Algebraic Precomputations in Differential Cryptanalysis

Martin Albrecht*[1], Carlos Cid[1], Thomas Dullien[2], Jean-Charles Faugère[3], and Ludovic Perret[3]

[1] Information Security Group, Royal Holloway, University of London
Egham, Surrey TW20 0EX, United Kingdom
{M.R.Albrecht,carlos.cid}@rhul.ac.uk
[2] Lehrstuhl für Kryptologie und IT-Sicherheit, Ruhr-Universität Bochum
44780 Bochum, Germany
Thomas.Dullien@ruhr-uni-bochum.de
[3] SALSA Project - INRIA (Centre Paris-Rocquencourt)
UPMC, Univ Paris 06 - CNRS, UMR 7606, LIP6
104, avenue du Président Kennedy 75016 Paris, France
jean-charles.faugere@inria.fr, ludovic.perret@lip6.fr

Abstract. Algebraic cryptanalysis is a general tool which permits one to assess the security of a wide range of cryptographic schemes. Algebraic techniques have been successfully applied against a number of multivariate schemes and stream ciphers. Yet, their feasibility against block ciphers remains the source of much speculation. At FSE 2009 Albrecht and Cid proposed to combine differential cryptanalysis with algebraic attacks against block ciphers. The proposed attacks required Gröbner basis computations during the online phase of the attack. In this work we take a different approach and only perform Gröbner basis computations in a pre-computation (or offline) phase. In other words, we study how we can improve "classical" differential cryptanalysis using algebraic tools. We apply our techniques against the block ciphers PRESENT and KTANTAN32.

1 Introduction

Algebraic cryptanalysis is a general tool which permits one to assess the security of a wide range of cryptographic schemes [2, 19, 18, 17, 15, 16, 21, 23, 22, 20]. As pointed out in the report [11], "*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 – amongst them block ciphers, which are the main concern of this paper.

The basic principle of algebraic 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). The secret can thus be derived by solving the equation system.

Algebraic techniques have been successfully applied against a number of multivariate schemes and in stream cipher cryptanalysis. On the other hand, their feasibility against block ciphers remains the source of much speculation [13, 12, 2, 17]. The sizes of the resulting equation systems are usually beyond the capabilities of current solving algorithms. Furthermore, the complexity estimates are complicated as the algebraic systems are highly structured; a situation where known complexity bounds are no longer valid.

While it is currently infeasible to *cryptanalyse* a block cipher by algebraic means alone, these techniques nonetheless have practical implications for block cipher cryptanalysis. For instance,

* This author was supported by the Royal Holloway Valerie Myerscough Scholarship.

Albrecht and Cid [1] proposed at FSE 2009 to combine differential cryptanalysis with algebraic attacks against block ciphers and demonstrated the feasibility of their techniques against reduced-round versions of the block cipher PRESENT [5]. In this approach, the key recovery was approached by solving (or showing lack of solutions in) equation systems that were much simpler than the full cipher.

In this paper, we elaborate on this approach. That is, we further shift the focus away from attempting to solve the full system of equations. It turns out that significant information can be gained without solving the equation system in the classical sense. Additionally to PRESENT, we also apply these concepts to the block cipher KTANTAN32 [9].

We recall that differential cryptanalysis was formally introduced by Eli Biham and Adi Shamir at Crypto'90 [4], and has since been successfully used to attack a wide range of block ciphers. In its basic form, the attack can be used to distinguish an n-bit block cipher from a random permutation. By considering the distribution of output *differences* for the non-linear components of the cipher (e.g. the S-Box), the attacker may be able to construct *differential characteristics* $P^{'} \oplus P^{''} = \Delta P \rightarrow \Delta C = C^{'} \oplus C^{''}$ for a number of rounds N that are valid with probability p. If $p \gg 2^{-n}$, then by querying the cipher with a large number of plaintext pairs with prescribed difference ΔP, the attacker may be able to distinguish the cipher from a random permutation by counting the number of pairs with the output difference predicted by the characteristic. A pair for which the characteristic holds is called a *right pair*.

By modifying the attack, one can use it to recover key information. Instead of characteristics for the full N-round cipher, the attacker considers characteristics valid for r rounds only ($r = N - R$, with $R > 0$). If such characteristics exist with non-negligible probability the attacker can guess some key bits of the last rounds, partially decrypt the known ciphertexts, and verify if the result matches the one predicted by the characteristic. Candidate (last round) keys are counted, and as random noise is expected for wrong key guesses, eventually a peak may be observed in the candidate key counters, pointing to the correct round key.

The number of right pairs that are needed to distinguish the right candidate key depends on the probability p of the characteristic, the number k of simultaneous subkey bits that are used for the practical decryptions and counted, the average count α of how many keys are suggested per analysed pair (excluding the wrong pairs than can be discarded before the counting), and the fraction β of the analysed pairs among all the pairs. If an attacker is looking for k subkey bit, they can count the number of occurrences of the 2^k possible key values in 2^k counters. The counters contain an average of $(m \cdot \alpha \cdot \beta)/2^k$ counts, where m is the number of pairs, $m \cdot \beta$ is the expected number of pairs to analyse, and α is the number of suggested keys on average. Since these suggestions are spread across 2^k counters, we divide by 2^k. The right subkey value is counted $m \cdot p$ times by the right pairs, plus the random counts for all the possible subkeys. The *signal to noise ratio* is therefore:

$$S/N = \frac{m \cdot p}{m \cdot \alpha \cdot \beta / 2^k} = \frac{2^k \cdot p}{\alpha \cdot \beta}.$$

Note that it would be sufficient to consider the probability p of the differential – i.e. the sum of all p_i for all characteristics with $\Delta P \rightarrow \Delta C$ – instead of the probability of the characteristic. However, in practice authors often work with the probabilities of characteristics because it is easier to estimate them.

Albrecht and Cid proposed in [1] three techniques (so-called *Attack-A*, *Attack-B* and *Attack-C*) which require Gröbner basis computations during the online phase of the attack. This limitation prevented them from applying their techniques to PRESENT-80 with more than 16 rounds, since computation time would exceed exhaustive key search. In this work we take a different approach and only perform Gröbner basis computations in a pre-computation (or offline) phase. That is, we

study how we can improve "classical" differential cryptanalysis using the algebraic tools available to us. More specifically, we aim to increase the signal to noise ratio S/N using algebraic techniques.

The paper is organised as follows. In Section 2 we establish the notation used throughout the paper. In Section 3 we provide a high level description of the main idea behind this work. In Section 4 we briefly describe the ciphers which we use to demonstrate our ideas. These ideas are applied to reduce the noise in Section 5 and to improve the signal in Section 6. Experimental results are also presented in Sections 5 and 6.

2 Notation

We consider N-round block ciphers with a B_s-bit blocksize and K_s-bit key size. When we consider substitution-permutation networks (SPN) we denote the inputs to the S-Box layer as X and the outputs as Y. We always consider the parallel encryption of two plaintexts $P' = (P'_0, \ldots, P'_{B_s-1})$ and $P'' = (P''_0, \ldots, P''_{B_s-1})$ which are related by the input difference $\Delta P = (\Delta P_0, \ldots, \Delta P_{B_s-1})$. Thus we have $P'_i \oplus P''_i = \Delta P_i$ for $0 \leq i < B_s$. We consider $r < N$ round differential characteristics and have that $N - r = R$. If a differential characteristic predicts that the j-th inputs to the i-th S-Box layer application are related by the difference $\Delta X_{i,j}$, then given the plaintext difference ΔP, we have that $X'_{i,j} \oplus X''_{i,j} = \Delta X_{i,j}$ is true with some non-negligible probability. The characteristic also predicts that the j-th outputs of the i-th S-box layer application are related by the difference $\Delta Y_{i,j}$.

Finally, we denote the equation system encoding the encryption of P' to C' under the key K as F' and the ideal spanned by the $f \in F'$ as I' (similarly for P'' and C''). If we write $X_{i,j}$ we refer to both $X'_{i,j}$ and $X''_{i,j}$ (similarly for $Y_{i,j}$). In general we start counting at zero, except for the rounds, which we start counting at one.

3 Main Idea

The main idea involves shifting the emphasis of previous algebraic attacks away from attempting to solve a equation system towards *using ideal membership as implication*. Instead of trying to solve an equation system arising from the cipher, we use Gröbner basis methods to calculate what a particular differential pattern *implies*.

To explain the main idea we start with a small example. Consider the 4-bit S-Box of PRESENT [5]. The S-Box can be completely described by a set of polynomials that express each output bit in terms of the input bits. One can consider a *pair* of input bits $X'_{1,0}, \ldots, X'_{1,3}$ and $X''_{1,0}, \ldots, X''_{1,3}$ and their respective output bits $Y'_{1,0}, \ldots, Y'_{1,3}$ and $Y''_{1,0}, \ldots, Y''_{1,3}$. Since the output bits are described as polynomials in the input bits, it is easy to build a set of polynomials describing the parallel application of the S-Box to the pair of input bits. Assume the fixed input difference of $(0, 0, 0, 1)$ holds for this S-Box. This can be described algebraically by adding the polynomials $X'_{1,3}+X''_{1,3} = 1$, $X'_{1,j} + X''_{1,j} = 0$ for $0 \leq j < 3$ to the set. As usual, the field equations are also added.

The set of equations now forms a description of the parallel application of the S-Box to two inputs with a fixed input difference. The ideal I spanned by these polynomials contains *all* polynomials that are *implied* by the set. If all equations in the generating set of the ideal evaluate to zero, it is clear that any element of I evaluates to zero. This means that *any equation in the ideal will always vanish* if it is assigned values generated by applying the S-Box to a pair of inputs with the above-mentioned input difference.

From a cryptographic point of view, it is important to understand what relations between output bits will hold for a particular input difference. As a consequence, we are looking for polynomials in *just the output bits* which are contained in I. Algebraically, we are trying to find elements of the ideal $I_Y = I \bigcap \mathbb{F}_2[Y'_{1,0}, \ldots, Y'_{1,3}, Y''_{1,0}, \ldots, Y''_{1,3}]$ where I is the ideal spanned by our original equations. A *deglex* Gröbner basis G_Y of this ideal can be computed using standard elimination techniques [3, p.168]. For this, we can set up a block or product ordering where all output variables are lexicographically smaller than any other variable in the system. In addition, we fix the *deglex* ordering among the output variables. Computing the Gröbner basis with respect to such an ordering gives us the Gröbner basis G_Y. We note that G_Y will contain the relations of lowest degree of I_Y due to the choice of term ordering. In our example, we obtain:

$$
\begin{aligned}
G_Y = [&Y'_{1,3} + Y''_{1,3} + 1,\\
&Y'_{1,0} + Y'_{1,2} + Y''_{1,0} + Y''_{1,2} + 1,\\
&Y''_{1,0}Y''_{1,2} + Y'_{1,2} + Y''_{1,0} + Y''_{1,1} + Y''_{1,3},\\
&Y''_{1,0}Y''_{1,1} + Y''_{1,0}Y''_{1,3} + Y''_{1,1}Y''_{1,2} + Y''_{1,2}Y''_{1,3} + Y'_{1,1} + Y''_{1,0} + Y''_{1,1},\\
&Y'_{1,2}Y''_{1,2} + Y''_{1,1}Y''_{1,2} + Y''_{1,2}Y''_{1,3},\\
&Y'_{1,2}Y''_{1,0} + Y''_{1,1}Y''_{1,2} + Y''_{1,2}Y''_{1,3} + Y'_{1,1} + Y'_{1,2} + Y''_{1,0} + Y''_{1,3},\\
&Y'_{1,1}Y''_{1,2} + Y'_{1,2}Y''_{1,1} + Y'_{1,2}Y''_{1,3} + Y''_{1,1}Y''_{1,2} + Y'_{1,1} + Y'_{1,2} + Y''_{1,1},\\
&Y'_{1,1}Y''_{1,1} + Y'_{1,1}Y''_{1,3} + Y''_{1,1}Y''_{1,2} + Y''_{1,1}Y''_{1,3} + Y''_{1,2}Y''_{1,3} + Y''_{1,1},\\
&Y'_{1,1}Y''_{1,0} + Y'_{1,2}Y''_{1,1} + Y'_{1,2}Y''_{1,3} + Y''_{1,0}Y''_{1,3} + Y''_{1,1}Y''_{1,2} + Y''_{1,2}Y''_{1,3} + Y'_{1,1} + Y''_{1,3},\\
&Y'_{1,1}Y'_{1,2} + Y'_{1,2}Y''_{1,3} + Y''_{1,1}Y''_{1,2} + Y''_{1,2}Y''_{1,3} + Y'_{1,2}].
\end{aligned}
$$

There is no other linear or quadratic polynomial $p \in I_Y$ which is not a simple algebraic combination of the polynomials in G_Y.

Of course, we can ask different questions instead of looking for low degree equations. For instance, we can query whether there are equations in the bits $Y'_{1,3}, Y''_{1,2}, Y''_{1,3}$ induced by the input difference by setting up the appropriate term ordering.

In order to formalise this idea, consider a function E (for example a block cipher). Assume that E can be expressed as a set of algebraic equations F over $\mathbb{F}$. If one application of the function can be described as a set of equations, d parallel applications to d different inputs (which we denote $P_0, \ldots, P_{d-1}$) can also be described as a set of equations. We call the set of equations relating the i-th input and output E_i and the matching polynomial system F_i. The outputs of these equations are called $C_0, \ldots, C_{d-1}$. Furthermore, assume any property Λ which holds on $P_0, \ldots, P_{d-1}$ and which can be expressed in a set of algebraic equations F_Λ. A natural question to ask is: How do properties on $P_0, \ldots, P_{d-1}$ affect properties on $C_0, \ldots, C_{d-1}$? We combine the sets of polynomials $\overline{F} = F_\Lambda \cup (\bigcup_{i=0}^{d-1} F_i)$ and consider the Ideal $I = \langle \overline{F} \rangle$ spanned by $\overline{F}$. Next, we compute the unique reduced Gröbner basis G_C of the ideal $I_C = I \cap \mathbb{F}[C_0, \ldots, C_{d-1}]$. Now G_C contains all "relevant" polynomials in $C_0, \ldots, C_{d-1}$, where "relevant" is determined by the term ordering.

As soon as we can compute the Gröbner basis G_C for the function E then we only need to collect the right polynomials from G_C. However, for many functions E computing G_C seems infeasible using current Gröbner basis techniques, implementations and computing power. Thus we have to relax some conditions hoping that we still can recover some equations using a similar technique. We provide below a few heuristics and techniques which still allow recovering *some* relevant equations.

Early Abort. To recover some properties we might not need to compute the complete Gröbner basis, instead we may opt to stop the computation at some degree D.

Replacing Symbols by Constants. It is possible to replace the symbols $P_0, \ldots, P_{d-1}$ by some constants satisfying the constraint Λ which further simplifies the computation. Of course any

polynomial recovered from such a computation would have to be checked against other values to verify that it actually holds in general or with high probability.

Choosing a Different Term Ordering. Instead of computing with respect to an elimination ordering, which is usually more expensive than a degree compatible ordering, we may choose to perform our computations with respect to an easier ordering such as *degrevlex*. Used together with **Early Abort**, we have no assurances about the uniqueness and completeness of the recovered system. However, we might still be able to recover some information.

Computing Normal Forms Only. We can also compute equations by computing normal forms only. For many ciphers it is possible to construct a Gröbner basis for the round transformation [7, 8] with respect to some elimination ordering without any polynomial reductions. These constructions exploit the fact that a system of polynomials is a Gröbner basis if each polynomial has a leading term which is pairwise prime with every other leading term. Using this property, we may construct a Gröbner basis for some elimination ordering for the inverse of the cipher, i.e. the decryption process, such that the input variables are lexicographically bigger than the output variables of some round. Furthermore, we construct the term ordering such that the variables of round $i-1$ are lexicographically bigger than the variables for round i. Furthermore, the symbols C_i are the lexicographically smallest.

If G' is such a Gröbner basis for r rounds for the first encryption and G'' such a Gröbner basis for r rounds for the second encryption, we can combine these bases to $G = G' \cup G''$ which still is a Gröbner basis. Now we can compute the normal form of $X'_i + X''_i + \Delta X_i$ with respect to G. This will eliminate all variables $> C_i$ as much as possible by construction. If this computation does not give equations in the C_i only, we may opt to perform an interreduction on several such equations hoping that this way the remaining key bits are eliminated. For example, such a Gröbner basis for one application of the PRESENT S-Box is

$$
\begin{aligned}
&X_{1,0} + Y_{1,0}Y_{1,1}Y_{1,3} + Y_{1,1}Y_{1,2}Y_{1,3} + Y_{1,2}Y_{1,3} + Y_{1,0} + Y_{1,1} + Y_{1,2} + Y_{1,3},\\
&X_{1,1} + Y_{1,0}Y_{1,1}Y_{1,3} + Y_{1,0}Y_{1,2}Y_{1,3} + Y_{1,1}Y_{1,2}Y_{1,3} + Y_{1,0}Y_{1,2} + Y_{1,0}Y_{1,3}\\
&\qquad + Y_{1,1}Y_{1,2} + Y_{1,1}Y_{1,3} + Y_{1,2}Y_{1,3} + Y_{1,0} + 1,\\
&X_{1,2} + Y_{1,0}Y_{1,1}Y_{1,3} + Y_{1,0}Y_{1,2}Y_{1,3} + Y_{1,1}Y_{1,2}Y_{1,3} + Y_{1,0}Y_{1,1} + Y_{1,0}Y_{1,2}\\
&\qquad + Y_{1,1}Y_{1,3} + Y_{1,0} + Y_{1,2} + Y_{1,3},\\
&X_{1,3} + Y_{1,0}Y_{1,2} + Y_{1,1} + Y_{1,3} + 1
\end{aligned}
$$

The normal form of the equation $X'_{1,3} + X''_{1,3} + 1$ with respect to G (i.e. two of such systems for X', Y' and X'', Y'') is $Y'_{1,0}Y'_{1,2} + Y''_{1,0}Y''_{1,2} + Y'_{1,1} + Y'_{1,3} + Y''_{1,1} + Y''_{1,3} + 1$.

4 Case Studies

To demonstrate our techniques, we consider the block ciphers PRESENT and KTANTAN32.

PRESENT [5] was proposed at CHES 2007 as an ultra-lightweight block cipher, enabling a very compact implementation in hardware, and therefore particularly suitable for RFIDs and similar devices. There are two variants of PRESENT: one with 80-bit keys and one with a 128-bit keys, denoted as PRESENT-80 and PRESENT-128 respectively.

PRESENT is an SP-network with a blocksize of 64 bits and both versions have 31 rounds. Each round of the cipher has three layers of operations: `keyAddLayer`, `sBoxLayer` and `pLayer`. The operation `keyAddLayer` is a simple subkey addition to the current state, while the `sBoxLayer` operation consists of 16 parallel applications of a 4-bit S-Box. The operation `pLayer` is a permutation of wires.

The PRESENT authors give a security analysis of their cipher by showing resistance against well-known attacks such as differential and linear cryptanalysis [5]. The best published differential attacks are for 16 rounds of PRESENT-80 [27] and 17 (and possibly up to 19) rounds [1] for PRESENT-128. Results on linear cryptanalysis for up to 26 rounds are available in [10, 24]. Bit-pattern based integral attacks [28] are successful up to seven rounds of PRESENT. A new type of attack, called statistical saturation attack, was proposed in [14] and shown to be applicable up to 24 rounds of PRESENT.

KTANTAN32 [9] was proposed at CHES 2009 and is the smallest cipher in a family of block ciphers proposed in [9]. It allows a very compact implementation in hardware. It has a blocksize of 32 bits and accepts an 80-bit key. The input is loaded into two registers L_2 and L_1 of 19 and 13 bit length respectively. A round transformation is then applied to these registers 254 times. This round function updates two bits using a quadratic function and performs rotations on the registers. After 254 rounds the content of L_2 and L_1 is outputed as the ciphertext.

The designers of KTANTAN consider a wide range of attacks in their security argument and show the cipher secure against differential, linear, impossible differential, algebraic attacks and some combined attacks. So far, there are no third-party security analyses of the cipher of which the authors are aware.

5 Reducing the Noise

Recall that in order to discard wrong pairs, [1] proposed a technique referred to as *Attack-C*. In this context, the attacker considers an equation system modelling only the rounds $> r$. The attacker is left with R rounds for each plaintext–ciphertext pair to consider. These are related by the output difference predicted by the differential. If we denote the equation system for the last R rounds of the encryption of P' to C' or P'' to C'' as F'_R or F''_R respectively. The algebraic part of *Attack-C* is a Gröbner basis computation on the polynomial system

$$F = F'_R \cup F''_R \cup \{X'_{r+1,i} + X''_{r+1,i} + \Delta X_{r+1,i} \mid 0 \leq i < B_s\}.$$

Whenever the Gröbner basis is equivalent to $\{1\}$, we know that the analysed pair could not have been a right pair. Thus the pair can be discarded. However, no strong assurances are given in [1] as to how many pairs are actually discarded by this technique[4].

In the this work, we consider the same system of equations as in *Attack-C* but replace the tuples of constants C' and C'' by tuples of symbols. By computing a Gröbner basis for the right elimination ordering (cf. Section 3), we can recover equations in the variables C' and C'' which must evaluate to zero on the actual ciphertext values as soon as the input difference for round $r+1$ holds. Once we recovered such equations we can calculate the probability that all these polynomials evaluate to zero for random values for C' and C''. This gives an estimate about the quality of the filter. Furthermore, if the equations in C' and C'' are of sufficiently small degree, this filter is much faster than *Attack-C* since no Gröbner basis has to be computed for each pair.

5.1 Case Study: PRESENT

We consider the differential from [27] and construct filters for PRESENT reduced to $14+R$ rounds. The same filter applies also to $10+R$, $6+R$ and $2+R$ rounds since the characteristic is iterative with a period of four rounds. The explicit polynomials in this section do not differ for PRESENT-80 and PRESENT-128.

[4] Note that *Attack-B* in [1] is guaranteed to distinguish right pairs eventually.

PRESENT 1R. We consider the polynomial ring $P =$

$$\begin{aligned}\mathbb{F}_2[\ & K_{0,0},\ldots,K_{0,79},\ K_{1,0},\ldots,K_{1,3},\\ & Y'_{1,0},\ldots,Y'_{1,63},\ Y''_{1,0},\ldots,Y''_{1,63},\ X'_{1,0},\ldots,X'_{1,63},\ X''_{1,0},\ldots,X''_{1,63},\\ & \ldots,\ K_{15,0},\ldots,K_{15,3},\\ & Y'_{15,0},\ldots,Y'_{15,63},\ Y''_{15,0},\ldots,Y''_{15,63},\ X'_{15,0},\ldots,X''_{15,63},\ X''_{15,0},\ldots,X''_{15,63},\\ & C'_0,\ldots,C'_{63},\ C''_0,\ldots,C''_{63}]\end{aligned}$$

and attach the following block ordering:

$$\underbrace{K_{0,0},\ldots,X''_{15,63}}_{\text{degrevlex}},\underbrace{C'_0,\ldots,C''_{63},C''_0,\ldots,C''_{63}}_{\text{degrevlex}}.$$

We set up an equation system as in *Attack-C* of [1], except that the ciphertext bits are symbols (C'_i and C''_i). Then, we compute the Gröbner basis up to degree $D = 3$ using PolyBoRi 0.6.3 [6] (as shipped with the Sage [26] mathematics software) with the option `deg_bound=3` and filter out any polynomial that contains non-ciphertext variables.

This computation returns 60 polynomials of which 58 are linear. These 58 linear polynomials are of the form $C'_i + C''_i$ for

$$i \in \{0,\ldots,6,8,\ldots 14,16,\ldots 22,24,\ldots 30,32,\ldots 38,40,\ldots 46,48,\ldots,63\}.$$

The remaining two polynomials are $(C'_{23} + C''_{23} + 1)(C'_7 + C'_{39} + C''_7 + C''_{39} + 1)$ and $(C'_{31} + C''_{31} + 1)(C'_{15} + C'_{47} + C''_{15} + C''_{47} + 1)$. The probability that all polynomials evaluate to zero on a random point is approximately $2^{-58.83}$.

PRESENT 2R. We extend the ring and the system from the 1R experiment in the obvious way and perform the same computation as before. This computation returns 65 polynomials of which 46 are linear. Forty linear polynomials are of the form $C'_i + C''_i$ and encode the information that the last round output difference of 10 S-Boxes must be zero (cf. [27]). The remaining 24 polynomials split into two sets F_0, F_2 of 12 polynomials in 24 variables each and the F_j do not share any variables with each other or the first 40 linear polynomials. The systems F_j are listed in Figure 2 in the Appendix. The probability that all polynomials evaluate to zero for a random point is $\approx 2^{-50.669}$.

For comparison, we construct random pairs C', C'' which pass this polynomial filter and notice that for *Attack-C* from [1] roughly every second such pair for Present-80 and 317 out of 512 for Present-128 will pass. Thus we expect *Attack-C* to pass with probability $\approx 2^{-51.669}$ for Present-80 and with probability $\approx 2^{-51.361}$ for Present-128. Finally, we recall that Wang's filter from [27] passes with probability $2^{-40} \cdot (5/16)^6 \approx 2^{-50.07}$.

PRESENT 3R. We extend the ring and the block ordering in the obvious way and compute a Gröbner basis with degree bound 3. The computation returns 28 polynomials of which 16 are linear. The linear polynomials have the form $C'_i + C''_i$ for

$$i \in \{3,7,11,15,19,23,27,31,35,39,43,47,51,55,59,63\}.$$

The remaining 12 polynomials are quadratic and cubic (cf. Figure 3 in the Appendix). The probability that all polynomials evaluate to zero on a random point is $\approx 2^{-18.296}$. To compare with *Attack-C*, we construct random pairs C', C'' which pass this polynomial filter. *Attack-C* will accept roughly 6 in 1024 pairs for Present-80 and 9 out of 1024 pairs for Present-128. Thus, we expect *Attack-C* to pass with probability $\approx 2^{-25.711}$ for Present-80 and $2^{-25.126}$ for Present-128.

PRESENT 4R. We extend the ring and the block ordering in the obvious way. With a degree bound $D = 3$ we recover

$$(C'_{32+j} + C''_{32+j} + 1)(C'_j + C''_j + 1)(C'_{16+j} + C'_{48+j} + C''_{16+j} + C''_{48+j})$$

for $0 \leq j < 16$. The probability that all polynomials evaluate to zero on a random point is $\approx 2^{-3.082}$. We verified experimentally that this bound is optimal by using the SAT solver CryptoMiniSat [25] on *Attack-C* systems in a 4R attack against Present-80-14. The solver returned an assignment which satisfies the equation system with probability $\approx 2^{-3}$. Thus, we conclude that *Attack-C* will roughly accept 1 in 8 pairs.

5.2 Case Study: KTANTAN

In Table 1 we give our results against Ktantan32. We used the best differential for 42 rounds as provided by the KTANTAN designers and extended it to 71 rounds. The characteristic has a probability of 2^{-31}. Below we present results for the degree bounded at four and at five respectively. For each degree bounds we give the number of degree 1-5 polynomials (denoted as $d = *$) we found. In the last column of each experiment we give the approximate probability that all the equations we found evaluate to zero for random values (denoted $\log_2 p$).

	degree bound = 4						degree bound = 5					
N	$d=1$	$d=2$	$d=3$	$d=4$	$d=5$	$\log_2 p$	$d=1$	$d=2$	$d=3$	$d=4$	$d=5$	$\log_2 p$
72	32	0	0	0	0	-32.0	32	0	0	0	0	-32.0
74	32	0	0	0	0	-32.0	32	0	0	0	0	-32.0
76	32	0	0	0	0	-32.0	32	0	0	0	0	-32.0
78	31	3	0	0	0	-32.0	31	3	0	0	0	-32.0
80	28	11	0	0	0	-31.4	28	11	0	0	0	-31.4
82	25	23	0	0	0	-31.0	25	23	0	0	0	-31.0
84	20	32	4	8	0	-29.0	20	32	4	32	0	-29.0
86	16	44	19	8	0	-25.7	16	46	23	75	106	< -24
88	12	39	54	96	0	-24.0	12	51	103	371	745	< -23
90	8	41	129	287	0	-23.0	8	42	133	612	1762	< -22
92	4	28	113	285	0	-20.0	4	33	133	743	2646	-20.4
94	1	20	94	244	0	-16.3	1	25	124	662	2345	-18.5
96	0	8	38	96	0	-12.8	0	8	52	287	1264	-14.3
98	0	3	8	29	0	-7.0	0	3	10	46	156	-9.1
100	0	1	3	13	0	-3.7	0	1	3	18	47	-4.6
102	0	0	0	2	0	-0.8	0	0	0	4	9	-0.9
103	0	0	0	1	0	-0.4	0	0	0	2	4	-0.4
104	0	0	0	0	0	0.0	N/A	N/A	N/A	N/A	N/A	N/A

Table 1. Decreasing the noise for Ktantan32.

6 Increasing the Signal

In this section, we consider the problem of increasing the amount of correct data that has to agree and is always suggested by a right pair. Increasing this value usually has considerable costs attached to it. First, more data needs to be managed and thus usually the counter tables get bigger. On average, we can expect each additional bit considered to double the size of these tables. Second,

in order to generate more data, more partial decryptions must be performed which increases the computation time. Additionally, the number of key bits that can be trial decrypted might be limited by the number of rounds R we can consider because of the quality of the filter.

In this work we use (non-linear) data available from the first few rounds instead of the last R rounds. Assume that we have an SP-network, a differential characteristic $\Delta = (\Delta P, \Delta Y_1, \ldots, \Delta Y_r)$ valid for r rounds with probability p, and (P', P'') a right pair for Δ (so that $\Delta P = P' \oplus P''$ and ΔY_r holds for the output of round r). For simplicity, let us assume that only one S-Box is active in round 1, with input $X'_{1,j}$ and $X''_{1,j}$ (restricted to this S-Box) for the plaintext P' and P'' respectively, and that there is a key addition immediately before the S-Box operation, that is

$$S(P'_j \oplus K_{0,j}) = S(X'_{1,j}) = Y'_{1,j} \text{ and } S(P''_j \oplus K_{0,j}) = S(X''_{1,j}) = Y''_{1,j}.$$

The S-Box operation S can be described by a (vectorial) Boolean function, expressing each bit of the output $Y'_{1,j}$ as a polynomial function (over $\mathbb{F}_2$) on the input bits of $X'_{1,j}$ and $K_{0,j}$. If (P', P'') is a right pair, then the polynomial equations arising from the relation $\Delta Y_{1,j} = Y'_{1,j} \oplus Y''_{1,j} = S(P'_j \oplus K_{0,j}) \oplus S(P''_j \oplus K_{0,j})$ gives us a very simple equation system to solve, with only the key variables $K_{0,j}$ as unknowns (and which do not vanish identically because we are considering nonzero differences). Consequently, right pairs suggest additional information about the key from the first round difference. In particular, if ΔY_1 holds with probability 2^{-b} then we can recover b bits of information about the key, if we found a right pair.

There is no *a priori* reason to restrict this argument from [1] to the first round. Let Δ, r, P', P'' be as before. We setup two equation systems F' and F'' involving P', C' and P'', C'' respectively and discard any polynomials from the rounds $> s$ where s is a small integer > 0. Previously we had $s = 1$. Finally, we add linear equations as suggested by the characteristic and use this system to recover information about the key from the first s rounds.

In order to avoid the potentially costly Gröbner basis computation for every candidate pair replace the tuples of constants P' and P'' by tuples of symbols. According to Section 3, we can compute polynomials involving only key variables and the newly introduced plaintext variables P' and P''. Assume that we can indeed compute the Gröbner basis with P' and P'' symbols for the first s rounds and the linear equations arising from the characteristic added. Assume further that the probability that the characteristic restricted to s rounds holds is 2^{-b} and that we computed m_s polynomials in the variables K_0, P' and P''. This means that we recover b bits of information when we evaluate all m_s polynomials such that we replace P' and P'' by their actual values.

This means that we have b bits of extra information and thus can write $S/N = \frac{2^{k+b} \cdot p}{\alpha \cdot \beta}$ without the overhead of performing any partial decryptions. However, we have to perform m_s polynomial evaluations (where we replace P' and P'' by their actual values) of relatively small low degree polynomials.

Case Study: PRESENT. We consider the first two encryption rounds and the characteristic from [27]. We set up a polynomial ring with two blocks such that the variables P_i and K_i are lexicographically smaller than any other variables. Within the blocks we chose a degree lexicographical term ordering. We set up an equation system covering the first two encryption rounds and added the linear equations suggested by the characteristic. Then, we eliminated all linear leading terms which are not in the variables P_i and K_i and computed a Gröbner basis up to degree five. This computation returned 22 linear and quadratic polynomials (we give the Gröbner basis for these polynomials in Figure 4). This system gives 8 bits of information about the key. Note that the first two rounds of the characteristic pass with probability 2^{-8}.

Case Study: KTANTAN32. We consider the first 24 rounds of KTANTAN32 and compute the full Gröbner basis. This computation recovers 39 polynomials. We list an excerpt in Figure 1 in the Appendix. As expected we observe that the characteristic also imposes restrictions on the plaintext. These eight equations allow us to recover up to four bits (depending on the value of P'_{19}) of information about the key.

Acknowledgements

We would like to thank William Stein for allowing us to run our experiments on his computers[5]. We would also like to thank anonymous referees for helpful comments.

Conclusion

In this work, we have introduced a novel application for the algebraic cryptanalysis of block ciphers. We propose a method which can improve "classical" differential cryptanalysis, by applying algebraic tools in a pre-computation phase. As such, we shift the focus from attempting to solve large systems of polynomial equations to recovering symbolic information about the underlying cipher. Although the use of algebraic techniques in general, and Gröbner basis methods in particular, in block cipher cryptanalysis has received some criticism within the cryptographic community, as it has been often the case that "simpler" techniques can perform favourably in many situations, we stress that the rich algebraic structure of Gröbner basis can offer many advantages and may give one a more subtle insight of the cipher structure. This can in turn be used in the cryptanalysis of the cipher. We note that *in principle* our techniques can recover an optimal amount of information and that in almost all cases considered in this work we were (almost) able to accomplish this. We expect that this approach is applicable to other cryptanalytical techniques such as linear and higher-order differential cryptanalysis and consider applying it as an area of future work.

References

1. Martin Albrecht and Carlos Cid. Algebraic Techniques in Differential Cryptanalysis. In *Fast Software Encryption 2009*, Lecture Notes in Computer Science, Berlin, Heidelberg, New York, 2009. Springer Verlag.
2. Gwenole Ars. *Applications des bases de Gröbner à la cryptographie*. PhD thesis, Université de Rennes I, 2005.
3. Thomas Becker and Volker Weispfenning. *Gröbner Bases - A Computational Approach to Commutative Algebra*. Springer Verlag, Berlin, Heidelberg, New York, 1991.
4. Eli Biham and Adi Shamir. Differential Cryptanalysis of DES-like Cryptosystems. In *Advances in Cryptology — CRYPTO 1990*, volume 537 of *Lecture Notes in Computer Science*, pages 3–72, Berlin, Heidelberg, New York, 1991. Springer Verlag.
5. Andrey Bogdanov, Lars R. Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew J. B. Robshaw, Yannick Seurin, and C. Vikkelsoe. PRESENT: An ultra-lightweight block cipher. In *Cryptographic Hardware and Embedded Systems - CHES 2007*, volume 7427 of *Lecture Notes in Computer Science*, pages 450–466, Berlin, Heidelberg, New York, 2007. Springer Verlag. Available at `http://www.crypto.rub.de/imperia/md/content/texte/publications/conferences/present_ches2007.pdf`.
6. Michael Brickenstein and Alexander Dreyer. PolyBoRi: A framework for Gröbner basis computations with Boolean polynomials. In *Electronic Proceedings of MEGA 2007*, 2007. Available at `http://www.ricam.oeaw.ac.at/mega2007/electronic/26.pdf`.

[5] Purchased under National Science Foundation Grant No. DMS-0821725.

7. Johannes Buchmann, Andrei Pychkine, and Ralf-Philipp Weinmann. A Zero-dimensional Gröbner Basis for AES-128. In *Fast Software Encryption 2006*, volume 4047 of *Lecture Notes in Computer Science*, pages 78–88. Springer Verlag, 2006.
8. Stanislav Bulygin and Michael Brickenstein. Obtaining and solving systems of equations in key variables only for the small variants of AES. *Mathematics in Computer Science*, 2008. conditionally accepted for special issue "Symbolic Computation and Cryptography".
9. Christophe De Cannière, Orr Dunkelman, and Miroslav Knežević. KATAN and KTANTAN — a family of small and efficient hardware-oriented block ciphers. In Christophe Clavier and Kris Gaj, editors, *Cryptographic Hardware and Embedded Systems - CHES 2009*, volume 5747 of *Lecture Notes in Computer Science*, pages 272–288. Springer Verlag, 2009.
10. Joo Yeon Cho. Linear cryptanalysis of reduced-round PRESENT. Cryptology ePrint Archive, Report 2009/397, 2009. available at `http://eprint.iacr.org/2009/397`.
11. Carlos Cid. D.STVL.7 algebraic cryptanalysis of symmetric primitives, 2008. available at `http://www.ecrypt.eu.org/ecrypt1/documents/D.STVL.7.pdf`.
12. Carlos Cid and Gaëtan Leurent. An Analysis of the XSL Algorithm. In *Advances in Cryptology — ASIACRYPT 2005*, volume 3788 of *Lecture Notes in Computer Science*, pages 333–352, Berlin, Heidelberg, New York, 2005. Springer Verlag.
13. Carlos Cid, Sean Murphy, and Matthew Robshaw. Small Scale Variants of the AES. In *Fast Software Encryption 2005*, volume 3557 of *Lecture Notes in Computer Science*, pages 145–162, Berlin, Heidelberg, New York, 2005. Springer Verlag. Available at `http://www.isg.rhul.ac.uk/~sean/smallAES-fse05.pdf`.
14. Baudoin Collard and Francois-Xavier Standaert. A Statistical Saturation Attack against the Block Cipher PRESENT. In *Topics in Cryptology – CT-RSA 2009*, pages 195–210, 2009.
15. Nicolas T. Courtois. Fast algebraic attacks on stream ciphers with linear feedback. In *Advances in Cryptology – CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 176–194, 2003.
16. Nicolas T. Courtois. Higher order correlation attacks, XL algorithm and cryptanalysis of Toyocrypt. In P.J. Lee and C.H. Lim, editors, *Information Security and Cryptology - ICISC 2002: 5th International Conference*, volume 2587 of *Lecture Notes in Computer Science*, Berlin, Heidelberg, New York, 2003. Springer Verlag.
17. Nicolas T. Courtois and Gregory V. Bard. Algebraic Cryptanalysis of the Data Encryption Standard. In Steven D. Galbraith, editor, *Cryptography and Coding – 11th IMA International Conference*, volume 4887 of *Lecture Notes in Computer Science*, pages 152–169, Berlin, Heidelberg, New York, 2007. Springer Verlag. pre-print available at `http://eprint.iacr.org/2006/402`.
18. Nicolas T. Courtois and Willi Meier. Algebraic attacks on stream ciphers with linear feedback. In Eli Biham, editor, *Advances in Cryptology — EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 345–359, Berlin, Heidelberg, New York, 2003. Springer Verlag.
19. Nicolas T. 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, Berlin, Heidelberg, New York, 2002. Springer Verlag.
20. Jean-Charles Faugère, Françoise Levy dit Vehel, and Ludovic Perret. Cryptanalysis of MinRank. In *Advances in Cryptology – CRYPTO 2008*, volume 5157 of *Lecture Notes in Computer Science*, pages 280–296, 2008.
21. Jean-Charles Faugère and Antoine Joux. Algebraic cryptanalysis of hidden field equation (hfe) cryptosystems using gröbner bases. In *CRYPTO*, pages 44–60, 2003.
22. Jean-Charles Faugère and Ludovic Perret. Cryptanalysis of $2r^-$ schemes. In *Advances in Cryptology – CRYPTO 2006*, volume 4117 of *Lecture Notes in Computer Science*, pages 357–372, 2006.
23. Jean-Charles Faugère and Ludovic Perret. Polynomial equivalence problems: Algorithmic and theoretical aspects. In *Advances in Cryptology – EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 30–47, 2006.
24. Jorge Nakahara Jr, Pouyan Seperhdad, Bingsheng Zhang, and Meiqin Wang. Linear (hull) and algebraic cryptanalysis of the block cipher PRESENT. In *The 8th International Conference on Cryptoplogy and Network Security - CANS 2009*, 2009.
25. Mate Soos, Karsten Nohl, and Claude Castelluccia. Extending SAT solvers to cryptographic problems. In Oliver Kullmann, editor, *Theory and Applications of Satisfiability Testing - SAT 2009*, volume 5584 of *Lecture Notes in Computer Science*, pages 244–257, Berlin, Heidelberg, New York, 2009. Springer Verlag.

26. William Stein et al. *SAGE Mathematics Software*. The Sage Development Team, 2008. Available at `http://www.sagemath.org`.
27. Meiqin Wang. Differential Cryptanalysis of reduced-round PRESENT. In Serge Vaudenay, editor, *Africacrypt 2008*, volume 5023 of *Lecture Notes in Computer Science*, pages 40–49, Berlin, Heidelberg, New York, 2008. Springer Verlag.
28. Muhammad Reza Z'Aba, Håvard Raddum, Matt Henricksen, and Ed Dawson. Bit-pattern based integral attacks. In Kaisa Nyberg, editor, *Fast Software Encryption 2008*, number 5086 in Lecture Notes In Computer Science, pages 363–381, Berlin, Heidelberg, New York, 2008. Springer Verlag.

A Explicit Polynomials

$$
\begin{aligned}
&(P'_{19}+1)(P'_3P'_8+P'_{10}P'_{12}+K_3+K_{53}+P'_7+P'_{18}+P'_{23}),\\
&P'_8P'_{10}P'_{19}+K_8P'_{19}+P'_3P'_8+P'_6P'_{19}+P'_{10}P'_{12}\\
+&P'_{16}P'_{19}+K_3+K_{53}+P'_7+P'_{18}+P'_{19}+P'_{23},\\
&P'_{19}P'_{22}+K_1+K_{11}+P'_6+P'_{11}+P'_{17}+P'_{21}+P'_{26},\\
&P'_{23}P'_{26}+K_{65}+P'_{21}+P'_{25}+P'_{30},\\
&P'_1+1,P'_2,P'_5+1,P'_9+1
\end{aligned}
$$

Fig. 1. Polynomials for the first two rounds of KTANTAN32.

$$
\begin{aligned}
&(C'_{57+j}+C''_{57+j})(C'_{53+j}+C''_{53+j}+1)(C'_{17+j}+C''_{17+j}),\\
&(C'_{57+j}+C''_{57+j})(C'_{53+j}+C''_{53+j}+1)(C'_{33+j}+C''_{33+j}),\\
&(C'_{57+j}+C''_{57+j}+1)(C'_{25+j}+C''_{25+j}),\\
&(C'_{57+j}+C''_{57+j}+1)(C'_{41+j}+C''_{41+j}),\\
&(C'_{53+j}+C''_{53+j}+1)(C'_{21+j}+C''_{21+j}),\\
&(C'_{53+j}+C''_{53+j}+1)(C'_{37+j}+C''_{37+j}),\\
&(C'_{53+j}+C''_{53+j}+1)(C'_{49+j}+C'_{57+j}+C''_{49+j}+C''_{57+j}+1),\\
&(C'_{49+j}+C''_{49+j}+1)(C'_{17+j}+C''_{17+j}),\\
&(C'_{49+j}+C''_{49+j}+1)(C'_{33+j}+C''_{33+j}),\\
&C'_{1+j}+C'_{33+j}+C'_{49+j}+C''_{1+j}+C''_{33+j}+C''_{49+j},\\
&C'_{5+j}+C'_{37+j}+C'_{53+j}+C''_{5+j}+C''_{37+j}+C''_{53+j},\\
&C'_{9+j}+C'_{41+j}+C'_{57+j}+C''_{9+j}+C''_{41+j}+C''_{57+j},
\end{aligned}
$$

Fig. 2. 2R polynomials for PRESENT with $j \in \{0, 2\}$.

$$(C'_{36}+C''_{36})((C'_4+C''_4)(C'_{20}+C'_{52}+C''_{20}+C''_{52}+1)+(C'_{20}+C''_{20}+1)(C'_{52}+C''_{52}+1)),$$
$$(C'_{37}+C''_{37})((C'_5+C''_5)(C'_{21}+C'_{53}+C''_{21}+C''_{53}+1)+(C'_{21}+C''_{21}+1)(C'_{53}+C''_{53}+1)),$$
$$(C'_{40}+C''_{40})((C'_8+C''_8)(C'_{24}+C'_{56}+C''_{24}+C''_{56}+1)+(C'_{24}+C''_{24}+1)(C'_{56}+C''_{56}+1)),$$
$$(C'_{41}+C''_{41})((C'_9+C''_9)(C'_{25}+C'_{57}+C''_{25}+C''_{57}+1)+(C'_{25}+C''_{25}+1)(C'_{57}+C''_{57}+1)),$$
$$(C'_{45}+C''_{45})((C'_{13}+C''_{13})(C'_{29}+C'_{61}+C''_{29}+C''_{61}+1)+(C'_{29}+C''_{29}+1)(C'_{61}+C''_{61}+1)),$$
$$(C'_{46}+C''_{46})((C'_{14}+C''_{14})(C'_{30}+C'_{62}+C''_{30}+C''_{62}+1)+(C'_{30}+C''_{30}+1)(C'_{62}+C''_{62}+1)),$$
$$(C'_{06}+C''_{06})((C'_{22}+C''_{22})(C'_{38}+C'_{54}+C''_{38}+C''_{54}+1)+(C'_{38}+C''_{38}+1)(C'_{54}+C''_{54}+1)),$$
$$(C'_{10}+C''_{10})((C'_{26}+C''_{26})(C'_{42}+C'_{58}+C''_{42}+C''_{58}+1)+(C'_{42}+C''_{42}+1)(C'_{58}+C''_{58}+1)),$$
$$(C'_{12}+C''_{12})((C'_{28}+C''_{28})(C'_{44}+C'_{60}+C''_{44}+C''_{60}+1)+(C'_{44}+C''_{44}+1)(C'_{60}+C''_{60}+1)),$$
$$(C'_{52}+C''_{52}+1)(C'_{20}+C''_{20}+1)(C'_4+C'_{36}+C''_4+C''_{36}),$$
$$(C'_{60}+C''_{60}+1)(C'_{28}+C''_{28}+1)(C'_{12}+C'_{44}+C''_{12}+C''_{44}),$$
$$(C'_{10}+C'_{42}+C'_{58}+C''_{10}+C''_{42}+C''_{58})(C'_2+C'_{34}+C'_{50}+C''_2+C''_{34}+C''_{50}).$$

Fig. 3. 3R polynomials for PRESENT.

$$(K_1+P'_1+1)(K_0+K_3+K_{29}+P'_0+P'_3),$$
$$(K_2+P'_2)(K_0+K_3+K_{29}+P'_0+P'_3),$$
$$K_1K_2+K_1P'_2+K_2P'_1+P'_1P'_2+K_0+K_1+K_3+K_{29}+P'_0+P'_1+P'_3,$$
$$(K_9+P'_9+1)(K_8+K_{11}+K_{31}+P'_8+P'_{11}),$$
$$(K_{10}+P'_{10})(K_8+K_{11}+K_{31}+P'_8+P'_{11}),$$
$$K_9K_{10}+K_9P'_{10}+K_{10}P'_9+P'_9P'_{10}+K_8+K_9+K_{11}+K_{31}+P'_8+P'_9+P'_{11},$$
$$(K_{49}+P'_{49}+1)(K_{41}+K_{48}+K_{51}+P'_{48}+P'_{51}),$$
$$(K_{50}+P'_{50})(K_{41}+K_{48}+K_{51}+P'_{48}+P'_{51}),$$
$$K_{49}K_{50}+K_{49}P'_{50}+K_{50}P'_{49}+P'_{49}P'_{50}+K_{41}+K_{48}+K_{49}+K_{51}+P'_{48}+P'_{49}+P'_{51},$$
$$(K_{57}+P'_{57}+1)(K_{43}+K_{56}+K_{59}+P'_{56}+P'_{59}),$$
$$(K_{58}+P'_{58})(K_{43}+K_{56}+K_{59}+P'_{56}+P'_{59}),$$
$$K_{57}K_{58}+K_{57}P'_{58}+K_{58}P'_{57}+P'_{57}P'_{58}+K_{43}+K_{56}+K_{57}+K_{59}+P'_{56}+P'_{57}+P'_{59},$$
$$K_5+K_7+P'_5+P'_7,$$
$$K_6+K_7+P'_6+P'_7,$$
$$K_{53}+K_{55}+P'_{53}+P'_{55},$$
$$K_{54}+K_{55}+P'_{54}+P'_{55}$$

Fig. 4. Polynomials for the first two rounds of PRESENT.