# Differential fault attack on DEFAULT

Chandan Dey[1], Sumit Kumar Pandey[2], Tapabrata Roy[1] and Santanu Sarkar[1]

[1] Indian Institute of Technology Madras, Chennai, India

[2] Indian Institute of Technology Jammu, Jammu, India

**Abstract**

Block cipher DEFAULT has been proposed as a differential fault analysis immune cipher at Asiacrypt 2021. In this paper, we consider the initial version of DEFAULT with no permutation involved in the last round and show that one can find the key in this version with complexity $2^{16}$ by injecting 112 faults. However, our idea does not work in the modified version of the cipher.

**Keywords:** Block cipher, DEFAULT, Differential fault attack, Linear structure
**Mathematics Subject Classification:** 94A60.

## 1 Introduction

A lot of research has been done since the first academic publication in the year 1996 by Boneh, DeMillo and Lipton [5] which discussed how to use a differential fault intentionally to break a cryptographic algorithm. At that time, the attack was mainly theoretical, but since then the technology has improved and most of these attacks are being used in practice. The techniques of introducing faults have been prospered with an intention of amendment of the proper functioning of cryptographic devices. These procedures include alterations in the voltage level of power supply, irregularity injections in the clock gestures, overheating the device, introduction of electromagnetic disturbances, exposing the device to immense luminous medium, etc.

The differential fault attack (DFA) on symmetric key cryptosystems [4] was introduced by Biham and Shamir. This was one of the earliest techniques that was invented to attack a block cipher by provoking a computational error. Later Piret et al. [6] introduced DFA against SPN based block cipher. In 2009 Mukhopadhyay [7] improved the fault based attack on AES. The idea of differential fault analysis is same as the classical differential cryptanalysis. The idea behind DFA is to obtain two ciphertexts out of which one will be a correct one whereas the other will be a faulty one. The faulty ciphertext is obtained

after introducing a possibly known differential in one of the rounds of the input. The adversary tries to guess the key bits by analyzing how the differential propagates over a small number of rounds.

In recent years, a section of cipher designers [3, 8] have concentrated upon building ciphers which can resist differential fault attacks. The block cipher DEFAULT [1] is one such attempt in this direction. The designers of DEFAULT claimed it be to be DFA-resistant. In this work, we consider the initial version of DEFAULT with no permutation involved in the last round and show that one can find the key in this version with complexity $2^{16}$ by injecting 112 faults.

# 2 Design of DEFAULT

DEFAULT is an SPN structured block cipher that uses SBoxes having Linear Structures to ensure both, protection against Differential Fault Analysis and as well as Differential Attacks. The design of DEFAULT consists of mainly two parts, namely DEFAULT LAYER and DEFAULT CORE. The DEFAULT LAYER is meant to corroborate security against the Differential Fault Attacks, whereas the DEFAULT CORE guarantees protection against other classical attacks. Thus, the DEFAULT LAYER, $\mathcal{L}$ is prepended and appended to the DEFAULT CORE, $\mathcal{E}$ during encryption. While decrypting, the inverse of the DEFAULT LAYER, $\mathcal{L}^{-1}$ is both prepended and appended to the inverse of the main cipher $\mathcal{E}^{-1}$. Thus, the encryption and decryption for DEFAULT are given as follows:

$$C = \mathcal{L} \circ \mathcal{E} \circ \mathcal{L}(P), \text{ and } \mathcal{L}^{-1} \circ \mathcal{E}^{-1} \circ \mathcal{L}^{-1}(C) = (\mathcal{L}^{-1} \circ \mathcal{E}^{-1} \circ \mathcal{L}^{-1}) \circ (\mathcal{L} \circ \mathcal{E} \circ \mathcal{L}(P)) = P,$$

$P$ and $C$ being the plaintext and the ciphertext respectively. Below we describe these two parts in details.

## 2.1 DEFAULT LAYER

DEFAULT LAYER is the portion of DEFAULT that has been intended to provide immunity against the Differential Fault Attacks. It comprises of an SBox and a permutation which takes a 128-bit message along with a 128-bit key as input. The state, $X$ is then further divided into 32 4-bit nibble words. DEFAULT LAYER involves a round function $\mathcal{R}$ consisting of four steps viz., SubCells, PermBits, AddRoundConstants and AddRoundKey. Each round is run 28 times before and after the DEFAULT CORE (Main Cipher).

**SubCells:** First a 128-bit plaintext, $X = x_{127}x_{126} \ldots x_0$; $x_0$ being the least significant bit, is fed into the algorithm which is further split into 32 four-bit nibble word as $X = \mathbf{x_{31}}||\mathbf{x_{30}}|| \ldots ||\mathbf{x_0}$, where each $\mathbf{x_i}$ is a 4-bit nibble word, $\mathbf{i} = 0, 1, \ldots, 31$. SubCells applies the 4-bit LS SBox, $S = 037ED4A9CF18B265$ to every nibble of the state: $\mathbf{x_i} \mapsto S(\mathbf{x_i})$, for $\mathbf{i} \in \{0, 1, \ldots, 31\}$.

**PermBits:** Following the SubCells operation we obtain a 128-bit word which is next permuted by the PermBits operation where the $i$-th bit of a state is mapped to the $P_{128}(i)$-th bit ($x_i \mapsto x_{P_{128}(i)}$ for all $i \in \{0, 1, \ldots, 127\}$) where the bit-permutation, $P_{128}$ is taken as the permutation of Gift-128 [2]. The permutation is provided in Table 1.

| $i$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $P_{128}(i)$ | 0 | 33 | 66 | 99 | 96 | 1 | 34 | 67 | 64 | 97 | 2 | 35 | 32 | 65 | 98 | 3 | 4 | 37 | 70 | 103 | 100 | 5 | 38 | 71 | 68 | 101 |
| $i$ | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 50 | 51 |
| $P_{128}(i)$ | 6 | 39 | 36 | 69 | 102 | 7 | 8 | 41 | 74 | 107 | 104 | 9 | 42 | 75 | 72 | 105 | 10 | 43 | 40 | 73 | 106 | 11 | 12 | 45 | 78 | 111 |
| $i$ | 52 | 53 | 54 | 55 | 56 | 57 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 | 73 | 74 | 75 | 76 | 77 |
| $P_{128}(i)$ | 108 | 13 | 46 | 79 | 76 | 109 | 14 | 47 | 44 | 77 | 110 | 15 | 16 | 49 | 82 | 115 | 112 | 17 | 50 | 83 | 80 | 113 | 18 | 51 | 48 | 81 |
| $i$ | 78 | 79 | 80 | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 90 | 91 | 92 | 93 | 94 | 95 | 96 | 97 | 98 | 99 | 100 | 101 | 102 | 103 |
| $P_{128}(i)$ | 114 | 19 | 20 | 53 | 86 | 119 | 116 | 21 | 54 | 87 | 84 | 117 | 22 | 55 | 52 | 85 | 118 | 23 | 24 | 57 | 90 | 123 | 120 | 25 | 58 | 91 |
| $i$ | 104 | 105 | 106 | 107 | 108 | 109 | 110 | 111 | 112 | 113 | 114 | 115 | 116 | 117 | 118 | 119 | 120 | 121 | 122 | 123 | 124 | 125 | 126 | 127 | | |
| $P_{128}(i)$ | 88 | 121 | 26 | 59 | 56 | 89 | 122 | 27 | 28 | 61 | 94 | 127 | 124 | 29 | 62 | 95 | 92 | 125 | 30 | 63 | 60 | 93 | 126 | 31 | | |

Table 1: DEFAULT permutation

**AddRoundConstant:** Subsequently, the PermBits operation yields a permuted 128-bit binary string. AddRoundConstant operation XORs a single bit "1" and a 6-bit round constant $C = c_5c_4c_3c_2c_1c_0$ to this cipher state at bit positions 127, 23, 19, 15, 11, 7 and 3 respectively: $x_{127} = x_{127} \oplus 1$, $x_{23} = x_{23} \oplus c_5$, $x_{19} = x_{19} \oplus c_4$, $x_{15} = x_{15} \oplus c_3$, $x_{11} = x_{11} \oplus c_2$, $x_7 = x_7 \oplus c_1$ and $x_3 = x_3 \oplus c_0$. Round constants are given in Table 2. These values are expressed as six bit words $C$ in each round and is added to the cipherstate.

**AddLayerKey:** Next, a layer key, $k^t = k^t_{127}k^t_{126}\ldots k^t_0$ is extracted from the Master Key (if the bit-length of the Master Key is more than 128 bits, then the first 128 bits are simply extracted) and XORed to the cipherstate bitwise. Intial version of DEFAULT LAYER does not have any key scheduling algorithm. So, the master key is XORed directly to state. In the modified version, authors proposed a key scheduling algorithm where 4 subkeys $K_0, K_1, K_2, K_3$ are generated from the 128 bit master key $K$. Here $K_0 = K$ and $K_{i+1} = \mathcal{R}'(\mathcal{R}'(\mathcal{R}'(\mathcal{R}'(K_i))))$ for $0 \leq i \leq 2$, where $\mathcal{R}'$ is a round function with all zero round key and all zero round constant except for the last bit (where only a single bit 1 is Xored). At round $i \geq 0$, subkey $K_{i \bmod 4}$ is used as the round key.

| Cipher | Round constants | # of rounds |
|---|---|---|
| DEFAULT LAYER | 1, 3, 7, 15, 31, 62, 61, 59, 55, 47, 30, 60, 57, 51, 39, 14, 29, 58, 53, 43, 22, 44, 24, 48, 33, 2, 5, 11 | 28 |
| Default CORE | 1, 3, 7, 15, 31, 62, 61, 59, 55, 47, 30, 60, 57, 51, 39, 14, 29, 58, 53, 43, 22, 44, 24, 48 | 24 |

Table 2: DEFAULT round constants

**DEFAULT CORE:** DEFAULT CORE is the middle part of DEFAULT and has been designed to prevent the classical cttacks. The design of DEFAULT CORE is mostly similar with that of DEFAULT LAYER. The only difference here is in the SBox, number of rounds and the key schedule. Here, the number of rounds is 24 and the involved Sbox is $S = 196F7C82AED$ and the round constants are listed in Table 2. For more details we refer to [1].

# 3 Idea of the Attack

For the sake of brevity we first define the Linear Structure of an SBox that we have used throughout of the paper.

**Definition 1.** *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a vectorial Boolean function, $n \in \mathbb{N}$, an element $\alpha \in \mathbb{F}_2^n$ is called a Linear Structure of $F$ if $F(x) \oplus F(x \oplus \alpha) = c$ holds $\forall x \in \mathbb{F}_2^n$ and for some constant $c \in \mathbb{F}_2^n$.*

## 3.1 Argument of [1] on DEFAULT against DFA

In this subsection we discuss differential fault attack (DFA) on DEFAULT LAYER. In [1], authors claimed that DEFAULT LAYER will give 64 bit security against DFA for 128 bit block cipher. Their arguments are as follows:

**DFA on DEFAULT LAYER:** As described earliar the DEFAULT LAYER consists of a total of 32 nibbles on each of which the same $4 \times 4$ SBox is operated. This SBox contains three non-zero linear structures namely $6, 9$ and $F$. Thus from the knowledge of $S(\alpha) \oplus S(\alpha \oplus \delta)$ for $0 \leq \alpha \leq F$, we have 4 options of input namely

$$\{\alpha, \alpha \oplus 6, \alpha \oplus 9, \alpha \oplus F\},$$

where $\delta$ corresponds to the fault. If the attacker injects faults in the last round and explore each of the 32 SBoxes separately, then for each SBox he/she will have 4 options for the 4 keybits. Therefore, to find the secret key he/she has to search among $4^{32} = 2^{64}$ possible keys.

**DFA on DEFAULT CORE:** SBoxes in DEFAULT CORE has no linear structure. So, the attacker can try to target this portion of the cipher. However, the attacker gets the output after the completion of DEFAULT LAYER. Using MILP, authors showed that even in this case, the attacker cannot find the key with complexity less than $2^{64}$.

## 3.2 Our attack idea

Here in this subsection we will demonstrate our fault model. We consider only the last two rounds of the DEFAULT LAYER as described below, one after another. For sake of simplicity during calculations, we do not XOR the corresponding round constants in the respective rounds, as adding constants does not provide better security against fault attacks. Also, we do not consider the

PermBits operation in the last round like GIFT [2]. Fault is usually injected in three possible ways, viz., bit based fault injection, nibble based fault injection and byte based fault injection. As the names suggest, faults are induced in a nibble, in a byte and simply in a bit respectively in the above three cases of fault injections. In our case of attack against DEFAULT, we consider nibble based fault injections only.

#### 3.2.1 Analysis at the last round

First, we contemplate the last round of the DEFAULT LAYER. Injecting faults in the beginning of the last round and then analysing the faulty ciphertexts reduces the key search complexity by a great margin. In this instance, we analyse each SBox separately as done by the designers. DEFAULT LAYER includes a 128-bit state size, i.e., 32 SBoxes get involved in each of the rounds. As mentioned above, $S$ denotes the SBox of the DEFAULT LAYER and $S^{-1}$ represents the inverse of that SBox. Let $m$ be the state at any random nibble. Clearly, there are 15 possible faults corresponding to that nibble, giving rise to fifteen different faulty states viz., $m \oplus i,\ i \in \{1, 2, \ldots, 15\}$. We denote these faults as $\delta_1, \delta_2, \ldots, \delta_{15}$. We consider the bits of the nibbles after the SBox operation. At those positions after adding the respective constants and also the keybits let $c$ be the corresponding 4-bit word. Similarly, let $c_1, c_2, \ldots, c_{15}$ be the 4-bit faulty words of the ciphertexts corresponding to the faults $\delta_1, \delta_2, \ldots, \delta_{15}$ respectively. If we consider the equation

$$S^{-1}(c_i \oplus k) \oplus S^{-1}(c \oplus k) = \delta_i \text{ for all } i \in \{1, 2, \ldots, 15\} \tag{1}$$

and try to find the possible choices of 4-bit keywords for $k$, those satisfy equation (1) for all $i \in \{1, 2, \ldots, 15\}$, then we get one among the following four partitions: $\{0, 5, A, F\}$, $\{1, 4, B, E\}$, $\{2, 7, 8, D\}$ and $\{3, 6, 9, C\}$.

**Lemma 1.** *Let $a \in \mathbb{F}_2^4$ be a linear structure for the SBox $S$. So, there exist $b \in \mathbb{F}_2^4$ satisfying*

$$S(x) \oplus S(x \oplus a) = b \ \forall\ x \in \mathbb{F}_2^4.$$

*Then, $a$ and $b$ satisfy*

$$S^{-1}(x) \oplus S^{-1}(x \oplus b) = a \ \forall\ x \in \mathbb{F}_2^4,$$

*i.e., $b$ is a linear structure for $S^{-1}$.*

*Proof.* As $S$ is a permutation, it is bijective and hence, so is $S^{-1}$. Thus, $\{S^{-1}(x) : x \in \mathbb{F}_2^4\} = \mathbb{F}_2^4$. So, from the equation

$$S(x) \oplus S(x \oplus a) = b \ \forall\ x \in \mathbb{F}_2^4$$

we have,

$$\begin{aligned} & S(S^{-1}(x)) \oplus S(S^{-1}(x) \oplus a) && = b \ \forall \ x \in \mathbb{F}_2^4, \\ \Leftrightarrow \quad & x \oplus S(S^{-1}(x) \oplus a) && = b \ \forall \ x \in \mathbb{F}_2^4, \\ \Leftrightarrow \quad & S(S^{-1}(x) \oplus a) && = x \oplus b \ \forall \ x \in \mathbb{F}_2^4, \\ \Leftrightarrow \quad & S^{-1}(x) \oplus a && = S^{-1}(x \oplus b) \ \forall \ x \in \mathbb{F}_2^4, \\ \Leftrightarrow \quad & S^{-1}(x) \oplus S^{-1}(x \oplus b) && = a \ \forall \ x \in \mathbb{F}_2^4. \end{aligned}$$

Hence, we have the desired result. □

**Remark 1.** *One can check* $S(x) \oplus S(x \oplus 6) = A, S(x) \oplus S(x \oplus 9) = F$ *and* $S(x) \oplus S(x \oplus F) = 5$ *for* $0 \leq x \leq F$. *From Lemma 1, we know* $S^{-1}$ *contains 3 non-zero linear structures namely* $5, A, F$. *The inverse of the SBox used in the DEFAULT LAYER is* $0AD15FE2B76C8439$. *One can see for any* $0 \leq x \leq F$, $S^{-1}(x) \oplus S^{-1}(x \oplus 5) = F, S^{-1}(x) \oplus S^{-1}(x \oplus A) = 6$ *and* $S^{-1}(x) \oplus S^{-1}(x \oplus F) = 9$.

**Theorem 1.** *Let* $b$ *be a linear structure for* $S$. *Then for the introduction of a single fault* $b$, *all the sixteen elements in the whole space* $\mathbb{F}_2^4$ *satisfy to be the possible key options for the fault analysis at the last round.*

*Proof.* Suppose $b$ be a linear structure for $S$. So, there exists $a$ in $\mathbb{F}_2^4$ such that

$$S(x) \oplus S(x \oplus b) = a \ \forall \ x \in \mathbb{F}_2^4.$$

Then by Lemma 1 we have,

$$S^{-1}(x) \oplus S^{-1}(x \oplus a) = b \ \forall \ x \in \mathbb{F}_2^4. \tag{2}$$

Let $m$ be a state in a nibble and $\delta$ be the induced fault injecting a faulty state $m' = m \oplus \delta$. These yields the original ciphertext, $c = S(m) \oplus k$ and a faulty ciphertext, $c' = S(m') \oplus k = S(m \oplus \delta) \oplus k$, $k$ being the 4-bit portion of the key corresponding to the considered nibble. Now, $S$ being a permutation, the set $\{S(x) : x \in \mathbb{F}_2^4\}$, and hence the set $\{S(x) \oplus k \oplus c : x \in \mathbb{F}_2^4\}$ contain all the 16 elements of $\mathbb{F}_2^4$. So, $\delta$ can be chosen in such a way that $S(m \oplus \delta) \oplus k \oplus c = a$, or in other words, $a = S(m') \oplus k \oplus c = c' \oplus c$, i.e., $c' = c \oplus a$. Next, from equation (1) we have,

$$S^{-1}(c \oplus k) \oplus S^{-1}(c' \oplus k) = \delta. \tag{3}$$

Thus, by the above argument,

$$S^{-1}(c \oplus k) \oplus S^{-1}(c \oplus k \oplus a) = \delta,$$

which forces $\delta$ to be equal to $b$, i.e.,

$$S^{-1}(c \oplus k) \oplus S^{-1}(c \oplus k \oplus a) = b. \tag{4}$$

But, from equation (2) it can be concluded that equation (4) must hold for all possible values of $c \oplus k$, and so, for all possible values of $k$ for a fixed value of $c$ (or in other words, for a fixed $m$). Hence, the theorem. □

**Remark 2.** *If $b_1$ and $b_2$ be two linear structures for $S$ and $a_1$ and $a_2$ be two elements in $\mathbb{F}_2^4$ satisfying $S(x) \oplus S(x \oplus b_i) = a_i$ and $S^{-1}(x) \oplus S^{-1}(x \oplus a_i) = b_i \ \forall \ x \in \mathbb{F}_2^4, \ i \in \{1, 2\}$, then injecting faults $b_1$ and $b_2$ we see that all the elements in $\mathbb{F}_2^4$ are solutions to equation* (1) *and as a result the intersection of the sets of possible keys corresponding to the above mentioned faults is $\mathbb{F}_2^4$. Clearly, consideration of each linear structure for $S$, as a fault, takes all the sixteen possible key candidates as options. Thus, as there are four linear structures of $S$, during the fault analysis it should be kept in mind that the faults must not be chosen among these four elements.*

**Observation 1.** *From Remark 2 it is evident that the faults at the last round must be chosen among the remaining twelve elements, viz., $\{1, 2, 3, 4, 5, 7, 8, A, B, C, D, E\}$, so that one does not end up in a situation where the intersection of the two sets of all possible 4-bits key candidates satisfying equation* (3) *is the whole set $\mathbb{F}_2^4$.*

*Consider the set $\mathcal{A} = \{(1, 7), (1, 8), (1, D), (2, 4), (2, B), (2, 13), (3, 5), (3, A), (3, C), (4, B), (4, D), (5, A), (5, C), (7, 8), (7, D), (8, D), (A, C), (B, D)\}$. We have observed that irrespective of any four bits input to a nibble if we induce two distinct faults $(x, y) \in \mathcal{A}$, the intersection of the two sets of solutions to 4-bit keywords satisfying equation* (3) *have eight elements and that of all other remaining pairs have four elements where the intersection is one of the following four partitions: $\{0, 5, A, F\}, \{1, 4, B, E\}, \{2, 7, 8, D\}$ and $\{3, 6, 9, C\}$.*

*Hence, if we simply induce a pair of faults from the set $\mathcal{F} = \{(x, y) | 1 \leq x, y \leq E, x \neq 6, x \neq 9, y \neq x, y \neq 6, y \neq 9\} \setminus \mathcal{A}$, we culminate in our desired situation.*

From the above observation, it is clear that there is no need to generate all possible faulty outputs for each SBox. So, in the last round only 64 faults are enough to find the all possible key candidates. For each SBox, the possible keys will be one of the four partitions mentioned above. Since in each partition there are four elements, so for each SBox, we have four possible key candidates. Hence, for 32 SBoxes there are $4^{32} = 2^{64}$ possible key candidates which is less than an exhaustive search.

### 3.2.2 Analysis at the penultimate round

Now, we consider the last two rounds of the initial version of DEFAULT LAYER where no key scheduling algorithm is involved. We try to filter the key candidates further from what we have got, following the above procedure after inducing faults in the last round. Here, we induce faults in the beginning of the second last round in a nibble and generate faulty ciphertexts. Now, from the faulty ciphertexts we apply two rounds inverse operation for the possible key candidates already obtained from the last round fault injection model and reduce the possible key candidates.

Suppose after 25 rounds $S^{26}$ be the 128-bit state and $s$ be any 4-bit nibble of the state. Let $c$ be the corresponding ciphertext. We induce two faults $(\delta_1, \delta_2) \in \mathcal{F}$ in the nibble $s$ of the state $S^{26}$ and generate the faulty ciphertexts

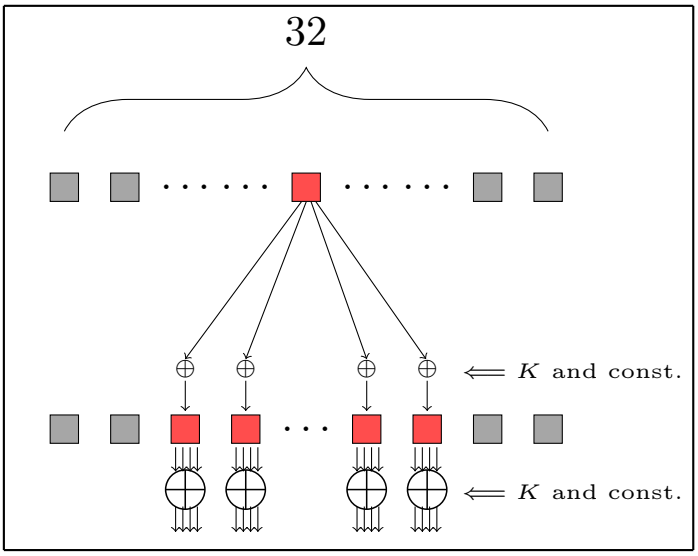


Figure 1: Fault at penultimate round. ■ corresponds to faulty nibble.

after 27-th round $c_1, c_2$. Now we use the following equation to filter the key candidates $k$ further,

$$\mathcal{R}^{-2}(c,k) \oplus \mathcal{R}^{-2}(c_i,k) = \delta_i \text{ for all } i \in \{1,2\}. \tag{5}$$

Therefore, the key candidates satisfying equation (5) for all $i \in \{1,2\}$ will be the possible filtered key candidates.

As mentioned above, here, we induce nibble based faults in the beginning state of the second last round which will activate 4 SBoxes in the last round. Then, the outputs of the four SBoxes be will be XOR-ed to 4 nibbles of the key $K$. Note that from what we have discussed earlier in the previous section we know that the fault analysis at the last round yields 4 possible options for each nibble of the key $K$ at respective positions. Next, we try to filter these $2^{64}$ many possible options for the key $K$ by inducing nibble based fault in the beginning state of the second last round. According to the last round fault model analysis for 4 nibbles of the key $K$, we have $4^4 = 2^8$ possible options. Suppose, we induce faults corresponding to the $j$-th SBox, $S_j$, $j \in \{0,1,2,3\}$ in the beginning of the second last round. Then this induced fault activates 4 SBoxes $S_0, S_8, S_{16}, S_{24}$ at the last round. Then in the last round, 4 nibbles $\mathbf{k_0}, \mathbf{k_8}, \mathbf{k_{16}}$ and $\mathbf{k_{24}}$ of the key $K$ are XOR-ed to the outputs of SBoxes $S_0, S_8, S_{16}$, and $S_{24}$ respectively to generate the 4 nibbles $\mathbf{c_0}, \mathbf{c_8}, \mathbf{c_{16}}$ and $\mathbf{c_{24}}$ of the ciphertext $C$. Here we have not considered the permutation in the last round as SPN structure based ciphers usually don't. For faulty ciphertexts $C^i$, $i \in \{1,2\}$, let $\mathbf{c_0^i}, \mathbf{c_8^i}, \mathbf{c_{16}^i}$ and $\mathbf{c_{24}^i}$ be the 4 nibbles corresponding to the faults $\delta_1$ and $\delta_2$, where $(\delta_1, \delta_2) \in \mathcal{F}$ at the

corresponding positions. Now, consider the following equations:

$$S_j^{-1}(P^{-1}(S_0^{-1}(\mathbf{c_0} \oplus \mathbf{k_0}) \oplus \mathbf{k_0}||S_8^{-1}(\mathbf{c_8} \oplus \mathbf{k_8}) \oplus \mathbf{k_8}||S_{16}^{-1}(\mathbf{c_{16}} \oplus \mathbf{k_{16}}) \oplus \mathbf{k_{16}}||$$
$$S_{24}^{-1}(\mathbf{c_{24}} \oplus \mathbf{k_{24}}) \oplus \mathbf{k_{24}})) \oplus S_j^{-1}(P^{-1}(S_0^{-1}(\mathbf{c_0^1} \oplus \mathbf{k_0}) \oplus \mathbf{k_0}||S_8^{-1}(\mathbf{c_8^1} \oplus \mathbf{k_8}) \oplus \mathbf{k_8}||$$
$$S_{16}^{-1}(\mathbf{c_{16}^1} \oplus \mathbf{k_{16}}) \oplus \mathbf{k_{16}}||S_{24}^{-1}(\mathbf{c_{24}^1} \oplus \mathbf{k_{24}}) \oplus \mathbf{k_{24}})) = \delta_1 \quad (6)$$

$$S_j^{-1}(P^{-1}(S_0^{-1}(\mathbf{c_0} \oplus \mathbf{k_0}) \oplus \mathbf{k_0}||S_8^{-1}(\mathbf{c_8} \oplus \mathbf{k_8}) \oplus \mathbf{k_8}||S_{16}^{-1}(\mathbf{c_{16}} \oplus \mathbf{k_{16}}) \oplus \mathbf{k_{16}}||$$
$$S_{24}^{-1}(\mathbf{c_{24}} \oplus \mathbf{k_{24}}) \oplus \mathbf{k_{24}})) \oplus S_j^{-1}(P^{-1}(S_0^{-1}(\mathbf{c_0^2} \oplus \mathbf{k_0}) \oplus \mathbf{k_0}||S_8^{-1}(\mathbf{c_8^2} \oplus \mathbf{k_8}) \oplus \mathbf{k_8}||$$
$$S_{16}^{-1}(\mathbf{c_{16}^2} \oplus \mathbf{k_{16}}) \oplus \mathbf{k_{16}}||S_{24}^{-1}(\mathbf{c_{24}^2} \oplus \mathbf{k_{24}}) \oplus \mathbf{k_{24}})) = \delta_2 \quad (7)$$

where $\mathbf{k_0} = k_0k_1k_2k_3, \mathbf{k_8} = k_{32}k_{33}k_{34}k_{35}, \mathbf{k_{16}} = k_{64}k_{65}k_{66}k_{67}$ and $\mathbf{k_{24}} = k_{96}k_{97}$ $k_{98}k_{99}$ and $S_j^{-1}$ operates on the 4 bits word after the inverse permutation operation $P^{-1}$ where fault is induced at the beginning of the second last round as shown in the Figure 1.

We use equations (6) and (7) to filter the $2^8$ possible options of $\mathbf{k_0}||\mathbf{k_8}||\mathbf{k_{16}}$ $||\mathbf{k_{24}}$ i.e., the corresponding 16 bits of the round key $K$. Considering these equations for $j = 0, 1, 2$ or 3, we explored all the possible options making an exhaustive search and checked that the possible options for $\mathbf{k_0}||\mathbf{k_8}||\mathbf{k_{16}}||\mathbf{k_{24}}$ reduces to $2^6$. Again for $j \in \{0, 1\}$ from the equations (6) and (7) we get $2^4$ possible options for $\mathbf{k_0}||\mathbf{k_8}||\mathbf{k_{16}}||\mathbf{k_{24}}$. So, we have reduced to the square root of the possible options for 16 bits of round key $K$ than the designers' claim. Further, if we consider equations (6) and (7) for $j \in \{0, 1, 2\}$, then we can reduce the possible options of $\mathbf{k_0}||\mathbf{k_8}||\mathbf{k_{16}}||\mathbf{k_{24}}$ to $2^2$. From the permutation involved in the design of DEFAULT it directly follows that there are 8 groups of 4 SBoxes. Here we have considered one group of 4 SBoxes and the corresponding 4 nibbles of the key $K$. Similarly, we take into account other group of SBoxes in the state beginning of the second last round, where we induce nibble based faults such that we can obtain the remaining part of the key $K$. Therefore, we consider 8 different groups of SBoxes in the beginning of the second last round and perform analysis same as above. Then for each 16 bits part of the key $K$ we have $2^2$ possible options. Therefore, for the whole key $K$ we have $(2^2)^8 = 2^{16}$ possible options that is listed in Table 3 and this search complexity is practical. We have written a Sage code and the source code can be found at `https://github.com/SantanuChennai/Fault`.

As discussed above, if we induce nibble based faults in the state at the beginning of the second last round of the new version of DEFAULT, then it will activate 4 SBoxes in the last round. So, 4 nibbles of the round key $K_2$ are XOR-ed before these active SBoxes and 4 nibbles of the round key $K_3$ are XOR-ed with the outputs of the SBox. If we consider both the nibbles of $K_2$ and $K_3$ then it is difficult to recover all the bits of $K_2$ and $K_3$ with search complexity less the than designers' claim. In our fault attack, we have first recovered the possible key candidates of last round key $K_3$ and then try to filter the key candidates inducing nibble based fault one round before the last round.

For the last round, we have 4 possible options for each nibble of the last round key $K_3$ that we have obtained inducing nibble based faults and analysing each SBox separately. But the problem is that if we consider the last two rounds of the DEFAULT, then both the subkeys $K_2$ and $K_3$ gets involved. Since in the initial version, $K_2 = K_3$ and 4 bit positions of $K_2$ are in the 16 bit positions of $K_3$ the case is quite easier to handle. But in the new version from inverse key scheduling algorithm we have checked that 128 bits of $K_3$ get involved in the algebraic expressions of 4 bits of $K_2$ that are used in the inverse round to verify the induced fault. So, if we know the possible options for the 16 bits of the round key $K_3$, then from that we can not verify the corresponding possible options of 4 bits of the round key $K_2$. Therefore, in this case we can not reduce the search complexity of the secret key than the designers' claim.

| **Round** | **Number of Faults** | **Attack Complexity** |
|---|---|---|
| Last Round | 64 | $2^{64}$ |
| Last two rounds | 80 | $2^{48}$ |
| Last two rounds | 96 | $2^{32}$ |
| Last two rounds | 112 | $2^{16}$ |

Table 3: Comparison of Attack Complexities with changing number of faults for initial version of DEFAULT

# 4 Conclusion

We have analysed the newly designed block cipher DEFAULT in which DEFAULT LAYER is introduced in the cipher to protect the cipher from DFA. We have observed that the initial version of DEFAULT LAYER fails to thwart DFA as a safeguard for the main cipher if we remove last round permutation. In the modified version, designers used key scheduling algorithm and our attack does not work on this version.

# References


[1] A. Baksi, S. Bhasin, J. Breier, M. Khairallah, T. Peyrin, S. Sarkar and S. M. Sim. DEFAULT: Cipher Level Resistance Against Differential Fault Attack. Asiacrypt 2021. Available at `https://eprint.iacr.org/2021/712`

[2] S. Banik, S. K. Pandey, T. Peyrin, Y. Sasaki, S. M. Sim and Y. Todo. GIFT: A Small Present - Towards Reaching the Limit of Lightweight Encryption. CHES 2017.

[3] C. Beierle, G. Leander, A. Moradi and S. Rasoolzadeh. CRAFT: Lightweight Tweakable Block Cipher with Efficient Protection Against DFA Attacks. IACR Trans. Symmetric Cryptol. 2019

[4] E. Biham and A. Shamir. Differential Fault Analysis of Secret Key Cryptosystems. CRYPTO 1997.

[5] D. Boneh, R. A. DeMillo and R. J. Lipton. On the Importance of Checking Cryptographic Protocols for Faults. EUROCRYPT 1997.

[6] G. Piret and J.-J. Quisquater. A differential fault attack technique against SPN structure, with application to the AES and KHAZAD. CHES 2003.

[7] D. Mukhopadhyay. An improved fault based attack of the advanced encryption standard. AFRICACRYPT 2009.

[8] T. Simon, L. Batina, J. Daemen, V. Grosso, P.M.C. Massolino, K. Papagiannopoulos, F. Regazzoni and N. Samwel. Friet: An authenticated encryption scheme with built-in fault detection. EUROCRYPT 2020