Algebraic Cryptanalysis of Block Ciphers Using Gröbner Bases

Vom Fachbereich Informatik
der Technischen Universität Darmstadt
genehmigte

Dissertation

zur Erlangung des Grades
Doktor rerum naturalium (Dr. rer. nat.)
von

Dipl.-Math. Andrey Pyshkin

aus Murmansk

Referenten:	Prof. Dr. Johannes Buchmann
	Prof. Dr. Jintai Ding
Tag der Einreichung:	25.02.2008
Tag der mündlichen Prüfung:	16.04.2008

Darmstadt, 2008
D17

Erklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit – mit Ausnahme der in ihr ausdrücklich genannten Hilfen – selbständig verfasst habe.

Wissenschaftlicher Werdegang des Verfassers

09/1997 - 06/2000	Studium der Mathematik und der Informatik an der Murmansker Staatlichen Pädagogischen Universität (Murmansk, Russland)
09/2000 - 06/2003	Studium der Mathematik an der Kaliningrader Staatlichen Universität (Kaliningrad, Russland), erfolgreich abgeschlossen (Diplom mit auszeichnung)
09/2003 - 03/2004	Wissenschaftlicher Mitarbeiter an der Kaliningrader Staatlichen Universität (Kaliningrad, Russland)
04/2004 - 03/2007	Stipendium der Marga und Kurt Möllgaard-Stiftung
04/2004 - dato	Doktorand an der Technischen Universität Darmstadt (Darmstadt, Deutschland)
04/2007 - dato	Wissenschaftlicher Mitarbeiter an der Technischen Universität Darmstadt (Darmstadt, Deutschland)

Publications

Andrey Bogdanov, Andrey Pyshkin. *Algebraic Side-Channel Collision Attacks on AES.* Cryptology ePrint Archive, Report 2007/477, 2007, http://eprint.iacr.org/;

Johannes Buchmann, Andrei Pyshkin, Ralf-Philipp Weinmann. *A Zero-Dimensional Groebner Basis for AES-128.* FSE 2006, LNCS 4047, pp. 78-88, Springer-Verlag;

Johannes Buchmann, Andrei Pyshkin, Ralf-Philipp Weinmann. *Block Ciphers Sensitive to Gröbner Basis Attacks.* CT-RSA 2006, LNCS 3860, pp. 313-331, Springer-Verlag;

Ulrich Kühn, Andrei Pyshkin, Erik Tews, Ralf-Philipp Weinmann. *Variants of Bleichenbacher's Low-Exponent Attack on PKCS#1 RSA Signatures.* accepted for SICHERHEIT 2008, GI-Verlag;

Fen Liu, Wen Ji, Lei Hu, Jintai Ding, Shuwang Lv, Andrei Pyshkin, Ralf-Philipp Weinmann. *Analysis of the SMS4 block cipher.* ACISP 2007, LNCS 4586, pp. 158-170, Springer-Verlag;

Erik Tews, Ralf-Philipp Weinmann, Andrei Pyshkin: *Breaking 104 bit WEP in less than 60 seconds.* WISA 2007, LNCS 4867, pp. 188-202, Springer-Verlag.

Acknowledgements

I am grateful to Prof. Dr. Johannes Buchmann for giving me the opportunity to join his research group, for promoting this thesis, and for his steady support. I want to thank Prof. Dr. Jintai Ding for accepting the task of the second referee.

I thank Ralf-Philipp Weinmann and Andrey Bogdanov for useful discussions, the continuous exchange of new ideas, and the productive collaboration.

I would like to thank my colleagues in the research group "Cryptography and Computer Algebra" at the Darmstadt University of Technology for such friendly environment in which is extremely nice to work and research. I also want to thank my parents, my wife Elena, my sister, and Maxim Anikeev, Mikhail Bogachev, Alexander Elokhov, Julia Mashkovich, and Dina Yarullina for their support and encouragement.

I want to thank the Marga und Kurt Möllgaard-Stiftung for the financial support provided me. I thank also Dr. Sergej I. Aleshnikov and Prof. Dr. Peter Roquette for their help and constant attention.

Abstract

This thesis investigates the application of Gröbner bases to cryptanalysis of block ciphers. The basic for the application is an algorithm for solving systems of polynomial equations via Gröbner basis computation. In our case, polynomial equations describe the key recovery problem for block ciphers, i.e., the solution of these systems corresponds to the value of the secret key.

First we demonstrate that Gröbner basis technique can be successfully used to break block ciphers, if the algebraic structure of these ciphers is relatively simple. To show this, we construct two families of block ciphers that satisfy this condition. However, our ciphers are not trivial, they have a reasonable block and key size as well as an acceptable number of rounds. Moreover, using suitable parameters we achieve good resistance of these ciphers against differential and linear cryptanalysis. At the same time, we design our ciphers so that the key recovery problem for each of them can be described by a system of simple polynomial equations. In addition, parameters of the ciphers can be varied independently. This makes the constructed families suitable for analysis of algebraic attacks. To study the vulnerable of such ciphers against Gröbner basis attack, we have performed experiments using the computer algebra system Magma. Results of these experiments are given and analyzed. Also, for a subset of these ciphers we present an efficient method to construct zero-dimensional Gröbner bases w.r.t. a degree-reverse lexicographical term order without a polynomial reduction. This reduces the key recovery problem to the problem of Gröbner basis conversion. Using known complexity bounds for the last problem, we estimate the maximum resistance of these ciphers against Gröbner basis attacks.

We show that our method can be also applied to the AES block cipher. In the thesis we describe the AES key recovery problem in the form of a total-degree Gröbner basis, explain how this Gröbner basis can be obtained, and study the cryptanalytic significance of this result.

Next, we investigate the semi-regularity of several polynomial representations for iterated block ciphers. We demonstrate that the constructed Gröbner basis for the AES is semi-regular. Then we prove that polynomial systems that are similar to the BES quadratic equations are not semi-regular as well as the AES systems of quadratic equations over GF(2) are not semi-regular over GF(2).

Finally, we propose a new method of side-channel cryptanalysis - algebraic collision attacks - and explain it by the example of the AES. The method is based on the standard power analysis technique, which is applied to derive an additional information from an implementation of a cryptosystem. In our case, this information is about generalized internal collisions occurring

between S-boxes of the block cipher. However, we use a new approach to recover the secret key from the obtained information. Taking into account a specific structure of the attacked cryptographic algorithm, we express the detected collisions as a system of polynomial equations and use Gröbner bases to solve this system. This approach provides significant advantages both in terms of measurements and post-processing complexity. Also, we use non-collisions to optimize our method. For the AES block cipher, we demonstrate several efficient algebraic collision attacks. The first of them works in the known-plaintext scenario and requires 5 measurements to derive the full secret key within several hours on a PC with success probability 0.93. This attack with 4 measurements recovers key in about 40% of all cases. The second attack works in the known plaintext/ciphertext pair scenario but leads to more efficient results: the key can be obtained in several seconds of offline computations with success probability of 0.82 for 4 measurements, and with probability close to 1 for 5 measurements. We also propose a successful algebraic collision attack on the AES with 3 measurements. The attack has a probability of 0.42 and needs 4.24 PC hours post-processing.

Zusammenfassung

In der vorliegenden Arbeit untersuchen wir die Anwendbarkeit von Gröbner Basen zur Kryptoanalyse von Blockchiffren. Eine der wichtigsten Anwendungen von Gröbner Basen ist der Lösung von Polynomial-Gleichungssystemen. Viele Kryptoverfahren lassen sich als Gleichungssysteme beschreiben, nicht alle von solchen Gleichungssystemen sind aber effizient lösbar.

Um zu untersuchen, wie gut Gröbner Basis-Angriffe auf Blockchiffre funktionieren und wie das von Parametern (die Größe des Blockes, die Anzahl der Runden, sowie der Grad der Polynome) der Chiffre abhängt, haben wir die zwei Familien der populärsten Klassen von modernen Blockchiffren, Feistel und SP Netzwerke, konstruiert und analysiert. Wir zeigen, dass es nicht triviale Blockchiffre gibt, die resistent gegen die linearen und differentiellen Angriffe sind, aber sich algebraisch angreifen lassen. Außerdem ist beschrieben, wie Gröbner Basen bezüglich die graduiert-lexikographische Termordnung für eine großen Untermenge dieser Blockchiffre effektiv berechnen werden können, d.h., algebraische Angriffe auf diese Chiffre werden auf das Problem, eine graduiert-lexikographische Gröbner Basis in die lexikographischen Termordnung umzurechnen, zurückgeführt. Durch bekannten Abschätzungen der Komplexität des letzten Problems schätzen wir die Effizient von Gröbner Basis-Angriffe in diesem Fall ab.

Die vorschlagene Methode lässt sich auch auf das AES-Verschlüsselungsverfahren anwenden. In der Dissertation erklären wir, wie eine graduiert-lexikographishe Gröbner Basis für den AES bekommen werden kann, sowie die Auswirkung dieser Gröbner Basis auf die Sicherheit des Verfahrens.

Dann betrachten wir die Semi-Regularität von verschiedenen Gleichungssystemen, die iterierte Blockchiffren beschreiben. Für reguläre und semi-reguläre Mengen von Polynomen sind Abschätzungen über die Komplexität der Gröbner Basis-Algorithmen bekannt. Man weiß auch, dass die Polynome, die ein Kryptosystem beschreiben, fast nie regulär sind. Es wurde aber vermutet, dass diese Polynome semi-regulär sind. Wir beweisen, dass diese Vermutung für iterierte Blockchiffren meistens falsch ist, u.a., quadratische Gleichungssysteme für den AES sind weder semi-regulär, noch semi-regulär über GF(2).

Schließlich demonstrieren wir, dass Seitenkanalangriffe sich durch Gröbner Basis-Methoden verbessern lassen. Unsere Methode basiert auf Kollisionen, die zwischen verschiedenen S-Boxen bei Verschlüsselung einiger Klartexte auftreten. Es ist bekannt, dass man solche Kollisionen mittels Differential Power Analysis nachweisen kann, falls die Implementierung des Verfahrens nicht gegen Seitenkanalangriffe abgesichert ist. Um den Schlüssel aus den festgestellte Kollisionen zu ziehen, beschreiben wir sie als Polynomial-

Gleichungssysteme. Wir zeigen, dass einige Klassen von diesen Systemen effektiv durch Gröbner Basen lösbar sind. Außerdem werden Nicht-Kollisionen zur Verbesserung der Methode benutzt. In der Dissertation werden drei Varianten dieser Angriffe auf den AES präsentiert. Die Erste verwendet Kollisionen zwischen S-boxen der ersten zwei Runden und braucht 5 oder 4 gemessene Klartexte, um den AES-Schlüssel mit Wahrscheinlichkeit 0.93 bzw. 0.4 zu ziehen. Falls Ein- und Ausgaben des Verfahrens dem Angreifer bekannt sind, kann man die S-boxen der ersten und letzten Runden betrachten. Algebraische Angriffe, die auf Kollisionen zwischen diesen S-boxen basieren, haben eine bessere Laufzeit sowie eine höhere Erfolgswahrscheinlichkeit. Wenn beide Varianten kombiniert werden, ist man in der Lage, den AES-Schlüssel mit 3 gemessenen Klartexten mit der Wahrscheinlichkeit 0.42 zu finden.

Contents

List of Tables

Chapter 1

Introduction

The concept of Gröbner bases for polynomial rings was introduced by Bruno Buchberger in 1965 [13] (see also [14, 15]). Since that time algorithmic solutions based on Gröbner bases were developed for some important problems of commutative algebra and algebraic geometry such as the ideal membership, radical and decomposition of ideal, conversion of parametric representation as well as solving systems of polynomial equations, etc. [5]. Hence it can be expected that Gröbner bases find a wide application area, including theoretical mechanics, biology, chemistry, sudoku solver, robotics, engineering design, statistics, coding theory, and cryptography. However, sometimes there is a significant difference (and also some time delay) between the possibility of application and practical application itself. Furthermore, some applications have additional requirements or specific conditions. So in cryptography, many cryptoschemes can be described by systems of polynomial equations, with secret parameters as variables. From the mathematical point of view, any of these system can be solved using Gröbner bases, this means that the corresponding algorithm provides with the set of all possible solutions after a finite number of iterations. But in cryptography only methods that give solutions faster than an exhaustive search are interesting. The complexity bounds given in [5, 18, 37, 38] do not allow to claim that the Gröbner basis computation is such method. On the other hand, the HFE public key cryptosystem was broken using Gröbner bases [36]. Further, Gröbner basis attacks on stream ciphers were proposed in [34], and in [52] an application of Gröbner bases to cryptanalysis of the SHA-1 hash function was discussed. Here it will be shown that Gröbner basis algorithms also cannot be ignored in the case of block ciphers.

In this thesis we study several aspects of algebraic cryptanalysis of block ciphers. First we analyze Gröbner basis attacks in a primary wording: for an examined block cipher and a known plaintext/ciphertext pair, one describes

the encryption process by a system of polynomial equations and tries to recover the secret key by solving the obtained system. For that purpose we have constructed two families of block ciphers. In contrast to [20], where small scale variants of the AES are considered to investigate algebraic attacks, our ciphers have a reasonable block and key size, 128 bits. Moreover, applying the wide trail strategy [29] to design the ciphers we make them resistant against differential and linear cryptanalysis. We show that some of these ciphers however can be broken by Gröbner basis attacks. This statement is based on experimental results as well as theoretical estimations. We propose an efficient method to obtain Gröbner bases w.r.t. a degree-reverse lexicographical term order for a subset of our block ciphers without a polynomial reduction. This reduces the key recovery problem for these ciphers to a Gröbner basis conversion. Using an upper bound for the complexity of the FGLM algorithm [35], we estimate the complexity of Gröbner basis attacks. These results are given in Chapter 4 of this thesis, they have been also published in [17].

The presented method (possibly in a slightly modified form) can be applied not only to academic ciphers. In this thesis, Chapter 5, this is illustrated with AES-128. We show how to produce a total-degree Gröbner basis for this cipher using a polynomial representation of the S-box. Important characteristics of this basis are found, and using them we explain why the application of the constructed Gröbner basis is difficult for cryptanalysis. Note that at the beginning of 2008, the existence of this Gröbner basis has no security implications for AES. The result have been published in [16].

Further, we verify the conjecture that polynomial systems for block ciphers are semi-regular. The concept of semi-regular sequences of polynomials was introduced by Bardet, Faugère, and Salvy [3], [2], [4]. Note that semi-regular sequences are defined both for homogeneous and non-homogeneous polynomials. The following has motived us to consider the semi-regularity of block cipher. First, by conjecture, evolved from computer experiments with random sequences, most sequences are semi-regular. Secondly, for the case of semi-regular sequences, the bounds for the complexity of the F_5 Gröbner basis algorithm and the XL algorithm were found ([3], [2], [4], and [1]). Combining these two statements, the authors of [3] have given complexity bound for algebraic attacks on block ciphers under the assumption that polynomial systems describing these ciphers are semi-regular. In Chapter 6 we check this assumption for several algebraic representations of iterated block ciphers. One case is similar to BES equations. Recall that Big Encryption System (BES), proposed by Murphy and Robshaw, is an embedding for the AES and can be expressed as a system of quadratic equations over $\mathrm{GF}(2^8)$ [47]. Using BES one can easily obtain quadratic equations over $\mathrm{GF}(2^8)$ for AES.

Such approach can be applied not only to AES, for example, [44] describes an embedding for SMS4, a block cipher that is used in the WAPI standard for protecting data packets in wireless networks. We show that systems derived this way are not semi-regular. Then we analyze systems of polynomial equations over GF(2) such as AES equations given in [24] or quadratic equations obtained for the block cipher KHAZAD, MISTY1, KASUMI, CAMELLIA, and SERPENT in [9]. We prove that such systems are not semi-regular over GF(2). Thus in two major cases the conjecture about semi-regularity of polynomial equations for block ciphers is wrong, and hence the estimation obtained for the complexity of the F_5 Gröbner basis algorithm and the XL algorithm must be used carefully. However, semi-regular sequences occur in cryptography: we show that the above Gröbner basis for the AES is semi-regular. But in this case we get a Gröbner basis directly.

Finally we demonstrate that side-channel collision attacks can be improved using Gröbner basis techniques. We propose a new cryptanalytic method called algebraic (side-channel) collision attacks and apply it to AES. As was shown in [50, 10] and [8], some AES implementations on 8-bit processors can be vulnerable against side-channel collision attacks. The main idea of these attacks is that by comparing the power consumption curves corresponding to different S-box operations one can detect whether the inputs to these S-boxes are equal. This works as follows. An attacker inputs one or more (possibly chosen) random plaintexts to an AES module. For each plaintext, the attacker measures and stores the power consumption curves for the time periods, where appointed S-boxes are executed. Then one looks for collisions in some S-boxes comparing the corresponding power curves. Here various methods can be applied: square differences, cross-correlation, wavelet analysis, etc. The basic attack introduced in [50] uses only internal collisions between S-boxes in the second round of different AES runs with chosen plaintexts; and S-boxes at different byte positions are not compared. However, if all instances of the AES S-box share the same implementation, for example, if the S-box is implemented as a separate routine, then one can detect collisions between any two input bytes to the S-box. Such collisions are called generalized internal collisions and they ware first applied to attack AES in [10]. After a necessary number of collisions are detected, one tries to derive the secret key from this information. In [50] internal collisions are employed to sift key candidates, while [10] uses linear algebra methods. The basic idea of our method is that a set of generalized internal collision corresponds to a system of non-linear equations. We show that some types of such systems can be quickly solved by Gröbner basis computation. Actually we found three efficient algebraic collision attacks. The first attack are based on collisions in the first two AES rounds and works in the known-plaintext

scenario. In the second attack, the system consists of nonlinear equations corresponding to all collisions within the first round, within the last round as well as between the first and last rounds; here the plaintext/ciphertext pairs must be known. We also combine these two approaches in the third attack. In addition, we demonstrate how non-collisions, i.e., S-box pairs that do not collide, can be used to optimize these attacks. The algebraic techniques allows one to mount collision attacks for 3 measurements with a probability of 0.42 and 4.24 PC hours post-processing, for 4 measurements with a probability of 0.82 in several seconds of offline computations as well as for 5 measurements with success probability close to 1 and several seconds post-processing. This is to be compared to 40 measurements with some non-negligible post-processing in [50] for a success probability > 0.5 and 6 measurements with approx. $2^{37.15}$ offline computations and a success probability of 0.85 or 5 measurements with $2^{45.5}$ offline computations and a probability of 0.55 in [10]. We describe algebraic collision attacks in Chapter 7, and a paper on this subject is [11].

Besides, the thesis have two chapters with preliminaries: Chapter 2 lists standard results on Gröbner basis and semi-regular sequences, and Chapter 3 describes several algebraic representations of the AES block cipher.

Chapter 2

Algebraic Background

This chapter provides a brief overview of Gröbner bases and semi-regular sequences. Good references giving a comprehensive introduction to Gröbner bases theory are [5] and [25]. For more details on semi-regular sequences we refer the reader to [4] (see also [2]). Statements that are given in this chapter without proof and explicit references can be found (possibly slightly modified) in the above works.

Let $R = \mathbb{F}[\mathcal{X}] = \mathbb{F}[x_1, \ldots, x_n]$ be a polynomial ring in n variables over a field $\mathbb{F}$. A power product of variables is called a *term*. By $\mathcal{T}$ denote the set of all terms in R. Then $\mathcal{T}_d \subset \mathcal{T}$ is the set of all terms of degree d. Here the degree of the term $t = x_1^{d_1} \ldots x_n^{d_n}$ is $\deg(t) = \sum_{i=1}^{n} d_i$. The product of a term and an element $c \in \mathbb{F}$ is called a *monomial*.[1] Let $f = \sum c(a_1, \ldots, a_n) x_1^{a_1} \ldots x_n^{a_n} \in R$ be a non-zero polynomial. By definition, put

$$T(f) = \{x_1^{a_1} \ldots x_n^{a_n} \in \mathcal{T} \colon c(a_1, \ldots, a_n) \neq 0\},$$
$$M(f) = \{c(a_1, \ldots, a_n) x_1^{a_1} \ldots x_n^{a_n} \colon c(a_1, \ldots, a_n) \neq 0\},$$

and $T_d(f) = T(f) \cap \mathcal{T}_d$. The *degree* of f, denoted by $\deg(f)$, is the maximal d such that $T_d(f) \neq \emptyset$. We say that f is homogeneous if $T(f) \subset \mathcal{T}_{\deg(f)}$. It is clear that every polynomial $f \in R \setminus \{0\}$ has a unique representation in the form $\sum_{i=1}^{m} f_i$, where $f_1, \ldots, f_m$ are non-zero homogeneous polynomials such that $\deg(f) = \deg(f_1) > \cdots > \deg(f_m)$. The homogeneous part f_1 of highest degree is called the *degree form* of f and is denoted by $DF(f)$.

Next, for any set of polynomials $S \subset R$ we define $T(S)$, $T_d(S)$, $DF(S)$,

[1] Let us note that here we adopt the conventions of [5]. For example the authors of [25] call a product of variables a monomial and refer to the product of variables and a coefficient as a term.

and $DF_d(S)$ as follows:

$$T(S) = \bigcup_{f \in S} T(f); \qquad T_d(S) = T(S) \cap \mathcal{T}_d;$$
$$DF(S) = \{DF(f) \colon f \in S\}; \qquad DF_d(S) = DF(S) \cap \mathcal{T}_d;$$

We use $\langle S \rangle$ to denote the ideal generated by all $f \in S$. A ideal $\mathfrak{I} \subset R$ is *zero-dimensional* if $\mathfrak{I} \cap \mathbb{F}[x_i] \neq \{0\}$ for all $1 \leq i \leq n$. In this case, we write $\dim \mathfrak{I} = 0$. It can be shown that $\dim \mathfrak{I} = 0$ iff the $\mathbb{F}$-vector space $R/\mathfrak{I}$ is finite-dimensional.

2.1 Term Orders

Definition 2.1.1. A *term order* $\preceq$ is a linear order on the set of terms $\mathcal{T}$ such that for all $t, t_1, t_2 \in \mathcal{T}$ the following conditions hold:

1. $1 = x_1^0 \dots x_n^0 \preceq t$;
2. if $t_1 \prec t_2$, then $t_1 t \prec t_2 t$.

If additionally $t_1 \prec t_2$ whenever $\deg(t_1) < \deg(t_2)$, then $\preceq$ is called a *total degree term order*. In this thesis the following term orders are used.

Lexicographical Term Order: By definition, $x_1^{d_1} \dots x_n^{d_n} \prec_{lex} x_1^{e_1} \dots x_n^{e_n}$ iff there exists some i with $1 \leq i \leq n$ such that $d_i < e_i$ and $d_j = e_j$ for all $1 \leq j \leq i-1$. Note that $\preceq_{lex}$ is not a total degree term order, since in this case

$$1 \prec_{lex} x_n \prec_{lex} x_n^2 \prec_{lex} \cdots \prec_{lex} x_{n-1} \prec_{lex} x_{n-1}^2 \prec_{lex} \dots$$

Degree Lexicographical Term Order: For all $t_1, t_2 \in \mathcal{T}$, define $t_1 \prec_{dlex} t_2$ iff either $\deg(t_1) < \deg(t_2)$ or if $\deg(t_1) = \deg(t_2)$ and $t_1 \prec_{lex} t_2$.

Degree Reverse Lexicographical Term Order: For any $t_1 = \prod_{i=1}^n x_i^{d_i} \in \mathcal{T}$ and $t_2 = \prod_{i=1}^n x_i^{e_i} \in \mathcal{T}$ we define $t_1 \prec_{DRL} t_2$ iff either $\deg(t_1) < \deg(t_2)$ or if $\deg(t_1) = \deg(t_2)$ and there exists some i with $1 \leq i \leq n$ such that $d_i > e_i$ and $d_j = e_j$ for all $i+1 \leq j \leq n$. Clearly, the degree lexicographical and degree reverse lexicographical term orders are examples of total degree term orders.

Let a term order $\preceq$ be fixed. For any two monomials at_1 and bt_2 with $t_1, t_2 \in \mathcal{T}$ and non-zero coefficients $a, b \in \mathbb{F}$, set

$$at_1 \preceq bt_2 \qquad \text{iff} \qquad t_1 \preceq t_2.$$

The maximal element of $T(f)$ w.r.t. $\preceq$ is called the *head term* of f and is denoted by $\mathrm{HT}(f)$. Likewise, $\mathrm{HM}(f) = \max_{\preceq}(M(f))$ is called the *head monomial* of f, and its coefficient, denoted by $\mathrm{HC}(f)$, is the *head coefficient* of f. Clearly, $\mathrm{HM}(f) = \mathrm{HC}(f) \cdot \mathrm{HT}(f)$. Also, for any $S \subset R$ put

$$\mathrm{HT}(S) = \{\mathrm{HT}(f) \colon f \in S\}.$$

2.2 Gröbner bases

Let $\preceq$ be a term order on $\mathcal{T}$. Let $G = \{g_1, \ldots, g_m\} \subset R$ be a set of polynomials. A polynomial $f \in R$ is called *reducible* modulo G, if there exists a term $t \in T(f)$ that is divisible by some head term of G. Algorithm 1 called *polynomial reduction* describes the generalized division of f by G for the multivariate case.

Algorithm 1 Polynomial Reduction

Require: A set $G = \{g_1, \ldots, g_m\} \subset R$ and $f \in R$

1: Set $h := f$
2: **while** h is reducible modulo G **do**
3: Select a monomial $mon \in M(h)$ such that $mon = a \cdot t$ with $a \in \mathbb{F}$ and $t = t_1 \cdot \mathrm{HT}(g_i) \in \mathcal{T}$ for some $1 \leq i \leq m$ and $t_1 \in \mathcal{T}$
4: Set $h := h - c \cdot t_1 \cdot g_i$, where $c = a/\mathrm{HC}(g_i)$
5: **end while**
6: Return h

The resulting polynomial h is called a *normal form* of f w.r.t. G and denoted by $\mathrm{NF}(f, G)$. We see that h is not reducible modulo G and there are $f_1, \ldots, f_m \in R$ such that

$$f - \sum_{i=1}^{m} f_i g_i = h$$

and $\mathrm{HT}(f_i g_i) \preceq \mathrm{HT}(f)$ for all $1 \leq i \leq m$. Since it is possible that by step 3 of the reduction algorithm some terms are divisible by several head terms of G, the result of the polynomial reduction is in general not uniquely defined. However any $f \in R$ has a unique normal form w.r.t. G whenever G is a Gröbner basis.

Definition 2.2.1. Let $\mathfrak{I} \subset R$ be an ideal. A finite set of polynomials $G \subset \mathfrak{I}$ is called a *Gröbner basis* of $\mathfrak{I}$ (w.r.t. $\preceq$) if $\langle \mathrm{HT}(G) \rangle = \langle \mathrm{HT}(\mathfrak{I}) \rangle$.

We will say that a set of polynomial $G = \{g_1, \ldots, g_m\}$ is a Gröbner basis if G is a Gröbner basis of the ideal $\langle G \rangle$. Also, we refer to Gröbner bases w.r.t. the lexicographical term order (the degree reverse lexicographical term order) as Lex (DRL) Gröbner bases.

For any ideal $\mathfrak{I} \subset R$ there exists a Gröbner basis. It can be derived from any finite set of generators using, for example, the Buchberger algorithm [5], the Faugère F_4 [31] or F_5 [32] algorithms. These algorithms are based on the following theorem.

Theorem 2.2.2. *Let $G \subset R$ be a finite set of polynomials. Then G is a Gröbner basis iff $NF(spol(g_i, g_j), G) = 0$ for any $g_i, g_j \in G$, where the polynomial $spol(g_i, g_j)$ called the S-polynomial of g_i and g_j is given by*

$$spol(g_i, g_j) = \frac{lcm(HT(g_i), HT(g_j))}{HM(g_i)} \cdot g_i - \frac{lcm(HT(g_i), HT(g_j))}{HM(g_j)} \cdot g_j.$$

The basic version of the Buchberger algorithm works as follows.

Algorithm 2 Gröbner basis Algorithm

Require: A set $G = \{g_1, \ldots, g_m\} \subset R$
1: Put $CP := \{(g_i, g_j) \colon \text{ for all } 1 \leq i < j \leq m\}$
2: **while** $CP \neq \emptyset$ **do**
3: Select $(f, g) \in CP$
4: $CP := CP \setminus \{(f, g)\}$
5: **if** $\mathrm{NF}(spol(f, g), G) \neq 0$ **then**
6: Put $CP := CP \cup \{(g, h) \colon \text{ for all } g \in G\}$ and $G := G \cup \{h\}$, where $h = \mathrm{NF}(spol(f, g), G)$
7: **end if**
8: **end while**
9: Return G

In this algorithm the elements of CP are called *critical pairs*. One of the main ideas of improved versions of Algorithm 2 is that for some critical pairs it is known without computing normal forms whether their S-polynomials are reduced to 0. The first statement of the following theorem is the first Buchberger criterion.

Theorem 2.2.3. *Let $G \subset R$ be a finite set of polynomials. If the head term of some $f, g \in G$ are coprime, i.e., $gcd(HT(f), HT(g)) = 1$, then $NF(spol(f, g), G) = 0$. In particular, if all elements of the set $HT(G)$ are pairwise coprime, then G is a Gröbner basis.*

Note that there are also other criteria that can be used to reduce the number of critical pairs.

The next way to speed up the Gröbner bases computation is to derive normal forms for several critical pairs simultaneously by using linear algebra techniques. This method is applied in the Faugère F_4 algorithm [31].

Furthermore, computing a Gröbner basis w.r.t. a total-degree order usually is faster than computing a lexicographical Gröbner basis for the same ideal. By this reason the following strategy often is used, if it is necessary to obtain a Lex Gröbner basis. First a DRL Gröbner basis is computed, then it is transformed to a Lex Gröbner basisby applied some order change algorithm. The FGLM algorithm [35] and the Gröbner Walk [22] as well as various variations of them are the most popular algorithms for performing Gröbner basis conversions. Note that the FGLM algorithm as described in [35] only works for zero-dimensional ideals, while the Gröbner Walk does not have such restriction.

The time complexity of the FGLM algorithm is estimated in the following theorem.

Theorem 2.2.4 (Theorem 5.1 of [35])**.** *Let $G_1 \subset R$ be a Gröbner basis w.r.t. a term order $\preceq_1$ of a zero-dimensional polynomial ideal $\mathfrak{I}$, and $D = dim(R/\mathfrak{I})$. Then by the FGLM algorithm we can convert G_1 into a Gröbner basis G_2 w.r.t. a term order $\preceq_2$ in $O(nD^3)$ field operations.*

For the space complexity of the algorithm, no bound is given in the original paper. We note that the dominant memory requirement of the FGLM algorithm is a $D \times nD$ matrix over $\mathbb{F}$. Thus the memory usage of the algorithm is upper bounded by $\lceil (nD^2 \log_2(\mathbb{F}))/8 \rceil + o(1)$ bytes.

We see that the complexity of the FGLM algorithm depends on the dimension of the $\mathbb{F}$-vector space $R/\mathfrak{I}$. This dimension can be computed as follows.

Theorem 2.2.5. *Let G be a Gröbner basis of the ideal $\mathfrak{I} \subset R$. Then*

$$\begin{aligned} \dim(R/\mathfrak{I}) &= \#\left\{t \in \mathcal{T} \colon HT(f) \nmid t \text{ for all } f \in \mathfrak{I}\right\} \\ &= \#\left\{t \in \mathcal{T} \colon HT(g) \nmid t \text{ for all } g \in G\right\}. \end{aligned}$$

For the Gröbner Walk, the running time strongly depends on the source and the target term order. No usable tight bounds on its time nor its space complexity are currently known to the author of this thesis.

2.3 Two Applications of Gröbner bases

Using Gröbner bases one can solve various problems in a polynomial ring, for example, the ideal membership and the equality of ideals, intersection of ideals and multivariate interpolation, radical and primary decomposition of ideal as well as invertibility of polynomial maps and solving polynomial systems. Now we describe two such applications of Gröbner bases, deciding the ideal membership problem and solving systems of polynomial equations.

2.3.1 Deciding the Ideal Membership Problem

Let $G = \{g_1, \ldots, g_n\} \subset R$ be a finite set of polynomials and $f \in R$. The ideal membership problem is to decide if $f \in \langle G \rangle$. The following theorem shows how Gröbner bases can be used to solve this problem.

Theorem 2.3.1. *Let $G \subset R$ be a Gröbner basis. Then any $f \in R$ has a unique normal form w.r.t. G. Moreover, $f \in \langle G \rangle$ iff $NF(f, G) = 0$.*

Thus, to determine whether a polynomial f lies in an ideal $\mathfrak{I} \subset R$, it is sufficient to reduce f modulo some Gröbner basis of $\mathfrak{I}$. In chapter 5 we construct a DRL Gröbner basis for the block cipher AES and study its application for testing key bytes.

2.3.2 Solving Systems of Polynomial Equations

The main application of Gröbner bases considered in this thesis is solving systems of multivariate polynomial equations. In the following chapters we describe the key recovery problem for several scenarios of attacks on block ciphers as polynomial systems over finite fields and show that some of these systems can be solved efficiently using Gröbner bases. Here a general algorithm for solving polynomial system is given.

Let $\mathbb{S} = \{p_1 = 0, \ldots, p_m = 0 \colon p_i \in R \text{ for all } 1 \leq i \leq m\}$ be a system of polynomial equations in n variables. By

$$\overline{\mathcal{V}}_{\mathbb{S}} = \{(a_1, \ldots, a_n) \in \overline{\mathbb{F}}^n \colon p_1(a_1, \ldots, a_n) = \cdots = p_m(a_1, \ldots, a_n) = 0\}$$

denote the set of all solution of the system $\mathbb{S}$ in the closure of $\mathbb{F}$. It can be proved that $\overline{\mathcal{V}}_{\mathbb{S}}$ is finite iff the ideal $\langle p_1, \ldots, p_m \rangle$ is zero-dimensional. Let G be a Gröbner basis w.r.t. an arbitrary term order of $\langle p_1, \ldots, p_m \rangle$. For short, we will say that G is a Gröbner basis of $\mathbb{S}$. There is a useful criterion for polynomial ideals to be zero-dimensional.

Theorem 2.3.2. *Let G be a Gröbner basis of an ideal $\mathfrak{I}$. Then* $\dim \mathfrak{I} = 0$ *iff for any* $i = \overline{1,n}$ *there is a polynomial* $g \in G$ *such that* $HT(g) = x_i^{d_i}$.

Thus any Gröbner basis of a polynomial system shows whether the number of zeroes of this system is finite. A Gröbner basis G is called *reduced* if no $g \in G$ is reducible modulo $G \setminus \{g\}$ and the head coefficient of each polynomial of G equals 1. A reduced Gröbner basis can be derived from any Gröbner basis $G = \{g_1, \ldots, g_m\}$ by the following procedure. Let $G_0 = G$. For all $1 \leq i \leq m$, first put $G_i := G_{i-1} \setminus \{g_i\}$, and if $\mathrm{HT}(g_i) \notin \mathrm{HT}(G_i)$, then

$$G_i := G_i \cup \{\mathrm{HC}(g_i)^{-1} \cdot \mathrm{NF}(g_i, G_i)\}.$$

Then G_m is a reduced Gröbner basis of $\langle G \rangle$. Note that for any ideal there exists a unique reduced Gröbner basis. The following theorem describe the relation between $\overline{\mathcal{V}}_{\mathbb{S}}$ and the reduced Gröbner basis of $\mathbb{S}$.

Theorem 2.3.3. *Let $\mathbb{S}$ be a system of polynomial equations. Then*

1. $\overline{\mathcal{V}}_{\mathbb{S}} = \emptyset$ *iff* $G = \{1\}$, *where G is the reduced Gröbner basis of $\mathbb{S}$ w.r.t. an arbitrary term order.*

2. $\overline{\mathcal{V}}_{\mathbb{S}} = \{(a_1, \ldots, a_n)\}$ *for some* $a_i \in \overline{\mathbb{F}}$ *iff* $G = \{x_1 + a_1, \ldots, x_n + a_n\}$, *where G is the reduced Gröbner basis of $\mathbb{S}$ w.r.t. an arbitrary term order.*

3. $\overline{\mathcal{V}}_{\mathbb{S}}$ *is finite iff for any* $i = \overline{1,n}$ *there exists a polynomial* $g_i \in \mathbb{F}[x_1, \ldots, x_i]$ *such that* $x_i^{d_i} + g_i \in G$ *with some* $d_i \geq 1$, *where G is the reduced Lex Gröbner basis of $\mathbb{S}$ and* $x_i^{d_i} \succ_{lex} HT(g_i)$.

As in the case of linear equations, some variables are (algebraic) independent w.r.t. $\mathbb{S}$, if the set $\overline{\mathcal{V}}_{\mathbb{S}}$ is infinite. However we show below that all systems considered in this thesis have finite number of solutions, and hence we can assume that $\mathbb{S}$ is so. From Theorem 2.3.3, we get an algorithm for solving systems of polynomial equations using Gröbner bases (Algorithm 3). The set $\overline{\mathcal{V}}_{\mathbb{S}}$ consists of all zeroes of the system $\mathbb{S}$ in the algebraic closure of $\mathbb{F}$. Clearly, the set $\mathcal{V}_{\mathbb{S}}$ of the solutions of $\mathbb{S}$ in $\mathbb{F}$ can be obtained, if in Algorithm 3 only the $\mathbb{F}$-zeroes of univariate polynomials are selected. For any finite field $\mathbb{F}$ however there exists also an other way to derive $\mathcal{V}_{\mathbb{S}} = \overline{\mathcal{V}}_{\mathbb{S}} \cap \mathbb{F}^n$. Let $\mathbb{F}$ have q elements. Then for any $\alpha \in \overline{\mathbb{F}}$, we have $\alpha \in \mathbb{F}$ iff the relation $\alpha^q = \alpha$ holds. The equations

$$x_i^q + x_i = 0$$

are called *field equations*. Thus the set of the zeroes of the system

$$\mathbb{S}' = \mathbb{S} \cup \{x_i^q + x_i = 0 \colon 1 \leq i \leq n\}$$

Algorithm 3 Solving Systems of Polynomial Equations

Require: A system $\mathbb{S} = \{p_1 = \cdots = p_m = 0 \colon p_i \in R \text{ for all } 1 \leq i \leq m\}$

1: Compute the reduced DRL Gröbner basis G_{DRL} of the ideal $\langle p_1, \ldots, p_m \rangle$.
2: **if** $G_{DRL} = \{1\}$ **then**
3: Return $\overline{\mathcal{V}}_{\mathbb{S}} = \emptyset$
4: **else if** $G_{DRL} = \{x_1 + a_1, \ldots, x_n + a_n\}$ **then**
5: Return $\overline{\mathcal{V}}_{\mathbb{S}} = \{(a_1, \ldots, a_n)\}$
6: **else**
7: Derive the reduced Lex Gröbner basis G_{lex} form G_{DRL} using a Gröbner basis conversion algorithm.
8: Compute $\overline{\mathcal{V}}_{\mathbb{S}}$ as follows. First compute all zeros Z_1 of the univariate polynomial $x_1^{d_1} + g_1 \in G_{lex} \cap \mathbb{F}[x_1]$. Then substitute each $z \in Z_1$ into each element of G_{lex}, and compute all zeros $Z_{2,z}$ of the resulting univariate polynomials in x_2, and so on.
9: Return $\overline{\mathcal{V}}_{\mathbb{S}}$
10: **end if**

is $\overline{\mathcal{V}}_{\mathbb{S}'} = \overline{\mathcal{V}}_{\mathbb{S}} \cap \mathbb{F}^n = \mathcal{V}_{\mathbb{S}}$ and can be found by Algorithm 3. Note that this strategy is good for finite fields with a relative small number of elements. Moreover, for the case of GF(2) with the field equations there exist several improved implementation of Gröbner basis computation algorithms PolyBoRi [12], magma. If a finite field has a large number of elements, it is usually more efficient to compute a Gröbner basis of $\mathbb{S}$ first, and then to select the zeroes of $\mathbb{S}$ in this field.

2.4 Semi-Regular Sequences

First we define the degree of regularity for polynomial ideals.

Let $\preceq$ be a total degree term order. In this case we have $\mathrm{HT}(f) = \mathrm{HT}(DF(f))$ for any $f \in R$, where $DF(f)$ is the degree form of a polynomial f. By definition, put

$$E(S) = \mathcal{T} \setminus \mathrm{HT}(S) \quad \text{and} \quad E_d(S) = E(S) \cap \mathcal{T}_d$$

for all set of polynomials $S \subset R$. Let $\mathfrak{I}$ be an ideal.

The *Hilber function* of $\mathfrak{I}$ is the map $H(\mathfrak{I}, \cdot) \colon \mathbb{Z}_{\geq 0} \to \mathbb{Z}_{\geq 0}$ such that

$$H(\mathfrak{I}, d) = \#E_d(\mathfrak{I}) = \#\mathcal{T}_d - \#\mathrm{HT}_d(\mathfrak{I})$$

for all $d \geq 0$.

Further, the power series

$$\sum_{d\geq 0} H(\mathfrak{I}, d)z^d$$

is called the *Hilbert series* of $\mathfrak{I}$.

From Theorem 2.3.2, it follows easily that $\dim \mathfrak{I} = 0$ iff there is a number d_0 such that $H(\mathfrak{I}, d_0) = 0$. The *degree of regularity* for a zero-dimensional ideal $\mathfrak{I}$ is defined by

$$D_{reg}(\mathfrak{I}) = \min\{d \geq 0 \colon H(\mathfrak{I}, d) = 0\}$$

It is clear that if $\dim \mathfrak{I} = 0$, then $H(\mathfrak{I}, d) = 0$ for all $d \geq D_{reg}(\mathfrak{I})$, and hence the Hilbert series is a polynomial.

If $\dim \mathfrak{I} \neq 0$, then we define the degree of regularity of $\mathfrak{I}$ to be ∞. We agree to the convention that $n < \infty$, for all $n \in \mathbb{Z}$.

2.4.1 General Case

To define semi-regular sequence, we describe trivial relations for a set of polynomials. A semi-regular sequence has no other relations up to degree of regularity. Moreover, the Gröbner basis algorithm F_5 does not generate useless critical pairs that are obtained from trivial relations [32],[2].

Let $P = \{f_1, \ldots, f_m\}$ be a set of polynomials. Before computing a Gröbner basis from P, the algorithm F_5 constructs Gröbner bases for the ideals $\langle f_1\rangle$, $\langle f_1, f_2\rangle$, $\ldots$, $\langle f_1, \ldots, f_{m-1}\rangle$. Like other Gröbner basis algorithms, it creates new polynomials using two basic operations, the multiplication by a monomial, and the reduction by a set of polynomials. Clearly, computations whose result is the zero polynomial are unnecessary. In some cases, it is known that the result of operations is 0. One such example is given in Theorem 2.2.3. Further, if $g \in \langle f_1, \ldots, f_{i-1}\rangle$ for $1 \leq i \leq m$, then $gf_i \in \langle f_1, \ldots, f_{i-1}\rangle$. In particular, $f_i f_j = f_j f_i$ for all i, j.

Definition 2.4.1. Suppose $(f_1, \ldots, f_m)$ is a sequence of homogeneous polynomials, and for any $i = 1, \ldots, m$ there is no polynomial $g \notin \langle f_1, \ldots, f_{i-1}\rangle$ such that $gf_i \in \langle f_1, \ldots, f_{i-1}\rangle$ and $\deg(gf_i) < D_{reg}(\mathfrak{I})$, where $\mathfrak{I} = \langle f_1, \ldots, f_m\rangle$. Then this sequence is called *semi-regular*.

A sequence of non-homogeneous polynomial $(f_1, \ldots, f_m)$ is semi-regular if the sequence of their degree forms $(DF(f_1), \ldots, DF(f_m))$ is semi-regular.

If for a sequence $(f_1, \ldots, f_m)$ and some $1 \leq i \leq m$ there exists a polynomial $g \notin \langle f_1, \ldots, f_{i-1}\rangle$ such that $gf_i \in \langle f_1, \ldots, f_{i-1}\rangle$ and $\deg(gf_i) = d <$

$D_{reg}(\mathfrak{I})$, then we will say that the sequence has a non-trivial relation of degree d.

For any power series $P = \sum_{i=0}^{\infty} \alpha_i z^i$ put $[P] = \sum_{i=0}^{D-1} \alpha_i z^i$, where

$$D = \min\{d \colon \alpha_d \leq 0\}$$

In [4] the authors give the following criterion for a sequence of polynomials to be semi-regular.

Theorem 2.4.2. *A sequence of homogeneous polynomials $(f_1, \ldots, f_m)$ is semi-regular iff the Hilbert series of the ideal $\mathfrak{I} = \langle f_1, \ldots, f_m \rangle$ equals*

$$\left[\frac{\prod_{i=1}^{m} (1 - z^{d_i})}{(1-z)^n} \right],$$

where $d_i = \deg f_i$.

Since the Hilbert series is dependent only on the ideal, we have:

Corollary 2.4.3. *Let π be a permutation of $\{1, \ldots, m\}$. If $(f_1, \ldots, f_m)$ is semi-regular, then $\left(f_{\pi(1)}, \ldots, f_{\pi(m)}\right)$ is also semi-regular.*

Thus we can say that a set of polynomial is semi-regular or not. We will say also that the system of equations is semi-regular, if the set of polynomials of this system is semi-regular.

Corollary 2.4.4. *Let $F = (f_1, \ldots, f_m)$ be a sequence of homogeneous polynomials.*

1. *If $m \leq n$ then F is semi-regular iff F is regular.*
2. *If $m \geq n$ and F is semi-regular, then* $\dim(\langle f_1, \ldots, f_m \rangle) = 0$.

2.4.2 Semi-Regular Sequence over GF(2)

Let $R_2 = \mathrm{GF}(2)[x_1, \ldots, x_n]$ and $\{f_1, \ldots, f_m\} \subset R_2$. As it was discussed in Subsection 2.3.2, sometimes it is necessary to compute a Gröbner basis of the ideal

$$\langle f_1, \ldots, f_m, x_1^2 + x_1, \ldots, x_n^2 + x_n \rangle.$$

In this case, the terms of all polynomials except $x_i^2 + x_i$ $(1 \leq i \leq n)$ are square-free. Furthermore, there are new trivial relations $f_j^2 = f_j$, where $1 \leq j \leq m$. Thus the definition of semi-regular sequence over GF(2) is slightly different from the general case [3]. Let us also remark that in this case the algorithm F_5 can be improved. We refer the reader to [3], [2] for more detail.

Definition 2.4.5. Suppose $(f_1, \ldots, f_m)$ is a sequence of homogeneous polynomials in R_2, and the terms of all polynomials are square-free. If for any $i = 1, \ldots, m$ there is no polynomial $g \notin \langle x_1^2, \ldots, x_n^2, f_1, \ldots, f_{i-1}, f_i \rangle$ such that

$$gf_i \in \langle x_1^2, \ldots, x_n^2, f_1, \ldots, f_{i-1} \rangle \text{ and } \deg(gf_i) < D_{reg}(\langle x_1^2, \ldots, x_n^2, f_1, \ldots, f_m \rangle),$$

then the sequence is called semi-regular over GF(2).

A sequence of non-homogeneous polynomial $(f_1, \ldots, f_m)$ is semi-regular over GF(2) if so is the sequence of their degree forms $(DF(f_1), \ldots, DF(f_m))$.

The degree $D_{reg}(\langle x_1^2, \ldots, x_n^2, f_1, \ldots, f_m \rangle)$ is called the degree of regularity of $(f_1, \ldots, f_m)$ over GF(2).

The criterion for a sequence of polynomials to be semi-regular over GF(2) is the following.

Theorem 2.4.6. *A sequence of homogeneous polynomials $(f_1, \ldots, f_m)$ is semi-regular over GF(2) iff the Hilbert series of $\langle x_1^2, \ldots, x_n^2, f_1, \ldots, f_m \rangle$ equals*

$$\left[\frac{(1+z)^n}{\prod_{i=1}^{m} (1 + z^{d_i})} \right],$$

where $d_i = \deg f_i$.

Corollary 2.4.7. *Let π be a permutation. If $(f_1, \ldots, f_m)$ is semi-regular over GF(2), then so is $(f_{\pi(1)}, \ldots, f_{\pi(m)})$.*

Chapter 3

AES

The *Advanced Encryption Standard* (AES) [48] is a block cipher with a relatively simple algebraic structure. It can be elegantly described using only operations over the finite field GF(2^8). In addition, two representations in the form of systems of polynomial quadratic equations in several variables have been proposed for AES [24, 47, 21]. In this chapter we briefly describe AES and show how the polynomial systems for it can be obtained. Good references for the design of Rijndael, the cipher that is used in AES, are [28] and [27].

3.1 Description of AES

According to [48], the length of each input and output data blocks for AES is equal to 128 bits, the key length may be 128, 192, or 256 bits. AES-n is the standard designation for AES with n-bit key ($n = 128, 192,$ and 256). Like most modern block cipher, AES is an iterated block cipher, i.e., its encryption consists of several rounds. An input of the first round is a *plaintext*, an input of any other round, *internal state*, is an output of the previous one. An output of the last round, *final state*, is a *ciphertext*. In each round a state is transformed using a *round function*. The round function of AES depends on a 128-bit *round key*, which is derived from a cipher key using the AES *key schedule*. Let $\mathcal{R}$ be the number of rounds, then

$$\mathcal{R} = \begin{cases} 10, & \text{for AES-128;} \\ 12, & \text{for AES-192;} \\ 14, & \text{for AES-256.} \end{cases}$$

Let us consider the AES round function. It consists of four state transformations: `SubBytes`, `ShiftRows`, `MixColumns`, and `AddRoundKey`. Note

that in the first and last round the AES round function is slightly modified. In order to make inputs of the first round dependent on the key, an initial round key is added to a plaintext. The last round has no `MixColumns` transformation. This makes the processes of the AES encryption and decryption more similar. The AES encryption is given by Algorithm 4. There are sev-

Algorithm 4 AES encryption

Require: plaintext P, round keys $(K_0, K_1, \ldots, K_{\mathcal{R}})$
Ensure: ciphertext C
1: $\Sigma_0 = \texttt{AddRoundKey}(P, K_0)$
2: **for** $i = 1, \ldots, \mathcal{R} - 1$ **do**
3: $\quad tmp = \texttt{SubBytes}(\Sigma_{i-1})$
4: $\quad tmp = \texttt{ShiftRows}(tmp)$
5: $\quad tmp = \texttt{MixColumns}(tmp)$
6: $\quad \Sigma_i = \texttt{AddRoundKey}(tmp, K_i)$
7: **end for**
8: $tmp = \texttt{SubBytes}(\Sigma_{\mathcal{R}-1})$
9: $tmp = \texttt{ShiftRows}(tmp)$
10: $C = \texttt{AddRoundKey}(tmp, K_{\mathcal{R}})$

eral ways to describe the transformations of the AES round function, for example using tables or bit operations. We describe them using finite field operations.

State Representation

Each AES state block consists of 16 bytes. Any byte $b = \mathtt{b_7 b_6 \ldots b_1 b_0} = \sum_{i=0}^{7} \mathtt{b_i} \cdot 2^i$ can be obviously represented as the element $\beta = \sum_{i=0}^{7} \mathtt{b_i} \cdot \xi^i$ of the finite field $\mathrm{GF}(2^8) = \mathrm{GF}(2)[\xi]$, where $\xi^8 + \xi^4 + \xi^3 + \xi + 1 = 0$ holds. We will use also the inverse representation to write a finite field element shortly.

Example 3.1.1.

$$\mathtt{01} = 0000\ 0001 \leftrightarrow 1 \in \mathbb{F}; \qquad \mathtt{63} = 0110\ 0011 \leftrightarrow \xi^6 + \xi^5 + \xi + 1 \in \mathbb{F};$$
$$\mathtt{02} = 0000\ 0010 \leftrightarrow \xi \in \mathbb{F}; \qquad \mathtt{AC} = 1010\ 1100 \leftrightarrow \xi^7 + \xi^5 + \xi^3 + \xi^2 \in \mathbb{F}.$$

Thus we see that any state block Σ can be interpreted as the square matrix over $\mathrm{GF}(2^8)$:

$$\begin{pmatrix} \sigma_0 & \sigma_4 & \sigma_8 & \sigma_{12} \\ \sigma_1 & \sigma_5 & \sigma_9 & \sigma_{13} \\ \sigma_2 & \sigma_6 & \sigma_{10} & \sigma_{14} \\ \sigma_3 & \sigma_7 & \sigma_{11} & \sigma_{15} \end{pmatrix}.$$

`SubBytes`

The first transformation of the AES round function is `SubBytes`. In this step, a bijective function $S : \mathbb{F} \to \mathbb{F}$, called *S-box*, is applied to each element of a state independently.

$$\begin{pmatrix} S(\sigma_0) & S(\sigma_4) & S(\sigma_8) & S(\sigma_{12}) \\ S(\sigma_1) & S(\sigma_5) & S(\sigma_9) & S(\sigma_{13}) \\ S(\sigma_2) & S(\sigma_6) & S(\sigma_{10}) & S(\sigma_{14}) \\ S(\sigma_3) & S(\sigma_7) & S(\sigma_{11}) & S(\sigma_{15}) \end{pmatrix}$$

The AES S-box is the composition of the multiplicative inverse f_{-1} in GF(2^8), a GF(2)-linear mapping L, and the addition of the element `63`. By definition,

$$f_{-1}(\theta) = \theta^{254} = \begin{cases} \theta^{-1}, & \text{if } \theta \in \mathbb{F} \setminus \{0\}; \\ 0, & \text{if } \theta = 0. \end{cases}$$

The polynomial representation of L over GF(2^8) is given by

$$\mathtt{8F}x^{2^7} + \mathtt{B5}x^{2^6} + \mathtt{01}x^{2^5} + \mathtt{F4}x^{2^4} + \mathtt{25}x^{2^3} + \mathtt{F9}x^{2^2} + \mathtt{09}x^2 + \mathtt{05}x.$$

Denote this polynomial by f_L. Like other components of the AES round function, the GF(2)-linear mapping L is invertible, and its inverse L' can be expressed as the following polynomial over GF(2^8):

$$f_{L'}(x) = \mathtt{6E}x^{2^7} + \mathtt{DB}x^{2^6} + \mathtt{59}x^{2^5} + \mathtt{78}x^{2^4} + \mathtt{5A}x^{2^3} + \mathtt{7F}x^{2^2} + \mathtt{FE}x^2 + \mathtt{05}x.$$

Since $S(\sigma) = f_L(f_{-1}(\sigma)) + \mathtt{63}$, we obtain the following polynomial representation of the AES S-box:

$$\mathtt{05}x^{254} + \mathtt{09}x^{253} + \mathtt{F9}x^{251} + \mathtt{25}x^{247} + \mathtt{F4}x^{239} + \mathtt{01}x^{223} + \mathtt{B5}x^{191} + \mathtt{8F}x^{127} + \mathtt{63}. \quad (3.1)$$

We see that this polynomial is sparse. The polynomial over GF(2^8) corresponding to the inverse S-box, $S'(\sigma) = f_{-1}(f_{L'}(\sigma + \mathtt{63}))$, is denser and given in Appendix A.

`ShiftRows`

The next transformation is `ShiftRows`. The ith row of the state matrix is cyclically shifted over i elements to left ($0 \le i \le 3$).

$$\begin{pmatrix} \sigma_0 & \sigma_4 & \sigma_8 & \sigma_{12} \\ \sigma_1 & \sigma_5 & \sigma_9 & \sigma_{13} \\ \sigma_2 & \sigma_6 & \sigma_{10} & \sigma_{14} \\ \sigma_3 & \sigma_7 & \sigma_{11} & \sigma_{15} \end{pmatrix} \mapsto \begin{pmatrix} \sigma_0 & \sigma_4 & \sigma_8 & \sigma_{12} \\ \sigma_5 & \sigma_9 & \sigma_{13} & \sigma_1 \\ \sigma_{10} & \sigma_{14} & \sigma_2 & \sigma_6 \\ \sigma_{15} & \sigma_3 & \sigma_7 & \sigma_{11} \end{pmatrix}.$$

`MixColumns`

To make each output element of state dependent on several input elements, a matrix multiplication is used. This step is called `MixColumns`. Let

$$D = \begin{pmatrix} \texttt{02} & \texttt{03} & \texttt{01} & \texttt{01} \\ \texttt{01} & \texttt{02} & \texttt{03} & \texttt{01} \\ \texttt{01} & \texttt{01} & \texttt{02} & \texttt{03} \\ \texttt{03} & \texttt{01} & \texttt{01} & \texttt{02} \end{pmatrix}, \tag{3.2}$$

and Σ be an input state of `MixColumns`, then the output state is $\Sigma' = D\Sigma$. For D, the inverse matrix is

$$D^{-1} = \begin{pmatrix} \texttt{0E} & \texttt{0B} & \texttt{0D} & \texttt{09} \\ \texttt{09} & \texttt{0E} & \texttt{0B} & \texttt{0D} \\ \texttt{0D} & \texttt{09} & \texttt{0E} & \texttt{0B} \\ \texttt{0B} & \texttt{0D} & \texttt{09} & \texttt{0E} \end{pmatrix}.$$

`AddRoundKey`

Unlike the first three transformations, `AddRoundKey` depends on a secret parameter, a round key. The round key K has the length of 128 bits, and is interpreted as the 4×4-matrix over $\mathrm{GF}(2^8)$. Then this matrix is added to the state matrix Σ.

$$\Sigma + K = \begin{pmatrix} \sigma_0 + k_0 & \sigma_4 + k_4 & \sigma_8 + k_8 & \sigma_{12} + k_{12} \\ \sigma_1 + k_1 & \sigma_5 + k_5 & \sigma_9 + k_9 & \sigma_{13} + k_{13} \\ \sigma_2 + k_2 & \sigma_6 + k_6 & \sigma_{10} + k_{10} & \sigma_{14} + k_{14} \\ \sigma_3 + k_3 & \sigma_7 + k_7 & \sigma_{11} + k_{11} & \sigma_{15} + k_{15} \end{pmatrix}.$$

In order to derive round keys, the AES key schedule is applied to a cipher key. This routine can be described also using operations over $\mathrm{GF}(2^8)$.

Key Schedule

Since there are slight differences in the key schedules for AES-128, AES-192, and AES-256, we consider these variants separately.

The 1th case: AES-128.

In the case of AES-128, a cipher key, K, is 16 bytes long and can be interpreted as the 4×4-matrix over $\mathrm{GF}(2^8)$. The rounds keys $K_0, K_1, \ldots, K_{10}$ are generated from K as follows. First, the initial round key K_0 is defined by $K_0 = K$. For $1 \leq i \leq 10$, the ith round key K_i is derived from the previous

one using the following formulas. Let the jth element of K_i be denoted by $k_{i,j}$ with $0 \le j \le 15$. Then

$$k_{i,0} = S(k_{i-1,13}) + k_{i-1,0} + \xi^{i-1}, \qquad k_{i,1} = S(k_{i-1,14}) + k_{i-1,1},$$
$$k_{i,2} = S(k_{i-1,15}) + k_{i-1,2}, \qquad k_{i,3} = S(k_{i-1,12}) + k_{i-1,3},$$

and $k_{i,j} = k_{i,j-4} + k_{i-1,j}$ for $4 \le j \le 15$.

The 2th case: AES-192.

In this case the length of a cipher key, K, is 24 bytes. First an *expanded key* is derived from the cipher key. The expanded key consists of 9 blocks $EK_0, \ldots, EK_8$, and each block has 24 elements of the finite field $\mathrm{GF}(2^8)$. For $i \ge 1$ the ith block $EK_i = (ek_{i,0}, \ldots, ek_{i,23})$ is generated from EK_{i-1}, where $EK_0 = K$. The generation formulas are similar to the formulas in the AES-128 case:

$$ek_{i,0} = S(ek_{i-1,21}) + ek_{i-1,0} + \xi^{i-1}, \qquad ek_{i,1} = S(ek_{i-1,22}) + ek_{i-1,1},$$
$$ek_{i,2} = S(ek_{i-1,23}) + ek_{i-1,2}, \qquad ek_{i,3} = S(ek_{i-1,20}) + ek_{i-1,3},$$

and $ek_{i,j} = ek_{i,j-4} + ek_{i-1,j}$ for $4 \le j \le 23$.

After then the round keys $K_0, \ldots, K_{12}$ are selected from the expanded key as follows. For $0 \le i \le 3$, we have

$$\begin{aligned} K_{3i} &= (ek_{2i,0}, \ldots, ek_{2i,15}), \\ K_{3i+1} &= (ek_{2i,16}, \ldots, ek_{2i,23}, ek_{2i+1,0}, \ldots, ek_{2i+1,7}), \\ K_{3i+2} &= (ek_{2i+1,8}, \ldots, ek_{2i+1,23}); \end{aligned}$$

and $K_{12} = (ek_{8,0}, \ldots, ek_{8,15})$. The elements $ek_{8,16}, \ldots, ek_{8,23}$ of EK_8 are not used.

The 3th case: AES-256.

Like the AES-192 case, the key schedule for AES-256 consists of the key expansion and the round key selection. The expanded key has 8 blocks $EK_0, \ldots, EK_7 \in \mathrm{GF}(2^8)^{32}$. As above, $EK_0 = K$. For $i \ge 1$ the elements $ek_{i,0}, \ldots, ek_{i,31}$ of the ith block EK_i is derived in the following way:

for $j = 0, 1, 2, 3$,

$$ek_{i,0} = S(ek_{i-1,21}) + ek_{i-1,0} + \xi^{i-1}, \qquad ek_{i,1} = S(ek_{i-1,22}) + ek_{i-1,1},$$
$$ek_{i,2} = S(ek_{i-1,23}) + ek_{i-1,2}, \qquad ek_{i,3} = S(ek_{i-1,20}) + ek_{i-1,3};$$

for $4 \le j \le 15$,

$$ek_{i,j} = ek_{i,j-4} + ek_{i-1,j};$$

for $j = 16, 17, 18, 19$,

$$ek_{i,j} = S(ek_{i,j-4}) + ek_{i-1,j};$$

and for $20 \leq j \leq 31$,

$$ek_{i,j} = ek_{i,j-4} + ek_{i-1,j}.$$

Then the round keys $K_0, \ldots, K_{14}$ are taken from the expanded key as follows. For $0 \leq i \leq 6$, we have

$$\begin{aligned} K_{2i} &= (ek_{i,0}, \ldots, ek_{i,15}), \\ K_{2i+1} &= (ek_{i,16}, \ldots, ek_{i,31}); \end{aligned}$$

and $K_{14} = (ek_{7,0}, \ldots, ek_{7,15})$. Note that since the elements $ek_{7,16}, \ldots, ek_{7,31}$ of EK_7 are not used, they are not computed.

3.2 Algebraic Representations of the AES Key Recovery Problem

Let m AES plaintext/ciphertext pairs be known, where $m \geq 1$ for AES-128, and $m \geq 2$ for AES-192, and AES-256. Then using the AES description given above it is easy to express the AES key recovery problem as a system of polynomial equations over GF(2^8). However in this case the output of the S-box is given by the polynomial S in the input, and hence the non-linear equations of the obtained system have degree 254. For the AES block cipher, there exist also several algebraic representations in the form of multivariate polynomial systems of quadratic equations over GF(2) [24] as well as over GF(2^8) [47]. Here we briefly describe these three AES expressions.

3.2.1 System of Equations over GF(2^8)

Let $((p_0, \ldots, p_{15}), (c_0, \ldots, c_{15})) \in \mathbb{F}^{16} \times \mathbb{F}^{16}$ be a known plaintext/ciphertext pair. Denote by $x_{i,j}$ the variable referring to the jth element of the state after the ith `AddRoundKey`, and by $k_{i,j}$ the variable referring to the jth element of the ith round key for $0 \leq i \leq 10$, and $0 \leq j \leq 15$. The system of equations over GF(2^8) in $x_{i,j}$, and $k_{i,j}$ consists of the four following parts:

1. plaintext/ciphertext equations

$$\begin{aligned} x_{0,0} + k_{0,0} + p_0 &= 0 & \qquad x_{10,0} + c_0 &= 0 \\ &\vdots & &\vdots \\ x_{0,15} + k_{0,15} + p_{15} &= 0 & \qquad x_{10,15} + c_{15} &= 0 \end{aligned}$$

2. equations, which correspond to the ith round of the AES encryption with $1 \leq i \leq \mathcal{R} - 1$:

$$\begin{pmatrix} x_{i,0}+k_{i,0} & x_{i,4}+k_{i,4} & x_{i,8}+k_{i,8} & x_{i,12}+k_{i,12} \\ x_{i,1}+k_{i,1} & x_{i,5}+k_{i,5} & x_{i,9}+k_{i,9} & x_{i,13}+k_{i,13} \\ x_{i,2}+k_{i,2} & x_{i,6}+k_{i,6} & x_{i,10}+k_{i,10} & x_{i,14}+k_{i,14} \\ x_{i,3}+k_{i,3} & x_{i,7}+k_{i,7} & x_{i,11}+k_{i,11} & x_{i,15}+k_{i,15} \end{pmatrix} + \\ +D \cdot \begin{pmatrix} S(x_{i-1,0}) & S(x_{i-1,4}) & S(x_{i-1,8}) & S(x_{i-1,12}) \\ S(x_{i-1,5}) & S(x_{i-1,9}) & S(x_{i-1,13}) & S(x_{i-1,1}) \\ S(x_{i-1,10}) & S(x_{i-1,14}) & S(x_{i-1,2}) & S(x_{i-1,6}) \\ S(x_{i-1,15}) & S(x_{i-1,3}) & S(x_{i-1,7}) & S(x_{i-1,11}) \end{pmatrix} = 0, \tag{3.3}$$

where the polynomial function S is given by (3.1), and the matrix D is given by (3.2). For example, for

$$x_{i,0} = k_{i,0} + \mathtt{02}S(x_{i-1,0}) + \mathtt{03}S(x_{i-1,5}) + \mathtt{01}S(x_{i-1,10}) + \mathtt{01}S(x_{i-1,15}),$$

we have

$$\begin{array}{ccccc} \mathtt{0A}x_{i-1,0}^{254}+\mathtt{0F}x_{i-1,5}^{254}+ & \mathtt{05}x_{i-1,10}^{254}+\mathtt{05}x_{i-1,15}^{254}+ & \mathtt{12}x_{i-1,0}^{253}+\mathtt{1B}x_{i-1,5}^{253} & + \\ \mathtt{09}x_{i-1,10}^{253}+\mathtt{09}x_{i-1,15}^{253}+ & \mathtt{E9}x_{i-1,0}^{251}+\mathtt{10}x_{i-1,5}^{251}+ & \mathtt{F9}x_{i-1,10}^{251}+\mathtt{F9}x_{i-1,15}^{251} & + \\ \mathtt{4A}x_{i-1,0}^{247}+\mathtt{6F}x_{i-1,5}^{247}+ & \mathtt{25}x_{i-1,10}^{247}+\mathtt{25}x_{i-1,15}^{247}+ & \mathtt{F3}x_{i-1,0}^{239}+\mathtt{07}x_{i-1,5}^{239} & + \\ \mathtt{F4}x_{i-1,10}^{239}+\mathtt{F4}x_{i-1,15}^{239}+ & \mathtt{02}x_{i-1,0}^{223}+\mathtt{03}x_{i-1,5}^{223}+ & \mathtt{01}x_{i-1,10}^{223}+\mathtt{01}x_{i-1,15}^{223} & + \\ \mathtt{71}x_{i-1,0}^{191}+\mathtt{C4}x_{i-1,5}^{191}+ & \mathtt{B5}x_{i-1,10}^{191}+\mathtt{B5}x_{i-1,15}^{191}+ & \mathtt{05}x_{i-1,0}^{127}+\mathtt{8A}x_{i-1,5}^{127} & + \\ \mathtt{8F}x_{i-1,10}^{127}+\mathtt{8F}x_{i-1,15}^{127}+ & \mathtt{01}x_{i,0}+\mathtt{01}k_{i,0}+ & \mathtt{63} = 0. & \end{array}$$

3. equations for the last round:

$$\begin{pmatrix} x_{10,0}+k_{10,0} & x_{10,4}+k_{10,4} & x_{10,8}+k_{10,8} & x_{10,12}+k_{10,12} \\ x_{10,1}+k_{10,1} & x_{10,5}+k_{10,5} & x_{10,9}+k_{10,9} & x_{10,13}+k_{10,13} \\ x_{10,2}+k_{10,2} & x_{10,6}+k_{10,6} & x_{10,10}+k_{10,10} & x_{10,14}+k_{10,14} \\ x_{10,3}+k_{10,3} & x_{10,7}+k_{10,7} & x_{10,11}+k_{10,11} & x_{10,15}+k_{10,15} \end{pmatrix} + \\ + \begin{pmatrix} S(x_{9,0}) & S(x_{9,4}) & S(x_{9,8}) & S(x_{9,12}) \\ S(x_{9,5}) & S(x_{9,9}) & S(x_{9,13}) & S(x_{9,1}) \\ S(x_{9,10}) & S(x_{9,14}) & S(x_{9,2}) & S(x_{9,6}) \\ S(x_{9,15}) & S(x_{9,3}) & S(x_{9,7}) & S(x_{9,11}) \end{pmatrix} = 0,$$

4. key schedule equations

$$\begin{pmatrix} k_{i,0} \\ k_{i,1} \\ k_{i,2} \\ k_{i,3} \\ k_{i,4} \\ \vdots \\ k_{i,15} \end{pmatrix} = \begin{pmatrix} k_{i-1,0} + S(k_{i-1,13}) & +\xi^{i-1} \\ k_{i-1,1} + S(k_{i-1,14}) & \\ k_{i-1,2} + S(k_{i-1,15}) & \\ k_{i-1,3} + S(k_{i-1,12}) & \\ k_{i-1,4} + k_{i,0} & \\ \vdots & \\ k_{i-1,15} + k_{i,11} & \end{pmatrix}. \tag{3.4}$$

3.2.2 Systems of Quadratic Equations over GF(2)

To derive a system of equations over GF(2) for AES, the finite field $\mathbb{F}$ is considered as a vector space over GF(2). In this case, each bit of the internal states and the round keys is a new variable, and for any variable v we have $v^2+v=0$. We use $v_{i,j}^{(e)}$ to denote the eth bit variable of the jth byte of a byte array $\mathbf{V}$ in round i, where $0 \leq e \leq 7$, $0 \leq j \leq 15$ and $0 \leq i \leq 9$. To describe AES elegantly, the three following byte arrays are used for each round:

- the internal state before the `SubBytes` transformation,

$$\mathbf{X}_i = (\mathbf{x}_{i,0}, \ldots, \mathbf{x}_{i,15});$$

- the internal state after the `SubBytes` transformation,

$$\mathbf{Y}_i = (\mathbf{y}_{i,0}, \ldots, \mathbf{y}_{i,15});$$

- the round key,

$$\mathbf{K}_{i+1} = (\mathbf{k}_{i+1,0}, \ldots, \mathbf{k}_{i+1,15}).$$

In addition, let $\mathbf{K}_0 = (\mathbf{k}_{0,0}, \ldots, \mathbf{k}_{0,15})$ be a vector of variables for the initial key. Thus we work in the polynomial ring

$$R_{A2} = \mathrm{GF}(2)\left[x_{i,j}^{(e)},\ y_{i,j}^{(e)},\ k_{i+1,j}^{(e)},\ k_{0,j}^{(e)}\right]$$

with $0 \leq e \leq 7$, $0 \leq j \leq 15$ and $0 \leq i \leq 9$. By $((\mathbf{p}_0, \ldots, \mathbf{p}_{15}), (\mathbf{c}_0, \ldots, \mathbf{c}_{15}))$ denote a known plaintext/ciphertext pair.

The only operation in AES that is non-linear over GF(2) is `SubBytes`. It is based on the S-box, which can be expressed as a system of quadratic equations over GF(2) in input/output variables. To derive this system, consider

the following equations with unknown coefficients $a_{r,s}, b_{r,s}, c_{r,s}, d \in \mathrm{GF}(2)$:

$$\sum_{e_1=0}^{6} \sum_{e_2=e_1+1}^{7} (a_{e_1,e_2} x^{(e_1)} x^{(e_2)} + c_{e_1,e_2} y^{(e_1)} y^{(e_2)}) + \sum_{e_3=0}^{7} \sum_{e_4=0}^{7} b_{e_3,e_4} x^{(e_3)} y^{(e_4)} + \\ + \sum_{e_5=0}^{7} (a_{e_5,e_5} x^{(e_5)} + c_{e_5,e_5} y^{(e_5)}) + d = 0.$$

Substituting the bits of each element $\alpha \in \mathbb{F}$ for the corresponding input variables $x^{(e)}$ and the bits of $S(\alpha) \in \mathbb{F}$ for the corresponding output variables $y^{(e)}$ in this equation, we obtain a system of 256 linear equations over $\mathrm{GF}(2)$ in the 137 unknowns $\{a_{r,s}, c_{r,s} \colon 0 \le r \le s \le 7\} \cup \{b_{r,s} \colon 0 \le r, s \le 7\} \cup \{d\}$. Every solution of this system corresponds to the coefficients of some quadratic equation for the S-box. It can easily be checked that less the field equations there exist exactly 39 linearly independent quadratic equations for the AES S-box. Note that if for equations the set of quadratic terms is restricted to $\{x^{(r)} y^{(s)} \colon 0 \le r, s \le 7\}$, i.e., $a_{r,s} = c_{r,s} = 0$ for all r and s, then the system has 23 equations. Thus for the AES encryption we get a set of disjoin systems of quadratic equations, each of these systems describes the relations between the variables of $\mathbf{x}_{i,j}$ and $\mathbf{y}_{i,j}$ for some $0 \le j \le 15$ and $0 \le i \le 9$.

Further, each bit variable of $\mathbf{X}_{i+1}$ is connected with several bit variables of $\mathbf{Y}_i$ and one of $\mathbf{K}_{i+1}$ by a linear equation. The coefficients of these equations can be easily derived from the description of the `ShiftRows`, `MixColumns`, and `AddRoundKey` transformations. Also, we have plaintext/ciphertext equations:

$$x_{0,j}^{(e)} + k_{0,j}^{(e)} + p_j^{(e)} = 0 \quad \text{and} \quad y_{9,j}^{(e)} + k_{10,j'}^{(e)} + c_{j'}^{(e)} = 0$$

with $0 \le j \le 15$, $0 \le e \le 7$, and $j' = 5 \cdot j \mod 16$. Quadratic equations for the S-boxes in the key schedule can be obtained as stated above. Linear equations are given by

$$k_{i+1,j+4}^{(e)} + k_{i+1,j}^{(e)} + k_{i,j+4}^{(e)} = 0$$

for all $0 \le e \le 7$, $0 \le j \le 11$ and $0 \le i \le 9$.

Thus the resulting system consists of 2368 linear equations and $200 \cdot \varepsilon + 3968$ quadratic equations, where 3968 of the quadratic equations are the field equations, and $\varepsilon = 39$, if all possible linearly independent equations for the S-box are included into the system, or $\varepsilon = 23$, if only equations with quadratic terms in the form of $x \cdot y$ are taken into account. Note that using linear equations some variables can be eliminated from this system, for example, either all $\mathbf{Y}_i$ variables or all $\mathbf{X}_i$ variables.

3.2.3 Embedding in the Big Encryption System (BES)

The block cipher *Big Encryption System* (BES) was introduced by Murphy and Robshaw in [47]. This cipher was constructed to be an extension of the AES. This means that there is an injective map $\psi\colon \mathrm{GF}(2^8)^{16} \to \mathrm{GF}(2^8)^{16\times 8}$ such that if the AES using a cipher key K takes a plaintext P to the ciphertext C, then the ciphertext $\psi(C)$ is the result of the BES encryption of the plaintext $\psi(P)$ using the cipher key $\psi(K)$. Let $\phi\colon \mathrm{GF}(2^8) \to \mathrm{GF}(2^8)^8$ be a vector conjugate mapping such that

$$\phi(\sigma) = \left(\sigma^{2^0}, \sigma^{2^1}, \sigma^{2^2}, \sigma^{2^3}, \sigma^{2^4}, \sigma^{2^5}, \sigma^{2^6}, \sigma^{2^7}\right)$$

for all $\sigma \in \mathrm{GF}(2^8)$. Then the AES is embedded in the BES using ψ that is given by

$$(\sigma_0, \ldots, \sigma_{15})^T \overset{\psi}{\longmapsto} (\phi(\sigma_0), \ldots, \phi(\sigma_{15}))^T.$$

The reason of this embedding is that the BES uses only algebraic operations in $\mathrm{GF}(2^8)$ and can be expressed as a sparse multivariate quadratic system over $\mathrm{GF}(2^8)$.

Consider the BES and the corresponding system of equations. In the BES the same finite field $\mathbb{F} = \mathrm{GF}(2^8)$ is used as in the AES. Any internal state and the round keys of the BES can be represented by 16×8 matrices over $\mathbb{F}$. In each of $\mathcal{R}$ rounds of the encryption a state is transformed using the BES round function, which is slightly modified in the last round. This round function consists of three state transformations: a parallel application of 128 S-boxes, an affine transformation over $\mathbb{F}$, and the round key addition. All operations are defined so that if an input of the BES encryption is a plaintext $\psi(P)$ and a secret key $\psi(K)$ for some $P, K \in \mathbb{F}^{16}$, then the first column of any BES internal state is some internal state of the AES encryption of P with the secret key K.

We write $x_{i,j}^{(e)}$ and $w_{i,j}^{(e)}$ for the internal state variables before and after the ith S-box application and $k_{i,j}^{(e)}$ for the ith round key variables.

The BES S-box takes each $\sigma \in \mathbb{F}$ to σ^{254}, i.e., here the BES uses only the first component f_{-1} of the AES S-box. So we have

$$x_{i,j}^{(e)} \cdot w_{i,j}^{(e)} = 1$$

for all $0 \le e \le 7$, $0 \le j \le 15$ and $0 \le i \le 9$. Note that the input of the S-box $x = 0$ and the corresponding output $w = 0$ do not satisfy of this equation. Further, the BES affine transformation process can be divided into several steps. First an internal state $\Sigma \in \mathbb{F}^{16\times 8}$ is converted by

$$\Sigma' = \Sigma \cdot L_B,$$

where L_B is the following 8×8-matrix over $\mathbb{F}$:

$$L_B = \begin{pmatrix} \mathtt{05} & (\mathtt{8F})^{2^1} & (\mathtt{B5})^{2^2} & (\mathtt{01})^{2^3} & (\mathtt{F4})^{2^4} & (\mathtt{25})^{2^5} & (\mathtt{F9})^{2^6} & (\mathtt{09})^{2^7} \\ \mathtt{09} & (\mathtt{05})^{2^1} & (\mathtt{8F})^{2^2} & (\mathtt{B5})^{2^3} & (\mathtt{01})^{2^4} & (\mathtt{F4})^{2^5} & (\mathtt{25})^{2^6} & (\mathtt{F9})^{2^7} \\ \mathtt{F9} & (\mathtt{09})^{2^1} & (\mathtt{05})^{2^2} & (\mathtt{8F})^{2^3} & (\mathtt{B5})^{2^4} & (\mathtt{01})^{2^5} & (\mathtt{F4})^{2^6} & (\mathtt{25})^{2^7} \\ \mathtt{25} & (\mathtt{F9})^{2^1} & (\mathtt{09})^{2^2} & (\mathtt{05})^{2^3} & (\mathtt{8F})^{2^4} & (\mathtt{B5})^{2^5} & (\mathtt{01})^{2^6} & (\mathtt{F4})^{2^7} \\ \mathtt{F4} & (\mathtt{25})^{2^1} & (\mathtt{F9})^{2^2} & (\mathtt{09})^{2^3} & (\mathtt{05})^{2^4} & (\mathtt{8F})^{2^5} & (\mathtt{B5})^{2^6} & (\mathtt{01})^{2^7} \\ \mathtt{01} & (\mathtt{F4})^{2^1} & (\mathtt{25})^{2^2} & (\mathtt{F9})^{2^3} & (\mathtt{09})^{2^4} & (\mathtt{05})^{2^5} & (\mathtt{8F})^{2^6} & (\mathtt{B5})^{2^7} \\ \mathtt{B5} & (\mathtt{01})^{2^1} & (\mathtt{F4})^{2^2} & (\mathtt{25})^{2^3} & (\mathtt{F9})^{2^4} & (\mathtt{09})^{2^5} & (\mathtt{05})^{2^6} & (\mathtt{8F})^{2^7} \\ \mathtt{8F} & (\mathtt{B5})^{2^1} & (\mathtt{01})^{2^2} & (\mathtt{F4})^{2^3} & (\mathtt{25})^{2^4} & (\mathtt{F9})^{2^5} & (\mathtt{09})^{2^6} & (\mathtt{05})^{2^7} \end{pmatrix}.$$

It is easy to see that for any $\sigma \in \mathbb{F}$ we have

$$\phi(\sigma) \cdot L_B = \phi(f_L(\sigma)),$$

where f_L is the interpolation polynomial over $\mathbb{F}$ for the GF(2)-linear mapping of the AES S-box. Then the internal state Σ' is added with the constant 16×8-matrix C over $\mathbb{F}$

$$C = \left.\begin{pmatrix} (\mathtt{63})^{2^0} & (\mathtt{63})^{2^1} & \dots & (\mathtt{63})^{2^6} & (\mathtt{63})^{2^7} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ (\mathtt{63})^{2^0} & (\mathtt{63})^{2^1} & \dots & (\mathtt{63})^{2^6} & (\mathtt{63})^{2^7} \end{pmatrix}\right\} 16 \text{ rows.}$$

Thus the `SubBytes` transformation of the AES is completely embedded into the BES round function. The next two linear transformation of the state matrix correspond to `ShiftRows` and `MixColumns` of the AES. The first of them is the row permutation given by

$$\pi = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 \\ 0 & 5 & 10 & 15 & 4 & 9 & 14 & 3 & 8 & 13 & 2 & 7 & 12 & 1 & 6 & 11 \end{pmatrix}.$$

After this permutation, the eth column of the resulting state is multiplied on the left by the 16×16-matrix $M_e = (a_{i,j}^{2^e})$, where $M_0 = (a_{i,j})$ is given by

$$M_0 = \begin{pmatrix} D & 0 & 0 & 0 \\ 0 & D & 0 & 0 \\ 0 & 0 & D & 0 \\ 0 & 0 & 0 & D \end{pmatrix}$$

with the 4×4-matrix D used in the `MixColumns` transformation of the AES. Like in the AES, this operation is not applied in the last round of the BES encryption.

In the last operation of the BES round function, the round key in the form of 16×8-matrix is added to the internal state. Also, before the first round an initial round key is added to each plaintext. The round keys are generated from the secret key using the BES key schedule, which is an extension of the AES key schedule. Let $K_B \in \mathbb{F}^{16\times 8}$ be a BES secret key. Then $K_0 = K_B$ is an initial key, and the ith round key K_i is generated from K_{i-1} as follows:

$$\begin{aligned}
K_{i,0} &= S_B(K_{i-1,13}) \cdot L_B + K_{i-1,0} + C' + V_i,\\
K_{i,1} &= S_B(K_{i-1,14}) \cdot L_B + K_{i-1,1} + C',\\
K_{i,2} &= S_B(K_{i-1,15}) \cdot L_B + K_{i-1,2} + C',\\
K_{i,3} &= S_B(K_{i-1,12}) \cdot L_B + K_{i-1,3} + C',\\
K_{i,j} &= K_{i,j-4} + K_{i-1,j}, \qquad \text{for } 4 \le j \le 15.
\end{aligned}$$

Here for $0 \le i \le 10$ and $0 \le j \le 15$

- $K_{i,j}$ is the jth row of the key K_i;
- $S_B(K_{i,j})$ is the result of the parallel application of the BES S-box to each element of the row $K_{i,j}$;
- the matrix L_B is given above, and
$$C' = \left((\mathsf{63})^{2^0}, (\mathsf{63})^{2^1}, \ldots, (\mathsf{63})^{2^6}, (\mathsf{63})^{2^7}\right);$$
- the vector V_i of round constants is given by
$$V_i = \left((\xi^{i-1})^{2^0}, (\xi^{i-1})^{2^1}, \ldots, (\xi^{i-1})^{2^6}, (\xi^{i-1})^{2^7}\right).$$

The key recovery problem for the block cipher BES can be described as follows. Put

$$R_B = \mathbb{F}\left[x_{i,j}^{(e)}, w_{i,j}^{(e)}, k_{i,j}^{(e)}, k_{10,j}^{(e)}\right]$$

with $0 \le i \le 9, 0 \le j \le 15$, and $0 \le e \le 7$. Let $(\mathbf{p}, \mathbf{c}) \in \mathbb{F}^{16\times 8} \times \mathbb{F}^{16\times 8}$ be a known plaintext/ciphertext pair. The BES systems denoted by $\mathbb{S}_B$ consists of equations for the encryption and the key schedule. The BES encryption of $\mathbf{p} = (p_j^{(e)})$ to $\mathbf{c} = (c_j^{(e)})$ is given by:

$$\left.\begin{aligned}
x_{0,j}^{(e)} + p_j^{(e)} + k_{0,j}^{(e)} &= 0\\
x_{i,j}^{(e)} \cdot w_{i,j}^{(e)} + 1 &= 0\\
x_{i+1,j}^{(e)} + \sum \alpha_{s,t} w_{i,s}^{(t)} + k_{i+1,j}^{(e)} &= 0\\
x_{9,j}^{(e)} \cdot w_{9,j}^{(e)} + 1 &= 0\\
c_j^{(e)} + \sum \tilde{\alpha}_{s,t} w_{9,s}^{(t)} + k_{10,j}^{(e)} &= 0
\end{aligned}\right\} \begin{aligned} &i = 0, \ldots, 8\\ &j = 0, \ldots, 15\\ &e = 0, \ldots, 7 \end{aligned} \qquad \text{(B)}$$

and the equations for the key schedule are:

$$\left.\begin{aligned} k_{i,j}^{(e)} + k_{i,j-4}^{(e)} + k_{i-1,j}^{(e)} &= 0 \\ k_{i-1,15}^{(e)} \left(\sum \beta_{e,s} \left(k_{i,0}^{(s)} + k_{i-1,0}^{(s)} + \gamma_i^{(s)} \right) \right) + 1 &= 0 \\ k_{i-1,11+t}^{(e)} \left(\sum \beta_{e,s} \left(k_{i,t}^{(s)} + k_{i-1,t}^{(s)} \right) \right) + 1 &= 0 \end{aligned} \right\} \begin{array}{l} i = 1, \ldots, 10 \\ j = 4, \ldots, 15 \\ e = 0, \ldots, 7 \\ t = 1, \ldots, 3 \end{array} \qquad (B')$$

All coefficients $\alpha_{i,j}$, $\tilde{\alpha}_{i,j}$, $\beta_{i,j}$, and $\gamma_i^{(e)}$ can be obtained from the description of the BES linear transformation and the key schedule.

Since the BES is an extension of the AES, the system $\mathbb{S}_B$ also describes the key recovery problem for the AES. However in this case the relations between conjugate elements of the internal states and the round keys of the embedded AES are not taken into account. This relations are given by

$$\left.\begin{aligned} (x_{i,j}^{(e)})^2 + x_{i,j}^{(e+1)} &= 0; & (x_{i,j}^{(7)})^2 + x_{i,j}^{(0)} &= 0; \\ (w_{i,j}^{(e)})^2 + w_{i,j}^{(e+1)} &= 0; & (w_{i,j}^{(7)})^2 + w_{i,j}^{(0)} &= 0; \\ (k_{i,j}^{(e)})^2 + k_{i,j}^{(e+1)} &= 0; & (k_{i,j}^{(7)})^2 + k_{i,j}^{(0)} &= 0; \\ (k_{10,j}^{(e)})^2 + k_{10,j}^{(e+1)} &= 0; & (k_{10,j}^{(7)})^2 + k_{10,j}^{(0)} &= 0; \end{aligned} \right\} \begin{array}{l} i = 0, \ldots, 9 \\ j = 0, \ldots, 15 \\ e = 0, \ldots, 6 \end{array} \qquad (B'')$$

Denote the system that consists of $\mathbb{S}_B$ and these equations by $\mathbb{S}_{EA}$. We see that if $v_i^2 + v_{i+1} = 0$ with $0 \leq i \leq 7$ and $v_8 = v_0$, then

$$v_i^{2^8} + v_i = 0$$

for all v_i. Hence system $\mathbb{S}_{EA}$ has only solutions in $\mathbb{F}$.

Chapter 4

Block Ciphers Sensitive to Gröbner Basis Attacks

In order to analyze the power of Gröbner basis attacks, we construct two parameterized families of block ciphers with a simple algebraic structure. The first family represents Feistel networks, and the other one represents SPN ciphers. The following parameters of this ciphers can be varied: the length of blocks, the number of S-boxes, the number of rounds, S-box functions, and linear transformations. Thus we can study how Gröbner basis attacks depend on this parameters. We show that there are sets of the parameters with which the ciphers are resistant against differential and linear cryptanalysis, but can be broken using Gröbner basis. Moreover we demonstrate that in some cases the key recovery problem can be easy reduced to a Gröbner basis conversion problem. Actually we construct a DRL Gröbner basis for block ciphers with a polynomial S-box. Since the time and space complexity of the FGLM Gröbner basis conversion algorithm is known, the theoretical upper bound for the time and space complexity of Gröbner basis attacks on this ciphers is obtained. The results given in this chapter were presented in [17].

4.1 Flurry and Curry: Two Families of Block Ciphers

In this section we present two families of iterated block ciphers. The first family has a Feistel network structure, and is called Flurry. The second family, called Curry, consists of SPN ciphers similar to Square [26], one of the Rijndael predecessors. We specify a cipher parameter space, which satisfies the following two conditions:

- the Flurry and Curry ciphers must be resistant against differential and linear cryptanalysis;
- the key recovery problem for this ciphers can be described by a relatively simpler system of polynomial equations.

We construct our ciphers using the wide trail strategy, an approach is applied to design Square, Rijndael, Twofish, etc. [29, 28]. In order that the second condition holds, for each cipher we fix a finite field $\mathbb{F} = \mathrm{GF}(2^n) = \mathrm{GF}(2)(\xi)$, where $n \in \{8, 16, 32, 64\}$, and ξ is a generating element of $\mathbb{F}$ over $\mathrm{GF}(2)$, and in the round function all operations are over $\mathbb{F}$. Further, any internal state of this cipher consists of elements of $\mathbb{F}$. For Flurry ciphers, the vector representation of the internal states is used, and all Curry internal states are written in matrix form.

4.1.1 Description of Flurry

First we describe the family $\textsc{Flurry}(n, m, r, f, D)$ of Feistel ciphers, which depends on the following parameters:

- $n, m \in \mathbb{N}$: the plaintext space, the ciphertext space, and the cipher key space are $\mathbb{F}^{2m} = \mathrm{GF}(2^n)^{2m}$, i.e., the block and key lengths are equal to $N = 2nm$ bits; also any round key consists of m elements of $\mathbb{F}$;
- $r \in \mathbb{N}$: the number of rounds;
- $f : \mathbb{F} \to \mathbb{F}$: a non-linear mapping giving the S-Box of the round function;
- $D = (d_{i,j}) \in \mathbb{F}^{m \times m}$: a matrix describing the linear transformation of the round function.

In each round the internal state is split into two halves, $L = (l_1, \ldots, l_m) \in \mathbb{F}^m$ and $R = (r_1, \ldots, r_m) \in \mathbb{F}^m$. Let $K = (k_1, \ldots, k_m) \in \mathbb{F}^m$ be a round key. It is derived from a cipher key using the Flurry key schedule. Then for Flurry the round function $\rho : \mathbb{F}^m \times \mathbb{F}^m \times \mathbb{F}^m \to \mathbb{F}^m \times \mathbb{F}^m$ is given by:

$$\rho(L, R, K) = (R, G(R, K) + L),$$

where $G : \mathbb{F}^m \times \mathbb{F}^m \to \mathbb{F}^m$ is composed of the round key addition, the parallel application of m S-Boxes, and the linear transformation using the matrix D:

$$G(r_1, \ldots, r_m, k_1, \ldots, k_m) = D \times \begin{pmatrix} f(r_1 + k_1) \\ f(r_2 + k_2) \\ \vdots \\ f(r_m + k_m) \end{pmatrix}.$$

A plaintext (L_0, R_0) is encrypted into a ciphertext (L_r, R_r) by implementing the round function ρ exactly r times with an additional key addition after the last round transformation:

$$\begin{aligned}(L_e, R_e) &= \rho(L_{e-1}, R_{e-1}, K_e) \qquad e = 1, 2, \ldots, r-1 \\ (L_r, R_r) &= \rho(L_{r-1}, R_{r-1}, K_r) + (K_{r+1}, K_{r+2})\end{aligned}$$

The inverse round function ρ^{-1} is given by

$$\rho^{-1}(L, R, K) = (G(L, K) + R, L),$$

and the decryption of a ciphertext is described as:

$$\begin{aligned}(L_{r-1}, R_{r-1}) &= \rho^{-1}(L_r + K_{r+1}, R_r + K_{r+2}, K_r) \\ (L_{e-1}, R_{e-1}) &= \rho^{-1}(L_e, R_e, K_e) \qquad e = r-1, r-2, \ldots, 1\end{aligned}$$

The key schedule

To generate the round keys from a cipher key, FLURRY uses an affine transformation over $\mathbb{F}$. The cipher key is split into two halves, $(K_L, K_R) \in \mathbb{F}^m \times \mathbb{F}^m$. Let $K_e \in \mathbb{F}^m$ be the eth round key with $1 \leq e \leq r+2$. Then we have:

$$\begin{aligned}&K_1 = K_L \qquad K_2 = K_R \\ &K_e = D \cdot K_{e-1} + K_{e-2} + v_e \qquad e = 3, 4, \ldots, r+2\end{aligned}$$

where D is the same matrix used in the round function of the cipher and the v_e are round constants:

$$v_e = ((\xi+1)^{e-1}, (\xi+1)^e, \ldots, (\xi+1)^{e+m-2})$$

4.1.2 Description of CURRY

Now we describe the cipher family CURRY(n, m, r, f, D), which has an SPN structure. In this case the cipher parameters are:

- $n, m \in \mathbb{N}$: the plaintext space, the ciphertext space and the cipher key space are $\mathbb{F}^{m \times m}$, where $\mathbb{F} = \mathrm{GF}(2^n)$, hence the block and key lengths are equal to $N = nm^2$ bits; moreover all internal states and round keys of CURRY cipher are represented by $m \times m$-matrices in $\mathbb{F}$;
- $r \in \mathbb{N}$: the number of rounds;
- $f : \mathbb{F} \to \mathbb{F}$: a bijective non-linear mapping giving the S-Box of the round function;

- $D = (d_{i,j}) \in \mathbb{F}^{m \times m}$: an invertible matrix used for diffusion.

The Curry round function $\rho : \mathbb{F}^{m \times m} \times \mathbb{F}^{m \times m} \rightarrow \mathbb{F}^{m \times m}$ is composed of four operations: the round key addition, a non-linear transformation G given by the parallel application of m^2 S-Boxes, matrix transposition, and matrix multiplication; thus ρ is defined as:

$$\rho(S, K) = D \cdot G(S + K)^T,$$

where S is a $m \times m$ state matrix, K is a round key, and

$$G((s_{i,j})) = (f(s_{i,j})).$$

A plaintext S_0 is encrypted into a ciphertext S_r by implementing the round function ρ exactly r times followed by an additional key addition after the last round:

$$\begin{aligned} S_e &= \rho(S_{e-1}, K_e) \qquad e = 1, 2, \ldots, r-1 \\ S_r &= \rho(S_{r-1}, K_r) + K_{r+1}. \end{aligned}$$

The inverse round function ρ^{-1} is given by:

$$\rho^{-1}(S, K) = G^{-1}((D^{-1} \cdot S)^T) + K,$$

and the decryption process consists of the following sequence of iterated steps:

$$\begin{aligned} S_{r-1} &= \rho^{-1}(S_r + K_{r+1}, K_r) \\ S_{e-1} &= \rho^{-1}(S_e, K_e) \qquad e = r-1, r-2, \ldots, 1 \end{aligned}$$

The key schedule

To derive the round keys $K_1, \cdots, K_{r+1}$, the Curry key schedule is applied to a cipher key $K \in \mathbb{F}^{m \times m}$. The key schedule is affine over $\mathbb{F}$, and consists of the following sequence of step:

$$\begin{aligned} &K_1 = K \\ &K_e = D \cdot K_{e-1}^T + A_e \qquad 2 \leq e \leq r+1 \end{aligned}$$

where D is the same matrix used in the round function and $A_e \in \mathbb{F}^{m \times m}$ is a matrix of round constants. For $2 \leq e \leq r+1$ and $1 \leq i, j \leq m$, the element $a_{i,j}$ of A_e is equal to $\xi^{e+(i-1)m+j}$.

4.1.3 Selected Parameters

For FLURRY and CURRY we now specify a set of S-Box functions and linear transformations so that the ciphers with these parameters have a good resistance against differential and linear cryptanalysis even if the number of rounds is low. An additional condition for the parameters is that the degree of FLURRY and CURRY equations, which will be described in the next subsection, must be relative small.

We use matrices of Maximum Distance Separable codes – *MDS matrices* for short – for the matrix D in the linear layer and the key schedule. We chose these types of linear transformations since they have optimal diffusion properties. This strategy is widely used in modern block cipher design; all ciphers following the wide trail design use diffusion optimal matrices. The matrix D_4 below actually is the matrix used in the `MixColumns` step of AES, D_2 is equivalent to a Pseudo-Hadamard Transform over $\mathbb{F}$.

The S-Box functions

The only non-linear components of FLURRY and CURRY are the S-Boxes. For our purpose we have selected several suitable functions from the set

$$\left\{ f_d \colon \mathbb{F} \to \mathbb{F},\ x \mapsto x^d \right\}.$$

From the point of view of linear and differential cryptanalysis, properties of power functions over finite fields of characteristic two, as their differential uniformity and nonlinearity, are well investigated [49, 6, 30].

Definition 4.1.1. Let $f : \mathbb{F} \to \mathbb{F}$ be a mapping and

$$\delta = \max_{\substack{a,b \in \mathbb{F} \\ a \neq 0}} \#\{x \in \mathbb{F} \colon\ f(x+a) = f(x) + b\}.$$

Then f is called *differentially δ-uniform.*

For any $a = \sum_{i=0}^{n-1} a_i \xi^i \in \mathbb{F}$ and $b = \sum_{i=0}^{n-1} b_i \xi^i \in \mathbb{F}$ we set

$$\langle a, b \rangle = \sum_{i=0}^{n-1} a_i b_i$$

Definition 4.1.2. The *nonlinearity* of a function $f : \mathbb{F} \to \mathbb{F}$ is defined as

$$\mathcal{N}(f) = \min_{\substack{a,b \in \mathbb{F} \\ b \neq 0}} \#\{x \in \mathbb{F} \colon\ \langle x, a \rangle \neq \langle f(x), b \rangle\}$$

Table 4.1: S-Box mappings over $\mathbb{F} = \mathrm{GF}(2^n)$ with $n \in \{8, 16, 32, 64\}$

function	bijective over $\mathbb{F}$	δ	$\mathcal{N}(f)$
$f_{-1}\colon \quad x \mapsto \begin{cases} x^{-1} & \text{if } x \neq 0 \\ 0 & \text{if } x = 0 \end{cases}$	yes	4	$2^{n-1} - 2^{\frac{n}{2}}$
$f_3\colon \quad x \mapsto x^3$	no	2	$\geq 2^{n-1} - 2^{\frac{n}{2}}$
$f_5\colon \quad x \mapsto x^5$	no	4	$\geq 2^{n-1} - 2^{\frac{n}{2}+1}$
$f_7\colon \quad x \mapsto x^7$	yes	≤ 6	$\geq 2^{n-1} - 3 \cdot 2^{\frac{n}{2}}$

Consider f_d with $d = 3, 5, 7$, and $d = 2^n - 2$. For the last function we will write f_{-1} instead of f_{2^n-2} and call it the *inversion S-box*, while f_3, f_5 and f_7 are called *monomial S-Boxes*. These functions have the following differential uniformity and nonlinearity.

Lemma 4.1.3. *1. f_3 is a 2-uniform mapping*

2. f_{-1} and f_5 are 4-uniform mappings.

3. f_7 has δ-uniformity of 6 or less.

Proof. Obviously for all $a, b \in \mathbb{F}$ with $a \neq 0$ the equation $x^7 + (x + a)^7 = b$ has at most 6 roots. For claims 1 and 2, see [49]. □

Lemma 4.1.4. *1. The nonlinearity of f_{-1} is $2^{n-2} - 2^{\frac{n}{2}}$.*

2. For a polynomial function $f : \mathbb{F} \to \mathbb{F}$ of degree d the following holds true: $\mathcal{N}(f) \geq 2^{n-1} - \lfloor \frac{d-1}{2} \rfloor 2^{\frac{n}{2}}$

Proof. For claim 1, see [30], for claim 2 see [19]. □

These results are summarized in Table 4.1. Note that the functions f_3 and f_5 are non-bijective, and hence they cannot be used as a Curry S-box.

The linear transformations

From the point of view of cryptanalysis, two important characteristic of a linear transformation are its differential and linear branch number [28]. Let $w(X)$ be the hamming weight of a vector $X = (x_1, \ldots, x_m) \in \mathbb{F}^m$, i.e., the number of non-zero coordinates of X.

Definition 4.1.5. Let M be a $m \times m$-matrix in $\mathbb{F}$. The *differential branch number* $\mathcal{B}_d(M)$ of M is defined as

$$\mathcal{B}_d(M) = \min_{X \in \mathbb{F}^m \setminus \{0\}} (w(X) + w(M \cdot X))$$

while the *linear branch number* $\mathcal{B}_l(M)$ is defined as $\mathcal{B}_l(M) = \mathcal{B}_d(M^T)$.

Since for any $m \times m$-matrix M and the column vector $X_0 = (1, 0, \ldots, 0)^T \in \mathbb{F}^m$ we have $w(M \cdot X_0) \leq m$, the differential and linear branch number of a linear transformation of $\mathbb{F}^m$ are bounded above by $m + 1$. An useful criteria for a linear transformation to have the maximal differential branch number is given in the following proposition [39].

Proposition 4.1.6. *Let M be a non-singular $m \times m$-matrix in $\mathbb{F}$. Then $\mathcal{B}_d(M) = m + 1$ iff any square submatrix of M is non-singular.*

It follows easily from this proposition that if $\mathcal{B}_d(M) = m + 1$, then $\mathcal{B}_d(M) = \mathcal{B}_l(M)$, and in this case it suffices to speak of *the branch number* $\mathcal{B}(M)$ of a matrix M.

For linear transformations of Flurry and Curry we use the following matrices:

$$D_1 = \begin{pmatrix} \xi \end{pmatrix} \qquad D_2 = \begin{pmatrix} \xi & 1 \\ 1 & 1 \end{pmatrix} \qquad D_4 = \begin{pmatrix} \xi & \xi+1 & 1 & 1 \\ 1 & \xi & \xi+1 & 1 \\ 1 & 1 & \xi & \xi+1 \\ \xi+1 & 1 & 1 & \xi \end{pmatrix}$$

Obviously, the matrices D_1 and D_2 have the maximal branch number for any $\mathbb{F} = \mathrm{GF}(2^n)$ with $n \geq 2$, and $\mathcal{B}(D_4) = 5$ whenever $n > 4$.

Lemma 4.1.7. *Let $\mathbb{F} = GF(2^n)$ with $n \geq 5$. Then $\mathcal{B}(D_1) = 2$, $\mathcal{B}(D_2) = 3$, and $\mathcal{B}(D_4) = 5$.*

4.1.4 Polynomial Representation of the Ciphers

In the following polynomial representations for Flurry and Curry are given. Like in the case of AES, to describe the transformation of a plaintext into a ciphertext here also intermediate state variables are used. We define the state of round 0 to be the initial state and call the variables of the initial state *plaintext variables*. Correspondingly the variables referring to the state after the execution of the last round are called *ciphertext variables*. The set of state variables of a cipher is denoted by $\mathcal{X}$, the set of expanded key variables by $\mathcal{K}$. All polynomials considered are then elements of the polynomial ring $R = \mathbb{F}[\mathcal{X} \cup \mathcal{K}]$.

Denote by $x_i^{(e)}$ the variable referring to the ith element of the Flurry cipher state after the eth application of the round function, and by $k_i^{(e)}$ the variable referring to the ith element of the eth round key. For SPN ciphers, we denote the *internal state variables* after the eth application of the round function by $x_{i,j}^{(e)}$ and the *expanded key variables* by $k_{i,j}^{(e)}$. By definition, $x_i^{(0)}$ is a Flurry plaintext variable, while a plaintext variable of Curry is denoted by $x_{i,j}^{(0)}$.

- Flurry(n, m, r, f, D)
  For Feistel ciphers the left half of the state in round e is identical to the right half of the state in round $e-1$, and we have the following mr trivial linear equations:

$$x_j^{(e)} + x_{j+m}^{(e-1)} = 0$$

  Let $D = (d_{i,j})$ with $1 \leq i, j \leq m$. Each monomial S-Box f_d of the cipher induces a polynomial equation of degree $d = \deg(f_d)$. Thus we get a total of mr non-linear equations of form:

$$x_{m+j}^{(e)} + x_j^{(e-1)} + \sum_{l=1}^{m} d_{j,l} \cdot \left(x_{m+l}^{(e-1)} + k_l^{(e)} \right)^d = 0$$

  with $1 \leq e \leq r,\ 1 \leq j \leq m$.

  In the case of the inversion S-Box f_{-1} such equations are of degree $2^n - 2$. However they can be closely approximated by quadratical equations as follows. Let $y_i^{(e)}$ be the additional variable referring to the output of the ith S-box in the eth round. Then,

$$y_i^{(e)} \cdot \left(x_{m+i}^{(e-1)} + k_i^{(e)} \right) = 1$$

  holds with probability $\frac{2^n-1}{2^n}$ for any $1 \leq i \leq m,\ 1 \leq e \leq r$, and

$$x_{m+j}^{(e)} + x_j^{(e-1)} + \sum_{l=1}^{m} d_{j,l} \cdot y_l^{(e)} = 0.$$

  Since for the Flurry linear transformation only invertible matrices D are selected, all additional variables $y_i^{(e)}$ can be eliminated from the system using the above linear equations. The resulting equations are then of the following form:

$$\left(x_{m+i}^{(e-1)} + k_i^{(e)} \right) \cdot \sum_{j=1}^{m} d'_{i,j} \cdot \left(x_{m+j}^{(e)} + x_j^{(e-1)} \right) + 1 = 0.$$

  Since after the last application of the round function the internal state of the cipher is added with the round key once more, the equations for the last round are of a slightly different form. Here the linear equations for the left half are given by

$$x_j^{(r)} + x_{j+m}^{(r-1)} + k_j^{(r+1)} = 0,$$

while for the right half of the state we have

$$x_{m+j}^{(r)} + x_j^{(r-1)} + k_j^{r+2} + \sum_{l=1}^{m} d_{j,l} \cdot \left(x_{m+l}^{(r-1)} + k_l^{(r)} \right)^d = 0,$$

in the case of a monomial S-box, and

$$\left(x_{m+j}^{(r-1)} + k_j^{(r)} \right) \cdot \sum_{l=1}^{m} d'_{j,l} \cdot \left(x_{m+l}^{(r)} + x_l^{(r-1)} + k_l^{r+2} \right) + 1 = 0,$$

if the S-box is inversion, where $1 \leq j \leq m$.

Also, we see that for $\textsc{Flurry}(n, m, r, f_{-1}, D_m)$ the obtained polynomial system is correct only with probability $\left(\frac{2^n-1}{2^n}\right)^{mr}$.

The linear equations for the key schedule of FLURRY can be written as:

$$k_j^{(e)} + k_j^{(e-2)} + (\theta + 1)^{et+j} + \sum_{l=1}^{m} d_{j,l} k_l^{(e-1)} = 0$$

with $2 \leq e \leq r,\ 1 \leq j \leq m$.

- $\textsc{Curry}(n, m, r, f, D)$
  In this case, no linear equations hold between intermediate state variables.

  Let $D = (d_{i,j})$ with $1 \leq i, j \leq m$. In the case of a monomial S-box f_d we get the following equations:

$$x_{i,j}^{(e)} + \sum_{l=1}^{m} d_{i,l} \cdot \left(x_{j,l}^{(e-1)} + k_{j,l}^{(e)} \right)^d = 0$$

  with $1 \leq e \leq r - 1$, and

$$x_{i,j}^{(r)} + k_{i,j}^{(r+1)} + \sum_{l=1}^{m} d_{i,l} \cdot \left(x_{j,l}^{(r-1)} + k_{j,l}^{(r)} \right)^d = 0$$

  for the last round; here in all equations $1 \leq i, j \leq m$. To describe the inversion S-box as a quadratic equation, again an additional variables are used. Denote by $y_{i,j}^{(e)}$ the variable in row i, column j of the state after the eth application of the S-box layer. Then the quadratic equation for the inversion S-box can be written as:

$$y_{i,j}^{(e)} \cdot \left(x_{i,j}^{(e-1)} + k_{i,j}^{(e)} \right) = 1$$

This equation holds with probability $\frac{2^n-1}{2^n}$. Further, the equations for the Curry linear transformation are of form

$$x_{i,j}^{(e)} + \sum_{l=1}^{m} d_{i,l} \cdot y_{j,l}^{(e)} \quad = \quad 0,$$

with $1 \leq i, j \leq m,\ 1 \leq e \leq r-1$, while in the last round they are slightly different:

$$x_{i,j}^{(r)} + k_{i,j}^{(r+1)} + \sum_{l=1}^{m} d_{i,l} \cdot y_{j,l}^{(r)} \quad = \quad 0.$$

Any linear transformation of the Curry round function is invertible, therefore all additional variables $y_{i,j}^{(e)}$ can be eliminated from the system using these linear equations. In this case we obtain the following system of quadratic equations:

$$\left(x_{i,j}^{(e-1)} + k_{i,j}^{(e)}\right) \cdot \sum_{l=1}^{m} d'_{j,l} \cdot \left(x_{l,i}^{(e)} + \delta_e k_{i,j}^{(r+1)}\right) + 1 = 0,$$

where $1 \leq i, j \leq m,\ 1 \leq e \leq r$, the parameter $\delta_e = 1$ if $e = r$, and $\delta_e = 0$ otherwise. However, this polynomial system does not hold with probability one but with probability $\left(\frac{2^n-1}{2^n}\right)^{m^2 r}$.

The linear equations for the key schedule can be expressed as follows:

$$k_{i,j}^{(e)} + (\theta)^{e+(i-1)m+j} + \sum_{l=1}^{m} d_{i,l} k_{l,j}^{(e-1)} = 0$$

with $2 \leq e \leq r+1,\ 1 \leq i, j \leq m$.

Note that the field equations $\mathfrak{v}^{2^n} + \mathfrak{v} = 0$ are not included in our polynomial systems.

4.2 Resistance against Classical Attacks

In this section we determine the strength of our cipher constructions against differential and linear cryptanalysis. Differential cryptanalysis is a chosen-ciphertext attack due to Biham and Shamir and was the first successful attack on the DES [7]. This type of attack exploits biases in the first order derivative of the cipher. For carefully chosen plaintexts with specific differences a

cryptanalyst makes assumption about their propagation through the cipher and predicts output differences in ciphertext pairs. If these predictions are correct with sufficiently high probability they allow an attacker to determine round key bits.

Linear cryptanalysis is a known plaintext attack that was devised by Matsui [46] to attack the DES. For this attack to succeed, the cryptanalyst has to construct a probable key-independent linear approximation for individual output bits of the cipher. By counting the number of time this linear approximation agrees with the actual output of the cipher she can establish which value for the key bit is more likely.

The notion of *practical security* of block ciphers against differential and linear cryptanalysis was introduced by Knudsen [43]. The exact definition of this notion is postponed to the end of Section 4.2.2. We will derive the number of rounds that will make our cipher practically secure against differential and linear cryptanalysis.

Note that our objective was not to evaluate the strength of our ciphers against all known attacks. Our ciphers may very well be vulnerable against one or several advanced attacks even if they resist standard linear and differential cryptanalysis. Indeed, as an example we argue that the choices we have made for the S-Boxes are very weak against interpolation attacks.

4.2.1 Estimating the Resistance against Differential and Linear Cryptanalysis

From the point of view of linear and differential cryptanalysis, two important characteristics for an iterated block cipher are the linear and differential probability of its round function.

Let $\rho : \mathrm{GF}(2)^N \to \mathrm{GF}(2)^N$ be a function for which we wish to compute the linear and differential probability. In the following X denotes a uniformly distributed random variable in $\mathrm{GF}(2)^N$.

Definition 4.2.1. The linear probability for a pair $(a, b) \in \mathrm{GF}(2)^N \times \mathrm{GF}(2)^N$ with $a \neq 0$ is defined as

$$\mathrm{LP}_\rho(a, b) = \left(2 \cdot \Pr_X \left\{\langle a, X\rangle = \langle b, \rho(X)\rangle\right\} - 1\right)^2$$

Here $\langle X, Y\rangle = \sum x_i y_i$ for any $X = (x_1, \ldots, x_N)$, $Y = (y_1, \ldots, y_N) \in \mathrm{GF}(2)^N$. In the above definition, a is called *input mask* and b is called *output mask* of a round. A vector of masks $A = (a_1, \ldots, a_{r+1})$ with $a_i \neq 0$ for all $1 \leq i \leq r$ is called *linear characteristic* of a cipher.

Definition 4.2.2. The differential probability for a pair $(\Delta x, \Delta y) \in \mathrm{GF}(2)^N \times \mathrm{GF}(2)^N$ with $\Delta x \neq 0$ is defined as

$$\mathrm{DP}_\rho(\Delta x, \Delta y) = \Pr_X \{\rho(X) + \rho(X + \Delta x) = \Delta y\}$$

The value Δx is called *input difference* of a round, while Δy is called *output difference*. A vector of differences $A = (a_1, \ldots, a_{r+1})$ with $a_i \neq 0$ for all $1 \leq i \leq r$ is called *differential characteristic* of a cipher.

Definition 4.2.3. Let Ω_L be the set of all linear characteristics and Ω_D the set of all differential characteristics of a cipher C with a round function ρ. The maximum linear characteristic probability (MLCP) of C is

$$\mathrm{MLCP}(C) = \max_{A \in \Omega_L} \prod_{i=1}^{r} \mathrm{LP}_\rho(a_i, a_{i+1})$$

Analogously the maximum differential characteristic probability (MDCP) of C is

$$\mathrm{MDCP}(C) = \max_{A \in \Omega_D} \prod_{i=1}^{r} \mathrm{DP}_\rho(a_i, a_{i+1})$$

To evaluate the linear and differential probability for an arbitrary function $\rho : \mathrm{GF}(2)^N \to \mathrm{GF}(2)^N$ can be hard, if N is large, e.g., $N = 128$. Fortunately, if the round function of a cipher has a SP structure, one can estimate MLCP and MDCP using properties of the S-boxes. Here fundamental parameters are the maximal linear and differential probability of the S-box function as well as the minimum number of active S-Boxes M over consecutive rounds of the cipher. Kanda [42] gives useful results on both SPN ciphers and Feistel ciphers with a SP round function; from these we derive the following theorem:

Theorem 4.2.4. *Suppose C is either a SPN cipher or a Feistel ciphers with a SP round function, p and q are the maximum differential and linear probabilities of all S-box functions respectively, and M is the minimal number of active S-Boxes. Then,*

$$\mathrm{MDCP}(C) \leq p^M \text{ and } \mathrm{MLCP}(C) \leq q^M.$$

Also, according to [42] the minimal number of active S-boxes can be estimated as follows:

Lemma 4.2.5. *The minimum number of active S-boxes in 4, 6, 8 consecutive rounds of a Feistel cipher with SP round function is lower bounded by $\mathcal{B}(D)$, $\mathcal{B}(D)+2$ and $2\mathcal{B}(D)+1$ respectively. For an SPN cipher the minimum number of active S-Boxes for $2r$ consecutive rounds is lower bounded by $r\mathcal{B}(D)$.*

Table 4.2: The maximum differential and linear probability, $p(f)$ and $q(f)$, of the S-Box function $f\colon \mathrm{GF}(2^n) \to \mathrm{GF}(2^n)$ and the miminum number M of active S-Boxes for $\textsc{Flurry}(n, m, r, f, D_m)$ and $\textsc{Curry}(n, m, r, f, D_m)$

S-box	$p(f)$	$q(f)$	M	Flurry			Curry	
f_{-1}	2^{2-n}	2^{2-n}	#rounds	$m=1$	2	4	$m=2$	4
f_3	2^{1-n}	$\leq 2^{2-n}$	$r=4$	2	3	5	6	10
f_5	2^{2-n}	$\leq 2^{4-n}$	$r=6$	4	5	7	9	15
f_7	$\leq 3 \cdot 2^{1-n}$	$\leq 3 \cdot 2^{2-n}$	$r=8$	5	7	11	12	20

4.2.2 Differential and Linear Cryptanalysis of Flurry and Curry

In this section we show how to compute upper bounds of MLCP and the MDCP of ciphers of the Flurry and Curry family. From these bounds we can deduce the number of rounds required to make an instance practically secure against differential and linear cryptanalysis.

The maximum differential probability of a function $f : \mathbb{F} \to \mathbb{F}$ can be calculated from δ as $p(f) = \frac{\delta}{\#\mathbb{F}}$ where δ is according to Definition 4.1.1. The maximum linear probability of a mapping $f : \mathbb{F} \to \mathbb{F}$ can be computed as

$$q(f) = \left(1 - \frac{2\mathcal{N}(f)}{\#\mathbb{F}}\right)^2$$

where $\mathcal{N}(f)$ is defined as in Section 4.1.3. From Theorema 4.2.4 it follows directly that for Flurry and Curry the MDCP is bounded by $p(f)^M$ while the MLCP is bounded by $q(f)^M$, where M is the miminum number of active S-Boxes. A lower bound for M is given in Lemma 4.2.5. Table 4.2 contains $p(f)$, $q(f)$, and M for the set of selected parameters of Flurry and Curry.

According to Knudsen [43], a block cipher with dependent round keys is practically secure against differential and linear cryptanalysis if the MLCP and the MDCP is too low for an attack to work under the assumption of independent round keys. Note however that for both r-round Feistel and r-round SPN ciphers, we need to consider the MLCP and MDCP of $r-2$ rounds because of attacks that guess bits of the first and the last round key, so-called 2R attacks.

4.2.3 Interpolation Attacks

Jakobsen and Knudsen presented interpolation attacks in [40] as a counterpoint to the growing trend of using algebraic S-Boxes such as those proposed

by Nyberg [49]. In fact, interpolation attacks can be seen as the first algebraic attacks on block ciphers. The underlying intuition of this attack is that the relationship between plaintext and ciphertext can be expressed as a tuple of polynomial expressions. If the degree of these polynomials is low enough, the coefficients of the polynomials can be interpolated from a number of plaintext/ciphertext pairs. A key–dependent equivalent of the encryption or the decryption algorithm has then been determined. In [40] upper bounds on the number of required pairs for known-plaintext interpolation attacks for selected examples are given. In general this number increases exponentially with the degree of the polynomial function describing the S-Box, the number of rounds and the number of elements in the internal state, while for the attacks we present in the next section it remains a constant quantity.

Courtois later improved on the work of Jakobsen and Knudsen and introduced an attack called General Linear Cryptanalysis [23]. In the same paper he also gives several examples of insecure ciphers based on inversion based S-Boxes that resist differential and linear cryptanalysis. His approach and his goals are quite different from ours however.

Flurry and Curry quite naturally are susceptible to interpolation attacks – their clean structure and the monomial S-boxes make them textbook examples. As a matter of fact, the cipher $\mathcal{PURE}$ presented in the original article is identical to the 64-bit cipher $\textsc{Flurry}(32, 1, r, f_3, I_1)$ sans key scheduling.

4.3 Attacks Using Gröbner Bases

In the following we describe Gröbner basis attacks on Flurry and Curry. These attacks are based on Algorithm 3 given in section 2.3.2. Since no theoretical works estimating the performance of Gröbner basis algorithms in the case of polynomial systems for block ciphers are currently known, we carried out experiments to study the resistance of our ciphers against Gröbner basis attacks. Results of these experiments are presented and analyzed in section 4.3.2. Then we show how to obtain a Gröbner basis w.r.t. a total-degree term order for Flurry and Curry with polynomial S-boxes by linear operations. Finally, a theoretical upper bounds for the time and space complexity of Gröbner basis attacks on such ciphers are given.

4.3.1 Key Recovery Using Gröbner Bases

The Gröbner basis attacks presented here work under the assumption that a small number of plaintext/ciphertext pairs are known. To determine the

secret key of a cipher Algorithm 3 is used. Suppose one cipher E from the FLURRY or CURRY family is fixed, K is a secret key, and $\Omega = \{(P, E_K(C))\}$ is a set of known plaintext/ciphertext pairs. The variable ordering should be such that the key variables of the first round are the least elements. Then an Gröbner basis attack on E works as follows:

1. Set up a polynomial system $\mathcal{P} = \{p_i = 0\}$ for the cipher as described in Section 4.1.4. The system $\mathcal{P}$ consists of both cipher and key schedule equations.

2. Request a pair $((P_1, \dots P_t), (C_1, \dots, C_t)) \in \Omega$. This gives rise to the following additional system of linear equations $\mathcal{G} = \{g_i = 0\}$:

$$\begin{aligned} x_1^{(0)} + P_1 &= 0 & x_1^{(r)} + C_1 &= 0 \\ &\vdots & &\vdots \\ x_t^{(0)} + P_t &= 0 & x_t^{(r)} + C_t &= 0 \end{aligned}$$

   where $x_1^{(0)}, \dots, x_t^{(0)}$ are the plaintext variables, and $x_1^{(r)}, \dots, x_t^{(r)}$ are the ciphertext variables. For FLURRY $t = 2 \cdot m$, and for CURRY $t = m^2$. Put $\mathbb{S} = \mathcal{P} \cup \mathcal{G}$.

3. Solve the system $\mathbb{S}$ in $\mathbb{F}$ using Algorithm 3. Since for a cipher with the inversion S-box the polynomial system does not hold with probability one, it is possible that in this case the result is $\mathcal{V}_\mathbb{S} = \emptyset$.

4. If $\mathcal{V}_\mathbb{S} = \emptyset$ go to Step 2, otherwise proceed.

5. Try all elements $k \in \mathcal{V}_\mathbb{S}$ as key candidates using other known pairs. If k does not encrypt P' to C' for some $(P', C') \in \Omega$, remove k from $\mathcal{V}_\mathbb{S}$, otherwise retain.

6. If $\mathcal{V}_\mathbb{S}$ contains more than one element, it is necessary to obtain additional plaintext/ciphertext pairs and repeat the previous step.

7. Terminate

For the system $\mathbb{S}$ of equations induced by FLURRY and CURRY with monomial S-boxes and any pair $(P, E_K(P))$, we always have $K \in \mathcal{V}_\mathbb{S}$. In the case the inversion S-box is used, the probability that $K \notin \mathcal{V}_\mathbb{S}$ is approximated by

$$1 - \left(\frac{2^n - 1}{2^n}\right)^q \approx 1 - e^{-q/2^n},$$

where $q = mr$ for the Flurry(n, m, r, f_{-1}, D_m), and $q = m^2 r$ for the Curry(n, m, r, f_{-1}, D_m). If $q \ll 2^n$, this probability is close to 0.

Further, the number of solutions of $\mathbb{S}$ in $\mathbb{F}$ is equivalent to the number of distinct keys encrypting P to C. The probability that $\#\mathcal{V}_{\mathbb{S}} = 1$ can be estimated as

$$\left(1 - \frac{1}{2^N}\right)^{2^N - 1} \approx \frac{1}{e},$$

where N is the bit length of a ciphertext and a secret key. Our experimental results confirm this estimation. Thus in about 36.8% of all cases, only one known plaintext/ciphertext pair is needed to recover the full secret key. In almost all other cases, the secret key can be determined, if a second pair is known. Indeed, the probability that there exists $K' \neq K$ such that $E_K(P) = E_{K'}(P)$ and $E_K(P') = E_{K'}(P')$ for random $P, P' \neq P$ is approximately equal to

$$1 - \left(1 - \frac{1}{2^N \cdot (2^N - 1)}\right)^{2^N - 1} \approx 1 - e^{-1/2^N}.$$

Let $\mathfrak{I}$ be the ideal generated by the set of polynomials $\mathcal{L} = (\bigcup_i \{p_i\}) \cup (\bigcup_i \{g_i\})$. We call this ideal the *key recovery ideal*. Algorithm 3 used in step 3 consists of DRL Gröbner basis computation, Gröbner basis conversion and computing zeroes of univariate polynomials over $\mathbb{F}$. Since in our case

$$2^n \gg \max\{\deg(f) \colon f \in \mathcal{L}\} = \begin{cases} d, & \text{if } E \text{ uses } f_d \text{ as S-box,} \\ 2, & \text{if } E \text{ uses } f_{-1} \text{ as S-box,} \end{cases}$$

the field equations are not added to $\mathbb{S}$. Instead of this, by solving of univariate polynomials only zeroes in $\mathbb{F}$ are selected. By Theorem 2.2.4 the run time of the FGLM algorithm for Gröbner basis conversion depends on $\dim_{\mathbb{F}}(R/\mathfrak{I})$. In our case we have

$$\dim_{\mathbb{F}}(R/\mathfrak{I}) \geq \overline{\mathcal{V}}_{\mathbb{S}},$$

and in general we can expect $\overline{\mathcal{V}}_{\mathbb{S}}$ to have a lot more elements than $\mathcal{V}_{\mathbb{S}}$. The best algorithm for factoring univariate polynomials is due to Kaltofen and Shoup [41] and has a complexity of $O(d_p^{1.815} n)$ field operations, where d_p is the degree of the polynomial. This degree is bounded above by

$$\min(2^n - 1, \dim_{\mathbb{F}}(R/\mathfrak{I})).$$

4.3.2 Experimental Results

We have performed experiments to analyze the resistance of Flurry and Curry using the computer algebra system Magma [53], version 2.11-8, on

an AMD Athlon 64 3200+ equipped with 1024 Megabytes of RAM running Linux. MAGMA implements Faugére's F4 algorithm [31] and is widely considered the best publicly available tool for computing Gröbner bases. We have chosen n and m such that the ciphers evaluated are 128-bit block ciphers.

Table 4.3 lists a number of instantiations of FLURRY and CURRY ciphers for which we were able to successfully recover the secret key; the 6, 8 and 10 round FLURRY ciphers are resistant to linear and differential cryptanalysis. We see that ciphers with inversion-based S-boxes are easier to break than ciphers which use a monomial S-box, even if the monomial is of very low degree. Furthermore we were unable to determine an a priori indicator for selecting the most efficient Gröbner basis conversion algorithm – in some cases FGLM was faster, in other cases the Gröbner walk; the same holds for the memory consumption.

4.3.3 Gröbner Bases without Polynomial Reductions

By solving a system of polynomial equations using Algorithm 3 the first step is to compute a DRL Gröbner basis of this system. In this section we show that for the key recovery ideal of FLURRY and CURRY with monomial S-boxes a DRL Gröbner basis w.r.t. a suitable term order can be obtained by applying linear operations only. To make this linear transformation easier to describe we use a vectorial representation for FLURRY and a matrix representation for CURRY.

First we prove the following lemma.

Lemma 4.3.1. *Let $R = \mathbb{F}[x_1, \ldots, x_n]$ be a polynomial ring with a term order. Suppose $G = \{g_1, \ldots, g_n\}$ is a set of polynomials such that $HT(g_i) = x_i^{d_i}$ for all $1 \leq i \leq n$ and some $d_i \geq 1$. Then G is a Gröbner basis, and $\dim_{\mathbb{F}}(R/\langle G\rangle) = \prod_{i=1}^{n} d_i$.*

Proof. Obviously, all head terms of G are pairwise coprime. By Theorem 2.2.3, G is a Gröbner basis. According to Theorem 2.2.5,

$$\dim(R/\langle G\rangle) = \#\left\{t \in \mathcal{T} \colon \mathrm{HT}(g) \nmid t \text{ for all } g \in G\right\}.$$

For any $t = x_1^{e_1} \ldots x_n^{e_n} \in \mathcal{T}$ and $1 \leq i \leq n$, we have $\mathrm{HT}(g_i) \nmid t$ iff $e_i < d_i$. Thus we get

$$\dim(R/\langle G\rangle) = \#\left\{x_1^{e_1} \ldots x_n^{e_n} \in \mathcal{T} \colon e_i < d_i \text{ for all } 1 \leq i \leq n\right\} = \prod_{i=1}^{n} d_i.$$

□

Table 4.3: Gröbner basis attacks on Flurry and Curry: Experimental results obtained with Magma

cipher	conversion	CPU time	memory used
$\text{Flurry}(64, 1, 4, f_{-1}, I_1)$	Walk	0.011 s	3.48 MBytes
$\text{Flurry}(64, 1, 4, f_{-1}, I_1)$	FGLM	0.011 s	3.48 MBytes
$\text{Flurry}(64, 1, 4, f_3, I_1)$	Walk	0.04 s	3.48 MBytes
$\text{Flurry}(64, 1, 4, f_3, I_1)$	FGLM	0.029 s	3.58 MBytes
$\text{Flurry}(64, 1, 4, f_5, I_1)$	Walk	1.28 s	3.97 MBytes
$\text{Flurry}(64, 1, 4, f_5, I_1)$	FGLM	2.3 s	6.36 MBytes
$\text{Flurry}(64, 1, 4, f_7, I_1)$	Walk	13.61 s	6.22 MBytes
$\text{Flurry}(64, 1, 4, f_7, I_1)$	FGLM	82.62 s	33.4 MBytes
$\text{Flurry}(64, 1, 6, f_{-1}, I_1)$	Walk	0.15 s	3.58 MBytes
$\text{Flurry}(64, 1, 6, f_{-1}, I_1)$	FGLM	0.059 s	3.58 MBytes
$\text{Flurry}(64, 1, 6, f_3, I_1)$	Walk	59.91 s	10.63 MBytes
$\text{Flurry}(64, 1, 6, f_3, I_1)$	FGLM	145.08 s	193.24 MBytes
$\text{Flurry}(64, 1, 8, f_{-1}, I_1)$	Walk	3.43 s	4.51 MBytes
$\text{Flurry}(64, 1, 8, f_{-1}, I_1)$	FGLM	1.46 s	4.46 MBytes
$\text{Flurry}(64, 1, 10, f_{-1}, I_1)$	Walk	115.44 s	14.74 MBytes
$\text{Flurry}(64, 1, 10, f_{-1}, I_1)$	FGLM	60.61 s	12.39 MBytes
$\text{Flurry}(64, 1, 12, f_{-1}, I_1)$	Walk	4194.28 s	99.97 MBytes
$\text{Flurry}(64, 1, 12, f_{-1}, I_1)$	FGLM	2064 s	142.90 MBytes
$\text{Flurry}(32, 2, 4, f_{-1}, D_2)$	Walk	216.53 s	25.58 MBytes
$\text{Flurry}(32, 2, 4, f_{-1}, D_2)$	FGLM	65.78 s	41.62 MBytes
$\text{Flurry}(16, 4, 2, f_{-1}, D_4)$	Walk	264 s	37.13 MBytes
$\text{Flurry}(16, 4, 2, f_{-1}, D_4)$	FGLM	26.119 s	18.56 MBytes
$\text{Curry}(32, 2, 3, f_{-1}, D_2)$	Walk	1750.87 sec	138.77 MBytes
$\text{Curry}(32, 2, 3, f_{-1}, D_2)$	FGLM	3676.26 sec	107.54 MBytes

The key idea of our method is to construct polynomial sets that describe the key recovery problem for Flurry and Curry and satisfy the condition of the previous lemma.

- $\text{Curry}(n, m, r, f_d, D)$
  In this case, $\mathbb{S}$ in the matrix form is given by

$$\left(x_{i,j}^{(e)}\right) + D \cdot \left(\left(x_{i,j}^{(e-1)} + k_{i,j}^{(e)}\right)^d\right)^T = 0;\ \left(x_{i,j}^{(0)} + p_{i,j}\right) = 0;$$

$$\left(k_{i,j}^{(e+1)}\right) + D \cdot \left(k_{i,j}^{(e)}\right)^T + \left(\xi^{e+(i-1)m+j+1}\right) = 0;\ \left(x_{i,j}^{(r)} + k_{i,j}^{(r+1)} + c_{i,j}\right) = 0;$$

  with $1 \le i, j \le m,\ 1 \le e \le r$, and a known plaintext/ciphertext pair $P = (p_{i,j}),\ C = (c_{i,j}) \in \mathbb{F}^{m \times m}$. We see that for any arbitrary total

degree term order the head terms of all polynomials in this system are univariate. Indeed, for each polynomial of the eth round of the encryption, either a power of a state variable of the preceding round or a power of a key variable of the current round occur as head term, while all non-constant terms of linear equations are univariate. Some head terms however occur more than once. Multiplying the matrices of all rounds of the encryption by D^{-1}, we obtain

$$D^{-1} \cdot \left(x_{i,j}^{(e)}\right) + \left(\left(x_{i,j}^{(e-1)} + k_{i,j}^{(e)}\right)^d\right)^T = 0;\ \left(x_{i,j}^{(0)} + p_{i,j}\right) = 0;$$
$$\left(k_{i,j}^{(e+1)}\right) + D \cdot \left(k_{i,j}^{(e)}\right)^T + \left(\xi^{e+(i-1)m+j+1}\right) = 0;\ \left(x_{i,j}^{(r)} + k_{i,j}^{(r+1)} + c_{i,j}\right) = 0.$$

Denote the set of polynomials of this system by G. Let $\preceq$ be the DRL term order defined on the set of variables as follows. Put $\mathcal{X}_e = \{x_{i,j}^{(e)} : 1 \le i,j \le m\}$ and $\mathcal{K}_e = \{k_{i,j}^{(e)} : 1 \le i,j \le m\}$ for all e. Then,

$$\mathcal{X}_0 \prec \mathcal{K}_0 \prec \mathcal{K}_1 \prec \cdots \prec \mathcal{K}_{r+1} \prec \mathcal{X}_1 \prec \mathcal{X}_2 \prec \cdots \prec \mathcal{X}_r,$$

where $M_1 \prec M_2$ with $M_1, M_2 \subset \mathcal{T}$ means $t_1 \prec t_2$ for all $t_1 \in M_1$ and $t_2 \in M_2$; and for all e, set $x_{i_1,j_1}^{(e)} \prec x_{i_2,j_2}^{(e)}$ and $k_{i_1,j_1}^{(e)} \prec k_{i_2,j_2}^{(e)}$ iff $(i_1-1)m + j_1 < (i_2-1)m + j_2$, i.e.,

$$x_{1,1}^{(e)} \prec \cdots \prec x_{1,m}^{(e)} \prec x_{2,1}^{(e)} \prec \cdots \prec x_{m,m}^{(e)};$$
$$k_{1,1}^{(e)} \prec \cdots \prec k_{1,m}^{(e)} \prec k_{2,1}^{(e)} \prec \cdots \prec k_{m,m}^{(e)}.$$

It can easily be checked that all head terms of G w.r.t. this term order are pairwise coprime. Actually,

$$\begin{aligned}\mathrm{HT}(G) = &\left\{x_{i,j}^{(0)}, x_{i,j}^{(r)} : 1 \le i,j \le m\right\} \bigcup \\ &\bigcup \left\{\left(k_{i,j}^{(0)}\right)^d, \left(x_{i,j}^{(e)}\right)^d : 1 \le i,j \le m,\ 1 \le e \le r-1\right\} \bigcup \\ &\bigcup \left\{k_{i,j}^{(e)} : 1 \le i,j \le m,\ 1 \le e \le r\right\},\end{aligned}$$

and no two polynomials of G have an identical head terms. Thus, by Lemma 4.3.1, we have constructed a Gröbner basis.

- Flurry(n, m, r, f, D)
  In this case the first step is the same as above. After multiplying the

vectors of all rounds of the encryption by D^{-1}, we have

$$X_{0,L} + P_L = 0; \qquad X_{0,R} + P_R = 0;$$
$$X_{r,L} + C_L = 0; \qquad X_{r,R} + C_R = 0;$$
$$D^{-1} \cdot \begin{pmatrix} x_{m+1}^{(e)} + x_1^{(e-1)} \\ \dots \\ x_{2m}^{(e)} + x_m^{(e-1)} \end{pmatrix} + \begin{pmatrix} \left(x_{m+1}^{(e-1)} + k_1^{(e)}\right)^d \\ \dots \\ \left(x_{2m}^{(e-1)} + k_m^{(e)}\right)^d \end{pmatrix} = 0$$
$$X_{e,L} + X_{e-1,R} = 0; \qquad K_{e+2} + D \cdot K_{e+1} + K_{e+2} + V = 0;$$

where $X_{j,L} = \left(x_1^{(j)}, \dots, x_m^{(j)}\right)^T$, $X_{j,R} = \left(x_{m+1}^{(j)}, \dots, x_{2m}^{(j)}\right)^T$ with $0 \leq j \leq r$, $K_1, \dots, K_{r+2}$ are the round key variables in the vector form, and $P = (P_L, P_R)$, $C = (C_L, C_R) \in \mathbb{F}^{2 \cdot m}$ is a known plaintext/ciphertext pair. Since an additional key addition is performed on both halves of the final state of the cipher, equations of the two last rounds look slightly different:

$$D^{-1} \cdot \begin{pmatrix} x_{m+1}^{(r-1)} + x_1^{(r-2)} + k_1^{(r+1)} \\ \dots \\ x_{2m}^{(r-1)} + x_m^{(r-2)} + k_1^{(r+1)} \end{pmatrix} + \begin{pmatrix} \left(x_{m+1}^{(r-2)} + k_1^{(r-1)}\right)^d \\ \dots \\ \left(x_{2m}^{(r-2)} + k_m^{(r-1)}\right)^d \end{pmatrix} = 0,$$
$$D^{-1} \cdot \begin{pmatrix} x_{m+1}^{(r)} + x_1^{(r-1)} + k_1^{(r+2)} \\ \dots \\ x_{2m}^{(r)} + x_m^{(r-1)} + k_1^{(r+2)} \end{pmatrix} + \begin{pmatrix} \left(x_{m+1}^{(r-1)} + k_1^{(r)}\right)^d \\ \dots \\ \left(x_{2m}^{(r-1)} + k_m^{(r)}\right)^d \end{pmatrix} = 0,$$
$$X_{r-1,L} + X_{r-2,R} = 0, \qquad X_{r,L} + X_{r-1,R} = 0.$$

Let $\preceq$ be the DRL term order defined on the set of variables as follows:

$$\mathcal{X}_{0,L} \prec \mathcal{X}_{0,R} \prec \mathcal{X}_{r,L} \prec \mathcal{X}_{r,R} \prec \mathcal{X}_{r-1,R} \prec \mathcal{K}_0 \prec \mathcal{K}_r \prec$$
$$\mathcal{K}_1 \prec \dots \prec \mathcal{K}_{r-1} \prec \mathcal{K}_{r+1} \prec \mathcal{K}_{r+2} \prec \mathcal{X}_{1,L} \prec \dots \prec \mathcal{X}_{r-1,R},$$

where

$$\mathcal{X}_{e,L} = \{x_i^{(e)} \colon 1 \leq i \leq m\},$$
$$\mathcal{X}_{e,R} = \{x_{i+m}^{(e)} \colon 1 \leq i \leq m\},$$
$$\mathcal{K}_e = \{k_i^{(e)} \colon 1 \leq i \leq m\}.$$

Also, set $x_i^{(e)} \prec x_j^{(e)}$ and $k_i^{(e)} \prec k_j^{(e)}$ iff $i < j$ for all e, i.e.,

$$x_1^{(e)} \prec \dots \prec x_m^{(e)} \prec x_{m+1}^{(e)} \prec \dots \prec x_{2m}^{(e)} \text{ and } k_1^{(e)} \prec \dots \prec k_m^{(e)}.$$

However, not all head terms of the above system now are pairwise coprime. Indeed, the nonlinear polynomials of the first and the last round have powers of key variables as head terms. These key variables are of the first and the last round respectively. For the first round this poses no problem. But for the last round the key schedule polynomials that produce the last round key have the same head terms. To obtain pairwise coprime head terms within each and across rounds, we therefore need to rewrite the key schedule equations. First, we express all round keys as a linear combination of the first two round keys. Let $M_1 = I$ be the $m \times m$ identity matrix, $M_2 = D$, and $V_3 = v_3$. For all $e \geq 3$, put $M_e = D \cdot M_{e-1} + M_{e-2}$ and $V_{e+1} = D \cdot V_e + v_e$. Then we have

$$K_e = M_{e-1} \cdot K_2 + M_{e-2} \cdot K_1 + V_e$$

with $3 \leq e \leq r+2$. Further, we write the second round key as a linear combination of the first and the last round key.

$$K_2 = M_{r-1}^{-1} \cdot (K_r + M_{r-2} \cdot K_1 + V_r)\,.$$

This results in all head terms being pairwise prime. We see that this work if M_{r-1} is invertible. We have checked by direct calculation that, this condition holds for D_1 and D_2 selected in Section 4.1.3 in all cases with $r \leq 20$, and for D_4, whenever $r \in \{3, 4, 6, 7, 9, 10, 12, 13, \dots\}$.

Denote the set of polynomials of the obtained system with the modified key schedule equations by G. By Lemma 4.3.1, G is a Gröbner basis. Moreover, $\mathrm{HT}(G) = T_1 \cup T_d$ with

$$T_1 = \mathcal{X}_{0,L} \cup \mathcal{X}_{0,R} \cup \mathcal{X}_{r,L} \cup \mathcal{X}_{r,R} \cup \mathcal{X}_{r-1,R} \cup \mathcal{K}_{r+1} \cup \mathcal{K}_{r+2} \cup_{i=1}^{r-1} (\mathcal{K}_i \cup \mathcal{X}_{i,L}),$$
$$T_d = \left\{t^d \colon t \in \mathcal{K}_0 \cup \mathcal{K}_r \cup_{e=2}^{r-2} \mathcal{X}_{e,R}\right\}.$$

We have shown how to obtain DRL Gröbner bases for a large subset of Flurry and Curry. For the described method, the S-box of a cipher does not need be monomial. Actually, the method works whenever the S-box is represented as a polynomial in the input variable. Furthermore, in this case $\dim_{\mathbb{F}}(R/\mathfrak{I})$ can be easily computed, where $\mathfrak{I}$ is the key recovery ideal for the cipher.

Proposition 4.3.2. *Let $\mathfrak{I}$ be the key recovery ideal of an instantiation of either a* Flurry *or a* Curry *with a polynomial S-box given by $f \in \mathbb{F}[x]$, G the polynomial set constructed for $\mathfrak{I}$ as described above, and $\preceq$ the selected term order. Then, G is a Gröbner basis of $\mathfrak{I}$, and the following holds:*

Table 4.4: Upper bounds on the complexity of breaking 128-bit Flurry and Curry ciphers with FGLM

cipher	n	$\dim(R/\mathfrak{I})$	# of operations	memory required (bytes)
$\text{Flurry}(32,2,4,f_3,D_2)$	8	$3^8 \approx 2^{12.68}$	$2^{41.0}$	$2^{30.4}$
$\text{Flurry}(32,2,4,f_5,D_2)$	8	$5^8 \approx 2^{18.58}$	$2^{58.7}$	$2^{42.2}$
$\text{Flurry}(32,2,4,f_7,D_2)$	8	$7^8 \approx 2^{22.46}$	$2^{70.4}$	$2^{49.9}$
$\text{Flurry}(32,2,6,f_3,D_2)$	12	$3^{12} \approx 2^{19.02}$	$2^{60.6}$	$2^{43.2}$
$\text{Flurry}(32,2,6,f_5,D_2)$	12	$5^{12} \approx 2^{27.86}$	$2^{87.2}$	$2^{61.3}$
$\text{Flurry}(32,2,6,f_7,D_2)$	12	$7^{12} \approx 2^{33.69}$	$2^{104.7}$	$2^{73.0}$
$\text{Flurry}(32,2,8,f_3,D_2)$	16	$3^{16} \approx 2^{25.36}$	$2^{80.0}$	$2^{56.7}$
$\text{Flurry}(32,2,8,f_5,D_2)$	16	$5^{16} \approx 2^{37.15}$	$2^{115.5}$	$2^{80.3}$
$\text{Flurry}(32,2,8,f_7,D_2)$	16	$7^{16} \approx 2^{44.92}$	$2^{138.8}$	$2^{95.8}$
$\text{Flurry}(16,4,4,f_3,D_2)$	16	$3^{16} \approx 2^{25.36}$	$2^{80.0}$	$2^{55.7}$
$\text{Flurry}(16,4,4,f_5,D_2)$	16	$5^{16} \approx 2^{37.15}$	$2^{115.5}$	$2^{79.3}$
$\text{Flurry}(16,4,4,f_7,D_2)$	16	$7^{16} \approx 2^{44.92}$	$2^{138.8}$	$2^{94.8}$
$\text{Curry}(32,2,3,f_7,D_2)$	12	$7^{12} \approx 2^{33.69}$	$2^{104.6}$	$2^{73.0}$

1. $\dim_{\mathbb{F}}(R/\mathfrak{I}) = \deg(f)^{mr}$ *for* $\text{Flurry}(n,m,r,f,D)$.

2. $\dim_{\mathbb{F}}(R/\mathfrak{I}) = \deg(f)^{rm^2}$ *for* $\text{Curry}(n,m,r,f,D)$.

It is clear that there is no need to make all computation every time, and G can be directly written for $\mathfrak{I}$. Thus we have reduced the key recovery problem for Flurry and Curry with polynomial S-boxes to a Gröbner basis conversion problem. Moreover, by Theorem 2.2.4 the complexity of the FGLM algorithm hinges on the number of variables and $\dim_{\mathbb{F}}(R/\mathfrak{I})$. Both of these parameters are known in our case. Therefore, we can estimate the maximum resistance of Flurry and Curry ciphers with polynomial S-Boxes against Gröbner basis attacks (see Table 4.4). We conjecture the constant factor in the estimate given in Theorem 2.2.4 to be approximately one cipher operation. Note that for the Curry cipher we need to use a bijective S-Box in the round function; the lowest degree S-Box function that is bijective is f_7.

The method described here does not work however for Flurry and Curry instances with inversion S-Boxes, as the head terms in these cases are never univariate. One example of a Gröbner basis for Flurry using f_3 as S-box is given in Appendix B.

Chapter 5

A zero-dimensional Gröbner basis for AES–128

In the previous chapter it was shown that for some ciphers a DRL zero-dimensional Gröbner basis can be calculated by hand. This reduces the key recovery problem for these block ciphers to a Gröbner basis conversion. In this chapter we apply the similar method to AES-128. First using a polynomial representation of the AES S-box over $\mathbb{F} = \mathrm{GF}(2^8)$ we show how a DRL zero-dimensional Gröbner basis can be derived in this case. Then we study the cryptanalytic significance of this Gröbner basis.

5.1 Construction of the DRL Gröbner basis

First let us remember the basic idea of the method used in the previous chapter to derive a DRL Gröbner basis without polynomial reduction. Let the S-box of a cipher be given by some polynomial in the input, then the head term of this polynomial w.r.t. any total degree term order is a power of the input variable. Since all inputs of the S-boxes are different, the head terms of the corresponding polynomials are pairwise prime. However by the linear transformation of the cipher this polynomials are mixed. If inverting the linear transformation we obtain polynomials with pairwise prime head terms, then by Theorem 2.2.3 the set of this polynomials is a Gröbner basis.

Now let us consider AES-128. For our method we cannot use the algebraic representations in the form of a system of quadratic equations, since in this case the head terms of the non-linear polynomials are not univariate, and hence not pairwise prime. Thus we will construct a DRL Gröbner basis for AES using the polynomial representation given in Section 3.2.1.

The first step of the construction is the following. In order to have poly-

nomials with pairwise prime head terms within all rounds of encryption, we rewrite system (3.3) of equations for the ith round ($1 \leq i \leq 9$) using the inverse matrix D^{-1}:

$$D^{-1} \cdot \begin{pmatrix} x_{i,0}+k_{i,0} & x_{i,4}+k_{i,4} & x_{i,8}+k_{i,8} & x_{i,12}+k_{i,12} \\ x_{i,1}+k_{i,1} & x_{i,5}+k_{i,5} & x_{i,9}+k_{i,9} & x_{i,13}+k_{i,13} \\ x_{i,2}+k_{i,2} & x_{i,6}+k_{i,6} & x_{i,10}+k_{i,10} & x_{i,14}+k_{i,14} \\ x_{i,3}+k_{i,3} & x_{i,7}+k_{i,7} & x_{i,11}+k_{i,11} & x_{i,15}+k_{i,15} \end{pmatrix} + \\ + \begin{pmatrix} S(x_{i-1,0}) & S(x_{i-1,4}) & S(x_{i-1,8}) & S(x_{i-1,12}) \\ S(x_{i-1,5}) & S(x_{i-1,9}) & S(x_{i-1,13}) & S(x_{i-1,1}) \\ S(x_{i-1,10}) & S(x_{i-1,14}) & S(x_{i-1,2}) & S(x_{i-1,6}) \\ S(x_{i-1,15}) & S(x_{i-1,3}) & S(x_{i-1,7}) & S(x_{i-1,11}) \end{pmatrix} = 0,$$

In the last round the `MixColumns` transformation is omitted, and each equation has the terms of one S-box polynomial:

$$\begin{gathered} S(x_{9,0}) + x_{10,0} + k_{10,0} = 0 \\ S(x_{9,1}) + x_{10,9} + k_{10,9} = 0 \\ \dots \\ S(x_{9,15}) + x_{10,4} + k_{10,4} = 0 \end{gathered}$$

It is easy to see that for the polynomials of these systems the set of the head terms w.r.t. any total degree term order is

$$\{x_{i,j}^{254} \colon 0 \leq i \leq 9,\ 0 \leq j \leq 15\},$$

and no two polynomial have the same head term.

Further, in the polynomial of a ciphertext equation

$$x_{10,j} + c_j = 0$$

with $0 \leq j \leq 15$ the head term is $x_{10,j}$, and it has no common non-trivial divisor with any other head term. The terms $x_{0,j}$ and $k_{0,j}$ of a plaintext polynomial

$$x_{0,j} + k_{0,j} + p_j$$

have the same degree. We choose a term order such that $x_{0,j} < k_{0,j}$ for any $j = \overline{0,15}$.

Finally let us consider the key schedule equations:

$$\begin{pmatrix} k_{i,0} \\ k_{i,1} \\ k_{i,2} \\ k_{i,3} \\ k_{i,4} \\ \vdots \\ k_{i,15} \end{pmatrix} = \begin{pmatrix} k_{i-1,0} + S(k_{i-1,13}) & +\xi^{i-1} \\ k_{i-1,1} + S(k_{i-1,14}) & \\ k_{i-1,2} + S(k_{i-1,15}) & \\ k_{i-1,3} + S(k_{i-1,12}) & \\ k_{i-1,4} + k_{i,0} & \\ \vdots & \\ k_{i-1,15} + k_{i,11} & \end{pmatrix} \tag{5.1}$$

where $1 \leq i \leq 10$. We see that the condition all head terms are pairwise prime does not hold any more. For example, the head term of the polynomial $S(k_{0,13}) + k_{1,0} + k_{0,0} + \mathtt{01}$ is $k_{0,13}^{254}$, and it is divisible by the head term of $x_{0,13} + k_{0,13} + p_{13}$. Using the polynomial S' for the inverse S-box we rewrite the key schedule equations as:

$$\begin{pmatrix} S'(k_{0,j} + k_{0,j-1} + \xi^{j-1}) \\ S'(k_{1,j} + k_{1,j-1}) \\ S'(k_{2,j} + k_{2,j-1}) \\ S'(k_{3,j} + k_{3,j-1}) \\ k_{4,j} + k_{4,j-1} \\ \vdots \\ k_{15,j} + k_{15,j-1} \end{pmatrix} + \begin{pmatrix} k_{13,j-1} \\ k_{14,j-1} \\ k_{15,j-1} \\ k_{12,j-1} \\ k_{0,j} \\ \vdots \\ k_{11,j} \end{pmatrix} = 0 \tag{5.2}$$

If the fixed term order is such that

$$k_{i,15} > k_{i,14} > \cdots > k_{i,0} > k_{i-1,15} > \cdots > k_{i-1,1} > k_{i-1,0}$$

with $1 \leq i \leq 10$, then the set of the head terms for the key schedule polynomials is

$$\{k_{i,j}^{254}, k_{i,h} \colon 1 \leq i \leq 10,\ 0 \leq j \leq 3,\ \text{and } 4 \leq h \leq 15\}.$$

Thus by using an appropriate term order the polynomials of the modified system for AES have pairwise prime head terms. For example, the DRL term order with the following order of the variables satisfies this condition:

$$\underbrace{x_{0,0} < \ldots < x_{0,15}}_{\text{plaintext variables}} < \underbrace{k_{0,0} < \ldots < k_{0,15}}_{\text{initial key variables}} <$$

$$\underbrace{k_{1,0} < \ldots < k_{1,15}}_{\text{first round key variables}} < \cdots < \underbrace{k_{10,0} < \ldots < k_{10,15}}_{\text{last round key variables}} <$$

$$\underbrace{x_{1,0} < \ldots < x_{1,15}}_{\text{first round internal state variables}} < \cdots < \underbrace{x_{9,0} < \ldots < x_{9,15}}_{\text{9th round internal state variables}} <$$

$$\underbrace{x_{10,0} < \ldots < x_{10,15}}_{\text{ciphertext variables}}$$

Denote the set of the obtained polynomials by $\mathcal{A}$, and this DRL term order by $<_{\mathcal{A}}$. Then by Theorem 2.2.3, $\mathcal{A}$ is a Gröbner basis relative to $<_{\mathcal{A}}$. Moreover, we see that $\mathcal{A}$ fulfill the condition of Theorem 2.3.2, and hence the ideal $\langle\mathcal{A}\rangle$ is zero-dimensional.

5.2 Exploiting the Gröbner basis

In this section we study the cryptanalytic significance of the Gröbner basis $\mathcal{A}$ constructed above. First we estimate the complexity of a conversion of $\mathcal{A}$ to a Lex Gröbner basis using the FGLM algorithm, then we find an invariant under the elimination of variables and explain why $\mathcal{A}$ cannot be used to guess parts of the round key.

5.2.1 Complexity of Gröbner basis Conversions

According to Theorem 2.2.4 the complexity of a conversion of $\mathcal{A}$ to a Lex Gröbner basis using the FGLM algorithm depends on the number of variables and the dimension of the $\mathbb{F}$-vector space $R/\langle\mathcal{A}\rangle$.

The set $\mathcal{A}$ consists of 200 polynomials with degree 254 and 152 linear polynomials in 352 variables, $x_{i,j},\ k_{i,j}$ with $0 \leq i \leq 10$ and $0 \leq j \leq 15$. Let $R = \mathbb{F}[\mathcal{V}]$, where $\mathcal{V} = \{x_{i,j}, k_{i,j} \colon 0 \leq i \leq 10,\ 0 \leq j \leq 15\}$. By Lemma 4.3.1 the vector space dimension of $R/\langle\mathcal{A}\rangle$ is:

$$dim(R/\langle\mathcal{A}\rangle) = 254^{200} \approx 2^{1598}.$$

This number is very huge. Though in Theorem 2.2.4 an upper bound of the run time is given, there is no reason to expect that the FGLM algorithm is efficient in this case.

The number of variables can be reduced by elimination. However the vector space dimension of the ideal is invariant under the elimination of all variables except the last round key variables. To prove this we need the following proposition:

Proposition 5.2.1. *Let $\mathfrak{I}'$ be a zero-dimensional ideal of $R' = \mathbb{K}[x_1, \ldots, x_n]$, $\mathfrak{I}$ an ideal of $R = R[x_{n+1}]$ and $\mathfrak{I}' = \mathfrak{I} \cap R'$. Then $\dim R/\mathfrak{I} = \dim R'/\mathfrak{I}'$ iff there exists a polynomial $g \in R'$ such that $x_{n+1} + g \in \mathfrak{I}$.*

Proof. The vector space dimension of an ideal does not depend on a term ordering. Let us fix a lexicographical term ordering such that x_{n+1} is the greatest variable. Let $\mathrm{RT}(\mathfrak{I})$ and $\mathrm{RT}(\mathfrak{I}')$ be defined as follows:

$$\begin{aligned} \mathrm{RT}(\mathfrak{I}) &= \{t \in \mathcal{T}(R) : s \nmid t \text{ for all } s \in \mathrm{HT}(\mathfrak{I})\} \\ \mathrm{RT}(\mathfrak{I}') &= \{t \in \mathcal{T}(R') : s \nmid t \text{ for all } s \in \mathrm{HT}(\mathfrak{I}')\} \subset \mathrm{RT}(\mathfrak{I}) \end{aligned}$$

By Proposition 6.51 of [5], $\dim R/\mathfrak{I} = \#\mathrm{RT}(\mathfrak{I})$. Thus it is sufficient to prove that $\#\mathrm{RT}(\mathfrak{I}) = \#\mathrm{RT}(\mathfrak{I}')$. Since $x_{n+1} \nmid t$ for any $t \in \mathcal{T}(R')$, the equality $\mathrm{RT}(\mathfrak{I}) = \mathrm{RT}(\mathfrak{I}')$ holds iff $x_{n+1} \in \mathrm{HT}(\mathfrak{I})$, i.e., there exists a polynomial $g \in R'$ such that $x_{n+1} + g \in \mathfrak{I}$.

□

Corollary 5.2.2. *Let* $R' = \mathbb{F}[k_{0,10}, \ldots, k_{15,10}]$, *and* $\langle \mathcal{A}' \rangle = \langle \mathcal{A} \rangle \cap R'$. *Then*

$$\dim R'/\langle \mathcal{A}' \rangle = \dim R/\langle \mathcal{A} \rangle = 254^{200}.$$

Proof. By induction using Proposition 5.2.1. □

So even eliminating all variables but the cipher key variables does not reduce the complexity of converting the Gröbner basis to a term order suitable for key recovery.

5.2.2 Ideal Membership Problem and Testing Keys

Reduction modulo a Gröbner basis is a simple way to verify the ideal membership of a polynomial (see Section 2.3.1). Since $\langle \mathcal{A} \rangle$ is a zero-dimensional ideal, it contains univariate polynomials for all key byte variables. These polynomials can be easily distinguish from others using the constructed Gröbner basis. Moreover, each of them obviously has a zero at $\mu_i \in \mathbb{F}$, where μ_i is the correct value of the corresponding key byte. However, the polynomial is more difficult than

$$k_i + \mu_i,$$

and it cannot be easily guessed. Indeed, the constructed polynomial system has solutions over the closure of the ground field, which means that we have to test for a polynomial $p = q \cdot \prod_j (k_i + C_j)^{t_j}$, where all $C_j \in \mathbb{F}$ are key byte candidates and q is product of polynomials that are irreducible over $\mathbb{F}$. In additional, the degree of p can be as large as $\dim(R/\langle \mathcal{A} \rangle)$. We see that the dimension of $R/\langle \mathcal{A} \rangle$ again plays an important role here: it equals the number of solutions over the closure of the ground field. As it was shown in the previous section, this number is obscenely large.

To eliminate all points of the variety that do not lie in $\mathbb{F}$, one can adjoin the set

$$\mathcal{F} = \{v^{256} + v : \quad \text{for all variables } v \in R\}$$

of all polynomials from the field equations to $\mathcal{A}$. Unfortunately, in this case we do not have a Gröbner basis anymore. What we have to do here is to compute the intersection of two varieties; this is usually achieved by computing the Gröbner basis of the sum of the corresponding ideals. We

have a Gröbner basis $\mathcal{A}$ describing AES and a second set of polynomials $\mathcal{F}$, which obviously forms a Gröbner basis relative to the same term order $<_{\mathcal{A}}$ too. It is however unclear how to exploit the Gröbner basis property of the input.

Finally, note that the results given in this chapter was first published in 2006 [16], but at the time of writing this thesis, a security implications of the above Gröbner basis for AES keeps to be unknown.

Chapter 6

Block Ciphers and Semi-Regular Sequences

In this chapter we analyze the semi-regularity of several polynomial representations for block ciphers. First we prove that the systems of equations for the FLURRY and CURRY ciphers with a polynomial S-box described in Chapter 4 as well as the Gröbner basis for the AES derived in Chapter 5 are semi-regular. Then we show that polynomial systems that are similar to the BES quadratic equations are not semi-regular. Finally we demonstrate that systems of equations over GF(2) for iterated block ciphers, for example the AES systems over GF(2), are not semi-regular over GF(2).

6.1 The Case of DRL Gröbner bases

Let us consider the DRL zero-dimensional Gröbner basis $\mathcal{A}$ for the AES described in Chapter 5. It has the following two properties. The number of polynomials is equal to the number of variables, and the head terms of these polynomials are pairwise prime and univariate. For polynomial systems with these properties we now prove the following statement:

Proposition 6.1.1. *Let $R = \mathbb{F}[x_1, \ldots, x_n]$ be a polynomial ring over $\mathbb{F}$. Suppose $G = \{g_1, \ldots, g_n\} \subset R$ is a set of polynomials such that $HT(g_i) = x_i^{d_i}$ for all $1 \leq i \leq n$; then G is semi-regular.*

Proof. Let $\tilde{g}_i = DF(g_i)$ be the degree form of g_i, where $1 \leq i \leq n$. Then for all i we have $\mathrm{HT}(\tilde{g}_i) = \mathrm{HT}(g_i) = x_i^{d_i}$. Let us show that the Hilbert series of the sequence $\tilde{G} = (\tilde{g}_1, \ldots, \tilde{g}_n)$ is

$$\left[\frac{\prod_{i=1}^{n}(1-z^{d_i})}{(1-z)^n}\right] = \prod_{i=1}^{n}\left(1 + z + \cdots + z^{d_i-1}\right).$$

Since the head terms of all polynomials are pairwise prime, $\{\tilde{g}_1, \ldots, \tilde{g}_n\}$ is a Gröbner basis, i.e., $\langle \mathrm{HT}(\tilde{g}_1), \ldots, \mathrm{HT}(\tilde{g}_n)\rangle = \mathrm{HT}(\langle \tilde{G}\rangle)$. Thus

$$E\big(\langle \tilde{G}\rangle\big) = T \setminus \mathrm{HT}\big(\langle \tilde{G}\rangle\big) = \left\{ \prod_{i=1}^{n} x_i^{a_i} : \text{for all } 0 \le a_i < d_i \right\}.$$

Consider the polynomial $h = \prod_{i=1}^{n} \left(1 + Y_i + \cdots + Y_i^{d_i - 1}\right) \in \mathbb{Q}[Y_1, \ldots, Y_n]$. Obviously, all coefficients of h is equal to 1, and

$$T(h) = \left\{ \prod_{i=1}^{n} Y_i^{a_i} : \text{for all } 0 \le a_i < d_i \right\}.$$

Therefore, we have $\#T_d(h) = \#E_d(\langle \tilde{G}\rangle)$ for all $d \ge 0$. It is easy to see that if we replace all Y_i by a new variable z in h, then for any d the coefficient at z^d of the polynomial $\bar{h}(z) = h(z, \ldots, z) \in \mathbb{Q}[z]$ is equal to the number of all terms of h which degree is d, i.e.,

$$\bar{h}(z) = \sum_{d \ge 0} \#T_d(h) z^d = \sum_{d \ge 0} \#E_d\big(\langle \tilde{G}\rangle\big) z^d$$

Thus $\bar{h}$ is the Hilbert series of $\tilde{G}$, and by Theorem 2.4.2, $\tilde{G}$ and G are semi-regular. □

Corollary 6.1.2. *The DRL Gröbner basis $\mathcal{A}$ for the* AES *is semi-regular.*

Corollary 6.1.3. *The systems of equations described in Section 4.1.4 for the* Flurry *and* Curry *ciphers with a polynomial f_d as S-box are semi-regular.*

Proof. The polynomials of any such system can be converted into a Gröbner basis, which satisfies the conditions of Proposition 6.1.1. Moreover, we see that both the initial set and the Gröbner basis have a same number of linear polynomials as well as polynomials of degree d. Since in Theorem 2.4.2 the form of the Hilbert series depends on the number of polynomials and their degree but not on the head terms, the initial set passes the criterion of semi-regularity. □

The set G in Proposition 6.1.1 is semi-regular, and it can be checked using only the first Buchberger criterion without polynomial reductions that G is a Gröbner basis. Therefore a lower bound for computing Gröbner bases of a semi-regular sequence is found.

6.2 The Case of BES Equations

Now we consider systems of quadratic equations for Flurry and Curry with the S-box f_{-1} (Section 4.1.4) as well as BES equations (Section 3.2.3). These systems have the following common property. The degree form of any quadratic polynomial has the form $X \cdot Y$, where X is a linear combination of input variables and Y is a linear combination of output variables for some round. In order to show that these systems are not semi-regular, we first prove the following more general proposition:

Proposition 6.2.1. *Let $R = \mathbb{F}[x_1, \ldots, x_n]$ be a polynomial ring over $\mathbb{F}$, and $G = \{g_1, \ldots, g_m\} \subset R$ be a set of polynomials, where $m \geq n$. Suppose that for some k no polynomial $g_i \in G$ has the term $x_k^{\deg(g_i)}$; then G is not semi-regular.*

Proof. Let $\tilde{g}_i = DF(g_i)$ be the degree form of g_i, where $1 \leq i \leq m$. Prove that the ideal $\langle \tilde{G} \rangle = \langle \tilde{g}_1, \ldots, \tilde{g}_m \rangle$ is not zero-dimensional. By assumption we have $T(\tilde{G}) \cap \left\{ x_k^d \colon d \geq 0 \right\} = \emptyset$. Therefore,

$$\mathrm{HT}\big(\langle \tilde{G} \rangle\big) \subset \left\{ tT(\tilde{G}) \colon t \in \mathcal{T} \right\} \subset \mathcal{T} \setminus \left\{ x_k^d \colon d \geq 0 \right\},$$

i.e. there is no polynomial $f \in \langle \tilde{G} \rangle$ such that $\mathrm{HT}(f) = x_k^d$ for any $d \geq 0$. From Theorem 2.3.2 it follows that $\dim(\langle \tilde{G} \rangle) > 0$. By Corollary 2.4.4, $\tilde{G}$ is not semi-regular, and so is G. □

We see that the investigated systems do not satisfy the conditions of this proposition. Indeed, in all cases any variable of the corresponding polynomial ring occurs in some linear polynomial. However these cases can be reduced to Proposition 6.2.1 by applying the following statement:

Proposition 6.2.2. *Let $R = \mathbb{F}[x_1, \ldots, x_n, y_1, \ldots, y_m]$ and $R' = \mathbb{F}[x_1, \ldots, x_n]$ be two polynomial rings over $\mathbb{F}$. Suppose*

$$G = \{y_i + g_i(x_1, \ldots, x_n), f_j(x_1, \ldots, x_n, y_1, \ldots, y_m)\} \subset R$$

with $1 \leq i \leq m, 1 \leq j \leq l$, and $\deg(g_i) = 1$. In addition, put $G' = \{f'_1, \ldots, f'_l\}$, where $f'_j(x_1, \ldots, x_n) = f_j(x_1, \ldots, x_n, g_1, \ldots, g_m)$. Then we have if G is semi-regular in R, so is G' in R'.

Proof. For $1 \leq i \leq m, 1 \leq j \leq l$, by $\tilde{g}_i$ and $\tilde{f}_j$ denote the degree form of g_i and f_i respectively, and by $\mathfrak{I}$ denote the ideal $\langle y_1 + \tilde{g}_1, \ldots, y_m + \tilde{g}_m, \tilde{f}_1, \ldots, \tilde{f}_l \rangle$.

It is obvious that $\tilde{f}_j\,(x_1, \ldots, x_n, \tilde{g}_1, \ldots, \tilde{g}_m)$ equals either $DF(f'_j)$ or 0. From the second case it follows easily that $\tilde{f}_j \in \langle y_1 + \tilde{g}_1, \ldots, y_m + \tilde{g}_m \rangle$, and G

is not semi-regular. Therefore, we have $\tilde{f}_j(x_1, \ldots, x_n, \tilde{g}_1, \ldots, \tilde{g}_m) = DF(f'_j)$ for all possible j.

Let $\mathcal{T}$ and $\mathcal{T}'$ be the set of terms in R and R' respectively. Take a total degree term order $\preceq$ on $\mathcal{T}$ such that $y_i \succ x_j$ for $1 \leq i \leq m, 1 \leq j \leq n$. Let F be a Gröbner basis of $\mathfrak{I}' = \langle DF(f'_1), \ldots, DF(f'_l)\rangle$ w.r.t the restriction of $\preceq$ to $\mathcal{T}'$, then $(\bigcup\{y_i + \tilde{g}_i\})\bigcup F$ is a Gröbner basis of $\mathfrak{I}$ w.r.t $\preceq$. Therefore,

$$\mathrm{HT}(\mathfrak{I}) = \mathrm{HT}(\mathfrak{I}')\bigcup\{y_i t \colon 1 \leq i \leq m, t \in \mathcal{T}\}$$

and we have

$$E_d(\mathfrak{I}) = \mathcal{T}_d \setminus \mathrm{HT}_d(\mathfrak{I}) = E_d(\mathfrak{I}') = \mathcal{T}'_d \setminus \mathrm{HT}_d(\mathfrak{I}'),$$

i.e., the Hilbert series $h_R(\mathfrak{I})$ of $\mathfrak{I} \subset R$ is equal to the Hilbert series $h_{R'}(\mathfrak{I}')$ of $\mathfrak{I}' \subset R'$. Thus, we get

$$h_{R'}(\mathfrak{I}') = h_R(\mathfrak{I}) = \left[\frac{\prod_{i=1}^{m}(1-z)\prod_{i=1}^{l}(1-z^{d_i})}{(1-z)^{m+n}}\right] = \left[\frac{\prod_{i=1}^{l}(1-z^{d_i})}{(1-z)^{n}}\right],$$

where $d_i = \deg(\tilde{f}_i) = \deg(DF(f'_i))$. By Theorem 2.4.2, G' is semi-regular. □

Corollary 6.2.3. *The* BES *system* $\mathbb{S}_B$ *is not semi-regular.*

Proof. As stated in Section 3.2.3, $\mathbb{S}_B$ consists of (B) and (B'). If we eliminate the variables $x^{(e)}_{i,j}$ and $w^{(e)}_{9,j}$ in (B) using the linear equations, we obtain the following system:

$$\left.\begin{aligned} w^{(e)}_{0,j}\left(k^{(e)}_{0,j} + p^{(e)}_{j}\right) + 1 &= 0 \\ w^{(e)}_{i+1,j}\left(\sum \alpha_{s,t} w^{(t)}_{i,s} + k^{(e)}_{i,j}\right) + 1 &= 0 \\ \left(\sum \alpha'_{q,r}\left(k^{(r)}_{10,q} + c^{(r)}_{q}\right)\right)\left(\sum \alpha_{s,t} w^{(t)}_{9,s} + k^{(e)}_{9,j}\right) + 1 &= 0 \end{aligned}\right\} \begin{array}{l} i = 0, \ldots, 8 \\ j = 0, \ldots, 15 \\ e = 0, \ldots, 7 \end{array}$$

The polynomials of this system and the system (B') obviously satisfy the conditions of Proposition 6.2.1 in the new polynomial ring

$$R'_B = \mathbb{F}\left[w^{(e)}_{i,j}, k^{(e)}_{i,j}, k^{(e)}_{10,j}\right]$$

with $0 \leq i \leq 9,\ 0 \leq j \leq 15$, and $0 \leq e \leq 7$. Hence they do not generate a semi-regular sequence. By Proposition 6.2.2 we have $\mathbb{S}_B$ is also not semi-regular. □

Similarly, we can prove that:

Corollary 6.2.4. *The systems of quadratic equations for* FLURRY *and* CURRY *with the inversion S-box are not semi-regular.*

Now let us consider $\mathbb{S}_{EA}$, the system of equations that describes an AES encryption embedded in the BES as given in 3.2.3. This system consists of $\mathbb{S}_B$ and the conjugate equations

$$\begin{array}{lll} \left(x_{i,j}^{(e)}\right)^2 + x_{i,j}^{(e+1)} = 0 & \left(w_{i,j}^{(e)}\right)^2 + w_{i,j}^{(e+1)} = 0 & \left(k_{i,j}^{(e)}\right)^2 + k_{i,j}^{(e+1)} = 0 \\ \left(x_{i,j}^{(7)}\right)^2 + x_{i,j}^{(0)} = 0 & \left(w_{i,j}^{(7)}\right)^2 + w_{i,j}^{(0)} = 0 & \left(k_{i,j}^{(7)}\right)^2 + k_{i,j}^{(0)} = 0 \end{array}$$

In this case, the degree forms of the polynomials obviously generate a zero-dimensional ideal, and we cannot use proposition 6.2.1. However we see that $x_{i,j}^{(e)} \notin \left\langle \left(x_{i,j}^{(e)}\right)^2 \right\rangle$ and

$$x_{i,j}^{(e)} \left(x_{i,j}^{(e)} w_{i,j}^{(e)} \right) \in \left\langle \left(x_{i,j}^{(e)}\right)^2 \right\rangle$$

for any i, j, and e, i.e., there are non-trivial polynomial relations of degree 3. Thus this system is also not semi-regular.

In the same way, it can be proved that the embedded system described in [44] for the Feistel cipher SMS4 is not semi-regular.

6.3 Polynomial Representation of the AES over GF(2)

Here we analyze the AES representation in the form of a multivariate polynomial system of quadratic equations over GF(2). As described in Section 3.2.2, each polynomial in this system has a small number of variables. Actually, there are several variants of the representation. First one can vary S-box quadratic equations by linear operations as well as using only a part of them. Also, one can write the system with more intermediate variables and linear equations, or eliminate some variables via these linear equations. We begin with the following lemma.

Lemma 6.3.1. *Let $R_2 = GF(2)[\mathcal{X}]$. Suppose $G = \{f_1, \ldots, f_m\} \subset R_2$ and $G' = \{f'_1, \ldots, f'_m\} \subset R_2$ are sets of homogeneous polynomials such that the terms of all polynomials are square-free, $\langle G \rangle = \langle G' \rangle$, and $N_d = N'_d$ for any $d \geq 0$, where N_d and N'_d are the numbers of all polynomials of degree d in G and G', respectively. Then we have G is semi-regular over $GF(2)$ iff so is G'.*

Proof. By Theorem 2.4.6, if G is semi-regular over GF(2), then the Hilbert series of $\langle x_1^2, \ldots, x_n^2, f_1, \ldots, f_m \rangle$ equals

$$\left[\frac{(1+z)^n}{\prod_{i=1}^{m}(1+z^{d_i})}\right] = \left[\frac{(1+z)^n}{\prod_{d\geq 0}(1+z^d)^{N_d}}\right],$$

where $d_i = \deg f_i$. Since $N_d = N'_d$ for all d, and

$$\langle x_1^2, \ldots, x_n^2, f_1, \ldots, f_m \rangle = \langle x_1^2, \ldots, x_n^2, f'_1, \ldots, f'_m \rangle,$$

G' is also semi-regular over GF(2). □

In other words, invertible linear transformations do not influence on the semi-regularity of polynomial sequences. In the next proposition the elimination of variables using linear equations is considered.

Proposition 6.3.2. *Let* $R'_2 = GF(2)[x_1, \ldots, x_n]$, $R_2 = R'_2[y_1, \ldots, y_m]$, *and*

$$G = \{y_i + g_i(x_1, \ldots, x_n), f_j(x_1, \ldots, x_n, y_1, \ldots, y_m)\} \subset R_2$$

with $1 \leq i \leq m$, $1 \leq j \leq l$, *and* $\deg(g_i) = 1$. *Suppose* $h_j(x_1, \ldots, x_n) = f_j(x_1, \ldots, x_n, g_1, \ldots, g_m)$ *and* $G' = \{h_1, \ldots, h_l\}$. *Then we have if* G *is semi-regular over* $GF(2)$ *in* R_2, *so is* G' *in* R'_2.

Proof. Let $\tilde{g}_i = DF(g_i)$, $\tilde{f}_j = DF(f_j)$, and $\tilde{h}_j = DF(h)$ with $1 \leq i \leq m$ and $1 \leq j \leq l$. If $\tilde{f}_j(x_1, \ldots, x_n, \tilde{g}_1, \ldots, \tilde{g}_m) \in \langle x_1^2, \ldots, x_n^2 \rangle$ for some j, then

$$f_j(x_1, \ldots, x_n, y_1, \ldots, y_m) \in \langle x_1^2, \ldots, x_n^2, y_1 + \tilde{g}_1, \ldots, y_m + \tilde{g}_m \rangle;$$

this contradicts the condition of the semi-regularity of G over GF(2). Hence $\tilde{f}_j(x_1, \ldots, x_n, \tilde{g}_1, \ldots, \tilde{g}_m) = \tilde{h}_j$ for all j, and there are square-free terms in each h_j. Further, it is clear that

$$\langle y_1+\tilde{g}_1, \ldots, y_m+\tilde{g}_m, x_1^2, \ldots, x_n^2, y_1^2, \ldots, y_m^2 \rangle = \langle y_1+\tilde{g}_1, \ldots, y_m+\tilde{g}_m, x_1^2, \ldots, x_n^2 \rangle.$$

Let $\mathfrak{I}$ denote the ideal

$$\begin{aligned}&\langle x_1^2, \ldots, x_n^2, y_1^2, \ldots, y_m^2, y_1 + \tilde{g}_1, \ldots, y_m + \tilde{g}_m, \tilde{f}_1, \ldots, \tilde{f}_l \rangle = \\ &\langle y_1 + \tilde{g}_1, \ldots, y_m + \tilde{g}_m, x_1^2, \ldots, x_n^2, \tilde{f}_1, \ldots, \tilde{f}_l \rangle.\end{aligned}$$

As was shown in Proposition 6.2.2, the Hilbert series $h_{R_2}(\mathfrak{I})$ of $\mathfrak{I} \subset R_2$ is equal to the Hilbert series $h_{R'_2}(\mathfrak{I}')$, where

$$\mathfrak{I}' = \langle x_1^2, \ldots, x_n^2, \tilde{h}_1, \ldots, \tilde{h}_l \rangle \subset R'_2.$$

Since G is semi-regular over GF(2), the Hilbert series of $\mathfrak{I}$ according to Theorem 2.4.6 is given by

$$h_{R_2}(\mathfrak{I}) = \left[\frac{(1+z)^{n+m}}{\prod_{i=1}^{m}(1+z)\cdot\prod_{j=1}^{l}(1+z^{d_j})}\right] = \left[\frac{(1+z)^{n}}{\prod_{j=1}^{l}(1+z^{d_j})}\right],$$

where $d_j = \deg(\tilde{f}_j) = \deg(\tilde{h}_j)$. By Theorem 2.4.6, G' is semi-regular over GF(2). □

Now we prove the following proposition.

Proposition 6.3.3. *Let $G = (g_1, \ldots, g_m)$ be a sequence of homogeneous polynomials in $R = GF(2)[\mathcal{X}] = GF(2)[x_1, \ldots, x_n]$. By D_{reg} denote the degree of regularity of the ideal $\langle x_1^2, \ldots, x_n^2, g_1, \ldots, g_m\rangle$. Suppose that for some subset of variables $X = \{x_{i_1}, \ldots, x_{i_k}\} \subset \mathcal{X}$ and some polynomial $g \in G$ the following conditions hold:*

1. *$\#X < D_{reg} - \deg(g)$;*
2. *for any $t \in T(g)$ there exist $x_{i_j} \in X$ and $x_l \notin X$ such that $t = x_{i_j}x_l t'$ with $t' \in \mathcal{T}_{\deg(g)-2}$.*

Then G is not semi-regular over GF(2).

Proof. Consider the polynomial $f = x_{i_1}\ldots x_{i_k}g$. By assumption, we have $\deg(f) < D_{reg}$ and $f \in \left\langle x_{i_1}^2, \ldots, x_{i_k}^2\right\rangle$. Put

$$J = \left\{x_1^2, \ldots, x_n^2\right\} \cup T(g).$$

Since any $t \in T(\langle J\rangle)$ is divisible by x^2 for some $x \in X$ or by $y \in \mathcal{X} \setminus X$, we have

$$x_{i_1}\ldots x_{i_k} \notin \langle J\rangle \supset \left\langle x_1^2, \ldots, x_n^2, g\right\rangle.$$

Thus G is not semi-regular. □

Corollary 6.3.4. *The* AES *system of quadratic equations over GF(2) is not semi-regular over GF(2).*

Proof. Consider the AES system resulting after elimination all $\mathbf{X}_i$ variables, which are corresponding to inputs to S-boxes, using linear equations. By Proposition 6.3.2, if this system is not semi-regular over GF(2), so is also the initial system. Let DF_A denote the set of the degree forms of polynomials in the AES system, and let $\mathcal{X}$ be the set of all variables in this system, i.e., $\mathcal{X}$ consists of all $\mathbf{Y}_i$ and $\mathbf{K}_i$ variables. In the case where the AES S-box

is described using 23 quadratic equations with a reduced number of terms (Section 3.2.2), we have a quadratic polynomial $f \in \mathrm{GF}(2)[\mathbf{k}_{0,0}, \mathbf{y}_{0,0}]$ such that

$$T_2(f) \subset \left\{k_{0,0}^{(i)} \cdot y_{0,0}^{(j)} \colon 0 \leq i, j \leq 7\right\}.$$

By Lemma 6.3.1, we can assume w.l.o.g. that such polynomial is also included in an S-box expression as 39 quadratic equations. Let $\tilde{f} = DF(f)$, then we see that

$$y_{0,0}^{(0)} \dots y_{0,0}^{(7)} \cdot \tilde{f} \in \left\langle \left(y_{0,0}^{(0)}\right)^2, \dots, \left(y_{0,0}^{(7)}\right)^2 \right\rangle.$$

To show that this is a non-trivial relation in DF_A, we need only to prove that the degree of regularity of DF_A over $\mathrm{GF}(2)$, denoted here by D_{reg}, is more than 10. Since the value of D_{reg} given in [3] is asymptotic, we do not use it. Let

$$U = \{y_{2i,4j}^{(0)} \colon 0 \leq i \leq 4,\ 0 \leq j \leq 3\} \quad \text{and} \quad u = \prod_{y \in U} y.$$

It can directly be checked that the variables of U occur only in quadratic terms of DF_A, and if $t \in T(DF_A)$ is such that $t = y \cdot t'$ for some $y \in U$, then $t' \in \mathcal{X} \setminus U$. Therefore any $t \in T(DF_A)$ does not divide u. We have $u \notin \langle J \rangle \supset \mathrm{HT}\left(\langle S \rangle\right)$, where

$$S = \left\{v^2 \colon v \in \mathcal{X}\right\} \cup DF_A, \qquad J = \left\{v^2 \colon v \in \mathcal{X}\right\} \cup T\left(DF_A\right).$$

Thus, $D_{reg} > \deg(u) = 20$. □

In [9] Biryukov and De Cannière have obtained polynomial system of quadratic equations for the block cipher Khazad, Misty1, Kasumi, Camellia, and Serpent. Using Proposition 6.3.3 it can be proved that these systems are not semi-regular over $\mathrm{GF}(2)$.

Chapter 7

Algebraic Collision Attacks on AES

In this chapter we use Gröbner bases to improve side-channel collision attacks on AES. Side-channel collision attacks were introduced in [51] and applied to AES in [50, 10]. These attacks work in two steps. First an attacker applies differential power analysis to a physical implementation of a cryptosystem to extract some additional secret information about this system. By the second step the attacker recovers the secret key using the derived information. In the case of AES, the attacker detects by comparing power consumption curves for S-box operations whether two input bytes to these S-boxes are equal. In the basic attack proposed in [50] only collisions occurring in the input bytes of the second round of different AES runs at equal byte positions are used. In [10] it was shown that the equality of inputs to various S-boxes can be detected. These collisions called *generalized internal collisions* can be described as a system of polynomial equations over $\mathrm{GF}(2^8)$ in key byte variables. In [10] only systems that can be solved by linear algebra methods were considered. To improve these results, in our attacks non-linear collisions as well as non-collisions are taken into account. Here we do not discuss side-channel techniques and focus on the key recovery problem under the assumption that generalized internal collisions, as described in [10], can be detected. For more details on differential power analysis including the AES case we refer the reader to [51], [50], [10], and [45].

7.1 Collisions in AES

By one or several AES runs a generalized internal collision occurs whenever input bytes to any two S-boxes are equal. Since each round of one AES en-

cryption has 16 S-boxes, there is a wide variety of possible collisions. However only some of these collisions can be efficiently exploited. In the following we describe several kinds of such useful collisions and how they can be used to recover the full AES secret key. The first two subsections recall the known collision attacks on AES from [50] and [10]. Then the linear and non-linear collisions used in our algebraic collision attacks as well as non-collisions are described.

Let us assume that $m \geq 2$ plaintexts denoted by $P^{(e)} = (p_0^{(e)}, \ldots, p_{15}^{(e)})$ with $1 \leq e \leq m$ are encrypted using AES-128 with a fixed secret key, $K = (k_0, \ldots, k_{15})$. Denote by $b_{i,j}^{(e)}$ the jth byte of the internal state before the ith application of the `SubBytes` transformation for eth AES run, and by $k_{i,j}$ the jth byte of the ith round key, where $0 \leq i \leq 9$ and $0 \leq j \leq 15$. In particular, we have $k_{0,j} = k_j$ and $b_{0,j}^{(e)} = p_j^{(e)} + k_j$ for any j. Also we assume that all plaintexts are known to an attacker.

7.1.1 Internal Collisions

In [50], Schramm, Leander, Felke, and Paar have proposed side-channel collision attacks on AES that are based on detecting internal collisions. An internal collision, as defined in [50], occurs, if $b_{i,j}^{(d)} = b_{i,j}^{(e)}$ for some i, j and $d \neq e$. We see that collisions between bytes of the first round give no information about the secret key. Indeed, $b_{0,j}^{(d)} = p_j^{(d)} + k_j$ and $b_{0,j}^{(e)} = p_j^{(e)} + k_j$ are equal iff $p_j^{(d)} = p_j^{(e)}$. Each byte of any state after the second round depends on all bytes of the secret key, while any $b_{1,j}^{(e)}$ depends on four bytes of the first round key and one byte of the second round key. For this reason only internal collisions between bytes of the second round are used in [50] to attack AES.

Suppose $b_{1,0} = b'_{1,0}$ for some two AES runs. Since

$$b_{1,0}^{(e)} = k_{1,0} + \mathtt{02} \cdot S(p_0^{(e)} + k_0) + \mathtt{03} \cdot S(p_5^{(e)} + p_5) + S(p_{10}^{(e)} + k_{10}) + S(p_{15}^{(e)} + k_{15})$$

for any $e = \overline{1, m}$, we have $b_{1,0}$ and $b'_{1,0}$ collide iff

$$\begin{aligned} &\mathtt{02} \cdot S(p_0 + k_0) + \mathtt{03} \cdot S(p_5 + p_5) + S(p_{10} + k_{10}) + S(p_{15} + k_{15}) = \\ &= \mathtt{02} \cdot S(p'_0 + k_0) + \mathtt{03} \cdot S(p'_5 + p_5) + S(p'_{10} + k_{10}) + S(p'_{15} + k_{15}). \end{aligned} \tag{7.1}$$

If $(p_0, p_5, p_{10}, p_{15}) \neq (p'_0, p'_5, p'_{10}, p'_{15})$, then (7.1) describes a non-trivial relation between four bytes of the secret key and can be used to reduce the set of possible keys. Similar equations in $\{k_0, k_5, k_{10}, k_{15}\}$, $\{k_3, k_4, k_9, k_{14}\}$, $\{k_2, k_7, k_8, k_{13}\}$, or $\{k_1, k_6, k_{11}, k_{12}\}$ are derived from internal collisions between bytes of the second round at other byte positions.

By definition, put

$$\begin{aligned} C(\alpha,\beta,k_0,k_1,k_2,k_3) = {} & \mathsf{02}\cdot(S(\alpha+k_0)+S(\beta+k_0))+ \\ & +\mathsf{03}\cdot(S(\alpha+k_1)+S(\beta+k_1))+ \\ & +S(\alpha+k_2)+S(\beta+k_2)+S(\alpha+k_3)+S(\beta+k_3) \end{aligned}$$

for any $\alpha,\beta,k_0,k_1,k_2,k_3 \in \mathrm{GF}(2^8)$. It is obvious that $C(\alpha,\beta,k_0,k_1,k_2,k_3) = 0$ iff $C(\alpha+\beta,0,k_0+\beta,k_1+\beta,k_2+\beta,k_3+\beta) = 0$. The optimized attack given in [50] works as follows. For every $\delta \in \mathrm{GF}(2^8)\setminus\{0\}$ the set

$$T_\delta = \{(k_0,k_1,k_2,k_3) \in \mathrm{GF}(2^8)^4 \colon C(\delta,0,k_0,k_1,k_2,k_3) = 0\}$$

is pre-computed and stored. Each set has on average 2^{24} elements. The number of the stored elements can be reduced approximately by a factor of 32 using the following property of T_δ.

Lemma 7.1.1. *If $(k_0,k_1,k_2,k_3) \in T_\delta$ for some $\delta \in GF(2^8)\setminus\{0\}$, then*

$$\begin{aligned} &(k_0+\delta_0,k_1+\delta_1,k_2+\delta_2,k_3+\delta_3) \in T_\delta, \\ &(k_0+\delta_0,k_1+\delta_1,k_3+\delta_3,k_2+\delta_2) \in T_\delta, \end{aligned}$$

where $\delta_0,\delta_1,\delta_2,\delta_3 \in \{0,\delta\}$.

Further, to derive the secret key an attacker inputs different plaintexts in the form of $(\alpha_e,\ldots,\alpha_e)$ with random values $\alpha_e \in \mathrm{GF}(2^8)$ to an AES module. For each plaintext, the attacker measures and stores the power consumption curves for the time periods, where $b^{(e)}_{1,0},\ldots,b^{(e)}_{1,15}$ are processed. Then one look for internal collisions in each byte comparing pairwise the corresponding power curves. To detect collisions various methods can be used, such as square differences, cross-correlation, wavelet analysis. If for some pair (α_e,α_d) an internal collision are detected, then the right value of four bytes of the secret key belong to the set

$$\{(k_0+\alpha_e,k_1+\alpha_e,k_2+\alpha_e,k_3+\alpha_e) \colon (k_0,k_1,k_2,k_3) \in T_{\alpha_e+\alpha_d}\}.$$

The key bytes corresponding the internal collision at the ith byte position of the second round given in Table 7.1. We see that any collisions in the bytes of one column provides a set of possible values of the same four key bytes. According to [50], the intersection of these sets has only one element after about four such collisions. Thus about 16 collisions (four collisions for each column) are required to recover the full secret key. If there is more than one key candidate, the attacker repeats the procedure to derive addition collisions or tests these candidates using known plaintext-ciphertext pairs.

Table 7.1: Internal collisions and corresponding key bytes

1th column	(k_0, k_1, k_2, k_3)	3th column	(k_0, k_1, k_2, k_3)
0	$(k_0, k_5, k_{10}, k_{15})$	8	(k_8, k_{13}, k_2, k_7)
1	$(k_5, k_{10}, k_{15}, k_0)$	9	(k_{13}, k_2, k_7, k_8)
2	$(k_{10}, k_{15}, k_0, k_5)$	10	(k_2, k_7, k_8, k_{13})
3	$(k_{15}, k_0, k_5, k_{10})$	11	(k_7, k_8, k_{13}, k_2)
2th column		4th column	
4	(k_4, k_9, k_{14}, k_3)	12	$(k_{12}, k_1, k_6, k_{11})$
5	(k_9, k_{14}, k_4, k_4)	13	$(k_1, k_6, k_{11}, k_{12})$
6	(k_{14}, k_3, k_4, k_9)	14	$(k_6, k_{11}, k_{12}, k_1)$
7	(k_3, k_4, k_9, k_{14})	15	$(k_{11}, k_{12}, k_1, k_6)$

Let Pr_m be the probability that for m random plaintexts at least one internal collision occurs in a single fixed byte. Obviously, $\mathrm{Pr}_m = 1$ if $m > 256$, and for $2 \leq m \leq 256$ we have

$$\mathrm{Pr}_m = 1 - \prod_{i=1}^{m-1}(1 - i/2^8).$$

Since $\mathrm{Pr}_m > (0.5)^{1/16}$ for any $m \geq 40$, after 40 measurements the attacker has the required number of internal collision at least in half of all cases.

7.1.2 Linear Generalized Internal Collisions

The concept of generalized internal collisions was proposed by Bogdanov in [10]. An generalized internal collision occurs, if $b_{i,j}^{(d)} = b_{r,s}^{(e)}$ by some two different S-box applications, i.e., $(i, j, d) \neq (r, s, e)$. The collisions between bytes of the first round ($i = r = 0$) are called *linear*. The linear collisions with $j = s$ is trivial because they occur iff $p_j^{(d)} = p_j^{(e)}$, and hence they can be rejected. If $b_{0,j} = b'_{0,s}$ with some $j \neq s$, we have

$$k_j + k_s = p_j + p'_s,$$

and k_j is known iff k_s is known. Thus a set of linear generalized collisions can be described as a system of linear equations over GF(2^8) in secret key byte variables:

$$\mathbb{S}: \quad \begin{cases} k_{j_1} + k_{j_2} = \Delta_1 \\ k_{j_3} + k_{j_4} = \Delta_2 \\ \dots \\ k_{j_{2n-1}} + k_{j_{2n}} = \Delta_n \end{cases} \tag{7.2}$$

Table 7.2: Offline complexity and success probabilities

Measurements, m	4	5	6	7	9	11	29
Linear equations, n	7.09	10.72	14.88	19.46	29.49	40.07	105.14
Independent variables, $d_{\mathbb{S}}$	8.81	5.88	3.74	2.20	1.15	1.04	1.00
Offline complexity ≤ 40 bit	34.70	37.34	37.15	34.74	21.36	12.11	8
$\Pr(d_{\mathbb{S}} \leq 5)$	0.037	0.372	0.854	0.991	1.000	1.000	1.000
Offline complexity ≤ 48 bit	43.90	45.50	44.30	41.14	21.36	12.11	8
$\Pr(d_{\mathbb{S}} \leq 6)$	0.092	0.548	0.927	0.997	1.000	1.000	1.000

Here any Δ_i is the sum of two known plaintext bytes. Note that in system (7.2) there are equations not necessarily for all 16 key bytes. Moreover, it was shown in [10] that this system has never a single solution. Let $\mathcal{K}_{\mathbb{S}}$ be a set of all free and missing variables for $\mathbb{S}$. Thus we have $d_{\mathbb{S}} = \#\mathcal{K}_{\mathbb{S}} \geq 1$ for any system. Since in this case there are $2^{8d_{\mathbb{S}}}$ key candidates, the correct key is identified using a known plaintext-ciphertext pair. The dependence of $d_{\mathbb{S}}$ on the number of measurements was analyzed in [10]. The results of this analysis is given in Table 7.2.

Thus, using linear collision attacks one can derive the secret key after 5 measurements in $2^{45.5}$ steps on average with a probability of 0.548, while with 6 measurements the attack works in $2^{37.15}$ steps and has a success probability of 0.85. We see also that after 11 measurements the expected offline attack complexity is about $2^{12.11}$, and practically all systems being solvable.

7.1.3 Non-linear Generalized Internal Collisions

To improve the results of the above collision attacks we consider linear collisions in combination with other kinds of generalized internal collisions. If input bytes $b_{i,j}^{(d)}$ and $b_{r,s}^{(e)}$ of two S-boxes collide, we have the simple linear equation over GF(2^8):

$$b_{i,j}^{(d)} + b_{r,s}^{(e)} = 0,$$

which corresponds to 8 linear equations over GF(2) in bit variables. On the other hand, each of these bytes depends on bytes of some plaintext and the secret key. This relation can be described by a system of polynomial equations, for example, using one of the AES representation given in Section 3.2. For all bytes except the inputs of the first round, the corresponding system is not linear, and so a generalized internal collision between $b_{i,j}^{(d)}$ and $b_{r,s}^{(e)}$ with $i \neq 1$ or $r \neq 1$ is called *non-linear*. It is clear that one can derive a system of equations for any subset of all detected generalized internal collision. The general idea of algebraic collision attacks is to extract some information

about the secret key by solving one of such systems. In our case we use Algorithm 3 with the Faugère F4 algorithm for Gröbner basis finding. Note that not for all subset of collisions the corresponding system can be solved efficient even if the number of detected collisions is large enough. In our attacks we use two types of non-linear collision, FS- and FL-collisions, defined below. Systems of equations corresponding to these collisions are specified in the next section, and results of analysis are given in Section 7.3.

First we consider collisions that occur in the AES between bytes of the first two rounds. We call them *FS-collisions*. We can distinguish between the following three subtypes of FS-collisions: linear collisions in the first round, nonlinear collisions between the first two rounds, and nonlinear collisions within the second round. Each non-trivial collision of the first subtype linearly binds two bytes of the secret key, while the other collisions describe non-linear relations between four or more key bytes.

Naturally, one can likewise consider collisions occurring between bytes of the first three, four, and so on rounds. However, in these cases the structure of obtained polynomial systems is more difficult. We propose a more efficient attack based on collisions between bytes of the first and last rounds. We call such collisions *FL-collisions*. An FL-collision can be one of the following types:

$$b_{0,i}^{(d)} = b_{0,s}^{(e)}, \quad b_{0,i}^{(d)} = b_{9,s}^{(e)}, \quad \text{and} \quad b_{9,i}^{(d)} = b_{9,s}^{(e)}$$

with $0 \leq i, s \leq 15$ and $1 \leq d, e \leq m$.

By comparing the corresponding power consumption curves for S-box operations one can also detect that $b_{i,j}^{(d)} \neq b_{r,s}^{(e)}$ for some $0 \leq i, r \leq 10$, $0 \leq j, s \leq 15$ and $1 \leq d, e \leq m$. In such case, we say that $\left(b_{i,j}^{(d)}, b_{r,s}^{(e)}\right)$ is a *non-collision*. In Section 7.3.3 we show how non-collisions can be used to improve collision attacks. Note also that four more S-boxes are applied in each round of the key schedule. Collisions and non-collisions with these S-boxes can be used in our attacks as well.

7.2 Algebraic Representation of Non-linear Collision

In this section we describe systems of polynomial equations for FS- and FL-collisions as well as combined systems.

7.2.1 FS-Collisions

For algebraic collision attacks based on FS-collisions, we use systems of quadratic equations over GF(2), which are derived from the polynomial representation of the AES given in Section 3.2.2. In this case, a polynomial system consists of two parts. One of them describes the first round of the AES encryption for all given plaintexts and the first round of the key schedule, while the equations of the second subsystem correspond to all detected FS-collisions. The set of variables consists of:

- 128 bit variables for the initial key; we use $k_{i,j}^{(0)}$ to denote the jth bit variable for the ith byte of the initial key;
- 128 bit variables for the first round key; we use $k_{i,j}^{(1)}$ to denote the jth bit variable for the ith byte of the first round key;
- $128 \cdot m$ bit variables for all inputs of the first S-box layer; we use $x_{i,j}^{(e)}$ to denote the jth bit variable for the ith byte of the internal state before the first `SubBytes` transformation by encryption of $P^{(e)}$;
- $128 \cdot m$ bit variables for all outputs of the first S-box layer; we use $y_{i,j}^{(e)}$ to denote the jth bit variable for the ith byte of the internal state after the first `SubBytes` transformation by encryption of $P^{(e)}$;
- $128 \cdot m$ bit variables for all inputs of the second S-box layer; we use $z_{i,j}^{(e)}$ to denote the jth bit variable for the ith byte of the internal state before the second `SubBytes` transformation by encryption of $P^{(e)}$;

where m is the number of different AES runs, $0 \leq i \leq 15,\ 0 \leq j \leq 7$, and $1 \leq e \leq m$.

For FS-collisions and m known plaintexts, each polynomial system we consider is the union of the following set of equations:

1. S-box equations of the first round for all AES runs. Each of these equations is quadratic over GF(2) and has only 16 variables, $x_{i,0}^{(e)}, \ldots, x_{i,7}^{(e)}$, and $y_{i,0}^{(e)}, \ldots, y_{i,7}^{(e)}$ for some $0 \leq i \leq 15,\ 1 \leq e \leq m$.
2. Linear equations that describe the composition of the `ShiftRows`, `MixColumns`, and `AddRoundKey` transformations of the first round for all known plaintexts. For each $z_{i,j}^{(e)}$, there is exactly one equation, and the polynomial of this equation is the sum of $k_{i,j}^{(1)}$ and a linear combination of some $y_{r,s}^{(e)}$.

3. Equations of the first round of the key schedule. Recall that to express the first four bytes of the round key, quadratic S-box equations are used, while equations for the other key bits are linear.

4. The plaintext equations:
$$x_{i,j}^{(e)} + k_{i,j}^{(0)} + p_{i,j}^{(e)} = 0,$$
for all i, j, e; here $p_{i,j}^{(e)}$ is the jth bit of the ith byte of $P^{(e)}$.

5. Equations for all detected FS-collisions. For a linear collision in the first round, i.e., if $b_{0,i}^{(d)} = b_{0,r}^{(e)}$ for some $0 \le i, r \le 15,\ 1 \le d, e \le m$, we get $x_{i,j}^{(d)} + x_{r,j}^{(e)} = 0$ with $0 \le j \le 7$. If $b_{0,i}^{(d)} = b_{1,r}^{(e)}$, then $x_{i,j}^{(d)} + z_{r,j}^{(e)} = 0$ for all $0 \le j \le 7$; and $z_{i,0}^{(d)} + z_{r,0}^{(e)} = \cdots = z_{i,7}^{(d)} + z_{r,7}^{(e)} = 0$ in the case $b_{1,i}^{(d)} = b_{1,r}^{(e)}$. In the same way, one can describe the case of $b = b'$, where b or b' is an input to some S-box of the first two rounds of key schedule.

6. The GF(2)-field equations for all used variables.

The number of variables and equations can be reduced as follows. For any AES runs, the initial state after the first round is unknown, and only the collision equations contain an information about the secret key in the constructed system. If an input byte to some S-box of the second round does not collide with any other input byte, then the linear equations for the corresponding $z_{i,0}^{(e)}, \ldots, z_{i,7}^{(e)}$ can be removed from the system. Combining the linear equations for all other $z_{i,j}^{(e)}$ with collision equations, we can rewrite the system without the variables of the second round. Further, all $x_{i,j}^{(e)}$ can be eliminated using the plaintext equations. Obviously, the new system can be directly written for any set of detected FS-collisions. We see also that the number of quadratic equations of the obtained system does not depend on this set.

7.2.2 FL-Collisions

In the last round of the AES encryption the `MixColumns` transformation is not applied, and we have
$$S(b_{9,j}^{(e)}) = k_{10,\pi(j)} + c_{\pi(j)}^{(e)}$$
for all $0 \le j \le 15,\ 1 \le e \le m$, where $C^{(e)} = (c_0^{(e)}, \ldots, c_{15}^{(e)})$ is the ciphertext and
$$\pi = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 \\ 0 & 5 & 10 & 15 & 4 & 9 & 14 & 3 & 8 & 13 & 2 & 7 & 12 & 1 & 6 & 11 \end{pmatrix}$$

is the permutation corresponding to `ShiftRows`. Assume that $C^{(1)}, \ldots, C^{(m)}$ are known. For FL-collisions, we get

$$\begin{array}{lll} b_{0,i}^{(d)} = b_{0,s}^{(e)} & \text{iff} & k_{0,i} + k_{0,s} = p_i^{(d)} + p_s^{(e)}; \\ b_{9,i}^{(d)} = b_{9,s}^{(e)} & \text{iff} & k_{10,\pi(i)} + k_{10,\pi(s)} = c_{\pi(i)}^{(d)} + c_{\pi(s)}^{(e)}; \\ b_{0,i}^{(d)} = b_{9,s}^{(e)} & \text{iff} & k_{0,i} + p_i^{(d)} = S^{-1}(k_{10,\pi(s)} + c_{\pi(s)}^{(e)}). \end{array}$$

We see that FL-collisions correspond to some relations between bytes of the initial key and the last round key. These relations can be obviously expressed as a system of quadratic equations over GF(2). Now we show how to derive a system of quadratic equations over GF(2^8) for these collisions. One way is to use the BES expression as is described in Section 3.2.3. However we have 8 variables per one key byte in this case. We describe a simpler system, which has only 32 variables.

It is clear that FL-collisions of the first two types can be expressed as linear equations over GF(2^8). Let us consider a nonlinear FL-collision of the third type. Its algebraic expression is given by:

$$S\big(k_{0,i} + p_i^{(d)}\big) = k_{10,j} + c_j^{(e)},$$

for some $0 \le i, j \le 15,\ 1 \le d, e \le m$. Recall that the AES S-box is the composition of the multiplicative inverse in the finite field GF(2^8), the GF(2)-linear mapping, and the XOR-addition of the constant `63`. Recall that the inverse of the GF(2)-linear mapping is given by the following polynomial over GF(2^8):

$$f(x) = \mathtt{6e}x^{2^7} + \mathtt{db}x^{2^6} + \mathtt{59}x^{2^5} + \mathtt{78}x^{2^4} + \mathtt{5a}x^{2^3} + \mathtt{7f}x^{2^2} + \mathtt{fe}x^2 + \mathtt{05}x.$$

Hence we have

$$\big(k_{0,i} + p_i^{(d)}\big)^{-1} = f\big(k_{10,j} + c_j^{(e)} + \mathtt{63}\big) = f\big(k_{10,j}\big) + f\big(c_j^{(e)} + \mathtt{63}\big).$$

If we replace $f(k_{10,j})$ by a new variable $\tilde{k}_{10,j}$, we obtain the quadratic equation

$$\Big(k_{0,i} + p_i^{(d)}\Big)\Big(\tilde{k}_{10,j} + f\big(c_j^{(e)} + \mathtt{63}\big)\Big) = 1,$$

which holds with probability $\frac{255}{256}$. The following proposition follows:

Proposition 7.2.1. *Solutions to the equation* $S(k_{0,i} + p_i^{(d)}) = k_{10,j} + c_j^{(e)}$ *coincides with solutions to the equation*

$$\Big(k_{0,i} + p_i^{(d)}\Big)\Big(\tilde{k}_{10,j} + f\big(c_j^{(e)} + \mathtt{63}\big)\Big) = 1$$

under the change of variables $\tilde{k}_{10,j} = f(k_{10,j})$ *with a probability of* $\frac{255}{256}$.

Moreover, if $k_{10,i} + k_{10,j} = \Delta_{(i,d),(j,e)} = c_i^{(d)} + c_j^{(e)}$ for some $0 \leq i, j \leq 15$ and $1 \leq d, e \leq m$, then we have

$$f(k_{10,i}) + f(k_{10,j}) = \tilde{k}_{10,i} + \tilde{k}_{10,j} = f(\Delta_{(i,d),(j,e)}).$$

Thus we derive for FL-collisions the system $\mathbb{S}$ of quadratic equations over GF(2^8) in 32 variables $\mathcal{K} = \{k_{0,i}, \tilde{k}_{10,i}\}_{0 \leq i \leq 15}$. Furthermore, each equation of the resulting system $\mathbb{S}$ has only two variables. We call such equations *binomial*. Note that the last round key is connected with the initial key by the AES key schedule, however equations for the key schedule as well as the field equations are not included in our polynomial systems in the case of FL-collisions. Also, we ignore the S-boxes of the last round of the key schedule.

7.2.3 Combined Systems of Equations

The systems of equations given above describe FS- and FL-collisions separately. In the following we show that for successful algebraic collision attack on AES a combined approach can be used also. In this case, FL-collisions must be however expressed by polynomial systems over GF(2). Moreover, equations that describe collisions between the S-boxes of two last rounds are included in these systems. Thus the crude combined systems hold the following equations over GF(2):

1. S-box equations of the first and last rounds, i.e., quadratic equations describing the relationship between input and output of the S-boxes at the first and last rounds;

2. Equations corresponding to the linear transformation of the first round;

3. Equations that describe the inverse linear transformation of the round next to the last one (round 9);

4. Key schedule equations for the first and last rounds; one of them describe the relationship between the first round key and the initial key, and the others bind the subkeys of two last rounds; the intermediate key schedule equations are not included in the systems;

5. Plaintext and ciphertext equations;

6. Collision equations; in this case only a part of equations express the equality of inputs to some S-boxes. These are equations derived from

one of the following collisions

$$b_{0,i} = b'_{0,j} \qquad b_{0,i} = b'_{1,j} \qquad b_{1,i} = b'_{1,j}$$
$$b_{0,i} = b'_{9,j} \qquad b_{1,i} = b'_{9,j}$$

with $0 \leq i, j \leq 15$. For each collision in one of the following forms

$$b_{0,i} = b'_{8,j} \qquad b_{8,i} = b'_{8,j}$$
$$b_{8,i} = b'_{9,j} \qquad b_{9,i} = b'_{9,j}$$

with $0 \leq i, j \leq 15$, we have 8 linear equations in variables for outputs of the corresponding S-boxes. Here $b_{r,i}$ or $b'_{r,i}$ can be also an input to an S-box in the key schedule, where $r = 0, 1, 8, 9$, and $12 \leq i \leq 15$. Note that equations for collisions in the form of $b_{1,i} = b'_{8,j}$ are not included in the system.

Using linear equations one can eliminate the variables that describe inputs of the first two rounds and outputs of the S-boxes of the last two rounds. Then the resulting systems have 512 variables for $K^{(1)}$, $K^{(2)}$, $K^{(10)}$, $K^{(11)}$ as well as $128 \cdot m$ variables for outputs of the first S-box layer and $128 \cdot m$ variables for inputs to the S-boxes of the last round, where m is the number of the known plaintext/ciphertext pairs.

7.2.4 Non-Collisions

A set of non-collisions can be also described as a system of polynomial equations over GF(2) as well as over GF(2^8). Suppose two bytes $b_1 = \{x_7 x_6 \dots x_0\}$ and $b_2 = \{y_7 y_6 \dots y_0\}$ do not collide, i.e., $b_1 \neq b_2$. Then bit variables satisfy the following equation over GF(2):

$$\prod_{i=0}^{7} (x_i + y_i + 1) = 0.$$

The corresponding equation over GF(2^8) is given by

$$(b_1 + b_2)^{255} + 1 = 0.$$

The degree of the first equation is equal to 8, and the number of the terms is exactly $3^8 = 6561$, while the equation over GF(2^8) has degree 255 and 257 terms. Both equations seem to be useless for Gröbner basis attacks. However, there are more constructive applications of non-collisions reducing the search for several unknown bytes. These are specific for the structure of nonlinear equation systems we use and are explained in Subsection 7.3.3.

7.3 Algebraic Analysis of Collisions

In this section we analyze the systems of equations constructed above. We show how many AES runs are needed to recover the full secret key. Several ways to accelerate the process of finding the Gröbner bases for the combined systems are introduced and discussed, including chains of variables in binomial equations, non-linear cycles and search optimization using non-collisions.

7.3.1 Expected Number of Collisions

First note that $b_1 = b_2 = \cdots = b_r$ will be considered as $r-1$ collisions for all $r \geq 2$. Then the number of collisions between $b_1, \ldots, b_n$ obviously is equal to $n - d$, where d is the number of different elements. Let $\left\{ {n \atop d} \right\}$ denote the Stirling number of the second kind, i.e., the number of ways to partition a set of n elements into d nonempty subsets. This number is given by

$$\left\{ {n \atop d} \right\} = \frac{1}{n!} \cdot \sum_{i=0}^{d} (-1)^{d-i} \binom{d}{i} i^n.$$

If $d < 2^8$, all subsets can be tagged with different elements of GF(2^8) in $256 \cdot 255 \cdot \cdots \cdot (256-d+1)$ ways. Therefore for n random elements of GF(2^8), the average number of collisions can be computed from

$$N(n) = \frac{1}{256^n} \cdot \sum_{d=1}^{n'} (n-d) \left\{ {n \atop d} \right\} \frac{256!}{(256-d)!},$$

where $n' = \min\{n, 256\}$. We use this formula to estimate the expected number of FS- and FL-collisions after the AES encryption of m random plaintexts. In the case of FS-collisions, the S-box operation is applied $32m$ times in the first two rounds of the encryption and 8 times in the first two rounds of the key schedule. For FL-collisions, $n = 32m+4$, since the S-boxes of the last round of the key schedule are not considered here. By combined approach, one looks for collisions between inputs to $64m + 16$ S-boxes, but ignores collisions in the form of $b_1 = b'_8$, where b_1 and b'_8 are input bytes of the second and next to last rounds respectively. The result for $m = 2, 3, 4, 5$ is given in Table 7.3.1. Note that some collisions can be trivial, that is collisions occurring independently of the secret key.

Since from a non-trivial collision one can derive some information about one key byte, at least 16 collisions are required to recover the full secret key, otherwise several bytes must be guessed. In the case of FL-collisions

Table 7.3: Expected number of collisions after m measurements, E_m

Measurements		2	3	4	5
Type of Coll.		E_m			
FS	$b_0 = b'_0$	2.35	4.86	8.18	12.27
	$b_0 = b'_1$	4.43	8.68	13.98	20.10
	$b_1 = b'_1$	2.35	4.86	8.18	12.27
	$\sum$	9.13	18.40	30.34	44.64
FL	$b_0 = b'_0$	2.35	4.86	8.18	12.27
	$b_0 = b'_9$	3.96	8.07	13.25	19.28
	$b_9 = b'_9$	1.86	4.15	7.28	11.18
	$\sum$	8.17	17.08	28.71	42.73
Comb.	$\sum$	22.69	56.74	90.04	128.62

as well as by the combined approach one needs more collisions, because the intermediate key schedule equations are not included in our systems and there are more independent variables. However, it is not necessary to know the value of all variables. It is enough to find the key bytes of either the first or last round.

7.3.2 Binomial Equations, Chains and Cycles

We consider only binomial equations in key variables. They describe linear collisions of the first and last rounds as well as non-linear FL-collisions. We see that each system for FL-collisions introduced in Subsection 7.2.2 consists of only these equations.

Let $\mathbb{S}$ be a system of nonlinear equations for FL-collisions. In this case, the set of variables is

$$\mathcal{K} = \{k_i, \tilde{k}_i \colon 0 \le i \le 15\},$$

where k_i and $\tilde{k}_i$ are the initial key and last round key variables, respectively. Consider a partition of $\mathcal{K}$

$$\mathcal{K} = \mathcal{K}_1 \cup \cdots \cup \mathcal{K}_n, \quad \mathcal{K}_i \cap \mathcal{K}_j = \emptyset, \ i \neq j$$

such that

1. for any $1 \le i < j \le n$ and any two variables $v_i \in \mathcal{K}_i$ and $v_j \in \mathcal{K}_j$ there is no equation in v_1, v_2 in $\mathbb{S}$;

2. no $\mathcal{K}_i$ has a partition that satisfies the first property.

We say that a subset of the variables is connected w.r.t. $\mathbb{S}$, if this subset has no partition that satisfies the first property. Thus each $\mathcal{K}_i$ is connected. Then the system $\mathbb{S}$ can be partitioned into n isolated subsystem $\mathbb{S}_i$ corresponding to $\mathcal{K}_i$. Pairs $(\mathcal{K}_i, \mathbb{S}_i)$ are called *chains*. Obviously, $\mathbb{S}_i = \emptyset$ iff $\mathcal{K}_i = \{v\}$ for some variable $v \in \mathcal{K}$. In this case the right value of the variable v can only be guessed. If $u, v \in \mathcal{K}_i$ for some i, then there exist $v_0 = u, v_1, \ldots, v_t = v \in \mathcal{K}_i$ such that $\mathbb{S}_i$ contains a binomial equations

$$p_1(v_0, v_1) = p_2(v_1, v_2) = \cdots = p_t(v_{t-1}, v_t) = 0.$$

The case $\mathbb{S}_i$ is a linear subsystem was considered in [10]. We have $rank(\mathbb{S}_i) = \#\mathcal{K}_i - 1$, and if we fix some variable $v_{i_1} \in \mathcal{K}_i$, the other variables $v_{i_j} \in \mathcal{K}_i$ are given by $v_{i_j} = v_{i_1} + \Delta_{i_j}$ with $\Delta_{i_j} \in \mathrm{GF}(2^8)$. Suppose now $\mathbb{S}_i$ has one or more quadratic equations, i.e., $\mathcal{K}_i \cap \{k_0, \ldots, k_{15}\} \neq \emptyset$ and $\mathcal{K}_i \cap \{\tilde{k}_0, \ldots, \tilde{k}_{15}\} \neq \emptyset$. Let $v \in \mathcal{K}_i$. Since $\mathcal{K}_i$ is connected, there is a relation between v and any other variable $x \in \mathcal{K}_i$. These relations can be expressed as linear or quadratic equations in two variables. Indeed, let $x, y, z \in \mathcal{K}_i$, and

$$x \cdot y + \alpha \cdot x + \beta \cdot y + \gamma = 0; \qquad x + z = \delta,$$

where $\alpha, \beta, \gamma, \delta \in \mathrm{GF}(2^8)$. If $v + x = \bar{\delta}$, we get

$$v \cdot y + \alpha \cdot v + (\beta + \bar{\delta}) \cdot y + (\gamma + \alpha \cdot \bar{\delta}) = 0; \qquad v + z = \delta + \bar{\delta}.$$

In the case $x \cdot v + \bar{\alpha} \cdot x + \bar{\beta} \cdot v + \bar{\gamma} = 0$ we have

$$\begin{aligned}&(v + \bar{\alpha})(x \cdot y + \alpha \cdot x + \beta \cdot y + \gamma) + (y + \alpha)(x \cdot v + \bar{\alpha} \cdot x + \bar{\beta} \cdot v + \bar{\gamma}) = \\ &(\beta + \bar{\beta}) \cdot v \cdot y + (\alpha \cdot \bar{\beta} + \gamma) \cdot v + (\bar{\alpha} \cdot \beta + \bar{\gamma}) \cdot y + (\bar{\alpha} \cdot \gamma + \alpha \cdot \bar{\gamma}) = 0,\end{aligned}$$

and

$$v \cdot z + \bar{\alpha} \cdot z + (\bar{\beta} + \delta) \cdot v + (\bar{\gamma} + \bar{\alpha} \cdot \delta) = 0.$$

We see that the degree of S-polynomials does not increase and is ≤ 2. Therefore a Gröbner basis for $\mathbb{S}$ can be found quickly.

Let us now show how many solutions $\mathbb{S}_i$ has in the non-linear case. As an example, if $\mathcal{K}_i = \{v, u\}$, and $\mathbb{S}_i$ has two non-linear equations in u, v, then v is a root of a quadratic equation in one variable. Therefore $\mathbb{S}_i$ has two solutions in this case. If $\#\mathbb{S}_i \geq 3$, then the solution is single. Generally, we say that $\mathcal{K}_i$ is strongly connected w.r.t. $\mathbb{S}_i$, if there is a non-linear equation $e \in \mathbb{S}_i$ such that $\mathcal{K}_i$ is connected w.r.t. $\mathbb{S}_i \setminus \{e\}$. Such chains are called *cycles*. It can be shown that in this case $\mathbb{S}_i$ has at most two solutions. Moreover, the solution is single, if $\mathcal{K}_i$ is strongly connected w.r.t. $\mathbb{S}_i \setminus \{e\}$.

Thus the number of solutions of the whole system $\mathbb{S}$ is equal to $\prod_{i=1}^{m} 2^{q_i}$, where $q_i \leq 1$, if $\mathcal{K}_i$ is strongly connected, and $q_i = 8$ otherwise. Note that both the number of initial key candidates and the number of last round key candidates can be less than the number of all solutions. Clearly, it is enough to consider one of these subsets, the minimal one. If it also has more that one solutions, the correct key can be detected using the key schedule as well as known plaintext/ciphertext pairs. Experimental results for FL-collisions are given in Section 7.3.4.

Further, consider a combined system. It has a subsystem that consists of FL-collision equations rewritten over GF(2). However, in this case the non-linear equations are not binomial. This has a dramatical impact on the run time of Gröbner basis computation. To soften this impact we first solve all cycles over GF(2^8), then substitute the obtained solution for the corresponding bit variables in the combined system, and solve the rest of equation over GF(2).

If a system over GF(2) cannot be solved in a reasonable time, one can guess several key byte and try to solve the system again. Here the chains that are not cycles are used. As shown above, the chains possess the property that one element of the chain uniquely defines all the other elements of the chain. The strategy applied below is to find h longest chains and to guess one byte in each of them. This allows to determine the maximal number of the unknowns in the system by the same cost of guessing.

7.3.3 Speedup Using Non-Collisions

In the case of combined systems as well as systems for FS-collisions, we sometimes guess h bytes before solving as described in the previous subsection. This means that we need to solve $2^{8 \cdot h}$ resulting systems to recover the secret key. In the most practical attacks, h can be 1 or 2. Now we show how the number of guesses can be reduced using non-collisions introduced in Subsection 7.2.4. Instead of constructing implicit degree 8 nonlinear equations, we make use of the non-collisions explicitly in the following way.

Let $C_1, \ldots, C_h$ denote the h longest chains of variables induced by binomial equations in question, each being of length $|C_i| = l_i$ with $1 \leq i \leq h$, and $C_i \cap C_j = \emptyset$ for $i \neq j$. The variables of the chain C_i are denoted by $c_{i,j}$ for $1 \leq j \leq l_i$. Each of them corresponds either to initial key byte or to final key byte. Set

$$B_{i,j} = (p_r^{(1)} + c_{i,j},\ p_r^{(2)} + c_{i,j}, \ldots,\ p_r^{(m)} + c_{i,j}),$$

if $c_{ij} = k_{0,r}$, and

$$B_{i,j} = (S^{-1}(c_r^{(1)} + c_{i,j}), S^{-1}(c_r^{(2)} + c_{i,j}), \ldots, S^{-1}(c_r^{(m)} + c_{i,j}))$$

in the case of $c_{i,j} = k_{10,r}$; here m is the number of measurements with known plaintext/ciphertext pairs. In other words, we connect each chain with the table of values of input bytes to the corresponding S-boxes. For any guess $(c_{1,1}, c_{2,1}, \ldots, c_{h,1})$, we obtain all other elements of chains using the collision equations and define the tables as above. Then we can register all collisions occurring in these tables and compare the resulting list with the set of all collisions as well as non-collisions detected by measure. It is clear that the list contains the true collisions, since they are used to derive elements of the chains. However if the guess is wrong, it is possible that in the tables we have a collision between two S-boxes, where non-collision was detected in reality. This allow to optimize the search for chain evaluations. The procedure can be formalized using Algorithm 5:

Algorithm 5 Sieving guesses with non-collisions and non-linear cycles

Require: h chains $C_1, \ldots, C_h$ of lengths $l_1, \ldots, l_h$ with 2^{8h} possible evaluations, and the list L of all collisions detected for these chains

1: **for** each guess $(c_{1,1}, \ldots, c_{h,1}) \in \{0, \ldots, 2^{8h} - 1\}$ **do**
2: **for** each chain $i = 1 : h$ **do**
3: Evaluate $B_{i,1}$
4: **for** each chain variable $c_{i,j}$ $j = 2 : l_i$ **do**
5: Evaluate $c_{i,j}$ and $B_{i,j}$ using chain equations
6: **end for**
7: **end for**
8: **for** each $(b_{i,j,k}, b_{r,s,t})$ of $\frac{l(l-1)}{2}$ pairs of table elements in $T = ((B_{1,1}, \ldots, B_{1,l_1}), \ldots, (B_{h,1}, \ldots, B_{h,l_h}))$ /* where $l = m \cdot \sum l_i$ */ **do**
9: **if** $b_{i,j,k} = b_{r,s,t}$ **then**
10: Verify if the corresponding collision ε lies in L
11: **if** $\varepsilon \notin L$ **then**
12: Go to 1 (contradiction is detected)
13: **end if**
14: **end if**
15: **end for**
16: Output the guess $(c_{11}, \ldots, c_{h1})$ as a candidate evaluation of the chains
17: **end for**

In our experiments we had $h \in \{1, 2, 3\}$. Table 7.3.3 and Table 7.3.3 show the speedup we obtained on average using the sieving technique in these cases.

Table 7.4: Number of candidate chain evaluations before and after sieving using non-collisions (with and without nonlinear cycles) averaged over 5000 samples for 3 measurements, the case of FL-collisions

h	Before	After (without cycles)	After (with cycles)	After on average	Average speed-up
1	256	168.9	93.7	149	1.72
2	2^{16}	$2^{14.36}$	$2^{13.23}$	$2^{14.08}$	3.78
3	2^{24}	2^{21}	$2^{19.7}$	$2^{20.77}$	9.38

Table 7.5: Number of candidate chain evaluations before and after sieving using non-collisions averaged over 1000000 samples for m measurements with $m = 4, 5$, and 6, the case of FS-collisions

m	6			5			4		
h	1	2	3	1	2	3	1	2	3
Before	256	2^{16}	2^{24}	256	2^{16}	2^{24}	256	2^{16}	2^{24}
After	256	$2^{12.82}$	$2^{18.07}$	256	$2^{14.08}$	$2^{20.38}$	256	$2^{15.26}$	$2^{22.44}$
Speed-up	1	9.09	60.96	1	3.77	12.33	1	1.67	2.94

Note that in the case of FL-collisions contradictions can occur also within one chain, since here we have non-linear relations between key bytes. Hence in this case we can use Algorithm 5 with any $h \geq 1$. For linear chains, the method works only if we guess two or more bytes. Further, in the case of a combined system, we first solve all cycles separately, as was described above. But these cycles can be also used after their solutions are found. Adding them to an input of Algorithm 5 we detect more false evaluations for other chains. For 3 measurements, cycles occur in about 25% of all cases (averaged over 1000000 samples).

7.3.4 Experimental Results

Solving Equations for FS-Collisions

The straightforward application of the Faugère F4 algorithm to the system constructed in Subsection 7.2.1 gives results superior to those in [10]. These are summarized in Table 7.3.4.

The system of nonlinear equations is considered over GF(2). For m inputs (m measurements) there are 128 variables of the first subkey $K^{(1)}$, 128 variables of the second subkey $K^{(2)}$ and $128 \cdot m$ intermediate variables for the output bits of the first round S-box layer. The collision-independent equations include $16 \cdot t \cdot m$ quadratic equations over GF(2) connecting the inputs

Table 7.6: Solving equation systems for FS-collisions over GF(2)

Measurements	5	5	4	4
Success prob.	0.425	0.932	0.042	0.397
Run time, s	142.8	7235.8	71.5	6456.0
Memory limit, MB	500	500	500	500
# of variables	896	896	768	768
# of linear equations	$96 + 8c$	$96 + 8c$	$96 + 8c$	$96 + 8c$
# of quadratic equations* ($t = 23$)	1932	1932	1564	1564
# of quadratic equations* ($t = 39$)	3276	3276	2652	2652

*without the field equations, the number of which equals the number of variable.

and outputs of the first round S-boxes, and $4 \cdot t$ quadratic and $12 \cdot 8 = 96$ linear equations connecting $K^{(1)}$ and $K^{(2)}$ using the key schedule relations. Here t is the number of quadratic equations used to express the AES S-box, and in our experiments we have considered the both variants described in Section 3.2.2, i.e., we have $t = 23$ or 39. Each of the three types of FS-collisions add 8 linear equations to the system, resulting in $8 \cdot c$ equations if c collision occurred. In additional, the field equations for all variables are included in the system.

The system is solved in the following way. First the system is passed to the F4 algorithm without modifications. If it is not solvable, one guesses the largest connected linear component and tries to solve the system again. As a criterion of solvability, the memory cost was used. The reason why we use this criterion is the following. Suppose for some system the Gröbner basis computation has a high memory cost, then this means that in internal steps of the F4 algorithm, the Macaulay matrices that must be transformed are large. Thus the Gröbner basis computation for this system must have also a high time cost. On the other hand, there are systems that can be solved much slower than on average but in a reasonable time and with the same memory cost. We set the memory limit for the Magma program to 500 MB. Actually one usual requires less than 300 MB memory in the case of solvable systems. Moreover, for the case where the Gröbner basis computation needs more memory, to guess the next largest chain seems to be a better strategy than to raise the memory limit. Also, comparing experimental results for systems with a different number of S-box equations, we conclude that for our attacks the variant with $t = 23$ is more suitable in terms of the time cost, e.g., the ratio of the run times is about 3.9 for $t = 39$ and $t = 23$. However, there are cases where the secret key can be derived only if all S-box equations

Table 7.7: Solving equation systems for FL-collisions over GF(2^8)

Measurements	5	4
Success probability	1.00	0.82
Time for finding Gröbner basis, ms	3	5
# of guesses	$\leq 2^{32}$	$\leq 2^{32}$
Memory limit, MB	500	500
Number of variables	32	32
Average number of equations	43.58	29.66

are included in the system.

It can be seen from Table 7.3.4 that for 5 measurements most ($> 93\%$) instances of the FS-system can be solved within several hours on a PC. For 4 measurements, less systems are solvable (about 40%) within approx. 2 hours. These attacks work in the plaintext scenario.

Solving Equations for FL-Collisions

FL-collisions lead, as a rule, to more efficient results. Each equation binds only two GF(2^8)-variables, since one deals with binomial equations introduced in Subsection 7.2.2. There are 32 variables $\mathcal{K}$ over GF(2^8). The algebraic relations on these variables are much simpler, since one has both plaintext and ciphertext bytes (more information related to the detected collisions). Moreover, there are nonlinear subsystems (cycles) solvable independently (see Subsection 7.3.3). On average there are 1.02 cycles covering 30.08 out of 32 GF(2^8)-variables for 5 measurements and 0.99 cycles covering 20.08 out of 32 GF(2^8)-variables for 4 measurements. Statistically there are 43.58 collisions for 5 measurements and 29.66 collisions for 4 measurements.

Table 7.3.4 contains the results for applying the F4 algorithm to FL-systems of nonlinear equations averaged over 10000 samples. After resolving the nonlinear subsystems using F4, we guess variables defining the remaining bytes in a way similar to the linear collision attacks (see Subsection 7.1.2 and Subsection 7.3.3). For 5 measurements practically all FL-systems are solvable in several seconds (2^{32} simple offline operations), an FL-system being solvable with a probability of 0.82 within several seconds (2^{32} simple offline operations) for 4 measurements.

Table 7.8: Solving combined equation systems

Measurements	3	3	3
Success prob.	0.072	0.419	0.698
Run time	98.31 sec	4.24 hours	22.03 days
Memory limit, MB	500	500	500
h, number of chains guessed	0	1	2

Solving the Combined Systems

Though FS- and, first of all, FL-systems perform well for 4 and 5 measurements, their solution for 3 measurements is either extremely improbable or rather infeasible. Here a combined approach has to be used.

To solve the nonlinear systems we executed Algorithm 6. The results

Algorithm 6 Solving combined systems of nonlinear equations

1: **if** there are nonlinear cycles in the binomial chains **then**
2: Resolve the cycles over GF(2^8) using F4 or brute-force
3: Define bytes of the dependent chains
4: **end if**
5: Find the h longest binomial chains
6: Execute Algorithm 5 for sieving chain evaluations
7: **for** each non-contradicting evaluation of h chains **do**
8: Find Gröbner basis for the reduced combined system of nonlinear equations with F4
9: **if** the Gröbner basis $\neq \{1\}$ **then**
10: Verify the key candidates using a known plaintext-ciphertext pair
11: **end if**
12: **end for**

of the application of this algorithm to the combined system of nonlinear equations (with additional collisions) for 3 measurements can be found in Table 7.3.4. The system is solvable with a probability of 0.698 within 22 days or with a probability of 0.419 within 4.24 hours or with a probability of 0.072 within several minutes.

Chapter 8

Summary and Outlook

The topic of this thesis is the application of Gröbner bases in cryptanalysis of block ciphers. We investigate several aspects of algebraic cryptanalysis, e.g., Gröbner basis attacks with a minimal known plaintext/ciphertext pairs, the algebraic structure of polynomial representations of ciphers as well as a combination of Gröbner basis methods with other types of cryptanalysis. Note that at the beginning it was not clear whether Gröbner bases can be successfully applied to cryptanalysis of block ciphers, since polynomial systems occurring here are, as a rule, very huge. However, the thesis provides with several examples, where such application is possible.

We have shown that Gröbner basis technique can be used to successfully mount key-recovery attacks on block ciphers with a large block and key size as well as a good resistance against differential and linear cryptanalysis. The algebraically structure of these cipher, however, must be relatively simple. We have constructed Flurry and Curry - two parametrized families of block ciphers representing Feistel networks and SPN ciphers, respectively. We have designed them so that all above conditions hold. Moreover, parameters of these ciphers, as the number of rounds, the number of S-boxes, S-box functions, linear transformations, etc., can be varied independently. This allows to study how algebraic attacks depend on these parameters. We have demonstrated that some of the constructed ciphers are vulnerable against practical Gröbner basis attacks. Also, we have shown how for a set of these ciphers the key recovery problem can be efficiently reduced to a Gröbner basis conversion problem. The method allows to obtain a DRL Gröbner basis for block ciphers with a polynomial S-box without a polynomial reduction. Using complexity bounds for the FGLM algorithm, we have derived an upper bound for the time and space complexity of Gröbner basis attacks on this ciphers.

Can our method be applied to other iterated block ciphers, for example,

to AES? The positive answer is given in Chapter 5. Using a polynomial representation of the S-box over GF(2^8), we have shown how to derive a DRL zero-dimensional Gröbner basis for AES-128. In this case, however, the degree of each non-linear polynomial is equal to 254. We have considered several possible approaches to use this Gröbner basis and shown that any of them is not suitable for successful attacks.

Note also that all Gröbner basis attacks on Flurry and Curry given above require a minimal number of known plaintext/ciphertext pairs. Moreover, even if more than one pairs are known, the attacker describes and solves a system of polynomial equations only for one of them, while the other pairs are used to detect the correct key in the case of several solutions. However, algebraic attacks with many known plaintext/ciphertext pairs is an intersting topic. At FSE'07 [33] Faugère proposed Gröbner basis attacks with several chosen plaintexts on Flurry. From experimental results it follows that in the case of monomial S-boxes (as defined in Chapter 4) a system of equations describing the encryption of several plaintexts can be solved more quickly than a system for a single pair. At the same time, this property does not hold for the inversion S-box, i.e., here the time of Gröbner basis computation increases with number of known plaintext/ciphertext pairs. This is an interesting result, since in the case of a single known plaintext/ciphertext pair Flurry with a monomial S-box is more secure against Gröbner basis attacks than the cipher with the same parameters but inversion S-box. Thus algebraic attacks with many known plaintext/ciphertext pairs need a further investigation.

Since successful Gröbner basis attacks on block ciphers are possible, it must be studied carefully how Gröbner basis algorithms depend on the structure of polynomial systems corresponding to block ciphers. One of the possible approaches is based on the notation of semi-regular sequences of polynomials (e.g., [3], [2], [4]). The behavior of the F_5 Gröbner basis algorithm and the XL algorithm in the case of semi-regular sequences are well understood ([3], [2], [4], and [1]). Unfortunately, we have shown that this concept is not very useful for cryptanalysis of block ciphers. Using the AES as an example, we have considered three algebraic representations for block ciphers. We have proved that the BES and AES polynomial equations over GF(2^8) are not semi-regular, and that the AES systems of quadratic equations over GF(2) are not semi-regular over GF(2). Our methods can be also used to analyze polynomial expressions for other iterated block ciphers. For example, it can be checked that the multivariate polynomial systems of equations, given for many cryptosystem in [9], are not semi-regular over GF(2). Thus in two major cases, polynomial systems described the key recovery problem for block ciphers are neither semi-regular nor semi-regular over GF(2). Our proof is

based on non-trivial relations between the degree forms of S-box equations. Since here the equations are non-homogeneous, these non-trivial relations do not imply the reduction to zero. In other word, not all of them form syzygies for the whole polynomials. We conjecture that the presence of a large number of such relations[1] reduces the complexity of Gröbner basis computations for non-homogeneous systems. It is an open problem whether there exist non-trivial relations of small degree between polynomials of different rounds in polynomial systems for iterated block ciphers.

Note that one case of semi-regular representation for block ciphers, however, has been found. We have shown that the Gröbner basis for the AES is semi-regular.

Further, we have presented a new method of side-channel cryptanalysis - algebraic collision attacks. Actually, the method applies the standard technique of power analysis to derive some information from a device, but takes an original approach to recover the secret key using this informations. In this thesis we apply it to attack AES, but we guess that this method works also for many other block ciphers, if their implementation is vulnerable against side-channel attacks. A necessary condition for algebraic collision attacks is that generalized internal collisions, as described in [10], can be detected. We have demonstrated that systems of polynomials equations that describe several subsets of generalized internal collisions can be successfully solved in a reasonable time by Gröbner basis computations. Moreover, our approach allows to improve collision attacks both in terms of measurements and post-processing complexity. Also, several ways to speed up algebraic collision attacks were proposed. For the AES block cipher, we have described several efficient algebraic collision attacks. One of them is based on generalized internal collisions occurring within the first two rounds. It works in the known-plaintext scenario and requires 5 measurements to derive the full secret key within several hours on a PC with success probability 0.93. This attack with 4 measurements recovers key in about 40% of all cases. The second attack works in the known plaintext/ciphertext pair scenario but leads to more efficient results: the key can be obtained in several seconds of offline computations with success probability of 0.82 for 4 measurements, and with probability close to 1 for 5 measurements. We also have proposed a successful algebraic collision attack on AES with 3 measurements. The attack has a probability of 0.42 and needs 4.24 PC hours post-processing. This is to be compared to 40 measurements with some non-negligible post-processing in [50] for a success probability > 0.5 and 6 measurements with approx. $2^{37.15}$

[1] This means that for polynomials $f_1, \ldots, f_m \in R = \mathbb{F}[\mathcal{X}]$ there are $g_1, \ldots, g_m \in R$ such that $\sum g_i DF(f_i) = 0$ and $\sum g_i f_i \neq 0$.

offline computations and a success probability of 0.85 or 5 measurements with $2^{45.5}$ offline computations and a probability of 0.55 in [10]. To solve the systems of polynomial equations derived from generalized internal collisions, we have used the Magma F4 algorithm implementation. As a future task, the application of the F5 Gröbner basis algorithm [32] and PolyBoRy [12] could be considered to solve these systems. Also, the application of algebraic collision attacks to other cryptographic construction, as block ciphers, stream ciphers, and message authentication codes, will be studied.

Finally, the combination of Gröbner basis algorithms with other methods of cryptanalysis is an interesting topic for the further research.

Appendix A

Polynomial interpolation of the inverse S-Box of Rijndael

$$\sigma^{-1} : \mathrm{GF}(2^8) \to \mathrm{GF}(2^8), \quad x \mapsto$$

$$\begin{aligned}
&\mathtt{05}x^{254} + \mathtt{CF}x^{253} + \mathtt{B3}x^{252} + \mathtt{16}x^{251} + \mathtt{55}x^{250} + \mathtt{C0}x^{249} + \mathtt{7A}x^{248} + \mathtt{01}x^{247} + \\
&\mathtt{22}x^{246} + \mathtt{D8}x^{245} + \mathtt{6B}x^{244} + \mathtt{A6}x^{243} + \mathtt{1F}x^{242} + \mathtt{0D}x^{241} + \mathtt{BC}x^{240} + \mathtt{49}x^{239} + \\
&\mathtt{85}x^{238} + \mathtt{B4}x^{237} + \mathtt{1B}x^{236} + \mathtt{5E}x^{235} + \mathtt{BD}x^{234} + \mathtt{18}x^{233} + \mathtt{1D}x^{232} + \mathtt{6D}x^{231} + \\
&\mathtt{C5}x^{230} + \mathtt{23}x^{229} + \mathtt{09}x^{228} + \mathtt{43}x^{227} + \mathtt{68}x^{226} + \mathtt{80}x^{225} + \mathtt{6C}x^{224} + \mathtt{CC}x^{223} + \\
&\mathtt{42}x^{222} + \mathtt{9F}x^{221} + \mathtt{0F}x^{220} + \mathtt{D2}x^{219} + \mathtt{3B}x^{218} + \mathtt{2C}x^{217} + \mathtt{5F}x^{216} + \mathtt{BE}x^{215} + \\
&\mathtt{AE}x^{214} + \mathtt{E4}x^{213} + \mathtt{93}x^{212} + \mathtt{8B}x^{211} + \mathtt{CB}x^{210} + \mathtt{65}x^{209} + \mathtt{C0}x^{208} + \mathtt{1E}x^{207} + \\
&\mathtt{8E}x^{206} + \mathtt{32}x^{205} + \mathtt{1D}x^{204} + \mathtt{A5}x^{203} + \mathtt{76}x^{202} + \mathtt{A9}x^{201} + \mathtt{2C}x^{200} + \mathtt{13}x^{199} + \\
&\mathtt{05}x^{198} + \mathtt{60}x^{197} + \mathtt{FD}x^{196} + \mathtt{1B}x^{195} + \mathtt{AB}x^{194} + \mathtt{64}x^{193} + \mathtt{C1}x^{192} + \mathtt{A8}x^{191} + \\
&\mathtt{7F}x^{190} + \mathtt{55}x^{189} + \mathtt{DB}x^{188} + \mathtt{EC}x^{187} + \mathtt{20}x^{186} + \mathtt{C4}x^{185} + \mathtt{DB}x^{184} + \mathtt{7E}x^{183} + \\
&\mathtt{92}x^{182} + \mathtt{80}x^{181} + \mathtt{A3}x^{180} + \mathtt{59}x^{179} + \mathtt{91}x^{178} + \mathtt{91}x^{177} + \mathtt{81}x^{176} + \mathtt{4E}x^{175} + \\
&\mathtt{11}x^{174} + \mathtt{DD}x^{173} + \mathtt{4E}x^{172} + \mathtt{D3}x^{171} + \mathtt{E3}x^{170} + \mathtt{19}x^{169} + \mathtt{E7}x^{168} + \mathtt{03}x^{167} + \\
&\mathtt{24}x^{166} + \mathtt{45}x^{165} + \mathtt{DA}x^{164} + \mathtt{EA}x^{163} + \mathtt{87}x^{162} + \mathtt{2D}x^{161} + \mathtt{23}x^{160} + \mathtt{82}x^{159} + \\
&\mathtt{38}x^{158} + \mathtt{B7}x^{157} + \mathtt{9E}x^{156} + \mathtt{B3}x^{155} + \mathtt{2A}x^{154} + \mathtt{3E}x^{153} + \mathtt{1C}x^{152} + \mathtt{EC}x^{151} + \\
&\mathtt{C3}x^{150} + \mathtt{45}x^{149} + \mathtt{ED}x^{148} + \mathtt{D5}x^{147} + \mathtt{2A}x^{146} + \mathtt{8D}x^{145} + \mathtt{ED}x^{144} + \mathtt{37}x^{143} + \\
&\mathtt{26}x^{142} + \mathtt{E0}x^{141} + \mathtt{BC}x^{140} + \mathtt{58}x^{139} + \mathtt{E2}x^{138} + \mathtt{6C}x^{137} + \mathtt{24}x^{136} + \mathtt{55}x^{135} + \\
&\mathtt{C7}x^{134} + \mathtt{AA}x^{133} + \mathtt{09}x^{132} + \mathtt{4F}x^{131} + \mathtt{82}x^{130} + \mathtt{CA}x^{129} + \mathtt{10}x^{128} + \mathtt{EE}x^{127} + \\
&\mathtt{1A}x^{126} + \mathtt{2E}x^{125} + \mathtt{40}x^{124} + \mathtt{27}x^{123} + \mathtt{81}x^{122} + \mathtt{92}x^{121} + \mathtt{B1}x^{120} + \mathtt{02}x^{119} + \\
&\mathtt{8B}x^{118} + \mathtt{87}x^{117} + \mathtt{7F}x^{116} + \mathtt{B0}x^{115} + \mathtt{6F}x^{114} + \mathtt{53}x^{113} + \mathtt{08}x^{112} + \mathtt{CB}x^{111} + \\
&\mathtt{03}x^{110} + \mathtt{B0}x^{109} + \mathtt{DF}x^{108} + \mathtt{1F}x^{107} + \mathtt{A7}x^{106} + \mathtt{A2}x^{105} + \mathtt{FE}x^{104} + \mathtt{8E}x^{103} + \\
&\mathtt{A8}x^{102} + \mathtt{E1}x^{101} + \mathtt{71}x^{100} + \mathtt{FF}x^{99} + \mathtt{55}x^{98} + \mathtt{5A}x^{97} + \mathtt{1D}x^{96} + \mathtt{9D}x^{95} + \\
&\mathtt{BF}x^{94} + \mathtt{E8}x^{93} + \mathtt{BA}x^{92} + \mathtt{6B}x^{91} + \mathtt{72}x^{90} + \mathtt{E3}x^{89} + \mathtt{04}x^{88} + \mathtt{D9}x^{87} + \\
&\mathtt{38}x^{86} + \mathtt{D3}x^{85} + \mathtt{B9}x^{84} + \mathtt{16}x^{83} + \mathtt{52}x^{82} + \mathtt{18}x^{81} + \mathtt{19}x^{80} + \mathtt{3E}x^{79} + \\
&\mathtt{9E}x^{78} + \mathtt{03}x^{77} + \mathtt{56}x^{76} + \mathtt{A6}x^{75} + \mathtt{71}x^{74} + \mathtt{03}x^{73} + \mathtt{E4}x^{72} + \mathtt{86}x^{71} + \\
&\mathtt{F5}x^{70} + \mathtt{B0}x^{69} + \mathtt{05}x^{68} + \mathtt{D1}x^{67} + \mathtt{10}x^{66} + \mathtt{E2}x^{65} + \mathtt{E5}x^{64} + \mathtt{CB}x^{63} + \\
&\mathtt{B1}x^{62} + \mathtt{F2}x^{61} + \mathtt{8E}x^{60} + \mathtt{C7}x^{59} + \mathtt{0C}x^{58} + \mathtt{A7}x^{57} + \mathtt{BF}x^{56} + \mathtt{46}x^{55} + \\
&\mathtt{0B}x^{54} + \mathtt{01}x^{53} + \mathtt{C5}x^{52} + \mathtt{A3}x^{51} + \mathtt{50}x^{50} + \mathtt{77}x^{49} + \mathtt{EA}x^{48} + \mathtt{05}x^{47} + \\
&\mathtt{65}x^{46} + \mathtt{8E}x^{45} + \mathtt{89}x^{44} + \mathtt{D4}x^{43} + \mathtt{6D}x^{42} + \mathtt{D3}x^{41} + \mathtt{75}x^{40} + \mathtt{65}x^{39} + \\
&\mathtt{13}x^{38} + \mathtt{2F}x^{37} + \mathtt{86}x^{36} + \mathtt{AF}x^{35} + \mathtt{7C}x^{34} + \mathtt{7B}x^{33} + \mathtt{85}x^{32} + \mathtt{C8}x^{31} + \\
&\mathtt{E8}x^{30} + \mathtt{04}x^{29} + \mathtt{7B}x^{28} + \mathtt{CF}x^{27} + \mathtt{2F}x^{26} + \mathtt{8A}x^{25} + \mathtt{9A}x^{24} + \mathtt{3D}x^{23} + \\
&\mathtt{CF}x^{22} + \mathtt{21}x^{21} + \mathtt{39}x^{20} + \mathtt{D9}x^{19} + \mathtt{29}x^{18} + \mathtt{73}x^{17} + \mathtt{F6}x^{16} + \mathtt{23}x^{15} + \\
&\mathtt{40}x^{14} + \mathtt{1B}x^{13} + \mathtt{B2}x^{12} + \mathtt{C0}x^{11} + \mathtt{6D}x^{10} + \mathtt{85}x^{9} + \mathtt{1C}x^{8} + \mathtt{8A}x^{7} + \\
&\mathtt{2C}x^{6} + \mathtt{BB}x^{5} + \mathtt{90}x^{4} + \mathtt{1E}x^{3} + \mathtt{7E}x^{2} + \mathtt{F3}x^{1} + \mathtt{52}
\end{aligned}$$

Appendix B

A DRL Gröbner basis for $\textsc{Flurry}(32, 2, 4, f_3, D_2)$

The following sequence of polynomials G is a degree-reverse lexicographic Gröbner basis for a $\textsc{Flurry}(32, 2, 4, f_3, D_2)$ for the following variable ordering:
$x_0 < x_1 < x_2 < x_3 < x_{16} < x_{17} < x_{18} < x_{19} < x_{14} < x_{15} < k_0 < k_1 < k_6 < k_7 < k_2 < k_3 < k_4 < k_5 < k_8 < k_9 < k_{10} < k_{11} < x_4 < x_5 < x_6 < x_7 < x_8 < x_9 < x_{10} < x_{11} < x_{12} < x_{13}$

$G = \{$
`plaintext:`
$x_0 + \theta^{31} + \theta^{29} + \theta^{27} + \theta^{24} + \theta^{22} + \theta^{21} + \theta^{19} + \theta^{13} + \theta^{11} + \theta^8 + \theta^7 + \theta^6 + \theta^4 + 1$
$x_1 + \theta^{31} + \theta^{30} + \theta^{29} + \theta^{22} + \theta^{21} + \theta^{15} + \theta^{14} + \theta^{11} + \theta^{10} + \theta^7 + \theta^6 + \theta^5 + \theta^3 + \theta$
$x_2 + \theta^{26} + \theta^{25} + \theta^{24} + \theta^{21} + \theta^{19} + \theta^{18} + \theta^{16} + \theta^{14} + \theta^8 + \theta^7 + \theta^6 + \theta^4 + \theta + 1$
$x_3 + \theta^{27} + \theta^{26} + \theta^{24} + \theta^{21} + \theta^{17} + \theta^{15} + \theta^{13} + \theta^{11} + \theta^9 + \theta^6 + \theta^4 + \theta$
`ciphertext:`
$x_{16} + \theta^{31} + \theta^{29} + \theta^{21} + \theta^{19} + \theta^{18} + \theta^{16} + \theta^{15} + \theta^{14} + \theta^{12} + \theta^4 + 1$
$x_{17} + \theta^{24} + \theta^{21} + \theta^{20} + \theta^{18} + \theta^{16} + \theta^{13} + \theta^{10} + \theta^9 + \theta^8 + \theta^6 + \theta^5 + \theta^3 + \theta + 1$
$x_{18} + \theta^{29} + \theta^{25} + \theta^{21} + \theta^{20} + \theta^{19} + \theta^{13} + \theta^{10} + \theta^9 + \theta^8 + \theta^7 + \theta^6 + \theta^5 + \theta^3$
$x_{19} + \theta^{29} + \theta^{27} + \theta^{26} + \theta^{20} + \theta^{13} + \theta^{10} + \theta^8 + \theta^5 + \theta^2$
`round 1:`
$x_4 + x_2$
$x_5 + x_3$
$k_0^3 + k_0^2 x_2 + k_0 x_2^2 + x_2^3 + C_1 x_7 + C_1 x_6 + C_1 x_1 + C_1 x_0$
$k_1^3 + k_1^2 x_3 + k_1 x_3^2 + x_3^3 + C_2 x_7 + C_1 x_6 + C_2 x_1 + C_1 x_0$
`round 2:`
$x_8 + x_6$
$x_9 + x_7$
$x_6^3 + x_6^2 k_2 + x_6 k_2^2 + k_2^3 + C_1 x_{11} + C_1 x_{10} + C_1 x_5 + C_1 x_4$
$x_7^3 + x_7^2 k_3 + x_7 k_3^2 + k_3^3 + C_2 x_{11} + C_1 x_{10} + C_2 x_5 + C_1 x_4$

`round 3:`

$x_{12} + x_{10}$

$x_{13} + x_{11}$

$x_{10}^3 + x_{10}^2 k_4 + x_{10} k_4^2 + k_4^3 + C_1 x_9 + C_1 x_8 + C_1 k_9 + C_1 k_8 + C_1 x_{15} + C_1 x_{14}$

$x_{11}^3 + x_{11}^2 k_5 + x_{11} k_5^2 + k_5^3 + C_2 x_9 + C_1 x_8 + C_2 k_9 + C_1 k_8 + C_2 x_{15} + C_1 x_{14}$

`round 4:`

$x_{14} + x_{16}$

$x_{15} + x_{17}$

$k_6^3 + k_6^2 x_{14} + k_6 x_{14}^2 + x_{14}^3 + C_1 x_{13} + C_1 x_{12} + C_1 k_{11} + C_1 k_{10} + C_1 x_{19} + C_1 x_{18}$

$k_7^3 + k_7^2 x_{15} + k_7 x_{15}^2 + x_{15}^3 + C_2 x_{13} + C_1 x_{12} + C_2 k_{11} + C_1 k_{10} + C_2 x_{19} + C_1 x_{18}$

`key expansion:`

$k_{11} + \theta^2 k_7 + (\theta^2 + \theta + 1) k_1 + \theta k_0 + \theta^4 + \theta^2$

$k_{10} + \theta^2 k_6 + \theta k_1 + k_0 + \theta^3 + \theta$

$k_9 + (\theta^2 + \theta) k_7 + (\theta + 1) k_6 + \theta^2 k_1 + (\theta + 1) k_0 + \theta^6 + \theta^5 + \theta^3 + 1$

$k_8 + (\theta + 1) k_7 + (\theta + 1) k_6 + (\theta + 1) k_1 + k_0 + \theta^5 + \theta^3$

$k_5 + (\theta^2 + \theta + 1) k_7 + \theta k_6 + \theta^2 k_1 + (\theta + 1) k_0 + \theta^6 + \theta^4 + \theta^3 + \theta$

$k_4 + \theta k_7 + k_6 + (\theta + 1) k_1 + k_0 + \theta^5 + \theta^4 + \theta^3 + 1$

$k_3 + \theta^2 k_7 + (\theta + 1) k_6 + (\theta^2 + \theta + 1) k_1 + \theta k_0 + \theta^6 + \theta^5 + \theta^4 + \theta$

$k_2 + (\theta + 1) k_7 + k_6 + \theta k_1 + k_0 + \theta^5 + \theta^2 + \theta + 1$

}

with $C_1 = (\theta + 1)^{-1}$ and $C_2 = 1 + (\theta + 1)^{-1}$

Appendix C

Details about the Computational Platform for Experiments

Two chapters of this thesis provides with a diversity of experimental results.

Experimental Results given in Chapter 4 were obtained using the computer algebra system Magma [53], version 2.11-8, on an AMD Athlon 64 3200+ equipped with 1024 Megabytes of RAM running Linux.

For Chapter 7 we implemented all our attacks in Magma V2.13-10 (including the AES algorithm itself as well as its random inputs) running on a dual-core AMD Opteron processor with 1 MB cache and clocked at 2613.39 MHz under Linux. We used only one execution thread of the Opteron processor and not more than 1 GB of RAM. Thus, it is claimed that the performance figures of our attacks on a standard PC with a single-thread processor and 1 GB RAM are comparable to those given in this thesis.

Bibliography

[1] Gwénolé 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 338–353. Springer–Verlag, 2004.

[2] Magali Bardet. *Étude des systèmes algébriques surdéterminés. Applications aux codes correcteurs et à la cryptographie.* PhD thesis, Université Paris 6, 2004.

[3] Magali Bardet, Jean-Charles Faugère, and Bruno Salvy. Complexity of Gröbner Basis Computation for Semi-Regular Overdetermined Sequences over GF(2) with Solutions in GF(2). Technical Report RR-5049, INRIA, 2003.

[4] Magali Bardet, Jean-Charles Faugère, Bruno Salvy, and Bo-Yin Yang. Asymptotic Behaviour of the Degree of Regularity of Semi-Regular Polynomial Systems. In P. Gianni, editor, *Mega 2005*, 2005.

[5] Thomas Becker and Volker Weispfenning. *Gröbner Bases – A Computational Approach to Commutative Algebra.* Springer–Verlag, 1991.

[6] Thomas Beth and Cunsheng Ding. On Almost Perfect Nonlinear Permutations. In Tor Helleseth, editor, *Advances in Cryptology – EUROCRYPT '93*, volume 765 of *Lecture Notes in Computer Science*, pages 65–76. Springer–Verlag, 1994.

[7] Eli Biham and Adi Shamir. Differential Cryptanalysis of DES-like Cryptosystems. In Alfred Menezes and Scott A. Vanstone, editors, *Advances in Cryptology – CRYPTO '90*, volume 537 of *Lecture Notes in Computer Science*, pages 2–21. Springer–Verlag, 1991.

[8] Alex Biryukov, Andrey Bogdanov, Dmitry Khovratovich, and Timo Kasper. Collision Attacks on Alpha-MAC and Other AES-based MACs. In *CHES'07*, LNCS. Springer-Verlag, 2007.

[9] Alex Biryukov and Christophe De Cannière. Block Ciphers and Systems of Quadratic Equations. In Thomas Johansson, editor, *Fast Software Encryption – FSE 2003*, Lecture Notes in Computer Science, pages 274–289. Springer–Verlag, 2003.

[10] Andrey Bogdanov. Improved Collision Attacks on AES. In *The 14th Annual Workshop on Selected Areas in Cryptography (SAC 2007), Ottawa, Ontario, Canada*, LNCS. Springer-Verlag, 2007.

[11] Andrey Bogdanov and Andrey Pyshkin. Algebraic Side-Channel Collision Attacks on AES. Cryptology ePrint Archive, Report 2007/477, 2007. `http://eprint.iacr.org/`.

[12] M. Brickenstein and A. Dreyer. POLYBORI: A Gröbner basis framework for Boolean polynomials. Technical report, Fraunhofer-Institut für Techno- und Wirtschaftsmathematik ITWM, 2007.

[13] Bruno Buchberger. *Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal*. PhD thesis, 1965.

[14] 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. *J. Symb. Comput.*, 41(3-4):475–511, 2006.

[15] Bruno Buchberger. Comments on the translation of my PhD thesis. *J. Symb. Comput.*, 41(3-4):471–474, 2006.

[16] Johannes Buchmann, Andrei Pyshkin, and Ralf-Philipp Weinmann. A Zero-Dimensional Gröbner Basis for AES-128. In Matthew Robshaw, editor, *Fast Software Encryption – FSE 2006*, Lecture Notes in Computer Science, pages 78–88. Springer–Verlag, 2006.

[17] Johannes Buchmann, Andrei Pyshkin, and Ralf-Philipp Weinmann. Block Ciphers Sensitive to Gröbner Basis Attacks. In David Pointcheval, editor, *CT-RSA*, volume 3860 of *Lecture Notes in Computer Science*, pages 313–331. Springer, 2006.

[18] Leandro Caniglia, André Galligo, and Joos Heintz. Some New Effectivity Bounds in Computational Geometry. In Teo Mora, editor, *AAECC*, volume 357 of *Lecture Notes in Computer Science*, pages 131–151. Springer, 1988.

[19] Jung Hee Cheon, Seongtaek Chee, and Choonsik Park. S-boxes with Controllable Nonlinearity. In Jacques Stern, editor, *Advances in Cryptology – EUROCRYPT '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 286–294. Springer–Verlag, 1999.

[20] Carlos Cid, Sean Murphy, and Matt Robshaw. Small Scale Variants of the AES. In Helena Handschuh and Henri Gilbert, editors, *Fast Software Encryption – FSE 2005*, Lecture Notes in Computer Science, pages 145–162. Springer–Verlag, 2005.

[21] Carlos Cid, Sean Murphy, and Matthew Robshaw. *Algebraic Aspects of the Advanced Encryption Standard*, volume 148. Springer US, 2006.

[22] Stéphane Collart, Michael Kalkbrener, and Daniel Mall. Converting Bases with the Gröbner Walk. *Journal of Symbolic Computation*, 24(3/4):465–469, 1997.

[23] Nicolas Courtois. The Inverse S-box, Non-linear Polynomial Relations and Cryptanalysis of Block Ciphers. In Hans Dobbertin, Vincent Rijmen, and Aleksandra Sowa, editors, *AES 4 Conference*, volume 3373 of *Lecture Notes in Computer Science*, pages 170–188. Springer–Verlag, 2005.

[24] Nicolas Courtois and Josef Pieprzyk. Cryptanalysis of Block Ciphers with Overdefined Systems of Equations. In Yuliang Zheng, editor, *Advances in Cryptology – ASIACRYPT 2002*, volume 2501 of *Lecture Notes in Computer Science*, pages 267–287. Springer–Verlag, 2002.

[25] David A. Cox, John B. Little, and Don O'Shea. *Ideals, Varieties, and Algorithms*. Springer–Verlag, NY, 2nd edition, 1996. 536 pages.

[26] Joan Daemen, Lars Knudsen, and Vincent Rijmen. The block cipher Square. In Eli Biham, editor, *Fast Software Encryption – FSE 1997*, volume 1267 of *Lecture Notes in Computer Science*, pages 149–165. Springer–Verlag, 1997.

[27] Joan Daemen and Vincent Rijmen. AES Proposal: Rijndael.

[28] Joan Daemen and Vincent Rijmen. *The Design of Rijndael: The Wide Trail Strategy*. Springer–Verlag, 2001.

[29] Joan Daemen and Vincent Rijmen. The Wide Trail Design Strategy. In Bahram Honary, editor, *IMA Int. Conf.*, volume 2260 of *Lecture Notes in Computer Science*, pages 222–238. Springer, 2001.

[30] Hans Dobbertin. One-to-One Highly Nonlinear Power Functions on GF(2^n). *Applicable Algebra in Engineering, Communication and Computing*, 9(2):139–152, 1998.

[31] Jean-Charles Faugère. A New Efficient Algorithm for Computing Gröbner bases (F4). *Journal of Pure and Applied Algebra*, 139(1-3):61–88, June 1999.

[32] Jean-Charles Faugère. A New Efficient Algorithm for Computing Gröbner Bases without Reduction to Zero (F5). In *ISSAC*, pages 75–83. ACM, 2002.

[33] Jean-Charles Faugère. Gröbner Bases. Applications in Cryptology. Invited Talk at FSE'07 in Luxemburg. Available at `http://fse2007.uni.lu/slides/faugere.pdf`, March 2007.

[34] Jean-Charles Faugère and Gwénolé Ars. An Algebraic Cryptanalysis of Nonlinear Filter Generators using Gröbner bases. Technical report, INRIA, 2003.

[35] Jean-Charles Faugère, P. Gianni, Daniel Lazard, and Teo Mora. Efficient Computation of Zero-Dimensional Gröbner Bases by Change of Ordering. *Journal of Symbolic Computation*, 16(4):329–344, 1993.

[36] Jean-Charles Faugère and Antoine Joux. Algebraic Cryptanalysis of Hidden Field Equation (HFE) Cryptosystems Using Gröbner Bases. In Dan Boneh, editor, *CRYPTO*, volume 2729 of *Lecture Notes in Computer Science*, pages 44–60. Springer, 2003.

[37] Amir Hashemi and Daniel Lazard. Complexity of Zero-Dimensional Grbner bases. Technical Report RR-5660, INRIA, 2005.

[38] Amir Hashemi and Daniel Lazard. Sharper Complexity Bounds for Zero-dimensional Gröbner bases and Polynomial System Solving. Technical Report RR-5491, INRIA, 2005.

[39] Seokhie Hong, Sangjin Lee, Jongin Lim, Jaechul Sung, Dong Hyeon Cheon, and Inho Cho. Provable Security against Differential and Linear Cryptanalysis for the SPN Structure. In Bruce Schneier, editor, *FSE*, volume 1978 of *Lecture Notes in Computer Science*, pages 273–283. Springer, 2000.

[40] Thomas Jakobsen and Lars Knudsen. The Interpolation Attack on Block Ciphers. In Eli Biham, editor, *Fast Software Encryption – FSE 1997*, volume 1267 of *Lecture Notes in Computer Science*, pages 28–40. Springer–Verlag, 1997.

[41] Erich Kaltofen and Victor Shoup. Subquadratic-time Factoring of Polynomials over Finite FIelds. *Mathematics of Computation*, 67(223):1179–1197, 1998.

[42] Masayuki Kanda. Practical Security Evaluation against Differential and Linear Cryptanalyses for Feistel Ciphers with SPN Round Function. In Douglas R. Stinson and Stafford E. Tavares, editors, *Selected Areas in Cryptography – SAC 2000*, volume 2012 of *Lecture Notes in Computer Science*, pages 324–338. Springer–Verlag, 2001.

[43] Lars R. Knudsen. Practically Secure Feistel Ciphers. In Ross J. Anderson, editor, *Fast Software Encryption – FSE 1993*, volume 809 of *Lecture Notes in Computer Science*, pages 211–221. Springer–Verlag, 1994.

[44] Fen Liu, Wen Ji, Lei Hu, Jintai Ding, Shuwang Lv, Andrei Pyshkin, and Ralf-Philipp Weinmann. Analysis of the SMS4 Block Cipher. In Josef Pieprzyk, Hossein Ghodosi, and Ed Dawson, editors, *ACISP*, volume 4586 of *Lecture Notes in Computer Science*, pages 158–170. Springer, 2007.

[45] Stefan Mangard, Elisabeth Oswald, and Thomas Popp. *Power Analysis Attacks: Revealing the Secrets of Smart Cards*. Springer–Verlag, 2007.

[46] M. Matsui. Linear Cryptanalysis Method for DES Cipher. In Douglas R. Stinson, editor, *Advances in Cryptology – CRYPTO '93*, volume 773 of *Lecture Notes in Computer Science*, pages 386–387. Springer–Verlag, 1994.

[47] Sean Murphy and Matthew J.B. Robshaw. Essential Algebraic Structure within the AES. In Moti Yung, editor, *Advances in Cryptology – CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 1–16. Springer–Verlag, 2002.

[48] National Institute of Standards and Technology. FIPS-197: Advanced Encryption Standard, November 2001. Available at http://csrc.nist.gov/publications/fips/.

[49] Kaisa Nyberg. Differentially Uniform Mappings for Cryptography. In Tor Helleseth, editor, *Advances in Cryptology – EUROCRYPT '93*, volume 765 of *Lecture Notes in Computer Science*, pages 55–64. Springer–Verlag, 1994.

[50] Kai Schramm, Gregor Leander, Patrick Felke, and Christof Paar. A Collision-Attack on AES: Combining Side Channel- and Differential-Attack. In Marc Joye and Jean-Jacques Quisquater, editors, *CHES*, volume 3156 of *Lecture Notes in Computer Science*, pages 163–175. Springer, 2004.

[51] Kai Schramm, Thomas J. Wollinger, and Christof Paar. A New Class of Collision Attacks and Its Application to DES. In Thomas Johansson, editor, *FSE*, volume 2887 of *Lecture Notes in Computer Science*, pages 206–222. Springer, 2003.

[52] Makoto Sugita, Mitsuru Kawazoe, Ludovic Perret, and Hideki Imai. Algebraic Cryptanalysis of 58-Round SHA-1. In Alex Biryukov, editor, *FSE*, volume 4593 of *Lecture Notes in Computer Science*, pages 349–365. Springer, 2007.

[53] University of Sydney Computational Algebra Group. The Magma Computational Algebra System, 2004. `http://magma.maths.usyd.edu.au/magma/`.