard - AES, 4th International Conference, AES 2004, Bonn, Germany, May 10-12, 2004, Revised Selected and Invited Papers*, volume 3373 of* Lecture Notes in Computer Science*, pages 11–15. Springer, 2004.*

[BJV04] *Thomas Baignères, Pascal Junod, and Serge Vaudenay. How Far Can We Go Beyond Linear Cryptanalysis ? In* ASIACRYPT '04*, volume 3329 of* Lecture Notes in Computer Science*, pages 432–450. Springer, 2004.*

[BKL+07] *Andrey Bogdanov, Lars R. Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew J. B. Robshaw, Yannick Seurin, and C. Vikkelsoe. PRESENT : An Ultra-Lightweight Block Cipher. In Pascal Paillier and Ingrid Verbauwhede, editors,* Cryptographic Hardware and Embedded Systems - CHES 2007*, volume 4727 of* Lecture Notes in Computer Science*, pages 450–466. Springer, 2007.*

[BL08] *Marcus Brinkmann and Gregor Leander. On the classification of APN functions up to dimension five.* Des. Codes Cryptography*, 49(1-3) :273–288, 2008.*

[BL10] *Carl Bracken and Gregor Leander. A highly nonlinear differentially 4 uniform power mapping that permutes fields of even degree.* Finite Fields and Their Applications*, 16(4) :231–242, 2010.*

[BRS67] *E. R. Berlekamp, H. Rumsey, and G. Solomon. On the Solution of Algebraic Equations over Finite Fields.* Information and control*, 10 :553–564, 1967.*

[BS90] *Eli Biham and Adi Shamir. Differential Cryptanalysis of DES-like Cryptosystems. In Alfred Menezes and Scott A. Vanstone, editors,* CRYPTO 1990*, volume 537 of* Lecture Notes in Computer Science*, pages 2–21. Springer, 1990.*

[BS91] *Eli Biham and Adi Shamir. Differential Cryptanalysis of DES-like Cryptosystems.* Journal of Cryptology*, 4(1) :3–72, 1991.*

[BS93] *Eli Biham and Adi Shamir. Differential Cryptanalysis of the Full 16-round DES. In* CRYPTO'92*, volume 740 of* Lecture Notes in Computer Science*, pages 487–496. Springer, 1993.*

[BV08] *Thomas Baignères and Serge Vaudenay. The Complexity of Distinguishing Distributions. In* ICITS 2008*, volume 2008/5155 of* Lecture Notes in Computer Science*, pages p. 210–222, 2008.*

[Cam60] *Lucien Le Cam. An Approximation Theorem for the Poisson Binomial Distribution. In* Pacific Journal of Mathematics*, volume 10, pages 1181–1197, 1960.*

[Can06] *Anne Canteaut.* Analyse et conception de chiffrements à clef secrète. *Habilitation à diriger des recherches, Université Pierre et Marie Curie, Paris 6, september 2006.*

[Car69] *Leonard. Carlitz. Kloosterman sums and finite field extensions.* Acta Arithmetica*, 16 :179–183, 1969.*

[CCCF00] *Anne Canteaut, Claude Carlet, Pascale Charpin, and Caroline Fontaine. Propagation characteristics and correlation-immunity of highly nonlinear boolean functions. In* Advances in Cryptology - EUROCRYPT'2000*, volume 1807 of* Lecture Notes in Computer Science*, pages 507–522. Springer-Verlag, 2000.*

[CCCF01] *Anne Canteaut, Claude Carlet, Pascale Charpin, and Caroline Fontaine. On Cryptographic Properties of the Cosets of* $R(1, m)$*.* IEEE Trans. Inform. Theory*, 47(4) :1494–1513, May 2001.*

[CCD00] *Anne Canteaut, Pascale Charpin, and Hans Dobbertin. Binary m-sequences with three-valued crosscorrelation : A proof of Welch conjecture.* IEEE Transactions on Information Theory*, 46(1) :4–8, January 2000.*

[CCZ98] *Claude Carlet, Pascale Charpin, and Victor Zinoviev. Codes, Bent Functions and Permutations Suitable For DES-likeCryptosystems.* Designs, Codes and Cryptography*, 2(2) :125–156, November 1998.*

[CD96] *Thomas Cusick and Hans Dobbertin. Some new 3-valued crosscorrelation functions of binary m-sequences.* IEEE Trans. Inform. Theory*, 42(4) :1238–1240, 1996.*

[CDDL06] *Anne Canteaut, Magnus Daum, Hans Dobbertin, and Gregor Leander. Finding nonnormal bent functions.* Discrete Applied Mathematics*, 154(2) :202–218, 2006.*

[Cha04] *Pascale Charpin. Normal Boolean functions.* Journal of Complexity*, 20(2-3) :245–265, 2004.*

[CHN08] *Joo Yeon Cho, Miia Hermelin, and Kaisa Nyberg. A new technique for multidimensional linear cryptanalysis with applications on reduced round serpent. In Pil Joong Lee and Jung Hee Cheon, editors,* Information Security and Cryptology - ICISC 2008*, volume 5461 of* Lecture Notes in Computer Science*, pages 383–398. Springer, 2008.*

[Cho10] *Joo Yeon Cho. Linear Cryptanalysis of Reduced-Round PRESENT. In Josef Pieprzyk, editor,* Topics in Cryptology - CT-RSA 2010*, volume 5985 of* Lecture Notes in Computer Science*, pages 302–317. Springer, 2010.*

[CHZ07] *Pascale Charpin, Tor Helleseth, and Victor Zinoviev. Propagation characteristics of* x^{-1} *and Kloosterman sums.* Finite Fields and Their Applications*, 13(2) :366–381, 2007.*

[CKK+01] *Jung Hee Cheon, MunJu Kim, Kwangjo Kim, Jung-Yeun Lee, and SungWoo Kang. Improved Impossible Differential Cryptanalysis of Rijndael and Crypton. In* ICISC*, volume 2288 of* Lecture Notes in Computer Science*, pages 39–49. Springer, 2001.*

[Cro05] *Paul Crowley. Truncated Differential Cryptanalysis of Five Rounds. Technical Report 2005/073, eSTREAM, http ://www.ecrypt.eu.org/stream/papers.html, 2005.*

[CS09] *Baudoin Collard and François-Xavier Standaert. A Statistical Saturation Attack against the Block Cipher PRESENT. In* CT-RSA-2009*, volume 5473/2009 of* Lecture Notes In Computer Science*, pages 195–210. Springer, 2009.*

[CT91] *Thomas M. Cover and Joy A. Thomas.* Information theory*. Wiley series in communications. Wiley, 1991.*

[CV95] *Florent Chabaud and Serge Vaudenay. Links between differential and linear cryptanalysis. In* Advances in Cryptology - EUROCRYPT'94*, volume 950 of* Lecture Notes in Computer Science*, pages 356–365. Springer-Verlag, 1995.*

[CV02] *Anne Canteaut and Marion Videau. Degree of Composition of Highly Nonlinear Functions and Applications to Higher Order Differential Cryptanalysis. In Lars R. Knudsen, editor,* Advances in Cryptology - EUROCRYPT 2002,*, volume 2332 of* Lecture Notes in Computer Science*, pages 518–533. Springer, 2002.*

[DES77] *FIPS PUB 46-3 national bureau of standard DES. Data Encryption Standard(DES), 1977.*

[DK07] *Orr Dunkelman and Nathan Keller. A New Criterion for Nonlinearity of Block Ciphers .* IEEE transaction on information theory*, 53(11) :3944–3957, November 2007.*

[DKR97] *Joan Daemen, Lars R. Knudsen, and Vincent Rijmen. The Block Cipher Square. In Eli Biham, editor,* Fast Software Encryption, FSE'97*, volume 1267 of* Lecture Notes in Computer Science*, pages 149–165. Springer, 1997.*

[DN03] *Herbert A. David and H. N. Nagaraja.* Order Statistics (third edition)*. Wiley series in Probability Theory. SV, 2003.*

[Dob98] *Hans Dobbertin. One-to-one highly nonlinear power functions on* $GF(2^n)$*.* Applicable Algebra in Engineering, Communication and Computing*, 9 :139–152, 1998.*

[Dob99a] *Hans Dobbertin. Almost Perfect Nonlinear power functions on* $GF(2^n)$ *: the Niho case.* Information and Computation*, 151(1-2) :57–72, 1999.*

[Dob99b] *Hans Dobbertin. Almost Perfect Nonlinear power functions on* $GF(2^n)$ *: the Welch case.* IEEE Transactions on Information Theory*, 45(4) :1271–1275, 1999.*

[Dob00] *Hans Dobbertin. Almost Perfect Nonlinear Power Functions on* $GF(2^n)$ *: a new class for n divisible by 5. In* Proceedings of Finite Fields and Applications Fq5*, pages 113–121, Augsburg, Germany, 2000. Springer-Verlag.*

[DR99] *Joan Daemen and Vincent Rijmen. AES proposal : the Rijndael block cipher, 1999.*

[DR05] *Joan Daemen and Vincent Rijmen. Probability distributions of Correlation and Differentials in Block Ciphers. Cryptology ePrint Archive, Report 2005/212, 2005.* `http://eprint.iacr.org/`*.*

[Gal68] *Robert G. Gallager.* Information Theory and Reliable Communication. *John Wiley and Sons, 1968.*

[GG99a] *Sololom W. Golomb and Guang Gong. Periodic binary sequences with the trinomial property.* IEEE Trans. Inform. Theory, *45(4) :1276–1279, 1999.*

[GG99b] *Guang Gong and Solomon W. Golomb. Transform domain analysis of DES.* IEEE Trans. Inform. Theory, *45(6) :2065–2073, 1999.*

[Gil97] *Henri Gilbert.* Cryptanalyse statistique des algorithmes de chiffrement et sécurité des schémas d'authentification. *Thèse de doctorat, Université Paris 11 Orsay, 1997.*

[Gol68] *Solomon W. Golomb. Theory of transformation groups of polynomials over $GF(2)$ with applications to linear shift register sequences.* Information Sciences, *1 :87–109, 1968.*

[HCN09] *Miia Hermelin, Joo Yeon Cho, and Kaisa Nyberg. Multidimensional Extension of Matsui's Algorithm 2. In Orr Dunkelman, editor,* Fast Software Encryption, FSE 2009, *volume 5665 of* Lecture Notes in Computer Science, *pages 209–227. Springer, 2009.*

[Her05] *D. Hertel. A Note on the Kasami Power Function. Technical Report 436, eprint, 2005.*

[HN10] *Miia Hermelin and Kaisa Nyberg. Dependent Linear Approximations : The Algorithm of Biryukov and Others Revisited. In Josef Pieprzyk, editor,* Topics in Cryptology - CT-RSA 2010, The Cryptographers' Track at the RSA Conference 2010, *volume 5985 of* Lecture Notes in Computer Science, *pages 318–333. Springer, 2010.*

[HP08] *Doreen Hertel and Alexander Pott. Two results on maximum nonlinear functions.* Des. Codes Cryptography, *47(1-3) :225–235, 2008.*

[HX01] *Henk D.L. Hollmann and Qing Xiang. A proof of the Welch and Niho conjectures on crosscorrelations of binary m-sequences.* Finite Fields and their Applications, *7(2) :253–286, 2001.*

[JP07] *Antoine Joux and Thomas Peyrin. Hash Functions and the (Amplified) Boomerang Attack. In Alfred Menezes, editor,* Advances in Cryptology - CRYPTO 2007, *volume 4622 of* Lecture Notes in Computer Science, *pages 244–263. Springer, 2007.*

[JR94] *Burton S. Kaliski Jr. and Matthew J. B. Robshaw. Linear cryptanalysis using multiple approximations. In Yvo Desmedt, editor,* Advances in Cryptology - CRYPTO-1994, *volume 839 of* Lecture Notes in Computer Science, *pages 26–39. Springer, 1994.*

[Jun01] *Pascal Junod. On the Complexity of Matsui's Attack. In* SAC '01, *volume 2259 of* Lecture Notes in Computer Science, *pages 199–211. Springer, 2001.*

[Jun03] *Pascal Junod. On the Optimality of Linear, Differential, and Sequential Distinguishers. In* EUROCRYPT '03, *volume 2656 of* Lecture Notes in Computer Science, *pages 17–32. Springer, 2003.*

[JV03] *Pascal Junod and Serge Vaudenay. Optimal Key Ranking Procedures in a Statistical Cryptanalysis. In Thomas Johansson, editor,* Fast Software Encryption, FSE 2003, *volume 2887 of* Lecture Notes in Computer Science, *pages 235–246. Springer, 2003.*

[JW93] *Heeralal Janwa and Richard M. Wilson. Hyperplane Sections of Fermat Varieties in P^3 in Char.2 and Some Applications to Cyclic Codes. In Gérard D. Cohen, Teo Mora, and Oscar Moreno, editors,* Applied Algebra, Algebraic Algorithms and Error-Correcting Codes, AAECC-10*, volume 673 of* Lecture Notes in Computer Science*, pages 180–194. Springer, 1993.*

[Kas71] *Tadao Kasami. The weight enumerators for several classes of subcodes of the second order binary Reed-Muller codes.* Information and Control*, 18 :369–394, 1971.*

[KB96] *Lars Knudsen and Thomas Berson. Truncated Differentials of SAFER. In* Fast Software Encryption*, volume 1039/1996 of* Lecture Notes in Computer Science*, pages 15–26. Springer Berlin / Heidelberg, 1996.*

[KHCH96] *P.V. Kumar, T. Helleseth, A.R. Calderbank, and A.R. Hammons. Large families of quaternary sequences with low correlation.* IEEE-IT*, IT-42(2) :579–592, 1996.*

[Knu95] *Lars Knudsen. Truncated and Higher Order Differentials. In* Fast Software Encryption, FSE'94*, volume 1008/1995 of* Lecture Notes in Computer Science*, pages 196–211. Springer Berlin / Heidelberg, 1995.*

[KR07] *Lars .R. Knudsen and Vincent Rijmen. Known-Key Distinguishers for Some Block Ciphers. In* Advances in cryptology - ASIACRYPT 2007*, volume 4833 of* Lecture Notes in Computer Science*, pages 315–324. Springer, 2007.*

[KRW99] *Lars R. Knudsen, Matt Robshaw, and David Wagner. Truncated Differentials and Skipjack. In* CRYPTO 99*, volume 1666/1999 of* Lecture Notes in Computer Science*, pages 165–180. Springer-Verlag, 1999.*

[KW02] *Lars R. Knudsen and David. Wagner. Integral cryptanalysis. In* Fast Software Encryption - FSE 2002*, volume 2365 of* Lecture Notes in Computer Science*, pages 112–127. Springer-Verlag, 2002.*

[Lai94] *Xuejia Lai. Higher order derivatives and differential cryptanalysis. In* Symposium on communication, Coding and cryptography*, in honor of J. L. Massey on the occasion of his 60'th birthday, 1994.*

[Lea10] *Gregor Leander. Small scale variants of the block cipher PRESENT. Cryptology ePrint Archive, Report 2010/143, 2010.* `http://eprint.iacr.org/2010/143`.

[Lea11] *Gregor Leander. On Linear Hulls, Statistical Saturation Attacks, PRESENT and a Cryptanalysis of PUFFIN. In Kenneth G. Paterson, editor,* Advances in Cryptology - EUROCRYPT 2011*, volume 6632 of* Lecture Notes in Computer Science*, pages 303–322. Springer, 2011.*

[LH94] *Susan K. Langford and Martin E. Hellman. Differential-linear cryptanalysis. In Yvo Desmedt, editor,* Advances in Cryptology - CRYPTO '94*, volume 839 of* Lecture Notes in Computer Science*, pages 17–25. Springer, 1994.*

[LHL+02] *S. Lee, S. Hong, S. Lee, J. Lim, and S. Yoon. Truncated differential cryptanalysis of camellia. In* Information Security and Cryptology — ICISC 2001*, volume 2288 of* Lecture Notes in Computer Science*, pages 287–342, 2002.*

[LM91] *Xuejia Lai and James L. Massey. Markov Ciphers and Differentail Cryptanalysis. In* EUROCRYPT-91*, Lecture Notes in Computer Science, pages 17–38. Springer-Verlag, 1991.*

[LP07] *Gregor Leander and Alexander Poschmann. On the Classification of 4 Bit S-Boxes. In* Proceedings of the 1st international workshop on Arithmetic of Finite Fields*, volume 4547/2007 of* Lecture Notes In Computer Science*, pages 159–176. Springer, 2007.*

[Luc01] *Stefan Lucks. The Saturation Attack - A Bait for Twofish. In Mitsuru Matsui, editor,* Fast Software Encryption, FSE-2001*, volume 2355 of* Lecture Notes in Computer Science*, pages 1–15. Springer, 2001.*

[Mat93] *Mitsuru Matsui. Linear cryptanalysis method for DES cipher. In* EUROCRYPT '93*, volume 765 of* Lecture Notes in Computer Science*, pages 386–397. Springer, 1993.*

[Mat94] *Mitsuru Matsui. The First Experimental Cryptanalysis of the Data Encryption Standard. In* CRYPTO '94*, volume 839 of* Lecture Notes in Computer Science*, pages 1–11. Springer, 1994.*

[Mat97] *Mitsuru Matsui. New Block Encryption Algorithm MISTY. In* Fast Software Encryption - FSE 1997*, volume 1267 of* Lecture Notes in Computer Science*, pages 54–68. Springer Verlag, 1997.*

[McE87] *Robert J. McEliece.* Finite Fields for Computer Scientists and Engineers*. Kluwer, Boston, 1987.*

[MG00] *Marine Minier and Henri Gilbert. Stochastic Cryptanalysis of Crypton. In Bruce Schneier, editor,* Fast Software Encryption, FSE 2000*, volume 1978 of* Lecture Notes in Computer Science*, pages 121–133. Springer, 2000.*

[Min02] *Marine Minier.* Preuves d'Analyse et de Sécurité en Cryptologie à clé Secrète*. Thèse de doctorat, Université de Limoges, 2002.*

[MSAK99] *Shiho Moriai, Makoto Sugita, Kazumaro Aoki, and Masayuki Kanda. Security of E2 against Truncated Differential Cryptanalysis. In Howard M. Heys and Carlisle M. Adams, editors,* Selected Areas in Cryptography, SAC'99*, volume 1758 of* Lecture Notes in Computer Science*, pages 106–117. Springer, 1999.*

[MSD10] *Hamid Mala, Mohsen Shakiba, and Mohammad Dakhilalian. New impossible differential attacks on reduced-round Crypton.* Computer Standards & Interfaces*, 32(4) :222–227, 2010.*

[MT99] *Mitsuru Matsui and Toshio Tokita. Cryptanalysis of a reduced version of the block cipher e2. In Lars R. Knudsen, editor,* Fast Software Encryption, FSE '99,*, volume 1636 of* Lecture Notes in Computer Science*, pages 71–80. Springer, 1999.*

[NGG06] *Yassir Nawaz, Guang Gong, and Kishan Chand Gupta. Upper bounds on algebraic immunity of boolean power functions. In Matthew J. B. Robshaw, editor,* Fast Software Encryption, FSE 2006*, volume 4047 of* Lecture Notes in Computer Science*, pages 375–389. Springer, 2006.*

[NGG09] *Yassir Nawaz, Kishan Chand Gupta, and Guang Gong. Algebraic immunity of S-boxes based on power mappings : analysis and construction.* IEEE Transactions on Information Theory*, 55(9) :4263–4273, 2009.*

[NK92] *Kaisa Nyberg and Lars R. Knudsen. Provable security against differential cryptanalysis. In Ernest F. Brickell, editor,* Advances in Cryptology

- CRYPTO '92, *volume 740 of* Lecture Notes in Computer Science, *pages 566–574. Springer, 1992.*

[NSZW09] *Jorge Nakahara, Pouyan Sepehrdad, Bingsheng Zhang, and Meiqin Wang. Linear (Hull) and Algebraic Cryptanalysis of the Block Cipher PRESENT. In Juan A. Garay, Atsuko Miyaji, and Akira Otsuka, editors,* Cryptology and Network Security, CANS 2009, Proceedings, *volume 5888 of* Lecture Notes in Computer Science, *pages 58–75. Springer, 2009.*

[Nyb91] *Kaisa Nyberg. Perfect nonlinear S-boxes. In* Advances in Cryptology — EUROCRYPT '91, *volume 547/1991 of* Book Series Lecture Notes in Computer Science, *pages 378–386. Springer Berlin / Heidelberg, 1991.*

[Nyb94] *Kaisa Nyberg. Differentially uniform mappings for cryptography. In* Eurocrypt-93, *volume 765 of* Lecture Notes in Computer Science, *pages 55–64. Springer-Verlag, 1994.*

[Nyb96] *Kaisa Nyberg. Generalized Feistel Networks. In* ASIACRYPT '96, *volume 1163 of* Lecture Notes in Computer Science, *pages 91–104. Springer, 1996.*

[Ohk09] *Kenji Ohkuma. Weak Keys of Reduced-Round PRESENT for Linear Cryptanalysis. In Michael J. Jacobson Jr., Vincent Rijmen, and Reihaneh Safavi-Naini, editors,* Selected Areas in Cryptography, 2009, Revised Selected Papers, *volume 5867 of* Lecture Notes in Computer Science, *pages 249–265. Springer, 2009.*

[ÖVTcK09] *Onur Özen, Kerem Varici, Cihangir Tezcan, and Çelebi Kocair. Lightweight Block Ciphers Revisited : Cryptanalysis of Reduced Round PRESENT and HIGHT. In Colin Boyd and Juan Manuel González Nieto, editors,* Information Security and Privacy, 2009, Proceedings, *volume 5594 of* Lecture Notes in Computer Science, *pages 90–107. Springer, 2009.*

[PSLL03] *Sangwoo Park, Soo Hak Sung, Sangjin Lee, and Jongin Lim. Improving the Upper Bound on the Maximum Differential and the Maximum Linear Hull Probability for SPN Structures and AES. In Thomas Johansson, editor,* Fast Software Encryption,FSE 2003, *volume 2887 of* Lecture Notes in Computer Science, *pages 247–260. Springer, 2003.*

[Sel08] *Ali Aydin Selçuk. On Probability of Success in Linear and Differential Cryptanalysis.* Journal of Cryptology, *21(1) :131–147, 2008.*

[SKU+00] *Makoto Sugita, Kazukuni Kobara, Kazuhiro Uehara, Shuji Kubota, and Hideki Imai. Relationships among Differential, Truncated Differential, Impossible Differential Cryptanalyses against Word-Oriented Block Ciphers like RIJNDAEL, E2. In* AES Candidate Conference, *pages 242–254, 2000.*

[Tez10] *Cihangir Tezcan. The Improbable Differential Attack : Cryptanalysis of Reduced Round CLEFIA. In Guang Gong and Kishan Chand Gupta, editors,* Progress in Cryptology - INDOCRYPT 2010, *volume 6498 of* Lecture Notes in Computer Science, *pages 197–209. Springer, 2010.*

[THK99] *Hidema Tanaka, Kazuyuki Hisamatsu, and Toshinobu Kaneko. Strenght of MISTY1 without FL Function for Higher Order Differential Attack. In Marc P. C. Fossorier, Hideki Imai, Shu Lin, and Alain Poli, editors,* Applied Algebra, Algebraic Algorithms and Error-Correcting Codes,AAECC-13, *volume 1719 of* Lecture Notes in Computer Science, *pages 221–230. Springer, 1999.*

[Vau03] *Serge Vaudenay. Decorrelation : A Theory for Block Cipher Security.* Journal of Cryptology*, 16(4) :249–286, 2003.*

[Vid05] *Marion Videau.* Critères de Sécurité des algorithmes de Chiffrement à Clé Secrète*. Thèse de doctorat, Université Paris 6, 2005.*

[Wag99] *David Wagner. The Boomerang Attack. In Lars R. Knudsen, editor,* Fast Software Encryption, 6th International Workshop, FSE '99*, volume 1636 of* Lecture Notes in Computer Science*, pages 156–170. Springer, 1999.*

[Wan08] *Meiqin Wang. Differential Cryptanalysis of Reduced-Round PRESENT. In Serge Vaudenay, editor,* Progress in Cryptology - AFRICACRYPT 2008*, volume 5023 of* Lecture Notes in Computer Science*, pages 40–49. Springer, 2008.*

[ZRHD08] *Muhammad Reza Z'aba, Håvard Raddum, Matthew Henricksen, and Ed Dawson. Bit-pattern based integral attack. In Kaisa Nyberg, editor,* FSE-2008*, volume 5086 of* Lecture Notes in Computer Science*, pages 363–381. Springer, 2008.*

[ZZ95] *Yuliang Zheng and Xian-Mo Zhang. The criterion for global avalanche characterics of cryptographic functions.* Journal of Universal Computer Science*, 1(5) :320–337, 1995.*

[ZZ99] *Yuliang Zheng and Xian-Mo Zhang. Plateaued functions.* Information and Communication Security, ICICS'99*, 1726 :224–300, 1999.*

Table des figures

Liste des Algorithmes

Liste des tableaux

Table des matières