Algebraic Techniques in Differential Cryptanalysis

Martin Albrecht* and Carlos Cid

Information Security Group,
Royal Holloway, University of London
Egham, Surrey TW20 0EX, United Kingdom
{M.R.Albrecht,carlos.cid}@rhul.ac.uk

Abstract. In this paper we propose a new cryptanalytic method against block ciphers, which combines both algebraic and statistical techniques. More specifically, we show how to use algebraic relations arising from differential characteristics to speed up and improve key-recovery differential attacks against block ciphers in some situations. To illustrate the new technique, we apply it to reduced round versions of the cipher PRESENT, an ultra lightweight block cipher proposed at CHES 2007, particularly suitable for deployment in RFID tags.

1 Introduction

The two most established cryptanalytic methods against block ciphers are linear cryptanalysis [20] and differential cryptanalysis [2]. These attacks are statistical in nature, in which the attacker attempts to construct probabilistic patterns through as many rounds of the cipher as possible, in order to distinguish the cipher from a random permutation, and ultimately recover the key. Due to their very nature, these attacks require a very large number of plaintext–ciphertext pairs, ensuring that (usually) they rapidly become impractical. In fact, most modern ciphers have been designed with these attacks in mind, and therefore do not generally have their security affected by them.

A new development in block cipher cryptanalysis are the so-called algebraic attacks [12, 21, 7]. In contrast to linear and differential cryptanalysis, algebraic attacks attempt to exploit the algebraic structure of the cipher. In its most common form, the attacker expresses the encryption transformation as a large set of multivariate polynomial equations, and subsequently attempts to solve the system to recover information about the encryption key.

The proposal of algebraic attacks against block ciphers has been the source of much speculation; while a well-established technique against some stream ciphers constructions [11], the viability of algebraic attacks against block ciphers remains the subject to debate. On one hand these attack techniques promise to allow the cryptanalyst to recover secret key bits given only one or very few plaintext–ciphertext pairs. This is due to the fact that algebraic attacks do not

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

rely on probabilistic properties of the cipher. On the other hand, the runtime of algebraic attacks against block ciphers is not well understood, and it is currently not clear whether algebraic attacks can cryptanalyse any proposed block cipher faster than other more established techniques.

A promising approach however is to combine both statistical and algebraic techniques in block cipher cryptanalysis. In fact, many proposed algebraic approaches already involve statistical components. For instance, the equation systems usually considered for the AES [21, 7], use the *inversion equation* $xy = 1$ for the S-Box. While this equation only holds with probability $p = 255/256$, it may well offer some advantages when compared with the correct equation $x^{254} = y$ representing the S-Box (which due to its very high degree, is usually considered impractical). Further recent examples include key bit guesses [9], the use of SAT-solvers [1] and the Raddum-Semaev algorithm [22] for solving polynomial equations. In this paper we propose a new attack technique that combines results from algebraic and differential cryptanalysis.

The paper is structured as follows. First, we briefly describe differential and algebraic cryptanalysis and give the basic idea of the attack in Section 2. We then describe the block cipher PRESENT in Section 3 and existing attacks against a reduced round version of PRESENT (Section 3.2). In Section 4 we describe the application of our new attack technique against reduced round versions of PRESENT. We give a brief discussion of the attack and possible extensions in Section 5.

2 Overview of the New Attack Technique

The two building blocks of our proposed attack are differential and algebraic cryptanalysis.

2.1 Differential Cryptanalysis

Differential cryptanalysis was first introduced by Eli Biham and Adi Shamir at Crypto'90 [3], 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 a n-bit block cipher from a random permutation as follows: 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* $P^{'} \oplus P^{''} = \Delta P \rightarrow \Delta C_r = C_r^{'} \oplus C_r^{''}$ for a number of rounds r that are valid with probability p. If we have $p \gg 2^{-n}$, 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 by counting the number of pairs with the output difference predicted by the characteristic.

By modifying the attack, one can use it to recover key information. Instead of characteristics for the full cipher, the attacker considers characteristics valid for one (or two) rounds shorter. Again, if such characteristics exist with non-negligible probability, the attacker can guess some key bits of the last round,

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[1].

Note that due to its statistical nature, differential cryptanalysis requires a very large number of plaintext–ciphertext pairs (for instance, approximately 2^{47} chosen plaintext pairs are required to break DES [4]). Many extensions and variants of differential cryptanalysis exist, such as Boomerang attack [23] and truncated and higher-order differentials [19]. The technique is however very well understood, and most modern ciphers are designed to resist to differential cryptanalysis. This is often achieved by carefully selecting the cipher's non-linear operations and diffusion layer (the AES is a prime example of this approach [13]).

2.2 Algebraic Cryptanalysis

Algebraic cryptanalysis against block ciphers is an attack technique that has recently received a lot of attention, particularly after it was proposed in [12] against the AES and Serpent block ciphers. In its basic form, the attacker attempts to express the cipher as a set of low degree (often quadratic) equations, and then solve the resulting system. As these systems are usually very sparse, overdefined, and structured, it is conjectured that they may be solved much faster than generic non-linear equation systems. Several algorithms have been used and/or proposed to solve these systems including the Buchberger algorithm, XL and variants [10, 26, 12] , J-C Faugère's F_4 and F_5 algorithms [14, 15], and the Raddum-Semaev algorithm [22]. However, these methods have had so far limited success in targeting modern block ciphers, and to the authors' best knowledge, no modern block cipher, with practical relevance, has been successfully attacked using algebraic cryptanalysis faster than with other techniques.

2.3 Algebraic Techniques in Differential Cryptanalysis

The first idea in extending algebraic cryptanalysis is to use more plaintext–ciphertext pairs to construct the equation system. Indeed, given two equation systems F' and F'' for two plaintext–ciphertext pairs (P', C') and (P'', C'') under same encryption key K, we can simply combine these equation systems to form a system $F = F' \cup F''$. Note that while F' and F'' share the key and key schedule variables, they do not share most of the state variables. Thus by considering two plaintext–ciphertext pairs the cryptanalyst gathers twice as many equations, involving however many new variables. Experimental evidence indicates that this technique may often help solving a system of equations at least up to a certain number of rounds [16].

The second step is to consider probabilistic relations that may arise from differential cryptanalysis. This gives rise to the *Attack-A* described below.

[1] In some variants, as described in [4], no candidate key counters are required; see Section 5 for a brief discussion of this attack.

Attack-A. For the sake of simplicity, we assume the cipher is a SP-Network, which iterates layers of non-linear transformations (e.g. S-Box operations) and affine transformations. Now consider a differential characteristic for a number of rounds $\Delta = (\delta_0, \delta_1, \ldots, \delta_r)$, where $U'_{i-1} \oplus U''_{i-1} = \delta_{i-1} \rightarrow \delta_i = U'_i \oplus U''_i$ is a one-round difference arising from round i and valid with probability p_i. We can thus assume that the characteristic Δ is valid with probability $p = \prod p_i$. Assuming that r is close to the full number of rounds N_r, then a "folk theorem" [23] in differential cryptanalysis states that a cipher is broken if p is large enough such that $p \gg 2^{-n}$, where $n = min(\text{blocksize}, \text{keysize})$.

Note that each one-round difference gives rise to equations relating the input and output pairs for active S-Boxes. Indeed, let $X'_{i,j}$ and $X''_{i,j}$ denote the j-th bit of the input to an S-Box in round i for the systems F' and F'', respectively. Similarly, let $Y'_{i,j}$ and $Y''_{i,j}$ denote the corresponding output bits. Then we have that the expressions

$$X'_{i,j} + X''_{i,j} = \Delta X_{i,j} \rightarrow \Delta Y_{i,j} = Y'_{i,j} + Y''_{i,j},$$

where $\Delta X_{i,j}, \Delta Y_{i,j}$ are known values predicted by the characteristic, are valid with some non-negligible probability q. Similarly, for non-active S-Boxes (that is, S-Boxes that are not involved in the characteristic Δ and therefore have input/output difference zero), we have the relations

$$X'_{i,j} + X''_{i,j} = 0 = Y'_{i,j} + Y''_{i,j}$$

also valid with a non-negligible probability.

Now if we consider the equation system $F = F' \cup F''$, we can combine F and all such linear relations arising from the characteristic Δ. This gives rise to an equation system $\overline{F}$ which holds with probability p. If such a system is solved for approximately $1/p$ pairs of plaintext–ciphertext, we expect at least one non-empty solution, which should yield the encryption key. The benefit of this approach is the we expect the system $\overline{F}$ to be easier to solve than the original system F' (or F''), because many linear constrains were added without adding any new variables. However, we do not know *a priori* how difficult it will be to solve the system approximately $1/p$ times; we would need to perform experiments to learn the exact performance of the attack for specific ciphers. Yet we are inclined to believe that this approach may offer advantages when compared to conventional algebraic attacks, especially in situations where high probability characteristics exist (which may however not be valid for enough rounds to allow conventional differential attacks to be successful).

Attack-B. When performing differential cryptanalysis the attacker is required to construct differential characteristics Δ that are valid for a large number of rounds r and with non-negligible probability p. By encrypting a large number of plaintext pairs, the attacker uses *right pairs* (i.e. pairs of plaintext that satisfy the characteristic Δ) and their predicted output value to verify key bit guesses. Block cipher designers protect against differential cryptanalysis by ensuring that

if such differential characteristics exist, then $r \ll N_r$, where N_r is the number of rounds of the full cipher. This ensures that backward key guessing is impractical.

Now, assume that we have a SP-network, a differential characteristic $\Delta = (\delta_0, \delta_1, \ldots, \delta_r)$ valid for r rounds with probability p, and (P', P'') a right pair for Δ (so that $\delta_0 = P' \oplus P''$). For simplicity, let us assume that only the i-th S-Box is active in round 1, with input X'_i and X''_i for the plaintext P' and P'' respectively, and that there is a key addition immediately before the S-Box operation, that is

$$S(P'_i \oplus K_{0,i}) = S(X'_i) = Y'_i \text{ and } S(P''_i \oplus K_{0,i}) = S(X''_i) = Y''_i.$$

The S-Box operation S can usually be described by a (vectorial) Boolean function, expressing each bit of the output Y'_i as a polynomial function (over $\mathbb{F}_2$) on the input bits of X'_i and $K_{0,i}$. If (P', P'') is a right pair, then the polynomial equations arising from the relation $\Delta Y_i = Y'_i \oplus Y''_i = S(P'_i \oplus K_{0,i}) \oplus S(P''_i \oplus K_{0,i})$ give us a very simple equation system to solve, with only the key variables $K_{0,i}$ as unknowns (and which does not vanish identically because we are considering nonzero differences, c.f. Section 5). Consequently, if we had an effective distinguisher to determine whether (P', P'') is a right pair, we could learn some bits of information about the round keys involved in the first round active S-Boxes.

Experimentally, we found that, for some ciphers and up to a number of rounds, *Attack-A* can be used as such a distinguisher. More specifically, we noticed that finding a contradiction (i.e. the Gröbner basis equal to $\{1\}$) was much faster than computing the full solution of the system if the system was consistent (that is, when we have a right pair). Thus, rather than fully solving the systems to eventually recover the secret key as suggested in *Attack-A*, the *Attack-B* proceeds by measuring the time t it maximally takes to find that the system is inconsistent (say, using the Buchberger algorithm) and assume we have a right pair if this time t elapsed without a contradiction. One needs to be able to experimentally estimate the time t, but for some ciphers this appears to be an efficient form of attack.

Attack-C. Experimental evidence with PRESENT (c.f. Section 4) indicates that *Attack-B* in fact only relies on the differential $\delta_0 \to \delta_r$ rather than the characteristic Δ when finding contradictions in the systems. In fact, the runtimes for finding contradictions for a 17-round version with a 14-round differential characteristic did not differ significantly from the runtimes for the same task with 5-round version and 2-round characteristic. This indicates that the computational difficulty is mostly determined by the difference $N_r - r$, the number of "free" rounds. We thus define a new attack (*Attack-C*) for which we remove the equations for all rounds $\leq r$.

This significantly reduces the number of equations and variables, and after these equations are removed we are left with $N_r - r$ rounds for each plaintext–ciphertext pair to consider; these are related by the output difference predicted by the differential. As a result, the algebraic computation is essentially equivalent

to solving a related cipher of $2(N_r - r)$ rounds (from C' to C'' via the predicted difference δ_r) using an algebraic meet-in-the-middle attack [7].

Again, we attempt to solve the system and wait for a fixed time t to find a contradiction in the system. If no contradiction is found, we assume that the differential $\delta_0 \rightarrow \delta_r$ holds. In contrast to *Attack-B*, we cannot be certain about the output difference of the first round active S-Boxes (as the attack now distinguishes pairs that satisfy *differentials* rather than *characteristics*). However, the attack can be adapted such that we can still recover first round key bits. Another option is to attempt to solve the resulting smaller system, to fully recover the encryption key.

To study the viability of these attacks, in the following sections we describe experiments with reduced-round versions of the block cipher PRESENT.

3 The Block Cipher PRESENT

PRESENT [5] was proposed by Bogdanov et al. 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. In our experiments, we consider reduced round variants of both ciphers denoted as PRESENT-K_s-N_r, where $K_s \in \{80, 128\}$ represents the key size in bits and $1 \leq N_r \leq 31$ represents the number of rounds.

PRESENT is a 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 given in Table 1. The operation `pLayer` is a permutation of wires given by the rule that the bit at position $s \cdot j + i$ ($0 \leq j < B$, $0 \leq i < s$) is moved to position $B \cdot i + j$, where $s = 4$ is the S-Box width and $B = 16$ is the number of parallel S-Boxes.

Table 1. The S-Box Table

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

In both versions, these three operations are repeated $N_r = 31$ times. On the final round, an extra subkey addition is performed. The subkeys are derived from the user-provided key in the key schedule, which by design decision is also quite simple and efficient. For the 80-bit variant the user-supplied key is stored in a key register K and represented as $k_{79}k_{78} \ldots k_0$. At round i the 64-bit round key $K_i = k_{i,63}k_{i,62} \ldots k_{i,0}$ consists of the 64 upmost bits of the current contents of

register K:

$$K_i = k_{i,63}k_{i,62}\dots k_{i,0} = k_{79}k_{78}\dots k_{16}.$$

After round key K_i is extracted, the key register $k = k_{79}k_{78}\dots k_0$ is updated in the following way:

1. $[k_{79}k_{78}\dots k_1k_0] = [k_{18}k_{17}\dots k_{20}k_{19}]$
2. $[k_{79}k_{78}k_{77}k_{76}] = S[k_{79}k_{78}k_{77}k_{76}]$
3. $[k_{19}k_{18}k_{17}k_{16}k_{15}] = [k_{19}k_{18}k_{17}k_{16}k_{15}] \oplus \text{round\_counter}$

The key schedule for 128-bit keys is quite similar and presented in Appendix II of [5]. We note that the difference between the 80-bit and 128-bit variants is only the key schedule. In particular, both variants have the same number of rounds (i.e. $N_r = 31$). The cipher designers explicitly describe in [5] the threat model considered when designing the cipher, and acknowledge that the security margin may be somewhat tight. Although they do not recommend immediate deployment of the cipher (especially the 128-bit version), they strongly encourage the analysis of both versions.

3.1 An Equation System for PRESENT

We give 9 quadratic equations for the PRESENT S-Box:

$$\begin{aligned}
0 &= x_1y_0 + x_0y_1 + x_1y_1 + x_0y_3 + x_1y_3 + x_0 + x_3 + y_0 + y_2 + 1,\\
0 &= x_0y_0 + x_2y_0 + x_0y_1 + x_0y_3 + x_0 + x_1 + x_2 + x_3 + y_0 + y_1,\\
0 &= x_0y_0 + x_1y_1 + x_2y_1 + x_0y_2 + x_0 + x_1 + x_3 + y_0 + y_1,\\
0 &= x_0y_1 + x_0y_2 + x_1y_2 + x_0y_3 + x_1 + x_2 + x_3 + y_2 + y_3,\\
0 &= x_0y_0 + x_0y_1 + x_1y_1 + x_0y_2 + x_2y_2 + x_0y_3 + x_2y_3 + x_0 + x_1 + x_2 + y_1 + 1,\\
0 &= x_0y_0 + x_3y_0 + x_0y_2 + x_0y_3 + x_0 + x_1 + y_1 + 1,\\
0 &= x_1y_0 + x_0y_1 + x_1y_1 + x_3y_1 + x_0y_2 + x_2y_2 + x_1 + x_3 + y_0 + y_1 + y_3,\\
0 &= x_0y_0 + x_0y_2 + x_3y_2 + x_0y_3 + x_0 + x_2 + y_0 + y_1 + y_3,\\
0 &= x_1y_0 + x_2y_2 + x_0y_3 + x_3y_3 + x_0 + x_3 + y_0 + y_3 + 1.
\end{aligned}$$

The *degrevlex* Gröbner basis for this equation system gives rise to 21 quadratic equations and one cubic equation and this basis is used for all computations in this paper. These equations are given in Appendix A.

Each round of PRESENT introduces $2 \cdot 64$ new state variables for the input and output bits of the S-Box and thus we have $128 \cdot N_r$ state variables for PRESENT. The 80-bit key schedule has 80 user-provided key variables and 4 new key variables per round to account for one S-Box in the key schedule per round. Thus we have

$$N_v = (128 + 4) \cdot N_r + 80 = 132 \cdot N_r + 80$$

variables. Each round gives rise to $22 \cdot 16$ S-Box equations, 64 key addition equations and 22 key schedule equations. We also have one additional key addition (64 linear equations) at the end and N_v field equations (i.e. equations of the form $x_i^2 + x_i = 0$). Thus we have

$$N_e = (22 \cdot 16 + 22 + 64)N_r + 64 + N_v = 570 \cdot N_r + 144$$

equations. If we use two plaintext–ciphertext pairs, we double the number of equations and variables for the state variables but not the number of equations and variables for the key variables[2]. Thus for PRESENT-80-31 we would have a system of 8140 variables in 34742 equations if we consider two plaintext–ciphertext pairs.

3.2 Differential Cryptanalysis of 16 Rounds of PRESENT-80

In the original proposal [5], the designers of PRESENT show that both linear and differential cryptanalysis are infeasible against the cipher. In [24, 25] M. Wang provides 24 explicit differential characteristics for 14 rounds of PRESENT. These are true with probability 2^{-62} and within the the theoretical bounds provided by the PRESENT designers. Wang's attack is reported to require 2^{65} memory accesses to cryptanalyse 16 rounds of PRESENT-80. We use his characteristics (see Appendix B for an example of one of these characteristics) to mount our attack. Furthermore, we also make use of the filter function presented in [24], which we briefly describe below.

Consider for example the differential characteristic provided in Appendix B. It ends with the difference $\delta = \texttt{1001} = \texttt{9}$ as input for the two active S-Boxes of round 15. According to the difference distribution table of the PRESENT S-Box, the possible output differences are `2`, `4`, `6`, `8`, `12` and `14`. This means that the least significant bit is always zero and thus the weight of the output difference (with the two active S-Box) is at most 6. It then follows from the cipher diffusion layer that at most six S-Boxes are active in round 16; these S-Boxes are x_4, x_6, x_8, x_{10}, x_{12} and x_{14}. Thus we can discard any pair for which the outputs of round 16 have non-zero difference in the positions arising from the output of S-Boxes other than the ones listed above. There are thus 10 such S-Boxes, and we expect to be able to discard 2^{40} pairs using this filter.

Furthermore, it also follows from the cipher diffusion layer that the active S-Boxes in round 16 (which are at most six, as described above) will have input difference `1` and thus all possible output differences are `3`, `7`, `9`, `13` (and `0`, in case the S-Box is inactive). Using this filter we can thus discard $\frac{16}{5}^6 = 2^{10.07}$ pairs of those considered before. Overall we may discard approximately 2^{50} pairs as incorrect. We expect to be able to construct a similar filter function for all the 24 differential characteristics presented in [25].

[2] If the weight on the difference ΔP is small however, some variables for the first few rounds may be the same for both systems.

4 Experimental Results

To mount the attacks against reduced-round versions of PRESENT, we generate systems of equations for pairs of encryptions with prescribed difference as described in Sections 3.1 and 3.2. Then all computations are performed in the polynomial ring

$$\begin{aligned}\mathbb{F}_2\,[\; & K_{N_r,0}, K_{N_r,1}, K_{N_r,2}, K_{N_r,3}, Y'_{N_r,0}, \ldots, Y'_{N_r,63}, X'_{N_r,0}, \ldots, X'_{N_r,63},\\ & Y''_{N_r,0}, \ldots, Y''_{N_r,63}, X''_{N_r,0}, \ldots, X''_{N_r,63}, \ldots, K_{1,0}, K_{1,1}, K_{1,2}, K_{1,3},\\ & Y'_{1,0}, \ldots, Y'_{1,63}, X'_{1,0}, \ldots, X'_{1,63}, Y''_{1,0}, \ldots, Y''_{1,63}, X''_{1,0}, \ldots, X''_{1,63},\\ & K_0, \ldots, K_{79}],\end{aligned}$$

where $X_{i,j}$ and $Y_{i,j}$ represent the j-th input and output bits, respectively, of the i-th S-Box application. For the attack we set up the augmented equation system $\overline{F}$ as described in Section 2, by adding linear equations for the differentials predicted by the 14-round characteristic given in the Appendix. For PRESENT this is equivalent to adding 128 linear equations per round of the form $\Delta X_{i,j} = X'_{i,j} + X''_{i,j}$ and $\Delta Y_{i,j} = Y'_{i,j} + Y''_{i,j}$ where $\Delta X_{i,j}$ and $\Delta Y_{i,j}$ are the values predicted by the characteristic (these are zero for non-active S-Boxes).

To perform the algebraic part of the attack, we use either the SINGULAR [17] routine `groebner` with the monomial odering *degrevlex* or the POLYBORI [6] routine `groebner_basis` with the option `faugere=True` and the monomial ordering `dp_asc` to compute a Gröbner basis for $\overline{F}$. We note the time t this routine takes to detect a contradiction for a given differential length of r, and assume we have a pair satisfying the characteristic (or differential, in *Attack-C*) if this time t elapsed without a contradiction.

We performed experiments for *Attack-B* and *Attack-C*. Runtimes for *Attack-C* are given in Table 4 (runtimes for *Attack-B* are given in Appendix C). The times were obtained using William Stein's `sage.math.washington.edu` computer (1.8Ghz Opteron, 64GB RAM, purchased under National Science Foundation Grant No. 0555776). The attack was implemented in the mathematics software Sage [18].

If a characteristic Δ is valid with probability p, then after approximately $1/p$ attempts we expect to find a right pair, and can thus set up our smaller systems for each first round active S-Box. These equation systems are explicitly:

$$\begin{aligned}
X'_0 &= K_0 + P'_0, \quad X'_1 = K_1 + P'_1, \quad X'_2 = K_2 + P'_2, \quad X'_3 = K_3 + P'_3,\\
Y'_0 &= X'_0X'_1X'_3 + X'_0X'_2X'_3 + X'_0 + X'_1X'_2X'_3 + X'_1X'_2 + X'_2 + X'_3 + 1,\\
Y'_1 &= X'_0X'_1X'_3 + X'_0X'_2X'_3 + X'_0X'_2 + X'_0X'_3 + X'_0 + X'_1 + X'_2X'_3 + 1,\\
Y'_2 &= X'_0X'_1X'_3 + X'_0X'_1 + X'_0X'_2X'_3 + X'_0X'_2 + X'_0 + X'_1X'_2X'_3 + X'_2,\\
Y'_3 &= X'_0 + X'_1X'_2 + X'_1 + X'_3,\\
X''_0 &= K_0 + P''_0, \quad X''_1 = K_1 + P''_1, \quad X''_2 = K_2 + P''_2, \quad X''_3 = K_3 + P''_3,\\
Y''_0 &= X''_0X''_1X''_3 + X''_0X''_2X''_3 + X''_0 + X''_1X''_2X''_3 + X''_1X''_2 + X''_2 + X''_3 + 1,\\
Y''_1 &= X''_0X''_1X''_3 + X''_0X''_2X''_3 + X''_0X''_2 + X''_0X''_3 + X''_0 + X''_1 + X''_2X''_3 + 1,
\end{aligned}$$

$$
\begin{aligned}
Y_2'' &= X_0'' X_1'' X_3'' + X_0'' X_1'' + X_0'' X_2'' X_3'' + X_0'' X_2'' + X_0'' X_1'' X_2'' X_3'' + X_2'', \\
Y_3'' &= X_0'' + X_1'' X_2'' + X_1'' + X_3'', \\
\Delta Y_0 &= Y_0' + Y_0'', \quad \Delta Y_1 = Y_1' + Y_1'', \quad \Delta Y_2 = Y_2' + Y_2'', \quad \Delta Y_3 = Y_3' + Y_3'',
\end{aligned}
$$

where ΔY_i are the *known* difference values predicted by the characteristic.

After substitution of $P_i', P_i'', \Delta Y_i$ and elimination of the variables X_i', X_i'' in the system above, we get an equation system with four equations in the four key variables. If we compute the reduced Gröbner basis for this system we recover two relations of the form $K_i + K_j(+1) = 0$ for two keybits K_i,K_j per S-Box, i.e. we recover 2 bits of information per first round active S-Box[3].

Table 2. Times in seconds for *Attack-C*

N_r	K_s	r	p	#trials	SINGULAR	#trials	POLYBORI
4	80	4	2^{-16}	10	$0.07 - 0.09$	50	$0.05 - 0.06$
4	80	3	2^{-12}	10	$6.69 - 6.79$	50	$0.88 - 1.00$
4	80	2	2^{-8}	10	$28.68 - 29.04$	50	$2.16 - 5.07$
4	80	1	2^{-4}	10	$70.95 - 76.08$	50	$8.10 - 18.30$
16	80	14	2^{-62}	10	$123.82 - 132.47$	50	$2.38 - 5.99$
16	128	14	2^{-62}	0	N/A	50	$2.38 - 5.15$
16	80	13	2^{-58}	10	$301.70 - 319.90$	50	$8.69 - 19.36$
16	128	13	2^{-58}	0	N/A	50	$9.58 - 18.64$
16	80	12	2^{-52}	0	N/A	5	> 4 hours
17	80	14	2^{-62}	10	$318.53 - 341.84$	50	$9.03 - 16.93$
17	128	14	2^{-62}	0	N/A	50	$8.36 - 17.53$
17	80	13	2^{-58}	0	N/A	5	> 4 hours

4.1 Complexity of the Attacks

To compare with the results of [24], we applied *Attack-C* against reduced round versions of PRESENT-80. Using this approach we expect to learn 4 bits of information about the key for PRESENT-80-16 in $(1 \pm \epsilon) \cdot 2^{62-50.07} \cdot 6$ seconds to perform the consistency checks using $(1 \pm \epsilon) \cdot 2^{62}$ chosen plaintext–ciphertext pairs, where 6 seconds represents the highest runtime to find a contradiction we have encountered in our experiments. Even if there are instances that take slightly longer to check, we assume that this is a safe margin because there are many shorter runtimes. This time gives a complexity of $(1 \pm \epsilon) \cdot 2^{62}$ ciphertext difference checks and $(1 \pm \epsilon) \cdot 2^{11.93} \cdot 6 \cdot 1.8 \cdot 10^9 \approx 2^{46}$ cpu cycles to find a right pair on the given 1.8 Ghz Opteron CPU. We assume that a single encryption

[3] This is as expected, since the probability of the differential used in the first round S-Box is 2^{-2}; see Lemma 1.

costs at least two cpu cycles per round – one for the S-Box lookup and one for the key addition – such that a brute force search would require approximately $16 \cdot 2 \cdot 2^{80} \approx 2^{85}$ cpu cycles and two plaintext–ciphertext pairs due to the small blocksize.

In [25], 24 different 14-round differentials were presented, involving the S-Boxes x_0,x_1,x_2,x_{12},x_{13},x_{14} in the first round, each having either `7` or `15` as plaintext difference. From these we expect to recover 18 bits[4] of key information by repeating the attack for those S-Box configurations. We can then guess the remaining $80 - 18 = 62$ bits, and the complete attack has a complexity of $6 \cdot (1 \pm \epsilon) \cdot 2^{62}$ filter function applications, $6 \cdot (1 \pm \epsilon) \cdot 2^{46}$ cpu cycles for the consistency checks and 2^{62} PRESENT applications to guess the remaining key bits[5]. The attack in [24] on the other hand requires 2^{65} memory accesses. While this is a different metric – memory access – from the one we have to use in this case – cpu cycles – we can see that our approach has roughly the same time complexity, since the 2^{62} filter function applications cost at least 2^{62} memory accesses. However, our attack seems to have a slightly better data complexity because overall six right pairs are sufficient. When applying the attack against PRESENT-128-16, we obtain a similar complexity. We note however that for PRESENT-K_s-16, we can also make use of backward key guessing to recover more key bits. Because we have distinguished a right pair already we expect the signal to noise ratio to be quite high and thus expect relatively few wrong suggestions for candidate keys.

Note that we cannot use the filter function for 17 rounds, thus the attack against PRESENT-80-17 gives a worse performance when compared to exhaustive key search. However, it may still be applied against PRESENT-128-17. Indeed, we expect to learn 4 bits of information for PRESENT-128-17 in $(1 \pm \epsilon) \cdot 2^{62} \cdot 18$ seconds using $(1 \pm \epsilon) \cdot 2^{62}$ chosen plaintext–ciphertext pairs. This time is equivalent to $(1 \pm \epsilon) \cdot 2^{62} \cdot 18 \cdot 1.8 \cdot 10^9 \approx 2^{97}$ cpu cycles. If this approach is repeated 6 times for the different active S-Boxes in the PRESENT differentials, we expect to learn 18 bits of information about the key. We can then guess the remaining $128 - 18 = 110$ bits and thus have a complexity in the order of 2^{110} for the attack.

We can also attack PRESENT-128-18 using *Attack-C* as follows. First note that the limiting factor for the attack on PRESENT-128-18 is that we run out of plaintext–ciphertext pairs due to the small blocksize. On the other hand, we still have not reached the time complexity of 2^{128} for 128-bit keysizes. One way to make use of this fact is to again consider the input difference for round 15 and iterate over all possible output differences. As discussed in Section 3.2, the example differential in Appendix B ends with `9` as input difference for both active S-Boxes of round 15, and the possible output differences are `2`, `4`, `6`, `8`, `12` and `14` (c.f. [24]). Thus we have six possible output differences and two active S-Boxes in round 15, which result in 36 possible output differences in total. We expect to learn 4 bits of information about the key for PRESENT-128-18 in

[4] We are not able to recover 24 bits because we learn some redundant information.

[5] Note that the attack can be improved by managing the plaintext–ciphertext pairs more intelligently and by using the fact that we can abort a PRESENT trial encryption if it does not match the known differential.

$(1 \pm \epsilon) \cdot 36 \cdot 2^{62} \cdot 18$ seconds using $(1 \pm \epsilon) \cdot 2^{62}$ chosen plaintext–ciphertext pairs. This time is equivalent to $(1 \pm \epsilon) \cdot 36 \cdot 2^{62} \cdot 18 \cdot 1.8 \cdot 10^9 \approx 2^{102}$ cpu cycles. Again, we can iterate this process six times to learn 18 bits of information about the key and guess the remaining information with an complexity of approximately 2^{110}.

Note that we were unable to reliably find contradictions if $N_r - r \geq 4$ within 4 hours (compared to 18 seconds for three additional rounds).

5 Discussion of the Attack

While the attack has many similarities with conventional differential cryptanalysis, such as the requirement of a high probability differential Δ valid for r rounds and the use of filter functions to reduce the workload, there are however some noteworthy differences. First, *Attack-C* should require slightly fewer plaintext–ciphertext pairs for a given differential characteristic to learn information about the key than conventional differential cryptanalysis, because the attacker does not need to wait for a peak in the partial key counter. Instead one right pair is sufficient. Second, the attack recovers more key bits if many S-Boxes are active in the first round. This follows from our reliance on those S-Boxes to recover key information. Also note that while a high probability differential characteristic is required, the attack recovers more bits per S-Box if the differences for the active S-Box in the first round are of low probability. This is a consequence of the simple Lemma below:

Lemma 1. *Given a differential Δ with a first round active S-Box with a difference that is true with probability 2^{-b}, then* Attack-B *and* Attack-C *can recover b bits of information about the key from this S-Box.*

Finally, key-recovery differential cryptanalysis is usually considered infeasible if the differential Δ is valid for r rounds, and r is much less than the full number of rounds N_r, since backward key guessing for $N_r - r$ rounds may become impractical. In that case the *Attack-C* proposed here could *possibly* still allow the successful cryptanalysis of the cipher. However, this depends on the algebraic structure of the cipher, as it may be the case that the time required for the consistency check is such that the overall complexity remains below the one required for exhaustive key search.

We note that *Attack-C* shares many properties with the differential cryptanalysis of the full 16-round DES [4]. Both attacks are capable of detecting a right pair without maintaining a candidate key counter array. Also, both attacks use active S-Boxes of the outer rounds to recover bits of information about the key once such a right pair is found. In fact, one could argue that *Attack-C* is a generalised algebraic representation of the technique presented in [4]. From this technique *Attack-C* inherits some interesting properties: first, the attack can be carried out fully in parallel because no data structures such as a candidate key array need to be shared between the nodes. Also, we allow the encryption keys to change during the data collection phase because exactly one right pair is sufficient to learn some key information. However, once we try to learn further key

bits by repeating the attack with different characteristics we require the encryption key not to change. We note however that while the attack in [4] seems to be very specific to the target cipher (e.g. DES), *Attack-C* can in principle be applied to any block cipher. Another way of looking at *Attack-C* is to realise that it is in fact is a quite expensive but exact filter function: We invest more work in the management of the outer rounds using algebraic techniques.

In the particular case of PRESENT-80-N_r, our attack seems to offer only marginal advantage when compared with the differential attack presented in [24]: it should require slightly less data to distinguish a right pair and similar overall complexity. On the other hand, for PRESENT-128-N_r this attack seems to perform better than the one in [24]. As in this case the limiting factor is the data and not the time complexity of the attack, i.e. we run out of plaintext–ciphertext pairs before running out of computation time, the attack has more flexibility. In that situation our attack is able to break two more rounds than the best published attack against reduced round PRESENT-128.

The use of Gröbner bases techniques to find contradictions in propositional systems is a well known idea [8]. In the context of cryptanalysis, it is also a natural idea to try to detect contradictions to attack a cipher. However, in probabilistic approaches used in algebraic attacks, usually key bits are guessed. This is an intuitive idea because polynomial systems tend to be easier to solve the more overdefined they are and because the whole system essentially depends on the key. Thus guessing key bits is a natural choice. However this simplification seems to bring few benefits to the attacker, and more sophisticated probabilistic approaches have so far been ignored to the authors' best knowledge. The method proposed in this paper can thus highlight the advantages of combining conventional (statistical) cryptanalysis and algebraic cryptanalysis.

Future research might also investigate the use of other well established statistical cryptanalysis techniques in combination with algebraic cryptanalysis such as linear cryptanalysis (defining a version of *Attack-A* in this case is straightforward), higher order and truncated differentials, the boomerang attack or impossible differentials.

We note that this attack may also offer a high degree of flexibility for improvements. For example, the development of more efficient algorithms for solving systems of equations (or good algebraic representation of ciphers that may result in more efficient solving) would obviously improve the attacks proposed. For instance, by switching from SINGULAR to POLYBORI (and reversing the variable ordering) for *Attack-B*, we were able to make the consistency check up to 60 times faster. Another possible set of tools to speed up the consistency check are SAT-solvers, which have received some cryptanalytic interest recently [1]. As an illustration, if for instance an attacker could make use of an optimised method to find contradictions in $t \ll 2^{128-62} = 2^{66}$ cpu cycles for PRESENT-128-20, this would allow the successful cryptanalysis of a version of PRESENT with 6 more rounds than the best known differential, which is considered "a situation without precedent" by the cipher designers [5]. Unfortunately with the available computer resources, we are not able to verify whether this is currently feasible.

Finally, as our results depend on experimental data and the set of data we evaluated is rather small due to the time and memory consuming nature of our experiments, we make our claims verifiable by providing the source code of the attack via the first author's website[6]. Also, we ran simulations with small numbers of rounds to verify that the attack indeed behaves as expected. We are of course aware that it is in general difficult to reason from small scale examples to bigger instances.

6 Conclusion

We propose a new cryptanalytic technique combining differential cryptanalysis and algebraic techniques. We show that in some circumstances this technique can be effectively used to attack block ciphers, and in general may offer some advantages when compared to differential cryptanalysis. As an illustration, we applied it against reduced versions of PRESENT-80 and PRESENT-128, and showed how it compared against differential cryptanalysis of the same ciphers. While this paper has no implications for the security of either PRESENT-80 or PRESENT-128, it was shown that the proposed techniques can improve upon existing differential cryptanalytic methods using the same difference characteristics.

Acknowledgements

The work described in this paper has been supported in part by the European Commission through the IST Programme under Contract IST-2002-507932 ECRYPT. We would like to thank William Stein for allowing the use of his computer, and Sean Murphy and Matt Robshaw for helpful comments.

References

1. Gregory V. Bard. *Algorithms for Solving Linear and Polynomial Systems of Equations over Finite Fields with Applications to Cryptanalysis*. PhD thesis, University of Maryland, 2007. available at `http://www.cs.umd.edu/~jkatz/THESES/bard_thesis.pdf`.
2. Eli Biham and Adi Shamir. Differential Cryptanalysis of DES-like Cryptosystems. *Journal of Cryptology*, 4(1):3–72, 1991.
3. 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. Springer Verlag, 1991.
4. Eli Biham and Adi Shamir. Differential Cryptanalysis of the Full 16-round DES. In *Advances in Cryptology — CRYPTO 1992*, volume 740 of *Lecture Notes in Computer Science*, pages 487–496, Berlin Heidelberg New York, 1991. Springer Verlag.

[6] `http://www.informatik.uni-bremen.de/~malb/`

5. A. Bogdanov, L.R. Knudsen, G. Leander, C. Paar, A. Poschmann, Matthew Robshaw, Y. Seurin, and C. Vikkelsoe. PRESENT: An ultra-lightweight block cipher. In *CHES 2007*, volume 7427 of *Lecture Notes in Computer Science*, pages 450–466. Springer Verlag, 2007. 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`.
7. Carlos Cid, Sean Murphy, and Matthew Robshaw. *Algebraic Aspects of the Advanced Encryption Standard*. Springer Verlag, 2006.
8. Matthew Clegg, Jeffery Edmonds, and Russell Impagliazzo. Using the Groebner basis algorithm to find proofs of unsatisfiability. In *Proceedings of the 28th ACM Symposium on Theory of Computing*, pages 174–183, 1996. available at `http://www.cse.yorku.ca/~jeff/research/proof_systems/grobner.ps`.
9. 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. available at `http://eprint.iacr.org/2006/402`.
10. Nicolas T. Courtois, Alexander Klimov, Jacques Patarin, and Adi Shamir. Efficient Algorithms for Solving Overdefined Systems of Multivariate Polynomial Equations. In *Advances in Cryptology — EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 392–407. Springer Verlag, 2000.
11. Nicolas T. Courtois and Willi Meier. Algebraic attacks on stream ciphers with linear feedback. In *Advances in Cryptology — EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 345–359. Springer Verlag, 2003.
12. Nicolas T. Courtois and Josef Pieprzyk. Cryptanalysis of Block Ciphers with Overdefined Systems of Equations. Cryptology ePrint Archive, Report 2002/044, 2002. available at `http://eprint.iacr.org/2002/044`.
13. Joan Daemen and Vincent Rijmen. *The design of Rijndael: AES - the Advanced Encryption Standard*. Springer Verlag, 2002.
14. Jean-Charles Faugère. A New Efficient algorithm for Computing Gröbner Basis (F4), 1999. available at `http://modular.ucsd.edu/129-05/refs/faugere_f4.pdf`.
15. Jean-Charles Faugère. A New Efficient Algorithm for Computing Gröbner Bases without Reduction to Zero (F5). In *Proceedings of ISSAC*, pages 75–83. ACM Press, 2002.
16. Jean-Charles Faugère. Groebner bases. Applications in cryptology. FSE 2007 – Invited Talk, 2007. available at `http://fse2007.uni.lu/v-misc.html`.
17. Gert-Martin Greuel, Gerhard Pfister, and Hans Schönemann. Singular 3.0. A Computer Algebra System for Polynomial Computations, Centre for Computer Algebra, University of Kaiserslautern, 2005. available at: `http://www.singular.uni-kl.de`.
18. The SAGE Group. *SAGE Mathematics Software (Version 2.9)*. The SAGE Group, 2007. available at `http://www.sagemath.org`.
19. L.R. Knudsen. Truncated and higher order differentials. In *Fast Software Encryption 1995*, volume 1008 of *Lecture Notes in Computer Science*, pages 196–211. Springer Verlag, 1995.

20. M. Matsui. Linear Cryptanalysis Method for DES Cipher. In *Advances in Cryptology — EUROCRYPT 1993*, volume 765 of *Lecture Notes in Computer Science*, pages 386–397. Springer Verlag, 1993. available at `http://homes.esat.kuleuven.be/~abiryuko/Cryptan/matsui_des.PDF`.
21. Sean Murphy and Matthew Robshaw. Essential Algebraic Structure Within the AES. In M. Yung, editor, *Advances in Cryptology — CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 1–16. Springer Verlag, 2002. available at `http://www.isg.rhul.ac.uk/~mrobshaw/rijndael/aes-crypto.pdf`.
22. Håvard Raddum and Igor Semaev. New technique for solving sparse equation systems. Cryptology ePrint Archive, Report 2006/475, 2006. available at `http://eprint.iacr.org/2006/475`.
23. David Wagner. The boomerang attack. In *Fast Software Encryption 1999*, volume 1636 of *Lecture Notes in Computer Science*, pages 156–170. Springer Verlag, 1999. available at `http://www.cs.berkeley.edu/~daw/papers/boomerang-fse99.ps`.
24. Meiqin Wang. Differential cryptanalysis of PRESENT. Cryptology ePrint Archive, Report 2007/408, Version: 20080108:005109, 2007. available at `http://eprint.iacr.org/2007/408`.
25. Meiqin Wang. Private communication: 24 differential characteristics for 14-round present we have found, 2008.
26. Bo-Yin Yang, Jiun-Ming Chen, and Nicolas T. Courtois. On Asymptotic Security Estimates in XL and Gröbner Bases-Related Algebraic Cryptanalysis. In *Proceedings of Information and Communications Security; 6th International Conference 2004*, volume 3269 of *Lecture Notes in Computer Science*, pages 401–413. Springer Verlag, 2004.

A S-Box Equations for PRESENT

$$
\begin{aligned}
0 &= y_2x_3 + y_3x_3 + x_1x_3 + x_2x_3 + x_3,\\
0 &= y_0x_3 + y_3x_3 + x_1x_3 + x_2x_3 + y_0 + y_3 + x_1 + x_2 + x_3 + 1,\\
0 &= x_1x_2 + y_3 + x_0 + x_1 + x_3,\\
0 &= x_0x_2 + y_3x_3 + x_1x_3 + x_2x_3 + y_0 + y_1 + y_3 + x_0 + x_2 + x_3,\\
0 &= y_3x_2 + y_3x_3 + x_1x_3 + y_0 + y_1 + y_3 + x_0 + x_2 + x_3,\\
0 &= y_0x_2 + y_1x_2 + y_1x_3 + y_3x_3 + y_1 + x_0 + x_1 + x_2 + 1,\\
0 &= x_0x_1 + y_3x_3 + x_1x_3 + x_2x_3 + y_1 + y_2 + x_1 + x_2 + x_3 + 1,\\
0 &= y_3x_1 + y_3x_3 + x_2x_3 + y_1 + y_2 + y_3 + x_0 + x_1 + x_2 + 1,\\
0 &= y_2x_1 + y_2x_2 + y_3x_3 + x_0x_3 + x_1x_3 + y_0 + x_0 + 1,\\
0 &= y_1x_1 + y_2x_2 + y_1x_3 + x_0x_3 + x_1x_3 + y_0 + y_1 + y_2 + y_3 + x_2 + x_3,\\
0 &= y_0x_1 + y_1x_2 + y_1x_3 + x_0x_3 + x_2x_3 + y_1 + y_2 + y_3 + x_0 + x_1 + 1,\\
0 &= y_3x_0 + y_1x_2 + y_2x_2 + y_1x_3 + y_3x_3 + x_0x_3 + x_2x_3 + y_0 + y_1 + y_2 + x_1 + x_3,\\
0 &= y_2x_0 + y_1x_2 + x_0x_3 + y_0 + y_1 + y_2 + x_1 + x_2 + x_3,\\
0 &= y_1x_0 + y_1x_3 + x_0x_3 + x_1x_3 + x_2x_3 + y_0 + y_2 + y_3 + x_0 + x_1 + x_3 + 1,\\
0 &= y_0x_0 + y_2x_2 + y_1x_3 + x_1x_3 + y_0 + y_1 + y_3 + x_0 + x_3,\\
0 &= y_2y_3 + y_1x_3 + y_3x_3 + x_0x_3 + x_2x_3 + y_0 + y_1 + y_2 + y_3 + x_0,\\
0 &= y_1y_3 + y_1x_2 + y_2x_2 + y_1x_3 + y_3x_3 + x_0x_3 + x_1x_3 + y_1 + y_3 + 1,\\
0 &= y_0y_3 + y_1x_3 + y_3x_3 + x_0x_3 + x_1x_3 + y_0 + y_2 + x_1 + x_3 + 1,\\
0 &= y_1y_2 + y_1x_3 + x_0x_3 + x_1x_3 + y_0 + x_0 + 1,\\
0 &= y_0y_2 + y_1 + y_3 + x_3 + 1,\\
0 &= y_0y_1 + y_1x_3 + x_0x_3 + x_2x_3 + y_0 + y_1 + y_2 + x_2 + x_3 + 1,\\
0 &= y_1x_2x_3 + y_1x_2 + y_3x_3 + x_0x_3 + x_1x_3 + x_2x_3 + y_3 + x_0 + x_1 + x_2.
\end{aligned}
$$

B 14-round Differential Characteristic for PRESENT

Rounds		Differences	Pr	Rounds		Difference	Pr
I		$x_2 = 7$,$x_{14} = 7$	1				
R1	S	$x_2 = 1$,$x_{14} = 1$	2^{-4}	R8	S	$x_8 = 9$,$x_{10} = 9$	2^{-4}
R1	P	$x_0 = 4$,$x_3 = 4$	1	R8	P	$x_2 = 5$,$x_{14} = 5$	1
R2	S	$x_0 = 5$,$x_3 = 5$	2^{-4}	R9	S	$x_2 = 1$,$x_{14} = 1$	2^{-6}
R2	P	$x_0 = 9$,$x_8 = 9$	1	R9	P	$x_0 = 4$,$x_3 = 4$	1
R3	S	$x_0 = 4$,$x_8 = 4$	2^{-4}	R10	S	$x_0 = 5$,$x_3 = 5$	2^{-4}
R3	P	$x_8 = 1$,$x_{10} = 1$	1	R10	P	$x_0 = 9$,$x_8 = 9$	1
R4	S	$x_8 = 9$,$x_{10} = 9$	2^{-4}	R11	S	$x_0 = 4$, $x_8 = 4$	2^{-4}
R4	P	$x_2 = 5$,$x_{14} = 5$	1	R11	P	$x_8 = 1$, $x_{10} = 4$	1
R5	S	$x_2 = 1$,$x_{14} = 1$	2^{-6}	R12	S	$x_8 = 9$, $x_{10} = 9$	2^{-4}
R5	P	$x_0 = 4$,$x_3 = 4$	1	R12	P	$x_2 = 5$, $x_{14} = 5$	1
R6	S	$x_0 = 5$,$x_3 = 5$	2^{-4}	R13	S	$x_2 = 1$, $x_{14} = 1$	2^{-6}
R6	P	$x_0 = 9$,$x_8 = 9$	1	R13	P	$x_0 = 4$, $x_3 = 4$	1
R7	S	$x_0 = 4$,$x_8 = 4$	2^{-4}	R14	S	$x_0 = 5$, $x_3 = 5$	2^{-4}
R7	P	$x_8 = 1$,$x_{10} = 1$	1	R14	P	$x_0 = 9$, $x_8 = 9$	1

C Times in seconds for *Attack-B*

N_r	K_s	r	p	#trials	SINGULAR	#trials	POLYBORI
4	80	4	2^{-16}	20	$11.92 - 12.16$	50	$0.72 - 0.81$
4	80	3	2^{-12}	10	$106.55 - 118.15$	50	$6.18 - 7.10$
4	80	2	2^{-8}	10	$119.24 - 128.49$	50	$5.94 - 13.30$
4	80	1	2^{-4}	10	$137.84 - 144.37$	50	$11.83 - 33.47$
8	80	5	2^{-22}	0	N/A	50	$18.45 - 63.21$
10	80	10	2^{-44}	0	N/A	20	$3.24 - 3.92$
10	80	9	2^{-40}	0	N/A	20	$21.43 - 26.41$
10	80	8	2^{-34}	0	N/A	20	$21.73 - 38.96$
10	80	7	2^{-30}	0	N/A	10	$39.27 - 241.17$
10	80	6	2^{-26}	0	N/A	20	$56.30 - > 4$ hours
16	80	14	2^{-62}	0	N/A	20	$43.42 - 64.11$
16	128	14	2^{-62}	0	N/A	20	$45.59 - 65.03$
16	80	13	2^{-58}	0	N/A	20	$80.35 - 262.73$
16	128	13	2^{-58}	0	N/A	20	$81.06 - 320.53$
16	80	12	2^{-52}	0	N/A	5	> 4 hours
16	128	12	2^{-52}	0	N/A	5	> 4 hours
17	80	14	2^{-62}	10	$12,317.49 - 13,201.99$	20	$55.51 - 221.77$
17	128	14	2^{-62}	10	$12,031.97 - 13,631.52$	20	$94.19 - 172.46$
17	80	13	2^{-58}	0	N/A	5	> 4 hours
17	128	13	2^{-58}	0	N/A	5	> 4 hours