Algebraic Cryptanalysis of Curry and Flurry Using Correlated Messages

Jean-Charles Faugère and Ludovic Perret

SALSA Project
INRIA, Centre Paris-Rocquencourt
UPMC, Univ Paris 06, LIP6
CNRS, UMR 7606, LIP6
104, avenue du Président Kennedy
75016 Paris, France
`jean-charles.faugere@inria.fr, ludovic.perret@lip6.fr`

Abstract. In this paper, we present an algebraic attack against the **Flurry** and **Curry** block ciphers [12,13]. Usually, algebraic attacks against block ciphers only require *one* message/ciphertext pair to be mounted. In this paper, we investigate a different approach. Roughly, the idea is to generate an algebraic system from the knowledge of several well chosen correlated message/ciphertext pairs. **Flurry** and **Curry** are two families of ciphers which fully parametrizable and having a sound design strategy against the most common statistical attacks; i.e. linear and differential attacks. These ciphers are then targets of choices for algebraic attacks. It turns out that our new approach permits to go one step further in the (algebraic) cryptanalysis of difficult instances of **Flurry** and **Curry**. To explain the behavior of our attack, we have established an interesting connection between algebraic attacks and high order differential cryptanalysis [32]. From extensive experiments, we estimate that our approach – that we will call "algebraic-high order differential" cryptanalysis – is polynomial when the Sbox is a power function. As a proof of concept, we have been able to break **Flurry/Curry** – up to 8 rounds – in few hours. We have also investigated the more difficult (and interesting case) of the inverse function. For such function, we have not been able to bound precisely the theoretical complexity, but our experiments indicate that our approach permits to obtain a significant practical gain. We have attacked **Flurry/Curry** using the inverse Sbox up to 8 rounds.

1 Introduction

A fundamental problem in cryptography is to evaluate the security of widely used cryptosystems against the most powerful techniques. To this end, several *general* methods have been proposed : linear cryptanalysis [34], differential cryptanalysis [8,9,10], *etc* . . . . *Algebraic cryptanalysis* can be described as a general framework that permits to asses the security of a wide range of cryptographic schemes [4,15,16,17,26,27,28,29]. As pointed in [20] "*the recent proposal and development of algebraic cryptanalysis is now widely considered an important breakthrough in* *the analysis of cryptographic primitives*. It is a powerful technique that applies potentially to a wide range of cryptosystems, in particular block ciphers.

The basic principle of such cryptanalysis is to model a cryptographic primitive by a set of algebraic equations. The system of equations is constructed in such a way as to have a correspondence between the solutions of this system, and a secret information of the cryptographic primitive (for instance, the secret key of a block cipher). This line of research is somehow inspired by C.E. Shannon who stated that: "*Breaking a good cipher should require as much work as solving a system of simultaneous equations in a large number of unknowns of a complex type.*"(Commu–nication Theory of Secrecy Systems, 1949). Shannon relates then the security of a cryptosystem to the difficulty of solving a set of algebraic equations, and lays then the foundation of algebraic cryptanalysis.

In theory, any cryptosystem can be modeled by a set of algebraic equations over a finite field [30]. In fact, it is usual that the same cryptographic primitive can be described by several algebraic systems. However, it is an open research problem how to optimally model a cryptosystem so that it is easiest to solve. Thus, it is one of the most crucial aspects of algebraic cryptanalysis to derive the best system with respect to equations solving.

Algebraic techniques have been successfully applied against a number of multivariate schemes and in stream cipher cryptanalysis [4,15,16,17,26,27,28,29]. On the other hand, its feasibility against block ciphers remains the source of speculations [19,21,33,4,3]. The main problem is that the size of the corresponding algebraic system is so huge (thousand of variables and equations) that nobody is able to predict correctly the complexity of solving such polynomial systems.

However, it is worth to remark that the algebraic systems are huge but highly structured. The equations are very sparse and the round structure of the block ciphers implies a similar structure on the algebraic equations. Secondly, there is not a unique algebraic description of a cryptographic primitive. Although it is an open issue how to optimally model a cryptosystem, it is crucial to use this degree of freedom to derive the best system with respect to equations solving.

Typically, algebraic cryptanalysis against block ciphers only requires *one* message/ciphertext pair to be mounted. In this paper, we present a novel approach. The basic idea is to generate an algebraic system from the knowledge of several well chosen correlated message/ciphertext pairs. It turns our that this system is easier to solve in practice. To explain this behavior, we have established an interesting connection between algebraic attacks and high order differential cryptanalysis [31,32].Interesting enough, this is on the line of a new trend in algebraic cryptanalysis which is to combine statistical and algebraic techniques. Albrecht and Cid [1,2] recently proposed to mix differential and algebraic cryptanalysis to attack PRESENT.

As already explained, the algebraic cryptanalysis of a block cipher usually leads to huge systems of equations. For this reason, it makes sense to first experiment such attacks on "scalable" block ciphers. For this reason, Cid, Murphy, and Robshaw [18] described small scale variants of AES. In the same vain, Buchmann, Pyshkin and Weinmann [12,13] described two families of Feistel (**Flurry**) and SPN (**Curry**) block ciphers which are fully parametrizable. The main goal of **Curry** and **Flurry** was probably to be relevant families of ciphers for experimenting algebraic attacks. To do so, the encryption process of these ciphers can be easily described by a set of algebraic equations. On the other hand, these ciphers have a sound design strategy against linear [34] and differential [8,9,10] attacks. The aim was to mimic as much as possible the design criteria of widely used modern block ciphers. Therefore, any new successful algebraic cryptanalysis against **Curry**/**Flurry** can be a step toward more efficient algebraic cryptanalysis against industrial block ciphers such as AES.

1.1 Organization of the Paper: Main Results

After this introduction, the paper is organized as follows. In Section 2, we introduce the families of Feistel and SPN block ciphers **Flurry** and **Curry** respectively [12,13]. We briefly recall some security features of the ciphers.

In this last section (Section 3), we will present results that we have obtained when mounting two refined algebraic attack strategies. First, we show that the use of a sparse version of FGLM [23] permits to obtain a practical gain w.r.t. to the attack presented by Buchmann, Pyshkin and Weinmann [12]. However, this attack remains limited since its theoretical complexity is exponential in the number of rounds and the size of the plaintext space.

To overcome this limitation, we have investigated the possibility of using a small amount of suitably chosen message/ciphertext pairs to improve the efficiency of algebraic attacks. Precisely, we propose to use correlated messages. It appears that this approach permits to go one step further in the (algebraic) cryptanalysis of of difficult instances of **Flurry** and **Curry**. To explain the behavior of our attack, we have established an interesting connection between algebraic attacks and high order differential cryptanalysis [32]. From extensive experiments, we estimate that our approach – that we will call "algebraic-high order differential" cryptanalysis – is polynomial when the Sbox is a power function. As a proof of concept, we have been able to break **Flurry**/**Curry** – up to 8 rounds – in few hours. We have also thoroughly investigated the inverse function. Although we have not been able to bound precisely the theoretical complexity, our experiments indicate that our approach permits to obtain a significant practical gain. We have been able to break **Flurry**/**Curry** using the inverse Sbox up to 8 rounds.

2 The Flurry and Curry Block Ciphers

In this part, we describe the main concern of this paper, namely **Curry** and **Flurry** [12,13]. We will briefly recall the algebraic description of such ciphers, and highlight some security features of these ciphers.

In the rest of this paper $\mathbb{K} = \mathbb{F}_2(\theta)$ will denote a finite field of size $k = 2^n, n \in \{8, 16, 32, 64\}$. The number of rounds will be denoted by $r \in \mathbb{N}$. $D \in \mathcal{M}_{m\times m}(\mathbb{K})$ is a matrix describing the linear diffusion mapping of the round function. This matrix D is also used in the key scheduling. We refer to [12] for the exact description of these matrices, but we mention that these matrices have been chosen to have an optimal diffusion. Finally, f is a non-linear function describing the Sbox chosen as the power function $f(x) = f_p(x) : x \in \mathbb{K} \mapsto x^p \in \mathbb{K}$, with $p \in \{3, 5, 7\}$, or the inverse function $f(x) = f_{\text{inv}}(x) : x \in \mathbb{K} \mapsto x^{k-2} \in \mathbb{K}$.

2.1 The Feistel Case: Flurry

First we describe the family of Feistel ciphers **Flurry**(n, t, r, f, D), with $t \in \mathbb{N}$ being the size of a message block. To add to this confusion, we will also use the notation $m = \frac{t}{2}$ for the half-size of a block (t is assumed to be even).

We will denote by $L = (\ell_1, \ldots, \ell_m) \in \mathbb{K}^m$ $\big($resp. $R = (r_1, \ldots, r_m) \in \mathbb{K}^m$ $\big)$ the left (resp. right) part of the current state, and by $K = (k_1, \ldots, k_m) \in \mathbb{K}^m$ a key. The round function $T : \mathbb{K}^m \times \mathbb{K}^m \times \mathbb{K}^m \to \mathbb{K}^m \times \mathbb{K}^m$ is then defined as :

$$T(L, R, K) = \big(R, \big(f(r_1 + k_1), \ldots, f(r_m + k_m)\big) \cdot D + L\big) .$$

Let $K = (K_0, K_1) \in \mathbb{K}^m \times \mathbb{K}^m$ be the initial key. The subkey used at round $i, 2 \le i \le r + 1$ is :

$$K_i = K_{i-1} \cdot D + K_{i-2} + v_i,$$

where $v_i = \big((\theta + 1)^i, (\theta + 1)^{i+1}, \ldots, (\theta + 1)^{i+m-1}\big) \in \mathbb{K}^m$.

A message $m = (L_0, R_0) \in \mathbb{K}^m \times \mathbb{K}^m$ is encrypted into a ciphertext $c = (L_r, R_r) \in \mathbb{K}^m \times \mathbb{K}^m$ by iterating the round function T as follows :

$$\begin{aligned} (L_i, R_i) &= T(L_{i-1}, R_{i-1}, K_{i-1}), \text{ for all } i, 1 \le i \le r - 1, \\ c = (L_r, R_r) &= T(L_{r-1}, R_{r-1}, K_{r-1}) + (K_r, K_{r+1}). \end{aligned}$$

It is not difficult to describe the encryption process by a set of algebraic equations. Actually, this was a design policy [12,13]. To keep the degree as low a possible, we have to introduce new variables :
– $\{x_{i,j}\}_{1 \le i \le (r-1)}^{1 \le j \le t}$ corresponding to the internal states of the cipher,
– and $\{k_{i,j}\}_{1 \le i \le r+1}^{1 \le j \le m}$ corresponding to the initial/expanded key.
We will denote by $\mathcal{R}_{\text{Flurry}}$ the polynomial ring $\mathbb{K}\big[\{x_{i,j}\}_{1 \le i \le (r-1)}^{1 \le j \le t}, \{k_{i,j}\}_{1 \le i \le r+1}^{1 \le j \le m}\big]$. For a pair plaintext/ciphertext $(m, c) \in \mathbb{K}^t \times \mathbb{K}^t$, we will denote by : $\mathcal{P}_{\text{Flurry}}(m, c) \subset \mathcal{R}_{\text{Flurry}}$, the set of all algebraic equations describing a **Flurry** encryption process.

2.2 The SPN Case: Curry

Curry(n, m, r, f, D) is a family of SPN ciphers which is also fully parametrizable. The plaintext, ciphertext and secret key spaces are space dimension is $m \in \mathbb{N}$ are $\mathbb{K}^{m \times m}$. The round function $T : \mathbb{K}^{m \times m} \times \mathbb{K}^{m \times m} \to \mathbb{K}^{m \times m}$ of **Curry** is given by :

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

with $G : X = \{x_{i,j}\} \in \mathbb{K}^{m \times m} \to G(X) = \{f(x_{i,j})\} \in \mathbb{K}^{m \times m}$ being the parallel application of the SBox function f to the components of a matrix $X \in \mathbb{K}^{m \times m}$.

A plaintext $m = S_0 \in \mathbb{K}^{m\times m}$ is transformed into a ciphertext $c \in \mathbb{K}^{m\times m}$ by iterating the round function T exactly r times followed by a last key addition :

$$\begin{aligned} S_\ell &= T(S_{\ell-1}, K_{\ell-1}), \text{ for all } \ell, 1 \le \ell \le r-1,\\ c = S_r &= T(S_{r-1}, K_{r-1}) + K_r. \end{aligned}$$

The master key $K_0 \in K^{m\times m}$ is used at the first round; subsequent round keys $K_i, i \ge 1$ are computed using the formula :

$$K_i = K_{i-1} \cdot D + M_i,$$

with $M_i = \{\theta^{i+(j-1)m+k}\}_{1\le j,k\le m} \in \mathbb{K}^{m\times m}$ being a (constant) matrix depending of the round.

As for **Flurry**, there is no problem to construct a set of algebraic equations modeling the encryption process of **Curry**. We have to introduce new variables : $\{x_{i,j}^\ell\}_{1\le i,j\le m}^{1\le \ell\le (r-1)}$ corresponding to the internal states of the cipher, and $\{k_{i,j}^\ell\}_{1\le \ell\le r}^{1\le i,j\le m}$ corresponding to the initial/expanded key.

Using an obvious notation, $\mathcal{R}_{\text{Curry}}$ will denote $\mathbb{K}\Big[\{x_{i,j}^\ell\}_{1\le i,j\le m}^{1\le \ell\le (r-1)}, \{k_{i,j}^\ell\}_{1\le \ell\le r}^{1\le i,j\le m}\Big]$, and $\mathcal{P}_{\text{Curry}}(m,c) \subset \mathcal{R}_{\text{Curry}}$, the set of algebraic equations describing **Curry** $\big($for a plaintext/ciphertext $(m,c)\big)$.

3 Improved Algebraic Attacks against Curry and Flurry

This section is divided into two parts. First, we show that the Buchmann, Pyshkin and Weinmann (BPW) attack against **Curry** and **Flurry** [12] can be improved using a "fast version" of FGLM. The goal is to illustrate the gain that we can obtain using sparse algebra techniques. This will permit to have a tight complexity estimates of the BPW attack, and then a good basis to comparison with others attacks. In particular with respect to the practical behavior of the new attack presented in the second part of this section.

3.1 Practical Improvements of the Buchmann, Pyshkin and Weinmann Attack

We start by recalling a surprising result of to Buchmann, Pyshkin and Weinmann [12,13]. They proved that for a well chosen ordering $\prec^*$, the polynomials of $\mathcal{P}_{\text{Flurry}}$ and $\mathcal{P}_{\text{Curry}}$ already form a Gröbner basis [22,11]. It has to be noted that a similar result holds for AES-128 [14]. The key-recovery problem is then reduced to changing the order of a Gröbner basis. More precisely, solving $\mathcal{P}_{\text{Flurry}}$ (resp. $\mathcal{P}_{\text{Curry}}$) is equivalent to compute a Lex-Gröbner basis knowing a $\prec^*$–Gröbner basis. This can be done by using FGLM [23], and a precise complexity estimates can be then given [23,12,13].

Lemma 1. *Let $\mathcal{I}_{\text{Flurry}}$ (resp. $\mathcal{I}_{\text{Curry}}$) be the ideal generated by the polynomials of $\mathcal{P}_{\text{Flurry}}$ (resp. $\mathcal{P}_{\text{Curry}}$). It holds that :*

$$\dim_{\mathbb{K}}\left(\mathcal{R}_{\text{Flurry}}/\mathcal{I}_{\text{Flurry}}\right) = \deg(f)^{t\cdot r},\ \textit{for } \textbf{\textit{Flurry}}(n,t,r,f,D)$$
$$\dim_{\mathbb{K}}\left(\mathcal{R}_{\text{Curry}}/\mathcal{I}_{\text{Flurry}}\right) = \deg(f)^{m^2\cdot r}, \textit{for } \textbf{\textit{Curry}}(n,m,r,f,D).$$

These results are not valid if f is the inverse function [12].

The complexity of FGLM is polynomial in the dimension of the quotient. We can use sparse linear algebra [35] techniques to decrease the exponent and obviously improve the efficiency of FGLM. To illustrate this fact, we will present experimental results. In the table below, we have quoted:
– the practical results obtained in [12,13] using FGLM. The authors have used the version available in Magma (version 2.11-8).
– the dimension $\dim_{\mathbb{K}}(\mathcal{R}/\mathcal{I})$ of the quotient.
– The timings obtained with our sparse version of FGLM. This version of the algorithm has been implemented in C within the FGb software[1].

		[12]	This paper
Flurry(n,t,r,f,D)	$\dim_{\mathbb{K}}(\mathcal{R}/\mathcal{I})$	F_4+FGLM (Magma)	"fast" FGLM (Fgb)
Flurry$(64,2,4,f_3,I_1)$	3^4	< 0.1 s.	< 0.1 s.
Flurry$(64,2,4,f_5,I_1)$	5^4	2.3 s.	< 0.1 s.
Flurry$(64,2,4,f_7,I_1)$	7^4	82.62 s.	19.4 s.
Flurry$(64,2,6,f_3,I_1)$	3^6	145.08 s.	2.1 s.

Interpretation of the Results. We observe that there is a non-negligible practical gain when using a sparse version of FGLM. Anyway, this approach becomes quickly impractical due to huge dimension of $\mathcal{R}_{\text{Flurry}}/\mathcal{I}_{\text{Flurry}}$ (resp. $\mathcal{R}_{\text{Curry}}/\mathcal{I}_{\text{Curry}}$). This is mainly due to the fact that the field equations are not included in $\mathcal{P}_{\text{Flurry}}$ (resp. $\mathcal{P}_{\text{Curry}}$). Therefore, the variety associated with these systems will mostly contain spurious solutions (solutions over the algebraic closure of $\mathbb{K}$). However, this is to our knowledge the best (algebraic) attacks proposed so far against **Flurry** and **Curry**. We will now present an alternative approach for attacking these ciphers.

3.2 On the Use of Several Plaintext/Ciphertext Pairs

The key recovery systems $\mathcal{P}_{\text{Flurry}}$ and $\mathcal{P}_{\text{Curry}}$ are constructed from the knowledge of only one plaintext/ciphertext pair (Section 2). In this part, we investigate the possibility of using few pairs of plaintext/ciphertext. Namely, we select $N > 1$ messages $m_1,\ldots,m_N$ and request the corresponding ciphertexts $c_1,\ldots,c_N$. Instead of trying to solve each system $\mathcal{P}_{\text{Flurry}}(m_i,c_i)$ individually, the idea is to solve a new key recovery system: $\mathcal{P}^N_{\text{Flurry}} = \bigcup_{i=1}^N \mathcal{P}_{\text{Flurry}}(m_i,c_i)$. Obviously, the secret key will be also a solution of this larger system. The set of equations $\mathcal{P}^N_{\text{Curry}}$ is defined similarly.

Note that for each pair (m_i,c_i), we have to introduce new variables corresponding to the internal states of the cipher. On the other hand, the variables

[1] `http://fgbrs.lip6.fr/jcf/Software/FGb/index.html`

corresponding to the key will remain the same for each pair (m_i, c_i). Again, we emphasize that the field equations are not included in the systems. Remark that the result described previously (Sec. 3.1) for Flurry/Curry systems only holds when $N = 1$.

Using N random messages. Unfortunately, when we select *randomly* N messages $m_1, \ldots, m_N$, this approach will not lead to any improvement w.r.t. to the use of a single plaintext/ciphertext pair. Even worse, we have observed that the new systems are harder to solve in practice. To illustrate this fact, we have quoted few results that obtained for $\mathbf{Flurry}(8, 2, 2, D_2, f_{\text{inv}})$ with different values of N. The F_5 [25] and FGLM [23] algorithms have been implemented in C within the FGb software. We used this implementation for computing Gröbner bases. We have included the time T necessary for solving the systems and the total number of solutions Nb_{sol}.

N	1	2	3
T	0.43 s.	25.8 s.	16 m. 42 s.
Nb_{sol}	184	1	1

We have increased the number of equations/variables (corresponding to intermediate states), but the maximum degree reached during the Gröbner basis computation remains stable, i.e. the systems are not easier to solve. Interestingly enough, as soon as $N > 1$, the variety associated to $\mathcal{P}^N_{\text{Flurry}}$ (resp. $\mathcal{P}^N_{\text{Curry}}$) contains – most of the time – only one solution corresponding to the secret key; thus only one direct (DRL) Gröbner basis computation is needed.

3.3 Algebraic-High Order Differential Style Cryptanalysis

The difficulty is to find a suitable way to incorporate the additional knowledge of several message/ciphertext pairs. Or, stated differently, how to choose such pairs to improve the efficiency of the solving step. To do so, we have considered a classical cryptatanalytic tool, namely high order differentials [32,31], as a filter to select *correlated messages*. This will permit to decrease the complexity of the solving step. To explain the intuition behind our approach, we introduce few new definitions. The *derivative* (or *finite difference*) of a mapping $f : \mathbb{K}^n \mapsto \mathbb{K}^m$ at a point $r \in \mathbb{K}^n$ is defined as follows: $\Delta_r f(x) = f(x + r) - f(x)$. Remark that if the components of f are of an algebraic degree d, then the components of $\Delta_r f(x)$ will be of an algebraic degree $\leq d$. To further decrease this degree, we can consider the ith derivative of f at points $r_1, \ldots, r_i$ which is recursively [32] defined as: $\Delta^{(i)}_{r_1,\ldots,r_i} f(x) = \Delta_{r_i}\left(\Delta^{(i-1)}_{r_1,\ldots,r_{i-1}} f(x)\right)$.

Now, let $L[r_1, \ldots, r_i]$ be the set of all binary linear combinations of the $r_1, \ldots, r_i$. It is well known that: $\Delta^{(i)}_{r_1,\ldots,r_i} f(x) = \sum_{\delta \in L[r_1,\ldots,r_i]} f(x + \delta)$. We can now explain how we have generated correlated the messages.

First, we will consider the most basic case, i.e. $N = 2$. We randomly select a message m_0, a difference r_1, and construct the key recovery system :

$$\mathcal{P}^2 = \mathcal{P}(m_0, c) \cup \mathcal{P}(m_0 + r_1, c'), \text{ with } \mathcal{P} \in \{\mathcal{P}_{\text{Flurry}}, \mathcal{P}_{\text{Curry}}\} .$$

Now, let $T(\mathbf{X_i}, K_i) = T_{K_i}(\mathbf{X_i})$ be the round function (for **Flurry** $X_i \in \mathbb{K}^t$, and for **Curry** $X_i \in \mathbb{K}^{m\times m}$) of **Flurry** or **Curry** (K_i is the subkey used at round i). For the first round, we have :

$$\mathbf{X_1}^{(0)} - T_{K_1}(m_0) = \mathbf{0} \in \mathcal{P}^2, \tag{1}$$

$$\mathbf{X_1}^{(1)} - T_{K_1}(m_0 + r_1) = \mathbf{0} \in \mathcal{P}^2. \tag{2}$$

$\mathbf{X_1}^{(0)}$ (resp. $\mathbf{X_1}^{(1)}$) being the first intermediate variables corresponding to m_0 (resp. $m_1 = m_0 + r_1$). From now on, a boldfaced letter will refer to a vector.

From (1) and (2), we deduce that the equations $\mathbf{X_1}^{(1)} - \mathbf{X_1}^{(0)} = \Delta_{r_1} T_{K_1}(m_0)$ are in the ideal generated by $\mathcal{P}^2$. Thus, by simply taking two pairs, we have created new equations relating the intermediates variables $\mathbf{X_1}^{(1)}, \mathbf{X_1}^{(0)}$, and the variables corresponding to K_1. The new equations are of degree strictly smaller than the initial equations of $\mathcal{P}^2$. We can iterate the process. Let $r_1, \ldots, r_N$ be a set of $N \geq 2$ linearly dependent vectors, and m_0 be a random message. We consider now the system :

$$\mathcal{P}^N = \bigcup_{r \in L[r_1,\ldots,r_N]} \mathcal{P}(m_0 + r, c_r), \text{ with } \mathcal{P} \in \{\mathcal{P}_{\text{Flurry}}, \mathcal{P}_{\text{Curry}}\}.$$

We will denote by $\mathbf{X_i}^{(j)}$ the intermediates variables used at the ith round and corresponding to the jth message[2], and c_r will be the encryption of $m_0 + r$. For the first round, we have that for all $k, 1 \leq k \leq \#L[r_1, \ldots, r_N]$:

$$\mathbf{X_1}^{(k)} - \mathbf{X_1}^{(0)} = \Delta_r T_{K_1}(m_0) \in \mathcal{P}^N, \text{ with } r \in L[r_1, \ldots, r_N].$$

As previously, we have created new low-degree equations corresponding to derivatives. But, we will also generate low-degree equations corresponding to high order derivatives. For instance, let $r_1, r_2 \in L[r_1, \ldots, r_N]$. It holds that:

$$\mathbf{X_1}^{(0)} - T_{K_1}(m_0) = \mathbf{0} \in \mathcal{P}^N, \qquad \mathbf{X_1}^{(1)} - T_{K_1}(m_0 + r_1) = \mathbf{0} \in \mathcal{P}^N,$$
$$\mathbf{X_1}^{(2)} - T_{K_1}(m_0 + r_2) = \mathbf{0} \in \mathcal{P}^N, \qquad \mathbf{X_1}^{(3)} - T_{K_1}(m_0 + r_1 + r_2) = \mathbf{0} \in \mathcal{P}^N.$$

Therefore, $\mathbf{X_1}^{(3)} - \sum_{k=0}^{2} \mathbf{X_1}^{(k)} = \Delta_{r_1,r_2} T_{K_1}(m_0) \in \mathcal{P}^N$. The ideal generated by $\mathcal{P}^N$ will now include linear relations between the intermediates variables $\mathbf{X_1}^{(j)}$. Moreover, such new linear equations will induce derivatives and high order derivatives in the subsequent rounds of the cipher. In our case, we know that $\mathbf{X_1}^{(3)} = \sum_{k=0}^{2} \mathbf{X_1}^{(k)}$. If we consider the second round $\mathbf{X_2}^{(0)} = T_{K_2}(\mathbf{X_1}^{(0)}) \in \mathcal{P}^N$, and $\mathbf{X_2}^{(3)} = T_{K_2}(\mathbf{X_1}^{(0)} + \mathbf{X_1}^{(1)} + \mathbf{X_1}^{(2)}) \in \mathcal{P}^N$. The equations $\mathbf{X_2}^{(3)} - \mathbf{X_2}^{(0)} = \Delta_{\mathbf{X_1}^{(1)} + \mathbf{X_1}^{(2)}} T_{K_2}(\mathbf{X_1}^{(0)})$ is then in the ideal generated by $\mathcal{P}^N$. In function of N, this phenomena will be propagated throughout the rounds of the cipher to generate high order differentials, and so new low-degree equations between the intermediates variables. This permits to establish an interesting

[2] We supose w.l.o.g. an explicit ordering on the elements of $L[r_1, \ldots, r_N]$.

connection between algebraic attacks and high order differential cryptanalysis [32,31]; that we can call "algebraic-high order differential" cryptanalysis.

Experimental results. To summarize, our attack works as follows. Let $N > 1$ be an integer. We fix $m_0 = (0, \ldots, 0)$, and $r_1 = (1, \ldots, 0)$. We construct differences r_i, for all $i, 2 \leq i \leq N$, using the relation $r_{i+1} = \theta \cdot r_i$. After that, we have to solve the system :

$$\mathcal{P}^N = \bigcup_{r \in U \subset L[r_1,\ldots,r_N]} \mathcal{P}(m_0 + r, c_r), \text{ with } \mathcal{P} \in \{\mathcal{P}_{\text{Flurry}}, \mathcal{P}_{\text{Curry}}\},$$

c_r being the encryption corresponding to $m_0 + r$. Note that in practice, we have not considered all the 2^N elements of $L[r_1, \ldots, r_N]$ but a smaller subset $U \subset L[r_1, \ldots, r_N]$ of size N. As explained previously, the ideal generated $\mathcal{P}^N$ includes many new low-degree equations corresponding to all the derivatives of order less than $\lfloor \ln(N) \rfloor$. We expect that these new equations will allow the ease the Gröbner basis computation. We will see that this is indeed the case in practice.

In next table, we have quoted the results we have obtained on **Flurry** and **Curry** with different values of N. Note that most of the parameters have been chosen for having a secret key of 128-bit. In this case, the ciphers considered are immune against differential/linear attacks when the number of rounds is ≥ 4 [12,13]. The experimental results have been obtained with a cluster of Xeon bi-processors 3.2 Ghz, with 64 Gb of Ram. It is well known that the efficiency of the Gröbner basis computation can vary in function of the order used. In our experiments, we have used the order proposed in [12,13]. In the table, we have included :
– T : total time of our attack; Nb_{op} : number of basic operations;
Mem : Maximum memory usage;
$D_{\max}$: the maximal degree reached during the Gröbner basis computation.
T_{BPW} : estimated complexity of the attack of [12]. Remark that this attack can not be mounted for f_{-1}.

Flurry(n, t, r, f, D)	T_{BPW}	N	$D_{\max}$	T	Nb_{op}	Mem
Flurry$(16, 2, 6, f_{-1}, I_1)$		3	3	0.6 s.	2^{25}	1.8 Gb.
Flurry$(16, 2, 7, f_{-1}, I_1)$		3	4	0.4 s.	2^{24}	1 Gb.
Flurry$(16, 2, 8, f_{-1}, I_1)$		4	4	37.6 s.	2^{31}	1.4 Gb.
Flurry$(16, 2, 9, f_{-1}, I_1)$		10	4	37296 s.	2^{41}	6.4 Gb.
Flurry$(16, 4, 5, f_{-1}, D_2)$		2	4	0.5 s.	$2^{24.2}$	1.7 Gb.
Flurry$(16, 4, 6, f_{-1}, D_2)$		4	4	810.3 s.	$2^{36.0}$	4.6 Gb.
Flurry$(16, 8, 5, f_{-1}, D_4)$		3	4	3755.2 s.	$2^{37.5}$	5.4 Gb.
Flurry$(16, 4, 6, f_3, D_2)$	$\approx 2^{114}$	14	3	3.4 s.	$2^{27.4}$	1.3 Gb.
Flurry$(16, 4, 8, f_3, D_2)$	$\approx 2^{152}$	**90**	3	1952 s.	$2^{36.1}$	117 Gb.
	$\approx 2^{152}$	100	3	2058 s.	$2^{36.2}$	130 Gb.
Flurry$(16, 8, 6, f_3, D_4)$	$\approx 2^{228}$	20	3	35.8 s.	$2^{26.1}$	47 Gb.

Curry(n, m, r, f, D)	T_{BPW}	N	$D_{\max}$	T	Nb_{op}	Mem
Curry$(32, 4, 3, f_3, D_2)$	$\approx 2^{57}$	2	10	0.01 s.	$2^{12.7}$	2.4 Gb.
	$\approx 2^{57}$	17	6	0.01 s.	$2^{14.5}$	18.8 Gb.
	$\approx 2^{57}$	**20**	3	0.01 s.	$2^{11.5}$	2.1 Gb.

Interpretation of the Results. For power Sboxes, we can observe that our approach is significantly faster than BPW attack [12]. The most important is to observe that – in all cases – we have been able to find a number of pairs $N^* > 1$ such that the maximal degree reached during the Gröbner basis computation is equal to the degree d of the Sbox function. In practice, we have found this N^* by performing the following test incrementally on $N \geq 2$. We compute a DRL Gröbner basis of $\mathcal{P}^N_{\text{Flurry}}$ (resp. $\mathcal{P}^N_{\text{Curry}}$). If the maximal degree reached during this computation is greater than d then we stop the computation and set $N \leftarrow N+1$, otherwise $N \leftarrow N^*$. We can extrapolate the (experimental) complexity of our attack. Let b_{F} (resp. b_{C}) be the number of variables of the system $\mathcal{P}^{N^*}_{\text{Flurry}}$ (resp. $\mathcal{P}^{N^*}_{\text{Curry}}$). Let $\omega, 2 < \omega \leq 3$ be the linear algebra constant. The complexity of our attack is :

– $\mathcal{O}\left(b_{\text{F}}^{\deg(f)\cdot\omega}\right)$, for $\mathbf{Flurry}(n,t,r,f,D)$, and
– $\mathcal{O}\left(b_{\text{C}}^{\deg(f)\cdot\omega}\right)$, for $\mathbf{Curry}(n,m,r,f,D)$

We have then a polynomial time complexity for solving **Flurry** and **Curry** for (pure) power Sboxes. These results are no longer valid for the inverse function. For this Sbox, the maximum degree reached during the computation is more difficult to predict. On the other hand, we observed that our technique permits to have a significant gain of efficiency also in this case. The use of correlated messages has permitted to go one step further in the algebraic cryptanalysis of hard instances of **Flurry** instantiated with the inverse SBox. However, it is not clear that there exists an optimal number of correlated messages N^* such that the maximal degree reached during the Gröbner basis computation is bounded by a constant (for instance, 3 or 4). This deserves further investigations.

Conclusion. As explained in the introduction, a new trend in algebraic cryptanalysis is to combine statistical and algebraic techniques. Albrecht and Cid [1,2] recently proposed to mix differential and algebraic cryptanalysis. Note that there is a fundamental difference between our approach and the technique of Albrecht and Cid [1,2]. In this work, the equations derived from high order differentials are automatically (explicitly) generated during the Gröbner basis computation. In [1,2], linear equations derived from the knowledge of a differential are explicitly added to a key recovery system.

References

1. Albrecht, M., Cid, C.: Algebraic Techniques in Differential Cryptanalysis, `http://eprint.iacr.org/2007/137`
2. Albrecht, M., Cid, C.: Algebraic Techniques in Differential Cryptanalysis. In: Proceedings of the First International Conference on Symbolic Computation and Cryptography, SCC 2008, Beijing, China (April 2008)
3. Ars, G., Faugère, J.-C., Imai, H., Kawazoe, M., Sugita, M.: Comparison Between XL and Gröbner Basis Algorithms. In: Lee, P.J. (ed.) ASIACRYPT 2004. LNCS, vol. 3329, pp. 338–353. Springer, Heidelberg (2004)

4. Ars, G.: Applications des Bases de Gröbner à la Cryptographie. Thèse de doctorat, Université de Rennes I (2004)
5. Bardet, M.: Étude des systèmes algébriques surdéterminés. Applications aux codes correcteurs et à la cryptographie. Thèse de doctorat, Université de Paris VI (2004)
6. Bardet, M., Faugère, J.-C., Salvy, B.: On the Complexity of Gröbner Basis Computation of Semi-Regular Overdetermined Algebraic Equations. In: Proc. of International Conference on Polynomial System Solving (ICPSS), pp. 71–75 (2004)
7. Bardet, M., Faugère, J.-C., Salvy, B., Yang, B.-Y.: Asymptotic Behaviour of the Degree of Regularity of Semi-Regular Polynomial Systems. In: Proc. of MEGA 2005, Eighth Inter. Symposium on Effective Methods in Algebraic Geometry (2005)
8. Biham, E., Shamir, A.: Differential Cryptanalysis of DES-like Cryptosystems. In: Menezes, A., Vanstone, S.A. (eds.) CRYPTO 1990. LNCS, vol. 537, pp. 2–21. Springer, Heidelberg (1991)
9. Biham, E., Shamir, A.: Differential Cryptanalysis of DES-like Cryptosystems. Journal of Cryptology 4(1), 3–72 (1991)
10. Biham, E., Shamir, A.: Differential Cryptanalysis of of the Full 16-round DES. In: Brickell, E.F. (ed.) CRYPTO 1992. LNCS, vol. 740, pp. 487–496. Springer, Heidelberg (1993)
11. Buchberger, B.: Ein algorithmisches Kriterium fur die Lösbarkeit eines algebraischen Gleichungssystems (An Algorithmical Criterion for the Solvability of Algebraic Systems of Equations). Aequationes mathematicae 4(3), 374–383 (1970); English translation in: Buchberger, B., Winkler, F. (eds.) Grobner Bases and Applications. In: Proceedings of the International Conference 33 Years of Gröbner Bases, RISC, Austria. Lecture Note Series, vol. 251, pp. 535–545. London Mathematical Society, Cambridge University Press (1998)
12. Buchmann, J., Pyshkin, A., Weinmann, R.-P.: Block Ciphers Sensitive to Gröbner Basis Attacks. In: Pointcheval, D. (ed.) CT-RSA 2006. LNCS, vol. 3860, pp. 313–331. Springer, Heidelberg (2006)
13. Buchmann, J., Pyshkin, A., Weinmann, R.-P.: Block Ciphers Sensitive to Gröbner Basis Attacks – Extended version, `http://eprint.iacr.org/2005/200`
14. Buchmann, J., Pyshkin, A., Weinmann, R.-P.: A Zero-Dimensional Gröbner Basis for AES-128. In: Robshaw, M.J.B. (ed.) FSE 2006. LNCS, vol. 4047, pp. 78–88. Springer, Heidelberg (2006)
15. Courtois, N., Pieprzyk, J.: Cryptanalysis of Block Ciphers with Overdefined Systems of Equations. In: Zheng, Y. (ed.) ASIACRYPT 2002. LNCS, vol. 2501, pp. 267–287. Springer, Heidelberg (2002)
16. Courtois, N., Meier, W.: Algebraic Attacks on Stream Ciphers with Linear Feedback. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 345–359. Springer, Heidelberg (2003)
17. Courtois, N.: Fast Algebraic Attacks on Stream Ciphers with Linear Feedback. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 176–194. Springer, Heidelberg (2003)
18. Cid, C., Murphy, S., Robshaw, M.J.B.: Small Scale Variants of the AES. In: Gilbert, H., Handschuh, H. (eds.) FSE 2005. LNCS, vol. 3557, pp. 145–162. Springer, Heidelberg (2005)
19. Cid, C., Murphy, S., Robshaw, M.J.B.: Algebraic Aspects of the Advanced Encryption Standard. Springer, Heidelberg (2006)

20. Cid, C., Albrecht, M., Augot, D., Canteaut, A., Weinmann, R.-P.: Algebraic Cryptanalysis of Symmetric Primitives. In: Deliverable STVL, ECRYPT - European Network of Excellence Collaborations (July 2008), `http://hal.archives-ouvertes.fr/docs/00/32/86/26/PDF/D-STVL-7.pdf`
21. Cid, C., Leurent, G.: An Analysis of the XSL Algorithm. In: Roy, B. (ed.) ASIACRYPT 2005. LNCS, vol. 3788, pp. 333–352. Springer, Heidelberg (2005)
22. Cox, D.A., Little, J.B., O'Shea, D.: Ideals, Varieties, and algorithms: an Introduction to Computational Algebraic Geometry and Commutative algebra. In: Undergraduate Texts in Mathematics. Springer, New York (1992)
23. Faugère, J.C., Gianni, P., Lazard, D., Mora, T.: Efficient Computation of Zero-Dimensional Gröbner Bases by Change of Ordering. Journal of Symbolic Computation 16(4), 329–344 (1993)
24. Faugère, J.-C.: A New Efficient Algorithm for Computing Gröbner Basis: F_4. Journal of Pure and Applied Algebra 139, 61–68 (1999)
25. Faugère, J.-C.: A New Efficient Algorithm for Computing Gröbner Basis without Reduction to Zero: F_5. In: Proceedings of ISSAC, pp. 75–83. ACM Press, New York (July 2002)
26. Faugère, J.-C., Joux, A.: Algebraic Cryptanalysis of Hidden Field Equation (HFE) Cryptosystems using Gröbner bases. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 44–60. Springer, Heidelberg (2003)
27. Faugère, J.-C., Perret, L.: Polynomial Equivalence Problems: Algorithmic and Theoretical Aspects. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 30–47. Springer, Heidelberg (2006)
28. Faugère, J.-C., Perret, L.: Cryptanalysis of $2R^-$ Schemes. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 357–372. Springer, Heidelberg (2006)
29. Faugère, J.-C., Levy-dit-Vehel, F., Perret, L.: Cryptanalysis of MinRank. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 280–296. Springer, Heidelberg (2008)
30. Garey, M.R., Johnson, D.B.: Computers and Intractability. A Guide to the Theory of NP-Completeness. W.H. Freeman, New York (1979)
31. Knudsen, L.R.: Truncated and Higher Order Di fferentials. In: Preneel, B. (ed.) FSE 1994. LNCS, vol. 1008, pp. 196–211. Springer, Heidelberg (1995)
32. Lai, X.: Higher Order Derivatives and Differential Cryptanalysis. In: Communications and Cryptography, pp. 227–233. Kluwer Academic Publishers, Dordrecht (1994)
33. Lim, C.-W., Khoo, K.: An Analysis of XSL Applied to BES. In: Biryukov, A. (ed.) FSE 2007. LNCS, vol. 4593, pp. 242–253. Springer, Heidelberg (2007)
34. Matsui, M.: Linear Cryptanalysis Method for DES Cipher. In: Helleseth, T. (ed.) EUROCRYPT 1993. LNCS, vol. 765, pp. 386–397. Springer, Heidelberg (1994)
35. Wiedemann, D.H.: Solving sparse linear equations over finite fields. IEEE Trans. Information Theory IT-32, 54–62 (1986)