On the Optimality of Differential Fault Analyses on CLEFIA

Juliane Krämer, Anke Stüber, Ágnes Kiss
Security in Telecommunications
Technische Universität Berlin
Germany
Email: {juliane,anke,agnes}@sec.t-labs.tu-berlin.de

Abstract—**Differential Fault Analysis is a powerful cryptanalytic tool to reveal secret keys of cryptographic algorithms. By corrupting the computation of an algorithm, an attacker gets additional information about the secret key. In 2012, several Differential Fault Analyses on the AES cipher were analyzed from an information-theoretic perspective. This analysis exposed whether or not the leaked information was fully exploited. It revealed if an analysis was already optimal or if it could still be improved. We applied the same approach to all existing Differential Fault Analyses on the CLEFIA cipher. We show that only some of these attacks are already optimal. We improve those analyses which did not exploit all information. With one exception, all attacks against CLEFIA-128 reach the theoretical limit after our improvement. Our improvement of an attack against CLEFIA-192 and CLEFIA-256 reduces the number of fault injections to the lowest possible number reached to date.**

Keywords: CLEFIA, Differential Fault Analysis, Fault Attack

I. Introduction

The security of cryptographic algorithms does not only rely on their mathematical correctness. The implementation of an algorithm is also analyzed and even attacked to break cryptographic systems. An attack which actively alters the computation of an algorithm by inducing software or hardware faults is called fault attack. A Differential Fault Analysis (DFA) is a specific form of a fault attack. After inducing a fault into one or several computations of a cryptographic algorithm, the secret key of this algorithm is revealed by analyzing the difference between correct and faulty results of the computation. For symmetric algorithms, Differential Fault Analyses were first described in 1997 [6]. Since then, they were successfully applied to various symmetric ciphers and their key schedule, e.g., DES [6], [14], AES [1], [12], and CLEFIA. They were also applied to other cryptographic algorithms, such as stream ciphers [4], [11], and hash functions [10].

In 2012, Sakiyama et al. analyzed the optimality of several Differential Fault Analyses on the Advanced Encryption Standard (AES) [15]. They developed a model which quantifies the amount of information a certain fault can deliver. Then, they analyzed attacks from seven publications from an information-theoretic perspective. The information-theoretic optimality does not imply that such an attack is also optimal from other points of view, e.g., a non-optimal attack might be easier to conduct in practice. However, contrary to an optimal attack, an attack which is not optimal can still be improved in the given framework, i.e., the key space can be further reduced or a key space with the same size can be determined with less fault injections. Sakiyama et al. found that some of the attacks were not optimal, and they proposed improved attacks for AES-192 and AES-256 which reach the theoretical limits.

We applied the approach from Sakiyama et al. to analyze the information-theoretic optimality of Differential Fault Analyses on CLEFIA. The CLEFIA cipher is a 128-bit block cipher which was proposed by Sony Corporation in 2007 [17]. Since then, several analyses on the cipher and attacks against it have been published, including side channel attacks which exploit cache accesses [13] and Impossible Differential Attacks [21].

For this paper, we analyzed six published Differential Fault Analyses on CLEFIA [2], [3], [8], [19], [20], [22]. To the best of our knowledge, these are all DFAs on CLEFIA to date. They cover attacks against CLEFIA with all possible key lengths, which are 128, 192, and 256 bits. As it was the case for the AES analysis, we identified some optimal attacks, while others did not exploit all available information. With one exception, we optimized all attacks against CLEFIA-128 which were shown not to be optimal, so that they now reach the theoretical limits. We also considerably improved an attack against CLEFIA with longer keys. The physical realization of fault attacks [5], as well as countermeasures to prevent them, are not relevant for our analysis and are thus out of scope of this work.

Our Contribution: We analyzed all published Differential Fault Analyses on CLEFIA from an information-theoretic perspective with the techniques introduced in [15]. Our results show that some of these attacks are optimal, while others do not exploit all available information. With one exception, we optimized all attacks against CLEFIA-128 which proved not to be optimal. The optimized attacks reach the theoretical limits and thus exploit all available information. For longer keys, all DFAs were shown not to be optimal in our analysis. We considerably improved one of them. The improved attack is the best known attack against CLEFIA-192 and CLEFIA-256 to date. We validated our theoretical findings by simulating the DFAs with software implementations using the original CLEFIA code [18]. In these implementations, we reached the results as predicted.

Organization: The rest of this work is structured as follows: in Section II, we explain the theory of Differential Fault Analyses and present the CLEFIA cipher. We also give some background information on information theory. In Section III, we describe the DFAs on CLEFIA that we analyzed for this work. The methodology and the results of our analysis are described in Section IV. In Section V, we validate our analytical results. We explain how we optimized the non-optimal attacks against CLEFIA-128 and describe our improvement for CLEFIA-192 and CLEFIA-256. Finally, we conclude in Section VI.

II. Background

We first explain Differential Fault Analysis. Then, we present the CLEFIA cipher and provide background knowledge on information theory.

A. Differential Fault Analysis

The first fault attack against a cryptographic algorithm was presented in 1997, when Boneh, DeMillo and Lipton published an attack against the RSA signature scheme [7]. They showed that the RSA modulus can be factorized if an attacker induces a fault into the computation of one of the two parts of the signature generation in case the RSA CRT version is used. With a single faulty signature and a correct signature under the same secret key, an attacker can thereby reveal the secret key.

Inspired by this so-called Bellcore attack, still in 1997, Biham and Shamir described the idea of these kinds of attacks against symmetric cryptographic algorithms [6]. Such an attack is called Differential Fault Analysis (DFA), or Differential Fault Attack [12]. For such an attack, an attacker needs at least one correct ciphertext and one faulty ciphertext. Thus, she has to have the ability to induce faults on the cryptographic primitive level. These faults can be described in detailed fault models, which include the location and the timing of the fault, and the number of bits and bytes which are affected by the fault. A fault can, for example, affect one byte in the register storing the first four bytes of the state (location) in the penultimate round (timing). The assumed fault model gives the attacker partial information about the difference between certain states of the correct and the faulty computations, although she will not know the concrete value of the fault in most scenarios. Since the attacker also knows the correct and faulty ciphertext, and thereby their difference, she can deduce information about the secret key. Small differences in the fault models might crucially affect the capabilities and the complexity of the attacks [6]. For the attacks analyzed in this work, the attacker is assumed to have full control on the timing and the location of the fault, and to be able to induce not permanent, but transient faults.

B. CLEFIA

The 128-bit block cipher CLEFIA was developed by Sony Corporation and presented in 2007 [17][1]. To be compatible with AES, CLEFIA supports key lengths of 128, 192, and 256 bits. Contrary to AES, however, CLEFIA is a Feistel cipher with four 32-bit data lines which are used during r rounds throughout the encryption and decryption process. Depending on the key lengths, the number of rounds is 18, 22, and 26, respectively. According to the four data lines, $P_i \in \{0,1\}^{32}$, $i \in \{0,\dots,3\}$ denote the four 32-bit parts of the plaintext P, so that $P = P_0|P_1|P_2|P_3$. Similarly, the state is denoted by $T = T_0|T_1|T_2|T_3$ and the ciphertext by $C = C_0|C_1|C_2|C_3$.

As shown in Figure 1, CLEFIA uses $2r$ round keys during the encryption. In the k^{th} round, RK_{2k} and RK_{2k+1} are used for $k \in \{0,\dots,r-1\}$. Moreover, four whitening keys are used, from which WK_0 and WK_1 are XORed with P_1 and P_3 at the beginning of the encryption process, while WK_2 and WK_3 are XORed to the final T_1 and T_3 as last step of the encryption. The 4-branch, r-round generalized Feistel structure, that the encryption algorithm uses after the initial and before the final key whitening phases, is usually referred to as $GFN_{4,r}$. Its inverse function,

[1]All figures in this section are taken from [17].

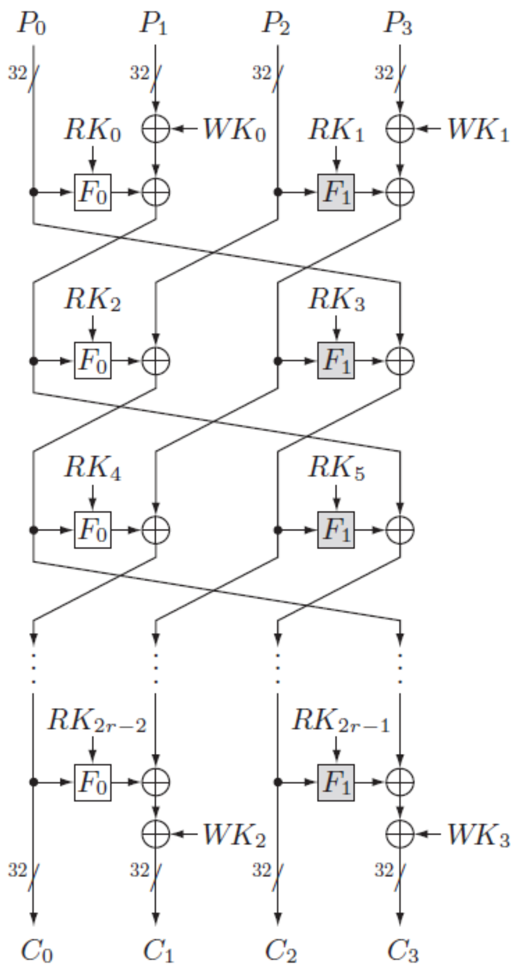

Figure 1: CLEFIA Encryption algorithm

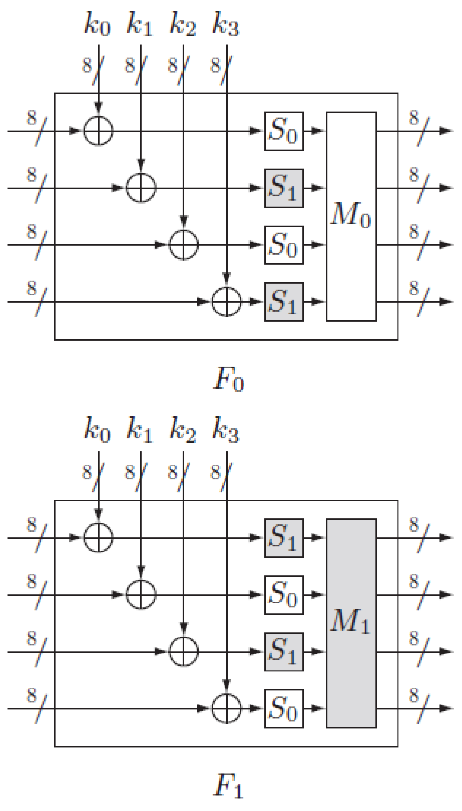

Figure 2: F-functions F_0 and F_1

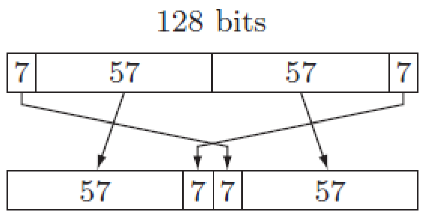

Figure 3: DoubleSwap function Σ

$GFN_{4,r}^{-1}$, is used during decryption, where the order of the round keys, the whitening keys and the direction of the Feistel mechanism are reversed.

In every round, two 32-bit F-functions F_0 and F_1 are used. They are shown in Figure 2. Both these F-functions first XOR the input with the round key and then make use of the 8-bit S-boxes S_0 and S_1. Afterwards, F_0 and F_1 contain a diffusion layer provided by the corresponding diffusion matrices M_0 and M_1. Here, the transpose of the state T is split into 8-bit vectors which are multiplied with the respective matrix in $GF(2^8)$. These matrices are self-invertible, i.e., their inverses are themselves.

$$M_0 = \begin{pmatrix} 1 & 2 & 4 & 6 \\ 2 & 1 & 6 & 4 \\ 4 & 6 & 1 & 2 \\ 6 & 4 & 2 & 1 \end{pmatrix} \quad M_1 = \begin{pmatrix} 1 & 8 & 2 & 10 \\ 8 & 1 & 10 & 2 \\ 2 & 10 & 1 & 8 \\ 10 & 2 & 8 & 1 \end{pmatrix}.$$

The whitening keys WK_i, $i \in \{0,\dots,3\}$ and the round keys RK_i, $i \in \{0,\dots,2r-1\}$ are calculated from the initial secret key K during a key schedule procedure, cf. Algorithm 1. The key scheduling algorithm for CLEFIA-128 first generates a 128-bit intermediate key L. For this process, the original secret key K and twenty-four 32-bit constant values $\mathrm{CON}_{i(32)}$, $i \in \{0,\dots,23\}$, are used as input to the 4-branch Feistel structure $GFN_{4,12}$, which is used through 12 rounds. After that, for the generation of the round keys and the whitening keys, the intermediate key L is used together with thirty-six additional 32-bit constants $\mathrm{CON}_{i(32)}$, $i \in \{24,\dots,59\}$, and the original secret key K, cf. Algorithm 1. The so-called DoubleSwap function $\Sigma : \{0,1\}^{128} \to \{0,1\}^{128}$ is shown in Figure 3.

Algorithm 1 Key-Schedule for CLEFIA-128

```
$L \leftarrow GFN_{4,12}(\mathrm{CON}^{128}_{0(32)},\ldots,\mathrm{CON}^{128}_{23(32)},K_0,\ldots,K_3)$
$WK_0|WK_1|WK_2|WK_3 \leftarrow K$
**for** $i = 0$ **to** $8$ **do**
    $T \leftarrow L \oplus (\mathrm{CON}^{128}_{24+4i(32)}|\mathrm{CON}^{128}_{24+4i+1(32)}|$
                $\mathrm{CON}^{128}_{24+4i+2(32)}|\mathrm{CON}^{128}_{24+4i+3(32)})$
    $L \leftarrow \Sigma(L)$
    **if** $i$ is odd **then**
        $T \leftarrow T \oplus K$
    **end if**
    $RK_{4i}|RK_{4i+1}|RK_{4i+2}|RK_{4i+3} \leftarrow T$
**end for**
```

For CLEFIA-192 and CLEFIA-256, two 128-bit values K_L and K_R are generated from K and by means of these, intermediate keys L_L and L_R are calculated using fourty 32-bit constant values and $GFN_{8,10}$, the 8-branch Feistel structure with 10 rounds. Then, the round keys and whitening keys are generated from L_L, L_R, K_L and K_R by means of another fourty-four 32-bit constants. The key scheduling process of CLEFIA-192 and CLEFIA-256 is described in detail in [17].

C. Information Theory

The foundations of information theory have been laid by Claude E. Shannon in 1948 [16]. The Shannon entropy for a discrete random variable X with $p(X = x_i) = p_i$, $i = 1,\ldots,n$ and $\{x_1,\ldots,x_n\} \subsetneq \{0,1\}^*$, is

$$H(X) := -\sum_{i=1}^{n} p_i \log_2(p_i). \tag{1}$$

It quantifies the uncertainty when the value of the random variable X is to be predicted.

For two discrete random variables X and Y, their joint entropy can be defined as well as the respective conditional entropies. The joint entropy of X and Y, $H(XY)$ or $H(X,Y)$, is

$$H(XY) := -\sum_{i=1}^{n}\sum_{j=1}^{m} p(X = x_i, Y = y_j)\log_2(p(X = x_i, Y = y_j)). \tag{2}$$

We have $H(XY) \leq H(X) + H(Y)$, while for stochastically independent X and Y, equality holds.

Following the definition by Shannon, the conditional entropy of X is the average of the entropy of X for each value of Y, weighted according to the probability of getting that particular Y [16]. Thus, the entropy of X conditioned on Y, i.e., the conditional entropy $H(X \mid Y)$, is defined as

$$H(X \mid Y) := \sum_{j=1}^{m} p(Y = y_j) H(X \mid Y = y_j), \tag{3}$$

with

$$H(X \mid Y = y_j) := -\sum_{i=1}^{n} p(X = x_i | Y = y_j)\log_2(p(X = x_i | Y = y_j)). \tag{4}$$

Using the definition of the conditional probability [9], Equation 3 can be transformed to

$$H(X \mid Y) = H(XY) - H(Y). \tag{5}$$

III. Differential Fault Analyses on CLEFIA

For this work, we analyzed six Differential Fault Analyses on CLEFIA. They were published between 2007 and 2013. They are, to the best of our knowledge, all published DFAs on CLEFIA.

The first DFA was presented already in 2007 [8]. The attack against CLEFIA-128 uses independent random byte faults at six different positions of the algorithm. These are induced in T_0 and T_2 in rounds 15, 16 and 17 and help to reveal RK_{30}, RK_{31}, $RK_{32} \oplus WK_3$, $RK_{33} \oplus WK_2$, RK_{34} and RK_{35}. Thus, the attack needs at least 6 faulty encryptions. However, the authors state that the fault inductions have to be repeated until all bytes are recovered. They had to induce at least 18 faults in their simulations. Based on the recovered round keys, the original secret key can be revealed by analyzing the key scheduling algorithm. If the key size is 192 or 256, the same procedure has to be applied in rounds $r - 9$ to $r - 1$. Here, the simulated attacks needed 54 faults to be successful.

One year later, these results were improved [19]. Takahashi and Fukunaga encrypt a random plaintext, which does not have to be known, with the same secret key three times for CLEFIA-128. They insert four-byte faults in the 16th round in two of these encryptions, one into F_0 and one into F_1. The authors use the fact that „a fault corrupts the intermediate values of the fault-injection round and the subsequent rounds“. Thus, they obtain more information out of a single fault, since they also analyze how the differences propagate through the next two rounds. After analyzing rounds 16 to 18, 2^{19} candidates for the round keys are left. By applying the inverse of the DoubleSwap function and $GFN^{-1}_{4,12}$ to all round key candidates, the 128-bit secret key can be uniquely identified. In 2010, the authors adapted the same attack to keys with 192 and 256 bits, where 10.78 faults are needed on average [20].

Still in 2010, multiple-byte Differential Fault Analyses were described for CLEFIA-128 [22]. The authors propose three attack models, including the first attack which exploits fault injections in the final round. This attack exploits faults in the inputs of F_0 and F_1 in rounds 18, 17, and 16. The authors consider multiple-byte faults, so that each single fault can affect up to four bytes, i.e., 32 bits. The second attack builds on [8] and induces faults into the inputs of F_0 and F_1 in both the 15th and 17th round, but extends their fault model to multiple bytes. The third attack also builds upon a previous one. For this attack, which targets the 16th round, however, the strict four-byte fault model of [19] has been loosened to a one-to-four-byte model, so that the attack presented in [19] can be seen as a special case of this more general attack description. According to the authors, the minimum amount of faulty ciphertexts for these three attacks to be successful is 5 to 6, 6 to 8 and 2. These faults help to reveal the secret key completely for the first two attacks, while in the third attack, 2^{19} candidates for the secret key remain. Unfortunately, the description of these three attacks is not always easy to follow, and several statements in [22] are inconsistent with one another. The description of these attacks in the work at hand is to the best of our knowledge.

Since all previous attacks targeted the last four rounds of the algorithm, in 2012 it was analyzed if protecting the last four rounds of CLEFIA-128 would counter Differential Fault Analyses [2]. The authors show that it is sufficient to induce two random byte-faults in the computation of F_0 and F_1 in round 14. It is assumed that the faults are induced before the diffusion operation of the F-functions, cf. Figure 3 in [2]. These two faults are enough to uniquely reveal the secret key by exploiting the propagation of the faults in the final four rounds. Thus, it is not sufficient to protect the last four rounds of CLEFIA against such attacks.

One year later, the same authors presented attacks on CLEFIA with all possible key lengths [3]. Here, they also start to scrutinize how much information a certain fault can provide. Regarding the presented attacks, they first describe their attack against CLEFIA-128 [2]. Afterwards, they describe how to uniquely reveal the secret key in case of CLEFIA-192 and CLEFIA-256. Here, they induce two random faults each in the computation of F_0 and F_1, in rounds $r-4$ and $r-8$. Thus, they induce eight faults altogether and their attack can not be prevented by protecting only the last four rounds. They can reveal the whole secret key, no matter if it has 192 or 256 bits.

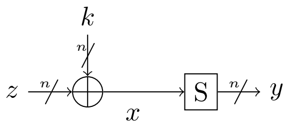

Figure 4: General cipher model using an n-bit S-box S.

IV. Information-Theoretic Analysis of DFAs on CLEFIA

In this section, we first explain the methodology we used to analyze the optimality of Differential Fault Analyses on CLEFIA. The approach is based on [15] and was adapted to the CLEFIA algorithm. Afterwards, we evaluate the optimality of all published DFAs against CLEFIA from an information-theoretic perspective.

A. General Methodology

Similar to [15], we utilize a simple general cipher model using an n-bit S-box in order to explain the idea of estimating the amount of information that is leaked from the injection of a fault. Let S be a bijective n-bit S-box, x the n-bit input and y the n-bit output of S. Let z be the n-bit value that the n-bit key k is added to and $x = z \oplus k$. Figure 4 shows the model with its inputs and outputs. Let $k, x_1, x_2, y_1, y_2, z_1, z_2 \in \{0,1\}^n$ be the values that occur in two executions of the encryption algorithm under the same key and $\Delta x, \Delta y, \Delta z \in \{0,1\}^n$ the respective differences. We assume that the attacker can observe the values y_1 and y_2 and calculate $\Delta y, x_1, x_2, \Delta x$ and Δz from them and from her knowledge of S. However, z_1 and z_2 can not be derived from those values. Furthermore, let $K, X_1, X_2, Y_1, Y_2, \Delta X, \Delta Y$ be discrete random variables with possible values of $k, x_1, x_2, y_1, y_2, \Delta x, \Delta y \in \{0,1\}^n$.

Due to the bijectivity of the S-box S, there is a one-to-one correspondence between (x_1, x_2, k) and (y_1, y_2, k) and it follows that

$$H\left(X_1 X_2 K\right) = H\left(Y_1 Y_2 K\right). \tag{6}$$

Furthermore, ΔY is uniquely determined from X_1 and X_2, since

$$\Delta y = \mathrm{S}\left(x_1\right) \oplus \mathrm{S}\left(x_2\right). \tag{7}$$

As K is stochastically independent of X_1 and X_2, we have

$$\begin{aligned} & H\left(X_1 X_2 K\right) \\ & \overset{(2)}{=} H\left(X_1 X_2\right) + H(K) \\ & \overset{(7)}{=} H\left(X_1 X_2 \Delta Y\right) + H(K) \\ & \overset{(5)}{=} H\left(X_1 X_2 \mid \Delta Y\right) + H(\Delta Y) + H(K). \end{aligned} \tag{8}$$

ΔY is uniquely determined from Y_1 and Y_2, and Y_2 is uniquely determined from Y_1 and ΔY. It follows that

$$\begin{aligned} & H(Y_1Y_2K) \\ &= H(KY_1Y_2) \\ &\overset{(5)}{=} H(K \mid Y_1Y_2) + H(Y_1Y_2) \\ &= H(K \mid Y_1Y_2) + H(Y_1\Delta Y) \\ &\overset{(5)}{=} H(K \mid Y_1Y_2) + H(Y_1 \mid \Delta Y) + H(\Delta Y). \end{aligned} \tag{9}$$

Now we estimate $H(Y_1 \mid \Delta Y)$. We have

$$H(Y_1 \mid \Delta Y) \le \log_2(2^n) = n = H(K). \tag{10}$$

Furthermore, X_1 and Y_1 are uniquely determined from each other, since

$$y_1 = \mathrm{S}(x_1). \tag{11}$$

As K can be calculated from Z_1 and $K \oplus Z_1$, and K is stochastically independent of Z_1 and ΔY, we have

$$\begin{aligned} 0 \le\ & H(Y_1 \mid \Delta Y) - H(Y_1 \mid Z_1\Delta Y) \\ &\overset{(11)}{=} H(Y_1 \mid \Delta Y) - H(X_1 \mid Z_1\Delta Y) \\ &= H(Y_1 \mid \Delta Y) - H(K \oplus Z_1 \mid Z_1\Delta Y) \\ &= H(Y_1 \mid \Delta Y) - H(K \mid Z_1\Delta Y) \\ &= H(Y_1 \mid \Delta Y) - H(K). \end{aligned} \tag{12}$$

From Equations 10 and 12 it follows that

$$H(Y_1 \mid \Delta Y) = H(K), \tag{13}$$

and therefore

$$\begin{aligned} & H(X_1X_2 \mid \Delta Y) \\ &\overset{(8)}{=} H(X_1X_2K) - H(\Delta Y) - H(K) \\ &\overset{(6),(13)}{=} H(Y_1Y_2K) - H(\Delta Y) - H(Y_1 \mid \Delta Y) \\ &\overset{(9)}{=} H(K \mid Y_1Y_2). \end{aligned} \tag{14}$$

ΔX is uniquely determined from X_1 and X_2, and X_2 is uniquely determined from X_1 and ΔX. Thus, we have

$$\begin{aligned} & H(K \mid Y_1Y_2) \\ &\overset{(14)}{=} H(X_1X_2 \mid \Delta Y) \\ &= H(X_1\Delta X \mid \Delta Y) \\ &\overset{(5)}{=} H(X_1 \mid \Delta X\Delta Y) + H(\Delta X \mid \Delta Y). \end{aligned} \tag{15}$$

The term $H(X_1 \mid \Delta X\Delta Y)$ depends on the differential property of the S-box S and is not affected by the type of the attack and its underlying fault model. On the other hand, the term $H(\Delta X \mid \Delta Y)$ depends on the difference ΔX. If the adversary has information on ΔX from the fault model of the attack, she can reduce $H(\Delta X \mid \Delta Y)$.

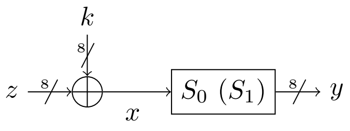

Figure 5: Simple cipher model using an 8-bit CLEFIA S-box S_0 (S_1).

B. General Methodology Adapted to CLEFIA

Now we refine the described general model by setting $n = 8$ and plugging in the S-boxes S_0 and S_1 of the cipher CLEFIA in place of the S-box S. The refined model is shown in Figure 5. It shows the XORing of the round key and the application of the S-boxes S_0 and S_1 within the F-functions F_0 and F_1 of the cipher CLEFIA, as shown in Figure 2.

To analyze the values of $H(X_1 \mid \Delta X\Delta Y)$ and $H(\Delta X \mid \Delta Y)$, we take a look at the number of possible solutions for x_1 that satisfy

$$\Delta y = S_i(x_1) \oplus S_i(x_1 \oplus \Delta x) \tag{16}$$

for $S_i \in \{S_0, S_1\}$. The number of possible solutions to Equation 16 for x_1 when Δx and Δy are given can be derived from the values of Table 1 in [19]: we divide the values by 256 since we do not regard the values for the 8-bit part k of the round key but solve the equation for x_1 instead. Furthermore, we discard the values for $\Delta x, \Delta y = 0$ since we assume them to be nonzero. Hence, we discard 510 occurrences of 0 and one occurrence of 256 possible values for x_1. For the S-box S_0, the amount of occurrences of 0 to 10 possible solutions can be found in Table I. For the S-box S_1, for every Δy we obtain 0 for 128 values of Δx, 2 for 126 values of Δx and 4 for one value of Δx. Note that the numbers for S_1 equal the numbers for the AES S-box analyzed in [15].

Solutions for x_1	Occurrences
0	39511
2	19501
4	5037
6	848
8	119
10	9
Total	65025 ($= 255^2$)

Table I: Solutions to Equation 16 for S_0 and x_1 with fixed Δx and Δy.

Now we calculate $H(K \mid Y_1Y_2)$ for the case that we have no information on the fault model, i.e., $X_1, \Delta X$, and K are independent and identically distributed. Then, we calculate $H(K \mid Y_1Y_2)$ again with additional information on the fault model.

For S_0, we have

$$\begin{aligned}
&H_{S_0}(\Delta X \mid \Delta Y)\\
&= \sum_{i=1}^{255} P(\Delta Y = \Delta y_i)\, H_{S_0}(\Delta X \mid \Delta Y = \Delta y_i)\\
&= \frac{1}{255}\cdot\Bigg(-19501\cdot\frac{2}{256}\cdot\log_2\left(\frac{2}{256}\right)\\
&\quad - 5037\cdot\frac{4}{256}\cdot\log_2\left(\frac{4}{256}\right) - 848\cdot\frac{6}{256}\cdot\log_2\left(\frac{6}{256}\right)\\
&\quad - 119\cdot\frac{8}{256}\cdot\log_2\left(\frac{8}{256}\right) - 9\cdot\frac{10}{256}\cdot\log_2\left(\frac{10}{256}\right)\Bigg)\\
&\approx 6.535,
\end{aligned} \tag{17}$$

$$\begin{aligned}
&H_{S_0}(X_1 \mid \Delta X \Delta Y)\\
&= \sum_{i=1}^{255} P(\Delta Y = \Delta y_i)\, H_{S_0}(X_1 \mid \Delta X \Delta Y = \Delta y_i)\\
&= \frac{1}{255}\cdot\Bigg(19501\cdot\frac{2}{256}\cdot 1 + 5037\cdot\frac{4}{256}\cdot 2\\
&\quad + 848\cdot\frac{6}{256}\cdot\log_2(6) + 119\cdot\frac{8}{256}\cdot 3\\
&\quad + 9\cdot\frac{10}{256}\cdot\log_2(10)\Bigg)\\
&\approx 1.465,
\end{aligned} \tag{18}$$

$$H_{S_0}(K \mid Y_1 Y_2) \overset{(9)}{=} 6.535 + 1.465 = 8 = H(K). \tag{19}$$

For S_1, since its differential property is equal to the one of the AES S-box, the calculations are analogous to the calculations 5 and 6 from [15] and are thus omitted in the present work. We have

$$H_{S_1}(\Delta X \mid \Delta Y) = \frac{447}{64}, \tag{20}$$

$$H_{S_1}(X_1 \mid \Delta X \Delta Y) = \frac{65}{64}, \tag{21}$$

$$H_{S_1}(K \mid Y_1 Y_2) \overset{(9)}{=} \frac{447}{64} + \frac{65}{64} = 8 = H(K). \tag{22}$$

Since $H_{S_0}(K \mid Y_1 Y_2) = H_{S_1}(K \mid Y_1 Y_2) = H(K)$, no information on K can be obtained without information on the fault model.

We repeat the calculation using some assumptions on the fault model. Let $\mathcal{X} \subseteq \{0,1\}^n$ be the set of values that ΔX can take in the employed fault model.

As a coarse estimate for $H_{S_0}(\Delta X \Delta Y)$, we consider only the values for Δx and Δy that allow at least one possible solution for x_1, i.e., $19501 + 5037 + 848 + 119 + 9 = 25514$ of $255^2 = 65025$ values. Let X_1 and ΔX be independent and identically distributed over $\{0,1\}^n$ and $\mathcal{X}$, and ΔY identically distributed over $\{0,1\}^n$. We have

$$\begin{aligned}
H_{S_0}(\Delta X \Delta Y) &\approx \log_2\left(\frac{25514\cdot|\mathcal{X}|\cdot 2^n}{65025}\right),\\
H_{S_0}(\Delta Y) &\approx n,
\end{aligned} \tag{23}$$

and therefore

$$H_{S_0}(\Delta X \mid \Delta Y) \overset{(23)}{\approx} \log_2(|\mathcal{X}|) - 1.349. \tag{24}$$

Furthermore, we have

$$\begin{aligned}
&H_{S_0}(X_1 \mid \Delta X \Delta Y)\\
&= \sum_{\substack{\Delta x \in \mathcal{X}\\ \Delta y \in \{0,1\}^n}} \Big[P(\Delta X = \Delta x \mid \Delta Y = \Delta y)\cdot\\
&\quad H_{S_0}(X_1 \mid \Delta X = \Delta x \;\; \Delta Y = \Delta y)\Big]\\
&= \frac{19501\cdot 2^n}{65280}\cdot\frac{2}{2^n}\cdot 1 + \frac{5037\cdot 2^n}{65280}\cdot\frac{4}{2^n}\cdot 2\\
&\quad + \frac{848\cdot 2^n}{65280}\cdot\frac{6}{2^n}\cdot\log_2(6) + \frac{119\cdot 2^n}{65280}\cdot\frac{8}{2^n}\cdot 3\\
&\quad + \frac{9\cdot 2^n}{65280}\cdot\frac{10}{2^n}\cdot\log_2(10)\\
&\approx 1.465,
\end{aligned} \tag{25}$$

so from Equation 9 we find

$$H_{S_0}(K \mid Y_1 Y_2) \overset{(9)}{\approx} \log_2(|\mathcal{X}|) + 0.115 \tag{26}$$

as an upper bound for $H_{S_0}(K \mid Y_1 Y_2)$. Without information on the fault model we have $\mathcal{X} = \{0,1\}^n$, so maximally $H_{S_0}(\Delta X \mid \Delta Y) \approx n - 1.349$. Hence, we define

$$\begin{aligned}
m_{S_0} &:= (n - 1.349) - H_{S_0}(\Delta X \mid \Delta Y)\\
&\overset{(24)}{\approx} n - \log_2(|\mathcal{X}|)
\end{aligned} \tag{27}$$

as the amount of information leaked from a fault injected before the application of the S-box S_0. Thus, the amount of information a certain fault can yield from an information-theoretic perspective depends on the amount of values the fault can attain. For S_1, again the calculation is analogous to the one from [15] and we use the definition

$$\begin{aligned}
m_{S_1} &:= (n - 1) - H_{S_1}(\Delta X \mid \Delta Y)\\
&\approx n - \log_2(|\mathcal{X}|)
\end{aligned} \tag{28}$$

for the amount of information leaked from a fault injected before the application of the S-box S_1. Since the estimations for both S_0 and S_1 lead to the same definition for the amount of leaked information, we define

$$m \approx n - \log_2(|\mathcal{X}|). \tag{29}$$

Differential Fault Attack	Location	Timing	m	t	$\lvert\mathcal{K}\rvert$	Optimality
Chen, Wu, Feng [8]	1 random byte	15, 16, 17	118.006	18	1	2 faults suffice
Takahashi, Fukunaga [19], [20]	4 known bytes	16	96.023	2	1	optimal
Zhao, Wang, Gao [22]	4 known bytes	16, 17, 18	96.023	12	1	2 faults suffice
	4 known bytes	15, 17	96.023	8	1	2 faults suffice
	4 known bytes	16	96.023	2	2^{19}	$\lvert\mathcal{K}\rvert$ can be 1
Ali, Mukhopadhyay [2], [3]	1 known byte	14	120.006	2	1	optimal
Proposed improvement on [8]	1 random byte	**15**	118.006	**2**	1	**optimal**
Proposed improvement on [22]	4 known bytes	**16**	96.023	**2**	**1**	**optimal**

Table II: Information-theoretic optimality of DFAs against CLEFIA-128. The fault model is described by the location and the timing of the faults. The number of key bits that can be learned from a single fault is denoted with m, and t denotes the number of faults that the respective authors use to reduce the key space to $|\mathcal{K}|$ candidates.

C. Results of the Information-Theoretic Analysis

Using these formulas which we just derived, we will now present the results of our information-theoretic analysis regarding the optimality of all existing Differential Fault Attacks against CLEFIA. We start with the attacks against CLEFIA-128 and present the results of the attacks against CLEFIA with longer keys afterwards. The results of our analysis are summarized in Table II for CLEFIA-128 and in Table III for CLEFIA-192 and CLEFIA-256.

Attacks against CLEFIA-128

The first Differential Fault Attack against CLEFIA-128 was published in 2007 by Chen et al. [8]. It uses 18 faults that are injected in one random byte of a four-byte register, so we have $2^8 - 1$ possible faults in four possible locations and the size of the set of possible values for ΔX is $|\mathcal{X}| = \left(2^8 - 1\right) \cdot 4$. With the formula for the amount of leaked information derived in the previous section, we have

$$m \overset{(29)}{\approx} 128 - \log_2\left(\left(2^8 - 1\right) \cdot 4\right) \approx 118.006, \tag{30}$$

so in theory, one fault is sufficient to reduce the key space to 2^{10} and two faults leak enough information to uniquely identify the key. Since the attack needs 18 faults to identify the key, it is not optimal from an information-theoretic point of view.

In 2008 Takahashi and Fukunaga published the second Differential Fault Attack against CLEFIA-128 [19]. It is identical to the attack against CLEFIA-128 from their 2010 paper [20], in which they adapt the attack to CLEFIA-192 and CLEFIA-256. Their attack uses two faults that are injected in four bytes with known position to reduce the key space to $2^{19.02}$ in the first step. Then the key is recovered through an exhaustive search utilizing the key schedule, but no plaintexts. From the fault model we have $|\mathcal{X}| = \left(2^8 - 1\right)^4$, so

$$m \overset{(29)}{\approx} 128 - \log_2\left(\left(2^8 - 1\right)^4\right) \approx 96.023 \tag{31}$$

is the amount of information leaked from one fault. Therefore, two faults are needed to uniquely identify the key. As the attack uses only two faults we consider it optimal.

The next three Differential Fault Attacks by Zhao et al. [22] from 2010 are described for multiple-byte faults that affect one-to-four known bytes in the calculation. For these fault models, we have $|\mathcal{X}| = \left(2^8 - 1\right)^i$, $i \in \{1, \ldots, 4\}$, so for the amount of leaked information we get

$$m \overset{(29)}{\approx} 128 - \log_2\left(\left(2^8 - 1\right)^1\right) \approx 120.006, \tag{32}$$

$$m \overset{(29)}{\approx} 128 - \log_2\left(\left(2^8 - 1\right)^2\right) \approx 112.011, \tag{33}$$

$$m \overset{(29)}{\approx} 128 - \log_2\left(\left(2^8 - 1\right)^3\right) \approx 104.017, \tag{34}$$

$$m \overset{(29)}{\approx} 128 - \log_2\left(\left(2^8 - 1\right)^4\right) \approx 96.023 \tag{35}$$

in the cases of one-to-four affected bytes. The authors give results only for the case of four-byte faults. Hence, we included only this case in Table II. The authors state in their Section 6 that their first attack uses six faults that affect eight bytes, but since the faults are injected in four-byte registers and their description of the attack in their Section 3 also mentions one-to-four-byte faults, we assume four-byte faults. In case they really simulated eight-byte faults, these would have been two independent four-byte faults during one computation. Thus, their six faults count as twelve faults in our model, cf. Table II. However, from Equation 35 we find that only two faults are needed to recover the secret key. As the attack needs more faults, it is not optimal from an information-theoretic perspective. The second attack from [22] uses eight faults. Again, from Equation 35 we know that two faults leak enough information to uniquely identify the key, so this attack is not optimal either. The third proposed attack equates to the attack from Takahashi and Fukunaga in the case of four-byte faults. Zhao et al. state that with two faults the attack reduces the key space to 2^{19}. Since in theory two faults are enough to uniquely identify the key, this attack is not optimal.

In 2012 another Differential Fault Attack against CLEFIA-128 was described by Ali and Mukhopadhyay [2]. It is identical to the attack against CLEFIA-128 from their 2013 paper [3], in which they adapt the attack to CLEFIA-192 and CLEFIA-256. The attack works with faults that affect one known byte, so we have $|\mathcal{X}| = 2^8 - 1$. They need two faults to recover the key. Since we have

$$m \overset{(29)}{\approx} 128 - \log_2 \left(2^8 - 1\right) \approx 120.006, \qquad (36)$$

two faults are needed to leak enough information to uniquely identify the key. As the attack succeeds with only two faults, it optimally exploits the information leaked from the faults.

Attacks against CLEFIA-192 and CLEFIA-256

As it was the case for the DFAs against AES-192 and AES-256 in [15], we observed that all previous results against CLEFIA-192 and CLEFIA-256 are not optimal from the information-theoretic perspective.

The first Differential Fault Attack against CLEFIA-192 and CLEFIA-256 by Chen et al. [8] from 2007 uses 54 faults to recover the 192-bit or 256-bit key. The fault model is the same as in their attack against CLEFIA-128: the faults are injected in one random byte in a four-byte register. Thus, we have $2^8 - 1$ possible faults in four possible locations and $|\mathcal{X}| = \left(2^8 - 1\right) \cdot 4$. We have

$$m \overset{(29)}{\approx} 128 - \log_2 \left(\left(2^8 - 1\right) \cdot 4\right) \approx 118.006, \qquad (37)$$

so in theory two and three faults are sufficient to uniquely identify the 192-bit and 256-bit key, respectively. As the attack needs 54 faults, it is not optimal.

In 2010 Takahashi and Fukunaga adapted their Differential Fault Attack against CLEFIA-128 to longer keys [20]. Their attack needs on average 10.78 faults that affect four known bytes, so the fault model gives $|\mathcal{X}| = \left(2^8 - 1\right)^4$. The amount of information leaked from one fault is

$$m \overset{(29)}{\approx} 128 - \log_2 \left(\left(2^8 - 1\right)^4\right) \approx 96.023, \qquad (38)$$

so in theory two and three faults are enough to recover the 192-bit and 256-bit key, respectively. However, since $2 \cdot 96 = 192$, an attack with only two faults will most probably not succeed in revealing a 192-bit key. Nevertheless, since the attack needs 10.78 faults on average, it is not optimal.

The most recent Differential Fault Attack against CLEFIA-192 and CLEFIA-256 was published in 2013 by Ali and Mukhopadhyay [3]. Analogously to their attack against CLEFIA-128, it works with faults that affect one known byte, so we have $|\mathcal{X}| = 2^8 - 1$. The attack needs eight faults to recover the key. We have

$$m \overset{(29)}{\approx} 128 - \log_2 \left(2^8 - 1\right) \approx 120.006, \qquad (39)$$

so again two and three faults leak enough information to uniquely identify the 192-bit and 256-bit key, respectively. Since the attack needs eight faults, it is not optimal, but still the best known DFA against CLEFIA-192 and CLEFIA-256 from an information-theoretic perspective.

V. Improvement of the Non-Optimal DFAs

We will now present improvements of those Differential Fault Analysis methods which were shown not to be optimal and thereby validate the results of Section IV. We will show that with one exception, all previously non-optimal attacks against CLEFIA-128 can be improved to be optimal from an information-theoretic perspective in their own fault models. For CLEFIA-192 and CLEFIA-256, we achieve a considerable improvement in one of the algorithms. The improved version requires significantly less fault injections than before. In Section V-C, we validate our findings by simulating as well the existing as our proposed attacks. The simulation results are presented in Table V and Table VI.

Throughout this section, improving an attack from an information-theoretic perspective means using less faults or reducing the key space. For the non-optimal attacks, we seeked for an improved DFA in the same fault model, i.e, we used only a subset of the faults injected in the original attack in order to reveal the secret key.

Before presenting our improvements, we describe the basic idea of Differential Fault Analysis methods on CLEFIA that is used in order to reveal the round keys and thereafter the secret key K. Analyzing the j^{th} round, the attacker calculates the input Z_k^j of the F-function, where $k \in \{0, 1\}$ denotes whether F_0 or F_1 is considered. It can be calculated by means of the correct ciphertext and some of the round keys of later rounds. The difference ΔZ_k^j of the inputs of the F-function is calculated with a correct and a faulty ciphertext. $\Delta Z_{k,i}^j$ with $i \in \{0, \ldots, 3\}$ denotes as well the input difference of one of the four S-boxes used in F_k. In order to obtain the output differences $\Delta Y_{k,i}^j$ of the S-boxes, the inverse of the corresponding diffusion matrix M_k is applied to the 32-bit output difference of F_k. This way we retrieve the input-output differences for the S-boxes, using which we

Differential Fault Attack	Location	Timing	m	t	$\lvert\mathcal{K}\rvert$	Optimality
Chen, Wu, Feng [8]	1 random byte	$r-9,\ldots,r-1$	118.006	54	1	2/3 faults suffice
Takahashi, Fukunaga [20]	4 known bytes	$r-8, r-5, r-2$	96.023	10.78	1	2/3 faults suffice
Ali, Mukhopadhyay [3]	1 known byte	$r-8, r-4$	120.006	8	1	2/3 faults suffice
Proposed improvement on [8]	1 random byte	$\mathbf{r-7, r-4}$	118.006	**8**	1	2/3 faults suffice

Table III: Information-theoretic optimality of DFAs against CLEFIA-192/256. The fault model is described by the location and the timing of the faults. The number of key bits that can be learned from a single fault is denoted with m, and t denotes the number of faults that the respective authors use to reduce the key space to $|\mathcal{K}|$ candidates.

can deduce differential equations for all 8-bit states:

$$\Delta Y^j_{k,i} = \mathrm{S}(Z^j_{k,i} \oplus RK_{2j-2+k,i} \oplus \Delta Z^j_{k,i}) \oplus \mathrm{S}(Z^j_{k,i} \oplus RK_{2j-2+k,i}) \tag{40}$$

Here, we have $i \in \{0,\ldots,3\}$ and S denotes the S-box used in the certain state in case of F_k, as shown in Figure 2. The difference distribution table of an S-box stores all the values of $Z^j_{k,i} \oplus RK_{2j-2+k,i}$ corresponding to a choice $(\Delta Z^j_{k,i}, \Delta Y^j_{k,i})$. Table I shows the possible numbers of solutions for $Z^j_{k,i} \oplus RK_{2j-2+k,i}$ in case of S_0, and we described the case of S_1 in Section IV. If j is odd, the corresponding whitening key according to the Feistel structure is also XORed to this value. Therefore, after using the difference distribution tables, a limited number of candidates remains for each of the four 8-bit parts of the round key in case the input-output differences are nonzero values, because the fault affected the round. After recovering the necessary round keys, the original secret key K can be deduced by analyzing the key scheduling of CLEFIA.

A. Improvements on CLEFIA-128

Two of the analyzed six attacks against CLEFIA-128 are already optimal, and we improved three of the remaining four. In Table II we see the attacks already existing along with our proposed improvements. Since the two attacks of [22] that we improved are improved with the same technique, they are subsumed in Table II

For the deduction of the secret 128-bit key K, in case of CLEFIA-128, the most efficient algorithm uses the values of RK_{30}, RK_{31}, $RK_{32} \oplus WK_3$, $RK_{33} \oplus WK_2$, RK_{34} and RK_{35}. Thus, we need to recover these, by examining the input-output differences in the last three rounds.

1) Optimization of the Attack by Chen, Wu and Feng: The fault model used by Chen et al. is the byte-oriented model of random faults [8]. A one-byte fault is induced into the register composed of four bytes in an intermediate step. The attacker knows the register into which the fault is injected, but does not have any knowledge of the concrete location or the value of the fault. Each fault is

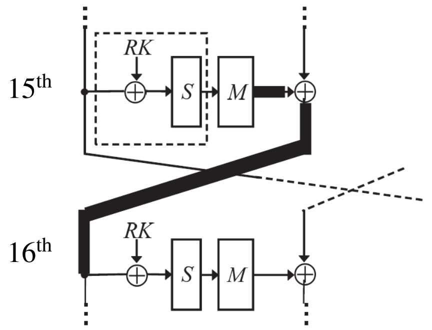

Figure 6: Fault injection and propagation area in the attack against CLEFIA-128 from [19], [20].[2]

injected before a diffusion matrix in a certain round, so that a single random byte fault causes four-byte faults in the next round. In their original attack, they inject three faults into each of six locations in the 15th, 16th, and 17th round.

To make this attack optimal, we will show that only two of the six faults induced in the 15th round are enough to uniquely reveal the secret key. In this fault model, the analysis presented by Takahashi and Fukunaga in Section 6 of their paper [19] can be borrowed. Takahashi and Fukunaga claim that in their attack, the fault injection area can be chosen from two possible areas. One is the area in the 15th round within the dashed rectangle in Figure 6. Here, any bit in any byte can be corrupted. The other area is a total of four bytes in the region after the diffusion matrix of round 15, denoted by a bold line in Figure 6. Afterwards, they only use the fact that on the bold line of the 16th round, all four bytes are corrupted due to the fault propagation. Thus, during the information-theoretic analysis, we examined the latter injection area. The first injection area is the same as the one used by Chen et al. in [8], which means that if we use the two random byte faults injected into the 15th round, we can borrow the technique of Takahashi and Fukunaga shown in Algorithm 2.

As shown in Figure 6 in [19], the fault propagates through the last three rounds. It has an effect on the

[2]Figure taken from [19].

input-output differences starting from round 16. Thus, an attacker can recover a limited number of round key candidates for the round keys. By means of each combination of round key candidates, a possible secret key can be calculated. Then, among these $2^{19.02}$ candidates, the original secret key is verified uniquely through a verification process.

Algorithm 2 Algorithm of Takahashi and Fukunaga

```
**Input**: $C, \overline{C}, \overline{\overline{C}}$
**Output**: $K$
obtain $\{RK_{35}, RK_{34}\}$
**for** candidates in $\{RK_{35}, RK_{34}\}$ **do**
    obtain $\{RK_{33} \oplus WK_2, RK_{32} \oplus WK_3\}$
    **for** candidates in $\{RK_{33} \oplus WK_2, RK_{32} \oplus WK_3\}$ **do**
        obtain $\{RK_{30}, RK_{31}\}$
        calculate $K$
        **if** $((WK_2|WK_3) = (K_2|K_3))$ **then**
            return $K$
        **end if**
    **end for**
**end for**
```

2) Optimization of the Attacks by Zhao, Wang and Gao: Zhao et al. use a different fault model in [22], exploiting multiple-byte faults. They inject one to four random byte-faults into an intermediate register. The attacker does not have any knowledge of the concrete location or the value of the faults. With these looser conditions the authors claim their attacks to be more practical. Despite this, throughout Section 6 of their paper, they analyze their attack only with four-byte faults. Though these are only minimal values, we also use these results in Table II. When injecting multiple-byte faults in practice, much more faults are necessary to succeed, since less bytes are affected by a fault injection and thus less bytes of the input difference are nonzero.

The authors present three attacks in their paper. The difference between these are the distances between the rounds where faults are injected: the first attack uses faults in the last three rounds, while the second attack uses faults in the penultimate round and two rounds above, and the third attack uses faults only in the round before the penultimate round. Since the third attack uses the least numbers of faults, we start by optimizing that.

The third attack presented by Zhao et al. already uses at least two faults, injected into the 16th round. As before, after identifying candidates for the round keys, they deduce the corresponding secret key candidates and verify one of them as the original secret key. Since this verification process is not described, we assume that they do a brute-force search on a known plaintext-ciphertext pair. As shown before, this type of exhaustive search is not necessary, since the verification process from some of the other attacks can be used [2], [8], [19]. It exploits the property of the CLEFIA key schedule procedure that two of the whitening keys (WK_2, WK_3) store the last two words of the original secret key (K_2, K_3), and by means of this, it uniquely verifies the original secret key. If we use this technique, our attack will be optimal from the information-theoretic perspective, since we do not perform any encryptions on plaintexts.

In case of their first attack, they inject at least 12 faults into the 18th, 17th and 16th rounds altogether, and by means of these faults, they identify the secret key uniquely. If we use the analysis from the above described and improved third attack, we use only two of these faults, the ones injected into the 16th round. Therefore, we reduced the number of faults injected to two, which is claimed in Table II in order to achieve optimality for this attack.

The second attack presented by Zhao et al. injects faults in two rounds. First, they induce at most four faults into two locations in the 17th round, by means of which they deduce the four round keys of the last two rounds. After this, they inject faults the same way into the 15th round and compute the remaining two round keys necessary to reveal the secret key. By examining these four injection points, no existing algorithm can reveal the secret key using only two of the faults. The faults injected in the 17th round can only recover the last four round keys, since they do not affect the 16th round input-output differences. On the other hand, an analysis with two four-byte faults injected in the 15th round is not possible with the existing techniques lacking the knowledge on the value of the fault. If the value of the fault was known, the method by Ali and Mukhopadhyay [2] could be used with the fault value instead of the fault pattern.

Therefore, we did not improve the second attack of Zhao et al., but optimized the first and third attack.

B. Improvements on CLEFIA-192 and CLEFIA-256

In case of the analysis on the attacks against CLEFIA with longer keys, we can see in Table III that there is no existing algorithm which is optimal from the information-theoretic perspective.

In order to deduce the secret 192- or 256-bit key, the most efficient algorithm needs to recover RK_{30}, RK_{31}, $RK_{32} \oplus WK_3$, $RK_{33} \oplus WK_2$, RK_{34}, RK_{35}, $RK_{36} \oplus WK_2$, $RK_{37} \oplus WK_3$, RK_{38}, RK_{39}, $RK_{40} \oplus WK_3$, $RK_{41} \oplus WK_2$, RK_{42} and RK_{43}. Thus, a successful attack needs to calculate the input-output differences of at least the last seven rounds.

The first attack was presented by Chen et al. in 2007 and uses 54 byte faults [8]. The next attack was proposed three years later by Takahashi and Fukunaga and makes use of only 10.78 faults on average [20]. The attack which has the least fault number was presented by Ali and Mukhopadhyay [3]. They inject 8 faults in order to retrieve the 192-bit or 256-bit secret keys.

All these attacks identify the secret key uniquely, yet the best attack from an information-theoretic perspective is the last proposed method by Ali and Mukhopadhyay. Unfortunately, their technique can not be generalized to the other fault models in a simple way because it uses the value of the faults they inject strictly into the first byte of a given register. This register is found before the diffusion matrix of round $r-4$ and $r-8$, so the fault propagates with a given fault pattern shown in Figure 5 of [3]. The attack uses this fault pattern during the calculations of eight round keys. However, we can improve the analysis described by Chen et al [8].

1) Improvement of the Attack by Chen, Wu and Feng: Chen et al. inject the faults in the same area of a round as Ali and Mukhopadhyay in [3], though not strictly in the first, but randomly into one of the four bytes of the register. They induce altogether 54 faults into rounds $r-9$ to $r-1$, i.e., 6 faults are induced in each round. Half of the faults are induced in T_0, and half of the faults are induced in T_2. We, instead, mix the analyses of Ali and Mukhopadhyay [3] and Takahashi and Fukunaga [20], and we apply this mixed technique to the fault model of Chen et al.

Firstly, we use four faults injected only into round $r-4$ (two into T_0, two into T_2). An injected fault f will imply one of four different fault patterns in case of both diffusion matrices M_0 and M_1, depending on which byte the fault was induced into. After calculating the fault patterns summarized in Table IV, the algorithm from [3] can be borrowed. Whenever we use the fault pattern for calculating the input-output differences of the S-boxes, we go through all the four possible patterns for both the fault injections, which means two times 16 checks altogether.

Byte disturbed	Pattern with M_0	Pattern with M_1
1st	$\{f, 2f, 4f, 6f\}$	$\{f, 8f, 2f, 10f\}$
2nd	$\{2f, f, 6f, 4f\}$	$\{8f, f, 10f, 2f\}$
3rd	$\{4f, 6f, f, 2f\}$	$\{2f, 10f, f, 8f\}$
4th	$\{6f, 4f, 2f, f\}$	$\{10f, 2f, 8f, f\}$

Table IV: Fault patterns after injection of a byte fault

After determining the first eight necessary round keys, we use another four faults. We do not inject them four, but three rounds earlier, into round $r-7$. Here, we use the analysis technique from [20] to recover the rest of the necessary round keys. In Section V-A, it is explained why this attack can be directly applied to the fault model of Chen et al.

Originally, Chen et al. injected 6 faults in 9 rounds each, altogether 54 faults. After using only 8 of these faults injected into rounds $r-4$ and $r-7$, we have all the necessary information to calculate the secret key. This way we reduced the number of fault injections to the lowest possible number reached to date for CLEFIA-192 and CLEFIA-256. Our attack cannot be prevented by protecting only the last four rounds of the algorithm. As we will show in Section V-C, it shows a better succes rate than the DFA from [3].

C. Validation of Our Results

We simulated the existing and improved algorithms for CLEFIA-128 and CLEFIA-192. We checked for each method whether it can reveal the secret key with the claimed minimal number of faults. Even though our results show that in a significant number of cases, the analyses succeed with the claimed number of faults, the probability for success varies according to the different algorithms. Nevertheless, we showed that the results presented in Table II are valid and also verified our improvement from Table III. Table V and VI summarize our findings.

1) Methodology: We validated our results on the information-theoretic analysis of the DFAs on the CLEFIA block cipher by means of implementing the analyzed DFA methods and our improvements. As a basis, we used the existing Reference ANSI C code provided by Sony Corporation in 2008 [18]. This reference code does not include any optimizations for high-speed or low-cost implementations. Since our aim was to show that our theoretical results work in practice, we did not need these properties. Thus, this reference code provides a sufficient base for our work.

All our code is written in C. It was executed on a personal computer (Intel Core^{TM} i5-4200U 2.3 GHz CPU and 4GB RAM). Firstly, we implemented the fault injection procedures for the different fault models by modifying the encryption algorithm of the original CLEFIA code. For injecting random faults, we used the built-in pseudorandom number generator of the C language, `rand()` which returns an integer value between 0 and `RAND_MAX`. The value of `RAND_MAX` is constant and is specified in the standard library of C. The random values what we used are ranging between 0 and 255, therefore we used the remainder of the generated random number divided by 256. Secondly,

Differential Fault Attack	Timing	t	$\lvert\mathcal{K}\rvert$	Experiments	Success
Chen, Wu, Feng [8]	15, 16, 17	18	1	2,000	99.1%
Takahashi, Fukunaga [19], [20]	16	2	1	100	97%
Zhao, Wang, Gao [22]	16,17,18	12	1	2,000	81.3%
	15, 17	8	1	2,000	68.7%
	16	2	2^{19}	100	97%
Ali, Mukhopadhyay [2], [3]	14	2	1	2,000	91.45%
Proposed improvement on [8]	15	2	1	100	97%
Proposed improvement on [22]	16	2	1	100	97%

Table V: Experimental results on existing and proposed Differential Fault Analyses on CLEFIA-128. With t faults, we obtained the reduced key space $\mathcal{K}$ in 100 or 2,000 simulation experiments with the given success rate.

we implemented all key recovery procedures from the different DFA methods. Depending on the DFA, during the first phase, they recover either the necessary round keys uniquely or the possible candidates for each round key. Then, during the second phase, they recover the secret key by means of these round keys. In case we only have possible round keys, the identification of the secret key among its candidates also takes place in the second phase of these analyses.

2) Simulation Results for CLEFIA-128: We note that in case of the attacks by Zhao et al. [22], we analyzed them with strictly four-byte faults injected, since the authors only gave fault numbers for this case as well. With the looser multiple-byte fault model, they need much more faults to recover the key, since each of the four bytes has to be affected by a fault at least twice by these fault injections.

Our results verify the trade-off between the probability for success and the number of faults injected: Chen et al. use three faults in order to recover a round key uniquely [8], while other attacks use only two for the same purpose. Thus, their attack needs 18 faults for the six necessary round keys, while the first attack by Zhao et al. uses only 12 [22]. The second attack by the same authors uses 8 faults injected into two rounds only, thus recovering all round keys with less faults. In our simulations, these attacks needed around 0.25 seconds on average, and we executed 2,000 experiments.

The optimal attack by Takahashi and Fukunaga [19] uses only two faults. We used this attack to improve others. Since the 19.02-bit brute-force search needs around 9.5 minutes on average in our setup, we mounted only 100 experiments. The third attack by Zhao et al. [22] with four-byte faults differs from the attack by Takahashi and Fukunaga only in the secret key verification process. Despite the difference from an information-theoretic perspective, they succeed in case of the same samples. The two optimized attacks based on Chen et al. [8] and on Zhao et al. [22] are identical with the attack by Takahashi and Fukunaga: the former uses the dashed rectangle of Figure 6 as fault injection area, and the latter uses the bold line on the same figure.

In case of the attack by Ali and Mukhopadhyay [2], [3] we experimented the probability of success in practice by using the code with known fault values, thus avoiding the brute force search concerning these. Ali and Mukhopadhyay state in Section V-C of [3] that the value of the fault may cancel out during some of the round key calculations. Compared to the other methods using the same amount of faults, this makes the attack less successful as well in our experiments.

3) Simulation Results for CLEFIA-192/256: Takahashi and Fukunaga adapted their attack against CLEFIA-128, but since the brute-force search space would grow to more than 2^{50} otherwise, they identify the round keys of the last six rounds uniquely by injecting at least two faults in each of four location. Then, to identify the remaining necessary round keys, they use Algorithm 2 with two more faults. On average they inject 10.78 faults, and the least fault number is 10 using their method. Thus, we mounted 100 experiments with 10 faults injected.

The attack by Ali and Mukhopadhyay [3] uses only 8 faults, but they succeed with less probability due to the facts that only two faults are used to recover one round key uniquely, and that these faults may cancel out as it happened in case of their analysis on CLEFIA-128.

For our improved method based on the attack by Chen et al. [8], we used both the other two analyses. First, we used the method by Ali and Mukhopadhyay to calculate the round keys of the last four rounds. Afterwards, we calculated the remaining six round keys by injecting faults three rounds earlier and using the method of Takahashi and Fukunaga. This way the value of the fault cancels out less often in our attack than in case of the attack by Ali and Mukhopadhyay, which in practice means around 8% improvement in Table VI.

Differential Fault Attack	Timing	t	$\|\mathcal{K}\|$	Experiments	Success
Chen, Wu, Feng [8]	$r-9,\ldots,r-1$	54	1	2,000	98.3%
Takahashi, Fukunaga [20]	$r-8, r-5, r-2$	10	1	100	51%
Ali, Mukhopadhyay [3]	$r-8, r-4$	8	1	2,000	43.4%
Proposed improvement on [8]	$r-7, r-4$	8	1	2,000	51.2%

Table VI: Experimental results on existing and proposed Differential Fault Analyses on CLEFIA-192/256. With t faults, we obtained the reduced key space $\mathcal{K}$ in 100 or 2,000 simulation experiments with the given success rate.

VI. Conclusion

The information-theoretic approach allowed us to determine whether or not the existing Differential Fault Analyses on CLEFIA are optimal. The analysis shows that an attacker needs at least two faults to reveal the secret 128-bit key. Based on these findings, we successfully improved all but one attack against CLEFIA-128. Now, from an information-theoretic perspective, the improved Differential Fault Analyses on CLEFIA-128 all reach the theoretical limit. For longer keys, we considerably improved one of the existing attacks. Our proposed attack reaches the lowest number of faults to date. However, Differential Fault Analyses on CLEFIA-192 and CLEFIA-256 can still be improved theoretically and an optimal attack still needs to be discovered.

Acknowledgment.

Juliane Krämer gratefully acknowledges support by the Deutsche Telekom Stiftung.

References

[1] Subidh Ali and Debdeep Mukhopadhyay. A Differential Fault Analysis on AES Key Schedule Using Single Fault. In *2011 Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC)*, pages 35–42, 2011.

[2] Subidh Ali and Debdeep Mukhopadhyay. Protecting Last Four Rounds of CLEFIA is Not Enough Against Differential Fault Analysis. *IACR Cryptology ePrint Archive*, 2012:286, 2012.

[3] Subidh Ali and Debdeep Mukhopadhyay. Improved Differential Fault Analysis of CLEFIA. In *2013 Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC)*, pages 60–70, 2013.

[4] Subhadeep Banik and Subhamoy Maitra. A Differential Fault Attack on MICKEY 2.0. In *Cryptographic Hardware and Embedded Systems - CHES 2013*, pages 215–232. Springer Berlin Heidelberg, 2013.

[5] Hagai Bar-El, Hamid Choukri, David Naccache, Michael Tunstall, and Claire Whelan. The sorcerer's apprentice guide to fault attacks. *IACR Cryptology ePrint Archive*, 2004:100, 2004.

[6] Eli Biham and Adi Shamir. Differential Fault Analysis of Secret Key Cryptosystems. In *CRYPTO*, volume 1294 of *Lecture Notes in Computer Science*, pages 513–525. Springer Berlin Heidelberg, 1997.

[7] Dan Boneh, Richard A. DeMillo, and Richard J. Lipton. On the Importance of Checking Cryptographic Protocols for Faults. EUROCRYPT'97, pages 37–51. Springer Berlin Heidelberg, 1997.

[8] Hua Chen, Wenling Wu, and Dengguo Feng. Differential Fault Analysis on CLEFIA. In *ICICS*, volume 4861 of *Lecture Notes in Computer Science*, pages 284–295. Springer Berlin Heidelberg, 2007.

[9] William Feller. *An Introduction to Probability Theory and Its Applications*, volume 1. Wiley, January 1968.

[10] Wieland Fischer and Christian A. Reuter. Differential Fault Analysis on Grøstl. In *2012 Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC)*, pages 44–54, 2012.

[11] S. Karmakar and D.R. Chowdhury. Differential Fault Analysis of MICKEY-128 2.0. In *2013 Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC)*, pages 52–59, 2013.

[12] Gilles Piret and Jean-J. Quisquater. A Differential Fault Attack Technique Against SPN Structures, with Application to the AES and KHAZAD. In *Cryptographic Hardware and Embedded Systems - CHES 2003*, pages 77–88. Springer Berlin Heidelberg, 2003.

[13] Chester Rebeiro, Rishabh Poddar, Amit Datta, and Debdeep Mukhopadhyay. An Enhanced Differential Cache Attack on CLEFIA for Large Cache Lines. In *INDOCRYPT*, volume 7107 of *Lecture Notes in Computer Science*, pages 58–75. Springer Berlin Heidelberg, 2011.

[14] Matthieu Rivain. Differential Fault Analysis on DES Middle Rounds. In *Cryptographic Hardware and Embedded Systems - CHES 2009*, pages 457–469. Springer Berlin Heidelberg, 2009.

[15] Kazuo Sakiyama, Yang Li, Mitsugu Iwamoto, and Kazuo Ohta. Information-Theoretic Approach to Optimal Differential Fault Analysis. *IEEE Transactions on Information Forensics and Security*, 7(1):109–120, 2012.

[16] Claude Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423, 623–656, July, October 1948.

[17] Taizo Shirai, Kyoji Shibutani, Toru Akishita, Shiho Moriai, and Tetsu Iwata. The 128-Bit Blockcipher CLEFIA (Extended Abstract). In *FSE*, volume 4593 of *Lecture Notes in Computer Science*, pages 181–195. Springer Berlin Heidelberg, 2007.

[18] Sony Corporation. The 128-bit Blockcipher CLEFIA, Reference ANSI C code. http://www.sony.co.jp/Products/cryptography/clefia/download/data/clefia_ref.c, 2008. accessed 2014/05/05.

[19] Junko Takahashi and Toshinori Fukunaga. Improved Differential Fault Analysis on CLEFIA. In *2008 Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC)*, pages 25–34, 2008.

[20] Junko Takahashi and Toshinori Fukunaga. Differential Fault Analysis on CLEFIA with 128, 192, and 256-Bit Keys. *IEICE Transactions*, 93-A(1):136–143, 2010.

[21] Yukiyasu Tsunoo, Etsuko Tsujihara, Maki Shigeri, Teruo Saito, Tomoyasu Suzaki, and Hiroyasu Kubo. Impossible Differential Cryptanalysis of CLEFIA. In *FSE*, volume 5086 of *Lecture Notes in Computer Science*, pages 398–411. Springer Berlin Heidelberg, 2008.

[22] Xin-jie Zhao, Tao Wang, and Jing zhe Gao. Multiple Bytes Differential Fault Analysis on CLEFIA. *IACR Cryptology ePrint Archive*, 2010:78, 2010.