EURASIP Journal on Advances in Signal Processing
a SpringerOpen Journal

RESEARCH Open Access

New differential fault analysis on PRESENT

Nasour Bagheri, Reza Ebrahimpour* and Navid Ghaedi

Abstract

In this paper, we present two differential fault analyses on PRESENT-80 which is a lightweight block cipher. The first attack is a basic attack which induces a fault on only one bit of intermediate states, and we can obtain the last subkey of the block cipher, given 48 faulty cipher texts on average. The second attack can retrieve the master key of the block cipher, given 18 faulty cipher texts on average. In the latter attack, we assume that we can induce faults on a single nibble of intermediate states. Given those faulty cipher texts, the computational complexity of attacks is negligible.

Keywords: Lightweight; Block cipher; PRESENT; Differential fault analysis

1 Introduction

Boneh et al. introduced the *fault attack* in September 1996 [1]. Later, in October 1996, Biham and Shamir published an attack on secret key cryptosystems called differential fault analysis (DFA) which combined the ideas of fault attack and differential attack [2]. Given a plaintext, the DFA attack derived information about the secret key by examining the differences between a related cipher text resulting from a correct encryption and a cipher text of the same plaintext resulting from a faulty encryption. Normally, a faulty cipher text is taken by giving external impact on a device with voltage variation, glitch, laser, etc. A fault may be induced with these external impacts; however, we know neither the location nor the value of the fault. This attack is commonly used to analyze the security of cryptosystems. DFA have been employed to attack several block ciphers where DES [3,4], AES [5-11], PRINTCIPHER [12], Camellia [13], CALEFIA [14,15], RC4 [16], SMS4 [17], and ARIA [18] are examples. In general, there are two techniques to apply a DFA attack on a block cipher. The first one assumes that the intermediate states were corrupted by the fault injection and tries to recover the key, while the second form assumes that the key schedule algorithm was corrupted by the fault injection and tries to recover the key.

PRESENT is an ultra-lightweight block cipher proposed at CHES 2007 by Bogdanov et al. This block cipher is an SP network-based cipher and consists of 31 rounds. The block length is 64 bits, and two key lengths of 80 and 128 bits are supported [19], denoted by PRESENT-80 and PRESENT-128, respectively. Several basic attacks such as differential cryptanalysis, linear cryptanalysis, and their variants have been applied on PRESENT already [20-23]. In this paper, we concentrate on the security of PRESENT-80 against DFA attack.

1.1 Previous works

A DFA on PRESENT-80 key schedule has been proposed by Wang et al. [24]. They assumed that the key schedule was corrupted by the fault injection and tried to recover the key. They have reduced the master key search space of PRESENT-80 to 2^{29} with 64 pairs of correct and faulty cipher texts on average. In addition, Zhao et al. [25] proposed a fault propagation pattern-based DFA on PRESENT-80/128. They reduce the master key search space of PRESENT-80/128 to $2^{14.7}$ and $2^{21.1}$, respectively, with 8 and 16 pairs of correct and faulty cipher texts on average.

1.2 Paper contributions

In this paper, we present two differential fault analyses on PRESENT-80. For the first attack, we assume that the fault occurs on only one bit of the intermediate states. In the second attack, we induce a fault on a single nibble of the intermediate states and we employ our first attack to retrieve the master key. Our attack can recover the master key, given 18 faulty cipher texts on average.

*Correspondence: ebrahimpour@ipm.ir
Electrical Engineering Department, Shahid Rajaee Teacher Training University, Tehran, Iran

2 Paper organization

The notations used in the paper are presented in Section 3. PRESENT is briefly described in Section 3.2. In Section 4, we describe our basic attack. We present another DFA attack based on the basic attack in Section 5. Finally, we conclude the paper in Section 6.

3 Notations and preliminaries

3.1 Notations

Throughout the paper, we use the following notations:

- P : denotes the plaintext.
- K : denotes the master key.
- S^i: denotes the correct state before ith round's S-box layer and S^i_j denotes the jth bit of S^i.
- S^{i*}: denotes the faulty state before ith round's S-box layer.
- S^iN^j: denotes the jth nibble of state before ith round's S-box layer and $S^iN^j_n$ denotes its nth bit of the nibble.
- B^i: denotes the state after ith round's S-box layer and B^iN^j denotes the jth nibble of B^i.
- B^{i*}: denotes the faulty state after ith round's S-box layer and $B^{i*}N^j$ denotes the jth nibble of B^{i*}.
- K^i: denotes ith subround key and K^i_j denotes the jth bit of K^i.
- C : denotes the correct cipher text.
- D : denotes the faulty cipher text.

3.2 Description of PRESENT

The PRESENT block cipher, depicted in Figure 1, is an ultra-lightweight substitution-permutation network and consists of 31 rounds. The block length is 64 bits. The cipher supports 80 and 128-bit secret keys. Each round of PRESENT consists of three stages: key addition, non-linear substitution layer, and bit-wise permutation layer.

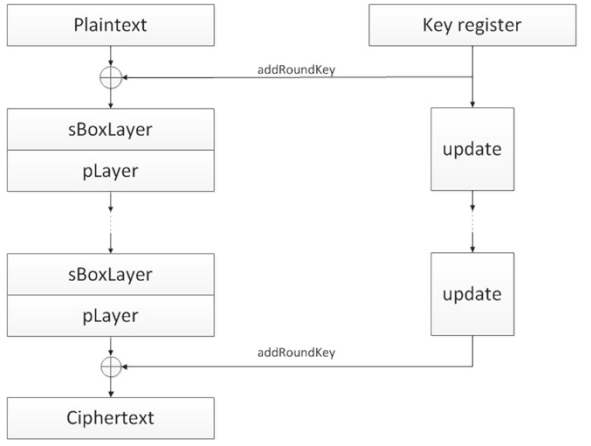

Figure 1 PRESENT encryption algorithm.

3.2.1 Key addition

As the first step of ith round, for $1 \leq i \leq 32$, the current state is combined using bit-wise *XOR* with the ith round subkey.

$$S^i_j \rightarrow S^i_j \oplus K^i_j, \quad \text{for} \quad 0 \leq j \leq 63 \quad \text{and} \quad 1 \leq i \leq 32$$

where K^{32} is used for postwhitening.

3.2.2 Substitution layer (S-box)

The output of the key addition stage goes through the S-box layer (Table 1) which is the non-linear operation of the round. The S-box used in PRESENT is a single 4-bit to 4-bit S-box which is applied 16 times in parallel. The action of the S-box is shown by the following table.

3.2.3 Permutation layer P

Finally, the output of the S-box layer is rearranged to the permutation layer (Table 2). Following this operation, bit i of the current state is moved to the bit position $P(i)$.

The key schedule algorithm supports two key lengths of 80 and 128 bits. However, we do not explain the key schedule algorithm here because it is not directly relevant to our attack. For more information about the key schedule, the interested reader can refer to [19].

4 The basic DFA attack

In this section, we introduce a basic DFA attack against PRESENT-80 which is based on the injection of a fault on only one bit of intermediate states at the beginning of the last round's S-box layer. The given attack requires a single-bit fault in the last round of the cipher which may sound difficult in practice. However, the recent results [26] show the feasibility of this type of DFA attacks.

The main idea of the given attack is to obtain S^{31} by searching in the differences distribution table of the S-box (see Appendix 1), then we can obtain K^{32}. Injection of a single-bit fault on the jth nibble of S^{31}, $S^{31}N^j$, can be on $S^{31}N^j_0$, $S^{31}N^j_1$, $S^{31}N^j_2$, or $S^{31}N^j_3$. We denote the difference between the jth nibble of the correct state $S^{31}N^j$ and the jth nibble of the faulty state $S^{31*}N^j$ by a, i.e., $a = S^{31}N^j \oplus S^{31*}N^j$. Hence, there are four possible fault differences for this nibble, $a \in \{1, 2, 4, 8\}$. In Figure 2, an example of the fault required for the basic attack is given where the first bit of the first nibble of S^{31} is corrupted.

Through the attack, we assume that the master key and the input plaintext remain fixed. Moreover, for each faulty cipher text, exactly one bit of S^{31} is randomly modified. To recover the last round key, following this attack,

Table 1 The substitution layer (S-box)

	S-box action															
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

Table 2 The permutation layer

	***i* and *P*(*i*)**															
i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
P(*i*)	0	16	32	48	1	17	33	49	2	18	34	50	3	19	35	51
i	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
P(*i*)	4	20	36	52	5	21	37	53	6	22	38	54	7	23	39	55
i	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
P(*i*)	8	24	40	56	9	25	41	57	10	26	42	58	11	27	43	59
i	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
P(*i*)	12	28	44	60	13	29	45	61	14	30	46	62	15	31	47	63

the adversary processes each nibble of intermediate state independently. To implement this attack, the adversary should obtain each nibble of S^{31}, $S^{31}N^j$ for $1 \leq j \leq 16$. Obtaining each nibble would be possible if we are given, on average, three distinct faulty cipher texts, assuming that the faults are injected on different bits of the nibble. To find out the nibble of S^{31} which is involved in the faults, the adversary may reverse the difference output of P layer (P layer$^{-1}(C \oplus D)$) following the approaches presented in Algorithms 1 and 2 to obtain one nibble of S^{31}. Algorithm 1 arranges the faulty cipher texts and correct the cipher text in 16 groups of difference outputs of last round's S-box layer, i.e., $\Delta B^{31}N^j$ for $1 \leq j \leq 16$. In each specific group, the bits of faults occurred on the bits of the same nibble and the related nibble of S^{31}, $S^{31}N^j$, can be obtained following Algorithm 2. On the other hand, Algorithm 1 determines for each faulty cipher text, which the nibble has been corrupted. Hence, at the end of Algorithm 1, we expect to receive three difference outputs of last round's S-box layer for each nibble. It is possible to repeat Algorithm 2 and retrieve whole nibbles of S^{31}. Given S^{31} and the correct cipher text C, K^{32} can be extracted as follows:

$$K^{32} = P - \text{Layer}(S - \text{Layer}(S^{31})) \oplus C$$

Hence, following the given attack, it is possible to retrieve K^{32} given on average 48 faulty cipher texts and related correct cipher text for the fixed value of input message P (Additional details of the basic attack are given in Appendix 2).

If we can induce a fault on one bit of several nibbles of S^{31} for each fault experiment, we can reduce the required number of faulty cipher texts to obtain S^{31} and K^{32}. In this case, we can still use Algorithms 1 and 2 for obtaining S^{31} and K^{32}. Obviously, the injection of a single bit fault for a nibble of S^{31} generates a faulty nibble at the output of last round's S-box layer, and single bit faults injection on several nibbles of S^{31} generates several faulty nibbles at the output of last round's S-box layer.

Algorithm 1 Determining the faulty output of last round's S-box layer

Step 1. Consider the correct and faulty cipher text, and compute the difference output of last round's S-box layer by $\Delta B^{31} = P - \text{Layer}^{-1}(C \oplus D)$.

Step 2. Find the nibble(s) of S^{31} which is (are) involved in the fault(s), and save this (these) nibble(s) of the difference output(s) of last round's S-box layer, $\Delta B^{31}N^j$, in related group(s).

Step 3. Consider another faulty cipher texts, and repeat step1 and step 2 till when there are at least 3 distinct recodes for each nibble of the difference output(s) of last round's S-box layer.

Step 4. Apply Algorithm 2 on each group.

Algorithm 2 Obtain one nibble of S^{31}

Step 1. Consider a member of this group ($\Delta B^{31}N^j$). Then set up a list L containing all pairs that makes this difference output of S-box by searching in the differences distribution table of the S-box (remember that difference input can be $a \in \{1, 2, 4, 8\}$).

Step 2. Repeat step 1 with another member of this group, and store candidates in the new list M.

Step 3. Consider L and M Then remove non-member common of L.

Step 4. Go to step 2 until there remains only one candidate in L.

In Figure 3, the flowchart of this attack is depicted.

Given K^{32}, a similar attack can be used to retrieve S^{30} and K^{31} and finally recover the master key. However, the number of required faulty cipher texts would be increased (96 faulty cipher texts on average), but in the next section, we show an approach to obtain the master key given 18 faulty cipher texts on average.

Remarks 1. *It must be noted that sometimes faulty cipher texts will be repeated (if we inject a fault on the same bit*

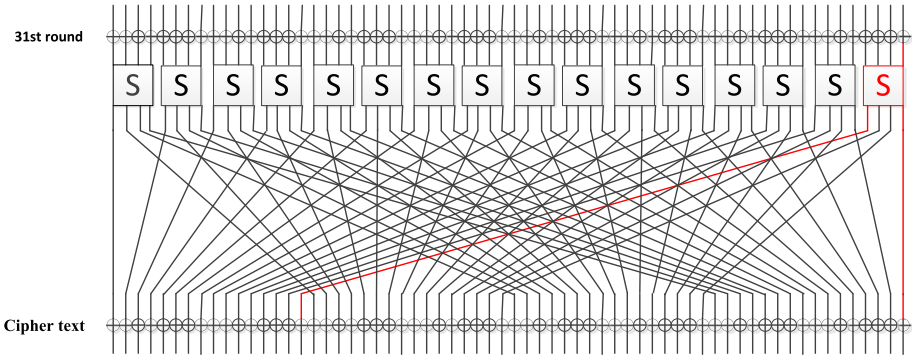

Figure 2 An example of an injection of a single bit fault on S^{31}.

of S^{31}), and we exclude this faulty cipher text. We collect three unique faulty cipher texts on average for obtaining each nibble of S^{31}.

5 The second DFA attack

In the previous section, we described a basic attack that, on each experiment, induces a fault on a single bit of intermediate state of the last round. In this section, to retrieve the whole master key, we extend the previous attack such that the adversary can inject a fault on one nibble rather than a bit.

In the design criteria of PRESENT, it has been stated that:

1. *The four input bits to an S-box come from four distinct S-boxes of the same group.*
2. *The input bits to a group of 4 S-boxes come from 16 different S-boxes.*
3. *The four output bits from a particular S-box enter four distinct S-boxes, each of which belongs to a distinct group of S-boxes in the subsequent round.*
4. *The output bits of S-boxes in distinct groups go to distinct S-boxes.*

Following the above criteria, we state that:

- Inducing a fault on only one nibble of intermediate states at the beginning of the ith round's S-box layer, S^i, leads to fault occurrence on one bit of some S-boxes of the next round, where on average, the inputs of two S-boxes would be corrupted at the beginning of S^{i+1}.
- Inducing a fault on only one nibble of intermediate states at the beginning of the ith round's S-box layer, S^i, leads to fault occurrences on one bit of some S-boxes of the next round, S^{i+1}, where on average the inputs of two S-boxes would be corrupted at the beginning of S^{i+1}. These differences are propagated to extra S-boxes in $(i+2)$th round. However, following the designing criteria, for any corrupted input of S-boxes at $(i+1)$th and $(i+2)$th round, we have difference on one bit. Therefore, the injection of a fault on a nibble of intermediate states at the beginning of the 29th round's S-box layer, on average, provides the adversary with faults on single bits of four nibbles of S^{31}.

Now, following the above argument, we assume that a fault on only one nibble of intermediate states at the beginning of the 29th round's S-box layer has occurred. We can use this approach to reduce the number of required faulty cipher texts in the basic attack (same as when we can induce the single bit faults on several nibbles of S^{31} in the basic attack). An example of fault injection at the beginning of the 29th round's S-box layer and its propagation toward 31st round has been depicted in Figure 4. In this picture, for the sake of simplicity, we have assumed that the first nibble of S^{29} is corrupted. It can be seen that eight nibbles of S^{31} have been corrupted, and also, only one bit of inputs of each faulty S-boxes at 31st round is corrupted. Therefore, we can apply the basic attack on the received data. However, in the new fault model, we receive on average four faulty nibbles at the output 31st round in each fault injection (four nibbles of the output of the last round's S-box layer were corrupted). Hence, the required number of faulty cipher texts to obtain the S^{31} is reduced to 12.

Whenever we obtained S^{31} and K^{32}, we can also retrieve K^{31}. To retrieve K^{31}, from the first phase of attack which retrieves K^{32}, we have 24 nibbles of the faulty output of the 30th round's S-box layer ($B^{30*}N^j$) on average, and we require extra 24 nibbles of the faulty output of the 30th round's S-box layer to obtain S^{30} and K^{31}. We can obtain those faulty states by inducing six extra faults on nibbles of intermediate state at the beginning of the 28th round's S-box layer on average.

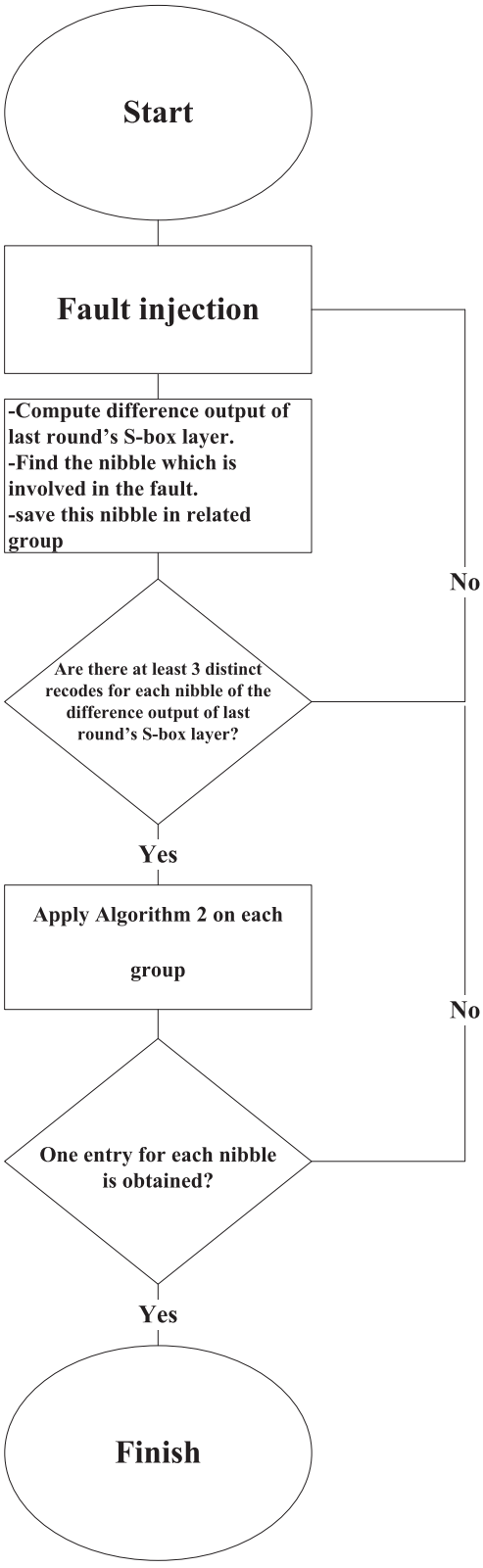

Figure 3 The flowchart of DFA on present.

Therefore, we can obtain K^{32} given 12 faulty cipher texts on average, where we have injected faults on the nibbles of S^{29}. To retrieve K^{31}, we require also to induce faults on the nibbles of intermediate state at the beginning of the 28th round's S-box layer. Following the basic attack (with some change in notation and extent formula, see Appendix 1), to retrieve K^{31}, we require additional six faulty cipher texts on average. Given the round keys K^{31} and K^{32}, it is possible to determine the master key uniquely. Hence, in total, it is possible to recover the master key of PRESENT-80 given 18 faulty cipher texts on average (An toy example of attacks are given in Appendix 3).

To compare our approach with [25], it must be noted that they have injected fault on the 29th round and tried to find the difference input of S-box 31st round where they have got some candidates for S-box inputs of 31st round and then recovered K^{32} and K^{31}. On the other hand, the given attack in this paper does not require to find the difference input of S-box 31st round, and it recovers the S-box input of 31st round by searching in the differences distribution table of S-box and then obtains K^{32}.

Any secure cryptosystems should be protected against DFA attacks. So, any implementation of PRESENT also requires to protect the last round of encryption against the basic DFA attack and protect the 29th and 30th rounds of encryption against the second DFA attack. Therefore, an implementation of PRESENT requires to protect the three last rounds against our DFA attacks.

Remarks 2. *Sometimes, we may receive useless faulty nibbles of cipher texts through the experiments which increase the attack complexity, i.e., number of required faulty cipher texts. Based on the random nature of the faults, and also the faults propagation property, it is possible to receive same errors on the same nibbles of the 31st round in different experiments. It should be noted that the errors that we induce in different experiments could be in different nibbles or even in the same nibble. These errors are propagated through the remaining rounds. However, if we receive several errors in a same bit of a nibble at the beginning of the 31st round, then only one of them is useful and the rest do not include new information. For example, following Figure 4 which is drawn for a fault on the first nibble of the internal state of 29th round, if we induce another fault on the same nibble in the next experiment and assume that the output difference of the related S-box at round 29 is 3, it has overlapped with the previous fault and it does not provide us with new information. It must be noted that we collect the unique nibble of faulty cipher texts and unique nibbles of faulty cipher texts on average to obtain one nibble of* S^{31}.

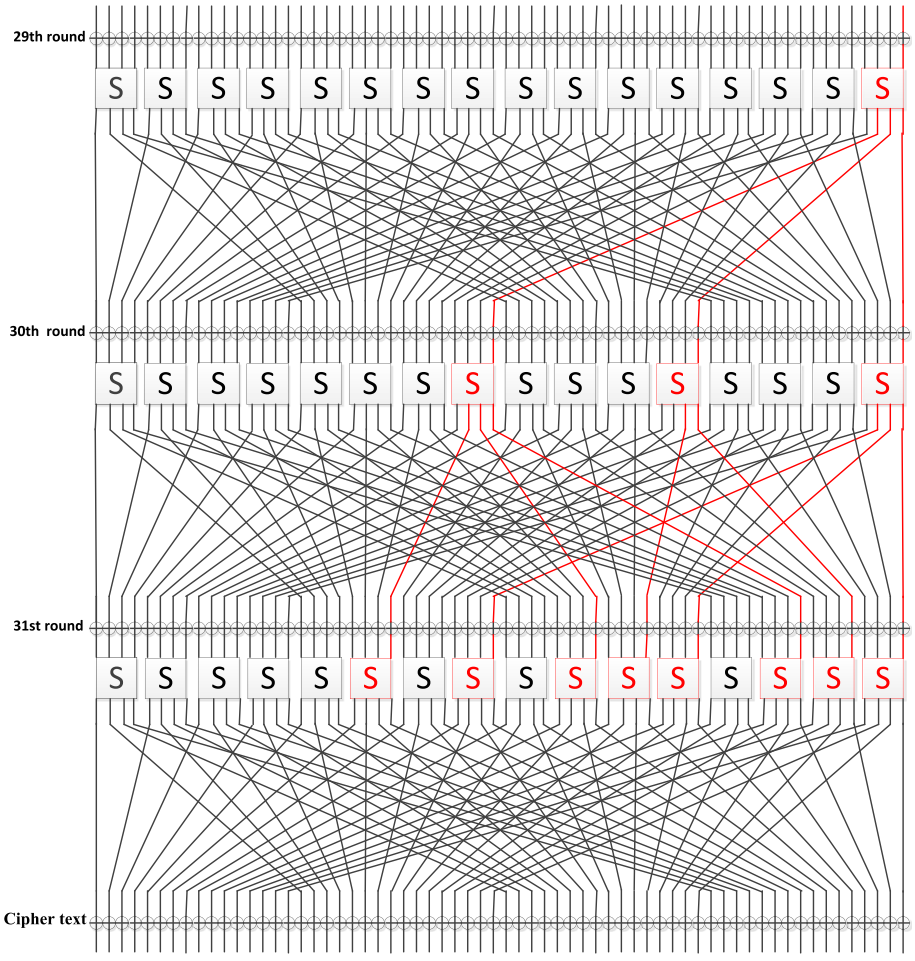

Figure 4 The example of our fault model.

6 Conclusions

In this paper, we introduced two new DFAs on PRESENT-80. Our attacks are based on the injection of the faults on the intermediate state of the cipher and can retrieve the last round key and the master key efficiently. The basic attack which requires to induce a fault on a single bit of nibbles at the input of the last round requires 48 faulty cipher texts on average, and the extended attack which retrieves the master key induces a fault on a nibble of the intermediate states and requires 18 faulty cipher texts on average.

Appendix 1

The difference distribution table of the S-box and the categorization algorithm for the faulty cipher texts

The difference distribution table of the S-box is shown in Table 3. The categorization algorithm for the faulty cipher texts is shown in Algorithm 3.

Algorithm 3 Categorize the faulty cipher texts and cipher text in 16 groups based on the difference output of the 30th round's S-box layer

Step 1. Consider the correct and faulty cipher text, and compute difference output of 30th round's S-box layer by $\Delta B^{30} = P - \text{Layer}^{-1}[S - \text{Layer}^{-1}(P - \text{Layer}^{-1}(C \oplus D)) \oplus K^{32}]$.

Step 2. Find nibble(s) of S^{30} which is (are) involved in the fault, and save this (these) nibble(s) of the difference output of 30th round's S-box layer, $\Delta B^{30} N^j$, in the related group(s).

Step 3. Consider another faulty cipher text, and repeat step1 and step 2 till when there are at least 3 distinct recodes for each nibble of difference output(s) of 30th round's S-box layer.

Step 4. Do Algorithm 4 for each group.

Algorithm 4 Obtain one nibble of S^{30}, given that K^{31}=*P*- Layer(*S*-Layer(S^{30}))⊕ S^{31}

Step 1. Consider a member of this group, $\Delta B^{30}N^{j}$, then set up a list L containing all pairs that makes this difference output of S-box by searching in the differences distribution table of the S-box (remember that difference input can be $a \in \{1, 2, 4, 8\}$).

Step 2. Repeat step 1 with another member of this group, and store candidates in the new list M.

Step 3. Consider L and M then remove non-common members from L.

Step 4. Go to step 2 until there remains only one candidate in L.

Appendix 2

Additional details of the basic attack

For each faulty cipher text, we have got a difference output of the last round's S-box layer, and we can determine possible inputs of the last round's S-box layer by searching in the differences distribution table of the S-box. In Table 4, all possible inputs of the last round's S-box layer for each difference output of the last round's S-box layer are shown (remember that difference input can be $a \in \{1, 2, 4, 8\}$).

In Table 4, it can be seen that given a faulty cipher text, eight candidates for S-box input with probability $\frac{5}{11}$ have been determined or four candidates for S-box input with probability $\frac{6}{11}$ have been determined.

In Table 5, all S-box difference outputs for each S-box input, with one bit difference in S-box input, are depicted.

Given in Table 5, we see that the sets of S-box difference outputs for each specific input are different. Hence, we can obtain the exact value of the S-box input with four faulty cipher texts. In the following, we explain the states that two and three faulty cipher texts are given.

- When two faulty cipher texts are given, these states can occur:
  - Two faulty cipher texts determine eight candidates for S-box input. In this state, when we apply Algorithm 2, we obtain two, three, four, or five candidates for S-box input.
  - The first faulty cipher text determined eight candidates for S-box input, and the second faulty cipher text determined four candidates for S-box input. If we denote by A the sets of the candidates obtained with the first faulty cipher text except the correct value of S-box input and by B the sets of the candidates obtained with the second faulty cipher text except the correct value of S-box input, we can identify the correct value with probability:

$$P = P(A \cap B = \varnothing) = P(|A \cap B| = 0) = \frac{\binom{16}{7} \times \binom{16-7}{3}}{\binom{16}{7} \times \binom{16}{3}} \simeq 15\%$$

  - The first faulty cipher text determined four candidates for S-box input, and the second

Table 3 The differences distribution table of the S-box

	0_x	1_x	2_x	3_x	4_x	5_x	6_x	7_x	8_x	9_x	A_x	B_x	C_x	D_x	E_x	F_x
0_x	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1_x	0	0	0	4	0	0	0	4	0	4	0	0	0	4	0	0
2_x	0	0	0	2	0	4	2	0	0	0	2	0	2	2	2	0
3_x	0	2	0	2	2	0	4	2	0	0	2	2	0	0	0	0
4_x	0	0	0	0	0	4	2	2	0	2	2	0	2	0	2	0
5_x	0	2	0	0	2	0	0	0	0	2	2	2	4	0	0	0
6_x	0	0	2	0	0	0	2	0	2	0	0	4	2	0	0	4
7_x	0	4	2	0	0	0	2	0	2	0	0	0	2	0	0	4
8_x	0	0	0	2	0	0	0	2	0	2	0	4	0	2	0	4
9_x	0	0	2	0	4	0	2	0	2	0	0	0	2	0	4	0
A_x	0	0	2	2	0	4	0	0	2	0	2	0	0	2	2	0
B_x	0	2	0	0	2	0	0	0	4	2	2	2	0	2	0	0
C_x	0	0	2	0	0	4	0	2	2	2	2	0	0	0	2	0
D_x	0	2	4	2	2	0	0	2	0	0	2	2	0	0	0	0
E_x	0	0	2	2	0	0	2	2	2	2	0	0	2	2	0	0
F_x	0	4	0	0	4	0	0	0	0	0	0	0	0	0	4	4

Table 4 All possible inputs of last round's S-box layer for each difference output of last round's S-box layer

Difference output of the S-box	0011	0101	0110	0111	1001	1010	1011	1100	1101	1110	1111
All possible S-box inputs	1100	1100	0011	0111	0000	0010	0001	0010	0010	0001	0111
	1101	1110	0111	0110	0001	0000	1001	0110	0011	0011	1111
	1110	1101	1001	1010	0100	1011	0110	1000	1001	1010	1000
	1111	1111	1011	1011	0101	1111	1110	1010	1000	1110	0000
that make this difference output	0100	0001		1000	1001				0101		
	0110	0101		1100	1101				0111		
	1011	0000		0101	0010				1100		
	0011	0100		1101	1010				1000		

faulty cipher text determined four candidates for S-box input.
In this state, we can identify the correct value with probability:

$$P = P(A \cap B = \varnothing) = P(|A \cap B| = 0) = \frac{\binom{16}{3} \times \binom{16-3}{3}}{\binom{16}{3}^2} \simeq 51\%$$

- When three faulty cipher texts are given, these states can occur:
  - All faulty cipher texts determined eight candidates for S-box input.
    If we denote by C the sets of the candidates obtained with the third faulty cipher text except the correct value of S-box input, we can identify the correct value with probability:

$$P = P(A \cap B \cap C = \varnothing) = P(|A \cap B \cap C| = 0) = \sum_{k=1}^{4} P(|A \cap B| = k, |A \cap B \cap C| = 0) = \sum_{k=1}^{4} P(|A \cap B| = k) \times P(|A \cap B \cap C| = 0 \,/\, |A \cap B| = k) = \frac{\binom{16}{7} \times \binom{7}{k} \times \binom{16-7}{7-k}}{\binom{16}{7}^2} \times \frac{\binom{16}{k} \times \binom{16-k}{7}}{\binom{16}{k} \times \binom{16}{7}} \simeq 18\%$$

  - The first and second faulty cipher texts determine eight candidates for S-box input, and the third faulty cipher text determine four candidates for S-box input.
    In this state, we can identify the correct value with probability:

$$P = P(A \cap B \cap C = \varnothing) = P(|A \cap B \cap C| = 0) = \sum_{k=1}^{4} P(|A \cap B| = k, |A \cap B \cap C| = 0) = \sum_{k=1}^{4} P(|A \cap B| = k) \times P(|A \cap B \cap C| = 0 \,/\, |A \cap B| = k) = \frac{\binom{16}{7} \times \binom{7}{k} \times \binom{16-7}{7-k}}{\binom{16}{7}^2} \times \frac{\binom{16}{k} \times \binom{16-k}{3}}{\binom{16}{k} \times \binom{16}{3}} \simeq 50\%$$

  - The first faulty cipher texts determine eight candidates for S-box input, and the second and third faulty cipher texts determine four candidates for S-box input.
    In this state, we can identify the correct value with probability:

$$P = P(A \cap B \cap C = \varnothing) = P(|A \cap B \cap C| = 0) = \sum_{k=1}^{4} P(|A \cap B| = k, |A \cap B \cap C| = 0) = \sum_{k=1}^{4} P(|A \cap B| = k) \times P(|A \cap B \cap C| = 0 \,/\, |A \cap B| = k) = \frac{\binom{16}{7} \times \binom{7}{k} \times \binom{16-7}{3-k}}{\binom{16}{7} \times \binom{16}{3}} \times \frac{\binom{16}{k} \times \binom{16-k}{3}}{\binom{16}{k} \times \binom{16}{3}} \simeq 62\%$$

Table 5 All possible S-box difference outputs for each S-box input

S-box input	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
All possible difference	0011	1010	1001	0011	0011	0101	0011	0110	0111	0110	0111	0011	0011	0011	0011	0011
	1001	1001	1101	0110	0101	0111	0111	0111	1100	1001	1001	0111	0101	0101	0101	1010
output of the S-box	1110	1110	1010	1110	1001	1001	1011	1111	1111	1101	1100	0110	0111	0111	1110	0101
	1111	1011	1100	1101	1101	1101	1100	1101	1101	1011	1110	1010	1101	1001	1011	1111

Table 6 The details of our toy example

Cipher text	***xxx*0*xxx*1*xxx*0*xxx*1**			
Fault location	$S^{31}N_0^0$	$S^{31}N_1^0$	$S^{31}N_2^0$	$S^{31}N_3^0$
Faulty cipher texts	*xxx*0*xxx*0*xxx*1*xxx*0	*xxx*1*xxx*0*xxx*0*xxx*1	*xxx*1*xxx*0*xxx*1*xxx*1	*xxx*1*xxx*1*xxx*0*xxx*0
$P-\text{Layer}^{-1}(C\oplus D)$	0000000000000111	0000000000001100	0000000000001110	0000000000001001

Table 7 The details of Algorithm 2

Lists	***L***	***M***	**Update *L***	**New *M***	**Update *L***
Difference output of the S-box	0111	1001		1100	
All pairs of S-box input that make this difference output	0111	0001	1010	1010	1010
	0110	0000	0101	1000	
	1010	0100	1101	0001	
	1011	0101		0110	
	1000	1001			
	1100	1101			
	0101	0010			
	1101	1010			

Table 8 The key and more details of our example

Plaintext	***FFFFFFFFFFFFFFFF***
Key	00000000000000000000
S^{29}	*D*79*F*5742*CEE*5*A*802
S^{30}	60991*D*31835*A*96*DD*
S^{31}	*DE*21*CD*50*C*1*FDB*05*F*
K^{31}	8*BA*27*A*0*EB*8783*AC*9
K^{32}	6*DAB*31744*F*41*D*700
Cipher text	*A*112*FFC*72*F*68417*B*

Table 9 The value of fault in $S^{29}N^j$

j	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
The value of fault in $S^{29}N^j$	3	*B*	1	*A*	2	6	*C*	*D*	8	7	5	3	*F*	9	4	*B*

Table 10 The details of Algorithms 1 and 2

Algorithm	***j***	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Algorithm 1	$\Delta B^{31}N^j$	3	9	9	3	5	*A*	5	3	*A*	5	3	*D*	*B*	9	*E*	3
		5	*D*	5	6	7	3	*E*	5	5	7	5	5	5	*A*	3	7
		A	5	*A*	7	9	5	*B*	7	9	9	7	7	9	*C*	*B*	5
		F	7	*F*	*A*				*D*	*F*	*D*	9			*D*	5	9
Algorithm 2	$S^{31}N^j$	*F*	5	0	*B*	*D*	*F*	1	*C*	0	5	*D*	*C*	1	2	*E*	*D*
						5		*E*					5				

Appendix 3

A toy example of attacks

Assume that a message M is encrypted by PRESENT-80, the basic attack works as follows. For the sake of simplicity, we assume that the first nibble of S^{31}, $S^{31}N^0$, is A in hexadecimal ($(1010)_2$ in binary). So, if we inject a single-bit fault on $S^{31}N^0$, we obtain a faulty cipher text D. If we do this fault injection to other bits of $S^{31}N^0$, we have four faulty cipher texts that is shown in the following table (Table 6, note that throughout the attack, we do not know the fault location, and we put this value in this table only for the reader):

So we show each step of Algorithm 2 in the following table (Table 7):

After we obtain $S^{31}N^0$, we obtain 4 bits of K^{32}.

For verifying the second attack, we present an example that the message, key, and some details are shown in Table 8. We injected 16 faults on S^{31} in random and have got 16 faulty cipher texts. In Table 9, the value and location of faults were given (this table was given just for the reader); then, in Table 10, the results of Algorithms 1 and 2 were given.

In Table 10, two candidates existed for nibbles 4, 6, and 11 of S^{31}, so for obtaining the exact value of these nibbles, we have to inject at least three faults on S^{29}. In total, we obtain 13 nibbles of K^{32}, 9 nibbles of K^{31}, and two candidates for some nibbles of K^{32} and K^{31} with 16 faulty cipher texts (in this example). For recovering whole of K^{31}, we have to inject some faults as mentioned in Section 5.

Competing interests

The authors declare that they have no competing interests.

Received: 18 March 2012 Accepted: 4 July 2013
Published: 4 September 2013

References

1. D Boneh, RA DeMillo, RJ Lipton, ed. by W Fumy. On the importance of checking cryptographic protocols for faults (extended abstract). In *EUROCRYPT* (Springer Heidelberg, 1997), pp. 37–51
2. E Biham, A Shamir, ed. by BS Kaliski Jr. Differential fault analysis of secret key cryptosystems. In *CRYPTO* (Springer Heidelberg, 1997), pp. 513–525
3. L Hemme, ed. by M Joye, JJ Quisquater. A differential fault attack against early rounds of (Triple-)DES. In *CHES* (Springer Heidelberg, 2004), pp. 254–267
4. M Rivain, ed. by C Clavier, K Gaj. Differential fault analysis on DES middle rounds. In *CHES* (Springer Heidelberg, 2009), pp. 457–469
5. P Dusart, G Letourneux, O Vivolo, ed. by J Zhou, M Yung, and Y Han. Differential fault analysis on A.E.S. In *ACNS* (Springer Heidelberg, 2003), pp. 293–306
6. C Giraud, ed. by H Dobbertin, V Rijmen, and A Sowa. DFA on AES. In *AES Conference* (Springer Heidelberg, 2004), pp. 27–41
7. C Giraud, A Thillard, Piret and quisquater's DFA on AES revisited. IACR Cryptology ePrint Arch. **2010**, 440 (2010)
8. CH Kim, Differential fault analysis of AES: Toward reducing number of faults. IACR Cryptology ePrint Arch. **2011**, 178 (2011)
9. CH Kim, J-J Quisquater, ed. by G Grimaud, FX Standaert. New differential fault analysis on AES key schedule: two faults are enough. In *CARDIS* (Springer Heidelberg, 2008), pp. 48–60
10. G Piret, J-J Quisquater, ed. by CD Walter, CK Koc, and C Paar. A differential fault attack technique against SPN structures, with application to the AES and KHAZAD. In *CHES* (Springer Heidelberg, 2003), pp. 77–88
11. M Tunstall, D Mukhopadhyay, S Ali, ed. by CA Ardagna, J Zhou. Differential fault analysis of the advanced encryption standard using a single fault. In *WISTP* (Springer Heidelberg, 2011), pp. 224–233
12. N Bagheri, R Ebrahimpour, N Ghaedi, Differential fault analysis on PRINTcipher. IET Netw. **2**(1) (2013)
13. X jie Zhao, T Wang, An improved differential fault attack on camellia. IACR Cryptology ePrint Arch. **2009**, 585 (2009)
14. X jie Zhao, T Wang, J zhe Gao, Multiple bytes differential fault analysis on CLEFIA. IACR Cryptology ePrint Arch. **2010**, 78 (2010)
15. J Takahashi, T Fukunaga, Differential fault analysis on CLEFIA with 128, 192, and 256-bit keys. IEICE Trans. **93-A**(1), 136–143 (2010)
16. E Biham, L Granboulan, PQ Nguyen, in *Lecture Notes in Computer Science,* vol. 3557, ed. by H Gilbert, H Handschuh. FSE. Impossible fault analysis of RC4 and differential fault analysis of RC4 (Springer Heidelberg, 2005), pp. 359–367
17. R Li, B Sun, C Li, J You, Differential fault analysis on SMS4 using a single fault. Inf. Process. Lett. **111**(4), 156–163 (2011)
18. W Li, D Gu, J Li, Differential fault analysis on the ARIA algorithm. Inf. Sci. **178**(19), 3727–3737 (2008)
19. A Bogdanov, LR Knudsen, G Leander, C Paar, A Poschmann, MJB Robshaw, Y Seurin, C Vikkelsoe, ed. by P Paillier, I Verbauwhede. PRESENT: an ultra-lightweight block cipher. In *CHES* (Springer Heidelberg, 2007), pp. 450–466
20. JY Cho, ed. by J Piepryzk. Linear cryptanalysis of reduced-round PRESENT. In *CT-RSA* (Springer Heidelberg, 2010), pp. 302–317
21. O Özen, K Varici, C Tezcan, Çelebi Kocair, ed. by C Boyd, JG Nieto. Lightweight block ciphers revisited: cryptanalysis of reduced round PRESENT and HIGHT. In *ACISP* (Springer Heidelberg, 2009), pp. 90–107
22. M Wang, Differential cryptanalysis of PRESENT. IACR Cryptology ePrint Arch. **2007**, 408 (2007)
23. M Wang, ed. by MM Wang. Differential cryptanalysis of reduced-round PRESENT. In *AFRICACRYPT* (Springer Heidelberg, 2008), pp. 40–49
24. G Wang, S Wang, ed. by W Schindler, SA Huss. Differential fault analysis on PRESENT key schedule. In *CIS* (Springer Heidelberg, 2010), pp. 362–366
25. X jie Zhao, T Wang, S Guo, Fault-propagation pattern based DFA on SPN structure block ciphers using bitwise permutation, with application to PRESENT and PRINTcipher. IACR Cryptology ePrint Arch. **2011**, 86 (2011)
26. AP Mirbaha, J-M Dutertre, A-L Ribotta, M Agoyan, A Tria, D Naccache, in *IEEE International Conference on Technologies for Homeland Security,* Single-bit DFA using multiple-byte laser fault injection. Waltham, 8–10 Nov 2010

doi:10.1186/1687-6180-2013-145
Cite this article as: Bagheri *et al.*: New differential fault analysis on PRESENT. *EURASIP Journal on Advances in Signal Processing* 2013 **2013**:145.