UNIVERSITÉ PIERRE ET MARIE CURIE - PARIS 6

ÉCOLE DOCTORALE EDITE

T H È S E

CIFRE en partenariat avec THALES Communications & Security

pour obtenir le titre de

Docteur en Sciences

de l'Université Pierre et Marie Curie - Paris 6
Mention : Informatique

Présentée et soutenue par

Christopher Goyet

Cryptanalyse algébrique par canaux auxiliaires

Thèse dirigée par Jean-Charles Faugère
et Guénaël Renault

préparée au Laboratoire d'Informatique de Paris 6 (UPMC)
et au Laboratoire Chiffre de THALES (LCH)

soutenue le 7 Novembre 2012

Jury :

Rapporteurs :	Aline Gouget	-	Gemalto
	Francois-Xavier Standaert	-	Université catholique de Louvain
Directeurs :	Jean-Charles Faugère	-	INRIA
	Guénaël Renault	-	UPMC - Université Paris 6
Encadrant industriel :	Olivier Orcière	-	Thales Communications & Security
Examinateurs :	Jean-Claude Bajard	-	UPMC - Université Paris 6
	Claude Carlet	-	Université Paris 8

Laboratoire Chiffre de THALES Communications

À Thierry

Remerciements

Tout d'abord, je remercie Jean-Charles Faugère et Guénaël Renault qui m'ont dirigé et guidé pendant ces trois années de thèse. Ils ont toujours été disponibles et très attentifs à l'avancée de mes travaux. J'ai toujours considéré avec beaucoup d'attention leurs précieux conseils et leurs remarques fructueuses pendant les nombreuses discussions que nous avons eues. Je les remercie aussi sincèrement pour le temps qu'ils ont consacré aux relectures de ce manuscrit, mais surtout pour les nombreuses et fastidieuses répétitions dont j'ai tant besoin avant toute présentation. Je dois ainsi reconnaitre que j'ai énormément appris et évolué grâce à leur encadrement exigeant, tout en bénéficiant toujours d'une grande liberté. Je tiens aussi à dire que la bonne humeur de Guénaël et ses plaisanteries, d'ailleurs souvent douteuses, m'ont beaucoup aidé à retrouver le sourire dans les moments difficiles que j'ai traversés.

Je voudrais aussi remercier très chaleureusement Olivier Orcière pour ses encouragements, sa sincérité et sa disponibilité. Il m'a été d'un soutien inestimable, en trouvant toujours les mots justes pour me redonner la force d'avancer. Je n'oublierai jamais nos discussions enflammées, en particulier concernant mes difficultés avec l'anglais et je tiens donc à le remercier pour sa grande patience. En plus d'avoir bénéficié de son érudition, j'ai aussi énormément profité de ses immenses connaissances scientifiques.

J'adresse aussi tous mes remerciements à Aline Gouget et à François-Xavier Standaert qui me font l'honneur de rapporter cette thèse. Je dois à François-Xavier Standaert une reconnaissance toute particulière pour les différentes réunions très enrichissantes que j'ai pu avoir avec lui. Je tenais à le remercier très sincèrement pour le temps qu'il m'a consacré et pour les nombreux déplacements qu'il a dû faire. Ses encouragements m'ont beaucoup motivé et sa présence lors de plusieurs de mes présentations (je pense en particulier à COSADE2011) m'a énormément rassuré. J'en profite également pour le remercier d'avoir aussi rapporté mon avancement à mi-parcours.

Mes remerciements vont aussi à Claude Carlet pour l'intérêt qu'il a porté

à mon travail. J'ai beaucoup apprécié nos échanges et j'ai été particulièrement touché par sa gentillesse et sa bienveillance. Je le remercie sincèrement d'avoir accepté d'être président du jury de thèse.

Je remercie aussi Jean-Claude Bajard d'avoir accepté de faire partie de mon jury. J'apprécie tout spécialement sa bonne humeur et son humour.

Je remercie tous mes collègues du laboratoire Chiffre de Thales : Alexandre, Aurore, David, Émeline, Eric, Lydie, Matthieu, Olivier, Pascal, Philippe, Renaud, Sandra, Sonia, Sylvain. Toujours de bons conseils, ils m'ont tous beaucoup aidé à relativiser, en me remontant le moral et en me faisant partager leur expérience. Je remercie vivement Eric pour sa bienveillance et je tenais à dire que j'avais infiniment apprécié son soutien constant pendant ces 3 années souvent difficiles.

J'ai une pensée particulière pour Ange avec qui j'ai partagé le bureau lors de mes premiers mois à Thales. Il a été d'un grand soutien, toujours d'une immense gentillesse et un véritable repère pour moi.

Je suis aussi très heureux d'avoir partagé de grands moments avec Aurore. Je pense en particulier au week-end inoubliable qu'on a passé ensemble à naviguer sur le voilier du CE de Thales en Bretagne. Je la remercie aussi pour sa patience, sa compassion et sa bonne humeur pendant toutes ces années. Bien sûr, je n'oublierai pas non plus les bons moments passés avec tous les stagiaires (Romain, Thomas, Gaëtan, Marine, et à l'époque Frédéric, Matthieu et Sonia) que je suis toujours ravi de revoir. Je garderai aussi un très bon souvenir de Lydie, avec qui j'ai adoré discuter. Je tenais à la remercier pour tout ce qu'elle avait fait pour moi, mais aussi pour sa bonne humeur et ses encouragements qui m'ont beaucoup touché.

Évidemment, je remercie tous les membres de l'équipe POLSYS du LIP6 (en particulier Ludovic, Mohab et Elias) pour la bonne ambiance que vous apportez. Je tiens à remercier tout spécialement Mohab pour ses sages conseils après m'avoir écouté dans un moment où j'en avais particulièrement besoin. Je remercie aussi Martin d'avoir accepté de me remplacer à la conférence SAC2012. Je n'oublie pas Stef, avec qui il est toujours intéressant et agréable de discuter grâce à son humour et son intelligence remarquables.

Tous les thésards (Aurélien, Chenqi, Frédéric, Jules, Louise, Luk, Mourad, Pierre-Jean, Rina et Thibaut) avec qui j'ai partagé le bureau à l'UPMC savent à quel point leur présence, leur soutien et leur amitié ont été importants pour moi. Je pense que cette période de thèse, très particulière dans une vie, et qu'on a partagée tous ensemble, a créé des liens singuliers entre nous. Pour tout cela, je ne pourrai jamais assez les remercier et je leur souhaite de tout coeur une grande réussite.

Je remercie enfin mes parents et mes frères qui m'ont toujours soutenu et

encouragé. Malgré la distance, j'ai toujours ressenti intensément leur amour, ce qui m'a toujours permis d'avancer. Je pense aussi à tous mes amis du Sud (Julien, Florian, Robin, Laurent, Marc, Guillaume, Salomon, Smaranda, Kerwan, Micka, Steven) qui m'ont toujours soutenu et qui me manquent terriblement. Je remercie aussi tous mes amis de Paris (Christopher, Claire, Aurélia, Christophe, Jérome, Richard, Perceval, Julien) pour tous les bons moments passés ensemble et pour m'avoir changé les idées quand j'en avais besoin.

Enfin, je dédie ce manuscrit à Thierry, car c'est sans aucun doute la personne qui m'a le plus apporté ces 3 dernières années.

Table des matières

Remerciements **iii**

Introduction **1**

I Préliminaires **11**

1 Rappels sur la cryptographie **13**
1.1 Introduction 13
1.2 Rappels sur les corps finis 13
1.3 Cryptosystèmes symétriques 15
1.3.1 AES 15
1.3.2 PRESENT 20
1.4 Cryptosystèmes asymétriques 22
1.4.1 DSA 23
1.4.2 ECDSA 24

2 Résolution de systèmes polynomiaux **25**
2.1 Introduction 25
2.2 Résolution par bases de Gröbner 27
2.2.1 Idéaux et variétés 27
2.2.2 Bases de Gröbner 31
2.2.3 Algorithmes de calcul de base de Gröbner 34
2.3 Résolution par SAT solveurs 39
2.3.1 Satisfiabilité d'une forme normale conjonctive (CNF) . 39
2.3.2 Conversion d'un système polynomial en CNF 41
2.3.3 L'algorithme DPLL 44

3 Réduction de réseaux euclidiens **47**
3.1 Les réseaux euclidiens 47
3.2 Minimums d'un réseau euclidien et SVP 50
3.3 Réduction de réseaux 52
3.4 Algorithme LLL 53
3.5 Heuristique gaussienne 55

4 Attaques par canaux auxiliaires **57**
4.1 Introduction 57
4.2 Attaques passives 58
4.2.1 Analyse simple de consommation (SPA) 59
4.2.2 Analyse différentielle de consommation (DPA) 60
4.2.3 Analyse de consommation par corrélation (CPA) . . . 62
4.2.4 Attaques profilées (templates) 63
4.3 Attaques actives 64

II Contributions **67**

5 Attacking (EC)DSA Given Only an Implicit Hint **69**
5.1 Introduction 69
5.2 Possible application scenario 74
5.3 Embedding into a Lattice Problem 75
5.3.1 Shared MSB and LSB 76
5.3.2 Proposal for improvements 79
5.3.3 Blocks of shared bits 83
5.4 Experimental results 88
5.4.1 Shared MSB and LSB 88
5.4.2 Blocks of shared bits 91
5.4.3 Random number generator tests 93
5.5 Further developments 94

6 Analysis of the Algebraic Side Channel Attack **97**
6.1 Introduction 97
6.2 Algebraic Cryptanalysis and side channel information 102
6.2.1 Algebraic Immunity for Block ciphers 103
6.2.2 Algebraic Immunity of S-boxes with Leakage 105
6.3 The Hamming weight leakage model 106
6.4 Solving the complete system modeling the block cipher 110
6.4.1 Solving Strategy 111

6.4.2 A Sufficient Condition of Success 111
6.4.3 Consecutive leakages 114
6.5 Characterization of a family of resistant S-Boxes 115
6.6 Experiments 118
6.6.1 Experiments against PRESENT using the Hamming weight model 118
6.6.2 Experiments against AES using the Hamming weight model 122
6.6.3 A study of several S-Boxes and 8-bit Hamming weight leakage model 125
6.7 Conclusion 130

7 Les attaques ASCA étendues à d'autres modèles de fuites 131
7.1 Introduction 131
7.2 La distance de Hamming 132
7.2.1 La distance de Hamming entre l'entrée et la sortie des boîtes-S 132
7.2.2 La distance de Hamming entre chiffrements successifs . 136
7.3 Un modèle pour le poids de Hamming avec erreurs ? 140
7.4 Attaques par faute avec méthodes algébriques 145
7.5 Les "attaques par collision" 148
7.6 Conclusion 150

Bibliographie 166

Résumé 168

Introduction

> *"In the Information Age, cryptography is about political power [...]. It is about the right to privacy, freedom of speech, freedom of political association, freedom of the press, freedom from unreasonable search and seizure, freedom to be left alone." Phil Zimmermann, fondateur de PGP.*

L'utilisation de la cryptographie est un droit pleinement reconnu depuis 2004 par la loi française [1]. Cette liberté, finalement assez récente, est une garantie à la liberté de communication et du respect de la vie privée auquel nous sommes tous très attachés. Pour autant, nous ne ressentons pas la nécessité de chiffrer toutes nos lettres et données personnelles afin de les protéger d'éventuels regards indiscrets. L'évocation de la cryptographie provoque même souvent, par méconnaissance de son importance, une réaction d'indifférence sous prétexte qu'un "honnête citoyen n'a rien à cacher". En réalité, depuis que le monde est rentré dans l'âge de l'information, nous avons tous quotidiennement recours à la cryptographie, la plupart du temps probablement sans en être conscient. Elle est devenue indispensable par exemple pour effectuer des transactions financières, notamment lorsqu'on utilise sa carte bancaire, mais aussi pour garantir la sécurité de la plupart des cartes à puces, des passeports numériques, du vote électronique, des services sur internet, des produits télévisuels, des consoles de jeux, des téléphones portables, etc.

Essentielle aux systèmes d'informations actuels, la cryptographie procure des moyens de protection des données par des algorithmes cryptographiques assurant la confidentialité mais aussi l'authentification et l'intégrité des données. Les algorithmes cryptographiques sont des fonctions mathématiques très difficiles à inverser sans une information particulière, tenue secrète, appelée clef . La sécurité repose donc, en pratique, sur la difficulté à retrouver

1. Article 30 de la loi 2004-575 du 21 juin 2004.

la clef secrète à partir d'informations publiques. C'est l'objectif de la cryptanalyse d'estimer cette difficulté, et ainsi d'évaluer la sécurité des algorithmes cryptographiques. Dans ce but, de nombreuses méthodes mathématiques ont été proposées, les plus classiques étant la cryptanalyse linéaire [Mat93] et la cryptanalyse différentielle [BS91].

Le sujet de cette thèse se rattache à la cryptanalyse algébrique qui consiste à modéliser judicieusement une primitive cryptographique sous la forme d'un système d'équations algébriques. Ce système d'équations est construit de manière à établir une correspondance entre les solutions de ce système et les informations secrètes du cryptosystème en question. La sécurité du cryptosystème repose alors sur la complexité de résolution d'un tel système, complexité qu'il s'agit d'estimer de manière théorique, et pratique, avec des outils fournis par les dernières avancées du calcul formel. Beaucoup de méthodes puissantes de résolution de systèmes polynomiaux ont déjà été proposées : bases de Gröbner (Buchberger, FGLM, F4, F5 [Buc65, CLO07, FGLM93, Fau99, Fau02]), SAT-solver ([ES03, SNC09]), les méthodes de type XSL ([CL05, AFI$^+$04]), par résultants ([EM07]), etc. Ces techniques de résolution ont été particulièrement efficaces contre un grand nombre de schémas multivariés et contre quelques chiffrements par flot ([Ars05, CP02, CM03, FJ03, FP06b, FP06a, FdVP08]), mais elles restent impraticables contre les chiffrements par blocs ([MR02, Bar04, CL05, Ars05]). En effet, la taille des systèmes d'équations polynomiales correspondant est tellement importante (des milliers de variables et d'équations de hauts degrés) qu'on n'est pas encore capable de prédire correctement la complexité de résolution de tels systèmes polynomiaux. D'un autre côté, les cryptosystèmes asymétriques, comme RSA ([RSA78]) ou (EC)DSA ([FIP09]) par exemple, se modélisent par des systèmes d'équations très petits (souvent pas plus d'une seule équation) mais pour lesquels la recherche de solutions entières non triviales pose aussi beaucoup de difficultés. D'autres méthodes de résolution plus adaptées sont utilisées dans ces cas-là, par exemple des méthodes à base de réduction de réseaux (LLL [LLL82]) ou bien les méthodes de Coppersmith ([Cop96b, Cop96a]) pour la recherche de petites solutions entières de systèmes polynomiaux.

Ainsi, la cryptanalyse algébrique, comme toute la cryptanalyse en générale, étudie la résistance mathématique des algorithmes cryptographiques. Elle ne porte donc que sur une partie d'un système cryptographique. Or, la sécurité fournie par un appareil implémentant un algorithme cryptographique ne dépend pas uniquement de la robustesse mathématique de cet algorithme, mais bien de la sécurité de l'ensemble du système. Comme disait P. Kocher en 1999 : “A correct implementation of a strong protocol is not

necessarily secure" ([KJJ99]). En effet, en cryptanalyse on suppose implicitement que les informations disponibles pour un attaquant ne sont limitées qu'à certaines données publiques, en général seulement le chiffré, le message clair et en asymétrique, la clef publique. Or, en pratique, d'autres informations peuvent être récupérées par un attaquant à travers d'autres parties moins sécurisées du système, c'est-à-dire à travers ce qu'on appelle un canal auxiliaire. Par exemple, si la clef privée est sauvegardée dans une zone mémoire mal protégée, il peut être plus facile d'attaquer cette zone que d'attaquer mathématiquement l'algorithme. Les fuites physiques sont un autre exemple de canal auxiliaire possible. En effet, des mesures physiques permettent d'obtenir des informations sur les états internes d'un composant lors du chiffrement ou du déchiffrement. Des méthodes d'analyse, généralement statistiques, de ces mesures peuvent permettre de retrouver des informations sur la clef secrète. On peut citer, par exemple, la mesure avec un oscilloscope de la consommation électrique, qui est à la base de nombreuses attaques particulièrement efficaces ([MOP07, ABDM00, BCO04, KJJ99]). Ou bien encore, celles exploitant les mesures du champ magnétique, du temps de calcul, etc.

L'objectif de cette thèse est d'évaluer comment une information extérieure peut permettre d'accélérer significativement les méthodes de cryptanalyse algébrique. On suppose que cette information extérieure est obtenue par canal auxiliaire, suite à des mesures physiques ou bien suite à un comportement anormal provoqué par des attaques actives du type injection de fautes ou infection par un logiciel malveillant. On se focalise donc sur la partie de cryptanalyse algébrique en ne s'intéressant pas à la réalisation pratique de l'acquisition par canal auxiliaire. Toutefois, nous veillons à ce que les hypothèses sur les informations externes soient relativement raisonnables, pour toujours correspondre à des attaques par canaux auxiliaires réalistes. Par exemple, on peut supposer qu'un certain nombre de bits ou que des relations entre certains bits sont connus. Appliqués à la cryptographie asymétrique, ces travaux ont conduit à la publication d'une nouvelle attaque contre les cryptosystèmes de type DSA. Dans le cas de la cryptographie symétrique, des concepts de calcul formel pour la résolution de systèmes ont été utilisés pour analyser les attaques algébriques par canaux auxiliaires récemment introduites par M. Renauld, F.X. Standaert et N. Veyrat-Charvillon ([RS09a, RSVC09]). Nous détaillons ci-après ces contributions.

Attaquer (EC)DSA avec seulement une information implicite. La signature numérique permet d'identifier l'auteur d'un message numérique,

de la même façon qu'une signature manuscrite pour un document papier. La loi française [2] reconnait d'ailleurs depuis 2001 la même valeur juridique à une signature numérique qu'à une signature manuscrite. Cependant, la sécurité de la signature numérique doit être établie par une procédé cryptographique qualifié qui garantit également l'intégrité du document signé en plus de l'identité du signataire. Les procédés de signatures interviennent surtout dans les protocoles de communication sécurisée pour garantir l'étape d'authentification. La sécurité des schémas de signature numérique est donc indispensable pour assurer la sécurité des systèmes d'informations.

Les schémas de signature utilisent la cryptographie asymétrique et les fonctions de hachage. Une partie de la sécurité est donc basée sur des cryptosystèmes asymétriques exploitant généralement des problèmes venant de la théorie des nombres, comme le problème de la factorisation (RSA) et le problème du calcul du logarithme discret dans un groupe fini. Le chapitre 5 propose une cryptanalyse avec oracle du Digital Signature Algorithm (DSA [FIP94]) dont la sécurité est basée sur la difficulté du calcul du logarithme discret. Les résultats présentés sont cependant valables pour toutes les variantes du DSA, telles que la variante sur les courbes elliptiques (ECDSA [JMV01]) ou encore les schémas de signature ElGamal [ElG85] et Schnorr [Sch90]. Dans ces schémas de signature, chaque utilisateur possède une clef privée associée à une clef publique qui sont telles que la clef privée est le logarithme discret de la clef publique dans un groupe particulier. Pour signer un message, l'utilisateur a nécessairement besoin de sa clef privée et d'un nombre généré aléatoirement appelé la clef éphémère. Cette clef éphémère doit rester secrète et doit être renouvelée pour chaque nouveau message à signer.

De nombreux travaux proposent déjà l'utilisation d'un oracle contre DSA et ses variantes [HGS01, NS02, NS03, MNS01, BGM97, Pou09], la notion d'oracle permettant de modéliser la quantité d'information supplémentaire disponible pour un attaquant. Ces attaques exploitent généralement la connaissance d'un petit nombre de bits des clefs éphémères pour retrouver la clef privée. Dans un contexte où un attaquant possède suffisamment de telles informations supplémentaires, le schéma de signature devient donc totalement vulnérable. D'un point de vue pratique, de telles attaques se sont ensuite révélées réalisables grâce à l'analyse de canaux auxiliaires, avant que le développement de contre-mesures efficaces permettent d'éviter ces fuites d'informations. Ces contre-mesures sont fréquemment utilisées de nos jours et l'hypothèse qu'un attaquant puisse déterminer la valeur d'un

2. Décret 2001-272 du 30 mars 2001.

ensemble de bits des clefs secrètes n'est donc plus justifiée. Cependant, May et Ritzenhofen ont montré à PKC2009 [MR09] que dans le contexte de la factorisation, un attaquant n'avait pas besoin de connaitre la valeur explicite de certaines valeurs secrètes, car seulement une information implicite pouvait suffire. Dans leur article, l'information exploitée est la connaissance de deux modules RSA tels qu'un facteur d'un module partage quelques bits avec un facteur de l'autre module. Mais les valeurs des bits restent inconnues de l'attaquant tant que la factorisation n'a pas été obtenue, ce qui donne un caractère implicite à l'information nécessaire. May et Ritzenhofen ont donné un algorithme efficace basé sur la réduction de réseau qui factorise deux modules en temps quadratique pourvu que deux de leur facteurs partagent suffisamment de bits en commun. Les résultats de May et Ritzenhofen ont ensuite été améliorés et étendus dans [SM09, FMR10, MSS10].

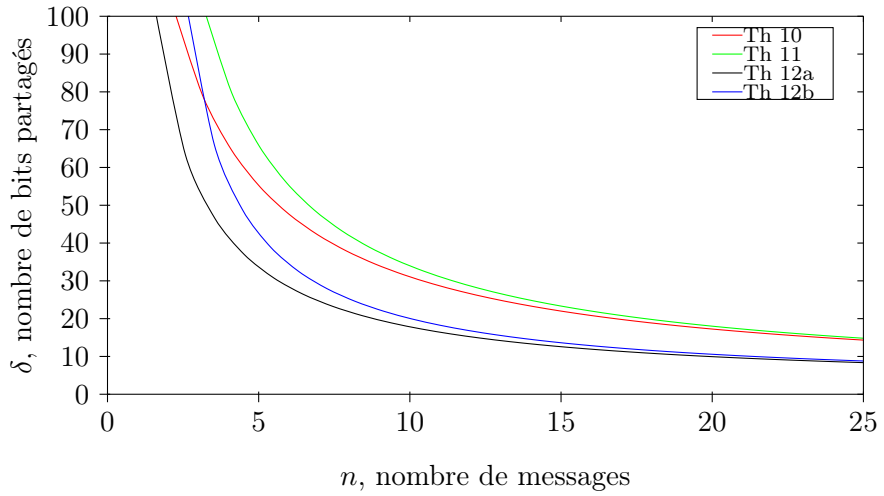


FIGURE 1 – Bornes théoriques sur l'information implicite contre (EC)DSA

Une information implicite similaire peut-elle être exploitée dans le cas d'algorithmes basés sur le problème du logarithme discret ? Le chapitre 5 répond par l'affirmative en présentant une attaque contre les schémas de signatures de type DSA en supposant seulement qu'une information implicite sur les clefs éphémères est connue. Plus précisément, on suppose qu'on dispose de plusieurs messages signés par un schéma du type DSA, et tels que les clefs éphémères utilisées pour construire les signatures partagent un certain nombre δ de bits en commun. Cette information est implicite dans

le sens où les valeurs des bits en commun restent inconnues tant que la clef secrète reste inconnue. On montre alors que cette information implicite peut être extraite en construisant judicieusement un réseau contenant un vecteur court dont les coefficients permettent de retrouver la clef secrète. L'attaque aboutit donc si ce vecteur peut être retrouvé facilement, par exemple avec l'algorithme de réduction de réseau LLL ([LLL82]), donc si le vecteur est assez court. Cela se produit quand les clefs éphémères partagent suffisamment de bits. La figure 1 montre, dans différents contextes et en supposant l'heuristique gaussienne vérifiée, le nombre δ de bits en commun théoriquement nécessaires en fonction du nombre n de messages disponibles.

Ces résultats théoriques sont vérifiés par des implantations en MAGMA. On obtient par exemple, que seulement 4 LSB partagés entre les clefs éphémères utilisées pour signer 100 messages sont suffisants avec un taux de succès atteignant 100%. Avec seulement 3 LSB partagés, il faut au moins 200 messages signés vérifiant l'information implicite. Ces expérimentations pratiques confirment l'efficacité théorique de l'attaque proposée, avec des temps de calculs qui restent inférieurs à 5s. Les applications possibles de ce type d'attaque exploitant une information implicite sont nombreuses. En plus des applications déjà proposées dans [LPS04, MR09, FMR10], on peut aussi penser aux attaques actives et invasives (blocage de registre), ainsi qu'une manipulation malicieuse des générateurs de nombres aléatoires. On a d'ailleurs vérifié que les tests statistiques classiques n'étaient pas en mesure de détecter la présence de cette information implicite dans un flot de nombres aléatoires.

Les résultats de ce chapitre ont été publiés dans un article pour la *conference on Selected Areas in Cryptography 2012* (SAC2012) [FGR12], raison pour laquelle il est rédigé en anglais.

Analyse des attaques ASCA. Les attaques algébriques par canaux cachés (ASCA) font partie d'un nouveau type d'attaques introduites à CHES 2009 ([RSVC09]) par M. Renauld, F.X. Standaert et N. Veyrat-Charvillon. Elles combinent la cryptanalyse algébrique classique avec des attaques par canaux cachés afin de bénéficier des avantages de ces deux attaques. Différentes méthodes ont d'ailleurs déjà été proposées pour combiner les attaques par canaux cachés avec des attaques algébriques ([HP06, KSP04, BKP08, MME10]), ou bien avec de la cryptanalyse différentielle ([HP06, KSP04]) et linéaire ([RT09b, RT09a]). Mais contrairement aux autres méthodes, les attaques ASCA restent efficaces avec un nombre restreint de mesures et pourraient même réussir avec l'observation du chiffrement d'un seul texte clair

uniquement. Les implémentations masquées pourraient aussi être attaquées efficacement ([RSVC09]). L'idée principale des ASCA est de commencer par une phase d'acquisition où des informations de fuites sont récupérées par un canal auxiliaire suivant un modèle de fuite fixé, avant d'être ensuite utilisées dans une phase de cryptanalyse algébrique pour retrouver la clef secrète. Néanmoins, les informations récupérées par canal auxiliaire sont supposées être fiables pour pouvoir être utilisées dans la phase algébrique, bien que de récents articles ([OKPW10, ZZG$^+$12]) montrent que les ASCA pourraient être efficace en présence d'erreurs. La phase algébrique consiste à modéliser le cryptosystème et le modèle de fuite par un système d'équations polynomiales et à résoudre ce système pour retrouver la valeur de la clef secrète.

L'objectif principal de nos travaux présentés au chapitre 6 est d'expliquer l'efficacité de cette attaque en décrivant un critère de la réussite et finalement de trouver les conditions théoriques pour prévenir les attaques algébriques par canaux cachés. Pour cela, nous supposons les mêmes hypothèses que dans [RSVC09, RS09a, RS09b, RS10], en particulier qu'une phase initiale d'acquisition nous a déjà fourni la séquence des valeurs de fuites. Ainsi nous ne nous concentrons que sur la phase de cryptanalyse algébrique. Nous cherchons à utiliser principalement des techniques de résolutions par bases de Gröbner plutôt que des SAT solveurs quand cela est possible. En effet, bien que les SAT solveurs soient souvent plus efficaces que les calculs de bases de Gröbner consommant beaucoup de mémoire, ils dépendent d'heuristiques qui rendent la résolution imprévisible et les calculs de complexité difficiles à obtenir. Par ailleurs, on montre que la complexité des résolutions par bases de Gröbner lors de ces attaques dépend d'une nouvelle notion d'immunité algébrique et de la distribution des informations de fuite. Cette immunité algébrique avec fuite est définie comme le degré et le nombre des relations de plus bas degré définissant une boite noire (SBoxes, MixColumns, dérivation de clef, etc) avec son information de fuite. Il s'agit d'une mesure précise pour estimer la complexité de résolution par calcul de base de Gröbner et ainsi fournit un bon critère pour prévoir l'efficacité de la résolution. Notre étude des attaques ASCA s'est limitée aux cryptosystèmes PRESENT et AES. L'étude de leurs boîtes-S montre que les fuites d'informations peuvent être utilisées pour linéariser partiellement le système d'équations. Par exemple, la probabilité d'obtenir au moins 64 (resp. 130) relations linéaires par tour est d'environ 50% pour PRESENT (resp. AES). On fournit ainsi une explication théorique relativement précise des résultats expérimentaux obtenus dans [RSVC09, RS09a, RS09b, RS10], et on peut décrire des conditions nécessaires pour de possibles contre-mesures. Par exemple, si les boîtes de substitution sont remplacées par des fonctions minimisant le critère alors les

deux attaques algébriques deviennent impraticables. Le chapitre 6 reprend des résultats précédemment publiés en anglais dans *Journal of Cryptographic Engineering* [CFGR12]. Ces travaux ont également été présentés aux *workshop on Constructive Side-Channel Analysis and Secure Design 2011* (COSADE2011) [FGR11].

Dans le chapitre 6, on se place sous les mêmes hypothèses que dans les articles [RS09a, RSVC09], en particulier le modèle de fuite choisi est le poids de Hamming des valeurs d'entrées et de sorties des boîtes-S. Mais d'autres modèles de fuites peuvent être intéressants. Le chapitre 7 est donc consacré à l'étude de l'influence du modèle de fuite. En particulier, on montre que le critère reste pertinent lorsqu'il est appliqué avec différents modèles de la distance de Hamming. Le premier modèle étudié est la distance de Hamming entre l'entrée et la sortie des boîtes-S. On montre que ce modèle permet encore de linéariser partiellement le système d'équations qui modélise le cryptosystème. Mais en comparaison avec le modèle du poids de Hamming, le nombre de relations linéaires obtenues est beaucoup plus faible. Plus précisément, l'espérance du nombre de relations linéaires est de 2.3 pour PRESENT et de $1,4$ pour l'AES (alors que ces valeurs sont supérieures à 7 pour le modèle du poids de Hamming), ce qui explique pourquoi l'attaque par calcul de bases de Gröbner est beaucoup plus difficile avec le modèle de la distance de Hamming. Cependant, le critère de réussite montre que le coût d'une recherche exhaustive sur l'entrée de l'étape de substitution diminue significativement, ce qui permet d'obtenir des taux de réussites importants pour les attaques par SAT solveur (de l'ordre de 70% d'attaques réussies avec PRESENT après 3 heures). Le second modèle étudié est la distance de Hamming entre des chiffrements successifs (cf. section 7.2.2). Ce nouveau modèle ne permet pas la linéarisation partielle des étapes de substitutions. Mais la borne sur le coût d'une attaque par SAT solveur donnée par le critère de réussite reste du même ordre que pour le modèle de la distance de Hamming entre l'entrée et la sortie des boîtes-S. Les expériences confirment que le critère est toujours pertinent avec ses deux modèles de fuite.

En plus de l'étude de ces deux modèles de la distance de Hamming, on montre que le choix d'un modèle de fuite peut apporter une première solution pour la gestion des informations de fuite avec erreur. Dans cette étude, on choisit le modèle du poids de Hamming avec une incertitude entre deux valeurs adjacentes de poids possibles. Le critère de réussite se révèle encore pertinent dans ce modèle particulier et permet de prévoir correctement les résultats expériences. Finalement, comme perspectives de recherche, on voit que les attaques par fautes, ([PQ03]) ou encore que les attaques par collisions internes, ([BKP08]) peuvent aussi être considérées comme des ASCA

en utilisant d'autres modèles de fuite particuliers

Plan de lecture Nous avons choisi de présenter nos travaux en séparant les rappels des contributions. La première partie contient les présentations les préliminaires et les outils que nous utilisons dans les chapitres de la seconde partie.

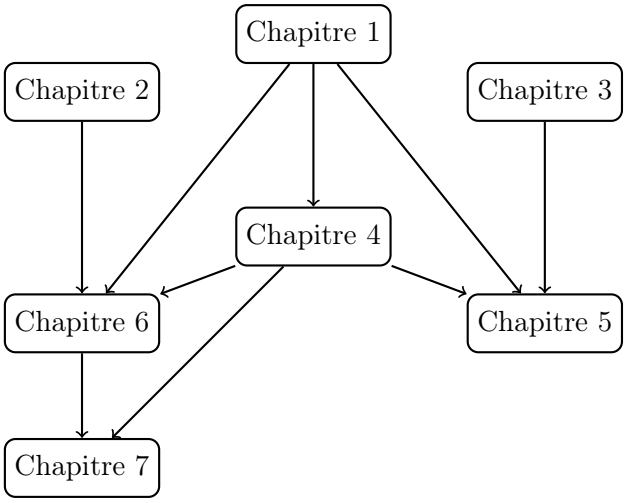


FIGURE 2 – Plan de lecture

Le chapitre 1 contient la description des systèmes cryptographiques que nous avons étudiés : l'AES et PRESENT pour les cryptosystèmes symétriques, et le DSA ainsi que sa variante sur les courbes elliptiques ECDSA pour ce qui concerne les cryptosystèmes asymétriques. L'exploitation des canaux auxiliaires pour la récupération des informations de fuite, nécessaire dans tous les chapitres de contribution, fait l'objet du chapitre 4.

Le chapitre 3 présente les méthodes de réduction de réseaux qui seront utilisées dans l'attaque sur (EC)DSA avec information implicite du chapitre 5. Ces deux chapitres peuvent être lu de manière indépendante des autres chapitres à condition de connaitre les schémas de signature de type DSA.

Le chapitre 2 est consacré à la résolution de systèmes polynomiaux par le calcul de bases de Gröbner , ainsi que par SAT solveurs pour les systèmes booléens. Ces deux méthodes sont successivement utilisées dans l'analyse des attaques ASCA étudiées dans les chapitres 6 et 7.

Première partie

Préliminaires

Chapitre 1

Rappels sur la cryptographie

1.1 Introduction

Les algorithmes cryptographiques peuvent se classer en deux grandes catégories : les chiffrements symétriques et les chiffrements asymétriques. Un algorithme de chiffrement est dit symétrique quand il utilise la même clef aussi bien pour chiffrer que pour déchiffrer. Historiquement, c'est la plus ancienne forme de chiffrement (e.g. Vigenère, Enigma, [Sin99]). Une contrainte importante des chiffrements symétriques est la nécessité d'avoir préalablement échangé la clef secrète avec son interlocuteur. La cryptographie asymétrique a été inventé pour résoudre ce problème ([DH76]). En effet, un chiffrement asymétrique utilise une clef publique pour le chiffrement qui est différente de la clef privée de déchiffrement, cette dernière devant rester secrète. La sécurité des chiffrements asymétriques repose sur le principe que la clef privée doit être difficile à retrouver à partir de la clef publique. Contrairement aux chiffrements symétriques, un système asymétrique présente l'avantage de pouvoir envoyer un message chiffré sans la nécessité de partager un secret.

Dans ce chapitre, on décrit les algorithmes cryptographiques utilisés dans cette thèse, en commençant par les cryptosystèmes symétriques AES et PRESENT. Concernant les cryptosystèmes asymétriques, nous détaillons le schéma de signature DSA et sa variante sur les courbes elliptiques ECDSA qui font l'objet du chapitre 5.

1.2 Rappels sur les corps finis

Les algorithmes cryptographiques étudiés dans cette thèse utilisent de l'arithmétique sur les corps finis. On a donc besoin de présenter quelques

rappels sur les corps finis. Pour une présentation plus complète, voir par exemple le livre de Lidl et Niederreitter [LN96].

Un corps fini est constitué d'un ensemble fini d'éléments $\mathbb{F}$ muni de deux lois de composition interne, l'addition et la multiplication, formant un corps commutatif. L'ordre d'un corps fini est son nombre d'éléments. Il existe un corps fini d'ordre q si et seulement si q est une puissance d'un nombre premier. Si q est une puissance d'un nombre premier, alors il existe à isomorphisme près un unique corps fini d'ordre q, qu'on note $\mathbb{F}_q$. Dans ce cas, $q = p^m$ avec p un nombre premier appelé la caractéristique de $\mathbb{F}_q$ et m un entier appelé le degré d'extension de $\mathbb{F}_q$. Un corps fini $\mathbb{F}_p$ avec p premier (où le degré d'extension vaut donc 1) est appelé un corps premier et $\mathbb{F}_p$ est alors isomorphe à l'anneau $\mathbb{Z}/p\mathbb{Z}$ constitué de l'ensemble des entiers $\{0, 1, \ldots, p\}$ muni de l'addition et de la multiplication modulo p.

Pour construire les corps finis $\mathbb{F}_{p^m}$ qui ne sont pas premiers (i.e. $m > 1$), on utilise la structure d'anneau euclidien de l'ensemble $\mathbb{F}_p[X]$ des polynômes en une variable à coefficients dans $\mathbb{F}_p$. En effet, pour $f(X), g(X) \in \mathbb{F}_p[X]$, il existe un unique quotient $q(x) \in \mathbb{F}_p[X]$ et un unique reste $r(X) \in \mathbb{F}_p[X]$ tels que

$$f(X) = q(X)g(X) + r(X)$$

avec $r(X)$ de degré strictement inférieur au degré de $g(X)$. L'algorithme de division sur les polynômes est similaire à la division sur les entiers. On dit alors que $g(X)$ divise $f(X)$ lorsque le reste de la division de $f(X)$ par $g(X)$ est le polynôme nul. Deux polynômes $f(X)$ et $g(X)$ sont dit équivalents modulo $p(X)$ si le polynôme $p(X)$ divise le polynôme $f(X) - g(X)$ (i.e. si $f(X)$ et $g(X)$ ont le même reste pour la division par $p(X)$). On peut maintenant définir l'anneau $\mathbb{F}_p[X]/(f)$ des polynômes modulo $f(X)$ comme étant le quotient de $\mathbb{F}_p[X]$ par la relation d'équivalence ainsi obtenue. L'anneau $\mathbb{F}_p[X]/(f)$ est donc l'ensemble des polynômes de $\mathbb{F}_p[X]$ de degré inférieur au degré de $f(X)$, avec l'addition et la multiplication définies comme dans $\mathbb{F}_p[X]$ suivies d'une réduction modulo $f(X)$. Sur les polynômes, la notion analogue à la primarité est appelée l'irréductibilité, et on a alors la propriété suivante :

Proposition 1. *Soit $f(X) \in \mathbb{F}_p[X]$ avec p un nombre premier. L'anneau $\mathbb{F}_p[X]/(f)$ est un corps si et seulement si $f(X)$ est irréductible, c'est-à-dire si $f(X)$ ne peut pas se factoriser comme un produit de deux polynômes de degrés non-nuls sur $\mathbb{F}_p[X]$.*

On peut montrer que pour tout nombre premier p et tout entier $n > 0$, il existe au moins un polynôme irréductible $f(X)$ de degré n dans $\mathbb{F}_p[X]$ et

donc il existe au moins un corps fini avec p^n éléments. De plus, tous les corps finis à p^n éléments sont isomorphes. On peut aussi montrer que le groupe multiplicatif d'un corps fini $\mathbb{F}_{p^n} \simeq \mathbb{F}_p[X]/(f)$ est un groupe cyclique d'ordre $p^n - 1$.

Exemple 1. *Soit le polynôme irréductible* $f(X) = X^4 + X + 1 \in \mathbb{F}_2[X]$. *Le corps fini* $\mathbb{F}_{2^4}$ *peut donc être défini par* $\mathbb{F}_2[X]/(f)$. *Par exemple, en posant* $g(X) = X^3 + X^2 + 1$ *et* $h(X) = X^3 + 1$, *on calcule*

- $g(X) + h(X) = (X^3 + X^2 + 1) + (X^3 + 1) = X^2$
- $g(X)h(X) = X^3 + X^2 + X + 1$, *car* $(X^3 + X^2 + 1)(X^3 + 1) = X^6 + X^5 + X^2 + 1$ *et* $X^6 + X^5 + X^2 + 1 \mod (X^4 + X + 1) = X^3 + X^2 + X + 1$.
- *l'inverse de* $g(X)$ *est le polynôme* X^2. *En effet,* $(X^3 + X^2 + 1)(X^2) = X^5 + X^4 + X^2$ *et donc* $g(X)X^2 = X^5 + X^4 + X^2 \mod (X^4 + X + 1) = 1$.

Les éléments des corps $\mathbb{F}_{2^m}$ *sont souvent représentés par des nombres binaires de* m *bits constitués de la liste des coefficients du polynôme correspondant. Par exemple,* $g(X)$ *peut être représenté par le nombre binaire* (1101).

1.3 Cryptosystèmes symétriques

Dans cette section, on décrit les chiffrements par blocs AES et PRESENT. Ces deux systèmes de chiffrement sont des réseaux de substitution-permutation (SPN), c'est-à-dire qu'ils sont constitués d'itérations d'une *fonction de tour*, elle-même formée d'une étape de substitution et d'une permutation. Ces systèmes contiennent également un algorithme de *diversification de clef*.

1.3.1 AES

En 2001, le système de chiffrement Rijndael [FIP01] a été choisi comme l'*Advanced Encryption Standard* (AES) par le NIST, après un processus de sélection parmi différents candidats. Cet algorithme a été évalué et sélectionné suivant des critères de sécurité, de coût, de performance, d'efficacité et de facilité d'implantation.

L'AES prend toujours en entrée un bloc de 128 bits, mais trois longueurs de clef sont possibles, dont dépend le nombre d'itérations. On a donc trois variantes de l'AES :

- AES-128 possède une clef de 128 bits et utilise 10 tours,
- AES-192 possède une clef de 192 bits et utilise 12 tours,
- AES-256 possède une clef de 256 bits et utilise 14 tours.

Nous décrivons seulement la version 128 bits de l'AES, ce qui est suffisant pour cette thèse. Voir [FIP01] pour une description des autres versions.

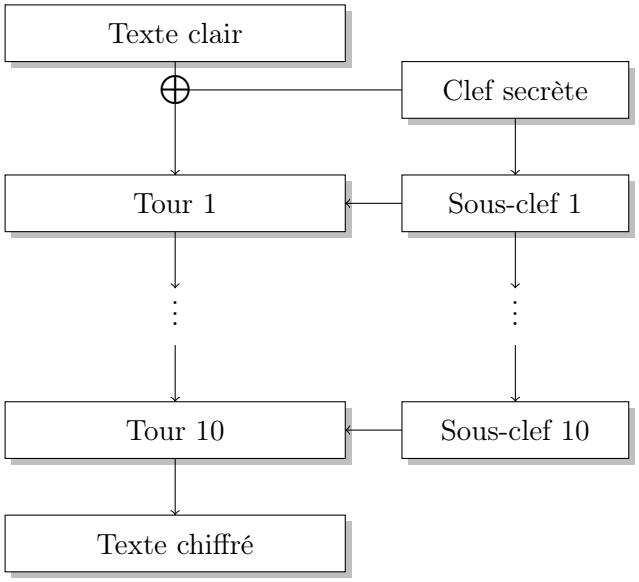


FIGURE 1.1 – Schéma de l'algorithme AES

L'algorithme se déroule suivant la figure 1.1 : la clef secrète est d'abord combinée au texte clair au moyen d'un ou-exclusif, suivit des 10 tours de l'AES (figure 1.1). Pour chaque tour, une sous-clef est dérivée de la sous-clef précédente.

Un tour normal de l'AES (figure 1.2) est constitué de la succession des opérations suivantes :

1. une étape de substitution appelée *SubBytes*,
2. une permutation *ShiftRows*,
3. une permutation *MixColumns* (application affine),
4. une opération *AddRoundKey* d'ajout d'une sous-clef au moyen d'un ou-exclusif.

Tous les tours sont identiques sauf le dernier tour qui ne contient pas la transformation MixColumns.

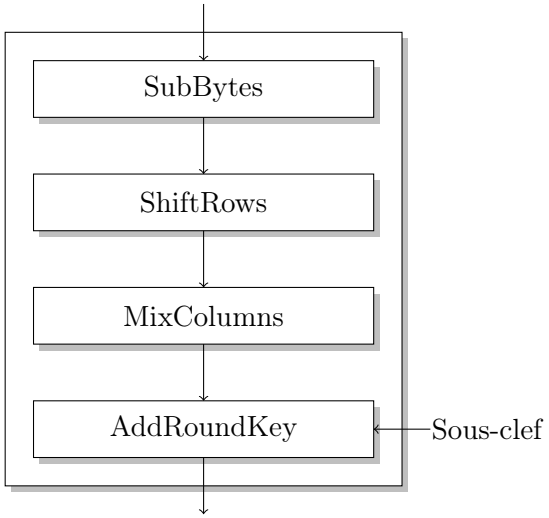


FIGURE 1.2 – Description d'un tour de l'AES

Pour décrire les différentes opérations, on découpe les 128 bits d'un bloc en 16 octets, qu'on note $x_0, \ldots, x_{15}$. Ces octets sont alors représentés sous forme d'un tableau d'octets de 4 lignes et 4 colonnes (figure 1.3), appelé **State**, sur lequel sont successivement effectuées les différentes opérations de l'AES.

x_0	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}

x_0	x_4	x_8	x_{12}
x_1	x_5	x_9	x_{13}
x_2	x_6	x_{10}	x_{14}
x_3	x_7	x_{11}	x_{15}

FIGURE 1.3 – State contient initialement les 16 octets du texte clair

SubBytes

L'opération SubBytes est une transformation non linéaire consistant à substituer chaque octet de State par un autre octet suivant une table de substitution (figure 1.4). Cette table de substitution, qu'on peut aussi appeler "boîte-S", est définie algébriquement en composant deux opérations dans un corps fini :

1. l'inversion dans le corps fini $\mathbb{F}_{2^8}$ (le 0, qui n'a pas d'inverse, est envoyé sur lui-même)
2. une transformation affine dans $\mathbb{F}_2$:

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

où les b_i sont les bits de l'octet obtenu après l'inversion.

		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
	1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
	2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
	3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
	4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
	5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
	6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
	7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
	8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
	9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
	a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
	b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
	c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
	d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
	e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
	f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

FIGURE 1.4 – Table de substitution de l'AES pour l'octet xy noté sous forme hexadecimale. (Source [FIP01])

ShiftRows

L'opération ShiftRows agit sur State de la façon suivante :

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

c'est-à-dire que les octets des trois dernières lignes subissent plusieurs rotations cycliques vers la gauche (respectivement 1, 2 et 3).

MixColumns

L'opération MixColumns est une application linéaire qui agit sur State colonne par colonne. On peut la représenter comme une multiplication par une matrice d'éléments dans $\mathbb{F}_{2^8}$:

$$\begin{array}{|c|} \hline x_i \\ \hline x_{i+1} \\ \hline x_{i+2} \\ \hline x_{i+3} \\ \hline \end{array} \xrightarrow{\text{MixColumns}} \begin{pmatrix} X & X+1 & 1 & 1 \\ 1 & X & X+1 & 1 \\ 1 & 1 & X & X+1 \\ X+1 & 1 & 1 & X \end{pmatrix} \begin{array}{|c|} \hline x_i \\ \hline x_{i+1} \\ \hline x_{i+2} \\ \hline x_{i+3} \\ \hline \end{array}$$

En considérant chaque colonne comme un polynôme de $\mathbb{F}_{2^8}[z]$, cette opération peut aussi être vue comme le résultat d'une multiplication dans $\mathbb{F}_{2^8}[z]/(z^4+1)$ par le polynôme $a(z) = (X+1)z^3 + z^2 + z + X$ (cf. [FIP01]).

Diversification de la clef

Pour chaque tour, une nouvelle sous-clef est dérivée à partir de la sous-clef du tour précédent par l'algorithme de diversification de clef. L'opération AddRoundKey consiste alors à ajouter la sous-clef du tour courant au moyen d'un ou-exclusif sur chaque octet de State. Il reste donc juste à décrire la dérivation de clef. Une sous-clef est constituée de 16 octets qu'on découpe en 4 mots de 4 octets chacun.

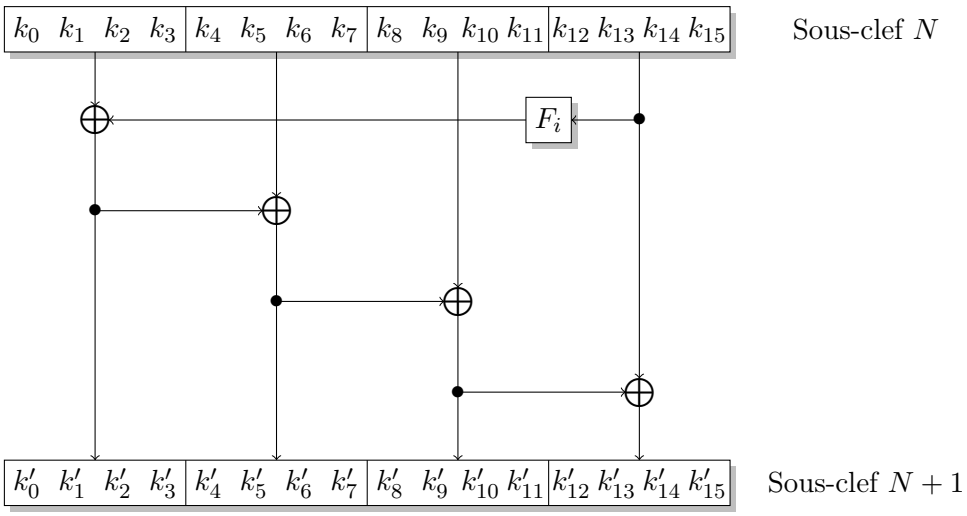


FIGURE 1.5 – Algorithme de diversification de clef de l'AES

Une fonction non linéaire F_i est appliquée sur le dernier mot formé des octets $k_{12}, k_{13}, k_{14}, k_{15}$. Cette fonction consiste à appliquer successivement, sur chacun des 4 octets d'entrée, la boîte-S, suivit d'une rotation circulaire vers la gauche et enfin d'additionner $X^{i-1} \in \mathbb{F}_{2^8}$ au premier octet (Voir [FIP01] pour plus de détails). La figure 1.5 montre comment les différents octets de la sous-clef suivante sont obtenus par différents ou-exclusifs.

1.3.2 PRESENT

PRESENT est un chiffrement par bloc proposé par Bogdanov *et al.* à CHES 2007 [BKL+07]. Il a été conçu dans l'objectif de fournir un chiffrement "ultra-léger" afin d'être implanté dans des environnements à fortes contraintes, tel que pour les systèmes RFID par exemple. En plus de la sécurité, de la facilité d'implantation et de l'efficacité, la simplicité serait le principal critère ayant guidé les auteurs lors de sa conception.

PRESENT se présente comme un SPN composé de 31 tours. La taille d'un bloc est de 64 bits et la clef peut être de 80 ou 128 bits. La version 80 bits est recommandée par les auteurs pour des raisons de performance. C'est également la seule version utilisée dans cette thèse. Nous nous limitons donc à décrire la version avec 80 bits de clef (voir [BKL+07] pour plus de détails).

Tout comme l'algorithme AES, le chiffrement PRESENT commence par ajouter les 64 premiers bits de la clef secrète au moyen d'un ou-exclusif,

avant d'exectuer les 31 tours. Un tour de PRESENT consiste en une étape de substitution non linéaire, appelée *sBoxLayer*, une étape de permutation des bits, appelée *pLayer* et enfin un ou-exclusif de la sous-clef correspondant au tour (cf. figure 1.6).

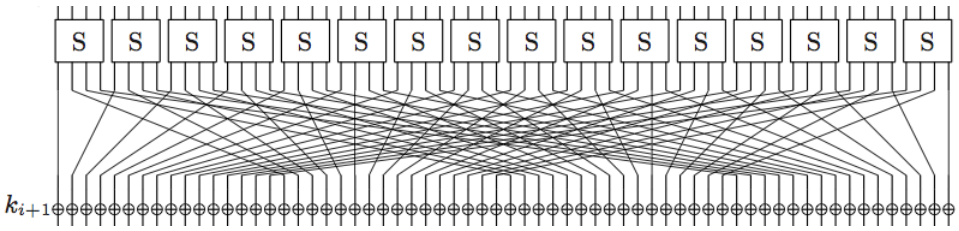


FIGURE 1.6 – Un tour de PRESENT

sBoxLayer

La couche de substitution utilise une unique boîte-S de 4-bits qui est appliqué 16 fois en parallèle sur chaque mot de 4-bits. La table suivante décrit la boîte-S avec une notation hexadécimale :

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S[x]$	C	5	6	B	9	0	A	D	3	E	F	8	4	7	1	2

pLayer

La permutation des bits utilisée dans PRESENT est donnée par la table suivante, où le bit i de pLayer est déplacé à la position $P(i) = 16i \mod 64$.

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

Diversification de la clef

PRESENT supporte des clefs de 80 ou 128 bits, mais nous ne décrivons que la diversification d'une clef de 80 bits. La clef est mise dans un registre K de 80 bits représentée par $K_{79}K_{78}\dots K_0$. Les bits de sous-clefs utilisés pour un tour N sont les 64 bits les plus à gauche du registre K. Après avoir effectué l'ajout des bits de clef à la fin d'un tour, le contenu du registre K doit être mise-à-jour pour le tour suivant. Pour cela, l'algorithme de diversification effectue successivement une rotation circulaire de 61 bits sur la gauche, puis les 4 bits les plus à gauche sont substitués suivant la boîte-S, et enfin un compteur est ajouté au registre avec un ou-exclusif sur les bits $(K_{19}K_{18}K_{17}K_{16}K_{15})$.

1.4 Cryptosystèmes asymétriques

En ce qui concerne les cryptosystèmes asymétriques, nos travaux n'ont porté que sur les schémas de signature du type DSA. Un schéma de signature permet de signer un document numérique. La signature numérique permet alors d'authentifier l'auteur d'un document électronique (comme une signature manuscrite), mais aussi de garantir l'intégrité du document. Le *Digital Signature Algorithm* (**DSA**), et sa variante *Elliptic Curve Digital Signature Algorithm* (**ECDSA**) utilisant des courbes elliptiques, font partie du *Digital Signature Standard* (**DSS**) spécification standardisée adoptée par le *National Institute of Standards and Technology* (NIST) [FIP09] aux Etats-Unis. La sécurité de ces schémas de signature est basée sur la difficulté du calcul du logarithme discret (DLP) dans un groupe cyclique fini : dans un groupe fini G engendré par g, étant donné un élément $y \in G$, le calcul du logarithme discret consiste à trouver un entier x tel que $g^x = y$.

On peut prouver qu'un algorithme générique pour calculer le logarithme discret dans tout groupe est nécessairement exponentiel ([Tes01, Sho97, Sha71, Pol78, Pol00]). Il existe cependant des algorithmes sous-exponentiels pour calculer le logarithme discret dans certaines classes de groupes particuliers. Par exemple, le logarithme discret peut être calculé en temps sous-exponentiel dans les groupes multiplicatifs des corps finis ([AD93]) ou sur certaines courbes hyperelliptiques ([ADH94, ADH99]). Inversement, il n'y a pas d'algorithme sous-exponentiel connu pour résoudre le DLP sur le groupe des points rationnels d'une courbe elliptique (ECDLP) bien choisie. Ainsi le problème ECDLP apparaissant comme étant plus dur que le DLP sur les groupes des corps finis, des paramètres de sécurités plus petits peuvent être utilisés sur les courbes elliptiques tout en gardant un niveau de sécurité

équivalent, ce qui procure des avantages intéressants : rapidité des calculs, petites tailles de clefs, etc. Ces avantages sont particulièrement importants dans des environnements où la puissance de calcul, l'espace de stockage, la bande passante, ou la consommation électrique sont limités, comme pour les systèmes embarqués par exemple (carte à puces, RFID, etc).

1.4.1 DSA

Le *Digital Signature Algorithm* (DSA) a été adopté en 1993 par le NIST pour faire partie de la spécification **DSS**. Après plusieurs révisions et améliorations, la description actuelle du DSA est disponible dans FIPS186-3 [FIP09]. Décrivons rapidement le fonctionnement de ce schéma de signature dont la sécurité est basée sur le problème du logarithme discret.

Soit p un nombre premier d'au moins 1024 bits et soit q un nombre premier d'au moins 160 bits tel que q divise $p-1$. DSA utilise alors l'unique sous-groupe G d'ordre q de $\mathbb{Z}_p^*$.

La clef privée est un entier $x \in \{1, \ldots, q-1\}$ et la clef publique est l'élément y de G vérifiant $y = g^x \pmod q$ où g est un générateur de G. Les paramètres p, q et g sont des éléments publiques.

Il est spécifié dans DSA que tout message doit être haché (originairement avec SHA-1) avant d'être signé. Pour signer un message m, l'auteur du message choisit alors un nombre aléatoire $k \in \{1, \ldots, q-1\}$ appelé le marquant (ou la clef éphémère), et calcule

$$r = (g^k \bmod p) \bmod q \quad \text{et} \quad s = k^{-1}(h(m) + xr) \bmod q \tag{1.1}$$

où $h(m)$ est le résultat d'un hachage cryptographique sur le message m.

La signature est alors le couple (r, s). Une signature est vérifiée lorsque

$$r = (g^u y^v \mod p) \bmod q$$

où

$$\begin{cases} u &= s^{-1}h(m) \bmod q \\ v &= s^{-1}r \bmod q \end{cases}$$

En effet, si la signature a été construite correctement, alors

$$y^v = g^{xs^{-1}r \bmod q}$$

donc

$$\begin{aligned} g^u y^v &= g^{s^{-1}(h(m)+xr) \bmod q} \\ &= g^k \end{aligned}$$

d'où la validité de la signature DSA :

$$r = (g^k \bmod p) \bmod q = (g^u y^v \mod p) \bmod q$$

1.4.2 ECDSA

L'*Elliptic Curve Digital Signature Algorithm* (**ECDSA**, [JMV01]) est une variante de DSA qui utilise la cryptographie sur les courbes elliptiques. L'algorithme ECDSA, proposé par Scott Vanstone en 1992 [Van92], a été standardisé en 2000, sa description actuelle étant disponibles dans FIPS186-3 [FIP09].

Soit p un nombre premier d'au moins 192 bits. ECDSA utilise le groupe des points rationnels d'une courbe elliptique E sur $\mathbb{Z}_p$, noté $E(\mathbb{Z}_p)$. Plus précisément, les calculs ont lieu dans un sous-groupe cyclique de $E(\mathbb{Z}_p)$ dans lequel le problème du logarithme discret est difficile. On cherche donc une courbe qui n'est pas supersingulière et telle que l'ordre de $E(\mathbb{Z}_p)$ contienne un grand facteur premier n. On renvoie à ANS X9.62 [X9.05] pour plus de précision. Soit alors un point $G \in E(\mathbb{Z}_p)$ d'ordre n.

La clef privée est un entier $a \in \{1, \ldots, n-1\}$ et la clef publique est un point Q de $E(\mathbb{Z}_p)$ vérifiant $Q = aG$. Les paramètres E, n et G sont aussi des éléments publiques.

Tout comme pour DSA, tout message doit être haché (originairement avec SHA-1) avant d'être signé. Pour signer un message m, l'auteur du message choisit alors un nombre aléatoire $k \in \{1, \ldots, n-1\}$ appelé le marquant (ou la clef éphémère), et calcule $kG = (x, y)$. Ensuite, il calcule

$$r = x \bmod n \quad \text{et} \quad s = k^{-1}(h(m) + ar) \bmod n$$

où $h(m)$ est le résultat d'un hachage cryptographique sur le message m.

La signature est alors le couple (r, s). Pour vérifier une signature, on commence par calculer

$$\begin{cases} u &= s^{-1}h(m) \bmod n \\ v &= s^{-1}r \bmod n \end{cases}$$

On pose alors

$$uG + vQ = (x_0, y_0)$$

et la signature est vérifiée lorsque

$$r = x_0 \bmod n$$

En effet, si la signature est correcte, on a $vQ = vaG$ d'où

$$uG + vQ = (s^{-1}(h(m) + ar) \bmod n)G = kG = (x, y)$$

et la validité de la signature ECDSA.

Chapitre 2

Résolution de systèmes polynomiaux

2.1 Introduction

La résolution des systèmes d'équations algébriques est un problème essentiel en calcul formel. Les nombreuses applications proviennent de différents domaines du calcul scientifique mais aussi de domaines industriels (Géométrie algorithmique, robotique, cryptographie, biologie...). En cryptanalyse algébrique en particulier, la sécurité des cryptosystèmes est mise à l'épreuve en cherchant à résoudre des systèmes d'équations polynomiales en plusieurs variables qui représentent des relations entre le texte en clair et le texte chiffré.

En toute généralité, le problème consiste donc à résoudre un ensemble d'équations $\{f_1(x_1, \ldots, x_n) = 0, \ldots, f_m(x_1, \ldots, x_n) = 0\}$ où les f_i sont des polynômes à coefficients dans un corps $\mathbb{K}$. Cet ensemble est appelé un système d'équations polynomiales et il est généralement noté $\{f_1, \ldots, f_m\}$ en omettant les égalités à zéro et les variables. Une solution du système est obtenue en spécifiant des valeurs pour les variables x_i qui appartiennent à la clôture algébrique de $\mathbb{K}$ et telles qu'elles satisfassent simultanément chaque équation du système. Le terme "résoudre" a alors plusieurs significations en fonction que le système ait une infinité de solutions, un nombre fini de solutions ou pas de solution du tout. Lorsqu'un système a une infinité de solutions, il est dit de dimension positive (en référence à la dimension de la variété algébrique associée) et la résolution consiste alors à trouver une description "intéressante" de l'espace des solutions. Les systèmes possédant un nombre fini de solutions ou aucune solution sont qualifiés respectivement de

systèmes zéro-dimensionnels et d'inconsistants. Suivant le contexte, résoudre un système consiste donc à calculer toutes les solutions, à trouver une seule solution ou bien à montrer que le système est inconsistant.

Dans le cas de la cryptographie, seules les solutions dans un corps fini $\mathbb{F}_q$ ont une signification. On restreint donc la résolution au problème de trouver les solutions dans le corps de base en ajoutant au système les équations de corps $x_i^q = x_i$ pour chaque variable x_i. S'il n'est pas inconsistant, le système obtenu possède alors un nombre fini de solutions. De plus, il possède aussi l'importante propriété d'être surdéterminé, c'est-à-dire d'avoir plus d'équations que d'inconnues (aux équations du système initial s'ajoutent les équations de corps pour chaque variables).

De nombreuses méthodes de résolution algébriques ont été proposées. La résolution des systèmes d'équations polynomiales peut être traitée par le calcul des bases de Gröbner ([Buc65, Buc06a, Buc06b]). Leur utilisation pour la résolution d'un système d'équations est donc présentée en détails dans la section 2.2. Une base de Gröbner est un ensemble générateur d'un idéal dans un anneau de polynômes qui possède des propriétés particulières. Une propriété caractéristique est par exemple de garantir l'unicité du reste de la division multivariée pour un ordre monomial donné. L'algorithme de Buchberger est historiquement le premier algorithme proposé pour calculer des bases de Gröbner, mais d'autres algorithmes généraux, en particulier les algorithmes F4 et F5 ([Fau99, Fau02]) ont permis, grâce à des implantations très performantes, la résolution d'applications hors d'atteinte par d'autres méthodes. La complexité du calcul d'une base de Gröbner d'un système possédant un nombre fini de solutions est cependant exponentiel en le nombre de variables. Afin d'améliorer la complexité de la résolution, des algorithmes spécifiques peuvent alors exploiter certaines propriétés des systèmes d'équations. Dans le cas de la résolution sur les corps finis, des approches hybrides combinant la recherche exhaustive avec le calcul de bases de Gröbner peuvent apporter un gain significatif ([BFP10]). Les algorithmes de type XL ([CKPS00, DBM$^+$08, MMDB08, MCD$^+$09]) sont aussi des algorithmes spécifiques à la recherche de solutions à coefficients dans un corps fini, mais il a été montré dans [AFI$^+$04, CL05, Ars05, ACFP12] qu'ils n'étaient pas plus efficaces que les algorithmes généraux du calcul des bases de Gröbner. Des algorithmes tirant parti des structures des systèmes d'équations, comme par exemple les systèmes avec symétries ([FR09]) ou les systèmes bi-homogènes ([FSEDS10]) ont aussi été proposés. L'utilisation de SAT solveur est également un moyen très efficace en pratique pour trouver une solution d'un système d'équations à coefficients dans $\mathbb{F}_2$. Une description de la méthode de résolution par SAT solveur est succinctement présentée dans la section 2.3

2.2 Résolution par bases de Gröbner

Dans cette thèse, l'analyse des attaques algébriques par canaux axillaires utilise principalement le calcul de bases de Gröbner comme un outil pour la résolution de systèmes polynomiaux. L'objectif de cette section est donc de présenter les propriétés et les algorithmes essentiels pour la résolution par calcul des bases de Gröbner. Pour une présentation plus détaillée des bases de Gröbner, nous renvoyons vers des ouvrages de références tels que [AL94, Frö98, EM07]. Dans une approche plus algorithmique, il y a aussi [Eis99, BKW93, CLO07]. Pour une étude des bornes de complexités, la référence est [Bar04].

Dans toute cette section, $\mathbb{K}$ est un corps et on se place dans l'anneau $\mathbb{K}[x_1, \ldots, x_n]$ des polynômes en n variables à coefficients dans $\mathbb{K}$.

2.2.1 Idéaux et variétés

Considérons un système de m équations polynomiales en n variables :

$$\begin{cases} f_1(x_1, \ldots, x_n) &= 0 \\ \vdots & \\ f_m(x_1, \ldots, x_n) &= 0 \end{cases}$$

qu'on identifie avec l'ensemble $S = \{f_1, \ldots, f_m\} \subset \mathbb{K}[x_1, \ldots, x_n]$ des m polynômes définissant ce système. On introduit alors un objet géométrique constitué de tous les zéros communs à tous les polynômes de S.

Définition 1 (Ensemble algébrique affine)**.** *Soit $\mathbb{L} \supseteq \mathbb{K}$ une extension de corps de $\mathbb{K}$. On pose*

$$\mathbf{V}_{\mathbb{L}}(S) = \{(z_1, \ldots, z_n) \in \mathbb{L}^n \mid \forall f \in S,\ f(z_1, \ldots, z_n) = 0\}$$

L'ensemble $\mathbf{V}_{\mathbb{L}}(S)$ est appelé l'ensemble algébrique affine défini par S sur $\mathbb{L}$. Par abus de langage, on dit aussi variété algébrique affine de S.

Exemple 2. *On a $\mathbf{V}_{\mathbb{L}}(\{1\}) = \emptyset$ et $\mathbf{V}_{\mathbb{L}}(\{0\}) = \mathbb{L}^n$.*

Afin de résoudre le système S, on va étudier son ensemble de solutions, c'est-à-dire $\mathbf{V}_{\mathbb{L}}(S)$. Généralement, on prend $\mathbb{L} = \overline{\mathbb{K}}$ la clôture algébrique de $\mathbb{K}$. Dans le contexte de la cryptographie où l'on s'intéresse surtout aux solutions sur un corps fini, on cherche plutôt à calculer $\mathbf{V}_{\mathbb{K}}(S)$ avec $\mathbb{K} = \mathbb{F}_q$.

Notons $I = \langle S \rangle$ l'idéal de $\mathbb{K}[x_1, \ldots, x_n]$ engendré par S :

$$I = \langle f_1, \ldots, f_m \rangle = \left\{ \sum_{i=1}^{m} a_i f_i \quad \text{avec } a_i \in \mathbb{K}[x_1, \ldots, x_n] \right\}$$

On remarque alors que tous les points de $\mathbf{V}_{\mathbb{L}}(S)$ annulent tous les polynômes de I, i.e. $\mathbf{V}_{\mathbb{L}}(S) = \mathbf{V}_{\mathbb{L}}(\langle S \rangle)$. La variété est indépendante de l'ensemble générateur choisi pour I. On notera donc $\mathbf{V}_{\mathbb{L}}(I)$ la variété associée à l'idéal I. On dira aussi que l'idéal I est zéro-dimensionnel si la variété associée $\mathbf{V}_{\overline{\mathbb{K}}}(I)$ (sur la clôture algébrique de $\mathbb{K}$) est finie.

L'objet algébrique déterminant pour l'ensemble des solutions est donc l'idéal engendré par le système d'équations et non pas les équations en elles-mêmes. Cette constatation nous a conduit à associer à tout idéal I un ensemble de points $\mathbf{V}(I)$. Mais réciproquement, à tout ensemble de points V peut être associé un idéal $\mathbf{I}(V)$ de l'anneau de polynômes.

Définition 2. *Soit V un sous-ensemble de $\overline{\mathbb{K}}^n$. On pose*

$$\mathbf{I}(V) = \{f \in \mathbb{K}[x_1, \ldots, x_n] \mid \forall (z_1, \ldots, z_n) \in V,\ f(z_1, \ldots, z_n) = 0\}.$$

L'idéal $\mathbf{I}(V)$ des polynômes s'annulant sur V est appelé l'idéal de V.

Rappelons que, pour tout corps $\mathbb{K}$, l'anneau $\mathbb{K}[x_1, \ldots, x_n]$ est noethérien. Tout idéal de $\mathbb{K}[x_1, \ldots, x_n]$, l'idéal $\mathbf{I}(V)$ en particulier, est donc engendré par un nombre fini de polynômes.

Exemple 3. $\mathbf{I}(\emptyset) = \mathbb{K}[x_1, \ldots, x_n]$.

Exemple 4. *Si $(z_1, \ldots, z_n) \in \mathbb{K}^n$, $\mathbf{I}(\{(z_1, \ldots, z_n)\}) = \langle x_1 - z_1, \ldots, x_n - z_n \rangle$.*

Exemple 5. *Pour toute partie $S \subseteq \mathbb{K}[x_1, \ldots, x_n]$, on a $\mathbf{V}(\mathbf{I}(\mathbf{V}(S))) = \mathbf{V}(S)$. Pour tout ensemble de points $X \subseteq \overline{\mathbb{K}}^n$, $\mathbf{I}(\mathbf{V}(\mathbf{I}(X))) = \mathbf{I}(X)$.*

L'opération $\mathbf{I}$ que l'on vient d'introduire n'est cependant pas l'inverse de l'opération $\mathbf{V}$ car en général $\mathbf{I}(\mathbf{V}(I)) \neq I$ comme le montre l'exemple suivant.

Exemple 6. *Prenons $I = \langle x_i^2 \rangle$, alors $\mathbf{I}(\mathbf{V}(I)) = \langle x_i \rangle \neq I$.*

En général, on a donc seulement $I \subset \mathbf{I}(\mathbf{V}(I))$. La correspondance exacte entre les ensembles algébriques affines et les idéaux nous est donnée par le théorème des zéros de Hilbert. Mais avant de pouvoir l'énoncer, nous avons besoin de définir le radical d'un idéal.

Définition 3 (Radical d'un idéal)**.** *Si I est un idéal d'un anneau A, le radical de I est un idéal défini par*

$$\sqrt{I} = \{a \in A \mid a^r \in I \text{ pour un certain entier } r > 0\}$$

l'ensemble des éléments de l'anneau dont une puissance appartient à I. On a donc toujours $I \subseteq \sqrt{I}$. Un idéal égal à son radical est appelé un idéal radical (on dit aussi idéal radiciel).

Exemple 7. $\mathbf{I}(X)$ *est un idéal radical pour tout ensemble de points* $X \subseteq \overline{\mathbb{K}}^n$.

Théorème 1 (Nullstellensatz ou Théorème des zéros de Hilbert)**.** *Soit* $\mathbb{K}$ *un corps algébriquement clos. Alors pour tout idéal* I *de* $\mathbb{K}[x_1, \ldots, x_n]$,

$$\mathbf{I}(\mathbf{V}(I)) = \sqrt{I}$$

En particulier, on obtient que $\mathbf{I}(\mathbf{V}(I)) = I$ si et seulement si I est radical. Le Nullstellensatz montre donc qu'on a une bijection entre les ensembles algébriques affines et les idéaux radicaux de $\mathbb{K}[x_1, \ldots, x_n]$. L'hypothèse sur la clôture algébrique de $\mathbb{K}$ est nécessaire comme le montre l'exemple suivant.

Exemple 8. $I = \langle x^2 + 1 \rangle \subset \mathbb{R}[x]$ *est un idéal radical, mais* $\mathbf{V}_{\mathbb{R}}(I) = \emptyset$ *d'où* $\mathbf{I}(\mathbf{V}(I)) = \mathbb{R}[x] \neq I$.

Le corollaire suivant concerne les idéaux zéros dimensionnels.

Corollaire 1. *Soit* $\mathbb{K}$ *un corps algébriquement clos et soit* I *un idéal de* $\mathbb{K}[x_1, \ldots, x_n]$. *Alors* $\mathbf{V}(I)$ *est fini si et seulement si* $\mathbb{K}[x_1, \ldots, x_n]/I$ *est un espace vectoriel de dimension finie sur* $\mathbb{K}$. *Dans ce cas, le nombre de points de* $\mathbf{V}(I)$ *est au plus* $\dim_{\mathbb{K}}(\mathbb{K}[x_1, \ldots, x_n]/I)$. *De plus, si* I *est radical, le nombre de points de* $\mathbf{V}(I)$ *est exactement égal à* $\dim_{\mathbb{K}}(\mathbb{K}[x_1, \ldots, x_n]/I)$.

Démonstration. Supposons $\mathbf{V}(I) = \{p_1, \ldots, p_m\}$ fini et notons leurs coordonnées par $p_i = (a_{i,1}, \ldots, a_{i,n})$. Posons n polynômes définis par

$$F_j = \prod_{i=1}^{m} (x_j - a_{i,j}),\ 1 \leq j \leq n$$

et notons $[F_i]$ leur classe dans $\mathbb{K}[x_1, \ldots, x_n]/I$. Par construction, les F_j s'annulent en tout point de $\mathbf{V}(I)$ et donc $F_j \in \mathbf{I}(\mathbf{V}(I))$. D'après le Nullstellensatz, $F_j \in \sqrt{I}$. Il existe donc un entier $r > 0$ tel que $F_j^r \in I$ pour tout j. Donc $[F_j]^r = 0$ dans $\mathbb{K}[x_1, \ldots, x_n]/I$, ce qui veut dire que, pour tout j, $[x_j]^{rm}$ est une combinaison linéaire de $[1], [x_j], \ldots, [x_j]^{rm-1}$.

Réciproquement, soient des points distincts $p_1, \ldots, p_m$ de $\mathbf{V}(I)$. Prenons des polynômes $F_1, \ldots, F_m \in \mathbb{K}[x_1, \ldots, x_n]$ tels que $F_i(p_i) = 1$ et $F_i(p_j) = 0$ pour $i \neq j$. Si $\sum \lambda_i [F_i] = 0, \lambda_i \in \mathbb{K}$, alors $\sum \lambda_i F_i \in I$, et donc $\lambda_j = \sum \lambda_i F_i(p_j) = 0$. Les $[F_i]$ sont donc linéairement indépendants dans $\mathbb{K}[x_1, \ldots, x_n]/I$ et on a alors $m \leq \dim_{\mathbb{K}}(\mathbb{K}[x_1, \ldots, x_n]/I)$.

Finalement, supposons I radical et montrons que $[F_1], \ldots, [F_m]$ engendre $\mathbb{K}[x_1, \ldots, x_n]/I$. Soit $[G] \in \mathbb{K}[x_1, \ldots, x_n]/I$ et posons $\lambda_i = G(p_i)$. On a alors $[G - \sum_{i=1}^m \lambda_i F_i] = 0$ d'où $[G] = \sum_{i=1}^m \lambda_i [F_i]$. □

Le Nullstellensatz et le corollaire 1 tels qu'ils ont été énoncés, nécessitent d'avoir un corps algébriquement clos. Or dans la suite de la thèse, on s'intéressera seulement aux solutions dans un corps fini, généralement dans $\mathbb{F}_2$ pour la cryptanalyse algébrique. On a donc besoin d'une version du théorème des zéros de Hilbert sur les corps finis.

Étant donné un système S à coefficients dans un corps fini $\mathbb{F}_q$, il suffit de rajouter les équations de corps $x_i^q - x_i = 0$ pour se limiter uniquement aux solutions du système dans $\mathbb{F}_q$. En effet, les zéros du polynôme $X^q - X \in \mathbb{F}_q[X]$ sont exactement les éléments du corps $\mathbb{F}_q$. L'ensemble algébrique affine défini par $I = \langle S \rangle = \langle \{f_1, \ldots, f_m\} \rangle$ sur $\mathbb{F}_q$ vérifie donc

$$\mathbf{V}_{\overline{\mathbb{F}_q}}(I + \langle x_1^q - x_1, \ldots, x_n^q - x_n \rangle) = \mathbf{V}_{\overline{\mathbb{F}_q}}(I) \cap \mathbb{F}_q^n = \mathbf{V}_{\mathbb{F}_q}(I) \tag{2.1}$$

On considère donc maintenant l'idéal $J = I + \langle x_1^q - x_1, \ldots, x_n^q - x_n \rangle = \langle f_1, \ldots, f_m, x_1^q - x_1, \ldots, x_n^q - x_n \rangle$ engendré par les équations de S ainsi que les équations de corps. Les points de $\mathbf{V}_{\overline{\mathbb{F}_q}}(J)$ appartiennent forcément au corps fini $\mathbb{F}_q$ et l'idéal J est donc un idéal zéro dimensionnel. De plus, on peut montrer que l'idéal J est radical :

Lemme 1. *Pour tout corps fini $\mathbb{F}_q$ et pour tout idéal $I \subseteq \mathbb{F}_q[x_1, \ldots, x_n]$, l'ideal $J = I + \langle x_1^q - x_1, \ldots, x_n^q - x_n \rangle$ est radical.*

Démonstration. Par définition, tout idéal est contenu dans son radical. On doit donc seulement montrer l'inclusion $\sqrt{J} \subseteq J$. Soit $p \in \sqrt{J}$. Il existe un entier $r > 0$ tel que $p^r \in J$. Or avec les équations de corps, pour tout $g \in J = I + \langle x_1^q - x_1, \ldots, x_n^q - x_n \rangle$ on a $g^q = g$. Sans perte de généralité, on peut supposer $r < q$ (sinon prendre $q^m > r$). On a alors

$$p = p^q = p^{q-r} - p^r \in J$$

. □

Théorème 2 (Nullstellensatz sur un corps fini)**.** *Pour tout idéal $I \subseteq \mathbb{F}_q[x_1, \ldots, x_n]$, on a*

$$\mathbf{I}(\mathbf{V}_{\mathbb{F}_q}(I)) = I + \langle x_1^q - x_1, \ldots, x_n^q - x_n \rangle$$

Démonstration. On applique le Nullstellensatz sur $\overline{\mathbb{F}_q}$ en utilisant le fait que l'idéal $J = I + \langle x_1^q - x_1, \ldots, x_n^q - x_n \rangle$ est radical, ce qui nous donne $\mathbf{I}(\mathbf{V}_{\overline{\mathbb{F}_q}}(J)) = J$. Or, on a $\mathbf{V}_{\overline{\mathbb{F}_q}}(J) = \mathbf{V}_{\mathbb{F}_q}(I)$ (cf. équation (2.1)), d'où le résultat. □

Ce qui nous donne aussi une version du corollaire 1 sur les corps finis.

Corollaire 2. *Soit I un idéal de $\mathbb{F}_q[x_1, \ldots, x_n]$. Le nombre de points de $\mathbf{V}_{\mathbb{F}_q}(I)$ est égal à $\dim_{\mathbb{F}_q}(\mathbb{F}_q[x_1, \ldots, x_n]/I + \langle x_1^q - x_1, \ldots, x_n^q - x_n\rangle)$.*

Les principaux résultats sur les variétés et les idéaux qui nous sont nécessaires ont été présentés. On peut maintenant définir les bases de Gröbner et énoncer les propriétés qui permettent d'obtenir une méthode de résolution de systèmes polynomiaux.

2.2.2 Bases de Gröbner

Dans le cas des polynômes à une seule variable, l'anneau $\mathbb{K}[x]$ est euclidien. On peut donc y définir une division euclidienne et bénéficier ainsi des méthodes d'arithmétique élémentaire, comme par exemple, pouvoir calculer le pgcd et ppcm de polynômes, donc calculer un générateur principal d'un idéal, résoudre le problème de l'appartenance à un idéal, calculer dans l'espace $\mathbb{K}[x]/I$, etc.

En revanche, l'anneau $\mathbb{K}[x_1, \ldots, x_n]$ des polynômes en $n > 1$ variables n'est pas principal. On ne peut donc plus définir naturellement la division et l'algorithme d'Euclide dans cet anneau. Plus généralement, on n'a plus de représentants canoniques des éléments de $\mathbb{K}[x_1, \ldots, x_n]/I$. Les bases de Gröbner vont apporter une réponse à ces problèmes, notamment en permettant de calculer modulo un idéal I.

On appelle *monôme* de degré d un produit de d variables et *terme* un monôme multiplié par un élément de $\mathbb{K}$, appelé son *coefficient*. Un polynôme est donc une somme de termes et son degré est le maximum des degrés des monômes qui le constituent. On commence par ordonner les monômes suivant un ordre total.

Définition 4 (Ordre monomial)**.** *Un ordre monomial $\prec$ est un ordre total sur l'ensemble des monômes de $\mathbb{K}[x_1, \ldots, x_n]$ tel que*

1. *si un monôme m est non constant, alors $1 \prec m$*
2. *si m_1, m_2, m_3 sont des monômes, alors $m_1 \prec m_2 \Rightarrow m_1 m_3 \prec m_2 m_3$*

Exemple 9. *Les ordres suivants sont des ordres monomiaux.*

- *L'ordre lexicographique : $x_1^{\alpha_1} \cdots x_n^{\alpha_n} \prec_{lex} x_1^{\beta_1} \cdots x_n^{\beta_n}$ si $\exists\ 1 \leq i \leq n$ tel que $\alpha_i < \beta_i$ et $\alpha_j = \beta_j$ pour tout $j < i$.*
- *L'ordre gradué lexicographique : $m_\alpha \prec_{glex} m_\beta$ si $\deg(m_\alpha) < \deg(m_\beta)$, ou bien, quand $\deg(m_\alpha) = \deg(m_\beta)$, si $m_\alpha \prec_{lex} m_\beta$*
- *L'ordre gradué lexicographique inverse : $m_\alpha = x_1^{\alpha_1} \cdots x_n^{\alpha_n} \prec_{grevlex} m_\beta = x_1^{\beta_1} \cdots x_n^{\beta_n}$ si $\deg(m_\alpha) < \deg(m_\beta)$, ou bien quand $\deg(m_\alpha) = \deg(m_\beta)$, si $\exists\ 1 \leq i \leq n$ tel que $\alpha_i > \beta_i$ et*

$\alpha_j = \beta_j$ pour tout $j > i$ (i.e. si $m_\alpha \succ_{lex} m_\beta$ pour l'ordre lexicographique dans $\mathbb{K}[x_n, \ldots, x_1]$).

On dit que les ordres glex (pour Graded Lexicographical) et grevlex (pour Graded Reverse Lexicographical) sont des ordres gradués car l'agencement s'effectue prioritairement sur le degré.

Les notions de *monôme dominant*, *coefficient dominant* et *terme dominant* (noté $\mathrm{LT}_\prec(f)$) d'un polynôme f, relativement à un ordre monomial, sont ainsi étendues aux polynômes multivariés.

Pour tout idéal I, on peut maintenant définir l'idéal des termes dominants de I.

Définition 5. *Soit $I = \langle f_1, \ldots, f_m \rangle \subset \mathbb{K}[x_1, \ldots, x_n]$ un idéal. On définit l'ensemble $LT_\prec(I)$ des termes dominants de l'idéal I par*

$$LT_\prec(I) = \{LT_\prec(f) \mid f \in I \backslash 0\}.$$

L'idéal des termes dominants $\langle LT_\prec(I) \rangle$ est défini comme l'idéal engendré par les éléments de $LT_\prec(I)$.

On remarque qu'on a l'inclusion $\langle \mathrm{LT}_\prec(f_1), \ldots, \mathrm{LT}_\prec(f_m) \rangle \subset \langle \mathrm{LT}_\prec(I) \rangle$, mais généralement il n'y a pas égalité comme le montre l'exemple suivant.

Exemple 10. *Soit $I = \langle f_1, f_2 \rangle \subset \mathbb{K}[x_1, x_2]$ où $f_1 = x_1 + x_2$ et $f_2 = x_1 - x_2$, avec l'ordre $\prec_{lex}$ (i.e. $x_1 \succ x_2$). Étant donné que $f_1 - f_2 = 2x_2 \in I$, on a $2x_2 \in \langle LT(I) \rangle$. D'un autre coté, $LT(f_1) = LT(f_2) = x_1$ et $2x_2 \notin \langle LT(f_1), LT(f_2) \rangle = \langle x_1 \rangle$.*

Pour un ordre monomial donné, on donne le nom de base de Gröbner à un ensemble de polynômes qui est tel que cette égalité est vérifiée, i.e. tel que l'idéal des termes dominants $\langle \mathrm{LT}_\prec(I) \rangle$ peut être engendré par les termes dominants de chacun des polynômes de la base :

Définition 6 (Base de Gröbner)**.** *Un sous-ensemble fini $\{g_1, \ldots, g_s\}$ d'un idéal I est appelé une base de Gröbner pour un ordre monomial $\prec$ si*

$$\langle LT_\prec(g_1), \ldots, LT_\prec(g_s) \rangle = \langle LT_\prec(I) \rangle.$$

Autrement dit, $\{g_1, \ldots, g_s\} \subset I$ est une base de Gröbner de I si et seulement si pour tout $f \in I$, $\mathrm{LT}(f)$ est divisible par un des $\mathrm{LT}(g_i)$.

Étant donné un ordre monomial, on peut montrer que tout idéal admet une base de Gröbner :

Proposition 2. *Soit $I \subset \mathbb{K}[x_1, \ldots, x_n]$ un idéal et soit $\prec$ un ordre monomial. Alors*

1. *il existe une base de Gröbner G de I relativement à l'ordre $\prec$*
2. *une base de Gröbner de I est une base de I (i.e. forme un ensemble générateur de I).*

Généralement, une base de Gröbner d'un idéal n'est pas unique, mais on peut rajouter les conditions suivantes pour garantir l'unicité :

Définition 7. *Une base de Gröbner réduite d'un idéal I est une base de Gröbner G telle que*

- *pour tout $g \in G$, le coefficient du terme dominant de g est égal à 1,*
- *pour tout $g \in G$, aucun monômes de g n'appartient à $\langle\ LT(G \backslash \{g\})\ \rangle$*

On a alors que tout idéal de $\mathbb{K}[x_1, \ldots, x_n]$ admet une unique base de Gröbner réduite pour un ordre monomial donné ([CLO07]).

Le théorème d'élimination suivant indique qu'une base de Gröbner pour l'ordre lexicographique possède des propriétés intéressantes pour résoudre un système polynomial :

Théorème 3 (Théorème d'Élimination)**.** *Soient un idéal $I \subset \mathbb{K}[x_1, \ldots, x_n]$ et G une base de Gröbner de I pour l'ordre monomial lexicographique où $x_1 \prec x_2 \prec \cdots \prec x_n$. Alors, pour tout $0 \leq l < n$, l'ensemble*

$$G_l = G \cap \mathbb{K}[x_{l+1}, \ldots, x_n]$$

est une base de Gröbner de l'idéal $I_l = I \cap \mathbb{K}[x_{l+1}, \ldots, x_n]$.

On remarque que $I_0 = I$ et que, pour $l = n - 1$, l'idéal d'élimination $I_{n-1} = I \cap \mathbb{K}[x_n]$ est constitué uniquement de polynômes univariés en x_n.

De plus, on peut montrer que l'idéal I est zéro-dimensionnel si et seulement si pour tout $i \in \{1, \ldots, n\}$, il existe un polynôme dans la base de Gröbner de I dont le terme dominant est une puissance de x_i (quelque soit l'ordre monomial fixé, voir [CLO07] Chap. 5 § 3 Th. 6). D'après le théorème d'élimination et pour un système zéro-dimensionnel, la base de Gröbner réduite $G = \{g_1, \ldots, g_s\}$ pour l'ordre lexicographique a donc une forme triangulaire

par bloc :

$$\begin{array}{r} g_1(x_1, x_2, \ldots, x_{n-1}, x_n) \\ \vdots \\ g_i(x_1, x_2, \ldots, x_{n-1}, x_n) \\ g_{i+1}(x_2, \ldots, x_{n-1}, x_n) \\ \vdots \\ \vdots \\ g_{s-j}(x_{n-1}, x_n) \\ \vdots \\ g_{s-1}(x_{n-1}, x_n) \\ g_s(x_n) \end{array}$$

À partir de cette base, on peut alors calculer l'ensemble des solutions du système en résolvant équation par équation.

2.2.3 Algorithmes de calcul de base de Gröbner

On a défini les bases de Gröbner et on a montré qu'elles apportaient une méthode pour la résolution de systèmes polynomiaux zéro-dimensionnels. Mais on n'a pas encore proposé de méthode pour construire en pratique une base de Gröbner d'un idéal donné. Dans cette partie, on va donc décrire des algorithmes de calcul de base de Gröbner, en commençant par l'algorithme de Buchberger, historiquement le premier a avoir été proposé ([Buc65]).

Pour ce faire, on a besoin de la notion de réduction d'un polynôme qui permet de formuler une généralisation de la division dans $\mathbb{K}[x_1, \ldots, x_n]$.

Définition 8. *Soient $f, g \in \mathbb{K}[x_1, \ldots, x_n]$ et soit $\prec$ un ordre monomial. On dit que f est réductible par g s'il existe un terme t de f qui soit divisible par le terme dominant de g. On notera alors $f \xrightarrow{g} r$ où $r = f - \frac{t}{LT_\prec(g)} g$.*

Par extension, f se réduit à r modulo $G \subset \mathbb{K}[x_1, \ldots, x_n]$ si $\exists\ g \in G$ tel que $f \xrightarrow{g} r$. Le reste de la division (ou forme normale) de f par G est un polynôme r obtenu après un nombre fini de réductions de f par G et tel que r ne soit plus réductible par G. En général, la forme normale d'un polynôme par un ensemble G de polynômes dépend de l'ordre dans lequel on effectue les réductions, et il n'y a donc pas unicité du reste. Mais lorsque G est une base de Gröbner, on a alors la propriété suivante.

Proposition 3. *Soit $G = \{g_1, \ldots, g_s\}$ une base de Gröbner d'un idéal $I \subset \mathbb{K}[x_1, \ldots, x_n]$. Alors pour tout $f \in \mathbb{K}[x_1, \ldots, x_n]$, il existe un unique*

polynôme $r \in \mathbb{K}[x_1, \ldots, x_n]$ tel que f se réduise à r modulo G. L'ordre dans lequel on choisit les g_i n'a pas d'importance. En particulier, on a $f \in I$ si et seulement si sa forme normale r par G est égale à 0.

Les bases des Gröbner permettent donc de calculer modulo un idéal, le représentant d'un polynôme modulo l'idéal est alors sa forme normale.

Définition 9 (S-polynôme)**.** *Soient $f, g \in \mathbb{K}[x_1, \ldots, x_n]$. On appelle S-polynôme de f et g le polynôme*

$$Spol(f,g) = \frac{\mu}{LT_{\prec}(f)} f - \frac{\mu}{LT_{\prec}(g)} g$$

où μ est le plus petit commun multiple des monômes dominants de f et g.

L'algorithme de Buchberger découle des S-polynômes et de la proposition suivante.

Proposition 4. *Soit $I \subset \mathbb{K}[x_1, \ldots, x_n]$ un idéal. Un ensemble générateur $G = \{g_1, \ldots, g_s\}$ de I est une base de Gröbner de I pour l'ordre monomial $\prec$ si et seulement si pour tout $i, j \in \{1, \ldots, s\}$, on a $Spol(g_i, g_j) \xrightarrow{G} 0$.*

À partir de cette proposition, on déduit l'algorithme 1 qui permet de construire une base de Gröbner à partir d'un ensemble générateur d'un idéal.

Algorithme 1: Algorithme de Buchberger [Buc65]

Entrées : Un ordre monomial $\prec$ et des polynômes $\{f_1, \ldots, f_r\} \subset \mathbb{K}[x_1, \ldots, x_n]$

Sortie : Une base de Gröbner $G = \{g_1, \ldots, g_s\}$ de l'idéal $I = \langle f_1, \ldots, f_r \rangle$.

```
1 G := {f_1, ..., f_r};
2 répéter
3 |   G' := G;
4 |   pour tous les g_i, g_j ∈ G' faire
5 |   |   Spol(g_i, g_j) --G'--> r;
6 |   |   si r ≠ 0 alors G := G ∪ {r};
7 |   fin
8 jusqu'à G := G';
9 retourner G;
```

La preuve de la terminaison de l'algorithme de Buchberger utilise le lemme de Dickson (voir par exemple [CLO07]), qui assure que la suite croissante des idéaux $\langle \mathrm{LT}(G) \rangle$ est stationnaire. Buchberger a aussi introduit plusieurs critères pour améliorer l'algorithme 1 ([Buc65]). L'idée principale est d'éviter un certain nombre de calculs inutiles en prévoyant les réductions à zéro des S-polynômes (voir [CLO07]).

Après avoir établit un lien entre algèbre des polynômes et algèbre linéaire, on peut montrer qu'un calcul de base de Gröbner peut se ramener à des calculs d'algèbre linéaire. Dans ce but, on introduit les matrices de Macaulay ([Mac02]), qui représentent les polynômes d'un idéal qui sont inférieurs à un degré donné.

Définition 10 (Matrice de Macaulay)**.** *Soient un idéal $I = \langle f_1, \ldots, f_r \rangle \subset \mathbb{K}[x_1, \ldots, x_n]$ et un ordre monomial $\prec$. La matrice de Macaulay en degré D de I est une matrice dont les colonnes sont indexées par les monômes de degré inférieur ou égal à D et les entrées sont les coefficients de tous les multiples tf_i de degré inférieur ou égal à D pour tout $i \in \{1, \ldots, r\}$, pour tout monôme t de degré $D - deg(f_i)$.*

$$Macaulay_D(\{f_1, \ldots, f_r\}) = \begin{array}{c} \\ \vdots \\ tf_i \\ \vdots \end{array} \begin{array}{c} \overset{\text{monômes de degré } \leq D}{} \\ \begin{pmatrix} \\ \text{coeff}(tf_i) \\ \\ \end{pmatrix} \end{array}$$

Tout polynôme de degré $\leq D$ de l'idéal I est donc représenté par une combinaison linéaire des lignes de cette matrice. Un calcul de base de Gröbner revient alors à appliquer sur une matrice de Macaulay (pour un degré suffisamment grand) un algorithme d'élimination de Gauss où la seule opération élémentaire autorisée est l'addition d'une ligne et d'une combinaison linéaire des lignes précédentes (la permutation de lignes ou de colonnes n'est pas autorisée). La matrice obtenue est dite sous forme échelon et on a le théorème suivant :

Théorème 4 (Lazard [Laz83])**.** *Soit un idéal $I = \langle f_1, \ldots, f_r \rangle \subset \mathbb{K}[x_1, \ldots, x_n]$. Il existe alors un entier $D \geq 0$ tel que les polynômes correspondant aux lignes de la forme échelonnée de la matrice de Macaulay de degré D forment une base de Gröbner de I.*

Remarque 1. *Une matrice de Macaulay n'est généralement pas de rang plein. En effet, les lignes de cette matrice ne sont pas indépendantes et on*

obtient donc de nombreuses lignes nulles après réduction sous forme échelonnée. Calculer une base de Gröbner à partir de cette matrice de Macaulay n'est pas efficace en pratique.

L'algorithme F_4 [Fau99] proposé par J.-C. Faugère est un algorithme de calcul de bases de Gröbner utilisant des outils d'algèbre linéaire. C'est l'algorithme de résolution par base de Gröbner que nous utilisons dans cette thèse et plus précisément son implémentation efficace disponible dans le logiciel de calcul formel MAGMA [BCP97]. Une description très simplifiée de l'algorithme F_4 est donnée par l'algorithme 2.

Algorithme 2: Algorithme F_4 (version simplifiée) [Fau99]

Entrées : F un sous-ensemble fini de $\mathbb{K}[x_1, \ldots, x_n]$
Sel une fonction de sélection de paires critiques.
Sortie : G un sous-ensemble fini de $\mathbb{K}[x_1, \ldots, x_n]$

```
1  G := F;
2  d := 0;
3  P := {(f, g) | f, g ∈ G, f ≠ g};
4  tant que P ≠ ∅ faire
5  |  d := d + 1;
6  |  P_d := Sel(P);
7  |  P := P \ P_d;
8  |  F_d := Reduction(P_d, G);
9  |  pour chaque h ∈ F_d faire
10 |  |  P := P ∪ {(h, g) | g ∈ G};
11 |  |  G := G ∪ {h};
12 |  fin
13 fin
14 retourner G;
```

Contrairement à l'algorithme de Buchberger qui ne traite qu'une seule paire critique à la fois, l'algorithme F_4 fait intervenir une méthode **Reduction** qui peut efficacement réduire un ensemble de paires simultanément. Sans décrire en détails cette fonction, notons simplement que la réduction simultanée est effectuée par le calcul de la forme échelon d'une matrice de type Macaulay qu'on souhaite la plus petite et la plus creuse possible (voir [Fau99] pour plus de détails). Cependant, cette matrice n'est généralement pas de rang plein, ce qui veut dire que des réductions à zéro sont encore calculées. Ces réductions à zéro inutiles peuvent être évitées par un critère de

sélection donné dans l'algorithme F_5 qui permet de construire des matrices dont la taille est optimale (voir [Fau02]).

La fonction Sel permet de sélectionner les paires critiques de la liste P. Cette fonction doit vérifier que si $P \neq \emptyset$ alors $Sel(P) \neq \emptyset$. On remarque que si la fonction $Sel(P)$ renvoie toujours une seule paire quelque soit P, alors la stratégie de sélection des paires critiques est celle de l'algorithme de Buchberger. La *stratégie normale pour l'algorithme* F_4 est la stratégie consistant à choisir les paires de plus petit degré. Dans ce cas, le calcul des bases de Gröbner est effectué de manière incrémentale, en considérant successivement des degrés croissants jusqu'à atteindre un degré suffisamment grand pour obtenir la base de Gröbner complète. De plus, le plus grand degré atteint lors du calcul fournit une borne sur le nombre de répétitions de la boucle ([4]-[13]) de l'algorithme F_4 et donc sur la complexité de calcul d'une base de Gröbner.

Pour un ordre du degré, le degré maximal atteint lors du calcul d'une base de Gröbner est borné par ce qu'on appelle le degré de régularité d'un idéal I :

Définition 11. *Le degré de régularité d'un idéal* $I = \langle f_1, \ldots, f_r \rangle$ *de* $\mathbb{K}[x_1, \ldots, x_n]$ *est donné par*

$$d_{reg} = \min \left\{ d \geq 0 \mid \dim_{\mathbb{K}}(\{f \in I^H, \deg(f) = d\}) = \binom{n+d-1}{n} \right\},$$

où I^H *est l'idéal engendré par les homogénéisés des polynômes* $f_1, \ldots, f_r$.

Le nombre d'opérations dans $\mathbb{K}$ nécessaires pour calculer une base de Gröbner peut alors être borné par le coût de la réduction de la matrice de Macaulay de degré d_{reg} (théorème 4). Le nombre de colonnes de cette matrice est égal au nombre de monômes de degré $\leq d_{reg}$, c'est-à-dire $\binom{n+d_{reg}}{d_{reg}}$. On obtient donc la proposition suivante :

Proposition 5. *Soit* I *un idéal de* $\mathbb{K}[x_1, \ldots, x_n]$. *Le nombre d'opérations dans* $\mathbb{K}$ *nécessaires pour calculer une base de Gröbner est*

$$\mathcal{O}\left(\binom{n+d_{reg}}{d_{reg}}^{\omega}\right)$$

où d_{reg} *est le degré de régularité de l'idéal* I *et* ω *est la constante d'algèbre linéaire (*$2 \leq \omega \leq 3$*).*

Le degré de régularité d'un idéal fournit donc un moyen d'apprécier la complexité de calcul d'une base de Gröbner. Malheureusement, il est difficile en général de calculer le degré de régularité d'un système donné.

Dans l'analyse des attaques algébriques par canaux auxiliaires (chapitre 6), on montre que l'étude locale des boîtes-S avec leurs informations de fuite permet de fortement linéariser le système d'équations modélisant le cryptosystème. Le grand nombre d'équations linéaires ainsi obtenues permet d'avoir un degré de régularité qui reste petit, ce qui limite le degré à atteindre lors des calculs de base de Gröbner.

2.3 Résolution par SAT solveurs

Dans le cas particulier de la résolution des systèmes polynomiaux définis sur $\mathbb{F}_2$, l'utilisation de SAT solveurs est une autre méthode particulièrement efficace en pratique. Les SAT solveurs sont des logiciels implantant des algorithmes pour le problème de la satisfiabilité (problème SAT en abrégé), i.e. pour décider si une formule propositionnelle est satisfaisable ou non. Or, une équation à coefficients dans $\mathbb{F}_2$ peut être représentée de manière équivalente par une formule booléenne. Ainsi résoudre un système d'équations dans $\mathbb{F}_2$ peut se ramener à trouver une solution du problème SAT correspondant.

2.3.1 Satisfiabilité d'une forme normale conjonctive (CNF)

Soit $\mathscr{P}$ un ensemble de symboles, appelés variables propositionnelles, auquel on associe l'ensemble des applications $\mathscr{P} \to \{\text{vrai, faux}\}$.

Définition 12. *Une application $\mathscr{P} \to \{vrai, faux\}$ est appelée une distribution sur $\mathscr{P}$.*

On définit une formule propositionnelle comme étant un élément de l'ensemble suivant :

Définition 13. *L'ensemble des formules propositionnelles sur $\mathscr{P}$ est le plus petit ensemble tel que :*

- *une variable propositionnelle de $\mathscr{P}$ est une formule,*
- *si P et Q sont des formules, alors $\neg P$ et $(P \wedge Q)$ sont des formules.*

Les distributions sur $\mathscr{P}$ s'étendent à l'ensemble des formules propositionnelles sur $\mathscr{P}$ en posant les valeurs de vérité (figure 2.1) attribuées à ces formules.

Bien que cet ensemble soit complet, c'est-à-dire que toute fonction booléenne peut être représentée par une formule de cet ensemble, on rajoute généralement les connecteurs usuels suivants :

P	$\neg P$
vrai	faux
faux	vrai

P	Q	$(P \wedge Q)$
vrai	vrai	vrai
vrai	faux	faux
faux	vrai	faux
faux	faux	faux

FIGURE 2.1 – Valeurs de vérité pour la négation et la conjonction.

Définition 14. *Soient P et Q des formules propositionnelles. Alors on définit*

- $(P \vee Q)$ *par* $\neg(\neg P \wedge \neg Q)$,
- $(P \Rightarrow Q)$ *par* $(\neg P \vee Q)$,
- $(P \Leftrightarrow Q)$ *par* $(P \wedge Q) \vee (\neg P \wedge \neg Q)$,
- $(P \oplus Q)$ *par* $((\neg P \wedge Q) \vee (P \wedge \neg Q))$

P	Q	$(P \vee Q)$
vrai	vrai	vrai
vrai	faux	vrai
faux	vrai	vrai
faux	faux	faux

P	Q	$(P \Rightarrow Q)$
vrai	vrai	vrai
vrai	faux	faux
faux	vrai	vrai
faux	faux	vrai

P	Q	$(P \Leftrightarrow Q)$
vrai	vrai	vrai
vrai	faux	faux
faux	vrai	faux
faux	faux	vrai

P	Q	$(P \oplus Q)$
vrai	vrai	faux
vrai	faux	vrai
faux	vrai	vrai
faux	faux	faux

FIGURE 2.2 – Valeurs de vérité pour la disjonction, l'implication, l'équivalence et le ou-exclusif.

Les tables de vérité correspondantes sont données par la figure 2.2.

En pratique, on omettra les parenthèses inutiles dans les formules propositionnelles, c'est-à-dire lorsque leur absence ne provoque pas d'ambiguïté.

L'interprétation des formules propositionnelles conduit à identifier celles qui ont les mêmes tables de vérités :

Définition 15. *Deux formules propositionnelles f et g sont logiquement équivalentes si pour toute distribution ϕ, on a $\phi(f) = \phi(g)$.*

Il existe différentes situations d'interprétations. On s'intéresse plus particulièrement ici au cas de la satisfiabilité :

Définition 16. *Une formule propositionnelle f est dite satisfiable, ou satisfaisable, s'il existe une distribution ϕ telle que $\phi(f) = vrai$.*

Le problème SAT nécessite généralement qu'une formule propositionnelle ait une structure particulière. Pour expliciter cette structure, nous avons préalablement besoin des définitions d'un littéral et d'une clause :

Définition 17. *Une variable propositionnelle x_i et sa négation $\neg x_i$ sont appelées des littéraux. Une clause c est une disjonction de d littéraux :*

$$c = l_1 \vee l_2 \vee \cdots \vee l_d.$$

Définition 18. *Une formule f est dite en forme normale conjonctive (CNF) si c'est une conjonction de clauses, i.e. une conjonction de disjonction de littéraux :*

$$f = (l_1 \vee l_2 \vee \cdots \vee l_d) \wedge (l'_1 \vee \dots) \wedge \dots.$$

D'après le théorème suivant, le problème SAT peut se limiter à considérer des formules en CNF sans perte de généralité :

Théorème 5. *Toute formule propositionnelle est équivalente à une formule en forme normale conjonctive.*

On remarque qu'une formule en CNF est satisfaite si et seulement si chacune de ses clauses est satisfaite. Ainsi quand on cherche à savoir si une formule est satisfiable, il peut être intéressant de chercher d'abord une formule équivalente en CNF. D'ailleurs, la plupart des SAT solveurs ne prennent en entrée que des formules en CNF.

Pour utiliser ces SAT-solveurs comme méthode de résolution, nous devons donc d'abord convertir les systèmes polynomiaux à résoudre dans $\mathbb{F}_2$ en une formule propositionnelle en CNF. La conversion est telle que les solutions du système de départ correspondent aux distributions qui satisfont la formule en CNF obtenue.

2.3.2 Conversion d'un système polynomial en CNF

Seules les solutions dans $\mathbb{F}_2$ nous intéresseront lorsque l'on cherchera à résoudre des systèmes polynomiaux dans l'anneau $\mathbb{F}_2[X_1, \dots, X_n]$. On ajoutera donc toujours les équations de corps $X_i^2 + X_i = 0$ $(1 \leq i \leq n)$ aux systèmes d'équations. Cela revient à considérer que les équations polynomiales sont

de la forme $P(X_1, \ldots, X_n) = 0$ où P est un élément de l'anneau quotient $\mathbb{F}_2[X_1, \ldots, X_n]/(X_1^2 + X_1, \ldots, X_n^2 + X_n)$. De plus, P peut être représenté de manière unique par sa forme algébrique normale

$$P(X_1, \ldots, X_n) = \sum_{S_i \subseteq \{1,\ldots,n\}} \prod_{j \in S_i} X_j$$

Partant de cette représentation, on peut naturellement associer au polynôme P une formule propositionnelle f sur $\mathscr{P} = \{x_1, \ldots, x_n\}$. En effet, faisons correspondre 0 avec faux, 1 avec vrai et les variables X_i de l'anneau aux variables propositionnelles x_i. La somme dans $\mathbb{F}_2$ correspond alors au ou-exclusif $\oplus$ et le produit à la conjonction $\wedge$. On a donc que f est de la forme

$$f = m_1 \oplus m_2 \oplus \cdots \oplus m_k$$

où m_i est la conjonction des variables correspondant au i-ème monôme.

Exemple 11. *Au polynôme $P(X_1, X_2, X_3) = X_1X_2 + X_2X_3$, on peut associer la formule propositionnelle $f = ((x_1 \wedge x_2) \oplus (x_2 \wedge x_3))$ sur $\mathscr{P} = \{x_1, x_2, x_3\}$.*

Les solutions de l'équation polynomiale $P = 0$ correspondent alors aux distributions qui satisfont $\neg f$. Malheureusement, la formule propositionnelle $\neg f$ ainsi obtenue n'est pas en CNF. Afin de pouvoir utiliser un SAT-solveur, il est alors nécessaire de trouver une formule équivalente en CNF.

Nous rappelons une méthode, proposée dans [CB07], afin de convertir efficacement un système polynomial dans $\mathbb{F}_2$ directement en CNF. On avait déjà remarqué qu'une formule en CNF peut être vue comme un ensemble de clauses à satisfaire simultanément. Pour transformer l'intégralité d'un système d'équations, on peut alors se contenter de transformer chaque équation en un ensemble de clauses.

Soit un polynôme $P \in \mathbb{F}_2[X_1, \ldots, X_n]/(X_1^2 + X_1, \ldots, X_n^2 + X_n)$ en forme algébrique normale. Pour convertir l'équation $P = 0$ en CNF, on commence par simplifier le problème en linéarisant P, i.e. on remplace chaque monôme de degré > 1 par une nouvelle variable M_i. On est donc ramené à deux sous-problèmes plus simples :

1. convertir l'équation linéaire ainsi obtenue en un ensemble de clauses,
2. pour chaque nouvelle variable M_i, produire les clauses qui la décrive.

Pour convertir l'équation linéaire de l'étape 1, il suffit de remarquer que $\sum_i^k M_i = 0$ si et seulement si le nombre de variables M_i non nulles est pair. Ainsi, on ne peut jamais annuler toutes les variables lorsqu'on en inverse un nombre impair. Inversement, $\sum_i^k M_i = 1$ si et seulement si le nombre de

variables M_i non nulles est impair, et donc on ne pourra jamais annuler toutes les variables lorsqu'on en inverse un nombre pair. Une équation linéaire en k variables est donc convertie en un ensemble de $\sum_{i=0}^{\lfloor k/2 \rfloor} \binom{k}{2i+1} = \sum_{i=0}^{\lfloor k/2 \rfloor} \binom{k}{2i} = 2^{k-1}$ clauses.

Il reste à convertir les équations $M_i = \prod_{j \in S_i} X_j$ traduisant l'ajout des nouvelles variables M_i (étape 2). Il y a deux cas à distinguer :

– Si $M_i = 0$ alors au moins une des variables X_j est nulle.

– Si $M_i = 1$ alors toutes les variables X_j ($j \in S_i$), sont non nulles

En faisant correspondre les variables M_i de l'anneau avec de nouvelles variables propositionnelles m_i, ces deux cas se traduisent par les formules propositionnelles :

$$\neg m_i \Rightarrow \bigvee_{j \in S_i} \neg x_j \qquad \text{et} \qquad m_i \Rightarrow \bigwedge_{j \in S_i} x_j$$

qui sont équivalentes aux clauses suivantes

$$m_i \vee \left(\bigvee_{j \in S_i} \neg x_j \right) \qquad \text{et} \qquad \bigwedge_{j \in S_i} (\neg m_i \vee x_j)$$

En notant d_i le degré du monôme M_i, on obtient donc $d_i + 1$ clauses pour décrire chaque nouvelle variable m_i.

Exemple 12. *Soit le système d'équations suivant dans* $\mathbb{F}_2[X_1, X_2, X_3, X_4]$ *:*

$$\begin{cases} X_1X_2 + X_2X_3 + X_2 + X_4 &= 0 \\ X_2X_3 + X_1 + X_3 &= 1 \end{cases}$$

On commence par linéariser le système en introduisant deux nouvelles variables M_1 *et* M_2 *:*

$$\begin{cases} M_1 + M_2 + X_2 + X_4 &= 0 \\ M_2 + X_1 + X_3 &= 1 \end{cases} \quad \textit{avec} \quad \begin{aligned} M_1 &= X_1X_2 \\ M_2 &= X_2X_3 \end{aligned}$$

Ces équations linéaires sont alors représentées par l'ensemble de clauses suivantes sur $\mathscr{P} = \{x_1, x_2, x_3, x_4, m_1, m_2\}$ *:*

$$\begin{array}{ll} \neg m_1 \vee m_2 \vee x_2 \vee x_4 & \\ m_1 \vee \neg m_2 \vee x_2 \vee x_4 & \\ m_1 \vee m_2 \vee \neg x_2 \vee x_4 & m_2 \vee x_1 \vee x_3 \\ m_1 \vee m_2 \vee x_2 \vee \neg x_4 & \neg m_2 \vee \neg x_1 \vee x_3 \\ \neg m_1 \vee \neg m_2 \vee \neg x_2 \vee x_4 & \neg m_2 \vee x_1 \vee \neg x_3 \\ \neg m_1 \vee \neg m_2 \vee x_2 \vee \neg x_4 & m_2 \vee \neg x_1 \vee \neg x_3 \\ \neg m_1 \vee m_2 \vee \neg x_2 \vee \neg x_4 & \\ m_1 \vee \neg m_2 \vee \neg x_2 \vee \neg x_4 & \end{array}$$

Les nouvelles variables m_1 et m_2 sont respectivement décrites par les clauses :

$$\begin{array}{ll} m_1 \vee \neg x_1 \vee \neg x_2 & m_2 \vee \neg x_2 \vee \neg x_3 \\ \neg m_1 \vee x_1 & \neg m_2 \vee x_2 \\ \neg m_1 \vee x_2 & \neg m_2 \vee x_3 \end{array}$$

Ces clauses forment une formule en CNF qui représente le système d'équations de départ. Pour résoudre ce système, il suffit donc de trouver les distributions qui satisfont toutes les clauses.

De nombreuses implémentations de ce procédé ont été proposées, avec différentes améliorations et optimisations possibles.

Une fois le système d'équations converti en un ensemble de clauses, les solutions sont recherchées grâce à un SAT-solveur.

2.3.3 L'algorithme DPLL

Le théorème de Cook, démontré en 1971, montre que le problème SAT est NP-complet [Coo71]. Cependant, les algorithmes de résolutions et leurs implantations se sont révélés très efficaces en pratique. Cette efficacité est surtout liée à l'utilisation de choix heuristiques pendant l'exécution. Une compétition est d'ailleurs organisée tous les ans afin de motiver le développement des SAT-solveurs, de comparer les différents algorithmes et de sélectionner les meilleurs heuristiques.

Nous présentons rapidement l'algorithme DPLL [DP60, DLL62] qui est à la base de la plupart des SAT-solveurs. Il permet de décider si une formule propositionnelle en forme normale conjonctive est satisfiable et dans ce cas retourne une distribution qui la satisfait. L'algorithme est basé sur une méthode de backtracking. Il procède en affectant une valeur de vérité à une variable bien choisie, puis vérifie récursivement si la formule une fois simplifiée est toujours satisfiable. Dans le cas contraire, la valeur de vérité affectée à la variable est changée puis on recommence pour vérifier récursivement que la formule est maintenant satisfiable. Si les deux cas retournent des résultats négatifs, alors la formule de départ n'était pas satisfiable, sinon une distribution est trouvée.

L'étape 7 de l'algorithme DPLL consiste à choisir la prochaine variable à fixer. L'efficacité pratique de l'algorithme va fortement dépendre de la façon dont est effectué ce choix. Les SAT solveurs utilisent alors différentes heuristiques à cette étape.

L'algorithme DPLL nécessite aussi une procédure de simplification à l'étape 5. En effet, une clause formée d'un seul littéral l dans une formule f en

Algorithme 3: Algorithme DPLL

Entrée : Une formule f en CNF constituée de m clauses $\{c_1, \ldots, c_m\}$ sur n variables $\{x_1, \ldots, x_n\}$.
Sortie : Une distribution qui satisfait f si f est satisfiable, $\varnothing$ sinon.

si *une des clauses n'est pas satisfaite, ou s'il y a une clause vide* **alors**
  retourner $\varnothing$;
fin
pour chaque *clause unitaire* ($c_i = l$) *dans* f **faire**
  simplifier f par c_i ;
fin
Choisir une variable x_i utilisée dans une clause de f;
si $S = DPLL(x_i \wedge f)$ *est non vide* **alors**
  retourner S;
sinon
  retourner DPLL($\neg x_i \wedge f$);
fin

CNF permet de fixer la valeur de la variable constituant ce littéral l. Toutes les autres clauses de f contenant l sont donc satisfaites et peuvent donc être enlevées. Inversement, toutes les clauses de f qui contiennent $\neg l$ doivent être satisfaites par les autres littéraux qui la composent. La présence du littéral $\neg l$ est donc superflu et on peut donc l'enlever. Cette simplification, aussi appelée propagation de clauses unitaires, est décrite dans l'algorithme 4.

Algorithme 4: Algorithme simplifier

Entrées : Une formule f en CNF constituée de m clauses $\{c_1, \ldots, c_m\}$ sur n variables $\{x_1, \ldots, x_n\}$ et un littéral l

pour chaque *clause* c_i *dans* f **faire**
  si c_i *contient* l **alors**
    enlever c_i de f
  sinon si c_i *contient* $\neg l$ **alors**
    enlever $\neg l$ dans c_i
  fin
fin

Dans la suite de cette thèse, nous utilisons les SAT solveurs CryptoMiniSat [SNC09], glucose2 [AS11a, AS09], glueminisat [AS11b] et plingeling [Bie11] qui sont les gagnants des compétitions SAT Race 2010 et SAT Race 2011.

Chapitre 3

Réduction de réseaux euclidiens

Dans ce chapitre, nous rappelons quelques définitions et propriétés sur les réseaux euclidiens, ainsi que l'algorithme de Lenstra–Lenstra–Lovász (LLL) de réduction de réseaux en temps polynomial ([LLL82]). Pour une présentation plus étendue, nous renvoyons le lecteur vers les ouvrages de références suivants [Mar96, Cas97, Sma98, MG02, Ste05, NV09]. Les problèmes liés aux chiffrements à clef publique qui nous intéresseront au chapitre 5, seront décrits par des systèmes d'équations modulaires à plusieurs inconnues. Ces systèmes seront construits de manière à ce que leurs petites solutions entières correspondent aux paramètres secrets. Pour trouver ces solutions, nous utiliserons des méthodes basées sur la réduction de réseaux euclidiens.

3.1 Les réseaux euclidiens

Nous considérons l'espace vectoriel $\mathbb{R}^n$ muni du produit scalaire canonique

$$\langle \mathbf{x}, \mathbf{y} \rangle = \langle (x_1, \ldots, x_n), (y_1, \ldots, y_n) \rangle = \sum_{i=1}^{n} x_i y_i,$$

conférant ainsi à $\mathbb{R}^n$ une structure d'espace vectoriel euclidien avec la topologie usuelle. Nous adoptons donc un point de vue topologique, ce qui nous permet de définir la notion d'ensemble discret.

Définition 19. *Un ensemble discret $\mathcal{D}$ est un sous-ensemble d'un espace topologique $\mathcal{E}$ sur lequel la topologie induite est la topologie discrète.*

Dit autrement, et dans le cas particulier où $\mathcal{E} = \mathbb{R}^n$, un sous-ensemble $\mathcal{D}$ de $\mathbb{R}^n$ est dit discret s'il n'a pas de point limite, c'est-à-dire lorsque

$$\forall \mathbf{x} \in \mathcal{D}, \exists \epsilon > 0 \text{ tel que } \mathcal{B}(\mathbf{x}, \epsilon) \cap \mathcal{D} = \{\mathbf{x}\}$$

avec $\mathcal{B}(\mathbf{x}, \epsilon) = \{\mathbf{y} \in \mathbb{R}^n \mid \|\mathbf{x} - \mathbf{y}\| < \epsilon\}$ la boule ouverte de rayon ϵ centrée en $\mathbf{x}$.

Exemples classiques :

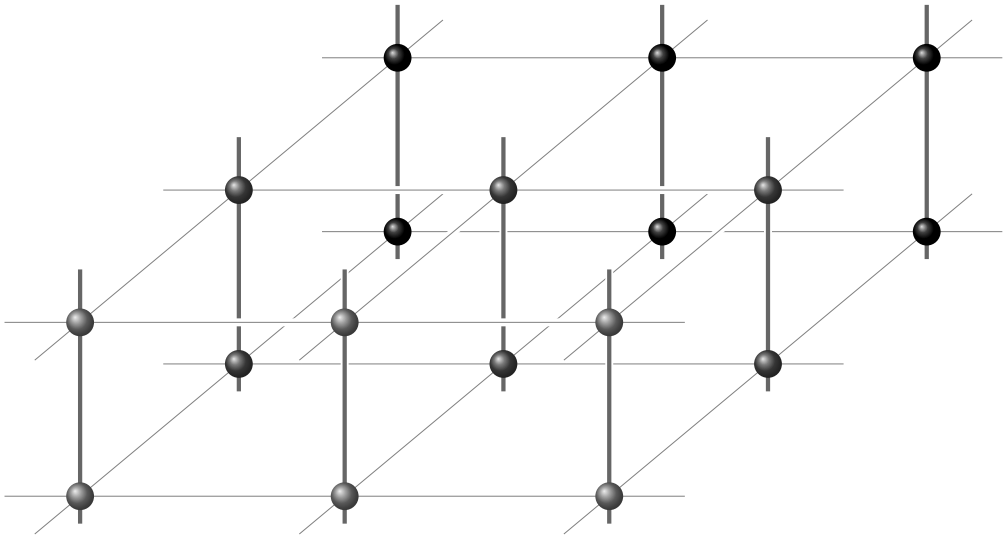
- $\mathbb{Z}^n$ est discret (prendre $\epsilon = 1/2$)
- $\mathbb{Q}^n$ et $\mathbb{R}^n$ ne sont pas discrets.
- $\{1/n \mid n \in \mathbb{N}^*\}$ est discret mais
- $\{1/n \mid n \in \mathbb{N}^*\} \cup \{0\}$ ne l'est pas car 0 est un point limite.
- Tout sous-ensemble d'un ensemble discret est discret.

On remarquera sur ces exemples que dénombrable et discret sont bien des notions distinctes.

Définition 20. *Un réseau euclidien Λ de $\mathbb{R}^n$ est un sous-groupe additif discret de $\mathbb{R}^n$.*

Géométriquement, un réseau euclidien de $\mathbb{R}^n$ est donc un arrangement régulier de points dans $\mathbb{R}^n$.

Exemple 13. *$\mathbb{Z}^n$ est un réseau euclidien de $\mathbb{R}^n$.*



La définition 20 implique en particulier que tout sous-groupe d'un réseau est un réseau. D'où la définition suivante :

Définition 21. *Un sous-réseau d'un réseau Λ est sous-groupe de Λ.*

Exemple 14. *Soient $a_1, \ldots, a_n$ et M des entiers. L'ensemble des éléments $(x_1, \ldots, x_n) \in \mathbb{Z}^n$ tels que $\sum_{i=1}^{n} a_i x_i \equiv 0 \pmod{M}$ est un sous-réseau de $\mathbb{Z}^n$.*

Nous rappelons maintenant quelques propriétés élémentaires sur les réseaux euclidiens :

Proposition 6 ([NV09])**.** *Soit Λ un réseau euclidien de $\mathbb{R}^n$. On a alors :*
- *Λ est fermé dans $\mathbb{R}^n$.*
- *Λ est infini dénombrable ou trivial.*
- *Pour tout sous-ensemble borné S de $\mathbb{R}^n$, $\Lambda \cup S$ est fini.*
- *$\exists \epsilon > 0$ tel que $\forall \mathbf{x} \in \Lambda$, $\Lambda \cup \mathcal{B}(\mathbf{x}, \epsilon) = \{\mathbf{x}\}$.*

Similairement aux espaces vectoriels, les réseaux seront souvent représentés par un ensemble générateur ou par une base :

Définition 22. *Soient $\mathbf{b}_1, \ldots, \mathbf{b}_d \in \mathbb{R}^n$. Lorsque l'ensemble des combinaisons linéaires des $\mathbf{b}_i$ à coefficients entiers forme un réseau Λ, i.e.*

$$\Lambda = \left\{ \sum_{i=1}^{n} a_i \mathbf{b}_i \mid a_i \in \mathbb{Z} \right\}$$

on dit que Λ est engendré par les vecteurs $\mathbf{b}_1, \ldots, \mathbf{b}_d$, ou encore que les $\mathbf{b}_1, \ldots, \mathbf{b}_d$ sont des générateurs de Λ. De plus, si les $\mathbf{b}_i$ sont linéairement indépendants, alors $(\mathbf{b}_1, \ldots, \mathbf{b}_d)$ forme une base du réseau Λ. Dans ce cas, chaque vecteur du réseau s'exprime de manière unique comme combinaison linéaire à coefficients entiers des éléments de la base.

Toutes les bases d'un même réseau Λ de $\mathbb{R}^n$ possèdent le même nombre d'éléments d, appelé la dimension ou le rang de Λ. Le réseau est dit de rang plein si $d = n$. Par contre, contrairement aux espaces vectoriels, un ensemble de d vecteurs linéairement indépendants d'un réseau de dimension d ne forment pas nécessairement une base. On a néanmoins le théorème d'existence suivant :

Théorème 6. *Tout réseau Λ de $\mathbb{R}^n$ possède au moins une base.*

On peut rajouter que tous les réseaux de dimension supérieure ou égale à 2 possèdent une infinité de bases. Toutes les bases d'un même réseau sont d'ailleurs reliées par une transformation unimodulaire.

3.2 Minimums d'un réseau euclidien et SVP

Soit Λ un réseau de $\mathbb{R}^n$ de dimension $d \geq 1$ et $\mathbf{b}_1, \ldots, \mathbf{b}_d \in \mathbb{R}^n$ une base de Λ. L'ensemble Λ étant discret, il existe un vecteur non-nul de Λ de norme euclidienne minimale. On dit que ce vecteur est un *plus court vecteur* du réseau, et sa norme, notée $\lambda_1(\Lambda)$, s'appelle le premier minimum de Λ. De la même façon, on définit aussi les minimums successifs du réseau :

Définition 23 (Minkowski)**.** *Pour $1 \leq i \leq d$, le i-ème minimum $\lambda_i(\Lambda)$ est le rayon de la plus petite boule fermée centrée en 0 qui contient au moins i vecteurs non nuls de Λ linéairement indépendants. On appelle $\lambda_1(\Lambda), \ldots, \lambda_d(\Lambda)$ les d minimums successifs de Λ.*

On remarquera que les minimums vérifient $\lambda_1(\Lambda) \leq \lambda_2(\Lambda) \leq \cdots \leq \lambda_d(\Lambda)$. De plus, ces minimums sont toujours atteints, c'est-à-dire qu'il existe toujours des vecteurs de Λ linéairement indépendants de normes égales aux minimums. Cependant, ces vecteurs ne forment pas nécessairement une base du réseau.

À partir d'une base du réseau, estimer précisément ses minimums ou trouver des vecteurs atteignant ses minimums sera d'autant plus difficile que la dimension du réseau sera grande. En particulier, le problème consistant à trouver un plus court vecteur dans un réseau est appelé **SVP** (*shortest vector problem*) et constitue l'un des principaux problèmes de l'algorithmique des réseaux euclidiens. En 1998, Ajtai [Ajt98] a démontré que ce problème est NP-difficile pour des réductions randomisées [1]. Et Khot a démontré en 2004 [Kho04] qu'approcher $\lambda_1(\Lambda)$ à une constante près est NP-difficile sous des réductions randomisées. Nous renvoyons au livre [MG02] pour une présentation de la complexité des problèmes sur les réseaux euclidiens.

Les meilleurs algorithmes pour résoudre le problème SVP sont l'algorithme déterministe de Kannan [Kan83] (et ses améliorations successives) de complexité $d^{O(d)}$ et l'algorithme probabiliste de Ajtai, Kumar et Sivakumar [AKS01] de complexité $2^{O(d)}$. Voir [HPS11] pour un état de l'art des algorithmes de résolution du SVP. Il existe cependant des algorithmes de complexité polynomiale qui fournissent des approximations du SVP d'un facteur exponentiel en la dimension du réseau, le plus connu étant l'algorithme LLL[LLL82] que nous détaillerons dans la section 3.4.

Il y a aussi des résultats reliant les minimums à d'autres invariants du réseau. Les deux théorèmes de Minkowski par exemple peuvent être énoncés

1. La classe des algorithmes de complexité polynomiale est étendue à ceux qui ne sont pas déterministes mais qui ont une forte probabilité de terminer correctement en temps polynomial. Voir [Ajt98] pour plus de précisions.

simplement à partir d'un invariant appelé le volume d'un réseau.

Définition 24. *Le déterminant de Gram de* $(\mathbf{b}_1, \ldots, \mathbf{b}_d) \in \mathbb{R}^n$ *noté* $\Delta(\mathbf{b}_1, \ldots, \mathbf{b}_d)$*, est défini comme le déterminant de la matrice* $d \times d$ *de Gram* $(\langle \mathbf{b}_i, \mathbf{b}_j \rangle)_{1 \leq i,j \leq d}$*. Le volume d'un réseau* $\Lambda = \langle \mathbf{b}_1, \ldots, \mathbf{b}_d \rangle$ *de* $\mathbb{R}^n$ *est alors défini par* $\mathrm{Vol}(\Lambda) = \Delta(\mathbf{b}_1, \ldots, \mathbf{b}_d)^{1/2}$*. C'est le volume (plus précisément, la mesure de Lebesgue) de l'espace délimité par le parallélépipède formé par les vecteurs de la base.*

Tout comme la dimension, le volume est un invariant du réseau, donc indépendant du choix de la base. À partir de la définition donnée, le volume d'un réseau peut donc être calculé dès que l'on connait au moins une base de ce réseau. Mais, il peut arriver qu'un réseau soit défini par un ensemble de générateurs ne formant pas une base. Ce cas se présentera à nous au chapitre 5. Nous avons donc besoin d'un autre moyen pour calculer le volume des réseaux engendrés par un ensemble de vecteurs linéairement dépendants :

Lemme 2 ([NV09])**.** *Un sous-réseau* L *de* Λ *est de rang plein si et seulement si son indice comme sous-groupe* $[\Lambda : L]$ *est fini. Dans ce cas, on a alors*

$$\mathrm{Vol}(L) = \mathrm{Vol}(\Lambda)[\Lambda : L].$$

Le volume permet entre autres de majorer les minimums d'un réseau grâce aux théorèmes de Minkowski :

Théorème 7 (Théorèmes de Minkowski)**.** *Soit* Λ *un réseau de dimension* d*. Alors on a :*

$$\lambda_1(\Lambda) \leq \sqrt{d}(\mathrm{Vol}\,\Lambda)^{1/d} \quad \text{et} \quad \prod_{i=1}^{d} \lambda_i(\Lambda) \leq \sqrt{d}^{\,d}\,\mathrm{Vol}(\Lambda).$$

L'inégalité de Hadamard est un autre résultat classique reliant les bases d'un réseau à son volume :

Lemme 3 (Inégalité de Hadamard)**.** *Soit* Λ *un réseau de* $\mathbb{R}^n$ *de dimension* n *et* $\mathbf{b}_1, \ldots, \mathbf{b}_n \in \mathbb{R}^n$ *une base de* Λ*. Alors*

$$\mathrm{Vol}(\Lambda) \leq \prod_{i=1}^{n} \|\mathbf{b}_i\|$$

avec égalité si et seulement si la base est une base orthogonale de $\mathbb{R}^n$*.*

3.3 Réduction de réseaux

Déterminer une suite de vecteurs de normes égales aux minimums successifs est au moins aussi difficile que la résolution du SVP, qui est NP-difficile [MG02]. Pour contourner ce problème, un autre point de vue est alors de se restreindre à seulement en chercher une bonne approximation. Ce problème algorithmique, appelé la réduction des réseaux, consiste donc, à partir d'une base d'un réseau, à calculer une autre base dite réduite, c'est-à-dire une base dont les longueurs de ses vecteurs sont "proches" des minimums successifs. D'après les théorèmes de Minkowski, on cherchera donc une base dont le produit des normes des vecteurs est de l'ordre du volume du réseau, à un facteur près à déterminer. Mais, réduire une base revient à la rendre aussi plus orthogonale (résultat de Hadamard, Lemme 3).

Cependant, un réseau ne possède pas de base orthogonale en général. Des nombreuses notions de réduction différentes ont donc été proposées avec des conditions de taille et d'orthogonalité plus ou moins fortes : réduction au sens de Hermite, de Minkowski, de Siegel, de Hermite-Korkine-Zolotarev (HKZ), de Schnorr-Euchner (BKZ), de Lenstra–Lenstra–Lovász (LLL), ...

La réduction LLL, proposée en 1982, est particulièrement intéressante puisqu'elle permet d'obtenir une base réduite en temps polynomial déterministe. Nous allons donner la définition et quelques propriétés de cette réduction, puis l'algorithme LLL pour calculer des bases LLL-réduites que nous utiliserons dans la suite. Cette réduction s'appuie sur le processus d'orthogonalisation de Gram-Schmidt qu'il convient donc de rappeler :

Définition 25 (Orthogonalisation de Gram-Schmidt)**.** *Soient des vecteurs* $\mathbf{b}_1, \ldots, \mathbf{b}_d \in \mathbb{R}^n$ *linéairement indépendants. L'orthogonalisation de Gram-Schmidt de* $\mathbf{b}_1, \ldots, \mathbf{b}_d$ *est la famille de vecteurs* $\mathbf{b}_1^*, \ldots, \mathbf{b}_d^*$ *définie par*

$$\begin{aligned} \mathbf{b}_1^* &= \mathbf{b}_1 \\ \mathbf{b}_i^* &= \mathbf{b}_i - \sum_{j=1}^{i-1} \mathrm{proj}_{\mathbf{b}_j^*}(\mathbf{b}_i) \quad \textit{pour } 2 \leq i \leq d \end{aligned}$$

où $\mathrm{proj}_{\mathbf{b}_j^*}(\mathbf{b}_i) = \frac{\langle \mathbf{b}_i, \mathbf{b}_j^* \rangle}{\|\mathbf{b}_j^*\|^2} \mathbf{b}_j^*$.

On définit alors la notion de LLL-reduction :

Définition 26 (LLL-réduction [LLL82])**.** *Une base* $(\mathbf{b}_1, \ldots, \mathbf{b}_d)$ *est dite LLL-réduite avec un facteur* $\delta \in]1/4, 1]$ *si les deux propriétés suivantes sont*

vérifiées :

$$\|\mathrm{proj}_{\mathbf{b}_j^*}(\mathbf{b}_i)\| \leq \frac{1}{2}\|\mathbf{b}_j^*\| \quad \textit{pour } 1 \leq j < i \leq d \tag{3.1}$$

$$\|\mathbf{b}_i^* + \mathrm{proj}_{\mathbf{b}_{i-1}^*}(\mathbf{b}_i)\| \geq \delta\|\mathbf{b}_{i-1}^*\| \quad \textit{pour } 2 \leq i \leq d \tag{3.2}$$

La première condition (3.1) est appelée *condition de réduction*. La seconde condition (3.2) s'appelle la *condition de Lovász*. Une base LLL-réduite satisfait les propriétés suivantes :

Proposition 7 ([LLL82])**.** *Soit Λ le réseau engendré par la base LLL-réduite $(\mathbf{b}_1, \ldots, \mathbf{b}_d)$ avec un facteur $\delta \in]1/4, 1]$. Alors on a*

- $\|\mathbf{b}_1\| \leq \left(\frac{1}{\delta - 1/4}\right)^{\frac{d-1}{4}} (\mathrm{Vol}\,\Lambda)^{\frac{1}{d}}$
- $\mathrm{Vol}\,\Lambda \leq \prod_{i=1}^{d} \|\mathbf{b}_i\| \leq \left(\frac{1}{\delta - 1/4}\right)^{\frac{d(d-1)}{4}} (\mathrm{Vol}\,\Lambda)$
- *Pour tout* $1 \leq i \leq d$, $\|\mathbf{b}_i\| \leq \left(\frac{1}{\delta - 1/4}\right)^{\frac{d-1}{2}} \lambda_i(\Lambda)$

En particulier, le premier vecteur $\mathbf{b}_1$ d'une base LLL-réduite est borné de la façon suivante

$$\lambda_1(\Lambda) \leq \|\mathbf{b}_1\| \leq \left(\frac{1}{\delta - 1/4}\right)^{\frac{d-1}{2}} \lambda_1(\Lambda)$$

Le vecteur $\mathbf{b}_1$ est ainsi une bonne approximation d'un plus petit vecteur, malgré un facteur $\left(\frac{1}{\delta - 1/4}\right)^{\frac{d-1}{2}}$ exponentiel en la dimension d du réseau.

3.4 Algorithme LLL

À partir d'une base quelconque d'un réseau Λ, l'algorithme LLL calcule en temps polynomial une base LLL-réduite de Λ de facteur $\delta \in]1/4, 1[$. Afin de toujours garder une base du réseau, on ne s'autorise que deux sortes d'opérations sur les vecteurs : des échanges entre vecteurs consécutifs et des réductions de la forme $\mathbf{b}_i := \mathbf{b}_i - m\mathbf{b}_j$, où m est un entier et $j < i$.

L'algorithme fonctionne alors par récurrence : sur des vecteurs $(\mathbf{b}_1, \ldots, \mathbf{b}_{i-1})$ formant déjà une famille LLL-réduite, on cherche à compléter cette base tronquée en rajoutant un nouveau vecteur qui satisfait, à la fois la condition de réduction (3.1) et la condition de Lovász (3.2). Pour obtenir la première condition (3.1), le vecteur suivant $\mathbf{b}_i$ sera réduit par rapport à $(\mathbf{b}_1, \ldots, \mathbf{b}_{i-1})$. Pour cela, il suffit de soustraire à $\mathbf{b}_i$ des multiples entiers des

Algorithme 5: Algorithme LLL

Entrées : Une base $(\mathbf{b}_1, \ldots, \mathbf{b}_d)$ du réseau Λ et $\delta \in]1/4, 1[$
Sortie : La base $(\mathbf{b}_1, \ldots, \mathbf{b}_d)$ LLL-réduite avec un facteur δ

```
1  i := 2;
2  tant que i ≤ d faire
3  |  Réduire b_i par (b_1, ..., b_{i-1});
   |  // Condition de réduction vérifiée par (b_1, ..., b_i)
4  |  si ||b*_i + proj_{b*_{i-1}}(b_i)|| ≥ δ||b*_{i-1}|| alors
   |  |  // La condition de Lovász est vérifiée par b_i
5  |  |  i := i + 1;
6  |  sinon
7  |  |  Echanger b_i et b_{i-1};
8  |  |  i := i − 1;
9  |  fin
10 fin
```

vecteurs précédents en imitant le processus d'orthogonalisation de Gram-Schmidt :

$$\mathbf{b}_i := \mathbf{b}_i - \sum_{j=1}^{i-1} \left\lfloor \frac{\|\mathrm{proj}_{\mathbf{b}_j}(\mathbf{b}_i)\|}{\|\mathbf{b}_j\|} \right\rceil \mathbf{b}_j$$

où $\lfloor . \rceil$ désigne l'entier le plus proche. Si $\mathbf{b}_i$ vérifie aussi la condition de Lovász (3.2), alors on peut le rajouter à la famille LLL-réduite en incrémentant l'indice i. Sinon, le vecteur $\mathbf{b}_i$ doit être placé avant le vecteur $\mathbf{b}_{i-1}$. Dans ce cas, la famille $(\mathbf{b}_1, \ldots, \mathbf{b}_{i-2}, \mathbf{b}_i)$ n'est pas, à priori, LLL-réduite. Après avoir échanger $\mathbf{b}_{i-1}$ et $\mathbf{b}_i$, il faut donc décrémenter l'indice i pour recommencer avec la famille LLL-réduite $(\mathbf{b}_1, \ldots, \mathbf{b}_{i-2})$.

La terminaison de cet algorithme n'est pas évidente puisque l'indice i, qui détermine la sortie de la récurrence (lorsque $i = n+1$), peut être décrémenté à l'étape [8] après avoir été incrémenté à l'étape [5]. Pourtant, on a le théorème suivant :

Théorème 8. *Soit $(\mathbf{b_1}, \ldots, \mathbf{b_d})$ une base d'un réseau Λ de $\mathbb{Z}^n$. L'algorithme LLL calcule une base LLL-réduite de Λ en temps polynomial ([LLL82]). Plus précisément, le nombre d'itérations (boucle [2]-[10]) est borné par $\mathcal{O}(d^2 \log B)$, où $B = \max_i(\|\mathbf{b_i}\|)$. L'implémentation efficace de la version flottante de Nguyen and Stehlé requiert $\mathcal{O}(d^5(d + \log B)\log B)$ opérations élémentaires ([NS09]).*

Tout au long de cette thèse, les réductions par l'algorithme LLL sont effectuées avec l'implémentation efficace dans MAGMA [BCP97] de la version flottante prouvée de Nguyen and Stehlé [Ste05, NS09, NV09].

3.5 Heuristique gaussienne

Sur des réseaux particuliers, il arrive souvent que l'approximation donnée par l'algorithme LLL soit suffisante pour que le premier vecteur de la base LLL-réduite soit un plus petit vecteur. La valeur exacte de $\lambda_1(\Lambda)$ étant inconnue à priori, des arguments probabilistes sont donc souvent utilisés pour estimer la difficulté pour trouver un plus petit vecteur dans un réseau donné. Nous allons brièvement donner les idées qui conduisent à établir une telle estimation, souvent appelée l'heuristique gaussienne par abus de langage.

Lemme 4. *Soit Λ un réseau de rang plein de $\mathbb{Z}^n$. Alors,*

$$\lim_{r\to\infty} \#\left(\mathcal{B}(0,r)\cap\Lambda\right) = \frac{\mathrm{Vol}(\mathcal{B}(0,r))}{\mathrm{Vol}(\Lambda)}$$

Pour r suffisamment grand, ce lemme justifie l'estimation suivante

$$\#\{\mathbf{x}\in\Lambda \mid \|\mathbf{x}\|\leq r\} \approx \frac{\mathrm{Vol}(\mathcal{B}(0,r))}{\mathrm{Vol}(\Lambda)}$$

Remarquons que la norme d'un plus petit vecteur est la distance minimale r telle que $\#\{\mathbf{x}\in\Lambda \mid \|\mathbf{x}\|\leq r\} = 1$. Or, en supposant que le réseau est de grande dimension, on a l'approximation suivante

$$\mathrm{Vol}(\mathcal{B}(\mathbf{x},r)) = \frac{\pi^{n/2}r^n}{\Gamma(1+n/2)} \approx \frac{\pi^{n/2}r^n}{(n/2e)^{n/2}} = \left(\sqrt{\frac{2\pi e}{n}}\, r\right)^n$$

Théorème 9 ([Ajt06])**.** *Sur des réseaux tirés aléatoirement, lorsque la dimension n est suffisamment grande et pour tout $1\leq i\leq n$, il y a une forte probabilité pour que l'on ait :*

$$\lambda_i(\Lambda) \approx \sqrt{\frac{n}{2\pi e}}\,\mathrm{Vol}(\Lambda)^{\frac{1}{n}}$$

On fait alors l'hypothèse que dans un réseau "choisi aléatoirement" de dimension n, un plus petit vecteur aura alors une taille d'environ

$$\lambda_1(\Lambda) \approx \sqrt{\frac{n}{2\pi e}}\,\mathrm{Vol}(L)^{\frac{1}{n}}$$

En pratique, on suppose généralement que les réseaux construits se comportent comme des réseaux aléatoires et que si un vecteur $\mathbf{x} \in \Lambda$ est plus petit que $\sqrt{\frac{n}{2\pi e}}\,\mathrm{Vol}(L)^{\frac{1}{n}}$, alors $\mathbf{x}$ doit être un plus petit vecteur de Λ. De plus, lorsque l'écart entre la norme de ce plus petit vecteur et $\sqrt{\frac{n}{2\pi e}}\,\mathrm{Vol}(L)^{\frac{1}{n}}$ est assez grand, ce plus petit vecteur semble facile à trouver par la réduction de réseaux. Dans ce cas, une méthode heuristique pour calculer un plus petit vecteur consiste alors à utiliser l'algorithme LLL.

Dans cette thèse, nous ferons l'hypothèse que les réseaux utilisés sont assez aléatoires pour vérifier l'heuristique gaussienne (chapitre 5). Cette hypothèse a été vérifiée par des résultats d'expériences pratiques.

Chapitre 4

Attaques par canaux auxiliaires

4.1 Introduction

On dénomme attaque par canaux auxiliaires (ou aussi attaque physique ou *Side Channel Attack*) toute attaque qui exploite l'implémentation d'un algorithme cryptographique sur un support physique. En effet, la sécurité fournie par un appareil mettant en œuvre un algorithme cryptographique ne dépend pas uniquement de la robustesse mathématique de cet algorithme, mais bien de la sécurité de l'ensemble du système.

En pratique, des informations sur le fonctionnement interne du système peuvent fuir pendant l'exécution de l'algorithme. Un attaquant peut ainsi récupérer les fuites d'information dépendant des opérations effectuées et des données manipulées, en particulier la clef secrète ou des variables intermédiaires dépendant de cette clef. On dit alors que l'attaquant a utilisé un canal auxiliaire pour collecter des fuites d'information. L'analyse de ces données permet ensuite de mener des attaques particulièrement efficaces.

Un grand nombre de techniques ont été proposées pour exploiter des canaux auxiliaires. On peut les distinguer en deux grandes catégories : les attaques passives et les attaques actives.

Les attaques passives sont celles qui se contentent d'observer un appareil cryptographique sans modifier son exécution et récupèrent généralement des informations par les variations de ses propriétés physiques, comme par exemple, le temps d'exécution, le rayonnement électromagnétique, ou la consommation électrique. L'interprétation de ces variations nécessite souvent le choix d'un modèle de fuite, c'est-à-dire d'une fonction de référence explicitant le lien entre les données manipulées et ces variations.

Dans les attaques actives, l'attaquant tente de perturber l'exécution afin

de provoquer un comportement anormal de l'appareil. Par exemple, en injectant une faute durant un calcul pour modifier le résultat attendu à la sortie de l'algorithme, et récupérer ainsi des fuites d'information exploitables. L'infection par des logiciels malveillants, tels que les trojans, peut aussi être considérée comme des attaques actives.

4.2 Attaques passives

La première publication d'une attaque par canaux auxiliaires contre une implémentation cryptographique a été proposée par Kocher en 1996 [Koc96]. Cette attaque tire profit du temps d'exécution de certaines opérations qui dépendent de la clef secrète. Les attaques par canaux auxiliaires ont ensuite été largement étudiées, en particulier celles exploitant la consommation du courant électrique ([KJJ99]) durant l'exécution d'un algorithme cryptographique.

Un circuit électronique consomme une certaine quantité de courant pour chaque opération qu'il effectue. Pour les circuits numériques en particulier, ce courant permet d'alimenter les portes logiques, de représenter les données dans les mémoires et de faire transiter ces données sur les bus de transmission. Par exemple, pour les circuits CMOS (principale technologie utilisée dans les dispositifs électroniques), la consommation totale de courant est la somme des consommations des cellules logiques le constituant. La consommation totale dépend donc surtout du nombre de cellules dans le circuit, de la façon dont elles sont reliées entre elles et de leurs consommations individuelles. Or, une cellule consomme surtout du courant lors d'un changement d'état. La consommation électrique du composant (ainsi que le champ électromagnétique) dépend donc fortement des données manipulées par le circuit.

D'un point de vue pratique, l'attaque commence par une première étape constituant la campagne d'acquisitions. Elle nécessite quelques équipements pour mesurer le canal auxiliaire pendant l'exécution de l'algorithme. Typiquement, le dispositif cryptographique (généralement un système embarqué) est commandé par un ordinateur à travers une interface dédiée. Pour une attaque visant une carte à puce par exemple, un lecteur de carte commandé par un ordinateur peut être utilisé pour lancer des chiffrements et récupérer les données chiffrés. Un appareil de mesure envoie alors à l'ordinateur les mesures effectuées tout au long du chiffrement pour un traitement ultérieur. Dans le cas d'une analyse de consommation du courant, on utilise un oscilloscope pour mesurer la différence de potentiel d'une résistance placée entre la source de courant et l'appareil visé. Dans le cas d'une analyse du champ élec-

tromagnétique, le signal est fourni par une sonde électromagnétique branchée à un oscilloscope.

Pour chaque exécution de l'algorithme, on obtient ainsi un ensemble de mesures qui forme une courbe en fonction du temps pour un couple (clef, message) donné et qu'on appelle trace de consommation, ou trace d'émission. Les traces contiennent toujours du bruit, c'est-à-dire qu'une part du signal mesuré est constitué de signaux indésirables variant aléatoirement. Le bruit a différentes sources, par exemple il peut venir de la source de courant, des différents composants électroniques, d'émissions thermiques ou électromagnétiques, ou encore de la qualité des appareils de mesure. La qualité des mesures dépend ainsi de la quantité de bruit qu'elles contiennent par rapport à la composante dépendant uniquement des données manipulées et des opérations effectuées.

Différentes techniques d'analyse des traces permettent ensuite de retrouver les informations secrètes. Lorsque le bruit est important, ces attaques consistent généralement à appliquer des méthodes statistiques sur un grand nombre de traces.

4.2.1 Analyse simple de consommation (SPA)

L'analyse simple de consommation consiste à retrouver la clef secrète en observant directement la forme d'une seule courbe de courant. Si le bruit n'est pas trop important, alors on peut chercher à identifier certains moments du chiffrement sur la courbe. Ainsi, les opérations conditionnelles sont particulièrement vulnérables puisqu'on pourra alors s'apercevoir de leur exécution ou de leur absence. Si la condition dépend d'une valeur secrète, alors on peut en déduire des informations intéressantes.

Un exemple classique est l'implémentation non protégée de l'exponentiation modulaire rapide (square and multiply) souvent utilisé dans les cryptosystèmes à clef publique comme RSA ou Diffie-Hellman. La figure 4.1 montre une trace de consommation lors de l'exécution d'un RSA utilisant une exponentiation modulaire rapide simple (i.e. non protégée). Dans ce cas, on peut directement lire la valeur des bits de l'exposant secret. En effet, pour chaque bit de l'exposant valant 1, une multiplication modulaire (M) est calculée en plus du carré modulaire (S).

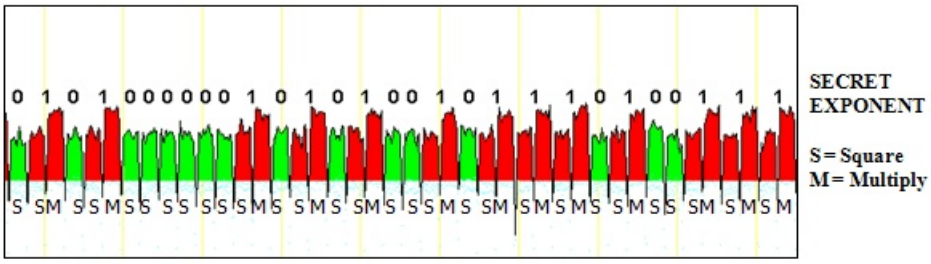


FIGURE 4.1 – SPA sur l'algorithme Square and Multiply

Une protection contre ce genre d'attaque consiste principalement à éviter les opérations conditionnelles. Dans l'exemple précèdent de l'exponentiation modulaire rapide, une protection possible serait donc de toujours calculer une multiplication modulaire (M) quelque soit la valeur des bits de l'exposant. Cependant, en contrepartie d'une plus grande sûreté contre la SPA, l'algorithme perd beaucoup en rapidité.

4.2.2 Analyse différentielle de consommation (DPA)

L'analyse différentielle de consommation (DPA pour *Differential Power Analysis*) est la principale attaque par analyse de consommation électrique. Elle a été proposée pour la première fois par P. Kocher, J. Jaffe et B. Jun en 1999 ([KJJ99]). Contrairement à la SPA, la DPA permet d'exploiter des traces de consommation très bruitées. En contrepartie, la DPA nécessite souvent un grand nombre de traces pour l'utilisation d'outils statistiques. Toutes les traces doivent cependant provenir de chiffrements effectués avec une clef secrète fixée.

En pratique, l'attaque commence par une campagne d'acquisition des traces de consommation de l'exécution de l'algorithme sur des données différentes. Suivant le contexte de l'attaque, chaque trace est généralement associée à un chiffré seul ou un texte clair. Une étape intermédiaire de l'algorithme doit ensuite être choisie telle que chaque bit b_i de cette étape intermédiaire puisse s'exprimer comme une fonction $b_i = f(d, k_i)$ du texte connu d et de k_i un petit nombre de bits de la clef. Par exemple, lors d'une attaque contre une implémentation d'un chiffrement par blocs avec clair connu, on peut commencer par cibler un bit de sortie de la première boîte-S du premier tour, dont la valeur ne dépend plus que des premiers bits de la clef.

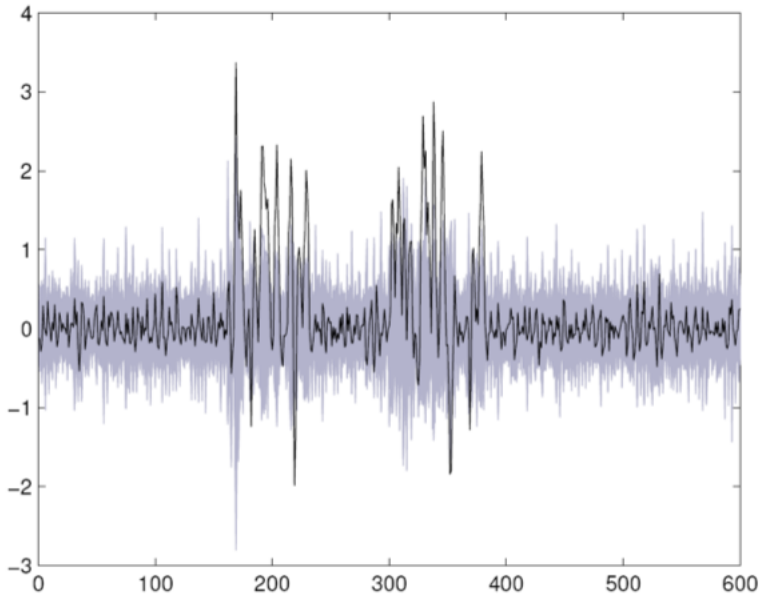


FIGURE 4.2 – Attaque DPA contre une implantation de l'AES. La différence correspondant à l'hypothèse correcte est tracée en noir. (Source : [Rva09])

Il ne reste plus qu'à appliquer une fonction de distinction sur la cible b_i choisie afin de déterminer la valeur la plus probable des bits de clef k_i correspondant. La différence de moyenne est la fonction de distinction originairement proposée pour la DPA. A partir d'une hypothèse sur la valeur des quelques bits k_i de la clef, la valeur prévue du bit $b_i = f(d, k_i)$ ciblé est calculée. Pour chaque hypothèse, les traces sont alors séparées en deux catégories, celles pour lesquelles le bit b_i est prévu égal à 1 et celle où il est prévu égal à 0. La différence Δ_{k_i} des moyennes des deux ensembles est calculée pour toutes les hypothèses sur k_i. Pour l'hypothèse correspondant à la bonne valeur de k_i, on s'attend à observer un pic de consommation à l'instant où les traces dépendent de la valeur du bit ciblé b_i (figure 4.2). Pour toutes les autres hypothèses, les moyennes des traces par catégorie tendent à devenir indépendantes de la valeur du bit ciblé b_i, et leurs différences tendront donc à s'annuler lorsque le nombre de traces augmente.

On peut ainsi retrouver toute la clef secrète en recommençant sur un autre bit cible.

4.2.3 Analyse de consommation par corrélation (CPA)

L'analyse de consommation par corrélation (CPA pour *Correlation Power Analysis*) fonctionne de façon très similaire à la DPA mais utilise un calcul de corrélation comme fonction de distinction ([CNK00, BCO04]). Cette attaque est donc souvent considérée comme une généralisation de la DPA.

Avec cette fonction de distinction, on peut aussi utiliser un modèle de fuite M, c'est-à-dire une fonction qui, pour des résultats supposés de calculs intermédiaires, renvoie une valeur linéairement corrélée à la consommation attendue. Le modèle de la distance de Hamming est un modèle souvent employé et qui consiste à compter le nombre de bits différents entre deux mots de données. Il est particulièrement adapté pour des attaques contre les circuits CMOS, la consommation de ces circuits étant principalement due aux changements d'états des cellules logiques.

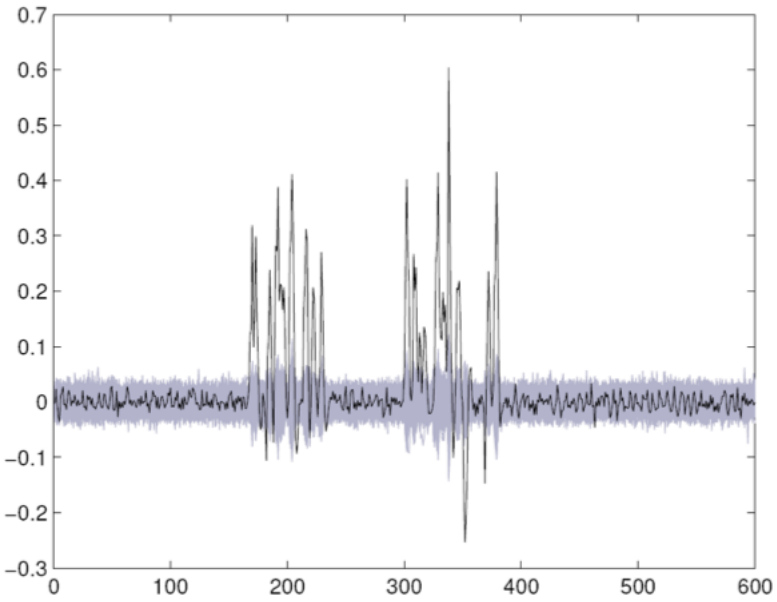


FIGURE 4.3 – Attaque CPA contre une implantation de l'AES. La différence correspondant à l'hypothèse correcte est tracée en noir. (Source : [Rva09])

En pratique, l'analyse commence, comme pour la DPA, par choisir un résultat intermédiaire $r_i = f(d, k_i)$ calculable à partir de la donnée connue

d et d'une hypothèse k_i sur un petit nombre de bits de la clef. L'attaquant peut donc calculer les valeurs intermédiaires r_i pour toutes les hypothèses de clef k_i et pour toutes les traces disponibles. La bonne hypothèse de clef est celle qui maximise le coefficient de corrélation $\rho[M(f(d, k_i)), T]$ entre les traces consommations enregistrées T et la valeur $M(r_i)$ fournie par le modèle de fuite sur la valeur intermédiaire prédite.

4.2.4 Attaques profilées (templates)

Dans les attaques précédentes, l'attaquant a une connaissance très limitée sur les caractéristiques de consommation de l'appareil attaqué. Au mieux, nous avons pu utiliser des modèles de fuites simplistes, comme la distance de Hamming. Au contraire, dans les attaques profilées ([CRR02]), l'attaquant va pouvoir acquérir la meilleure description possible des profils de consommation de l'appareil. Pour cela, on suppose généralement que l'attaquant a la maîtrise totale d'un autre exemplaire de l'appareil, sur lequel il peut donc choisir les données à chiffrer ainsi que la clef utilisée, et obtenir les traces correspondantes. Ces traces sont ensuite utilisées pour construire des profils de valeurs de fuites (ou templates) qui consistent à estimer des fonctions de densité de probabilité $t \mapsto P(t|k_i)$ pour toute trace t et pour toute hypothèse de clef k_i.

Les attaques profilées utilisent alors ces templates pour estimer le maximum de vraisemblance d'une hypothèse de clef k_i pour une trace donnée t, i.e. la probabilité conditionnelle $P(k_i|t)$. Par le théorème de Bayes, on peut calculer cette probabilité à partir de $P(k_i)$ (en supposant les hypothèses de clef uniformément distribuées) et d'après les templates qui fournissent les $P(t|k_i)$ pour toutes les hypothèses de clef k_i :

$$P(k_i|t) = \frac{P(t|k_i)P(k_i)}{\sum_j P(t|k_j)P(k_j)} \tag{4.1}$$

Une seule trace n'apporte souvent pas suffisamment d'information pour retrouver la clef secrète. On doit alors étendre cette approche à une multitude de traces T. En pratique, on suppose que ces traces sont mutuellement indépendantes, ce qui donne

$$P(k_i|T) = \frac{(\prod_{t \in T} P(t|k_i)) \cdot P(k_i)}{\sum_j (\prod_{t \in T} P(t|k_j)) \cdot P(k_j)} \tag{4.2}$$

L'efficacité des attaques templates repose principalement sur la précision des estimations des fonctions de densité de probabilités. En pratique, il y a de

nombreuses façons de construire ces templates : par recherche de points d'intérêts, pour des valeurs intermédiaires, avec un modèle de fuite, par analyse de corrélation, etc (cf. [MOP07, SLP05, OM07, EGH07, SGJB11]).

4.3 Attaques actives

Dans le cadre des attaques par canaux auxiliaires, les attaques actives désignent principalement les injection de fautes. Les attaques par fautes ont été introduites par Boneh, DeMillo et Lipton en 1997 ([BDL97]) concernant des cryptosystèmes à clef publique. Elles ont ensuite été étendues aux cryptosystèmes à clef secrète par Biham et Shamir ([BS97]).

L'accès direct à la puce est souvent requis pour ce genre d'attaques. Dans ce cas, on doit procéder à une décapsulation, i.e. que les couches protectrices doivent être enlevées pour exposer la tranche de silicone ([Sko05]). L'intégrité du dispositif étant altéré, on dit alors que l'attaque est invasive. De plus, la puce étant souvent constituée de plusieurs couches, l'étape de décapsulation peut être suivie d'une étape de *deprocessing*, pour avoir accès aux différentes couches et observer la structure interne de la puce. Remarquons qu'une fois le semi-conducteur entièrement accessible, de nombreuses techniques sont alors possibles pour récupérer les données secrètes en fonction de la technologie de fabrication de la puce : rétroingénierie, extraction des données en mémoire, microsondage, etc. Le circuit de la puce peut aussi être modifié en coupant un fil conducteur, ou au contraire en reliant deux points du circuit, en localisant et détruisant des fusibles de protection, en modifiant le contenu de certaines mémoires, etc (cf. [Sko05]). Une bonne connaissance du circuit semble cependant un préalable nécessaire à toutes ces techniques.

L'injection de fautes est provoqué par la modification, permanente ou temporaire, de l'état de quelques cellules mémoires. Différentes méthodes ont été proposées pour injecter des fautes en exploitant les propriétés physiques des composants : radiations, champ magnétique, flash lumineux, laser, variations rapides de la tension électrique ou du signal d'horloge (*glitchs*), etc. Les modifications mémoires peuvent être difficiles à prévoir. Or, les conséquences dépendent beaucoup des zones atteintes (RAM, bus mémoire, registres CPU, ...), et des données qu'elles contiennent à ce moment-là. Par exemple, une faute pourrait modifier une mémoire contenant une variable intermédiaire ou une instruction machine, ce qui pourrait provoquer une erreur de calcul ou perturber une exécution de code.

Les injections de fautes peuvent conduire en particulier à la corruption des calculs cryptographiques lorsqu'ils sont effectués sur une puce non proté-

gée. Les attaques par fautes consistent alors à analyser les résultats erronés obtenus afin de retrouver les informations secrètes. Mais la plupart des attaques par fautes nécessitent pour cela des hypothèses précises sur la faute, qu'on appelle modèle de faute, et qui décrit formellement l'effet de la faute provoquée. Un modèle de faute précise généralement qu'elles sont les données qui ont été corrompues et comment la faute affecte ces données. Des modèles communs considèrent la corruption d'un bit, d'un octet ou d'un mot machine particulier.

Une fois le modèle de faute choisi et motivé (description d'une réalisation réaliste en pratique), il reste à décrire théoriquement comment l'analyse peut exploiter l'information apportée par les fautes. Dans le cas des cryptosystèmes symétriques, l'analyse différentielle (Differential Fault Analysis ou DFA) est la principale attaque par fautes ([BS97, PQ03, BS03, Gir05]). Elle consiste à étudier les différences entre des chiffrés fautés et le chiffré correct, ce qui conduit généralement à réduire le nombre de clefs secrètes possibles. Le nombre de fautes nécessaires et la complexité algorithmique de l'attaque (souvent lié au nombre de clefs secrètes possibles restantes) constituent les principaux paramètres sur l'efficacité des attaques pour un modèle de faute donné.

Deuxième partie

Contributions

Chapitre 5

Attacking (EC)DSA Given Only an Implicit Hint

This chapter presents new results published in the Conference on Selected Areas in Cryptography (SAC 2012) [FGR12].

5.1 Introduction

The security of the main public-key cryptosystems is based on the difficulty of solving certain mathematical problems. In this context, the most commonly used problems come from Number Theory, most notably the integer factorization problem and the discrete logarithm on finite cyclic groups. For instance, an efficient factorization leads immediately to an attack on the RSA cryptosystem. The security of RSA is then partly based on the presumed difficulty of factoring large integers. Indeed, the most efficient published factoring algorithms have subexponential asymptotic running times ([Pom84, Len87, LL93]) and it is not known whether efficient factorization can be done in polynomial time on a classical Turing machine.

Another classical example is the discrete logarithm problem on a finite cyclic group, upon which is based the security of the ElGamal encryption system, the Diffie-Hellman key exchange and the DSA-like signature schemes. Despite the proven fact that a generic algorithm for computing discrete logarithms in any group is necessarily an exponential algorithm ([Tes01, Sho97, Sha71, Pol78, Pol00]), once again, subexponential algorithms are known to solve some instances of the discrete logarithm problem, i.e. on well-defined classes of groups. For instance, discrete logarithms can be computed in subexponential time on the multiplicative group of finite fields

([AD93]) or on some hyperelliptic curves ([ADH94, ADH99]). Conversely, there is no known subexponential algorithm to solve the discrete logarithm problem on the group of rational points of a well-chosen elliptic curve.

Instead of trying to solve directly a hard mathematical problem, we can rather look at which information should be added in order to solve this problem in polynomial time. With this objective in mind, Rivest and Shamir introduced in [RS86] the notion of oracle to formalize this approach and showed that a RSA modulus $N = pq$ of bit size n (with p and q balanced prime factors of n) can be factored in polynomial time as soon as the $n/3$ most significant bits (MSB) from one of the factors are known. This result was next improved with the so-called Coppersmith's method based on lattice basis reduction as introduced in [Cop96a], and which reduced the number of needed bits known to only one-half of the MSB of one of the factors. Beyond the theoretical interest, this additional information can be provided for instance by the help of side channel analysis and the discovery of some leaks of secret information in some cryptographic systems, and brought to the attacker under the form of an oracle.

In this chapter, we focus on the Digital Signature Algorithm (DSA) [FIP94] of which the security is based on the difficulty of computing discrete logarithms. The Elliptic Curve Digital Signature Algorithm (ECDSA) [JMV01] is the elliptic curve variant of DSA. The ElGamal [ElG85] and the Schnorr [Sch90] digital signature schemes are also variants of DSA but are rarely used in practice. Anyway, we note that all results here presented could be applied to any of these variants. Without redefining the DSA-like schemes (see section 1.4 for details), we only recall that each user is associated with a pair of private key/public key, such that the private key is the discrete logarithm in a given group of the public key. The user private key and a randomly generated number, called ephemeral key, are required to compute the signature of a message. The ephemeral key must remain secret and is to be renewed for any new message to be signed.

The first proposal of using an oracle on DSA comes from Howgrave-Graham and Smart in [HGS01] using the LLL lattice reduction algorithm ([LLL82]) to take benefit from the knowledge of a small number of bits in many ephemeral keys. For instance, they show experimentally that if only 8 bits out of 160 bits are known from each ephemeral keys for 30 signed messages, then the secret key is known in less than 10 seconds. However, these results were only heuristics, even though confirmed by experimentation. Nguyen and Shparlinski then presented in [NS02] the first polynomial time algorithm that provably recovers the secret DSA key if about $\log^{1/2}(q)$ LSB (or MSB) of each ephemeral key are known (q denoting the order of

the chosen group, see section 1.4) for a polynomially bounded number of corresponding signed messages. They also show that the case of arbitrary consecutive bits requires much more known bits (about twice as much). Finally, in addition of proving the heuristic attack of [HGS01], Nguyen and Shparlinski ([NS02]) also improved the experimental results of [HGS01] by showing that only 3 known bits of each ephemeral key for 100 signed messages are enough to make the attack feasible. The previous attack is adapted to the case of ECDSA [NS03] and other DSA-like signature schemes like the Nyberg-Rueppel variants of DSA [MNS01]. It is also necessary to mention another analysis that shows the threats associated with the use of private keys generated from an imperfect source of randomness. The attack of Bellare, Goldwasser and Micciancio [BGM97] shows that DSA is totally insecure if private keys are produced by weak pseudo-random number generator such as the Knuth's linear congruential generator. When private keys are smaller than a certain bound, Poulakis proposed in [Pou09] an attack on (EC)DSA using the LLL algorithm and an algorithm to compute the integral points of a class of conics.

Note that unlike the case of RSA, where the oracle gives a way to directly compute the factors of the modulus, all these methods against the DSA-Like cryptosystems bypass the problem of computing discrete logarithm, but rather take advantage of the particular form of the modular equality (1.1) defining the signature (see section 1.4).

At PKC 2009 [MR09], May and Ritzenhofen, in the context of factorization, highly restricted the power of the oracle. They did not assume that the oracle explicitly outputs bits but rather provides only implicit information. This unusual oracle applied against the cryptosystem RSA was formalized as follows : given an RSA modulus $N_1 = p_1 q_1$ as input, the oracle outputs a different RSA modulus $N_2 = p_2 q_2$ such that the factors of N_2 shared a certain amount of bits with the factors of the modulus N_1. The implicit nature of the information given by the oracle is due to the fact that the value of the shared bits remains unknown as long as the modulus N_1 or N_2 are not factored. Surprisingly, May and Ritzenhofen give an efficient lattice-based algorithm that provably factors N_1 and N_2 in quadratic time provided that the two moduli shared enough of their least significant bits (LSB). They also showed that this algorithm extends to an algorithm with more than one oracle query, which improves upon the required number of shared LSB. This cryptanalysis with the help of an implicit oracle was next extended to the case of shared MSB (and both LSB/MSB) in [SM09, FMR10] and the bound on the required number of shared bits was also improved.

In the case of DSA, an attack using an implicit information of a to-

tally different kind was already proposed by Leadbitter, Page and Smart in [LPS04] and made effective in [Tak06a, Tak06b]. From a theoretical point of view, this attack can also be formalized by queries to an oracle which returns a message signed with an ephemeral key of the form $k = y + 2^w y + 2^{2w} x$, i.e. such that the w first bits of k are equal to the w next bits. Experiments show that the repetition of a 4-bit window in the ephemeral keys of 20 signatures is enough to recover the secret key. This attack is motivated by side channel analysis. According to the authors, recovering some relation amongst the bits of the secret keys (called "second order leakage") is much more probable than determining the values of such bits because of implementation protections against side channel analysis. They also described many realistic scenarios where this type of leakage could occur.

Our contribution is to define an attack on DSA-like schemes using implicit information, like in [LPS04], but with an oracle very similar to the one introduced by May and Ritzenhofen ([MR09]). More precisely, we assume that on input of a signed message (m_1, s_1) an oracle outputs different signed message (m_2, s_2) signed with the same secret key and such that the ephemeral keys k_1 and k_2 (used to signed the messages m_1 and m_2 respectively) share a certain amount δ of bits. This oracle only gives implicit information about the bits of the ephemeral keys because the value of the shared bits remains unknown as long as the ephemeral keys stay unknown (or equivalently as long as the secret key stays unknown). In other words, we only know that there are equalities between δ unknown bits of the unknown ephemeral keys used to sign the given messages. We show that this implicit information should be extracted by constructing a lattice which contains a very short vector such that its components yield the secret key. The attack succeeds when this vector is found by the LLL lattice reduction algorithm ([LLL82]), that is when it is small enough. This happens when the ephemeral keys share enough bits δ. This method also works for an arbitrary number n of oracle's queries, each new piece of information decreasing the number of required shared bits.

As usual in lattice basis reduction problems, we have to use the Gaussian heuristic to find a condition on the number of shared bits δ in function of the number of messages n for this vector to be the shortest of the lattice. This condition can be improved by the use of a weighted Euclidean inner product instead of the canonical inner product during the reduction algorithm. A variant of this lattice is also proposed so that the complexity of the attack becomes independent of the secret key size and is polynomial time in n (assuming that the bound on δ is verified). In this case, the lattice is spanned by a set of linearly dependent vectors and the condition on δ is slightly

	Lattice spanned by	constraints on the secret key $\mathbf{a}$	Bounds on the number of shared bits δ function of the number of messages n
Canonical inner product	the basis M (5.5) of section 5.3.1	$\mathbf{a} \leq 2^{N-\delta}$ or exhaustive search on the δ msb (section 5.3.2)	$\delta \geq \frac{2N+(n-1)}{n+1} + \frac{c-\log_2(\frac{n+1}{n})}{2}$ (Theorem 10)
	linearly dependent rows vectors of the matrix (5.7) obtained by removing the second column of M	No constraint ($\mathbf{a}$ can be up to N bits)	$\delta \geq \frac{2N+(n-2)}{n} + \frac{c-\log_2(\frac{n}{n-1})}{2}$ (Theorem 11)
Weighted Euclidean	the basis M (5.5) of section 5.3.1	$\mathbf{a} \leq 2^{N-\delta}$ or exhaustive search on the δ msb (section 5.3.2)	$\delta \geq \frac{N+(n-1)}{n} + \frac{c(n+1)}{2n}$ (Theorem 12a)
	linearly dependent rows vectors of the matrix (5.7) obtained by removing the second column of M	No constraint ($\mathbf{a}$ can be up to N bits)	$\delta \geq \frac{N+(n-2)}{n-1} + \frac{cn}{2(n-1)}$ (Theorem 12b)

TABLE 5.1 – Summary of our results

deteriorated. A summary of our method and the proposed improvements can be found in table 5.1.

As an example of our results, the theorem 12 proves that under the Gaussian heuristic assumption, only 4 LSBs shared on each ephemeral keys of 100 signed messages are enough to make a never-failing attack and that with only 3 LSBs shared, the method needs about 200 signed messages. The result of experiments confirms these theoretical values with a computation time less than 5s and they even show that the number of messages can be most of the time reduced to an amount comparable to which is described in [NS02] despite of the weakness of the oracle. However, these experiments also showed that the success rate of this attack is about 90% when only 1 LSB is shared on each ephemeral key of about 400 signed messages. Interestingly enough, this improves the experimental results of [NS02] where the best experiment corresponds to 3 LSB known (even though LSB are not even known in our contribution).

A lot of possible applications of this kind of attack using implicit information, mostly collected by side channel, are given in [LPS04, MR09, FMR10]. In addition to weak and wrong implementations, we could also think destructive applications with a malicious manipulation of random generators for instance. Active attacks should also be considered.

We talk about these practical scenarios in section 5.2 and so we justify once again the existence of this implicit oracle. Then, the rest of this chapter is organized as follows. In section 5.3, we present our method by beginning with the case of shared MSB and LSB (subsection 5.3.1). Essential improvements are proposed in subsection 5.3.2. In subsection 5.3.3, we present our method when they are many blocks of shared bits. Finally, we present the result of experiments in section 5.4. Throughout this chapter, we will frequently use common results on euclidean lattice, so we refer the reader to the relevant chapter 3. In the same way, we refer to section 1.4 for a description of the (EC)DSA signature algorithm.

5.2 Possible application scenario

As stated in [LPS04] in reference to side channel analysis, “the assumption that an attacker may be able to determine a specific set of bits from the ephemeral secrets is less probable than when the original attacks were first published. It is far more probable that second order, seemingly innocuous information can still be recovered and used by the attacker, even if a defense against the first order leakage is implemented". Indeed, there

are always many situations where implicit information can be found despite the implementation of typically recommended countermeasures. Here, as in [LPS04, MR09, SM09, FMR10], the attacker is only assumed to be able to determine relations of equality between bits of the ephemeral keys. In addition to the three scenarii given as examples in [LPS04] where implicit information is collected by a power analysis or a timing attack (involving a fixed table implementation of elliptic curve point multiplication, address-bit DPA, and cache analysis), we suggest other scenarii which are specifically relevant to our model.

Using invasive attacks, an attacker could lock some bits of the register or memory containing the ephemeral key. This kind of attack largely depends on the implementation and requires a good knowledge of the target, which presupposes at least a partial reverse engineering of the chip. Lasers are then used to cut some wires or to modify the chip (see Skorobogatov thesis [Sko05]). More generally, a lot of fault attacks assume that some bits of the memory are flipped to zero (as in [NNTW05]). But it seems more general to assume that the attacked bits take an indeterminate value.

In addition to weak and wrong implementations, we could also think to destructive applications with a malicious manipulation of random generators for instance. This application could be possible on both embedded systems and software implementation, for instance with physical disturbances, invasive attacks or with malicious softwares. Moreover, the presence of a random number generator testing suite ([Bro11, RSN+10]) does not seem to be an effective countermeasure (see experimental results in section 5.4.3).

5.3 Embedding into a Lattice Problem

In this section, we study the security of (EC)DSA given a set of messages signed with the same secret key, and such that the secret ephemeral keys share a certain amount of bits. We recall that the values of these common bits are unknown to us. Thus, the information about the unknown ephemeral keys are implicitly given by the set of signed messages. In other words, we only know that there are some relations amongst the bits of the ephemeral keys used to sign the messages. To simplify the presentation of this method, the bit length of the modulus q is noted by N (i.e. we have $2^{N-1} < q < 2^N$).

We will show how the secret key can be revealed by lattice basis reduction provided that there are enough messages or relations between the bits of the ephemeral keys. These constraints are estimated by relating our method to the Gaussian heuristic (see Theorem 9 of Chapter 3). Since all the

complexities given in this paper depend on the complexity of computing a shortest vector in a given lattice, we set the following notation.

Notation 1. *The time complexity of computing a shortest vector of a d-dimensional lattice L of $\mathbb{Z}^n$ will be denoted by $\mathcal{C}(d, B)$, where $B = \log \max_i(\|\mathbf{b_i}\|)$. Notice that computing a shortest vector of any lattice is a NP-hard problem ([Ajt98]) called the Shortest Vector Problem (SVP).*

For certain families of lattices (i.e. under certain conditions on lattices), the shortest vector can be computed with the LLL Algorithm. In this case, we have that $\mathcal{C}(d, B) = \mathcal{O}(d^5(d+B)B)$, i.e. polynomial time in d and B ([NS05], see Chapter 3). In this paper, we seek to always stay in this situation.

We first present the case when most significant bits (MSB) and/or less significant bits (LSB) are shared and next the case when blocks of bits are shared.

5.3.1 Shared MSB and LSB

We first assume that we have n messages m_i $(i = 1, \ldots, n)$ with associated signatures (r_i, s_i) such that all the corresponding ephemeral keys $\mathbf{k}_i$ share a total of δ bits between the MSB and LSB independently of i (cf. Figure 5.1). Thus, they are of the form

$$\mathbf{k}_i = \mathbf{k} + 2^t \tilde{\mathbf{k}_i} + 2^{t'} \mathbf{k}' \quad \text{for all } i = 1, \ldots, n \tag{5.1}$$

where

$$0 \leq \mathbf{k} < 2^t, \quad 0 \leq \mathbf{k}' < 2^{N-t'}, \quad \delta = N - t' + t \quad \text{and} \quad 0 \leq \tilde{\mathbf{k}_i} < 2^{N-\delta}$$

with $\mathbf{k}$ and $\mathbf{k}'$ common for all the $\mathbf{k}_i$ (i.e. independent of i).

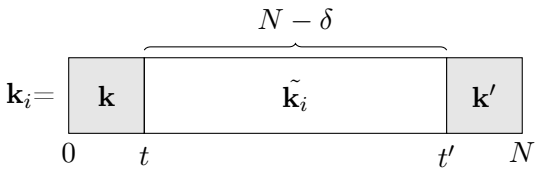


FIGURE 5.1 – Ephemeral keys

Note that all the values of $\mathbf{k}_i$, $\mathbf{k}$, $\tilde{\mathbf{k}_i}$ and $\mathbf{k}'$ are unknown.

In the n equations (1.1) defining the signature

$$\left\{\begin{array}{rcll} m_1 + \mathbf{a}r_1 - s_1\mathbf{k}_1 & \equiv & 0 & (\mathrm{mod}\ q) \\ m_2 + \mathbf{a}r_2 - s_2\mathbf{k}_2 & \equiv & 0 & (\mathrm{mod}\ q) \\ \vdots \quad \vdots & & \vdots & \vdots \\ m_n + \mathbf{a}r_n - s_n\mathbf{k}_n & \equiv & 0 & (\mathrm{mod}\ q) \end{array}\right. \tag{5.2}$$

we substitute the $\mathbf{k}_i$ by (5.1) and eliminate the common variables $\mathbf{k}$ and $\mathbf{k}'$. Then we have

$$\left\{\begin{array}{ll} (s_1^{-1}m_1 - s_2^{-1}m_2) + \mathbf{a}(s_1^{-1}r_1 - s_2^{-1}r_2) - 2^t(\tilde{\mathbf{k}_1} - \tilde{\mathbf{k}_2}) \equiv 0 & (\mathrm{mod}\ q) \\ (s_1^{-1}m_1 - s_3^{-1}m_3) + \mathbf{a}(s_1^{-1}r_1 - s_3^{-1}r_3) - 2^t(\tilde{\mathbf{k}_1} - \tilde{\mathbf{k}_3}) \equiv 0 & (\mathrm{mod}\ q) \\ \vdots \qquad\qquad \vdots \qquad\qquad \vdots & \vdots \\ (s_1^{-1}m_1 - s_n^{-1}m_n) + \mathbf{a}(s_1^{-1}r_1 - s_n^{-1}r_n) - 2^t(\tilde{\mathbf{k}_1} - \tilde{\mathbf{k}_\mathbf{n}}) \equiv 0 & (\mathrm{mod}\ q) \end{array}\right. \tag{5.3}$$

Let α_i, β_i, $\kappa_i \in \mathbb{Z}$ be such that

$$\left\{\begin{array}{rcll} \alpha_i & := & 2^{-t}(s_1^{-1}m_1 - s_i^{-1}m_i) & (\mathrm{mod}\ q) \\ \beta_i & := & 2^{-t}(s_1^{-1}r_1 - s_i^{-1}r_i) & (\mathrm{mod}\ q) \\ \kappa_i & := & \tilde{\mathbf{k}_1} - \tilde{\mathbf{k}_i} & \end{array}\right.$$

then (5.3) becomes

$$\left\{\begin{array}{rcll} \alpha_2 + \mathbf{a}\beta_2 - \kappa_2 & \equiv & 0 & (\mathrm{mod}\ q) \\ \alpha_3 + \mathbf{a}\beta_3 - \kappa_3 & \equiv & 0 & (\mathrm{mod}\ q) \\ \vdots \quad \vdots & & \vdots & \vdots \\ \alpha_n + \mathbf{a}\beta_n - \kappa_n & \equiv & 0 & (\mathrm{mod}\ q) \end{array}\right. \tag{5.4}$$

where $\mathbf{a}$ and κ_i are unknown, β_i and α_i are known. The set of solutions

$$L = \{(x_0, x_1, \ldots, x_n) \in \mathbb{Z}^{n+1} | x_0\alpha_i + x_1\beta_i - x_i \equiv 0 \pmod{q}, \quad i = 2, \ldots, n\}$$

forms an $(n+1)$-dimensional lattice spanned by the row vectors of the following basis matrix

$$M = \begin{pmatrix} 1 & 0 & \alpha_2 & \ldots & \alpha_n \\ 0 & 1 & \beta_2 & \ldots & \beta_n \\ 0 & 0 & q & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \ldots & q \end{pmatrix} \tag{5.5}$$

Note that $v_0 = (1, \mathbf{a}, \kappa_2, \kappa_3, \ldots, \kappa_n)$ is an element of the lattice L. Indeed by (5.4), there are $\lambda_2, \ldots, \lambda_n \in \mathbb{Z}$ such that

$$(1, \mathbf{a}, \lambda_2, \ldots, \lambda_n) \cdot M = v_0 \tag{5.6}$$

If we were able to find this vector v_0 in L, then we could recover the secret key $\mathbf{a}$. Thus, we would like to give some conditions so that the vector v_0 be a short vector in L, and therefore may be obtained by lattice basis reduction.

However, it is easy to see that the norm of v_0 is lower bounded by the secret key $\mathbf{a}$, which can be an integer of roughly N bits. The second component of v_0 is then much bigger than the next ones which are $(N-\delta)$-bit integers. Actually, v_0 has no reason to be a short vector of L while $\mathbf{a}$ is so high. Therefore we will first assume that the secret key $\mathbf{a}$ is smaller than $2^{N-\delta}$, before adapting the lattice to be able to find the secret keys up to N-bit size.

This temporary assumption makes v_0 short in the lattice L, but we are still unable to prove that it is the shortest vector of L. Therefore, the Gaussian heuristic (see Chapter 3), which is usually applied in this situation gives us a way to estimate the required number δ of shared bits in function of the number of available messages so that v_0 is likely to be the shortest vector of L.

Assumption 1. *The Gaussian heuristic (see Theorem 9 of Chapter 3) holds with the lattice L. Thus, if a vector v_0 is shorter than the Gaussian heuristic $\lambda_1(L) \approx \sqrt{\frac{d}{2\pi e}}\,\mathrm{Vol}(L)^{\frac{1}{d}}$ then it is a shortest vector of L.*

Experiments of the section 5.4 confirm this assumption which seems to be true in practice with the lattices proposed in our method.

Theorem 10. *Let n messages m_i $(i = 1, \ldots, n)$ with the associated signatures (r_i, s_i) such that the ephemeral keys $\mathbf{k}_i$ share a total of δ bits between the MSB and LSB. Under assumption 1 and, under the assumption that the secret key $\mathbf{a}$ is smaller than $2^{N-\delta}$, then $\mathbf{a}$ can be computed in time $\mathcal{C}(n+1, \frac{1}{2}\log_2(n-1) + N)$ as soon as*

$$\delta \geq \frac{2N + (n-1)}{n+1} + \frac{1 + \log_2(\pi e) - \log_2(\frac{n+1}{n})}{2}$$

Démonstration. First of all, we find an upper-bound for the norm of v_0. Under our assumptions, each coefficient of v_0 is an integer of about $(N-\delta)$

bits (except the first one which is equal to 1). Thus, we have the following inequality :

$$\|v_0\|^2 \leq \sum_{i=1}^{n} 2^{2(N-\delta)} = 2^{2(N-\delta)+\log_2 n}$$

On the other hand, thanks to the upper-triangular shape of the matrix M, the volume of L is easily computed as $\mathrm{Vol}(L) = q^{(n-1)} > 2^{(N-1)(n-1)}$. We now seek the condition on δ and n under which the norm of v_0 is smaller than the Gaussian heuristic :

$$2^{2(N-\delta)+\log_2(n)} \leq \frac{n+1}{2\pi e} 2^{2(N-1)\frac{n-1}{n+1}}$$

which is equivalent to

$$\delta \geq \frac{2N+(n-1)}{n+1} + \frac{1+\log_2(\pi e) - \log_2(\frac{n+1}{n})}{2}$$

□

5.3.2 Proposal for improvements

Until now, we made the assumption that $\mathbf{a} \leq 2^{N-\delta}$ to ease the exposition. Actually, this assumption is rarely verified by the secret key, so we have to adapt the previous attack to be able to find the secret key up to N-bit long. We suggest three ways, which may be concurrent to each other to reach this goal.

Exhaustive search :

If δ is reasonably small then the method comes down to the previous one with an exhaustive search on the δ-most significant bits of $\mathbf{a}$. Indeed, we have $\mathbf{a} = \tilde{\mathbf{a}} + 2^{N-\delta}a'$ with $\tilde{\mathbf{a}} < 2^{N-\delta}$ and $a' < 2^{\delta}$. An exhaustive search is then made on a' with the previous lattice L in which we set $\alpha_i = 2^{-t}(s_1^{-1}m_1 - s_i^{-1}m_i) + a'2^{-t}(s_1^{-1}r_1 - s_i^{-1}r_i)$. The bound given by the theorem 10 remains true with this method.

Remove the second column :

There is an other way to make the previous attack independent of the size of the secret key, but this trick needs a slightly modification of the LLL

Reduction Algorithm ([Poh87]). Let the lattice L' spanned by the row vectors of the following matrix

$$M' = \begin{pmatrix} 1 & \alpha_2 & \dots & \alpha_n \\ 0 & \beta_2 & \dots & \beta_n \\ 0 & q & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & q \end{pmatrix}, \quad n > 2 \tag{5.7}$$

The matrix M' is the matrix M without the second column. As previously, we have that $v'_0 = (1, \kappa_2, \kappa_3, \dots, \kappa_n)$ is an element of the lattice L'. Indeed, there are $\lambda_2, \dots, \lambda_n \in \mathbb{Z}$ such that

$$(1, \mathbf{a}, \lambda_2, \dots, \lambda_n) \cdot M' = v'_0 \tag{5.8}$$

However, the row vectors of this matrix M' are not linearly independent. Thus, they do not form a basis of the lattice L' and the original LLL algorithm can not be applied directly. In this case, we use the MLLL algorithm ([Poh87]) which is a variant of LLL in which the input vectors can be linearly dependent and has the same complexity as the LLL algorithm.

Remark 2. *Note that the secret key must be read in the transformation vector* $(1, \mathbf{a}, \lambda_2, \dots, \lambda_n)$ *and not in the reduced basis. We could also have removed the first column, but experiments show that the required number of messages is greater in this case.*

The lattice L' used in this case is different from the lattice L of theorem 10. The bound and the volume must be updated.

Lemma 5. *The volume of the lattice* L' *defined by the matrix* M' *given as (5.7) is equal to* q^{n-2}.

Démonstration. The sublattice S of L' spanned by the row vectors of the following matrix

$$\begin{pmatrix} 1 & \alpha_2 & \dots & \alpha_n \\ 0 & q & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & q \end{pmatrix}$$

is a n-dimensional lattice. The dimension of the lattice L' defined by the matrix M' given as (5.7) is equal to the dimension of its sublattice S, and therefore S is a full-rank sublattice of L' (see [NV09]). We consider a lattice

as a group and we have the classical relation between volume and index : $\mathrm{Vol}(S) = \mathrm{Vol}(L')[L' : S]$. Now, it is easy to see that $\mathrm{Vol}(S) = q^{n-1}$ and $[L' : S] = q$, from which we get $\mathrm{Vol}(L') = q^{n-2}$. □

The following theorem is directly derived from this lemma.

Theorem 11. *Let n messages m_i ($i = 1, \ldots, n$, $n > 2$) with associated signatures (r_i, s_i) such that the ephemeral keys $\mathbf{k}_i$ share a total of δ bits between the MSB and LSB (see Figure 5.1). Under the assumption 1, the secret key $\mathbf{a}$ can be computed in time $\mathcal{C}(n, \frac{1}{2}\log_2(n-1) + N)$ as soon as*

$$\delta \geq \frac{2N + (n-2)}{n} + \frac{1 + \log_2(\pi e) - \log_2(\frac{n}{n-1})}{2}$$

The required number of shared bits δ is slightly larger with the new lattice L' (5.7) than the one given in the theorem 10. Experiments confirm this fact but the success rate is now independent of the secret key size (see section 5.4).

Weighted Euclidean inner product :

In order to obtain the v_0 vector (5.6) (or similarly the v'_0 vector (5.8)) within a LLL-reduced basis, we can also use a weighted Euclidean inner product, to take advantage of the knowledge of the components size of the targeted vector. For example, we can take the following inner product of two vectors

$$\langle (x_0, \ldots, x_n), (y_0, \ldots, y_n) \rangle := \sum_{i=0}^{n} x_i y_i 2^{2(N - \lceil \log_2(v_{0,i}) \rceil)}$$

during the LLL algorithm. In practice, this trick drastically reduces the required number of shared bits δ (see section 5.4).

Remark 3. *Weights can be used without needing to change the norm. Indeed, it is equivalent to multiplying all columns of the lattice by the corresponding weight.*

As previously, a bound can be obtained with Gaussian Heuristic.

Theorem 12. *Let n messages m_i ($i = 1, \ldots, n$, $n > 2$) with associated signatures (r_i, s_i) such that the ephemeral keys $\mathbf{k}_i$ share a total of δ bits between the MSB and LSB. Under the assumption 1, the secret key $\mathbf{a}$ can be computed*

a. with the exhaustive search method in time $\delta\mathcal{C}(n+1, \frac{1}{2}\log_2(n)+\delta N)$ *as soon as*

$$\delta \geq \frac{N+(n-1)}{n} + \frac{(n+1)(1+\log_2(\pi e))}{2n} \tag{5.9}$$

b. with the lattice L' *(5.7) in time* $\mathcal{C}(n, \frac{1}{2}\log_2(n-1)+\delta N)$ *as soon as*

$$\delta \geq \frac{N+(n-2)}{n-1} + \frac{n(1+\log_2(\pi e))}{2(n-1)} \tag{5.10}$$

Démonstration. Let an integer $k \geq N-\delta$. We use a weighted Euclidean inner product such that each component of the targeted vector v_0 have the same size k (i.e. the i-th weight is equal to $k - \lceil \log_2(v_{0,i}) \rceil$).

a. With the exhaustive search method, a close approximation of the vector v_0 (5.6) is computed. Its norm is given by

$$\|v_0\|^2 = \sum_{i=1}^{n+1} v_{0,i}^2 2^{2(k-\lceil \log_2(v_{0,i}) \rceil)} \leq \sum_{i=1}^{n+1} 2^{2k} = 2^{2k+\log_2(n+1)}$$

and the volume of the lattice (5.5) is

$$\mathrm{Vol}(L) = 2^k 2^{k-(N-\delta)} (q 2^{k-(N-\delta)})^{(n-1)} \geq 2^{k(n+1)+n(\delta-1)-N+1}.$$

Using the Gaussian heuristic assumption, we have

$$2^{2k+\log_2(n+1)} \leq \frac{n+1}{2\pi e} (2^{k(n+1)+n(\delta-1)-N+1})^{\frac{2}{n+1}}$$

which is equivalent to (5.9).

b. We apply the same method with the n-dimensional lattice L' (5.7) and the seek vector v'_0 (5.8). We obtain

$$\|v'_0\|^2 = \sum_{i=1}^{n} v'^2_{0,i} 2^{2(k-\lceil \log_2(v'_{0,i}) \rceil)} \leq \sum_{i=1}^{n} 2^{2k} = 2^{2k+\log_2(n)}$$

and

$$\mathrm{Vol}(L') = \frac{2^k (q 2^{k-(N-\delta)})^{(n-1)}}{q} \geq 2^{n(k-1)+\delta(n-1)-N+2}.$$

The Gaussian heuristic assumption gives

$$2^{2k+\log_2(n)} \leq \frac{n}{2\pi e} (2^{n(k-1)+\delta(n-1)-N+2})^{\frac{2}{n}}$$

which is equivalent to (5.10).

Note that both results are independent of the variable k. □

5.3.3 Blocks of shared bits

The previous attack can be generalized to the case of ephemeral keys sharing several blocks of bits. Thus, we now assume that we have n messages m_i $(i = 1, \ldots, n)$ with associated signatures (r_i, s_i) such that the ephemeral keys $\mathbf{k}_i$ share a total of δ bits dispatched between l blocks of bits. We denote by δ_i the number of bits of the i-th block $\mathbf{b}_i$ at position p_i (see Figure 5.2).

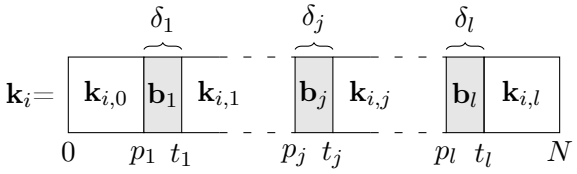


FIGURE 5.2 – Ephemeral keys

For convenience we simplify the notation as follows : let $\underline{t} = (t_1, \ldots, t_l)$ be a l-tuple of integers then we set $2^{\underline{t}} = (2^{t_1}, \ldots, 2^{t_l})$ Then the ephemeral key is of the form

$$\mathbf{k}_i = 2^{\underline{p}} \cdot \underline{\mathbf{b}} + 2^{\underline{t}} \cdot \underline{\mathbf{k}_i} \quad \text{for all } i = 1, \ldots, n \tag{5.11}$$

where $\underline{\mathbf{b}} = (\mathbf{b}_1, \ldots, \mathbf{b}_l)$ is the vector of shared bits blocks, with the position vector $\underline{p} = (p_1, \ldots, p_l)$ of the l blocks, and $\underline{\mathbf{k}_i} = (\mathbf{k}_{i,0}, \ldots, \mathbf{k}_{i,l})$ the vector of no shared bits blocks at positions $\underline{t} = (t_0, \ldots, t_l)$. After stating that $t_0 := 0$ and $p_{l+1} := N$, it follows that for all $i = 1, \ldots, n$, and for all $j = 1, \ldots, l$, we must have

$$t_j = p_j + \delta_j, \quad \delta = \sum_j \delta_j, \quad 0 \leq \mathbf{b}_j < 2^{\delta_j} \quad \text{and} \quad 0 \leq \mathbf{k}_{i,j} < 2^{(p_{j+1} - t_j)}$$

Note that the values of $\mathbf{k}_i$, $\mathbf{k}_{i,j}$ and $\mathbf{b}_j$ are all unknown.

In the n signature equations (1.1), we substitute the $\mathbf{k}_i$ by (5.11) and we eliminate the common variable $\underline{\mathbf{b}}$ then we have

$$\begin{cases} (s_1^{-1}m_1 - s_2^{-1}m_2) + \mathbf{a}(s_1^{-1}r_1 - s_2^{-1}r_2) - \sum_{j=0}^{l} 2^{t_j}(\mathbf{k}_{1,j} - \mathbf{k}_{2,j}) \equiv 0 (\mathrm{mod} q) \\ (s_1^{-1}m_1 - s_3^{-1}m_3) + \mathbf{a}(s_1^{-1}r_1 - s_3^{-1}r_3) - \sum_{j=0}^{l} 2^{t_j}(\mathbf{k}_{1,j} - \mathbf{k}_{3,j}) \equiv 0 (\mathrm{mod} q) \\ \quad \vdots \qquad\qquad \vdots \qquad\qquad\qquad \vdots \qquad\qquad\qquad \vdots \qquad\qquad \vdots \\ (s_1^{-1}m_1 - s_n^{-1}m_n) + \mathbf{a}(s_1^{-1}r_1 - s_n^{-1}r_n) - \sum_{j=0}^{l} 2^{t_j}(\mathbf{k}_{1,j} - \mathbf{k}_{n,j}) \equiv 0 (\mathrm{mod} q) \end{cases} \tag{5.12}$$

Let $\alpha_i,\ \beta_i \in \mathbb{Z}$ and $\underline{\kappa_i}, \underline{t} \in \mathbb{Z}^l$ be such that

$$\begin{cases} \alpha_i & := & (s_1^{-1}m_1 - s_i^{-1}m_i) \pmod q \\ \beta_i & := & (s_1^{-1}r_1 - s_i^{-1}r_i) \pmod q \\ \underline{\kappa_i} & := & (\mathbf{k}_{1,1}, \ldots, \mathbf{k}_{1,l}) - (\mathbf{k}_{i,1}, \ldots, \mathbf{k}_{i,l}) \\ \underline{t} & := & (t_1, \ldots, t_l) \end{cases}$$

then (5.12) becomes

$$\begin{cases} \alpha_2 + \mathbf{a}\beta_2 - 2^{\underline{t}} \cdot \underline{\kappa_2} & \equiv & \mathbf{k}_{1,0} - \mathbf{k}_{2,0} & \pmod q \\ \alpha_3 + \mathbf{a}\beta_3 - 2^{\underline{t}} \cdot \underline{\kappa_3} & \equiv & \mathbf{k}_{1,0} - \mathbf{k}_{3,0} & \pmod q \\ \quad\vdots \quad\quad \vdots & & \vdots & \quad\vdots \\ \alpha_n + \mathbf{a}\beta_n - 2^{\underline{t}} \cdot \underline{\kappa_n} & \equiv & \mathbf{k}_{1,0} - \mathbf{k}_{n,0} & \pmod q \end{cases} \tag{5.13}$$

where $\mathbf{a}$ and $\underline{\kappa_i}$ are unknown, β_i and α_i are known. Embedding these equations into the lattice L spanned by the row vectors of the following basis matrix

$$M = \left(\begin{array}{c|c} I_{l(n-1)+2} & \begin{array}{ccc} \alpha_2 & \ldots & \alpha_n \\ \beta_2 & \ldots & \beta_n \end{array} \\ \hline & \begin{array}{c} 2^{t_1} I_{(n-1)} \end{array} \\ \hline & \begin{array}{c} \vdots \end{array} \\ \hline & \begin{array}{c} 2^{t_l} I_{(n-1)} \end{array} \\ \hline 0 & qI_{(n-1)} \end{array}\right) \tag{5.14}$$

we obtain that

$$\begin{aligned} v_0 &= (1, \mathbf{a}, \mathbf{k}_{1,1} - \mathbf{k}_{2,1}, \ldots, \mathbf{k}_{1,l} - \mathbf{k}_{n,l}, \mathbf{k}_{1,0} - \mathbf{k}_{2,0}, \ldots, \mathbf{k}_{1,0} - \mathbf{k}_{n,0}) \\ &= (1, \mathbf{a}, \underbrace{\kappa_{1,1}, \ldots, \kappa_{n,1}}_{\kappa_{i,1}}, \ldots, \underbrace{\kappa_{i,1}, \ldots, \kappa_{i,l}}_{\kappa_{i,j}}, \ldots, \\ &\qquad\qquad \ldots, \underbrace{\mathbf{k}_{1,0} - \mathbf{k}_{2,0}, \ldots, \mathbf{k}_{1,0} - \mathbf{k}_{n,0}}_{\alpha_i + \mathbf{a}\beta_i - 2^{\underline{t'}} \cdot \underline{\kappa_i} \bmod q}) \end{aligned} \tag{5.15}$$

is an element of the lattice L. If we were able to find this vector v_0 in L, then we could recover the secret key $\mathbf{a}$.

Example 15. *For instance, if we have only one block (i.e. $l = 1$) of δ shared bits in the middle, then the n ephemeral keys are of the form*

$$\mathbf{k}_i = \mathbf{k}_{i,0} + 2^{p_1}\mathbf{b}_1 + 2^{t_1}\mathbf{k}_{i,1} \quad \textit{for all } i = 1, \ldots, n$$

where

$$\delta = \delta_1, \quad t_1 = p_1 + \delta_1, \quad 0 \leq \mathbf{k}_{i,0} < 2^{p_1}, \quad 0 \leq \mathbf{k}_{i,1} < 2^{N - t_1} \quad \textit{and} \quad 0 \leq \mathbf{b}_1 < 2^{\delta_1}$$

and α_i, β_i, $\kappa_i \in \mathbb{Z}$ *be such that*

$$\begin{cases} \alpha_i & := & (s_1^{-1}m_1 - s_i^{-1}m_i) \bmod q \\ \beta_i & := & (s_1^{-1}r_1 - s_i^{-1}r_i) \bmod q \\ \kappa_i & := & \mathbf{k}_{1,1} - \mathbf{k}_{i,1} \end{cases}$$

In this case, (5.13) becomes

$$\begin{cases} \alpha_2 + \mathbf{a}\beta_2 + 2^{t_1}\kappa_2 & \equiv & \mathbf{k}_{1,0} - \mathbf{k}_{2,0} & (\text{mod } q) \\ \alpha_3 + \mathbf{a}\beta_3 + 2^{t_1}\kappa_3 & \equiv & \mathbf{k}_{1,0} - \mathbf{k}_{3,0} & (\text{mod } q) \\ \quad\vdots \quad\quad \vdots & & \vdots & \vdots \\ \alpha_n + \mathbf{a}\beta_n + 2^{t_1}\kappa_n & \equiv & \mathbf{k}_{1,0} - \mathbf{k}_{n,0} & (\text{mod } q) \end{cases}$$

and the lattice L *is spanned by the row vectors of the following basis matrix*

$$M = \left(\begin{array}{cc|ccc|ccc} 1 & 0 & 0 & \dots & 0 & \alpha_2 & \dots & \alpha_n \\ 0 & 1 & 0 & \dots & 0 & \beta_2 & \dots & \beta_n \\ \hline 0 & 0 & 1 & \dots & 0 & 2^{t_1} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 & 0 & \dots & 2^{t_1} \\ \hline 0 & 0 & 0 & \dots & 0 & q & \dots & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 0 & 0 & \dots & q \end{array}\right)$$

The targeted element of L *is the vector*

$$\begin{aligned} v_0 &= (1, \mathbf{a}, \kappa_2, \dots, \kappa_n, \mathbf{k}_{1,0} - \mathbf{k}_{2,0}, \dots, \mathbf{k}_{1,0} - \mathbf{k}_{n,0}) \\ &= (1, \mathbf{a}, \kappa_2, \dots, \kappa_n, \lambda_2, \dots, \lambda_n) \cdot M \end{aligned}$$

The proposed improvements of the section 5.3.2 can also be (directly) applied in the case of shared bits blocks. Especially, a weighted Euclidean inner product gives rise to the following estimations of the required number of shared bits in function of the number of blocks and the number of available messages.

Theorem 13. *Let* n *messages* m_i *(*$i = 1, \dots, n$, $n > 2$*) with associated signatures* (r_i, s_i) *such that the ephemeral keys* $\mathbf{k}_i$ *share a total of* δ *bits dispatch between* l *blocks of bits. Under the assumption 1, the secret key* $\mathbf{a}$ *can be computed*

a. with the exhaustive search method (proposed in section 5.3.2) in time $\delta\mathcal{C}((l+1)(n-1)+2, \frac{1}{2}\log_2(n-1))$ *as soon as*

$$\delta \geq \frac{N + (n-1)}{n} + (1 + \log_2(\pi e))\frac{(l+1)(n-1)+2}{2n} \tag{5.16}$$

b. with the lattice L' obtained by removing the second column (5.7) in time $\mathcal{C}((l+1)(n-1)+1, \frac{1}{2}\log_2(n-1))$ as soon as

$$\delta \geq \frac{N+(n-2)}{n-1} + (1+\log_2(\pi e))\frac{(l+1)(n-1)+1}{2(n-1)} \tag{5.17}$$

Démonstration. The proof of this theorem is almost similar as the one developed in Section 5.3.2. Let an integer $k \geq N-\delta$. We use a weighted Euclidean inner product such that each component of the seek vector v_0 have the same size k (i.e. the i-th weight is equal to $k - \lceil \log_2(v_{0,i}) \rceil$).

a. With the exhaustive search method : first note that the dimension of the lattice L defined by (5.14) is

$$\dim(L) = (l+1)(n-1)+2$$

A close approximation of the norm of vector v_0 (5.15) is then computed :

$$\|v_0\|^2 = \sum_{i=1}^{\dim(L)} v_{0,i}^2 2^{2(k-\lceil \log_2(v_{0,i}) \rceil)} \leq \sum_{i=1}^{\dim(L)} 2^{2k} = 2^{2k}((l+1)(n-1)+2)$$

that is

$$\|v_0\|^2 \leq 2^{2k}\dim(L)$$

Next, the volume of the lattice (5.14) is

$$\mathrm{Vol}(L) = 2^k 2^{k-(N-\delta)}(q2^{k-p_1})^{(n-1)} \prod_{j=1}^{l} 2^{(k-(p_{j+1}-t_j))(n-1)}$$

but, we have

$$\begin{aligned}
&(k-p_1)(n-1) + \textstyle\sum_{j=1}^{l}(k-(p_{j+1}-t_j))(n-1) \\
&= (n-1)(k-p_1+\textstyle\sum_{j=1}^{l}(k-p_{j+1}+p_j+\delta_j)) \\
&= (n-1)(k-p_1+lk-\textstyle\sum_{j=1}^{l} p_{j+1} + \sum_{j=1}^{l} p_j + \sum_{j=1}^{l}\delta_j) \\
&= (n-1)(k(l+1)-N+\delta)
\end{aligned}$$

then

$$\mathrm{Vol}(L) = q^{n-1}2^k 2^{k-(N-\delta)} 2^{(n-1)(k(l+1)-N+\delta)} \geq 2^{k((l+1)(n-1)+2)+n(\delta-1)-N+1}$$

that is

$$\mathrm{Vol}(L) \geq 2^{k\dim(L)+n(\delta-1)-N+1}$$

Using the Gaussian heuristic assumption, we have

$$2^{2k+\log_2(\dim(L))} \leq \frac{\dim(L)}{2\pi e}(2^{k\dim(L)+n(\delta-1)-N+1})^{\frac{2}{\dim(L)}}$$

which is equivalent to

$$\delta \geq \frac{N+(n-1)}{n} + (1+\log_2(\pi e))\frac{\dim(L)}{2n}$$

b. Same computation with the lattice L' obtained by removing the second column of (5.14). The dimension of L' is

$$\dim(L') = (l+1)(n-1)+1$$

Then a close approximation of the norm of vector v_0' is

$$\|v_0'\|^2 \leq 2^{2k}((l+1)(n-1)+1) = 2^{2k}\dim(L')$$

Next, the volume of L' is

$$\mathrm{Vol}(L') = 2^k q^{n-2}(2^{k-p_1})^{(n-1)}\prod_{j=1}^{l} 2^{(k-(p_{j+1}-t_j))(n-1)}$$

which can be lower-bounded by

$$\mathrm{Vol}(L') \geq 2^{k((l+1)(n-1)+1)+(n-1)\delta-N-n+2}$$

that is

$$\mathrm{Vol}(L') \geq 2^{k\dim(L')+(n-1)\delta-N-n+2}$$

Then, using the Gaussian heuristic assumption, we have

$$2^{2k}\dim(L') \leq \frac{\dim(L')}{2\pi e}(2^{k\dim(L')+(n-1)\delta-N-n+2})^{\frac{2}{\dim(L')}}$$

which is equivalent to

$$\delta \geq \frac{N+(n-2)}{n-1} + (1+\log_2(\pi e))\frac{\dim(L')}{2(n-1)}$$

□

Bound on δ of theorem	n, number of messages							
	3	4	5	6	7	8	9	10
10	83	67	56	49	43	39	35	32
11	109	83	67	56	49	43	39	35
12a	57	44	36	30	27	24	21	20
12b	84	57	44	36	30	27	24	21

Bound on δ of theorem	n, number of messages											
	15	20	30	40	50	60	70	80	90	100	200	$\to \infty$
10	23	19	14	11	10	9	8	7	7	7	5	$\approx$3.05
11	25	19	14	11	10	9	8	8	7	7	5	$\approx$3.05
12a	14	12	9	8	7	6	6	6	5	5	4	$\approx$3.05
12b	15	12	9	8	7	6	6	6	5	5	4	$\approx$3.05

TABLE 5.2 – Some theoretical minimum for δ

5.4 Experimental results

In order to check the validity and the quality of the bounds on δ, we implemented the methods on the computational algebra system Magma V2.17-1 ([BCP97]). All the tests have essentially validated the Gaussian Heuristic assumption (assumption 1) and the fact that the first gap of the lattices (defined as λ_2/λ_1, see Chapter 3) is high enough so that the shortest vector can be computed with the LLL algorithm. Hence they show the effectiveness of our method. In the following, the length of q is fixed to $N = 160$.

We also verified in subsection 5.4.3 that a weak random generator leaking this kind of implicit information remains undetectable by standard RNG testing suite.

5.4.1 Shared MSB and LSB

First of all, we plot in figure 5.3) the graph of the four bounds on δ given by the theorems 10, 11 and 12 in function of the number of messages. The table 5.2 gives more details by listing some theoretical minimal integer values of the necessary number of LSB/MSB shared bits δ for a given number of messages.

We conducted experiments of this attack when the ephemeral keys have their δ LSB in common. For the same reason as the one explained in [NS02], the results for the cases of MSB or both MSB/LSB are not as good as in the LSB case, about one more bit being required. As the secret key is 160 bits long, then we used the independent key size method by removing the corresponding column of the lattice (see section 5.3.2). Additionally, we use a weighted Euclidean inner product during the LLL algorithm which

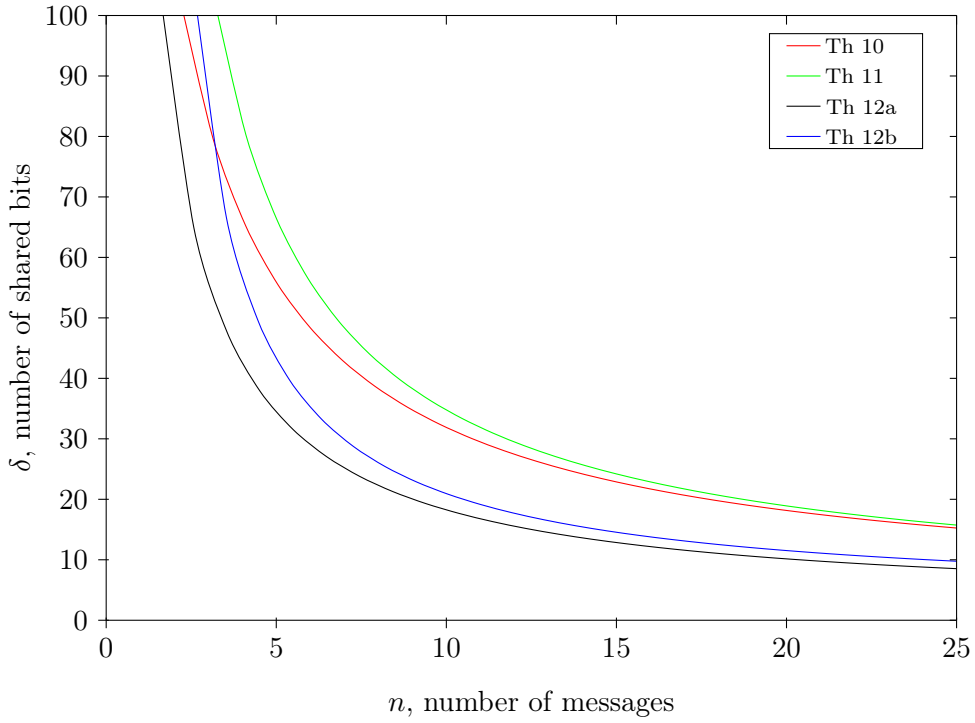


FIGURE 5.3 – Theoretical bounds of Theorems 10, 11 and 12

gives better results than the canonical inner product (more precisely, we use the weighted inner product given as an example in section 5.3.2). The experiments are then conducted under the theorem 12b conditions.

Note that we consider an attack to be successful when the secret key is found, that is when the targeted vector $\pm v_0$ (5.8) is a vector of the reduced basis.

For each δ and each n, we generated 100 tests and store the success rate in table 5.3 and 5.5. To compare experimental values to the theoretical bound, the success rates corresponding to the theoretical minimal value of δ for a given number of messages are written in red.

δ	n, Number of messages										
	150	160	170	180	190	200	250	300	400	500	600
1	0	0	0	2	8	10	35	56	91	99	99
2	58	63	73	80	85	100	100	100	100	100	100
3	99	99	99	99	100	100	100	100	100	100	100
Time (s)	2.9	3.2	3.5	3.8	4.1	4.2	6.3	8.5	15	27	44

δ	n, Number of messages										
	40	50	60	70	80	90	100	110	120	130	140
2	0	0	0	0	0	2	6	12	24	33	42
3	0	2	19	34	60	74	82	94	96	97	99
4	34	76	90	99	99	100	100	100	100	100	100
5	96	100	100	100	100	100	100	100	100	100	100
Time (s)	0.35	0.58	0.78	0.94	1.2	1.4	1.7	1.9	2.1	2.4	2.6

δ	n, number of messages										
	20	21	22	23	24	25	26	27	28	29	30
5	0	0	0	0	0	0	0	1	3	10	16
6	0	0	1	1	5	19	31	56	69	77	86
7	1	9	29	52	78	90	97	100	100	100	100
8	40	68	93	97	99	100	100	100	100	100	100
9	97	99	100	100	100	100	100	100	100	100	100
10	100	100	100	100	100	100	100	100	100	100	100
11	100	100	100	100	100	100	100	100	100	100	100
Time (s)	0.04	0.04	0.05	0.06	0.07	0.08	0.09	0.10	0.12	0.13	0.15

TABLE 5.3 – Success rate of LSB attack under the conditions of theorem 12b

The results show that we have a 100% success rate when δ verify the bound (5.10) of theorem 12 and then we could say that they confirm that the assumptions we made are justified. Moreover, we observe that we can often go a few bits beyond the theoretical bound on δ. Then, the success rate gradually decreases to zero percent with δ.

Another interesting result is that the attack still works with only 1 or 2 shared bits ($\delta = 1, 2$) when there are enough messages. This result is

particularly surprising considering that the theoretical limit is 3 and that the success rate can be higher than 90% (not to say 99% or 100% when $n > 500$).

δ	n, Number of messages										
	150	160	170	180	190	200	250	300	400	500	600
1	2	3	8	8	12	13	20	32	52	60	56
2	22	33	48	53	58	62	82	96	99	100	100
3	93	91	97	98	98	99	100	100	100	100	100
Time (s)	2.9	3.2	3.5	3.8	4.2	4.5	7	9.5	18	34	50

δ	n, Number of messages										
	40	50	60	70	80	90	100	110	120	130	140
2	1	3	0	1	1	7	3	7	10	11	21
3	1	2	4	14	21	39	43	65	81	77	92
4	5	34	68	77	88	93	96	97	100	99	100
5	63	94	99	100	100	100	100	100	100	100	100
6	98	100	100	100	100	100	100	100	100	100	100
Time (s)	0.35	0.58	0.78	0.94	1.2	1.4	1.7	1.9	2.1	2.4	2.6

δ	n, number of messages										
	20	21	22	23	24	25	26	27	28	29	30
5	0	0	0	0	0	0	0	1	1	0	3
6	0	0	0	2	2	3	6	18	29	54	61
7	0	2	12	25	38	58	82	85	90	97	97
8	14	39	71	85	93	99	99	100	99	100	100
9	78	96	99	99	100	100	100	100	100	100	100
10	99	100	100	100	100	100	100	100	100	100	100
11	100	100	100	100	100	100	100	100	100	100	100
12	100	100	100	100	100	100	100	100	100	100	100
Time (s)	0.04	0.04	0.05	0.06	0.07	0.08	0.09	0.10	0.12	0.13	0.15

TABLE 5.4 – Success rate of MSB attack under the conditions of theorem 12b

Note also that the step of lattice reduction of this attack is very fast (always less than a minute, or even less than a second up to $n \approx 70$).

5.4.2 Blocks of shared bits

Following the bound (5.17) of theorem 13 (better than the bound (5.16)), the table 5.6 gives some theoretical minimal integer values of the necessary number of total shared bits δ in function of the number of blocks and the number of messages. Under theses conditions, we conducted a large number of experiments (100 tests for each δ and each n) whose results are summarized

δ	n, number of messages																	
	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
84	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
80	84	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
75	1	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
70	0	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
65	0	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
60	0	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
57	0	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
55	0	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
50	0	1	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
45	0	0	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
44	0	0	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
40	0	0	80	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
39	0	0	45	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
38	0	0	6	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
37	0	0	3	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
36	0	0	0	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
35	0	0	0	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
34	0	0	0	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
33	0	0	0	98	100	100	100	100	100	100	100	100	100	100	100	100	100	100
32	0	0	0	72	100	100	100	100	100	100	100	100	100	100	100	100	100	100
31	0	0	0	39	100	100	100	100	100	100	100	100	100	100	100	100	100	100
30	0	0	0	3	100	100	100	100	100	100	100	100	100	100	100	100	100	100
29	0	0	0	1	100	100	100	100	100	100	100	100	100	100	100	100	100	100
28	0	0	0	0	99	100	100	100	100	100	100	100	100	100	100	100	100	100
27	0	0	0	0	88	100	100	100	100	100	100	100	100	100	100	100	100	100
26	0	0	0	0	46	100	100	100	100	100	100	100	100	100	100	100	100	100
25	0	0	0	0	8	100	100	100	100	100	100	100	100	100	100	100	100	100
24	0	0	0	0	0	99	100	100	100	100	100	100	100	100	100	100	100	100
23	0	0	0	0	1	84	100	100	100	100	100	100	100	100	100	100	100	100
22	0	0	0	0	0	44	100	100	100	100	100	100	100	100	100	100	100	100
21	0	0	0	0	0	5	100	100	100	100	100	100	100	100	100	100	100	100
20	0	0	0	0	0	0	83	100	100	100	100	100	100	100	100	100	100	100
19	0	0	0	0	0	0	24	100	100	100	100	100	100	100	100	100	100	100
18	0	0	0	0	0	0	4	93	100	100	100	100	100	100	100	100	100	100
17	0	0	0	0	0	0	0	35	98	100	100	100	100	100	100	100	100	100
16	0	0	0	0	0	0	0	4	80	100	100	100	100	100	100	100	100	100
15	0	0	0	0	0	0	0	0	16	96	100	100	100	100	100	100	100	100
14	0	0	0	0	0	0	0	0	0	50	97	100	100	100	100	100	100	100
13	0	0	0	0	0	0	0	0	0	3	45	100	100	100	100	100	100	100
12	0	0	0	0	0	0	0	0	0	0	3	58	97	100	100	100	100	100
11	0	0	0	0	0	0	0	0	0	0	0	7	45	96	99	100	100	100
10	0	0	0	0	0	0	0	0	0	0	0	0	0	30	77	98	100	100
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	11	46	85	100
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	6	40
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
Time (s)	.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.01	0.01	0.01	0.01	0.02	0.02	0.03	0.03	0.04

TABLE 5.5 – Success rate of LSB attack under the conditions of theorem 12b

in the table 5.7. Once more, we wrote in red the success rates corresponding to the theoretical minimal value of δ for a given number of messages.

Contrary to the case of shared LSB/MSB, we may have a success rate lower than 100% when δ was within the bound with the attack of blocks of shared bits (see Section 5.3.3). The reason is that the assumption of the theorem 13 may fail because of the occurrence of exceptionally short vectors. However, we observe that we can always go a few bits beyond the theoretical bound on δ keeping a good success rate.

Number of blocks l	n, number of messages												
	3	4	5	6	7	8	9	10	20	30	40	50	∞
1	86	59	46	38	32	29	26	23	14	11	10	9	6
2	88	61	48	40	34	31	28	26	16	13	12	11	8
3	90	63	50	42	37	33	30	28	18	15	14	13	10
10	105	78	64	56	51	47	44	42	32	30	28	27	24
20	125	98	85	77	71	67	65	62	53	50	49	48	44
30	145	119	105	97	92	88	85	83	73	71	69	68	65

TABLE 5.6 – Theoretical minimum for δ with theorem 13b

δ	n, number of messages								
	20	30	40	50	60	70	80	90	100
4	0	0	0	6	13	19	28	26	49
5	0	0	19	32	49	67	78	82	77
6	0	13	49	78	89	90	94	100	99
7	1	63	89	96	99	97	99	97	96
8	20	93	99	99	100	99	98	98	98
9	59	99	100	99	99	100	99	100	99
10	94	100	100	100	100	99	99	100	100
Time (s)	0.3	0.8	1.4	2.1	3.0	4.1	5	6	6.8

TABLE 5.7 – Success rate of theorem 13b with one block

We can also note that the computation time is longer than in the case LSB/MSB because of the larger dimension of lattices.

5.4.3 Random number generator tests

In the scenario with malicious PRNG, we verified that a defect is experimentally undetectable by conventional tests. Indeed, a 8 GByte bit sequence from the AES_OFB random number generator in Dieharder ([Bro11]), manipulated to contain enough implicit information, was used as input for the Dieharder test suite. More precisely, sequences of some randomly selected

bits are repeated in a predictable way. For instance, a sequence of 4 bits can be repeated 100 times following a predictable pattern in a random sequence corresponding to 2^{10} ephemeral keys. The pattern that describes the positions of corrupted nonces (i.e. sharing some bits) can be, for example, a function of the position of the first corrupted nonce. Therefore, in this case we need an additional step containing an exhaustive search to find the first corrupted nonce (of complexity of about 2^{10} with this example). All tests of the two referenced statistical test suites Dieharder statistical test suite ([Bro11]) and the NIST statistical test suite (STS) ([RSN+10]) have shown a random behavior at a high confidence level, our manipulations being then unnoticed when the number of shared bits matches the number of corrupted nonces to have a 100% success rate (see Table 5.3). These experiments show that these tests are not a proof of randomness even though they are common tools for initial validation. Finally, it has been shown that an exploitable bias is currently undetectable by conventional statistical tests.

5.5 Further developments

Throughout this work, we assumed that all ephemeral keys used in the attack shared a same block of bits. In this scenario, by the pigeonhole principle, our attack needs, in the worst case, $2^{\delta} + 1$ samples before obtaining only two signatures that have ephemeral keys with δ bits in common. However, from a practical perspective, we would like to use all the signatures generated by a signer, which is not possible for the moment. Assuming, as in [LPS04], that an attacker can determine, in practice, some relation amongst the bits of the secret ephemeral keys rather than their specific values, we present below how to solve this problem by slightly adapting the lattices of our method.

In this context, our attack can be naturally extended to the more general case where each ephemeral key $\mathbf{k}_i$ $(i = 1 \dots n)$ share δ bits with at least one other key $\mathbf{k}_j$ $(i \neq j)$ and not necessarily with all of them. For instance, we look at shared MSB/LSB, with the same notation as in section 5.3.1. For two fixed positions t and t', we can take the partition P of the set of all ephemeral keys corresponding to the equivalence relation $\mathcal{R}_{t,t'}$, defined such that related ephemeral keys share the first t MSB and the last t' LSB (which represent a total of δ bits). In a given equivalence class $[\mathbf{k}_j]$, if we have $\#[\mathbf{k}_j] \geq 2$ then we can apply the method of Section 5.3.1. For each

class $[\mathbf{k}_j]$, we obtain a system of $\#[\mathbf{k}_j] - 1$ modular equations as (5.4) with

$$\forall \mathbf{k}_i \in [\mathbf{k}_j] \text{ s.t. } i \neq j, \begin{cases} \alpha_i & := & 2^{-t}(s_j^{-1} m_j - s_i^{-1} m_i) & \mod q \\ \beta_i & := & 2^{-t}(s_j^{-1} r_j - s_i^{-1} r_i) & \mod q \\ \kappa_i & := & \tilde{\mathbf{k}_j} - \tilde{\mathbf{k}_i} \end{cases}$$

The set of all common solutions of the $\#P = \#\left(\{\mathbf{k}_i, i = 1..n\}/R_{t,t'}\right)$ systems described as above forms a lattice similar to (5.5) of dimension equal to $n - \#P$.

In the same way, other shared bits in other positions (i.e. when t and t' are not fixed) can be exploited by expanding the lattice with the corresponding columns. Also the same improvement can be developed in the case of shared bits blocks in the middle.

Another interesting problem is the study of the difficulty of the discrete logarithm problem when the partial information is related to the secret key instead of ephemeral keys. In this case, It seems that implicit information could be used similarly to the case of consecutive known bits [GTY07].

More generally, we could also imagine other forms of implicit information, i.e. an other relationships than just the equality between bits. For instance, an interesting open question is whether we can exploit inequalities or simple relationships between unknown bits.

Chapitre 6

Analysis of the Algebraic Side Channel Attack

This chapter gathers results previously published in the Journal Of Cryptographic Engineering [CFGR12] and in the workshop on Constructive Side-Channel Analysis and Secure Design 2011 [FGR11].

6.1 Introduction

Algebraic Side Channel Attacks (ASCA) are a new kind of attack recently introduced in [RSVC09] by Renauld, Standaert and Veyrat-Charvillon. It is a natural combination of classical algebraic cryptanalysis and side channel attacks which takes full advantage of both classical attacks. It should be mentioned that several methods combining side channel and algebraic attacks (see [Bog07, BKP08] for the first algebraic collision attacks) or differential attacks (see [SWP03, HP06, Bog08]) have already been suggested. As for these methods, the main idea of ASCA is to begin with an on-line phase where leakage information is recorded by a side channel, and to end with a powerful off-line phase where this data is used by algebraic cryptanalysis to recover the key. In contrary to standard Differential Power Analysis, the goal of the on-line phase is not to recover directly a bit of the key but it is only to catch a lot of partial information on the intermediate data manipulated during the encryption. Thus, all the leakages of all the cipher rounds are potentially useful. Contrary to the articles [SWP03, KSP04, HP06, Bog07, BKP08, Bog08, MME10], ASCA could succeed with the observation of a single encrypted plaintext and would work

with completely masked implementations ([RSVC09]). During this on-line phase, a leakage model is selected, for example a common leakage model is the Hamming weight of data transiting on a bus (see for instance [CJRR99, ABDM00] for a discussion of this model). Next, the off-line phase makes use of the collected leakage information in an algebraic attack. In the present study, the side channel information is assumed to be reliable for use in the algebraic attack phase, which is not the case with real measures because of the presence of noise. Even though recent works have shown how algebraic approaches may deal with errors ([OKPW10, AC10]), our goal is to start to explain and to understand the efficiency of ASCA, and more precisely the algebraic phase of ASCA with reliable leakage information. This algebraic attack phase consists of modeling the cryptosystem and the leakage model by a system of polynomial equations. Solving this system is equivalent to recovering the bits of the key. In classical algebraic cryptanalysis, solving the system of equations representing a modern block cipher remains a source of speculation because of the complexity of solving such polynomial systems. On the contrary, the system of equations obtained with the algebraic collision attacks ([Bog07, BKP08]) has been well detailed so that the complexity of resolution of such systems is well understood. On the other hand, in the ASCA context, the leakage model seems to provide enough information to efficiently solve in practice the system of equations, but the apparent simplicity of this solving step remained unexplained and its computational complexity was not enough analyzed.

In [RS09a] and [RSVC09], algebraic side-channel attacks are evaluated against 8-bit implementations of PRESENT and AES. The main leakage model studied is the Hamming weight model. Thus, the authors of [RS09a, RSVC09] (as in [HP06]) assume the knowledge of the Hamming weights of some intermediate computations. The system of equations representing the block cipher and the leakage model is translated into a satisfiability problem and solved by a SAT solver. Under these assumptions, this attack seems very powerful. Indeed, the key is always recovered in less than one minute if all the 8-bit Hamming weights after the XORs (in AddRoundKey and MixColumns functions) and after the substitution layers are known for a 31-round PRESENT and for a 10-round AES. When fewer Hamming weights are known, the number of consecutive rounds with Hamming weights is an important criterion for a successful attack (see figures 6.1 and 6.2). There are also some effective attacks in unknown plaintext/ciphertext situations or against masked implementations. It is clear that the known Hamming weights allow to exclude most of the possible values of the key, however, the success rate of these attacks depends on several parameters : the amount

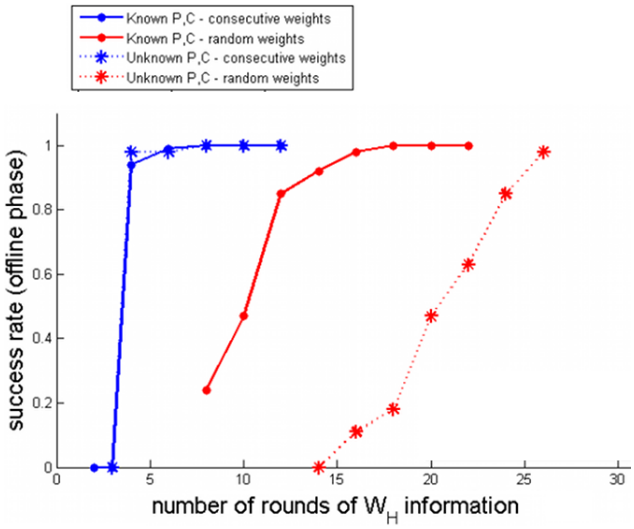


FIGURE 6.1 – Results of ASCA (taken from[RS09a]) with the SAT solver zChaff against 31-round PRESENT with partial WH leakages.

of available information, the leakage function or the shape of the system of equations. All these results are also very dependent on the heuristics used in the SAT solver, and so the experiments are very difficult to explain when SAT solver techniques are used.

The main goal of this chapter is to explain the effectiveness of this attack, to describe the criterion of success and therefore to find the theoretical conditions to prevent algebraic side channel attacks. To achieve this goal, Gröbner techniques are used instead of a SAT solver because of their computation without heuristics and so, more stable and more understandable. We also assume the same hypothesis as in [RS09a, RSVC09], particularly that an initial on-line phase provides a sequence of leakage information, and we only focus on the algebraic cryptanalysis phase. Furthermore, we do not discuss about side channel countermeasures, and we refer to [RS09a, RSVC09] for detailed discussions. We show in section 6.2 that the complexity of the Gröbner basis computation in these attacks depends on a new notion of algebraic immunity and on the distribution of leakage information. This Algebraic Immunity with Leakage is defined by the degree and also the number of lowest degree relations which are given by a black box (S-Boxes, Key derivation, etc) and its leakage information. This new algebraic immunity is completely related to the complexity of the Gröbner basis computation and thus, re-

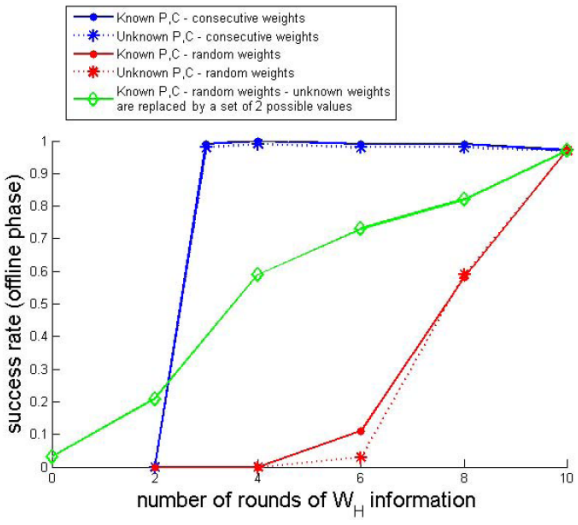


FIGURE 6.2 – Success rate of ASCA (taken from[RSVC09]) against an unprotected implementation of the AES in function of the amount of exploited leakages. One round of side-channel information is equivalent to 84 known Hamming weights.

presents a good criterion for effective Gröbner attacks. From this theoretical study and this new criterion, we deduce a new invariant, which could also be connected to SAT solvers efficiency. For a given block cipher, this invariant, denoted by N_B, is easily computed by a local study of the black boxes B defining the cryptosystem. More precisely, if we denote by B the black box, L the leakage model, l a value of the function L, and n the bus size, then the invariant is the function $N_B(l)$ giving the cardinality of the corresponding set $\{x \in \mathbb{F}_2^n \text{ s.t. } \mathrm{L}(x, B(x)) = l\}$. We prove that the number of linear relations given by the leakage l is greater or equal than $2n + 1 - N_B(l)$ (in practice, this bound is often achieved). Thus, from this new invariant, we deduce a sufficient condition for the weakness of a black box under a Gröbner attack with leakage. This condition corresponds to the case where the black box is such that $N_B(l)$ is small for a lot of values of l. We verify this condition in practice by using Gröbner basis techniques and some of the best SAT solvers. Thus, this invariant may be seen as a general algebraic sufficient condition (independent of the solving strategy) for an effective algebraic side channel attack. Even if this invariant does not provide a theoretical necessary condition of weakness, we successfully describe several scenarios of unsuccessful Gröbner and SAT solver attacks when $N_B(l)$ is large. For example, if the S-boxes are replaced by functions maximizing the function N_B then both algebraic attacks become impractical. The same holds when all leakage data maximizes N_B.

From this new theoretical point of view, we analyze the precedent results on ASCA presented in [RS09a, RSVC09] which were heuristic. To simplify and to keep the point of view of [RS09a, RSVC09], we start to study the 8-bit Hamming weight leakage. As most results of this chapter are valid independently of the leakage model, other leakage models of interest are discussed in the chapter 7. The results presented in Section 6.6 show that, with the Hamming weight model, the AES, PRESENT, CAMELLIA and SMS4 S-boxes are very weak with respect to this invariant : N_B is often small and the number of linear equations is on average 8 per S-box. The local study of these S-boxes shows that Hamming weight leakage information can be used to partly linearize the polynomial system of equations. Moreover, if N_B is very small then few input or output bits of the S-box can be recovered. In section 6.4, these local results are used to explain the recovery of the key bits. Especially in the case of consecutive rounds of Hamming weight leakages, the subkey bits can be easily deduced from previously recovered bits. Or else, we show that the system of equations representing the block cipher and the Hamming weight information contains enough linear equations to be efficiently solved : the probability of obtaining at least 64 (resp. 130) linear

relations is about 50% for PRESENT (resp. AES) for example. Moreover, we show that if the number of rounds increases, the number of linear relations which provide subkey bit increase (see Section 6.4.3). Consequently, this work fully explains the efficiency of the attack. Thanks to this understanding, an efficient solving strategy is developed for Gröbner attacks (Section 6.4). In section 6.5, the conditions for preventing algebraic side channel attacks are also discussed, and it seems that one of the safest way to design a block cipher resistant against all kinds of attacks is to increase the bus size.

6.2 Algebraic Cryptanalysis and side channel information

The basic principle of Algebraic Cryptanalysis is to model a cryptographic primitive by a set of algebraic equations over a finite field. The system of equations is constructed in such a way as to have a correspondence between the solutions of this system, and the secret information of the cryptographic primitive (for instance, the secret key of a block cipher). There are different ways to solve such a polynomial system : SAT solver, Gröbner basis, XSL([CP02]), resultant based methods, etc. In this chapter, we particularly use the Gröbner basis method, a powerful tool for solving a polynomial system. We refer to [BFS04, Bar04, BFSY05] for a discussion on the complexity of Gröbner basis computation of overdetermined algebraic equations over finited fields. The Faugère's F4 and F5 [Fau99, Fau02] algorithms are the most efficient algorithms to compute Gröbner basis, so in our experiments we used the efficient implementation of F4 by Magma software [BCP97]. These algorithms have been successfully applied against a number of multivariate schemes [FJ03, FP06a, FP06b, FdVP08] and in stream cipher cryptanalysis [CM03, Ars05], but they stay unpractical against block ciphers [CP02, CL05]. Indeed, the size of the corresponding algebraic system is so huge (thousand of variables and/or equations with high degree) that nobody is able to predict correctly the complexity of solving such polynomial systems. The degree of these equations stays high because of non-linear substitution-box layers (S-Boxes) and the multitude of rounds. One of the main goals of algebraic attacks is to describe these S-Boxes by low degree equations. The number of such equations gives a criterion to evaluate the block cipher resistance against algebraic attacks and it is called Algebraic Immunity (more precisely, it is also often called graph algebraic immunity, to distinguish it from other notions [AA05, AF05, Ars05, AK06, FM07, Car09, Car10]).

In algebraic side channel attack, we also assume the knowledge of addi-

tional information obtained by side channel, for instance Hamming weights of intermediate values. In the polynomial system modeling of our problem we take into account this assumption. In particular, we see each round of the block cipher as successive black boxes operating on n-bit data (*i.e.* n is the size of the bus). From the knowledge of the polynomial systems representing such a black box and the corresponding Hamming weight leakages, one can model the complete block cipher with leakages as a block diagonal system of equations (each block corresponding to a round). This definition of the model by splitting the different steps of size of n bits is used in our strategy for solving the entire system and it turns out that this algebraic system updated with equations corresponding to the leakage is easier to solve in practice. We will show in this section that the presence of this additional information may give rise to a number of independent linear relations. These relations enable us to mount an effective algebraic attack and that leads us to define a new notion of Algebraic Immunity with Leakage.

From now on, to make this study more general, the S-boxes, or any vectorial Boolean function is seen as a *black box*, denoted by B in the following. Let n be the bus size of B, $X_1, \ldots, X_n$ and $Y_1, \ldots, Y_n$ be respectively its input and output bits. To restrict the study to solutions with coefficients in the field $\mathbb{F}_2$ (and not in its algebraic closure $\overline{\mathbb{F}_2}$), we always add the following set of polynomials corresponding to the classical field equations

$$S_{\text{Field Eq.}} = \{X_i^2 - X_i, Y_i^2 - Y_i, 1 \leq i \leq n\} \tag{6.1}$$

into the polynomial systems. We will denote by $I_{\text{Field Eq.}}$ the ideal of $\mathbb{F}_2[X_1, \ldots, X_n, Y_1, \ldots, Y_n]$ generated by the set $S_{\text{Field Eq.}}$. Finally, the subset $S_B = \{F_1, \ldots, F_{k_B}\} \subset \mathbb{F}_2[X_1, \ldots, X_n, Y_1, \ldots, Y_n]$ is the finite set of Boolean functions defining the outputs of B as a (explicit or implicit) function of its inputs.

6.2.1 Algebraic Immunity for Block ciphers

The notion of Algebraic Immunity often refers to stream ciphers and Boolean functions, but in this chapter, we make reference to the Algebraic Immunity extended to Boolean vectorial functions (sometimes called "graph algebraic immunity"). This definition slightly varies from one article to the next. Thus, we first remind the definition of the Algebraic Immunity which we are going to use and we give an algorithm to compute it (see [AA05, AF05, Ars05, AK06, FM07, Car09, Car10]).

The Algebraic Immunity is defined as the lowest degree of algebraic relations of a Boolean vectorial function. More formally, let $B : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be

a black box and $I_B \subset \mathbb{F}_2[X_1, \ldots, X_n, Y_1, \ldots, Y_n]$ the ideal generated by the equations representing B and the field equations :

$$I_B = \langle S_B \cup S_{\text{Field Eq.}} \rangle$$

Définition 27. *The Algebraic Immunity of B is defined by $AI(B) = min\{deg(P), P \in I_B \setminus I_{Field\ Eq.}\}$*

Remarque 4. *The number of such lowest degree relations is also an important invariant related to I_B, and it is always computed at the same time as the Algebraic Immunity.*

To obtain the Algebraic Immunity of a black box B, we could compute a Gröbner basis of I_B with respect to a graded order([AF05]) :

Théorème 14. *The reduced Gröbner basis G_B of I_B with respect to a graded order contains a linear basis of the lowest relations of B (i.e. the polynomials $P \in I_B$ such that $deg(P) = AI(B)$).*

Démonstration. Every $f \in I_B$ is reduced to zero by a Gröbner basis of I_B. Thus, there is a polynomial $g \in G_B$ such that the leading monomial $LM(g)$ of g divides $LM(f)$. As we have a graded monomial order, $deg(g) = deg(LM(g))$ and $deg(f) = deg(LM(f))$. Thus, $deg(g) \leq deg(f)$ and we prove that G_B contains a linear generated family of the lowest relations of B. Then the definition of a reduced Gröbner basis implies that the linear generated family is a linearly independent family. □

Exemple 16. *The Algebraic Immunity of the function calculating the inverse over $\mathbb{F}_{2^8}$ (e.g. AES S-box) equals 2. Indeed, the inverse function is represented by a set of 39 quadratic equations over $\mathbb{F}_2$ ([CP02]) as well as over $\mathbb{F}_{2^8}$ ([Ars05]).*

We should also mention a proposition which is particularly relevant for the study of S-boxes of block ciphers :

Proposition 8. *The Algebraic Immunity is invariant under CCZ-equivalence of vectorial Boolean functions. Recall that two functions $B, B' : \mathbb{F}_2^n \to \mathbb{F}_2^n$ are called CCZ-equivalent (CCZ means Carlet-Charpin-Zinoviev) if their graphs $G_B = \{(x, B(x)); x \in \mathbb{F}_2^n\}$ and $G_{B'} = \{(x, B'(x)); x \in \mathbb{F}_2^n\}$ are affine equivalent, that is, if there exists an affine permutation $\mathcal{L} = (L_1, L_2)$ of $\mathbb{F}_2^n \times \mathbb{F}_2^n$ such that $\mathcal{L}(G_B) = G_{B'}$, i.e. such that $B(x) = y \Leftrightarrow B'(L_1(x, y)) = L_2(x, y)$.*

6.2.2 Algebraic Immunity of S-boxes with Leakage

In the previous section, the concept of *Algebraic Immunity* is defined as the lowest degree of algebraic relations of a Boolean vectorial function. In the ASCA context, we are looking for the lowest degree relations of B with leakage information (e.g. Hamming weights). Therefore, we need to introduce a slightly different notion of Algebraic Immunity to take the leakage into account. To do so, for every value l of the leakage model, we consider the ideal I_l of $\mathbb{F}_2[X_1, \ldots, X_n, Y_1, \ldots, Y_n]$ generated by the equations representing B, the field equations $S_{\text{Field Eq.}}$ and by L_l the set of equations representing the leakage information l, namely :

$$I_l = \langle S_B \cup L_l \cup S_{\text{Field Eq.}} \rangle \ .$$

From this ideal we can define this new notion of algebraic immunity.

Définition 28. *Let B be a black box and l the value of the leakage model L. The* Algebraic Immunity With Leakage*, denoted by $AI_L(B, l)$, is defined by*

$$AI_L(B, l) = min\{deg(P), P \in I_l \setminus I_{Field\ Eq.}\}$$

The number of linearly independent relations in I_l with degree $AI_L(B, l)$ will be denoted by $\#AI_L(B, l)$.

Similarly to the general notion of Algebraic Immunity, the relations of lowest degree can be explicitly obtained by the computation of a Gröbner basis of I_l with respect to a graded order (see [Ars05]).

An other important invariant related to I_l is the number of points in the associated variety $V(I_l)$, i.e. the set of common roots of the polynomials in I_l. This number which depends on the black box B and on the value l of the leakage function L, is also linked to our Algebraic Immunity With Leakage and it will be denoted by $N_B(l)$:

Définition 29. *$N_B(l)$ is defined as the number of points of the variety $V(I_l)$. In other words, $N_B(l)$ is equal to the cardinality of the set $\{x \in \mathbb{F}_2^n$ s.t. $L(x, B(x)) = l\}$*

We will show in Section 6.3 below that $AI_L(B, l)$ is equal to 1 in the cases of Hamming weight model. In this particular situation where $AI_L(B, l)$ is equal to one (ie. there is at least one linear equation in the Ideal I_l), we prove the following relation between $\#AI_L(B, l)$ and $N_B(l)$:

Proposition 9. *Let n be the bus size of B. If $AI_L(B,l)$ is equal to 1 and $N_B(l)$ is non-zero then*

$$N_B(l) \geq 2n + 1 - \#AI_L(B,l) \ .$$

Démonstration. As observed in Definition 29, we have $N_B(l) = \#V(I_l)$. Since I_l contains the field polynomials, it is radical. Its variety $V(I_l)$ is a subset of $\mathbb{F}_2^n$ and, from the Nullstellensatz, we have

$$\begin{aligned} \#V(I_l) &= \dim(\tfrac{\mathbb{F}_2[X_1,\ldots,X_n,Y_1,\ldots,Y_n]}{I_l}) \\ &= \dim(\mathrm{Span}(m^\alpha : \alpha \in \mathbb{N}^{2n},\ m^\alpha \notin \mathrm{LT}(I_l))) \end{aligned}$$

From this set of generating monomials, we only keep the set F of monomials which have a degree less than or equal than one :

$$F = \{m : \deg(m) \leq 1 \text{ and } m \notin \mathrm{LT}(I_l)\}$$

We have $\#V(I_l) \geq \dim(\mathrm{Span}(F))$. Now, let $E = \{m : \deg(m) \leq 1\}$ and $G = \{m : \deg(m) \leq 1 \text{ and } m \in \mathrm{LT}(I_l)\}$. It is clear that F is the set of monomials in E which are not in $\mathrm{LT}(I_l)$ and G is the set of the monomials in E which are in $\mathrm{LT}(I_l)$, thus the set E is equal to the following disjoint union $E = F \bigsqcup G$. Finally,

$$\begin{aligned} \#V(I_l) &\geq \dim(\mathrm{Span}(F)) \\ &= \dim(\mathrm{Span}(E)) - \dim(\mathrm{Span}(G)) \\ &= 2n + 1 - \#AI_L(B,l) \end{aligned}$$

□

We have defined the Algebraic Immunity with Leakage and showed its relation with $N_B(l)$. In the next section, these results will be useful to study an example of leakage model : the Hamming weight leakage model.

6.3 The Hamming weight leakage model

In [RS09a] and [RSVC09], algebraic side-channel attacks are evaluated against implementations of the block ciphers PRESENT and AES in 8-bit PIC microcontrolers. The main leakage model studied is the Hamming weight model, that is the number of bits set to 1 being processed at a given time (see for instance [CJRR99, ABDM00] for a discussion of this model). Thus the authors of [RS09a, RSVC09] assume the knowledge of the Hamming weights of some intermediate computations. In this section, we also assume that a

first on-line phase already provided leakages as the Hamming weights of the input and output of a black box B. The Algebraic Immunity with Leakage is of great help to study the influence of this additional information on the system of equations generated by B. Actually, we prove that the Algebraic Immunity with Leakage of B with Hamming weight model is equal to 1 for every possible black box B which is exceptionally small. Moreover, we show that there are at least two independent linear equations.

First of all, we need to describe the system of equations representing the Hamming weight of the data $x = (x_1, x_2, \ldots, x_n) \in \mathbb{F}_2^n$. Note that if the Hamming weight of x is equal to w then any product of $w+1$ bits is always null, which corresponds to the following system :

$$R_w : \left\{ \prod_{i \in I} X_i = 0 \ : \ I \subset \{1 \ldots n\} \text{ s.t. } |I| = w+1 \right\}$$

and there is only one product of w bits which is 1, which corresponds to the following system :

$$T_w : \left\{ \sum_{J \subset \{1 \ldots n\} \text{ s.t. } |J| = w} \left(\prod_{j \in J} X_j \right) = 1 \right\}$$

These two facts are sufficient to represent the Hamming weight model by the system of equations

$$L_w = R_w \cup T_w \tag{6.2}$$

Exemple 17. *For instance, the following system of equations is only satisfied by points $x = (x_1 \ldots x_4) \in \mathbb{F}_2^4$ such that $HW(x) = 1$:*

$$L_1 : \begin{cases} x_1 x_2 & = 0 \\ x_1 x_3 & = 0 \\ x_1 x_4 & = 0 \\ x_2 x_3 & = 0 \\ x_2 x_4 & = 0 \\ x_3 x_4 & = 0 \\ x_1 + x_2 + x_3 + x_4 & = 1 \end{cases}$$

Remarque 5. *The only sets L_w which contain a linear equation are the sets L_w with $w = 0$ or $w = 1$. When $w > 1$, these sets L_w never contain linear equations.*

Now that we have formulated the system of equations representing the Hamming weights, we are able to study the Algebraic Immunity With Leakage in the case of the Hamming weight model (see Section 6.2.2). We will show that the addition of these systems of equations representing the Hamming weight to the system modeling a black box B could give a lot of linear equations.

So, we consider the ideal $I_{w_{in},w_{out}}$ of the polynomial ring $\mathbb{F}_2[X_1,\ldots,X_n,Y_1,\ldots,Y_n]$ generated by the equations representing B, the field polynomials $S_{\text{Field Eq.}}$ and by $L_{w_{in}}$, $L_{w_{out}}$ the equations representing the Hamming weights w_{in}, w_{out} respectively (see the definition of L_w (6.2)), namely :

$$I_{w_{in},w_{out}} = \langle L_{w_{in}} \cup L_{w_{out}} \cup S_B \cup S_{\text{Field Eq.}} \rangle$$

In this case, the Algebraic Immunity With Leakage of the black box B with Hamming weight leakage is denoted by $AI_L(B, w_{in}, w_{out})$ and the number of linearly independent relations in $I_{w_{in},w_{out}}$ with degree $AI_L(B, w_{in}, w_{out})$ is denoted by $\#AI_L(B, w_{in}, w_{out})$.

We now prove that $AI_L(B, w_{in}, w_{out}) = 1$, and $\#AI_L(B, w_{in}, w_{out}) \geq 2$, i.e. that $I_{w_{in},w_{out}}$ contains at least two independent linear polynomials. This result is a consequence of the fact that there is always a linear relation in the ideal generated by the equations describing the Hamming weight. Although these two linear relations are not really useful in an algebraic attack, this result shows that the situation is completely different with the classical algebraic immunity.

Lemme 6. *Let $w \in \{0,\ldots,n\}$. The ideal $I_{hw}(w) \subset \mathbb{F}_2[X_1,\ldots,X_n]$, generated by L_w (6.2) and by $S_{\text{Field Eq.}}$, always contains the linear polynomial*

$$X_0 + \cdots + X_n + (w \ mod\ 2) \tag{6.3}$$

Démonstration. The system of equations L_w(6.2) defines the set $V(w) = \{x \in \mathbb{F}_2^n \text{ s.t. } HW(x) = w\} = HW^{-1}(w)$. Then, the radical ([CLO07]) ideal $I_{hw}(w)$ contains all polynomials vanishing on $V(w)$ (which is the variety of $I_{hw}(w)$ over $\mathbb{F}_2$). Clearly, the linear polynomial (6.3) vanishes on $V(w)$, hence it must be in $I_{hw}(w)$. □

We can now prove the following result :

Proposition 10. *Let G be the Gröbner basis of the ideal $I_{w_{in},w_{out}}$ for a graded monomial order. Then G contains at least 2 independent linear polynomials.*

Démonstration. With the same notation as in lemma 6, we could note that there exist $(w_{in}, w_{out}) \in \mathbb{N}^2$ such that $I_{w_{in},w_{out}} = I_B + I_{hw}(w_{in}) + I_{hw}(w_{out})$. According to Lemma 6,

$$f = X_1 + \cdots + X_n + (w_{in} \bmod 2) \in I_{w_{in},w_{out}}$$
$$f' = Y_1 + \cdots + Y_n + (w_{out} \bmod 2) \in I_{w_{in},w_{out}}$$

then there is $g \in G$ such that the leading monomial $LM(g)$ of g divides $LM(f)$, the leading monomial of f. As $I_{w_{in},w_{out}}$ is a proper ideal, $LM(g) = LM(f) = X_i$, i depending on order, and as the monomial order is a graded order, g is a linear polynomial. The same holds for f'. □

Thus, the Algebraic Immunity with Leakage is always 1 and there are at least 2 independent linear polynomials when Hamming weight equations are added to the system of equations corresponding to B. It is quite natural that these two linear relations are in the ideal I_l since they correspond to information added by the leakage. But, it is important to see that these results are general. They do not depend on the bus size n and on the black box B. What is more interesting is the fact that, when the black box is fixed as a S-box, the number of linear independent relations is (in general) larger than 2 (see Section 5.4). The aim of the analysis done here is to study the impact of these linear relations on an algebraic attack. For instance, when the Hamming weight of the inputs is equal to 0, we have $X_1 = X_2 = \cdots = X_n = 0$, and the Y_i are given by $(y_1, \ldots, y_n) = B(x_1, \ldots, x_n) = B(0, \ldots, 0)$. In this case, the Gröbner basis computation gives :

$$\begin{aligned} I_{0,w_{out}} &= I_B + I_{hw}(0) + I_{hw}(w_{out}) = I_B + I_{hw}(0) \\ &= \langle X_1, \ldots, X_n, Y_1 + y_1, \ldots, Y_n + y_n \rangle \end{aligned}$$

which means that all the X_i and Y_i are fixed by the resolution. In this case, the number of independent linear equations is maximal. This (trivial) example corresponds to the case where the input (or output) is fixed by the leakage.

The trivial cases are not the only cases where one can show that there are more than two linear relations in I_l without fixing the black box B. Indeed, for all pair $l = (w_{in}, w_{out})$ of Hamming weight leakages such that the Hamming weight of input or output is equal to 1 or $n-1$:

$$l \in \{(1, w_{out}), (w_{in}, 1), (n-1, w_{out}), (w_{in}, n-1) | w_{in}, w_{out} \in \{0, \ldots, n\}\}$$

the number of points $N_B(l)$ which satisfy this couple (Definition 29) is upper bounded by n. Then, according to proposition 9, for all black box B of bus size n we have $\#AI_L(B, l) \geq n + 1$.

When the black box B is fixed, the number of independent linear equations depends on the Hamming weight ordered pair of the input and output. Thus, N_B in the case of the Hamming weight model is denoted by $N_B(w_{in}, w_{out})$ and it is explicitly given by

$$\begin{aligned} N_B(w_{in}, w_{out}) &= \#(HW^{-1}(w_{in}) \cap B^{-1}(HW^{-1}(w_{out}))) \\ &= \#\{x \in \mathbb{F}_2^n \text{ s.t. } HW(x) = w_{in} \text{ and } HW(B(x)) = w_{out}\} \end{aligned}$$

Remarque 6. *$N_B(w_{in}, w_{out})$ is related to the likelihood*

$$\mathbb{P}(HW(x) = w_{in} \text{ and } HW(B(x)) = w_{out})$$

Indeed, if we assume an equiprobability distribution of input bytes, this likelihood equals $2^{-n} N_B(w_{in}, w_{out})$.

In the particular case of an 8-bit bus size, Proposition 9 gives that $\#AI_L(B, w_{in}, w_{out})$ is always greater than or equal to $17 - N_B(w_{in}, w_{out})$ when $N_B(w_{in}, w_{out})$ is non-zero. Section 5.4 shows that, for a lot of usual S-Boxes, $N_B(w_{in}, w_{out})$ is often small in the Hamming weight leakage model, and the interesting Hamming weight pairs are the ones such that $N_B(w_{in}, w_{out})$ is small. Indeed, in this case, the constraints on the X_i and Y_i variables are strong, and we obtain several linear equations (see Proposition 9). Moreover, if the integer $N_B(w_{in}, w_{out})$ is very small (typically ≤ 6 for an 8-bit bus) then some bits can even be fixed, this will be discussed in Section 6.4.3.

Remarque 7. *Note that there are a lot of linear relations because there are two Hamming weight leakages around B. If only one Hamming weight leakage (input or output) is added to the equations of B then the number of solutions satisfying this condition is equal to the binomial $\binom{n}{w}$ and this is generally (if $w \neq 0, n$) too big to fix some bits and to obtain interesting linear equations. This situation is very similar to the Hamming distance model (see Section 7.2.1).*

In the next section we will study more precisely the influence of the number of independent linear relations on the complete solving.

6.4 Solving the complete system modeling the block cipher

As explained in Section 6.2, the problem is modeled by a system of equations which has a particular block diagonal structure. More precisely, it is

composed of blocks of equations which correspond to the cipher rounds with the leakage information. Each block is composed of systems of equations corresponding to the black boxes (S-boxes, MixColumns) and leakages of corresponding input and output data. We can split the complete system into small systems and locally study each of them. This local study is described in Section 6.3. We showed that the equations of the leakage model with the equations modeling one black box could yield a lot of low degree equations, such as linear equations, or could even give values of intermediate variables. Thanks to this particular structure, we can developed an efficient solving strategy.

6.4.1 Solving Strategy

The inputs of our off-line problem can be seen as a finite sequence $\mathcal{L}$ of values corresponding to the leakages of the successive 8-bit black boxes of the block cipher. In order to efficiently solve the complete polynomial system, we first seek the elements in $\mathcal{L}$ that provide the greatest possible number of linear relations. To do so, according to Section 6.3, we sort the sequence $\mathcal{L}$ by increasing N_B. Following this order, the polynomial systems corresponding to the first elements are those that provide the most linear equations. Thus, rather than computing directly a Gröbner basis of the complete system, we first compute Gröbner bases of some of these smaller systems. In a second step, we solve the complete system with the additional linear relations computed during the first step. This strategy based on splitting allows us to have better control on the maximal degree reached during the second step.

Remarque 8. *This strategy allows us to select some of the leakage ordered pairs in $\mathcal{L}$, in particular one could reject the leakages with large N_B and small confidence in their measurements during the on-line phase.*

6.4.2 A Sufficient Condition of Success

In the last step of the solving strategy, a polynomial system with several independent linear equations has to be solved. The efficiency of this step is strongly correlated with the number of these linear relations. More precisely, the efficiency heavily depends on the number of independent linear relations between the inputs and the outputs of the black boxes B.

For computational feasibility, it is assumed that the dispersion through a round of a block cipher is so important that the rounds are supposed to be independent. The black boxes of the same round are also supposedly independent. Thus, under this assumption, for a given block cipher based on

n	35	47	56	64	69	78	90
$\mathbb{P}(R_{PRESENT} \geq n)$	99%	90%	70%	50%	30 %	10%	1%

FIGURE 6.3 – Probability to obtain more than n linear equations with one PRESENT's round

a black box B, the expected number of total linear relations only depends on the local study around B done in Section 6.2. From this local study, one can compute the distribution of the number of linear equations coming from the study of the polynomial system of one round.

Example with the block cipher PRESENT and Hamming weight leakage model.

In this example, the black box B is the 8-bit S-box built from two PRESENT S-boxes. From the study of the probability distribution of the random variable $L_{PRESENT}$ (see Section 5.4) measuring the number of linear relations obtained from the local study of B, one can see that half of the possible leakage values provide at least 8 linear independent relations. Thus, the expected number of linear equations obtained from the substitution layer of one round is equal to $8 \times E(L_{PRESENT}) \simeq 64$ for PRESENT and represents a practical behavior. Hence, the expected number of linear equations is about the same size as the block length for one round and $n \times k$ for the complete system with n rounds with leakages.

From the probability distribution of $L_{PRESENT}$, we computed the distribution of the random variable $R_{PRESENT}$ measuring the number of linear relations obtained with all S-boxes and all Hamming weight leakages of one round of PRESENT. Figure 6.3 shows some of these results. According to them, the probability of obtaining at least 64 linear relations is about 50% with PRESENT. Thus, the expected number of linear equations has a high probability of being reached even with a small number of rounds with leakage information.

Considering these results, we propose a sufficient condition for an effective Algebraic Side Channel Attack. This condition is defined in a very simple way. Indeed, the local study of the non-linear part of a block cipher seen as a black box done in Section 6.3 gives an easy way to estimate the total number of low degree equations. This local study relies on the new notion of Algebraic Immunity with Leakage, which is itself linked to the function N_B as proved in Section 6. This function is very easy to compute from the definition of the black box and provides the following general condition.

Sufficient condition of success. *Let B be an n-bit black box. If $N_B(l)$*

(see Definition 29) is small enough (say less than n) for half of the possible leakages l then a block cipher based on B is vulnerable to an Algebraic Side Channel Attack with Gröbner Basis method.

In a Gröbner basis point of view, this condition implies (from Proposition 9) that at least half of the leakages provide more than n independent linear relations. Thus, the final polynomial system will be at least "half" linearized and (generally) easy to solve. In practice, this condition of success has been successfully applied for different black boxes constructed from S-boxes of well-known block ciphers. For instance, we showed that N_B is often small for PRESENT S-box with the Hamming weight leakage model, which explained the linearization of the full system of equation.

This sufficient condition of success can be expressed more precisely in practice by the mean of a complexity bound. Indeed, the proposed invariant N_B can be used to compute the complexity of the exhaustive search on inputs (and outputs) of the corresponding layer. Recall that $N_B(l)$ is equal to the number of possible input (and output) values for a black box B giving the leakage l. Thus, for a given n-bit black box B, the number of possible values decreases from 2^n to $N_B(l)$ thanks to the leakage l. For a fixed encryption and for a given layer with black boxes B_1 to B_m with the corresponding known leakage l_1 to l_m, the cost of exhaustive search on this layer is precisely equal to $\prod_{i=1}^{m} N_{B_i}(l_i)$. When the possible values of N_B are (almost) evenly distributed then the expected value of N_B is also a very good indicator and could be used to compute the average complexity of the exhaustive search of an input and an output of a round. Once more, we could assume that the rounds, as well as the boxes of a given round, are independent. Thus, the expected number of possible input values of the layer is $E(N_B)^m$ with m the number of boxes per rounds. In this case, the sufficient condition of success could be restated by saying that the block cipher is vulnerable to ASCA if this expected value implies that the complexity to recover the secret key is less than the assumed computing power of an attacker. This is usually the case in practice for PRESENT with the Hamming weight leakage model, where we have $E(N_B) \simeq 12.29$ and the average complexity of the exhaustive search of the layer input is $E(N_B)^8 \simeq 2^{29}$ (by comparison with a complexity of 2^{64} without leakage information). This condition is only sufficient because it does not consider implications from one leakage to the entire system and the ability of tools such that sat-solvers to exploit them.

Such a condition of success explains why it is possible to attack a block cipher when all the leakages of all the rounds are known and gives a first complexity bound for this attack. On the other hand, the authors of [RS09a]

showed that three or four consecutive leakages rounds are sufficient to quickly solve the complete polynomial system (see figures 6.1 and 6.2 of section 6.1 where we reproduce the experimental results presented in [RS09a, RSVC09]). Counting the number of linear equations given by these rounds is not sufficient to explain the efficiency of the method, in particular for the unknown Plaintext and Ciphertext scenario and the similarity between these results and those obtained in the known PC scenario. Moreover, the sufficient condition expressed in terms of exhaustive search can be improved by taking into account the implications between different rounds. Thus, in the next section, we study the particular situation where the leakages correspond to a few consecutive rounds.

6.4.3 Consecutive leakages

First we consider the basic case of two face to face black boxes B_1 and B_2 of consecutive rounds (Figure 6.4). The best case is when N_{B_i} ($i = 1$ and 2) is small enough ($<< n$ the bus size), such that the linear relations coming from the local study at B_1 and B_2 successfully fixed intermediate bits. For example, assume that two face to face bits y_i and x_j in the output of B_1 and the input of B_2 (see Figure 6.4) are known by the local study of B_1 and B_2 with Hamming weight leakages. The complete system of equations contains the equations modeling the permutation layer and the bitwise XOR (see Section 6.2), hence, during the second step of the strategy, the subkey bit k_j is easily deduced from the knowledge of the value of y_i and x_j. Once a subkey bit is fixed, other subkey bits in other rounds can be found with the key schedule equations.

Remarque 9. *This point of view explains why it is harder to successfully attack the problem with unknown plaintext and ciphertext when we do not know such consecutive leakages (see figures 6.1 and 6.2 of section 6.1, when the number of rounds of Hamming weight information is less than* 15 *for PRESENT, and less than* 5 *for AES).*

Following our sufficient condition and the study done in this section, the solving efficiency is due to a small N_B in average for the black box B. In the following, we exhibit a family of black boxes which are as far removed as possible from the condition of success and we verify experimentally that they are more resistant to algebraic side channel attacks. The condition of success exhibited in this section is not a necessary condition but the results of the next section tend to show that it is very close to be criterion of success.

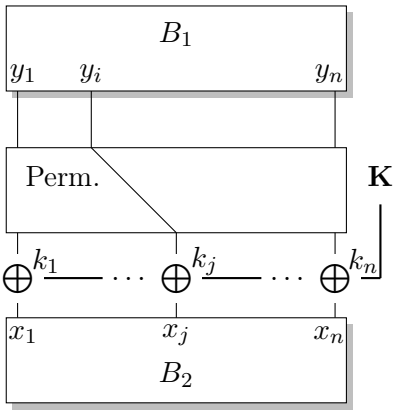


FIGURE 6.4 – Example of two consecutive rounds

6.5 Characterization of a family of resistant S-Boxes

Generally in the design of block ciphers using a substitution-permutation networks, S-boxes are the non-linear part. Thus, the S-boxes must be carefully chosen to make the cipher resistant against cryptanalysis. In our attacks, the S-boxes (seen as black boxes) leak information from the manipulated data. In this section, we only consider the situation covered by the Hamming weight model (i.e. a leakage l will be a couple of values corresponding to the Hamming weight of the input and output of the black box B). We should mention that a similarly study was already done for an other leakage model (the Hamming distance model) in [Pro05]. The knowledge of this additional information enables us to model this cipher component by a system of equations containing $\#AI_L$ linear relations. As seen in Section 6.3, $\#AI_L$ clearly depends on the Hamming weight pairs but also on the black box. The expected value of the number of such linear relations is very large for AES and PRESENT (greater than 7). These S-boxes are weak for our attacks and we are looking for a criterion for more resisting S-boxes. We study the influence of the black box on our criterion $N_B(w_{in}, w_{out})$, which is linked to $\#AI_L(B, w_{in}, w_{out})$. For that, we study the extreme case represented by the family of S-Boxes in complete contradiction with the sufficient condition of success exhibited in Section 6.4.2.

Since $N_B(w_{in}, w_{out})$ is explicitly given as the cardinality of the sets $\{x \in \mathbb{F}_2^n \text{ s.t. } HW(x) = w_{in} \text{ and } HW(B(x)) = w_{out}\}$, it is clear that a

bijective black box B maximizing $N_B(w_{in}, w_{out})$ for every possible pair (w_{in}, w_{out}), must be such that the sets $HW^{-1}(w_{in})$ and $B^{-1}(HW^{-1}(w_{out}))$ have precisely the same elements. Moreover, in this case, we have

$$N_B(w_{in}, w_{out}) = \binom{n}{w_{in}} = \binom{n}{w_{out}}$$

Indeed for all $w \in \mathbb{N}$ and for all bijective black box B, we have

$$\#HW^{-1}(w) = \binom{n}{w} \text{ and } \#B^{-1}(HW^{-1}(w)) = \binom{n}{w}$$

Actually, a bijective black box B maximizing $N_B(w_{in}, w_{out})$ for every possible pair (w_{in}, w_{out}), must satisfy

$$w_{in} = w_{out} \text{ or } w_{in} = n - w_{out}$$

Then such black box factors into

$$B(x) = \pi(x) \oplus f(HW(x))(1, \ldots, 1)$$

where π is a permutation stable under the Hamming weigth and f is a Boolean function such that $f(HW(x)) = f(n - HW(x))$.

Exemple 18. *The following table describes such an optimal 4-bit S-Box :*

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$B(x)$	0	B	D	C	E	6	9	8	7	5	3	1	A	2	4	F

where the factor f is given by

w_{in}	0	1	2	3	4
$f(w_{in})$	0	1	0	1	0

Therefore, there are optimal black boxes in the sense of maximizing N_B for every possible leakage, and experiments confirm that some of them are more resistant to our attacks (see Section 6.6). However, it seems that such black boxes are not resistant to differential and linear cryptanalysis. In the particular case of an even bus size, we are able to prove the following proposition :

Proposition 11. *Let n be an even number and let assume that the leakage model is the Hamming weight one. The n-bit black boxes in a complete contradiction with the sufficient condition of success (see Section 6.4.2) for an ASCA, i.e. satisfying $HW^{-1}(w_{in}) = B^{-1}(HW^{-1}(w_{out}))$, are not resistant to linear cryptanalysis. Actually, affine components exist for these S-boxes.*

Démonstration. Let B be an n-bit black box verifying the assumptions of the proposition. In this case, we already saw that

$$N_B(w_{in}, w_{out}) = \binom{n}{w_{in}} = \binom{n}{w_{out}}$$

Thus, $w_{in} = w_{out}$ or $w_{in} = n - w_{out}$ which implies that $w_{out} \equiv w_{in} \mod 2$ in the case where n is an even integer. Hence we have $(1, \ldots, 1) \cdot B(x) = (1, \ldots, 1) \cdot x$ and the nonlinearity of B is equal to 0. □

In the case where n is an odd number, we could find some ASCA-resistant black boxes with a linearity slightly less than 2^n. However, all these found black boxes stay weak against linear cryptanalysis. Hence, the question arises of determining whether ASCA optimally resistant S-boxes can reach a good nonlinearity. But we were not able either to prove the inexistence of ASCA optimally resistant black boxes with a strong resistance against linear cryptanalysis, in spite of the following rewriting using the Krawtchouk polynomials $K_{n,w}$, which establishes a relation between the function N_B and the Walsh coefficient (essential for computing the linearity of a box). Krawtchouk polynomials are classical orthogonal univariate polynomials over the rationals associated with the binomial distribution. They are defined by

$$K_{n,w}(X) = \sum_{j=0}^{n} (-1)^j \binom{X}{j} \binom{n-X}{w-j}$$

We have $K_{n,w}(0) = \binom{n}{w}$, and for all $a \in \mathbb{F}_2^n$, we have

$$\sum_{x \in \mathbb{F}_2^n | HW(x)=w} (-1)^{a \cdot x} = \sum_{j=0}^{n} (-1)^j \binom{HW(a)}{j} \binom{n - HW(a)}{w-j} = K_{n,w}(HW(a)) \tag{6.4}$$

By the inverse Fourier transform, we deduce from equation (6.4) that for all $w = 0, \ldots, n$,

$$\sum_{a \in \mathbb{F}_2^n} K_{n,w}(HW(a))(-1)^{a \cdot x} = \begin{cases} 2^n & \text{if } HW(x) = w \\ 0 & \text{otherwise} \end{cases}$$

Finally, for all $w_{in}, w_{out} = 0, \ldots, n$, we have

$$\begin{aligned} & N_B(w_{in}, w_{out}) \\ &= \#\{x \in \mathbb{F}_2^n | HW(x) = w_{in},\ HW(B(x)) = w_{out}\} \\ &= 2^{-2n} \sum_{a,b,x \in \mathbb{F}_2^n} K_{n,w_{in}}(HW(a)) K_{n,w_{out}}(HW(b))(-1)^{a \cdot x + b \cdot B(x)} \\ &= 2^{-2n} \sum_{a,b \in \mathbb{F}_2^n} K_{n,w_{in}}(HW(a)) K_{n,w_{out}}(HW(b)) B_b^{\mathcal{W}}(a) \end{aligned}$$

where $B_b^{\mathcal{W}}(a) := \sum_{x \in \mathbb{F}_2^n} (-1)^{a \cdot x + b \cdot B(x)}$ be the Walsh transform of B.

Actually, as already proposed in [RS09a], it seems that one of the safest way to design a substitution box resistant to all kinds of attacks is to increase the bus size. Indeed, $N_B(w_{in}, w_{out})$ also depends on the bus size n, and for all bijective black boxes, since $N_B(w_{in}, w_{out})$ defines a partition of $\mathbb{F}_2^n$ we have

$$\sum_{(w_{in}, w_{out}) \in \mathbb{N}^2} N_B(w_{in}, w_{out}) = 2^n$$

This shows that, in general, $N_B(w_{in}, w_{out})$ grows exponentially as n increases.

6.6 Experiments

In this part we show that the condition of success from Section 6.4.2 is supported by the experiments (positive and negative) we performed for PRESENT, AES and for the resistant S-box given in Section 6.5. Following our model described in Section 6.2, we build a complete system of polynomial equations as a function of the target cipher (PRESENT or AES) and as a function on the sequence $\mathcal{L}$ of leakage information which are taken into account. Since Magma [BCP97] provides an efficient implementation of the Faugère F4 algorithm, we use this computer algebra system for our experiments. For the SAT attacks, we use CryptoMiniSat [SNC09], glucose2 [AS11a, AS09], glueminisat [AS11b], and plingeling [Bie11] which are the winners of the SAT Race 2010 and SAT Race 2011. This section ends with a study of several S-Boxes used by other cryptosystems with the presence of 8-bit Hamming weight leakages.

6.6.1 Experiments against PRESENT using the Hamming weight model

Here, the leakage comes from the inputs and outputs of S-boxes (we see two consecutive 4-bit PRESENT S-boxes as an 8-bit one, see chapter 1 for a description of PRESENT).

Following the situation of [RS09a], we assume that the device leaks the Hamming weights of the values commuting on its 8-bit bus. In this case, two S-boxes are processed at the same time, which is represented by a 8-bit black box B. From the definition of B one can easily deduce the table 6.5 giving $N_B(w_{in}, w_{out})$ in function of the input/output couples. We compare this table with the one (see Table 6.6) given $\#AI_L(B, w_{in}, w_{out})$ in function of the same

w_{in} \ w_{out}	0	1	2	3	4	5	6	7	8
0	0	0	0	0	1	0	0	0	0
1	0	0	0	0	8	0	0	0	0
2	0	0	2	2	18	4	2	0	0
3	0	0	8	12	8	20	8	0	0
4	1	2	3	24	7	22	6	4	1
5	0	4	4	16	12	8	8	4	0
6	0	2	6	2	12	2	4	0	0
7	0	0	4	0	4	0	0	0	0
8	0	0	1	0	0	0	0	0	0

FIGURE 6.5 – $N_B(w_{in}, w_{out})$ where B is two PRESENT S-Boxes

couples. One can see that $N_B(w_{in}, w_{out})$ can give a good indicator when we are interested in maximizing the number of independent linear polynomials $\#AI_L(B, w_{in}, w_{out})$ during an algebraic attack. Even if $\#AI_L(B, w_{in}, w_{out})$ is not rigorously inversely proportional to $N_B(w_{in}, w_{out})$ it is close to be the case. A counterexample for this reciprocity is given by $N_B(3, 3) = 12$ which is less than $N_B(2, 4) = 18$ but $\#AI_L(B, 3, 3) = 5 < \#AI_L(B, 2, 4) = 8$.

Assume that the probability of appearing of an input byte of the 8-bit black box B is 1/256. With Figures 6.5 and 6.6, we can easily compute the probability distribution of the random variable L_B measuring the number of linear relations that we obtain by adding the leakage information to our system. The Figure 6.8 presents a chart providing the probabilities $\mathbb{P}(L_B = k)$ inside the sectors labeled by k. The integers k that give null probability are not shown. We could note that the probability that at least 8 independent linear relations are produced is about 50%. Moreover, the expected value of L_B is $\simeq 7.9$.

We also compare Table 6.5 with the one given the number of fixed bits (see Table 6.7) in function of the input/output couples. One can see that when N_B is small the number of fixed bits can be large.

We will now use all this knowledge to explain the experiments behaviors observed during an algebraic attack of PRESENT under different scenarios.

As in [RS09a], the knowledge of all Hamming weight pairs from the 31 rounds of PRESENT always leads to successful SAT solver and Gröbner attacks, in both cases known and unknown plaintext and ciphertext. Note that in this experiment, we already must apply our solving strategy (6.4.1).

w_{in} \ w_{out}	0	1	2	3	4	5	6	7	8
0					16				
1					9				
2			15	15	8	13	15		
3			9	5	9	5	9		
4	16	15	14	2	11	3	12	13	16
5		13	13	2	7	10	11	13	
6		15	12	15	7	15	14		
7			13		13				
8			16						

FIGURE 6.6 – $\#AI_L(B, w_{in}, w_{out})$ where B is two PRESENT S-Boxes

w_{in} \ w_{out}	0	1	2	3	4	5	6	7	8
0	0	0	0	0	16	0	0	0	0
1	0	0	0	0	0	0	0	0	0
2	0	0	8	10	0	4	8	0	0
3	0	0	0	0	0	0	0	0	0
4	16	10	4	0	0	0	4	6	16
5	0	4	4	0	0	0	2	4	0
6	0	10	2	10	0	6	8	0	0
7	0	0	6	0	4	0	0	0	0
8	0	0	16	0	0	0	0	0	0

FIGURE 6.7 – Number of fixed bits with two PRESENT S-Boxes

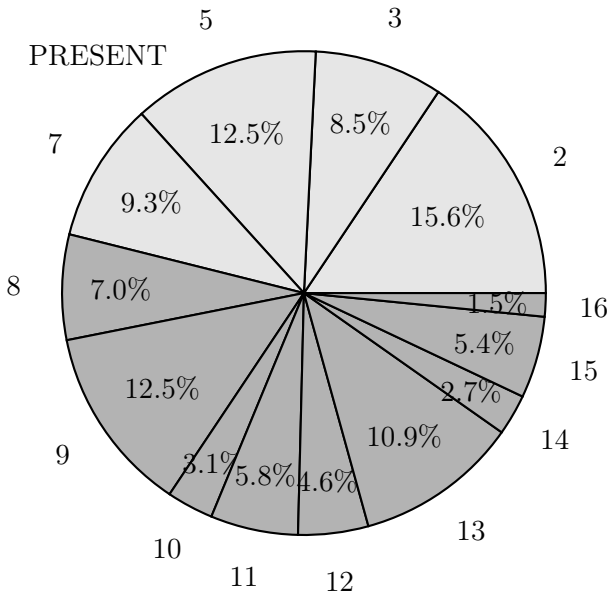


FIGURE 6.8 – $\mathbb{P}(L_B = k)$ for k with non zero probability and for HW leakages

More precisely, we first have to successively compute the Gröbner basis of each round before computing the last Gröbner basis of the whole system. Indeed, if we try to directly compute the Gröbner basis of the system of equations modeling the initial problem, then the maximum degree reached during the computation is too big, and the computation is very slow.

Similar experiments allow us to check Remark 7 : we explained that an important condition is the knowledge of the couple of Hamming weights around the S-box. So, we performed attacks on PRESENT with only the Hamming weights of input (or output) data of all S-Boxes. Our Gröbner basis solving strategy and SAT solvers are much less efficient, rather than failed, in this situation (which is very similar to the HD scenario, see Chapter 7). Indeed, without the knowledge of the plaintext and ciphertext, SAT solvers and Gröbner attacks always failed. It confirms that a large N_B, which could not fix enough intermediate bits (see Section 6.4.3), is not able to find subkey bits. Otherwise, if the plaintext and ciphertext were known, the low number of linear equations often allow some SAT solvers to recover the key. However, the solving step is very long with all used SAT solvers (once again comparable to the HD model) and the Gröbner attack fails with an out of memory if we take much than 2 or 3 rounds. These experiments confirmed the necessity of

always using weight couples.

Other experimental results confirmed that the number $N_B(w_{in}, w_{out})$ is a very good indicator to sort interesting Hamming weight couples for an ASCA. These experiments have checked Remark 8 for PRESENT. More precisely, one can reject the leakages (w_{in}, w_{out}) with large $N_B(w_{in}, w_{out})$ without consequences on the success rate. Conversely, if we reject the leakages (w_{in}, w_{out}) with small $N_B(w_{in}, w_{out})$ then all Gröbner basis attacks and all SAT attacks failed.

Following the condition of success, we also designed theoretically more resistant S-boxes in Section 6.5. Experiments with PRESENT, where we use such S-boxes as a substitute for original S-boxes, are also intractable (with both SAT and Gröbner) which confirm that these S-boxes are much more resistant to this kind of attack.

Thus, we are able to explain the experiments against PRESENT in [RS09a]. In particular, the study in Section 6.4.3 explained the successful attacks when we know consecutive leakages. Note that the success rates given in [RS09a, RSVC09] in the case of unknown plaintext and ciphertext attacks with randomly distributed leakage information (figures 6.1 and 6.2) can be explained by the pigeonhole principle. For the example involving 31 rounds of PRESENT, the success rate is greater than zero when the number of rounds with leakages reaches approximately 15. By the pigeonhole principle, this corresponds to the case where there are at least two consecutive rounds with leakage and thus Section 6.4.3 explained these experimental results.

6.6.2 Experiments against AES using the Hamming weight model

Here, the 8-bit leakage comes from the inputs and outputs of S-boxes and MixColumns.

The same study of the AES S-box B also shows (see Table 6.9 and Table 6.10) that $N_B(w_{in}, w_{out})$ is a very good indicator when we are interested in maximizing $\#AI_L(B, w_{in}, w_{out})$.

As before, we compute the probability distribution of the random variable L_B measuring the number of linear relations that we obtain by adding the leakage information to our system. The chart of the Figure 6.12 provides the probabilities $\mathbb{P}(L_B = k)$ inside the sectors labeled by k. The integers k that give null probability are not shown.

Surprisingly, the probability that at least 8 independent linear relations are produced is also about 50%, and the expected value of L_B is $\simeq 7.3$ which

w_{in} \ w_{out}	0	1	2	3	4	5	6	7	8
0	0	0	0	0	1	0	0	0	0
1	0	0	2	0	1	3	2	0	0
2	0	2	3	8	5	4	4	2	0
3	1	1	4	17	16	10	5	2	0
4	0	3	9	11	21	16	9	1	0
5	0	1	7	10	19	14	3	2	0
6	0	0	3	7	5	8	4	0	1
7	0	1	0	2	2	1	1	1	0
8	0	0	0	1	0	0	0	0	0

FIGURE 6.9 – $N_B(w_{in}, w_{out})$ where B is the AES S-Box

w_{in} \ w_{out}	0	1	2	3	4	5	6	7	8
0					16				
1			15		16	14	15		
2		15	14	9	12	13	13	15	
3	16	16	13	2	2	7	12	15	
4		14	8	6	2	2	8	16	
5		16	10	7	2	3	14	15	
6			14	10	12	9	13		16
7		16		15	15	16	16	16	
8				16					

FIGURE 6.10 – $\#AI_L(B, w_{in}, w_{out})$ where B is the AES S-Box

is almost equal as for PRESENT.

In the same way as PRESENT, we will now use all this knowledge to explain the experiments behaviors observed during an algebraic attack of AES under different scenarios.

As in [RSVC09] and [RS09a], the knowledge of all Hamming weight pairs (i.e. around the S-boxes and around the MixColumns) from the 10 rounds of AES always leads to successful SAT and Gröbner attacks, in the case of known plaintext and ciphertext. The ASCA in unknown plaintext and ciphertext scenario is effective only with SAT solvers, the Gröbner compu-

w_{in} \ w_{out}	0	1	2	3	4	5	6	7	8
0	0	0	0	0	16	0	0	0	0
1	0	0	10	0	16	6	12	0	0
2	0	10	7	2	3	2	6	10	0
3	16	16	2	0	0	0	2	10	0
4	0	10	0	0	0	0	2	16	0
5	0	16	1	0	0	0	5	8	0
6	0	0	6	1	3	1	6	0	16
7	0	16	0	10	8	16	16	16	0
8	0	0	0	16	0	0	0	0	0

FIGURE 6.11 – Number of fixed bits with the AES S-Box

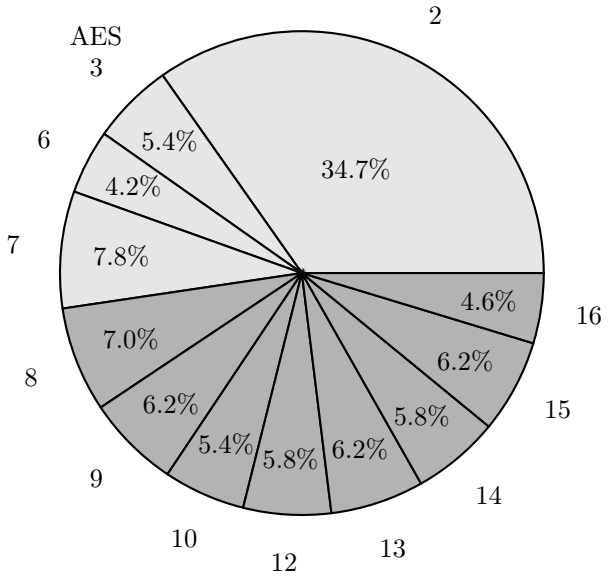


FIGURE 6.12 – $\mathbb{P}(L_B = k)$ for k with non zero probability and for HW leakages

tation requiring too much memory space.

Note that we only took the Hamming weights of the input and the output of the MixColumns step. Thus, the MixColumns transformation is seen as

a black box and then, contrary to the experiments described in the article [RSVC09], previous ASCA are effective against AES even if the MixColumns operation is implemented as a table look-up. This difference of experimental results may be explained by the advances in SAT solvers heuristics.

The MixColumns transformation being a linear step, we also wonder whether its leakage was required. The experiments with only the Hamming weights leakage of the S-boxes (i.e. without leakage from MixColumns) are effective with SAT solvers, even if the solving step is much longer (in average a couple of hours against several dozen seconds).

We also make experiments with the theoretically resistant S-boxes designed in Section 6, which confirm that AES with these S-boxes are more resistant to both SAT and Gröbner attacks.

6.6.3 A study of several S-Boxes and 8-bit Hamming weight leakage model

In this part, we provide tables showing the distribution of N_B for different S-Boxes with the Hamming weight leakage model. All the results show that these S-boxes are weak against an ASCA.

CAMELLIA S-Boxes

CAMELLIA [AIK+] is a block cipher developed by NTT and Mitsubishi in 2000. It is a Feistel cipher and uses four 8-bit S-Boxes. S-Box1 is given by a table and S-Box2, S-Box3 and S-Box4 are defined using S-Box1 as follows :

$$\begin{aligned} \text{S-Box2}[X] &= \text{S-Box1}[X] \lll 1 \\ \text{S-Box3}[X] &= \text{S-Box1}[X] \lll 7 \\ \text{S-Box4}[X] &= \text{S-Box1}[X \lll 1] \end{aligned}$$

where the symbol $\lll$ correspond to left rotation operation. Because of these definitions, it is easy to see that the four S-boxes have the same Hamming weights distribution. Thus, the Figures 6.13, 6.15 and 6.16 equally correspond to the four S-boxes.

w_{in} \ w_{out}	0	1	2	3	4	5	6	7	8
0	0	0	0	1	0	0	0	0	0
1	0	0	1	3	3	1	0	0	0
2	0	1	3	6	6	7	2	3	0
3	1	0	3	15	16	16	4	1	0
4	0	4	12	11	19	14	8	1	1
5	0	2	5	11	17	10	8	3	0
6	0	0	3	7	8	5	5	0	0
7	0	1	1	2	1	2	1	0	0
8	0	0	0	0	0	1	0	0	0

FIGURE 6.13 – $N_B(w_{in}, w_{out})$ where B is one of the CAMELLIA S-Boxes

w_{in} \ w_{out}	0	1	2	3	4	5	6	7	8
0				16					
1			16	14	14	16			
2		16	14	11	11	10	15	14	
3	16		14	3	2	2	13	16	
4		13	5	6	2	3	9	16	16
5		15	12	6	2	7	9	14	
6			14	10	9	12	12		
7		16	16	15	16	15	16		
8						16			

FIGURE 6.14 – $\#AI_L(B, w_{in}, w_{out})$ where B is one of the CAMELLIA S-Boxes

w_{in} \ w_{out}	0	1	2	3	4	5	6	7	8
0				16					
1			16	14	14	16			
2		16	14	11	11	10	15	14	
3	16		14	3	2	2	13	16	
4		13	5	6	2	3	9	16	16
5		15	12	6	2	7	9	14	
6			14	10	9	12	12		
7		16	16	15	16	15	16		
8						16			

FIGURE 6.15 – $\#AI_L(B, w_{in}, w_{out})$ where B is one of the CAMELLIA S-Boxes

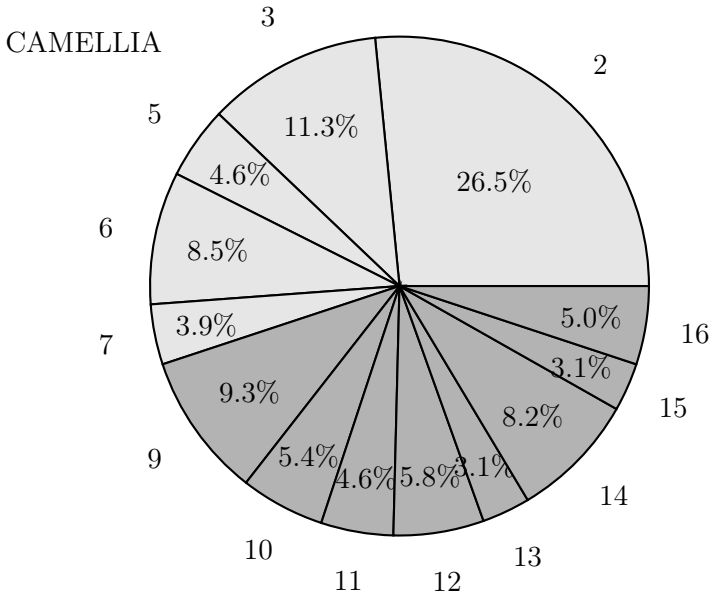


FIGURE 6.16 – $\mathbb{P}(L_B = k)$ for k with non zero probability and for HW leakages

SMS4 S-Box

SMS4 [Off06] is a block cipher used in the Chinese WLAN National Standard. It uses an 8-bit S-Box given by a table. The following Figures show its Algebraic Immunity with Hamming weights Leakage.

w_{in} \ w_{out}	0	1	2	3	4	5	6	7	8
0	0	0	0	0	0	1	0	0	0
1	0	0	1	1	4	2	0	0	0
2	0	1	4	7	9	4	1	2	0
3	0	2	7	12	10	14	8	3	0
4	1	2	6	23	16	14	7	1	0
5	0	2	6	8	21	11	5	2	1
6	0	0	2	3	9	7	7	0	0
7	0	1	1	2	1	3	0	0	0
8	0	0	1	0	0	0	0	0	0

FIGURE 6.17 – $N_B(w_{in}, w_{out})$ where B is the SMS4 S-box

w_{in} \ w_{out}	0	1	2	3	4	5	6	7	8
0						16			
1			16	16	13	15			
2		16	13	10	8	13	16	15	
3		15	10	5	7	3	9	14	
4	16	15	11	2	3	3	10	16	
5		15	11	9	3	6	12	15	16
6			15	14	8	10	10		
7		16	16	15	16	14			
8			16						

FIGURE 6.18 – $\#AI_L(B, w_{in}, w_{out})$ where B is the SMS4 S-box

w_{in} \ w_{out}	0	1	2	3	4	5	6	7	8
0						16			
1			16	16	13	15			
2		16	13	10	8	13	16	15	
3		15	10	5	7	3	9	14	
4	16	15	11	2	3	3	10	16	
5		15	11	9	3	6	12	15	16
6			15	14	8	10	10		
7		16	16	15	16	14			
8			16						

FIGURE 6.19 – $\#AI_L(B, w_{in}, w_{out})$ where B is the SMS4 S-box

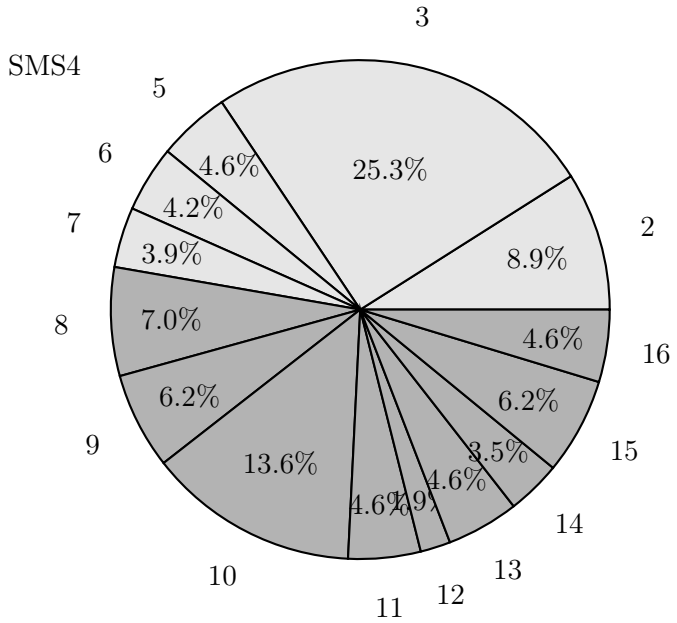


FIGURE 6.20 – $\mathbb{P}(L_B = k)$ for k with non zero probability and for HW leakages. The expected value of L_B is $\simeq 7.7$.

6.7 Conclusion

In this chapter we introduced a new criterion for the effectiveness of ASCA. This criterion rely on a new notion of algebraic immunity. In order to simplify the analysis of a given block cipher we introduce an invariant related to this block cipher. This invariant is a function which can be easily defined and computed from the definition of the given block cipher (no need of advanced algebra). From this new invariant we exhibit a sufficient condition of success of an ASCA and we were able to theoretically explain the experiments done by Renauld, Standaert and Veyrat-Charvillon in [RS09a, RSVC09] by studying the distribution of the values taken by this function. Experimental results confirmed our sufficient condition success and showed that it is a good condition for both effective Gröbner and SAT solver attacks. This understanding allowed us to design S-boxes optimally resistant to ASCA following this condition. However, we observe that these S-boxes are weak for linear and differential cryptanalysis, which confirms, as observed in [RS09a], that a large bus size can be prescribed to design a resistant block ciphers. In this paper, we chose to follow [RS09a, RSVC09] by only study the Hamming weight model, but other good leakage models can be selected. This will be the subject of the next chapter. More generally, extending the attack to deal with erroneous equations is one of the long-range research aims. The well understanding of the algebraic phase of this attack done in this chapter is already a first step in this direction.

Chapitre 7

Les attaques ASCA étendues à d'autres modèles de fuites

7.1 Introduction

L'analyse des attaques algébriques par canaux auxiliaires (ASCA) effectuée au chapitre précédent a permis d'avoir une bonne compréhension de l'étape de résolution algébrique, dans le cas de chiffrements symétriques laissant fuir le poids de Hamming de certaines valeurs intermédiaires. Jusqu'à présent, on s'est uniquement borné à ce modèle de fuite puisqu'il correspondait au modèle utilisé dans les précédentes publications sur les attaques ASCA (en particulier [RS09a, RSVC09]). Or, suivant les appareils visés, d'autres modèles génériques peuvent être plus aptes à décrire la consommation électrique. Par exemple, le modèle de la distance de Hamming est un modèle très souvent utilisé dans les attaques par canaux auxiliaires pour décrire la consommation des bus et des registres (cf. Chapitre 4). Il parait donc nécessaire d'étudier aussi plus précisément le modèle de la distance de Hamming pour les attaques ASCA.

D'une manière générale, c'est la question de l'influence du modèle de fuite sur la résolution algébrique qui est posée dans ce chapitre. En plus de l'étude du modèle de la distance de Hamming dans deux scénarios différents, nous verrons que cette question très générale apporte aussi une première ébauche de solution pour la gestion des informations auxiliaires bruitées. Toujours dans cette perspective, nous tenions aussi à mentionner que les attaques par faute ou encore les attaques par collision interne pouvaient aussi être considérées comme d'autres modèles de fuite exploitables en ASCA. Ce chapitre est donc essentiellement composé de l'exposé de quelques perspectives de recherche.

7.2 La distance de Hamming

Le modèle de la distance de Hamming est souvent plus adaptée que le modèle du poids de Hamming pour décrire la consommation électrique (voir [BCO04] ou [MOP07] pour une discussion sur ce sujet). En effet, la consommation d'un microcontroller, par exemple avec une technologie CMOS, dépend généralement du nombre de bits modifiés dans les registres ou sur les bus (en particulier à cause de la présence de nombreux composants tels que des condensateurs, des cellules logiques/bascules, etc). La variation de consommation d'une transition d'une valeur x à une valeur y est alors modélisée par $HD(x, y) = HW(x \oplus y)$.

Les fuites mesurées en pratique sur des dispositifs réels dépendent aussi fortement de l'implémentation. Ainsi, un grand nombre de scénarios différents peuvent être considérés avec le modèle de la distance de Hamming (par exemple, les points du circuit d'où sont mesurées les distances peuvent varier, ou encore le nombre de distances totales que l'on peut obtenir, etc). Le but de cette analyse étant principalement d'étudier l'influence des informations supplémentaires sur l'étape de résolution algébrique, nous ne chercherons pas forcement les scénarios les plus réalistes. Nous cherchons plutôt à vérifier la pertinence et à étendre l'analyse des attaques ASCA proposée au chapitre 6. Comme pour le modèle du poids de Hamming (Chapitre 6), nous supposerons donc qu'une première campagne d'acquisitions nous a fourni un ensemble de fuites dans le modèle de la distances de Hamming sur 8-bit.

Nous avons seulement considéré deux scénarios différents. Dans un premier temps, on suppose que les mesures fournies sont les distances de Hamming entre l'entrée et la sortie des boites de substitutions. Le scénario suivant considère plutôt les fuites qui apparaissent entre deux chiffrements successifs.

7.2.1 La distance de Hamming entre l'entrée et la sortie des boîtes-S

On reprend donc les outils développés pendant l'analyse des ASCA du chapitre 6 et plus particulièrement les résultats de la section 6.2 en les adaptant au modèle de la distance de Hamming.

On commence par remarquer que, comme dans le cas du poids de Hamming, l'immunité algébrique avec fuite est encore égal à 1 dans le modèle de la distance de Hamming pour toute boîte noire B. Ce résultat est en fait une conséquence du lemme 6 (section 6.3 du chapitre 6), puisque le système d'équations représentant une distance de Hamming est construit à partir

du système (6.2) dans lequel on a substitué les X_i par $X_i \oplus Y_i$. En effet, considérons l'idéal $I_d \subset \mathbb{F}_2[X_1, \ldots, X_n, Y_1, \ldots, Y_n]$ engendré par les équations représentant la boîte B, les équations de corps (6.1) et par L_d (6.2) les équations représentant le fait que le poids de Hamming du XOR entre l'entrée et la sortie est égal à d, c'est-à-dire :

$$I_d = \langle L_d(X_1 + Y_1, \ldots, X_n + Y_n) \cup S_B \cup S_{\text{Field Eq.}} \rangle$$

alors on a la proposition suivante :

Proposition 12. *Soit G la base de Gröbner de l'idéal I_d pour un ordre monomial du degré. G contient alors au moins un polynôme linéaire.*

Démonstration. D'après le Lemme 6 (section 6.3), on sait que l'idéal I_d contient toujours le polynôme

$$f = (X_0 + Y_0) + \cdots + (X_n + Y_n) + (d \bmod 2) \in I_d$$

Le reste de la preuve est donc similaire à la preuve de la Proposition 10 (section 6.3). □

Ainsi il y a toujours au moins un polynôme linéaire lorsque les équations de la distance de Hamming sont ajoutées au système d'équations décrivant la boîte B. De la même façon que pour le modèle du poids de Hamming, il peut y avoir beaucoup plus de relations linaires en fonction de la valeur de la distance de Hamming et de la boîte-S. Par la proposition 9 (Section 6.2.2), le nombre de relations linéaires linéairement indépendantes est encore relié au nombre de points qui satisfont cette distance de Hamming et on définit donc N_B dans le cas de la distance de Hamming par

$$N_B(d) = \#\{x \in \mathbb{F}_2^n \text{ tels que } HD(x, B(x)) = d\}$$

Malheureusement, contrairement au modèle du poids de Hamming, les expériences sur les boîtes-S de l'AES et PRESENT avec le modèle de la distance de Hamming sur 8-bit ont montré qu'il y a rarement plus d'un polynôme linéaire dans la base de Gröbner de l'idéal I_d. Les résultats sont présentés par les tableaux 7.1 et 7.2 qui détaillent les distributions des valeurs prisent par $\#AI_L(d)$ pour AES et PRESENT ainsi que les différentes valeurs de $N_{\text{PRESENT}}(d)$ et $N_{\text{AES}}(d)$ et le nombre de bits fixés en fonction de d.

Le petit nombre de relations linéaires pour AES et PRESENT s'explique par le fait que les valeurs prisent par $N_{\text{PRESENT}}(d)$ et $N_{\text{AES}}(d)$ sont bien

d	0	1	2	3	4	5	6	7	8
$N_B(d)$	0	0	16	56	81	64	30	8	1
$\#AI_L(B,d)$	0	0	10	3	1	1	1	9	16
Nb bits fixés	0	0	0	0	0	0	0	0	16

FIGURE 7.1 – modèle HD avec B représentant 2 boîtes-S de PRESENT (vues comme une boîte-S de 8 bits)

d	0	1	2	3	4	5	6	7	8
$N_B(d)$	0	12	31	48	67	59	32	7	0
$\#AI_L(B,d)$	0	5	1	1	1	1	1	10	0
Nb bits fixés	0	0	0	0	0	0	0	1	0

FIGURE 7.2 – modèle HD avec B la boîte-S de AES

plus grandes que dans le cas du modèle du poids de Hamming. Ainsi, on s'attend à ce que l'espérance du nombre de relations linéaires soit très petit. Intuitivement, l'espérance du nombre de relations linéaires sera plus grande pour la boîte-S de PRESENT que pour celle de l'AES. Plus précisément, l'espérance du nombre de relations linéaires est d'environ $2,3$ pour la boîte-S de PRESENT et $1,4$ pour celle de l'AES (en supposant une distribution équirépartie des octets en entrée), ce qui est beaucoup moins que dans le cas du modèle du poids de Hamming (où ces valeurs étaient alors comprises entre 7 et 8, cf Section 6.3). Il devient alors manifeste qu'une attaque par calcul de bases de Gröbner sera beaucoup plus difficile dans ce cas.

Des expériences sur PRESENT ont confirmé que l'attaque par base de Gröbner nécessite beaucoup trop de mémoire pour être praticable dans ce contexte. En effet, nous n'avons pas été en mesure d'effectuer une ASCA avec le modèle de la distance de Hamming par base de Gröbner sur 31 tours de PRESENT (une attaque sur seulement 3 tours de PRESENT a été réussie, mais une erreur pour mémoire insuffisante se produit lorsqu'il y a plus de tours).

Néanmoins, le critère suffisant de succès (Section 6.4.2) appliqué à $N_{\text{PRESENT}}(d)$ laisse penser que les ASCA seraient possibles dans certains contextes favorables. En supposant l'équirépartition des octets en entrée des boîtes-S, on calcule que l'espérance du nombre de valeurs possibles en entrée d'une boîte-S de PRESENT est donnée par $E(N_{\text{PRESENT}}) \simeq 2^{5,9}$, soit un

coût d'environ $E(N_{\text{PRESENT}})^8 \simeq 2^{47,2}$ en moyenne pour la recherche exhaustive de la valeur d'entrée d'une couche de substitution entière de PRESENT en présence des informations de fuite avec le modèle de la distance de Hamming sur 8-bits. Bien que ce coût reste assez élevé, on sait que ce n'est qu'une borne supérieure souvent beaucoup trop large. En effet, la simplicité de la dérivation de clef de PRESENT permet de propager facilement à un grand nombre de tours les informations obtenues à partir de l'étude locale des fuites des boîtes-S. On a d'ailleurs déjà remarqué que les SAT solveurs étaient capable d'exploiter toutes les implications d'une fuite dans un système d'équations (cf. Section 6.4). Il est donc aussi nécessaire d'expérimenter la résolution par SAT solveurs dans ce modèle de fuite.

Lorsque les textes clairs et chiffrés sont connus, on obtient alors que les contraintes sur les variables intermédiaires (contraintes apportées par les équations des distances de Hamming) semblent souvent suffisantes pour une résolution par SAT solveurs. Par exemple, le taux de réussite avec le SAT solveur CryptoMinisat ([SNC09]) est d'environ 70% au bout de 3 heures sur PRESENT avec les distances de Hamming sur 8-bits entre les entrées et les sorties de toutes les boîtes-S de tous les tours. Le temps de résolution par SAT solveurs semble cependant dépendre fortement des heuristiques utilisées et il est donc difficile de le corréler avec la distribution des valeurs de fuite.

D'un autre côté, comme nous l'avions prédit avec l'étude de la fonction N_{AES}, toutes les expériences d'ASCA contre l'AES avec le modèle de fuite de la distance de Hamming ont échoué (autant par base de Gröbner que par SAT solveurs). D'ailleurs, en appliquant le critère suffisant de réussite, on obtient $E(N_{\text{AES}}) \simeq 2^{5.6}$, donc une espérance du nombre de valeurs possibles pour un tour complet égale à $E(N_{\text{AES}})^{16} \simeq 2^{90}$. Cela nous donne une borne supérieure qui ne permet pas de garantir la réussite de l'ASCA. Cette différence des résultats d'expérience entre AES et PRESENT s'explique par la taille des clefs et des blocs, mais aussi par la structure algébrique bien plus complexe de l'AES (en particulier le keyschedule) qui semble contrer les ASCA avec seulement les distances de Hamming entre les entrées et les sorties des boîtes-S.

Enfin, les expériences en clairs/chiffrés inconnus sont toutes négatives, résultats prévisibles par la très faible proportion de bits fixés par les informations de fuite en entrée et en sortie des boîtes-S (cf tableaux 7.1 et 7.2). Cependant, supposer uniquement la connaissance des distances entre les entrées et les sorties des boites de substitutions est sûrement une restriction

beaucoup trop forte. En effet, si l'on est en mesure d'obtenir ces distances-là, alors on devrait également pouvoir obtenir les autres distances correspondant aux autres opérations effectuées entre deux couches de substitutions. Par exemple, on pourrait aussi supposer la connaissance de distances de Hamming autour du MixColumns de l'AES, ou bien les distances de Hamming entre les sorties d'une étape de substitution et l'entrée de l'étape de substitution du tour suivant. L'ajout de ces informations de fuite complémentaires augmenterait sûrement l'efficacité et le taux de succès de l'attaque. Mais l'étude du modèle de la distance de Hamming entre l'entrée et la sortie des boîtes-S reste intéressant puisque le but de ces expériences est avant tout de confirmer la pertinence des analyses et des critères proposés, tout en pouvant les étendre à d'autres modèles de fuite.

7.2.2 La distance de Hamming entre chiffrements successifs

Une autre variante du modèle de la distance Hamming que l'on s'est proposé d'étudier consiste cette fois-ci en la connaissance de la distance de Hamming entre des chiffrements successifs avec une clef secrète fixée. Selon notre habitude, on considère connue la distance de Hamming sur 8-bits. Plus précisément, il s'agit de la distance entre les entrées des boîtes de substitutions de deux chiffrements consécutifs. Il en va de même pour les sorties des boîtes de substitutions.

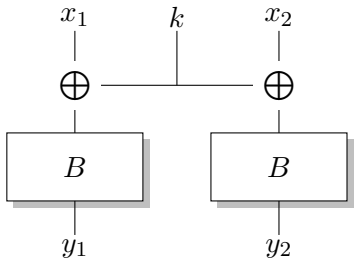


FIGURE 7.3 – Boîte B lors de deux chiffrements successifs avec la même clef

Comme présenté par la Figure 7.3, ce modèle fournit alors les distances de Hamming $HD(x_1 \oplus k, x_2 \oplus k) = HD(x_1, x_2) = HW(x_1 \oplus x_2)$ concernant les entrées et $HD(y_1, y_2) = HD(B(x_1 \oplus k), B(x_2 \oplus k))$ pour les sorties. En règle générale, la boîte B est une boîte de substitution (on prendra comme exemples les cas où B est la boîte-S de l'AES ou de PRESENT).

Pour le premier tour en particulier, x_1 et x_2 sont les octets de deux textes

clairs, qui sont supposés connus et seules les distances de Hamming des sorties apportent une information supplémentaire. La situation est similaire pour le dernier tour où seules les distances des entrées sont intéressantes, les chiffrés étant supposés connus.

Encore une fois, on adapte les outils développés pendant l'analyse des ASCA du chapitre 6 (N_B, $\#AI_L$, conditions suffisantes) à ce modèle particulier. On va voir qu'ils restent pertinents pour comprendre les résultats expérimentaux obtenus.

Supposons les entrées x_1 et x_2 connues et considérons maintenant l'idéal $I_{d_{out},x_1,x_2} \subset \mathbb{F}_2[K_1,\ldots,K_N,Y_1,\ldots,Y_n,Y'_1,\ldots,Y'_n]$ engendré par les équations représentant deux boîtes B (dont les bits de sorties sont modélisés par les variables Y_i et Y'_i), les équations de corps (6.1) et par $L_{d_{out}}$ (6.2) les équations représentant le fait que le poids de Hamming du XOR entre les sorties est égal à d_{out}, c'est-à-dire :

$$I_{d_{out},x_1,x_2} = \langle L_{d_{out}}(Y_1+Y'_1,\ldots,Y_n+Y'_n) \cup S_{B,x_1} \cup S_{B,x_2} \cup S_{\text{Field Eq.}} \rangle$$

Définition 30. *Dans le modèle de la distance de Hamming entre chiffrements successifs, l'immunité algébrique avec fuites d'une boîte B est défini par $AI_L(B,d_{out},x_1,x_2) = \min\{\deg(P), P \in I_{d_{out},x_1,x_2} \setminus I_{\text{Field Eq.}}\}$, le plus petit degré des polynômes appartenant à l'idéal I_{d_{out},x_1,x_2} privé des équations de corps. Le nombre des relations linéairement indépendantes de degré $AI_L(B,d_{out},x_1,x_2)$ est encore noté $\#AI_L(B,d_{out},x_1,x_2)$.*

Pour les mêmes raisons que dans la Section 7.2.1, l'immunité algébrique avec fuite vaut toujours 1 dans ce modèle, et $\#AI_L(B,d_{out},x_1,x_2)$ est donc égal au nombre de relations linéaires linéairement indépendantes.

De façon similaire au chapitre 6, on peut relier $\#AI_L$ avec le nombre de valeurs possibles pour k qui satisfont cette distance de Hamming (i.e. avec le nombre de points de $V(I_{d_{out},x_1,x_2})$). Pour x_1 et x_2 donnés, on pose donc, pour ce modèle particulier, $N_B(d_{out},x_1,x_2) = \#V(I_{d_{out},x_1,x_2})$ (Définition 29 du Chapitre 6) qui est alors égal au cardinal de l'ensemble

$$\{k \in \mathbb{F}_2^8 \text{ t.q. } HD(B(x_1 \oplus k), B(x_2 \oplus k)) = d_{out}\}$$

La proposition 9 du chapitre 6 n'est plus directement applicable dans ce modèle. Cependant, la même démonstration montre qu'on a maintenant l'inégalité suivante :

Proposition 13. *Soit n la taille de la boîte B. Si $AI_L(B,d_{out},x_1,x_2)$ est égal à 1 et que $N_B(d_{out},x_1,x_2)$ est non nul, alors*

$$N_B(d_{out},x_1,x_2) \geq 3n + 1 - \#AI_L(B,d_{out},x_1,x_2) \tag{7.1}$$

d_{out}	0	1	2	3	4	5	6	7	8
$N_B(d_{out}, x_1, x_2)$	0	0	32	48	64	64	32	16	0
$\#AI_L(B, d_{out}, x_1, x_2)$	0	0	15	12	9	9	14	20	0
Nb fixed bits	0	0	2	0	0	0	0	0	0

FIGURE 7.4 – modèle HD entre 2 chiffrements consécutifs avec PRESENT pour x_1=0xD8 et x_2=0xB1

d_{out}	0	1	2	3	4	5	6	7	8
$N_B(d_{out}, x_1, x_2)$	0	8	18	52	72	68	28	8	2
$\#AI_L(B, d_{out}, x_1, x_2)$	0	17	9	1	1	1	3	17	23
Nb fixed bits	0	0	0	0	0	0	0	0	4

FIGURE 7.5 – modèle HD entre 2 chiffrements consécutifs avec AES pour x_1=0xD8 et x_2=0xB1

À titre d'exemple, on a reporté dans les figures 7.4 et 7.5 les valeurs de $N_B(d_{out}, x_1, x_2)$ et $\#AI_L(B, d_{out}, x_1, x_2)$ obtenues avec PRESENT et AES pour x_1 et x_2 fixés à des valeurs arbitraires (x_1=0xD8 et x_2=0xB1). Dans ces exemples, on remarque que le nombre de relations linéaires est souvent important (bien que N_B soit grand). En fait, beaucoup de ces relations linéaires sont des égalités entre les $3n$ bits inconnus et se révèlent donc insuffisantes pour deviner la valeur de quelques bits de la clef. En supposant toujours l'hypothèse d'équirépartition des octets en entrées des boîtes-S, on peut calculer l'espérance $E(N_{PRESENT}(d_{out}, 0\text{xD8}, 0\text{xB1})) = 50 \simeq 2^{5.64}$ et $E(N_{AES}(d_{out}, 0\text{xD8}, 0\text{xB1})) \simeq 53.7 \simeq 2^{5.75}$. On a aussi calculé que $E(N_{PRESENT}) \simeq 71.97$ et $E(N_{AES}) \simeq 52.3$ en supposant l'équirépartition de l'ensemble des octets d'entrée (i.e. x_1, x_2 et k). Pour PRESENT, on remarque que la moyenne des N_B sur toutes les valeurs d'entrées possibles est plus grande que $E(N_{PRESENT}(d_{out}, 0\text{xD8}, 0\text{xB1}))$. En effet, beaucoup de couples d'entrées (x_1, x_2) conduisent à de très grands N_B. Par exemple lorsque $x_1 = x_2$ ou de manière équivalente, lorsqu'il n'y a qu'une seule distance possible sur les sorties (i.e. $d_{out} = 0$), on a donc

$$\forall x \in [0, \ldots, 255], N_B(0, x, x) = E(N_B) = 256$$

En pratique, des expériences sur le chiffrement par bloc PRESENT ont été menées. Les attaques par base de Gröbner nécessitaient beaucoup trop de mémoire pour être praticables dans ce contexte. Par contre, des attaques par

clef secrète	CF708A5E 7AC7F066 3FBA
1er texte clair	1A3759CA 97F9A3A9
2ème texte clair	64CB8CB9 DBC97346

nb de tours	8	10	12	15	16
$\prod N_B$ dernier tour	2^{47}	2^{48}	2^{50}	$2^{47.7}$	$2^{47.7}$
cryptominisat-2.9	2012	9984	2728	4581	19336
glucose2	1600	2794	12919	2716	17756

nb de tours	19	20	21	30	50
$\prod N_B$ dernier tour	$2^{43.5}$	2^{49}	2^{41}	$2^{47.4}$	$2^{47.2}$
cryptominisat-2.9	929	4080	2266	5973	11085
glucose2	4999	11144	2648	5050	2240

FIGURE 7.6 – Temps de résolution en secondes des expériences contre PRESENT avec la distance de Hamming de chiffrements consécutifs pour une clef et deux clairs fixés arbitrairement.

SAT solveur ont été possibles quand toutes les distances étaient connues sur tous les tours. Des temps de calculs pour les SAT solveurs cryptominisat-2.9 et glucose2 sont reportés dans la figure 7.6. Il faut tout de même remarquer que les temps de calculs dépendent énormément du SAT solveur utilisé, et varient beaucoup d'une instance à l'autre.

Suivant la condition suffisante de réussite, la recherche exhaustive sur les 64 bits utilisés de la dernière sous-clef peut être réduit à une complexité entre 2^{41} et 2^{50} par les informations de fuite (figure 7.6). Cependant, les SAT solveurs sont bien plus rapides puisqu'ils n'utilisent pas uniquement le dernier tour, mais aussi les informations des tours précédents (et particulièrement du premier tour) grâce à la simplicité de la dérivation de clef de PRESENT.

Comme dans la Section 7.2.1, les attaques contre PRESENT par calcul de bases de Gröbner ont nécessité trop de mémoire pour être en mesure de résoudre le système d'équations obtenu avec ce modèle de fuite.

Contrairement au modèle du poids de Hamming (cf. Chapitre 6), la connaissance du clair et du chiffré s'est révélée nécessaire dans ce modèle. En effet, toutes les tentatives d'attaques en clair/chiffré inconnus ont échoué lors des attaques par SAT solveurs. De même, les expériences sans fuites sur le premier tour et du dernier tour ont toutes échoué, ce qui confirmerait que les informations de fuite sur les tours extrémaux sont nécessaires.

Finalement, les expérimentations d'attaques contre AES avec ce modèle de fuite ont toutes échoué, (attaques par SAT solveurs et par calcul de bases de Gröbner), sûrement pour les mêmes raisons que dans la section 7.2.1 (scénario précédent avec distances de Hamming entre les entrées et les sorties des boîtes-S).

7.3 Un modèle pour le poids de Hamming avec erreurs ?

Les articles introduisant les attaques ASCA ([RS09a, RSVC09]) puis l'analyse de ces attaques (Chapitre 6 en particulier) supposaient toujours que les informations de fuite étaient très fiables et que les poids de Hamming récupérés lors de l'étape on-line ne contenaient pas d'erreur. Cette hypothèse vient du constat qu'un poids de Hamming erroné induit un système d'équations inconsistant. En effet, les équations modélisant le poids de Hamming erroné entrent en contradiction avec la solution attendue, le système entier obtenu ne contiendra donc aucune solution.

Cette hypothèse est cependant problématique : les informations supplémentaires obtenues par canaux auxiliaires proviennent généralement de mesures physiques (traces de consommation électrique par exemple), qui contiennent donc du bruit provenant de divers sources (cf. Chapitre 4). Suivant la quantité de bruit par rapport au signal utile, il y a une probabilité plus ou moins grande de faire une erreur dans la détermination de la valeur du poids de Hamming mesuré. Pour les attaques ASCA, l'étape d'acquisition nécessite donc des mesures de grande qualité (i.e. contenant très peu de bruit), généralement améliorées par un post-traitement de trace, comme par exemple la réduction du bruit en moyennant des traces obtenues par la répétition des mesures. Une réduction suffisante du bruit n'est cependant pas toujours possible suivant le contexte (par exemple quand seulement une trace de consommation est disponible) et dans tous les cas, une part de bruit sera toujours présent.

D'un point de vue pratique, il faut donc considérer l'étape de résolution en présence d'erreur. Une idée envisagée est d'utiliser d'autres méthodes de résolution des systèmes d'équations ([OKPW10, ZZG+12]) ou d'adapter certaines méthodes, comme les SAT solveurs par exemple, pour les rendre plus tolérants aux erreurs ([RSVC09]). Jusqu'à maintenant, ces approches se sont révélées très sensibles, le taux d'erreur supporté restant très faible et nécessitent une plus grande puissance de calcul. La question de l'application pratique de ces méthodes reste donc ouverte.

Bien qu'ayant toujours supposé les informations de fuite sans erreur, l'analyse théorique des attaques ASCA menée au Chapitre 6 a permis une bonne compréhension de l'étape algébrique (en particulier l'étape de résolution), ce qui pourrait donc être utile à la recherche de méthodes supportant la présence d'erreur. Cela nous avait d'ailleurs déjà amené à formuler une première proposition (voir Remarque 8 de la Section 6.4.1) pour réduire la présence d'erreur lors de l'étape de modélisation en systèmes d'équations polynomiales. Rappelons brièvement l'idée proposée : nous avons introduit la mesure N_B qui fournit un moyen d'estimer le niveau d'information apportée par tout poids de Hamming pour une boite noire B fixée. Les poids apportant le moins d'informations peuvent être omis sans grande conséquence pour l'efficacité de l'attaque. Avec un plus petit nombre de poids de Hamming pris en comptes, la probabilité d'ajout d'un poids erroné est donc réduite.

D'un autre coté, cette analyse a montré l'influence du modèle de fuite choisi. En particulier, on a vu que le modèle du poids de Hamming apportait beaucoup d'informations supplémentaires, à tel point qu'on pouvait même se passer du clair et du chiffré. D'autres modèles comme le modèle de la distance de Hamming, bien qu'apportant beaucoup moins d'information, peuvent cependant conduire à des attaques efficaces.

Cette constatation nous a conduit à la question de l'existence d'une modélisation différente des informations du poids de Hamming, qui aurait la particularité d'apporter moins d'information en contrepartie d'une plus grande tolérance aux erreurs. Plus précisément, pour estimer un poids de Hamming à partir de mesures (traces de consommation), on utilise généralement des attaques templates (voir Chapitre 4). Une probabilité de vraisemblance est donc associée à chaque hypothèse de valeur possible pour ce modèle de fuite. Une modélisation plus adaptée aux erreurs pourrait exploiter ces probabilités de vraisemblance. Une caractérisation plus précise du bruit serait surement nécessaire afin de justifier le choix d'un modèle d'erreur.

Un modèle d'erreur sur la détermination des poids de Hamming a été proposé dans [OKPW10], ainsi que dans [ZZG+12]. Il suppose que l'erreur de mesure correspond à 1 bit au plus. Il faut donc considérer la possibilité d'avoir, en plus du poids de Hamming donné, le poids directement supérieur ou directement inférieur. Bien que ce modèle s'appliquait dans un contexte différent (utilisant la résolution de *pseudo-Boolean optimization* problem (PBOPT) pour [OKPW10]), l'idée peut être adaptée en proposant un autre modèle de fuite.

w_{in} \ w_{out}	[0, 1]	[1, 2]	[2, 3]	[3, 4]	[4, 5]	[5, 6]	[6, 7]	[7, 8]
[0, 1]	0	0	0	10	10	0	0	0
[1, 2]	0	2	4	28	30	6	2	0
[2, 3]	0	10	24	40	50	34	10	0
[3, 4]	4	13	47	51	57	56	18	6
[4, 5]	8	13	47	59	49	44	22	10
[5, 6]	6	16	28	42	34	22	16	4
[6, 7]	2	12	12	18	18	6	4	0
[7, 8]	0	6	6	4	4	0	0	0

FIGURE 7.7 – $1024\mathbb{P}(w_{in}, w_{out})$ avec 2 boîtes-S de PRESENT

w_{in} \ w_{out}	[0, 1]	[1, 2]	[2, 3]	[3, 4]	[4, 5]	[5, 6]	[6, 7]	[7, 8]
[0, 1]	0	0	0	9	9	0	0	0
[1, 2]	0	2	4	28	30	6	2	0
[2, 3]	0	10	24	40	50	34	10	0
[3, 4]	3	13	47	51	57	56	18	5
[4, 5]	7	13	47	59	49	44	22	9
[5, 6]	6	16	28	42	34	22	16	4
[6, 7]	2	12	12	18	18	6	4	0
[7, 8]	0	5	5	4	4	0	0	0

FIGURE 7.8 – $N_B(w_{in}, w_{out})$ avec 2 boîtes-S de PRESENT

w_{in} \ w_{out}	[0, 1]	[1, 2]	[2, 3]	[3, 4]	[4, 5]	[5, 6]	[6, 7]	[7, 8]
[0, 1]	0	0	0	8	8	0	0	0
[1, 2]	0	15	13	4	4	11	15	0
[2, 3]	0	8	2	0	0	1	7	0
[3, 4]	14	4	0	0	0	0	4	12
[4, 5]	10	8	0	0	0	0	6	8
[5, 6]	11	8	0	0	2	3	8	13
[6, 7]	15	8	8	3	3	12	14	0
[7, 8]	0	12	12	13	13	0	0	0

FIGURE 7.9 – Nombre de relations linéaires avec 2 boîtes-S de PRESENT

w_{in} \ w_{out}	[0, 1]	[1, 2]	[2, 3]	[3, 4]	[4, 5]	[5, 6]	[6, 7]	[7, 8]
[0, 1]	0	0	0	0	0	0	0	0
[1, 2]	0	8	6	0	0	4	8	0
[2, 3]	0	0	0	0	0	0	0	0
[3, 4]	10	0	0	0	0	0	0	6
[4, 5]	0	0	0	0	0	0	0	0
[5, 6]	2	0	0	0	0	0	0	4
[6, 7]	10	0	0	0	0	0	8	0
[7, 8]	0	6	6	4	4	0	0	0

FIGURE 7.10 – Nombre de bits fixés avec 2 boîtes-S de PRESENT

Une extension du modèle du poids de Hamming serait donc de considérer que l'on connait, non plus un poids de Hamming, mais plusieurs poids de Hamming possibles pour une variable intermédiaire d'un chiffrement. L'ensemble des poids de Hamming possibles pourrait être construit à partir des probabilités de vraisemblance obtenues par analyse des mesures, en choisissant les valeurs les plus probables. Pour tenir compte de cette information supplémentaire, il faudra donc construire un système d'équations uniquement vérifié par tous les points ayant un de ses poids de Hamming possibles. Malheureusement, une telle modélisation par un système d'équations ne tient pas compte des probabilités des poids possibles. Cependant, il permet déjà de gérer les situations où les informations de fuite sont plus incertaines (i.e. à un bit près).

Nous nous proposons de considérer ce modèle étendu du poids de Hamming dans le cas où seulement 2 poids de Hamming adjacents sont possibles Les figures 7.8, 7.9 et 7.10 présentent les caractéristiques de ce modèle de fuite avec la boîte-S de PRESENT. Contrairement au modèle du poids de Hamming simple (Chapitre 6), il arrive que l'immunité algébrique avec fuite soit égale à 2 pour certaines fuites. Cela veut donc dire qu'il n'y a pas forcément de relation linéaire dans l'idéal considéré. D'un autre coté, il reste beaucoup de fuites qui conduisent à l'apparition de nombreuses relations linéaires, voire qui permettent de fixer la valeurs de certains bits intermédiaires.

Afin d'appliquer notre critère suffisant de réussite, il nous faut calculer la probabilité d'obtenir des informations sur les poids de Hamming possibles. En supposant encore l'équirépartition des octets en entrées des boîtes-S, ainsi que leur équirépartition dans les intervalles possibles, on obtient les probabilités données dans la figure 7.7. On remarque que contrairement aux modèles précédents, les valeurs prises par la variable aléatoire N_{PRESENT} ne correspondent plus aux probabilités d'apparition des informations de fuite. À partir de ces données, on peut calculer que $E(N_{\text{PRESENT}}) \simeq 35.5$ et que l'espérance du nombre d'équations linéaires est d'environ 2.6. Ce qui permet d'estimer que le coût moyen de la recherche exhaustive de la valeur d'entrée d'une couche de substitution entière est borné par $E(N_{\text{PRESENT}})^8 \simeq 2^{41.2}$. Bien que ce coût soit élevé (bien plus grand que dans le cas du modèle du poids de Hamming, voir Section 6.4.1), il reste inférieur à la valeur du critère calculé pour le modèle de la distance de Hamming (Section 7.2.1). On s'attend donc à ce que les attaques par SAT solveurs avec ce modèle soient généralement plus efficaces qu'avec la distance de Hamming. Quelques expériences réussies sur PRESENT confirment que ce modèle apporte suffi-

samment d'informations pour obtenir un bon taux de réussite. Inversement, aucune attaque par calcul de base de Gröbner n'a pu aboutir. De même, les expériences sur l'AES avec ce modèle de fuite sont restées négatives, du fait du nombre insuffisant d'informations apportées dans ce modèle (informations sur les poids de Hamming entre les boîtes-S). Il pourrait être intéressant de considérer des informations de fuite venant d'autres parties de l'AES, comme celles provenant du MixColumns par exemple, leur influence précise sur l'efficacité de la résolution n'étant pas évidente a priori. Une application de ce type de modèle de fuite proposé dans [ZZG+12], dans le cas des *cache attacks* contre l'AES, confirme, malgré un taux d'erreur important, sa grande efficacité pratique lorsque un grand nombre d'informations de fuite peuvent être obtenues.

Ce modèle du poids de Hamming avec possibilité d'erreur a cependant permis de montrer que l'étape de modélisation (des informations de fuite) pouvaient aussi, en complément de l'amélioration des étapes d'acquisition et de résolution, apporter des réponses au problème de la gestion des erreurs. De plus, un compromis entre ce modèle et le modèle du poids de Hamming du chapitre 6 est envisageable à partir des probabilités de vraisemblance fournies par les attaques template. En effet, on a considéré dans cette partie que l'information sur le poids de Hamming étaient toujours donnée par deux valeurs de poids possibles. Mais il n'est pas exclu qu'une étape d'acquisition de qualité soit capable de fournir quelques valeurs précises de poids de Hamming, ce qui améliorerait d'autant l'efficacité de l'attaque.

7.4 Attaques par faute avec méthodes algébriques

L'analyse de faute est une technique de cryptanalyse qui exploite les erreurs qui se produisent dans des calculs cryptographiques (cf Chapitre 4) afin de récupérer des informations sur la clef secrète. L'analyse des différences entre les textes correctement chiffrés et ceux produits par des calculs erronés est une technique d'analyse de faute sur les chiffrements par blocs, aussi appelée *Differential Fault Analysis* (DFA). De nombreux articles ont déjà été écrits sur les attaques DFA contre l'AES, dont les principaux sont [PQ03, BS03, Gir05, Muk09, TM09]. Les améliorations proposées pour les attaques par faute consistent à réduire le nombre de fautes nécessaires, à diversifier les modèles de faute, à exploiter des fautes sur des tours de plus en plus tôt, ou encore à étendre les attaques aux versions 192 et 256 bits de l'AES.

En considérant chaque chiffré fauté comme une information de fuite, les attaques par DFA peuvent être modélisées comme des attaques ASCA dans

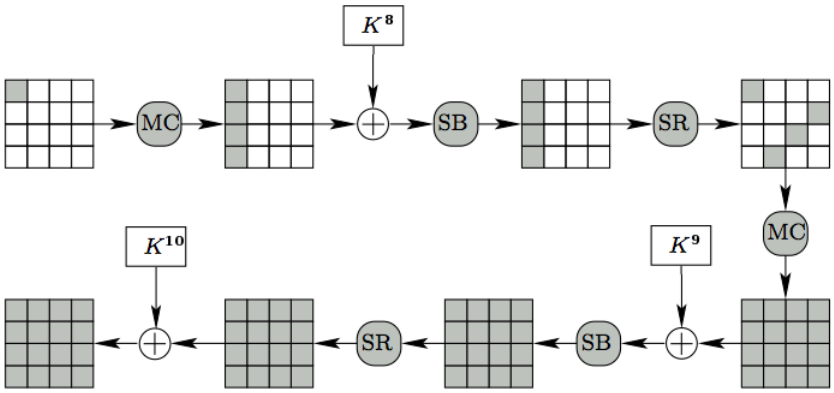


FIGURE 7.11 – Propagation d'une erreur sur un octet à l'entrée du MixColumns du tour 8

un modèle particulier. Retrouver la clef secrète revient donc à résoudre un système d'équations polynomiales modélisant notre chiffrement et les informations de fuite apportées par les chiffrés fautés suivant un modèle de faute donné. Ainsi nous nous inscrivons toujours dans l'objectif de ce chapitre qui consiste à évaluer l'impact d'informations extérieures sur l'efficacité des méthodes de cryptanalyse algébrique. Mais contrairement aux articles précédemment cités sur les DFA, ce point de vue permet d'apporter une méthode systématique d'attaque par faute, et ainsi, d'estimer le nombre minimum de fautes nécessaire pour un de ces modèles d'attaque. De plus, il semble que ce soit aussi le moyen le plus efficace pour utiliser des modèles de faute complexes (fautes sur plusieurs octets par exemple).

Cette résolution des attaques par faute avec des méthodes algébriques a été testée avec succès contre l'AES-128 par SAT solveur, en suivant le principe de l'attaque classique de Piret et Quisquater [PQ03], qui consiste à exploiter une faute sur un octet au début du tour 8 (voir figure 7.11). Les auteurs ont montré qu'un seul chiffré avec une faute provoquée avant le MixColumns du tour 8 permettait de réduire le nombre de clefs possibles à 2^{40} (réduit à 2^{34} dans [GT10]).

Il est à noter que le choix de la modélisation des fautes en équations polynomiales semble avoir une grande influence sur l'efficacité de la résolution. Plus précisément, nous avons commencé par modéliser par un système

d'équations le chiffrement complet du texte clair en un chiffré correct (tous deux connus). Avec de nouvelles variables représentant les valeurs intermédiaires après l'apparition de la faute (cases grises sur la figure 7.11), les équations modélisant le chiffrement de la faute sont ajoutées au système. La valeur du chiffré fauté étant connu, elle est aussi utilisée pour instancier les variables correspondantes. Bien que ce système d'équations semble contenir toutes les informations supplémentaires apportées par le chiffré fauté dans ce modèle de faute donné, il ne nous a pas été possible de le résoudre par SAT solveur.

La seule modélisation en système d'équations polynomiales qui a conduit à des attaques efficaces par SAT solveur est celle dont les nouvelles variables représentent plutôt les **différences** Δ entre les valeurs intermédiaires correctes x et les valeurs fautées $(\Delta \oplus x)$. Les équations ajoutées doivent donc représenter l'évolution jusqu'à la fin du chiffrement des différences entre les valeurs intermédiaires correctes et les valeurs fautées. En particulier, au niveau des boîtes-S, les systèmes d'équations doivent être de la forme

$$\Delta_S = S(x \oplus k) \oplus S(\Delta_x \oplus x \oplus k)$$

où Δ_x et Δ_S représentent les différences respectivement avant et après la substitution.

Contrairement aux précédents modèles étudiés (poids de Hamming au chapitre 6 et distance de Hamming aux sections 7.2.1 et 7.2.2), notre analyse des ASCA (le critère de réussite particulièrement) ne semble pas bien adapté à ce modèle de fuite particulier. Une analyse précise des systèmes d'équations obtenues avec ce modèle (suivant le modèle du chapitre 6, par exemple en utilisant des calculs de bases de Gröbner) reste donc à faire pour étudier des modèles de faute plus complexes. Une meilleure compréhension permettrait ainsi d'avoir une méthode générale et systématique pour exploiter les fautes. Cependant, dans ce modèle de faute simple, les arguments justifiant la complexité de la résolution sont les mêmes que ceux avancés dans [PQ03].

Suivant le même principe, on peut quand même construire des systèmes d'équations modélisant des fautes différentes afin de chercher à étendre les attaques par faute. Malheureusement, les fautes produites avant le MixColumns du tour 7 ne semblent pas exploitables par cette modélisation par manque d'un bon distingueur. Inversement, les fautes produites après le MixColumns du tour 8 sont difficilement exploitables par méthodes algébriques.

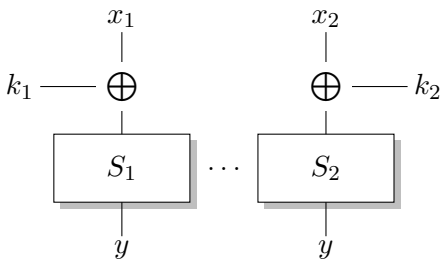


FIGURE 7.12 – Schéma d'une collision interne entre deux boîtes de substitutions S_1 et S_2 qui donne l'égalité $S_1(x_1 \oplus k_1) = S_2(x_2 \oplus k_2)$.

Ceci est dû à l'augmentation importante du nombre de fautes nécessaires et donc de la taille du système à résoudre ([KIM11]). L'utilisation de méthodes algébriques pour repousser les limites des attaques par faute nécessite donc sûrement une modélisation plus astucieuse. Des expériences sur d'autres cryptosystèmes et l'étude de modèles de faute plus réalistes restent encore à faire.

7.5 Les "attaques par collision"

Les attaques dites par détections de "collision interne" ([KSP04, LMV04, Bog07, BKP08, Bog08, MME10]) peuvent aussi être vues comme des attaques ASCA pour un modèle particulier. Dans ce cas, le modèle de fuite fournit des listes de "collision interne" *i.e.* qu'il permet de d'identifier les fonctions du cryptosystème qui renvoient les mêmes valeurs de sortie. Il faut bien remarquer qu'avec ce modèle, aucun bit intermédiaire n'est connu. On obtient uniquement des égalités entre certaines variables intermédiaires.

Typiquement, il peut y avoir de nombreuses collisions au niveau des boîtes-S (cf. figure 7.12). Elles ont été introduites dans [Bog07] appliquées sur l'AES. Ce type de collision se traduit facilement par des équations sur les bits de clef : $k_1 \oplus k_2 = x_1 \oplus x_2$.

Les collisions internes peuvent aussi se produire au niveau des octets en sorties des MixColumns de l'AES [KSP04]. Un exemple de telle collision est schématisé par la figure 7.13 qui fournie une équation de la forme

$$\begin{aligned} &3S_{i_1}(x_{i_1} \oplus k_{i_1}) \oplus S_{i_2}(x_{i_2} \oplus k_{i_2}) \oplus S_{i_3}(x_{i_3} \oplus k_{i_3}) \oplus 2S_{i_4}(x_{i_4} \oplus k_{i_4}) \\ &= S_{j_1}(x_{j_1} \oplus k_{j_1}) \oplus 2S_{j_2}(x_{j_2} \oplus k_{j_2}) \oplus 3S_{j_3}(x_{j_3} \oplus k_{j_3}) \oplus S_{j_4}(x_{j_4} \oplus k_{j_4}) \end{aligned}$$

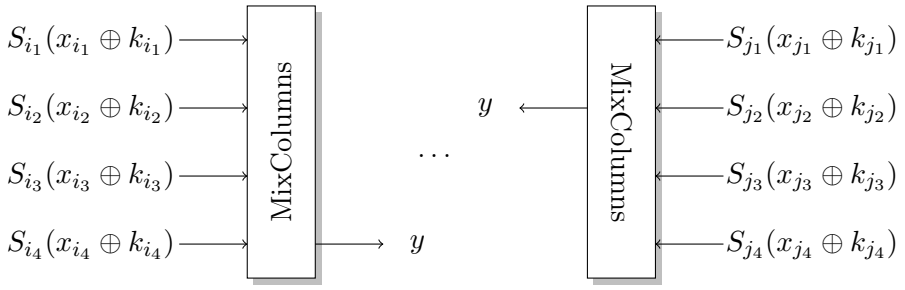


FIGURE 7.13 – Schéma d'une collision interne entre deux sorties de MixColumns.

Les figures 7.12 et 7.13 montrent des exemples de collision entre deux boîtes-S et entre deux MixColumns respectivement, mais d'autres collisions sont possibles (par exemple une collision entre un octet en sortie d'une boîte-S et un octet en sortie d'un MixColumns). Un modèle de fuite de type collision peut donc fournir deux octets égaux en sortie de deux opérations, qui peuvent d'ailleurs être différentes, sur n'importe quel tour, ou même sur des chiffrements différents. Mais jusqu'à maintenant, seules les collisions intervenant sur les tours extrémaux de l'AES (1, 2, 9, et 10) ont pu être mises en équations et exploitées (car dans ces équations, les valeurs des x_i sont connues). La complexité de résolution des systèmes ainsi obtenus en fonction du nombre et du type de collisions détectées est étudiée en détails dans [Bog07] et [BKP08].

En modélisant les attaques par collision comme des attaques ASCA, chaque chiffrement sera entièrement représenté par des systèmes d'équations et les équations de fuite seront ajoutées à ces systèmes. L'inconvénient est qu'on obtient alors des systèmes d'équations énormes puisque tout est modélisé, alors que seules les équations modélisant les collisions peuvent suffire, du moins pour les tours extrémaux (cf. les articles sur les collision, en particulier [BKP08]). On a cependant pu modéliser des attaques ASCA avec un modèle de fuite fournissant toutes les collisions sur les tours 1, 2, 9 et 10 de l'AES et obtenir des résultats comparables avec ceux déjà publiés en utilisant la résolution par SAT solveurs.

Ce changement de point de vue apporte quand même la capacité de produire facilement des équations pouvant modéliser n'importe quelle collision, en particulier des collisions faisant intervenir des tours internes. Cela permettrait d'étendre ces attaques et d'outrepasser les contre-mesures qui se

trouvent souvent uniquement sur les premiers et derniers tours. Nous avons donc essayé de tirer parti des collisions uniquement sur les tours internes 3 à 8 de l'AES. Les informations de fuite sont simplement rajoutées aux systèmes sous formes d'égalités entre variables intermédiaires. Malheureusement, les systèmes d'équations ainsi obtenus n'ont pas pu être résolus ni par calcul bases de Gröbner ni par des SAT solveurs, malgré un grand nombre de chiffrements différents pris en compte. Des méthodes de résolution plus adaptées ou une autre modélisation semblent nécessaires pour pouvoir étendre les attaques par collision aux tours internes. De plus le critère de réussite proposé pour les ASCA semble être assez mal adapté à ce modèle de fuite. En effet, ce critère a été défini pour s'appliquer sur un seul tour alors que les collisions sont réparties entre plusieurs tours différents.

7.6 Conclusion

Les attaques ASCA peuvent être étendues sous des hypothèses plus réalistes en utilisant différents modèles de fuite suivant le contexte et l'implémentation visée. On a vu que l'efficacité de l'attaque dépendait beaucoup de la quantité d'information apportée par le modèle de fuite. Par exemple, les attaques sont plus difficiles avec le modèle de la distance de Hamming qu'avec le poids de Hamming, en particulier l'étape de résolution demande bien plus de calculs. On a ansi pu adapter notre critère de réussite pour quelques autres modèles et montrer qu'il fournissait une bonne mesure pour estimer la quantité d'information qu'ils apportaient. Cela nous à aussi amené à considérer un modèle de fuite rendant les ASCA possibles en présence d'erreurs.

L'introduction de nouveaux modèles de fuite ont aussi permis de généraliser les attaques par faute et les attaques par collision, en les considérant comme des attaques ASCA. Cependant, on a pu voir que notre critère ne s'adaptait pas naturellement à ces modèles particuliers. En effet, notre critère s'appuie toujours sur une étude locale de l'influence de la fuite. Or ces modèles de fuite font interagir des opérations qui peuvent être très éloignées dans le cryptosystème. Un critère plus général prenant en compte des dépendances sur des tours différents est donc nécessaire pour mieux analyser les ASCA dans le cas général.

Bibliographie

[AA05] Frederik Armknecht and Gwénolé Ars. Introducing a New Variant of Fast Algebraic Attacks and Minimizing Their Successive Data Complexity. In *Mycrypt*, pages 16–32, 2005.

[ABDM00] Mehdi-Laurent Akkar, Régis Bevan, Paul Dischamp, and Didier Moyart. Power Analysis, What Is Now Possible... In *ASIACRYPT*, pages 489–502, 2000.

[AC10] Martin Albrecht and Carlos Cid. Cold Boot Key Recovery using Polynomial System Solving with Noise. In *2nd International Conference on Symbolic Computation and Cryptography*, 2010.

[ACFP12] Martin Albrecht, Carlos Cid, Jean-Charles Faugère, and Ludovic Perret. On the relation between the MXL family of algorithms and Gröbner basis algorithms. *Journal of Symbolic Computation*, 47 :926–941, 2012. in press.

[AD93] Leonard M. Adleman and Jonathan DeMarrais. A Subexponential Algorithm for Discrete Logarithms over All Finite Fields. In *Advances in Cryptology - CRYPTO '93, 13th Annual International Cryptology Conference, Proceedings*, Lecture Notes in Computer Science, pages 147–158, 1993.

[ADH94] Leonard M. Adleman, Jonathan DeMarrais, and Ming-Deh A. Huang. A Subexponential Algorithm for Discrete Logarithms over the Rational Subgroup of the Jacobians of Large Genus Hyperelliptic Curves over Finite Fields. In *Algorithmic Number Theory, Proceedings*, pages 28–40. Springer, 1994.

[ADH99] Leonard M. Adleman, Jonathan DeMarrais, and Ming-Deh A. Huang. A Subexponential Algorithm for Discrete Logarithms over Hyperelliptic Curves of Large Genus over GF(q). *Theoretical Computer Science*, 226(1-2) :7–18, 1999.

[AF05] Gwénolé Ars and Jean-Charles Faugère. Algebraic Immunities of Functions over Finite Fields. Research Report RR-5532, INRIA, 2005.

[AFI+04] Gwenolé Ars, Jean-Charles Faugère, Hideki Imai, Mitsuru Kawazoe, and Makoto Sugita. Comparison Between XL and Gröbner Basis Algorithms. In Pil Joong Lee, editor, *Advances in Cryptology - ASIACRYPT 2004*, volume 3329 of *Lecture Notes in Computer Science*, pages 157–167. Springer Berlin / Heidelberg, 2004.

[AIK+] K. Aoki, T. Ichikawa, M. Kanda, M. Matsui, S. Moriai, J. Nakajima, and T. Tokita. Camellia : A 128-Bit Block Cipher Suitable for Multiple Platforms.

[Ajt98] Miklós Ajtai. The Shortest Vector Problem in L_2 is *NP*-hard for Randomized Reductions (Extended Abstract). In *Proceedings of the 30th Symposium on the Theory of computing (STOC 1998)*, pages 10–19. ACM Press, 1998.

[Ajt06] Miklós Ajtai. Generating random lattices according to the invariant distribution. draft, March 2006.

[AK06] Frederik Armknecht and Matthias Krause. Constructing Single- and Multi-Output Boolean Functions with Maximal Immunity. In *Proceedings of ICALP 2006*, pages 180–191. Lecture Notes of Computer Science 4052, 2006.

[AKS01] Miklós Ajtai, Ravi Kumar, and Dandapani Sivakumar. A Sieve Algorithm for the Shortest lattice Vector Problem. In *Proceedings of the 33rd Annual ACM Symposium on the Theory of Computing (STOC 2001)*, pages 601–610. ACM Press, 2001.

[AL94] W. Adams and P. Loustaunau. *An Introduction to Gröbner Bases*. Amer Mathematical Society, 7 1994.

[Ars05] Gwénolé Ars. *Applications des bases de Gröbner en cryptographie*. PhD thesis, University of Rennes, 2005.

[AS09] Gilles Audemard and Laurent Simon. Predicting learnt clauses quality in modern SAT solvers. In *Proceedings of the 21st international Joint Conference on Artificial Intelligence*, IJCAI'09, pages 399–404, 2009.

[AS11a] Gilles Audemard and Laurent Simon. Glucose's Home Page. 2011.

[AS11b] Gilles Audemard and Laurent Simon. GlueMiniSat's Home Page. 2011.

[Bar04] Magali Bardet. *Étude des systèmes algébriques surdéterminés. Applications aux codes correcteurs et à la cryptographie.* PhD thesis, Université de Paris VI, 2004.

[BCO04] Eric Brier, Christophe Clavier, and Francis Olivier. Correlation Power Analysis with a Leakage Model. In *CHES'04*, pages 16–29, 2004.

[BCP97] Wieb Bosma, John Cannon, and Catherine Playoust. The MAGMA algebra system : the user language. *Journal of Symbolic Computation*, 24 :235–265, 1997.

[BDL97] Dan Boneh, Richard A. DeMillo, and Richard J. Lipton. On the importance of checking cryptographic protocols for faults. In *Proceedings of the 16th annual international conference on Theory and application of cryptographic techniques*, EUROCRYPT'97, pages 37–51. Springer-Verlag, 1997.

[BFP10] Luk Bettale, Jean-Charles Faugère, and Ludovic Perret. Hybrid approach for solving multivariate systems over finite fields. *Journal of Mathematical Cryptology*, 3 :177–197, 2010.

[BFS04] Magali Bardet, Jean-Charles Faugère, and Bruno Salvy. On the Complexity of Gröbner Basis Computation of Semi-Regular Overdetermined Algebraic Equations. In *Proc. of International Conference on Polynomial System Solving (ICPSS)*, pages 71–75, 2004.

[BFSY05] Magali Bardet, Jean-Charles Faugère, Bruno Salvy, and Bo-Yin Yang. Asymptotic Behaviour of the Degree of Regularity of Semi-Regular Polynomial Systems. In *Proc. of MEGA 2005, Eighth Inter. Symposium on Effective Methods in Algebraic Geometry*, 2005.

[BGM97] Mihir Bellare, Shafi Goldwasser, and Daniele Micciancio. "Pseudo-Random" Number Generation Within Cryptographic Algorithms : The DSS Case. In *CRYPTO*, volume 1294 of *Lecture Notes in Computer Science*, pages 277–291. Springer, 1997.

[Bie11] Armin Biere. Lingeling and Plingeling. 2011.

[BKL+07] A. Bogdanov, L. R. Knudsen, G. Le, C. Paar, A. Poschmann, M. J. B. Robshaw, Y. Seurin, and C. Vikkelsoe. PRESENT : An Ultra-Lightweight Block Cipher. In *CHES'07*. Springer, 2007.

[BKP08] Andrey Bogdanov, Ilya Kizhvatov, and Andrei Pyshkin. Algebraic Methods in Side-Channel Collision Attacks and Practical

Collision Detection. In *INDOCRYPT*, volume 5365 of *Lecture Notes in Computer Science*, pages 251–265. Springer, 2008.

[BKW93] Thomas Becker, Heinz Kredel, and Volker Weispfenning. *Gröbner Bases : A Computational Approach to Commutative Algebra*. Springer-Verlag, London, UK, 0 edition, 4 1993.

[Bog07] Andrey Bogdanov. Improved side-channel collision attacks on AES. In *Selected Areas in Cryptography*, volume 4876 of *Lecture Notes in Computer Science*. Springer, 2007.

[Bog08] Andrey Bogdanov. Multiple-differential side-channel collision attacks on AES. In *CHES*, volume 5154 of *Lecture Notes in Computer Science*, pages 30–44. Springer, 2008.

[Bro11] Robert G. Brown. DieHarder : A Random Number Test Suite. URL `http://www.phy.duke.edu/~rgb/General/dieharder.php`, 2011. C program archive dieharder, version 3.29.4b.

[BS91] Eli Biham and Adi Shamir. Differential Cryptanalysis of DES-like Cryptosystems. *Journal of Cryptology*, 4(1) :3–72, 1991.

[BS97] Eli Biham and Adi Shamir. Differential Fault Analysis of Secret Key Cryptosystems. In *Advances in Cryptology - CRYPTO '97*, volume 1294 of *Lecture Notes in Computer Science*, pages 513–525. Springer, 1997.

[BS03] Johannes Blomer and Jean-Pierre Seifert. Fault Based Cryptanalysis of the Advanced Encryption Standard (AES). In *Financial Cryptography*, volume 2742 of *Lecture Notes in Computer Science*, pages 162–181. Springer, 2003.

[Buc65] Bruno Buchberger. *Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal*. PhD thesis, University of Innsbruck, 1965.

[Buc06a] Bruno Buchberger. Bruno Buchberger's PhD thesis 1965 : An algorithm for finding the basis elements of the residue class ring of a zero dimensional polynomial ideal. *Journal of Symbolic Computation*, 41(3-4), 2006.

[Buc06b] Bruno Buchberger. Comments on the translation of my phd thesis. *Journal of Symbolic Computation*, 41(3-4) :471–474, 2006.

[Car09] Claude Carlet. *On the algebraic immunities and higher order nonlinearities of vectorial Boolean functions*, volume 13 of *NATO Science for Peace and Security Series, D : Information and Communication Security*, pages 104–116. IOS Press, 2009.

[Car10] Claude Carlet. *Vectorial Boolean Functions for Cryptography*, pages 398–469. Boolean Models and Methods in Mathematics, Computer Science, and Engineering. Cambridge University Press, 2010.

[Cas97] John William Scott Cassels. *An Introduction to the Geometry of Numbers*. Classics in Mathematics. Springer-Verlag, 1997. corrected reprint of the 1971 edition.

[CB07] Nicolas Courtois and Gregory V. Bard. Algebraic Cryptanalysis of the Data Encryption Standard. In *Cryptography and Coding, 11th IMA International Conference*, volume 4887 of *Lecture Notes in Computer Science*, pages 152–169. Springer, 2007.

[CFGR12] Claude Carlet, Jean-Charles Faugère, Christopher Goyet, and Guénaël Renault. Analysis of the Algebraic Side Channel Attack. *Journal of Cryptographic Engineering*, pages 1–18, 2012.

[CJRR99] Suresh Chari, Charanjit S. Jutla, Josyula R. Rao, and Pankaj Rohatgi. Towards Sound Approaches to Counteract Power-Analysis Attacks. In *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, pages 398–412. Springer-Verlag, 1999.

[CKPS00] Nicolas Courtois, Alexander Klimov, Jacques Patarin, and Adi Shamir. Efficient Algorithms for Solving Overdefined Systems of Multivariate Polynomial Equations. In *Advances in Cryptology - EUROCRYPT 2000, International Conference on the Theory and Application of Cryptographic Techniques*, volume 1807 of *Lecture Notes in Computer Science*, pages 392–407. Springer, 2000.

[CL05] Carlos Cid and Gaëtan Leurent. An Analysis of the XSL Algorithm. In *ASIACRYPT*, pages 333–352, 2005.

[CLO07] David A. Cox, John Little, and Donal O'Shea. *Ideals, Varieties, and Algorithms : An Introduction to Computational Algebraic Geometry and Commutative Algebra, 3/e (Undergraduate Texts in Mathematics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2007.

[CM03] Nicolas Courtois and Willi Meier. Algebraic Attacks on Stream Ciphers with Linear Feedback. In *EUROCRYPT*, pages 345–359, 2003.

[CNK00] Jean-Sébastien Coron, David Naccache, and Paul C. Kocher. Statistics and secret leakage. In *Financial Cryptography, FC 2000 Proceedings*, volume 1962 of *Lecture Notes in Computer Science*, pages 492–508. Springer, 2000.

[Coo71] Stephen Cook. The Complexity of Theorem-Proving Procedures. In *Proceedings of the third annual ACM symposium on Theory of computing*, STOC '71, pages 151–158. ACM, 1971.

[Cop96a] Don Coppersmith. Finding a Small Root of a Bivariate Integer Equation ; Factoring with High Bits Known. In *EUROCRYPT*, pages 178–189, 1996.

[Cop96b] Don Coppersmith. Finding a Small Root of a Univariate Modular Equation. In *EUROCRYPT*, pages 155–165, 1996.

[CP02] Nicolas Courtois and Josef Pieprzyk. Cryptanalysis of Block Ciphers with Overdefined Systems of Equations. In *ASIACRYPT*, pages 267–287, 2002.

[CRR02] Suresh Chari, Josyula R. Rao, and Pankaj Rohatgi. Template Attacks. In *Cryptographic Hardware and Embedded Systems - CHES 2002*, volume 2523 of *Lecture Notes in Computer Science*, pages 13–28. Springer, 2002.

[DBM+08] Jintai Ding, Johanes Buchmann, Mohamed Saied Emam Mohamed, Wael Said Abd Elmageed Mohamed, and Ralph-Philipp Weinmann. MutantXL. In *Symbolic Computation and Cryptography – SCC 2008*, pages 16–22, 2008.

[DH76] Whitfield Diffie and Martin E. Hellman. Multiuser Cryptographic Techniques. In *AFIPS National Computer Conference*, volume 45 of *AFIPS Conference Proceedings*, pages 109–112, 1976.

[DLL62] Martin Davis, George Logemann, and Donald W. Loveland. A Machine Program for Theorem-Proving. *Commun. ACM*, 5(7) :394–397, 1962.

[DP60] Martin Davis and Hilary Putnam. A Computing Procedure for Quantification Theory. *J. ACM*, 7(3) :201–215, 1960.

[EGH07] M. Abdelaziz Elaabid, Sylvain Guilley, and Philippe Hoogvorst. Template Attacks with a Power Model. *IACR Cryptology ePrint Archive*, 2007 :443, 2007.

[Eis99] David Eisenbud. *Commutative Algebra. With a View Toward Algebraic Geometry.* Berlin : Springer-Verlag, 0 edition, 2 1999.

[ElG85] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *Proceedings of CRYPTO 84 on Advances in cryptology*, pages 10–18. Springer-Verlag New York, Inc., 1985.

[EM07] Mohamed Elkadi and Bernard Mourrain. *Introduction à la résolution des systèmes polynomiaux*, volume 59 of *Mathématiques et Applications*. Springer, 2007.

[ES03] Niklas Eén and Niklas Sörensson. An Extensible SAT-solver. In *SAT*, volume 2919 of *Lecture Notes in Computer Science*, pages 502–518. Springer, 2003.

[Fau99] Jean-Charles Faugère. A New Efficient Algorithm for Computing Gröbner bases (F4). In *Journal of Pure and Applied Algebra*, pages 75–83. ACM Press, 1999.

[Fau02] Jean-Charles Faugère. A New Efficient Algorithm for Computing Gröbner bases without reduction to zero (F5). In *Proceedings of the 2002 International Symposium on Symbolic and Algebraic Computation*, ISSAC '02, pages 75–83, New York, NY, USA, 2002. ACM.

[FdVP08] Jean-Charles Faugère, Françoise Levy dit Vehel, and Ludovic Perret. Cryptanalysis of MinRank. In *CRYPTO*, pages 280–296, 2008.

[FGLM93] Jean-Charles Faugère, Patrizia M. Gianni, Daniel Lazard, and Teo Mora. Efficient computation of zero-dimensional gröbner bases by change of ordering. *J. Symb. Comput.*, 16(4) :329–344, 1993.

[FGR11] Jean-Charles Faugère, Christopher Goyet, and Guénaël Renault. Algebraic Side Channel Analysis. In *COSADE'11 : The 2nd International Workshop on Constructive Side-Channel Analysis and Secure Design*, pages 1–6, Fraunhofer SIT, 2011.

[FGR12] Jean-Charles Faugère, Christopher Goyet, and Guénaël Renault. Attacking (EC)DSA Given Only an Implicit Hint. In *Conference on Selected Areas of Cryptography*, Lecture Notes in Computer Science, pages 1–12, Ontario, 2012. Springer Berlin / Heidelberg. accepted.

[FIP94] FIPS. *Digital Signature Standard (DSS)*. National Institute of Standards and Technology (NIST), 1994.

[FIP01] FIPS. *Advanced Encryption Standard (AES)*. pub-NIST, pub-NIST :adr, nov 2001.

[FIP09] FIPS. *Digital Signature Standard (DSS)*. pub-NIST, pub-NIST :adr, 2009.

[FJ03] Jean-Charles Faugère and Antoine Joux. Algebraic Cryptanalysis of Hidden Field Equation (HFE) Cryptosystems Using Gröbner Bases. In *CRYPTO*, pages 44–60, 2003.

[FM07] Simon Fischer and Willi Meier. Algebraic Immunity of S-Boxes and Augmented Functions. In *FSE*, pages 366–381, 2007.

[FMR10] Jean-Charles Faugère, Raphaël Marinier, and Guénaël Renault. Implicit Factoring with Shared Most Significant and Middle Bits. In *Public Key Cryptography*, volume 6056 of *Lecture Notes in Computer Science*, pages 70–87. Springer, 2010.

[FP06a] Jean-Charles Faugère and Ludovic Perret. Cryptanalysis of $2R^-$ Schemes. In *CRYPTO*, pages 357–372, 2006.

[FP06b] Jean-Charles Faugère and Ludovic Perret. Polynomial Equivalence Problems : Algorithmic and Theoretical Aspects. In *EUROCRYPT*, pages 30–47, 2006.

[FR09] Jean-Charles Faugère and Sajjad Rahmany. Solving systems of polynomial equations with symmetries using SAGBI-Gröbner bases. In *ISSAC '09 : Proceedings of the 2009 international symposium on Symbolic and algebraic computation*, ISSAC '09, pages 151–158. ACM, 2009.

[Frö98] Ralf Fröberg. *An Introduction to Gröbner Bases*. John Wiley & Sons, 1998.

[FSEDS10] Jean-Charles Faugère, Mohab Safey El Din, and Pierre-Jean Spaenlehauer. Computing Loci of Rank Defects of Linear Matrices using Grobner Bases and Applications to Cryptology. In *ISSAC '10 : Proceedings of the 2010 international symposium on Symbolic and algebraic computation*, ISSAC '10, pages 257–264, New York, NY, USA, 2010. ACM. Best Student Paper Award.

[Gir05] Christophe Giraud. DFA on AES. In *AES Conference*, volume 3373 of *Lecture Notes in Computer Science*, pages 27–41. Springer, 2005.

[GT10] Christophe Giraud and Adrian Thillard. Piret and Quisquater's DFA on AES Revisited, 2010.

[GTY07] K. Gopalakrishnan, Nicolas Thériault, and Chui Zhi Yao. Solving discrete logarithms from partial knowledge of the key. In *INDOCRYPT'07*, Lecture Notes in Computer Science, pages 224–237. Springer, 2007.

[HGS01] Nick Howgrave-Graham and Nigel P. Smart. Lattice Attacks on Digital Signature Schemes. *Des. Codes Cryptography*, 23(3) :283–290, 2001.

[HP06] Helena Handschuh and Bart Preneel. Blind Differential Cryptanalysis for Enhanced Power Attacks. In *Selected Areas in Cryptography*, pages 163–173, 2006.

[HPS11] Guillaume Hanrot, Xavier Pujol, and Damien Stehlé. Algorithms for the Shortest and Closest Lattice Vector Problems. In *Coding and Cryptology - Third International Workshop (IWCC 2011)*, pages 159–190. Springer, 2011.

[JMV01] Don Johnson, Alfred Menezes, and Scott A. Vanstone. The Elliptic Curve Digital Signature Algorithm (ECDSA). *Intern. J. of Information Security*, 1(1) :36–63, 2001.

[Kan83] Ravi Kannan. Improved Algorithms for Integer Programming and Related Lattice Problems. In *Proceedings of the 15th Annual ACM Symposium on Theory of Computing (STOC 1983)*, pages 193–206. ACM Press, 1983.

[Kho04] Subhash Khot. Hardness of Approximating the Shortest Vector Problem in Lattices. In *Proceedings of the 45th Symposium on Foundations of Computer Science (FOCS 2004)*, pages 126–135. IEEE Computer Society Press, 2004.

[KIM11] Chong Hee KIM. Differential Fault Analysis of AES : Toward Reducing Number of Faults. Cryptology ePrint Archive, Report 2011/178, 2011. `http://eprint.iacr.org/`.

[KJJ99] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential Power Analysis. In *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference*, volume 1666 of *Lecture Notes in Computer Science*, pages 388–397. Springer, 1999.

[Koc96] Paul C. Kocher. Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In *Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference*, Lecture Notes in Computer Science, pages 104–113. Springer, 1996.

[KSP04] Patrick Felke Kai Schramm, Gregor Leander and Christof Paar. A Collision-Attack on AES Combining Side Channel and Differential Attack. In *CHES*, volume 3156 of *Lecture Notes in Computer Science*, pages 163–175. Springer, 2004.

[Laz83] Daniel Lazard. Gröbner-Bases, Gaussian elimination and resolution of systems of algebraic equations. In *Proceedings of Computer Algebra, EUROCAL '83, European Computer Algebra Conference*, volume 162 of *Lecture Notes in Computer Science*, pages 146–156. Springer, 1983.

[Len87] Hendrik W. Lenstra. Factoring Integers with Elliptic Curves. In *The Annals of Mathematics*, volume 126(3), pages 649–673, 1987.

[LL93] Arjen K. Lenstra and Hendrik W. Lenstra. *The Development of the Number Field Sieve*, volume 1554 of *Lecture Notes in Mathematics*. Springer-Verlag, Berlin, 1993.

[LLL82] Arjen Lenstra, Hendrik Lenstra, and László Lovász. Factoring polynomials with rational coefficients. In *Mathematische Annalen*, volume 261, no 4, pages 515–534, 1982.

[LMV04] Hervé Ledig, Frédéric Muller, and Frédéric Valette. Enhancing collision attacks. In *CHES*, volume 3156 of *Lecture Notes in Computer Science*, pages 176–190. Springer, 2004.

[LN96] Rudolf Lidl and Harald Niederreiter. *Finite Fields*. Encyclopedia of Mathematics and Its Applications. Cambridge University Press, 1996.

[LPS04] Peter J. Leadbitter, Dan Page, and Nigel P. Smart. Attacking DSA Under a Repeated Bits Assumption. In *CHES*, volume 3156 of *Lecture Notes in Computer Science*, pages 428–440. Springer, 2004.

[Mac02] Francis S. Macaulay. On some formula in elimination. In *London Mathematical Society*, volume 33, pages 03–27, 1902.

[Mar96] Jacques Martinet. *Les réseaux parfaits des espaces euclidiens*. Mathématiques. Masson, 1996.

[Mat93] Mitsuru Matsui. Linear Cryptoanalysis Method for DES Cipher. In *Advances in Cryptology - EUROCRYPT '93*, volume 765 of *Lecture Notes in Computer Science*, pages 386–397. Springer, 1993.

[MCD+09] Mohamed Saied Emam Mohamed, Daniel Cabarcas, Jintai Ding, Johannes Buchmann, and Stanislav Bulygin. MXL3 : An Efficient Algorithm for Computing Gröbner Bases of Zero-Dimensional Ideals. In *ICISC*, volume 5984 of *Lecture Notes in Computer Science*, pages 87–100. Springer, 2009.

[MG02] Daniele Micciancio and Shafi Goldwasser. *Complexity of lattice problems : a cryptographic perspective*. Kluwer Academic Press, 2002.

[MMDB08] Mohamed Saied Mohamed, Wael Said Mohamed, Jintai Ding, and Johannes Buchmann. MXL2 : Solving Polynomial Equations over GF(2) Using an Improved Mutant Strategy. In *Proceedings of the 2nd International Workshop on Post-Quantum Cryptography*, PQCrypto '08, pages 203–215. Springer-Verlag, 2008.

[MME10] Amir Moradi, Oliver Mischke, and Thomas Eisenbarth. Correlation-Enhanced Power Analysis Collision Attack. In *CHES*, volume 6225 of *Lecture Notes in Computer Science*, pages 125–139. Springer, 2010.

[MNS01] Edwin El Mahassni, Phong Q. Nguyen, and Igor Shparlinski. The Insecurity of Nyberg-Rueppel and Other DSA-Like Signature Schemes with Partially Known Nonces. In *CaLC*, volume 2146 of *Lecture Notes in Computer Science*, pages 97–109. Springer, 2001.

[MOP07] Stefan Mangard, Elisabeth Oswald, and Thomas Popp. *Power Analysis Attacks : Revealing the Secrets of Smart Cards*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2007.

[MR02] S. Murphy and M. J. B. Robshaw. Essential Algebraic Structure within the AES. In *Proceedings of Advances in Cryptology - CRYPTO 2002*, pages 1–16. Springer-Verlag, 2002.

[MR09] Alexander May and Maike Ritzenhofen. Implicit Factoring : On Polynomial Time Factoring Given Only an Implicit Hint. In *Public Key Cryptography*, volume 5443 of *Lecture Notes in Computer Science*, pages 1–14. Springer, 2009.

[MSS10] Subhamoy Maitra, Santanu Sarkar, and Sourav Sengupta. Factoring RSA Modulus Using Prime Reconstruction from Random Known Bits. In *AFRICACRYPT*, volume 6055 of *Lecture Notes in Computer Science*, pages 82–99. Springer, 2010.

[Muk09] Debdeep Mukhopadhyay. An Improved Fault Based Attack of the Advanced Encryption Standard. In *AFRICACRYPT*, volume 5580 of *Lecture Notes in Computer Science*, pages 421–434. Springer, 2009.

[NNTW05] David Naccache, Phong Q. Nguyen, Michael Tunstall, and Claire Whelan. Experimenting with Faults, Lattices and the DSA.

In *Public Key Cryptography*, volume 3386 of *Lecture Notes in Computer Science*, pages 16–28. Springer, 2005.

[NS02] Phong Q. Nguyen and Igor Shparlinski. The Insecurity of the Digital Signature Algorithm with Partially Known Nonces. *Journal of Cryptology*, 15 :151–176, 2002.

[NS03] Phong Q. Nguyen and Igor Shparlinski. The Insecurity of the Elliptic Curve Digital Signature Algorithm with Partially Known Nonces. *Designs, Codes and Cryptography*, 30(2) :201–217, 2003.

[NS05] Phong Q. Nguyen and Damien Stehlé. Floating-Point LLL Revisited. In *EUROCRYPT*, pages 215–233, 2005.

[NS09] Phong Q. Nguyen and Damien Stehlé. An LLL algorithm with Quadratic Complexity. In *SIAM Journal on Computing*, pages 874–903, 2009.

[NV09] Phong Q. Nguyen and Brigitte Vallée. *The LLL Algorithm : Survey and Applications*. Information Security and Cryptography. Springer, 2009.

[Off06] Office of State Commercial Cryptography Administration. The SMS4 Block Cipher (in Chinese), 2006. `http://www.oscca.gov.cn/UpFile/200621016423197990.pdf`.

[OKPW10] Yossef Oren, Mario Kirschbaum, Thomas Popp, and Avishai Wool. Algebraic Side-Channel Analysis in the Presence of Errors. In *CHES'10*, 2010.

[OM07] Elisabeth Oswald and Stefan Mangard. Template Attacks on Masking - Resistance Is Futile. In *Topics in Cryptology - CT-RSA 2007*, volume 4377 of *Lecture Notes in Computer Science*, pages 243–256. Springer, 2007.

[Poh87] Michael Pohst. A Modification of the LLL Reduction Algorithm. In *Journal of Symbolic Computation*, volume 4, pages 123–127, 1987.

[Pol78] John M. Pollard. Monte Carlo methods for index computation (mod p). In *Mathematics of Computation*, volume 32, pages 918–924, 1978.

[Pol00] John M. Pollard. Kangaroos, Monopoly and discrete logarithms. In *Journal of Cryptology*, volume 13, pages 437–447, 2000.

[Pom84] Carl Pomerance. The Quadratic Sieve Factoring Algorithm. In *Proceedings of EUROCRYPT 84 workshop on Advances in cryptology : theory and application of cryptographic techniques*, vo-

lume 209 of *Lecture Notes in Computer Science*, pages 169–182. Springer-Verlag, 1984.

[Pou09] Dimitrios Poulakis. Some Lattices Attacks on DSA and ECDSA. Cryptology ePrint Archive, Report 2009/363, 2009. `http://eprint.iacr.org/`.

[PQ03] Gilles Piret and Jean-Jacques Quisquater. A Differential Fault Attack Technique against SPN Structures, with Application to the AES and KHAZAD. In *CHES*, volume 2779 of *Lecture Notes in Computer Science*, pages 77–88. Springer, 2003.

[Pro05] Emmanuel Prouff. DPA Attacks and S-Boxes. In *FSE*, pages 424–441, 2005.

[RS86] Ronald Linn Rivest and Adi Shamir. Efficient factoring based on partial information. In *Proc. of a workshop on the theory and application of cryptographic techniques on Advances in cryptology—EUROCRYPT '85*, pages 31–34, New York, NY, USA, 1986. Springer-Verlag New York, Inc.

[RS09a] Mathieu Renauld and François-Xavier Standaert. Algebraic Side-Channel Attacks. In *Inscrypt 2009*. LNCS, Springer-Verlag, 2009.

[RS09b] Mathieu Renauld and François-Xavier Standaert. Combining Algebraic and Side-Channel Cryptanalysis against Block Ciphers. In *30-th Symposium on Information Theory in the Benelux*, 5 2009.

[RS10] Mathieu Renauld and François-Xavier Standaert. Representation-, Leakage- and Cipher- Dependencies in Algebraic Side-Channel Attacks. In *Industrial Track of ACNS 2010*, pages 1–18, 2010.

[RSA78] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Commun. ACM*, 21(2) :120–126, 1978.

[RSN+10] Andrew Rukhin, Juan Soto, James Nechvatal, Miles Smid, Elaine Barker, Stefan Leigh, Mark Levenson, Mark Vangel, David Banks, Alan Heckert, James Dray, and San Vo. A Statistical Test Suite of Random and Pseudorandom Number Generators for Cryptographic Applications. Tech. rep., National Institute of Standards and Technology (NIST), Special Publication 800-22 Revision 1a, 2010. `http://csrc.nist.gov/groups/ST/toolkit/rng/documentation_software.html`.

[RSVC09] Mathieu Renauld, François-Xavier Standaert, and Nicolas Veyrat-Charvillon. Algebraic Side-Channel Attacks on the AES : Why Time also Matters in DPA. In *CHES'09*, pages 97–111, Berlin, Heidelberg, 2009. Springer-Verlag.

[RT09a] Thomas Roche and Cédric Tavernier. Multi-Linear cryptanalysis in Power Analysis Attacks MLPA. *CoRR*, abs/0906.0237, 2009.

[RT09b] Thomas Roche and Cédric Tavernier. Side-channel attacks based on linear approximations. *IACR Cryptology ePrint Archive*, 2009 :269, 2009.

[Rva09] Matthieu Rvain. *On the Physical Security of Cryptographic Implementations*. PhD thesis, University of Luxembourg, September 2009.

[Sch90] Claus-Peter Schnorr. Efficient Identification and Signatures for Smart Cards. In *Proceedings of the 9th Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '89, pages 239–252. Springer-Verlag, 1990.

[SGJB11] Guo Shize, Deng Gaoming, Cao Jin, and Chen Bing. Enhanced Side Channel Template Attack with Correlation Analysis. *Instrumentation, Measurement, Computer, Communication and Control, International Conference on*, pages 504–507, 2011.

[Sha71] Daniel Shanks. Class number, a theory of factorization, and genera. In *Proceedings of Symposia in Pure Mathematics*, volume 20, pages 415–440. American Mathematical Society, Providence, 1971.

[Sho97] Victor Shoup. Lower Bounds for Discrete Logarithms and Related Problems. In *EUROCRYPT*, volume 1233, pages 256–266. Springer, 1997.

[Sin99] Simon Singh. *The code book : the science of secrecy from ancient Egypt to quantum cryptography*. Fourth Estate, 1999.

[Sko05] Sergei P. Skorobogatov. *Semi-invasive attacks - A new approach to hardware security analysis*. PhD thesis, University of Cambridge, 2005.

[SLP05] Werner Schindler, Kerstin Lemke, and Christof Paar. A Stochastic Model for Differential Side Channel Cryptanalysis. In *Cryptographic Hardware and Embedded Systems — CHES 2005*, volume 3659 of *Lecture Notes in Computer Science*, pages 30–46. Springer, 2005.

[SM09] Santanu Sarkar and Subhamoy Maitra. Further Results on Implicit Factoring in Polynomial Time. *Advances in Mathematics of Communications*, 3(2) :205–217, 2009.

[Sma98] Nigel Smart. *The Algorithmic Resolution of Diophantine Equations*, volume 41 of *London Mathematical Society Student Text*. Cambridge University Press, 1998.

[SNC09] Mate Soos, Karsten Nohl, and Claude Castelluccia. Extending SAT Solvers to Cryptographic Problems. In *SAT*, pages 244–257, 2009.

[Ste05] Damien Stehlé. *Algorithmique de la réduction de réseaux et application à la recherche de pires cas pour l'arrondi de fonctions mathématiques*. PhD thesis, Université Henri Poincaré - Nancy 1, 2005.

[SWP03] Kai Schramm, Thomas Wollinger, and Christof Paar. A new class of collision attacks and its application to DES. In *Fast Software Encryption - FSE '03, volume LNCS 2887 of LNCS*, pages 206–222. Springer-Verlag, 2003.

[Tak06a] Katsuyuki Takashima. Practical Application of Lattice Basis Reduction Algorithm to Side-Channel Analysis on (EC)DSA. *IEICE Transactions*, 89-A(5) :1255–1262, 2006.

[Tak06b] Katsuyuki Takashima. Practical modifications of Leadbitter et al.'s repeated-bits side-channel analysis on (EC)DSA. In *Proceedings of the 6th international conference on Information Security Applications*, WISA'05, pages 259–270. Springer-Verlag, 2006.

[Tes01] Edlyn Teske. Square-Root Algorithms For The Discrete Logarithm Problem (a Survey). In *Public Key Cryptography and Computational Number Theory, Walter de Gruyter*, pages 283–301, 2001.

[TM09] Michael Tunstall and Debdeep Mukhopadhyay. Differential Fault Analysis of the Advanced Encryption Standard using a Single Fault. Cryptology ePrint Archive, Report 2009/575, 2009. `http://eprint.iacr.org/`.

[Van92] Scott A. Vanstone. Responses to NIST's Proposal. *Communications of the ACM*, 35 :50–52, July 1992. (communicated by John Anderson).

[X9.05] American National Standard X9.62-2005. *The Elliptic Curve Digital Signature Algorithm (ECDSA)*. Public Key Cryptogra-

phy for the Financial Services Industry, Accredited Standards Committee X9, 2005.

[ZZG+12] Xinjie Zhao, Fan Zhang, Shize Guo, Tao Wang, Zhijie Shi, Huiying Liu, and Keke Ji. MDASCA : An Enhanced Algebraic Side-Channel Attack for Error Tolerance and New Leakage Model Exploitation. In Werner Schindler and Sorin Huss, editors, *Constructive Side-Channel Analysis and Secure Design*, volume 7275 of *Lecture Notes in Computer Science*, pages 231–248. Springer Berlin / Heidelberg, 2012.

Résumé La cryptanalyse algébrique consiste à modéliser une primitive cryptographique par un système d'équations polynomiales dont la résolution permet de retrouver la clef secrète. L'objectif de cette thèse est d'évaluer comment une information extérieure permet d'en accélérer significativement la résolution. Nous supposons que l'information extérieure est obtenue par canal auxiliaire, c'est-à-dire par des mesures physiques, ou bien suite à un comportement anormal provoqué par des attaques actives du type injection de fautes, ou bien encore à cause de la présence d'un logiciel malveillant.

Appliqués à la cryptographie asymétrique, ces travaux ont conduit à la publication d'une nouvelle attaque contre les schémas de signature de type DSA. Inspiré par la factorisation implicite de May et Ritzenhofen, cette attaque suppose que les clefs éphémères utilisées pour construire les signatures de plusieurs messages donnés partagent un certain nombre de bits en commun dont les valeurs sont inconnues.

En ce qui concerne les chiffrements par blocs, nous présentons une étude théorique des "Algebraic Side Channel Attacks" (ASCA) qui explique l'efficacité de ces attaques et qui permet de proposer des conditions théoriques de résistance. Nous utilisons principalement des techniques de résolution par base de Gröbner plutôt que par solveur SAT quand cela est possible. Nous montrons ainsi que la complexité d'une résolution par base de Gröbner dépend d'une nouvelle notion d'immunité algébrique et de la distribution des informations de fuite. Enfin, nous étendons les ASCA en considérant différents modèles de fuite et étudions l'influence de ces modèles sur l'efficacité de l'étape de résolution.

Abstract Algebraic cryptanalysis is a technique consisting in modeling a cryptographic primitive with a system of multivariate polynomial equations such that the secret key belongs to the set of solutions of this system. The goal of this thesis is to evaluate how external information can significantly speed up the resolution. We assume that external information is obtained by side channel, *i.e.* by physical measures, or by an anomalous behavior caused by active attacks with fault injection or even by the presence of malware.

Applied to asymmetric cryptography, this work led to the publication of a new attack against DSA-like signature schemes. Inspired by implicit factorization of May and Ritzenhofen, this new attack requires that the ephemeral keys used to sign several messages share a given number of bits without necessarily knowing the value of the shared bits.

On the other hand, with regard to the block ciphers, we present a theoretical study of "Algebraic Side Channel Attacks" (ASCA) in order to explain the effectiveness of the algebraic phase of these attacks and then we deduce theoretical conditions for resistance. When it is possible, we mainly use Gröbner basis method for the solving step rather than SAT solver. Then we show that the complexity of the Gröbner basis computation in these attacks depends on a new notion of algebraic immunity and the distribution of information leakage. Finally, we extend the ASCA by considering various leakage models and by studying the influence of these models on the effectiveness of the solving step.