IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 26, 2019, accepted September 17, 2019, date of publication September 26, 2019, date of current version October 10, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2944024

Information Theory-Based Quantitative Evaluation Method for Countermeasures Against Fault Injection Attacks

QIANG LIU, (Member, IEEE), BO NING, AND PENGJIE DENG

Tianjin Key Laboratory of Imaging and Sensing Microelectronic Technology, School of Microelectronics, Tianjin University, Tianjin 300072, China

Corresponding author: Qiang Liu (qiangliu@tju.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61574099, and in part by the Tianjin Municipal Transportation Science and Technology Development Plan Project under Grant 2017B-40.

ABSTRACT The integrated circuits are faced with the threats of fault injection attacks (FIAs), which employ the faulty results and differential fault analysis to retrieve the vital information processed by the circuits. Countermeasures should be taken to protect the circuits against FIAs. This paper proposes a security evaluation method to quantitatively analyze the resistance of block ciphers with a Substitution-Permutation Network structure against FIAs from an information-theoretic perspective. A quantitative security factor is defined based on the amount of information leakage. Specifically, an extended cipher model is proposed to numerically analyze the theoretical amount of information leakage, and an efficient approach is proposed to obtain the actual amount of information leakage. Experiments by applying the quantitative evaluation method to AES circuit validate feasibility, efficiency and scalability of the method. The experimental results show that the security factor can quantify the effects of different fault models and countermeasures. 2000 fault injections are sufficient to complete the evaluation under the considered fault models within 10 microseconds. The proposed method can be used during the circuit design stage as well as chip testing stage.

INDEX TERMS Fault injection attack, differential fault analysis, security quantitative evaluation, substitution-permutation network, information theory.

I. INTRODUCTION

Fault injection attack (FIA) has become a serious threat to the security of integrated circuits due to the development of low-cost fault injection techniques, such as clock glitch and power glitch [1], and efficient analysis methods such as differential fault analysis (DFA) [2], which employ the faulty results caused by FIA to retrieve secret information.

To resist FIA, countermeasures are proposed and can be classified into four categories [3]: cutting the access point, environment monitoring, fault detection and fault correction. The first type of countermeasure makes it difficult to inject faults into the circuits by cutting off the possible access points needed by the fault injection means [4]. The second type monitors the circuits' physical environment changes using various sensors such as voltage sensors and light sensors [5].

The associate editor coordinating the review of this manuscript and approving it for publication was Francesco Tedesco.

The third kind is based on spatial redundancy [6], temporal redundancy [7] and information redundancy [8] to detect faults caused by FIA. The last kind is based on redundancy to correct faults [9], [10]. Although faults can be injected into the circuits, but the outputs of the circuits are corrected. Different countermeasures bring different degrees of security and overheads. In practice, IC designers need a method to quantify circuit's security in order to evaluate various countermeasures and choose the effective one [11], [12].

A. PREVIOUS WORK

Currently, the evaluation of cryptographic circuits with the countermeasures mostly focuses on the error detection capability.

Multiple cocurrent error detection (CED) techniques are compared in [13] in terms of area overhead, power consumption, performance penalties and error detection capabilities. Errors affecting a single byte of the State matrix and random

errors affecting any subset of the State matrix are injected. The percentage of undetected errors was used to evaluate the error detection capability. In [14], the fault injection results at the configuration memory of a SRAM-based FPGA with different detection mechanisms were divided into four categories: detected unrecoverable error (DUE), silent data corruption (SDC), functional interruption (SEFI) and successful completion (OK). The ratio of each category was discussed and used to evaluate different detection mechanisms. The samller the ratio of SDC is, the stronger the detection capability of the detection mechanism is. In [15],the reliability is merged with overhead to compare Triple Modular Redundancy (TMR) and Concurrent Error Detection (CED) by defining the reliability improvement efficiency with respect to the overhead.

These evaluation methods mainly use the number of faulty outputs caused by FIA as the evaluation metric, which was widely used in circuits' reliability and fault tolerance study. The reliability usually is evaluated in terms of whether errors occur or not, while the security is evaluated in terms of whether vital information leaks or not. In other words, faulty outputs which do not leak security information are acceptable for security design.

Another type of evaluation method is based on cryptoanalysis. In [16], fault differential entropy (FDE) is proposed to analyze the resistance of concurrent error detection against DFA by judging whether the correct key is successfully distinguished or comparing the number of fault injections when the correct key is distinguished. However, this type of evaluation method is tailored for the specific cryptoanalysis, and it is difficult to experiment with all cryptanalysis approaches.

Overall, we found that existing methods do not directly analyze how secure the information processed by FIAs is, and just analyze the phenomena related to FIAs.

B. CONTRIBUTIONS

In this paper, a security quantitative evaluation method is proposed based on information theory. A metric called security factor is defined to quantify the difference between the theoretical amount of information leakage and the actual information leakage. The theoretical amount of information leakage can be derived from the FIA models. The actual amount of information leakage is calculated by the fault injection results at the hardware circuits. The closer the actual amount of information leakage is to the theoretical value, the lower the security of circuits is. The security factor tells us how secure the circuit design is against the FIAs and how the countermeasure designs can be improved. The method can be applied at the design stage as well as the chip testing stage.

The main contributions of this paper are

(1) A security factor based on information theory is defined, which can be used to quantify the security of circuits with countermeasure designs against FIA.

(2) A new method is proposed for deriving the amount of information leakage from an extended cipher model where FIA can be applied at different rounds of encryption.

(3) An efficient method for calculating the actual amount of information leakage is proposed. The method reduces the minimum number of fault injections by $255^b - 255$, and time required the security evaluation is reduced greatly.

This paper is organized as follows. Section II briefly introduces the original derivation method for the amount of information leakage. Section III proposes a new method of deriving the amount of information leakage and introduces the security factor to illustrate our security quantitative evaluation method. Section IV describes two algorithms of calculating the actual amount of information leakage. Section V gives the experimental setup and Section VI analyzes the experimental results. Finally, Section VII concludes the paper.

II. BACKGROUND AND PROBLEM STATEMENT

Information-theoretic approach which uses the amount of information leakage to judge if a DFA attack on AES is optimal was proposed in [17]. In this approach, the amount of information leakage corresponding to a fault model is computed to theoretically estimate the reduced key space for each fault. Each fault model defines a theoretical upper bound on the amount of information leakage and a theoretical lower bound on the reduced key space. The actual reduced key space for a DFA attack with a certain fault model is compared with the lower bound. If the actual reduced key space is close to the lower bound, the number of fault injection needed is relatively small and corresponding DFA is optimal. The notations used in this paper are as follow.

K	n-bit key
x	n-bit state variable where faults are injected
x_1	correct state variable
x_2	faulty state variable
Δx	difference between x_1 and x_2, i.e., the fault value
X	discrete random variable with possible values of x
ΔX	discrete random variable with possible values of Δx
y	n-bit output of $S(\cdot)$ or the last $SP(\cdot)$
y_1	correct output of $S(\cdot)$ or the last $SP(\cdot)$
y_2	faulty output of $S(\cdot)$ or the last $SP(\cdot)$
Δy	difference between y_1 and y_2
Y	discrete random variable with possible values of y
ΔY	discrete random variable with possible values of Δy
z	n-bit ciphertext of the encryption algorithm
z_1	correct ciphertext
z_2	faulty ciphertext
Δz	difference between z_1 and z_2
Z	discrete random variable with possible values of z
ΔZ	discrete random variable with possible values of Δz

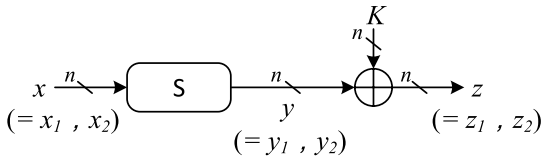


FIGURE 1. A simple cipher model using *n*-bit substitution.

The approach used a simple cipher model shown in Fig.1. x is the n-bit input of substitution function $S(\cdot)$, and y is the n-bit output, i.e., $y = S(x)$. Value z is the XORed output of y with secret key K. From Fig.1, we can obtain

$$z_1 \oplus z_2 = (y_1 \oplus k) \oplus (y_2 \oplus k) = y_1 \oplus y_2 = S(x_1) \oplus S(x_2). \tag{1}$$

From (1), we can observe that the difference Δy depends on x_1 and x_2, but is independent of K (***Condition 1***).

The amount of information leakage m is defined as the degree to which uncertainty of key is reduced based on the correct ciphertext and the faulty ciphertext, i.e., the mutual information of K and Z_1 Z_2 ($I(K; Z_1Z_2)$). According to the information theory,

$$m = I(K; Z_1Z_2) = H(K) - H(K|Z_1Z_2) \tag{2}$$

where $H(K)$ is the entropy of key, i.e., the number of bits n of K, and $H(K|Z_1Z_2)$ denotes the amount of additional information needed to describe K when Z_1Z_2 is known. The focus of calculating the amount of information leakage is to derive $H(K|Z_1Z_2)$.

Since there exists a one-to-one correspondence between (z_1, z_2) and $(z_1, \Delta z)$, $H(K|Z_1Z_2)$ can be derived as below [17].

$$\begin{aligned} H(K|Z_1Z_2) &= H(Z_1Z_2K) - H(Z_1Z_2) \\ &= H(Z_1Z_2K) - H(Z_1\Delta Z) \\ &= H(Z_1Z_2K) - H(\Delta Z) - H(Z_1|\Delta Z). \end{aligned} \tag{3}$$

$H(Z_1Z_2K)$ is equal to $H(X_1X_2K)$, because the substitution function $S(\cdot)$ and the XOR function are both reversible, and there exists a one-to-one correspondence between (x_1, x_2, k) and (z_1, z_2, k).

$$H(Z_1Z_2K) = H(X_1X_2K). \tag{4}$$

Because K is independent of X_1 and X_2, ΔY is uniquely determined from X_1 and X_2 according to ***Condition 1***. So, it holds that [17]

$$\begin{aligned} H(X_1X_2K) &= H(X_1X_2) + H(K) \\ &= H(X_1X_2\Delta Y) + H(K) \\ &= H(X_1X_2|\Delta Y) + H(\Delta Y) + H(K). \end{aligned} \tag{5}$$

Because $H(\Delta Z) = H(\Delta Y)$ and $H(Z_1|\Delta Z) = H(K)$ (the specific derivation is referred to [17]), $H(K|Z_1Z_2)$ is determined finally by (3), (4) and (5).

$$\begin{aligned} H(K|Z_1Z_2) &= H(X_1X_2|\Delta Y) \\ &= H(X_1\Delta X|\Delta Y) \\ &= H(\Delta X|\Delta Y) + H(X_1|\Delta X\Delta Y) \end{aligned} \tag{6}$$

Therefore, the amount of information leakage is (7).

$$m = n - H(\Delta X|\Delta Y) - H(X_1|\Delta X\Delta Y) \tag{7}$$

This formula was used to estimate the leaked information of the simple model under different DFA analysis approaches. The simple model shows the fault propagation process of only one-round encryption when faults are injected into the input of the last round. ***Condition 1*** is an important condition for (5). However, if the fault propagation process includes multi-round encryption, XOR operation exists before the last substitution function, and Δy at the last round is not independent of K. Thus, (5) is not valid, and formula (7) is not suitable for the fault propagation process of multi-round encryption. In practice, faults can be injected into early rounds of encryption. The presented paper extends the cipher model with multi-round encryption and derives the corresponding formula for m, which is then used in security evaluation.

III. SECURITY QUANTITATIVE EVALUATION METHOD

In this section, we introduce the security factor to quantify the security of cryptographic circuits against FIA with uniform fault models. The security factor is computed by comparing the theoretical amount of information leakage and the actual amount of information leakage. Thus, we first derive a new calculation formula for leaked information using an extended cipher model, and then define the security factor.

A. AMOUNT OF INFORMATION LEAKAGE

To consider the cases in which FIA is applied to various rounds of encryption and derive the theoretical amount of information leakage, a complex cipher model is shown in Fig.2. In the complex model, the substitution function $S(\cdot)$ is replaced by the substitution-permutation function $SP(\cdot)$, which includes not only the substitution function, but also the diffusion layer. Thus, the complex model depicts any block cipher with a Substitution-Permutation network (SPN) structure [18]. x denotes the state at the position of fault injection, y denotes the output of last $SP(\cdot)$, and z denotes the ciphertext of cryptographic algorithm, and is still the XORed output of y and K. The complex model shows the fault propagation process of multi-round encryption, which is an extension of the simple model. If fault propagation process includes multi-round operation, the dashed part is executed one or more times during fault propagation; if faults are injected into the last round, the dashed part is not included in the fault propagation process, and the complex model is degraded to the simple model.

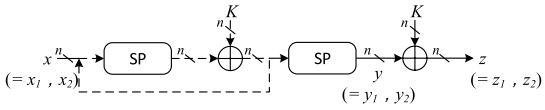


FIGURE 2. A complex cipher model.

Clearly, in the complex cipher model, Δy at the output of the last $SP(\cdot)$ is not only determined by x_1 and x_2 but also K.

Therefore, (5) is not valid in the complex model. However, because there exists a one-to-one correspondence between (x_1, x_2) and $(x_1, \Delta x)$, and X_1 is independent of ΔX, (5) can be modified to (8).

$$\begin{aligned} H(X_1X_2K) &= H(X_1X_2) + H(K) \\ &= H(X_1\Delta X) + H(K) \\ &= H(X_1) + H(\Delta X) + H(K). \end{aligned} \tag{8}$$

In this way, $H(X_1X_2K)$is only related to the model inputs, and is not affected by multi-round operations, suitable to the complex model.

We have $H(Z_1|\Delta Z) = H(Z_1)$ because Z_1 is independent of ΔZ. Thus, (3) can be replaced by (9).

$$H(K|Z_1Z_2) = H(Z_1Z_2K) - H(\Delta Z) - H(Z_1). \tag{9}$$

Since there still exists a one-to-one correspondence between (x_1, x_2, k) and (z_1, z_2, k), (4) is still valid, i.e., $H(Z_1Z_2K) = H(X_1X_2K)$, and because $H(X_1) = H(K) = H(Z_1) = n$, $H(K|Z_1Z_2)$ can be expressed as

$$\begin{aligned} H(K|Z_1Z_2) &= H(X_1X_2K) - H(\Delta Z) - H(Z_1) \\ &= H(X_1)+H(\Delta X)+H(K)-H(\Delta Z)-H(Z_1) \\ &= [H(X_1)+H(K)-H(Z_1)]+H(\Delta X)-H(\Delta Z) \\ &= n + H(\Delta X) - H(\Delta Z) \end{aligned} \tag{10}$$

Therefore, the new calculation formula for the amount of information leakage is

$$\begin{aligned} m &= n - (n + H(\Delta X) - H(\Delta Z)) \\ &= H(\Delta Z) - H(\Delta X) \end{aligned} \tag{11}$$

which is derived from (2) and (10). $H(\Delta Z)$ indicates the uncertainty of the difference between the correct ciphertext and the faulty ciphertext, and $H(\Delta X)$ indicates the uncertainty of fault value, which is determined by the specified uniform fault models.

Given an arbitrary random variable X with possible values of x, the distribution of X is uniform and the number of possible values of x is n_x. The probability of each x is equal to $\frac{1}{n_x}$. According to the information theory, the entropy of X is calculated by (12), where $p(x)$ denotes the probability of a possible value x.

$$\begin{aligned} H(X) &= -\sum_{x\in X} p(x)log_2(p(x)) \\ &= \sum_{x\in X} \frac{1}{n_x} log_2(n_x) \\ &= log_2(n_x) \end{aligned} \tag{12}$$

When a uniform fault model is specified, the number of possible values of ΔX denoted by $n_{\Delta x}$ is determined by the size of fault space, and the number of possible values of ΔZ denoted by $n_{\Delta z}$ is also determined because the detail of cryptographic algorithm is known. Due to the uniform distribution of ΔX, the entropy of ΔX is $log_2(n_{\Delta x})$ as shown in (12). However, the distribution of ΔZ is uncertain. When fault propagation process includes multi-round encryption, ΔZ is assumed to be uniform due to the uncertainty of X_1 and K and the multi-round confusion and diffusion. The entropy of ΔZ is $log_2(n_{\Delta z})$. When fault propagation process includes only one-round encryption, it is easy to obtain the probability of each possible value of ΔZ by enumerating all possible X_1 and ΔX which have limited combinations of values. The entropy of ΔZ is calculated by

$$H(\Delta Z) = -\sum_{\Delta z\in\Delta Z} p(\Delta z)log_2(p(\Delta z)), \tag{13}$$

which will be demonstrated by subsequent experimental results. Therefore, both $H(\Delta Z)$ and $H(\Delta X)$ can be derived from the specified uniform fault model and the targeted cryptographic algorithm theoretically.

B. SECURITY FACTOR

In practice, cryptographic algorithm is implemented with some countermeasures to resist the fault injection attack and reduce the leakage of secret information.

In particular, countermeasures could cause $Z_2 = Z_1$, so that ΔZ becomes 0. This kind of countermeasure can change the distribution of ΔZ. As a result, the actual amount of information leakage m' deviates from the theoretical amount of information leakage m.

For the cryptographic circuit which is implemented with the countermeasures, we use the security factor to quantitatively evaluate the effectiveness of the countermeasures, i.e., the security of the cryptographic circuit. The security factor denoted by γ quantifies the difference between the actual amount of information leakage and the theoretical amount of information leakage, as shown in (14).

$$\gamma = \frac{m - m'}{m} \tag{14}$$

In the next section, we will show how m' is derived.

IV. CALCULATION METHOD FOR ACTUAL AMOUNT OF INFORMATION LEAKAGE

For a specified fault model, $H(\Delta X)$ is definite, and the difference between the theoretical amount of information leakage and the actual amount of information leakage is $H(\Delta Z)$. The theoretical value of $H(\Delta Z)$ can be derived from the fault model and the cryptographic algorithm as described in Section III-A. The actual value of $H(\Delta Z)$ is calculated by the results of fault injection. In this section, we give guidance on how to calculate the actual values of $H(\Delta Z)$ to obtain the actual amount of information leakage. In addition, we propose an efficient method of calculating $H(\Delta Z)$ when ΔZ is multi-byte.

To calculate actual $H(\Delta Z)$, the probability of each possible ΔZ value is replaced by its frequency, which is based on the results of fault injection.

For a random plaintext, the correct ciphertext and the faulty ciphertext are recorded, and the difference ΔZ of the correct ciphertext and the faulty ciphertext is collected. Then, ΔZ is

processed to calculate the entropy of ΔZ, the actual amount of information leakage and the security factor. The process of generating $H(\Delta Z)$ and m' is shown below.

Algorithm 1 Process Method for m'

Input: f_1, set of possible values of $\Delta Z\{\Delta z_i, 0 \le i \le f_1 - 1\}$, $H(\Delta X)$
Output: $H(\Delta Z)$, m'
  //Step 1: Sorting
  HeapSort($\{\Delta z_i\}, f_1$)
  //Step 2: Counting
  $count\_dz \leftarrow 0$
  $H(\Delta Z) \leftarrow 0$
  $i \leftarrow 0$
  for $i < f_1$ **do**
    $text\_dz \leftarrow \Delta z_i$
    $i \leftarrow i + 1$, $count\_dz \leftarrow count\_dz + 1$
    while $text\_dz == \Delta z_i$ **do**
      $i \leftarrow i + 1$, $count\_dz \leftarrow count\_dz + 1$
    end while
    $H(\Delta Z) \leftarrow H(\Delta Z) + \frac{count\_dz}{f_1} \log_2 \frac{f_1}{count\_dz}$
    $count\_dz \leftarrow 0$
  end for
  //Step 3: Calculating m'
  $m' \leftarrow H(\Delta Z) - H(\Delta X)$

The calculation method of $H(\Delta Z)$ and m' is shown in Algorithm 1. The number of fault injections denoted by f_1, the set $\{\Delta z_i\}$ of possible values of ΔZ and $H(\Delta X)$ derived from the specified fault model are known. The calculation method includes three steps: sorting, counting and calculating m'. $H(\Delta Z)$ is calculated in the first two steps. In Step 1, $\{\Delta z_i\}$ is sorted in ascending order using the Heap Sort algorithm so that the samples with the same value become continuous. Heap Sort is chosen because of its low time complexity. In Step 2, the number of each possible values of ΔZ is counted by traversing the sorted data one by one. Then, the number of each possible values of ΔZ is divided by f_1 to calculate the frequency of each value. The entropy of ΔZ is calculated by (13), where $p(\Delta z)$ is replaced by $\frac{count\_dz}{f_1}$. Finally, m' is calculated by (11).

The time complexity of Algorithm 1 is $O(f_1 log f_1 + f_1)$, where the first term is the time complexity of Heap Sort and the second term is the time complexity of the counting step.

In the SPN structure, a one-byte fault may cause multi-byte error in ciphertext. Taking AES-128 as an example, when a single byte fault is injected into the input of the ninth round, four bytes in ciphertext are faulty, and ΔZ is four-byte. For four-byte ΔZ, there are 255^4 possible values. At least 255^4 ΔZ samples are required. According to the Bernoulli's law of large numbers, the larger the number of samples is, the closer the frequency of each possible ΔZ value is to its probability. Thus, the number of four-byte ΔZ samples needs to be greater than 255^4. Handling such large amount of data will cause problems in memory and time when running Algorithm 1.

Therefore, an efficient method of calculating $H(\Delta Z)$ is proposed, when ΔZ is multi-byte and each byte of ΔZ is independent of each other. We denote each byte of ΔZ by ΔZ_i, where the subscript i denotes the byte index. Obviously, the entropy of b-byte ΔZ can be considered as the joint entropy of all ΔZ_i, i.e., $H(\Delta Z_0 \Delta Z_1 ... \Delta Z_i ... \Delta Z_b)$. Because ΔZ_i is independent of each other, the joint entropy of all ΔZ_i is equal to the addition of each ΔZ_i entropy. Thus, $H(\Delta Z)$ can be calculated as shown in (15).

$$\begin{aligned} H(\Delta Z) &= H(\Delta Z_0 \Delta Z_1 ... \Delta Z_i ... \Delta Z_b) \\ &= H(\Delta Z_0) + H(\Delta Z_1) + ... + H(\Delta Z_i) + ... + H(\Delta Z_b). \end{aligned} \tag{15}$$

Algorithm 2 Efficient Process Method for m' Given b-Byte ΔZ

Input: f_2, b, $\{\Delta z_{ij}, 0 \le i \le b - 1, 0 \le j \le f_2 - 1\}$, $H(\Delta X)$
Output: $H(\Delta Z)$, m'
  $H(\Delta Z) \leftarrow 0$
  for $i = 0$ to $b - 1$ **do**
    //Step 1: Sorting
    HeapSort($\{\Delta z_{ij}\}, f_2$)
    //Step 2: Counting
    $count\_dz \leftarrow 0$
    $H(\Delta Z_i) \leftarrow 0$
    $i \leftarrow 0$
    for $j < f_2$ **do**
      $text\_dz \leftarrow \Delta z_{ij}$
      $j \leftarrow j + 1$, $count\_dz \leftarrow count\_dz + 1$
      while $text\_dz == \Delta z_{ij}$ **do**
        $j \leftarrow j + 1$, $count\_dz \leftarrow count\_dz + 1$
      end while
      $H(\Delta Z_i) \leftarrow H(\Delta Z_i) + \frac{count\_dz}{f_2} \log_2 \frac{f_2}{count\_dz}$
      $count\_dz \leftarrow 0$
    end for
    $H(\Delta Z) \leftarrow H(\Delta Z) + H(\Delta Z_i)$
  end for
  //Step 3: Calculating m'
  $m' \leftarrow H(\Delta Z) - H(\Delta X)$

Algorithm 2 shows the procedure. The input b is the number of bytes of ΔZ. The process of calculating $H(\Delta Z_i)$ is similar to Algorithm 1. Then, $H(\Delta Z)$ is obtained by adding all $H(\Delta Z_i)$, and m' is calculated. The time complexity of Algorithm 2 is $b \times O(f_2 log f_2 + f_2)$.

The advantage of Algorithm 2 is that the minimum number of fault injections f_2 is significantly smaller than f_1 in Algorithm 1.

For each one-byte ΔZ_i, the minimum number of ΔZ_i samples is 255. In addition, a fault injection can simultaneously provide samples for multiple one-byte ΔZ_i with different i. In other words, for the multi-byte ΔZ, the minimum number of fault injection required for Algorithm 2 is only $f_2 = 255$. Therefore, the execution time of Algorithm 2 is

roughly estimated to be $O(b \times 255log255 + b \times 255)$. However, the minimum f_1 required for Algorithm 1 is 255^b. The execution time of Algorithm 1 is estimated to be $O(b \times 255^b log255 + 255^b)$. Therefore, Algorithm 2 greatly reduces the execution time for the evaluation by reducing the number of fault injections.

V. EXPERIMENTAL SETUP

To verify the proposed evaluation method, we apply it to AES-128 encryption circuit. The experimental flow is shown in Fig.3. AES-128, fault injection, protection and evaluation are all simulated using C programs. 16-byte plaintexts are randomly generated, and random faults are injected into the specific byte position at the specific round according to the fault models. The set of ΔZ caused by fault injections is collected. Then, ΔZ is transferred to the evaluation module to calculate the entropy of ΔZ. At the same time, the entropy of ΔX and the theoretical amount of information leakage derived from the specified fault models are inputted to the evaluation module to calculate the actual amount of information leakage and the security factor. Prior to evaluation, ΔZ data set is pre-processed to simulate the effects of countermeasures. To simulate the different degrees of protection, different proportions of ΔZ are modified to zero. The greater protection is, the greater the proportion of ΔZ modified to 0 is.

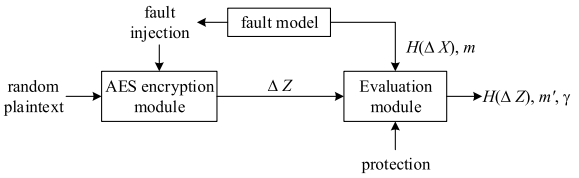


FIGURE 3. Experimental flow.

We use four fault models to attack AES-128. The four fault models are, respectively, the position-known 1-byte fault in the input state of the tenth round, the position-known 1-bit fault in the input state of the tenth round, the position-known 1-byte fault in the input state of the ninth round and the position-known 1-bit fault in the input state of the ninth round, which are abbreviated as *1-byte-10th*, *1-bit-10th*, *1-byte-9th* and *1-bit-9th*. The "position-known" means that attackers know which byte of the state the fault is injected into, but cannot control which bit is flipped, i.e., the fault value is random.

The experiments include three parts. The first part derives the theoretical amount of information leakage from the specified four fault models on AES-128. The experiment validates the extended cipher model and the theoretical information leakage analysis method.

The second part evaluates the methods for obtaining the actual amount of information leakage. For the *1-bit-10th* and *1-byte-10th* fault model, ΔZ is one-byte, and Algorithm 1 is used. For the *1-bit-9th* and *1-byte-9th* fault model, ΔZ is four-byte. Algorithm 2 is used and compared to Algorithm 1. To facilitate the sorting operation, ΔZ or ΔZ_i need to be converted to decimal.

The third part analyzes the influences of the fault models, the degree of protection and the number of fault injections on the actual amount of information leakage and the security factor. Evaluation time is analyzed under different fault models and different amounts of fault injections. For each fault model and fault injection amount, the evaluation method is executed ten times and the execution time is averaged over the ten times.

VI. EXPERIMENTAL RESULTS

A. THEORETICAL AMOUNT OF INFORMATION LEAKAGE

The theoretical amount of information leakage can be derived from the specified uniform fault model as described in Section III-A. When the *1-bit-9th* fault model is specified, ΔX includes 8 uniformly distributed fault values. Thus, $H(\Delta X)$ is 3 according to (12). Because of MixColumns at the ninth round, the fault injected into a state byte diffuses to four ciphertext bytes. The corresponding four bytes in ΔZ is non-zero, while the other bytes are always zero. Each non-zero byte has 255 possible values, so ΔZ includes 255^4 possible values. Due to the uncertainty of X_1 and K and the confusion of 2-round SubBytes, we assume that ΔZ is uniformly distributed. We obtain $H(\Delta Z) = log_2(255^4) = 4log_2255$. Thus, the theoretical amount of information leakage is about 29 according to (11). Similarly, we obtain that m under *1-byte-9th* is $(4log_2255 - log_2255 \approx 24)$.

For the fault models *1-bit-10th* and *1-byte-10th*, a fault injected into a single byte at the input of the tenth round causes only one faulty ciphertext byte. Since both ΔX and X are uniform, all possible values of X and all possible values of ΔX occur once when calculating ΔZ.

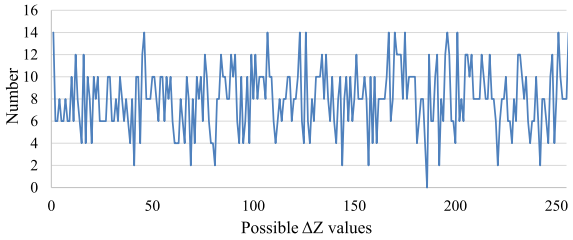


FIGURE 4. Distribution of ΔZ in the *1-bit-10th* model.

For the *1-bit-10th* fault model, ΔX has 8 possible values and X has 256 possible values. Thus, the total number of ΔZ is 2048 and $H(\Delta X)$ is 3 according to (12). The number of each values of ΔZ is counted by enumerating all cases analytically, and is shown in Fig.4. The distribution of ΔZ is not completely uniform. Therefore, we need to calculate the probability of each ΔZ which equals to the number of each values of ΔZ over 2048. As a result, the theoretical $H(\Delta Z)$ is 7.90 and m is 4.90.

For the *1-byte-10th* fault model, ΔX has 255 values and X has 256 values. Thus, the total number of ΔZ is 65280 and $H(\Delta X)$ is log_2255. Through the experiment of

calculating ΔZ, we find that the number of each possible values of ΔZ is always 256, and the distribution is uniform. For every $\Delta z \in \Delta Z$, $p(\Delta z)$ is $\frac{1}{255}$. Thus, the theoretical $H(\Delta Z)$ is $log_2 255$ and m is 0. When the theoretical amount of information leakage is 0, the correct ciphertext and the faulty ciphertext do not leak any information under this fault model. Thus, the *1-byte-10th* fault model should not be used for attacks.

B. ACTUAL AMOUNT OF INFORMATION LEAKAGE UNDER 1-BIT-9TH FAULT MODEL

The *1-bit-9th* fault model is used to evaluate Algorithm 1 and Algorithm 2 in obtaining the actual amount of information leakage. The number of fault injections is 10^5, 10^6, 10^7, and 10^8, respectively. The AES circuit does not contain countermeasures. The results are shown in Fig.5, where the theoretical amount of information leakage m is also shown. For Algorithm 1, the actual amount of information leakage increases as the number of fault injections increases, but the value is far below m. For Algorithm 2, the actual amount of information leakage is basically stable and very close to m. This demonstrates that Algorithm 2 is more efficient than Algorithm 1 for four-byte errors caused by attacks, and 10^5 fault injections are sufficient for security evaluation. The time required for Algorithm 2 is only 0.138s when the number of fault injection is 10^5.

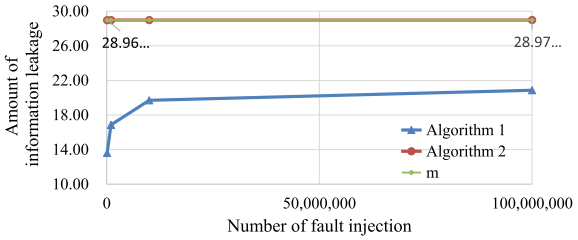


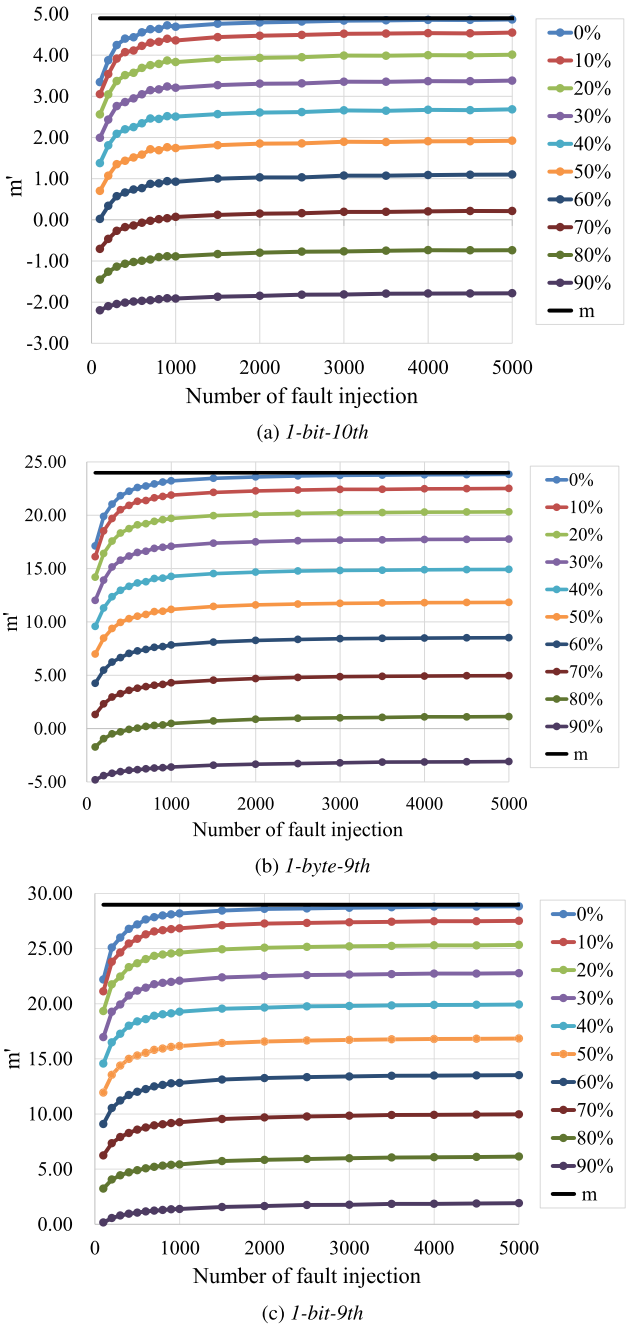
FIGURE 5. Comparison of Algorithm 1 and Algorithm 2 in obtaining the actual amount of information leakage, when the *1-bit-9th* fault model is applied to the original AES circuit and the number of fault injections is 10^5, 10^6, 10^7 and 10^8, respectively.

C. INFLUENCES OF FAULT MODEL, DEGREE OF PROTECTION, NUMBER OF FAULT INJECTIONS ON M′ AND γ

1) ACTUAL AMOUNT OF INFORMATION LEAKAGE

Fig.6 shows influences of the different fault models, the different degrees of protection and the different numbers of fault injections on the actual amount of information leakage. Lines of different colors indicate different degrees of protection (0-90%). The theoretical amount of information leakage (m) is also shown for comparison.

Firstly, when the fault model and the degree of protection are fixed, the actual amount of information leakage increases and becomes saturated as the number of fault injections increases. When the number of fault injections is 2000, the curve of the actual amount of information leakage is basically stable and the actual information leakage without protection (0%) is very close to the theoretical value.



(a) *1-bit-10th*

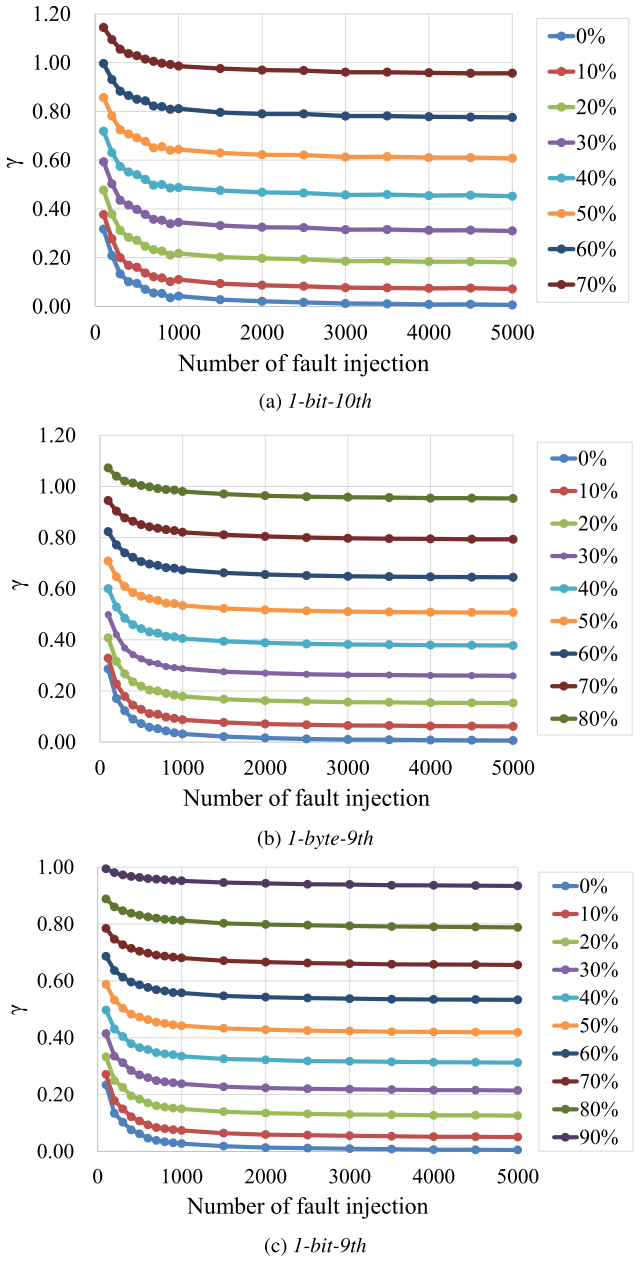
(b) *1-byte-9th*

(c) *1-bit-9th*

FIGURE 6. Influences of the fault model, the degree of protection and the number of fault injections on m'.

Secondly, given the fault model and the number of fault injections, the actual amount of information leakage decreases as the strength of protection increases, and is well less than the theoretical amount of information leakage. This result confirms to our expectation and demonstrates the effectiveness of different degrees of protection.

Thirdly, under the same number of fault injections and the same degree of protection, the actual amount of information



(a) *1-bit-10th*

(b) *1-byte-9th*

(c) *1-bit-9th*

FIGURE 7. **Influences of the fault model, the degree of protection and the number of fault injections on γ.**

leakage corresponding to different fault models is as follows: *1-bit-9th* > *1-byte-9th* > *1-bit-10th*. Therefore, the fault model *1-bit-9th* has the biggest threat to the AES circuit and efficient countermeasures targeting at the model should be developed.

2) SECURITY FACTOR

Fig.7 shows influences of the different fault models, the different degrees of protection and the different numbers of fault injections on the security factor. Contrary to the trend of actual information leakage amount, the curve of security factor shows a slight decline and becomes basically stable when the number of fault injections is 2000. Thus, 2000 can be regarded as the minimum number of fault injections and the minimum number of ΔZ samples required for Algorithm 2 when we simulate to evaluate the security of the AES-128 encrypted circuit under the *1-bit-10th*, *1-byte-9th* and *1-bit-9th* fault model. The value is larger than the theoretically minimum value 255, because some fault injections do not provide the required information.

When the fault model is fixed and the number of fault injections is greater than or equal to 2000, the impact of different degrees of protection on the security factor is very obvious. The security factor without protection is close to 0. The greater protection is, the greater the security factor is. 70%, 80% and 90% protection are sufficient for the fault models *1-bit-10th*, *1-byte-9th* and *1-bit-9th*, respectively.

The security factor can be used to clearly compare the security of a cryptographic circuits which take the countermeasures. Under the same fault model, the higher security factor means that the cryptographic circuit is more resistant to DFA. Besides, the security factor can be used to determine whether the security of a cryptographic circuit meets the requirement of a particular application scenario, thereby avoiding unnecessary resource overhead caused by excessive protection.

D. EVALUATION TIME

Fig.8 shows the variations in evaluation time as the number of fault injections increases under the *1-bit-10th*, *1-byte-9th* and *1-bit-9th* fault models. As the number of fault injection increases, the evaluation time for *1-bit-10th* is roughly stable, and for *1-byte-9th* and *1-bit-9th* linearly increases. This result shows the good scalability of the evaluation method.

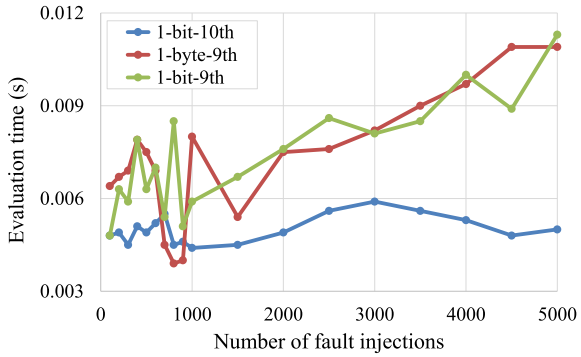


FIGURE 8. **Evaluation time under the fault models *1-bit-10th*, *1-byte-9th* and *1-bit-9th*.**

VII. CONCLUSION

This paper introduces a security quantitative evaluation method, which can be used to evaluate countermeasure designs against FIAs, based on the information theory. A security factor of a cryptographic circuit under specified fault models is obtained by comparing the theoretical amount of information leakage and the actual amount of information leakage. An extended cipher model is proposed

to numerically analyze the theoretical amount of information leakage, and an efficient approach is proposed to obtain the actual amount of information leakage. Experiments by applying the quantitative evaluation method to AES circuit validate feasibility, efficiency and scalability of the method.

In future, we will extend the quantitative evaluation method to other cipher models, for example, symmetrical Feistel Network. In addition to the case where faults are injected into the input of the encryption round, other fault injection positions should be carefully considered. For example, faults are injected between the substitution operation and the permutation operation.

REFERENCES

[1] A. Barenghi, L. Breveglieri, I. Koren, and D. Naccache, "Fault injection attacks on cryptographic devices: Theory, practice, and countermeasures," *Proc. IEEE*, vol. 100, no. 11, pp. 3056–3076, Nov. 2012.
[2] E. Biham and A. Shamir, "Differential fault analysis of secret key cryptosystems," in *Advances in Cryptology—CRYPTO*, B. S. Kaliski, Ed. Berlin, Germany: Springer, 1997, pp. 513–525.
[3] J.-M. Dutertre, J. J. A. Fournier, A.-P. Mirbaha, D. Naccache, J.-B. Rigaud, B. Robisson, and A. Tria, "Review of fault injection mechanisms and consequences on countermeasures design," in *Proc. 6th Int. Conf. Design Technol. Integr. Syst. Nanoscale Era (DTIS)*, Apr. 2011, pp. 1–6.
[4] Y. Monnet, M. Renaudin, and R. Leveugle, "Designing resistant circuits against malicious faults injection using asynchronous logic," *IEEE Trans. Comput.*, vol. 55, no. 9, pp. 1104–1115, Sep. 2006.
[5] S. H. Weingart, "Physical security devices for computer subsystems: A survey of attacks and defenses," in *Cryptographic Hardware and Embedded Systems—CHES*, Ç. K. Koç and C. Paar, Eds. Berlin, Germany: Springer, 2000, pp. 302–317.
[6] R. Karri, K. Wu, P. Mishra, and Y. Kim, "Fault-based side-channel cryptanalysis tolerant Rijndael symmetric block cipher architecture," in *Proc. IEEE Int. Symp. Defect Fault Tolerance VLSI Syst.*, Oct. 2001, pp. 427–435.
[7] P. Maistri and R. Leveugle, "Double-data-rate computation as a countermeasure against fault analysis," *IEEE Trans. Comput.*, vol. 57, no. 11, pp. 1528–1539, Nov. 2008.
[8] M. Bedoui, H. Mestiri, B. Bouallegue, and M. Machhout, "A reliable fault detection scheme for the AES hardware implementation," in *Proc. Int. Symp. Signal, Image, Video Commun. (ISIVC)*, Nov. 2016, pp. 47–52.
[9] M. Ayoob and W. Adi, "Fault detection and correction in processing AES encryption algorithm," in *Proc. 6th Int. Conf. Emerg. Secur. Technol. (EST)*, Sep. 2015, pp. 7–12.
[10] S. Sau, R. Paul, and K. Maity, "Correction of MEU errors in AES using multi bit errors correction technique," in *Proc. 2nd Int. Conf. Data Sci. Bus. Anal. (ICDSBA)*, Sep. 2018, pp. 167–173.
[11] R. Leveugle, "Early analysis of fault-based attack effects in secure circuits," *IEEE Trans. Comput.*, vol. 56, no. 10, pp. 1431–1434, Oct. 2007.
[12] R. Nyberg, J. Nolles, J. Heyszl, D. Rabe, and G. Sigl, "Closing the gap between speed and configurability of multi-bit fault emulation environments for security and safety-critical designs," in *Proc. 17th Euromicro Conf. Digit. Syst. Design (DSD)*, Aug. 2014, pp. 114–121.
[13] K. Bousselam, G. Di Natale, M.-L. Flottes, and B. Rouzeyre, "Evaluation of concurrent error detection techniques on the advanced encryption standard," in *Proc. 15th IEEE Eur. Test Symp.*, May 2010, p. 252.
[14] F. Benevenuti and F. L. Kastensmidt, "Evaluation of fault attack detection on SRAM-based FPGAs," in *Proc. 18th IEEE Latin Amer. Test Symp. (LATS)*, Mar. 2017, pp. 1–6.
[15] T. An, L. A. de Barros Naviner, and P. Matherat, "Evaluation of fault-tolerant composite field AES S-boxes under multiple transient faults," in *Proc. IEEE 11th Int. New Circuits Syst. Conf. (NEWCAS)*, Jun. 2013, pp. 1–4.
[16] X. Guo, D. Mukhopadhyay, C. Jin, and R. Karri, "Security analysis of concurrent error detection against differential fault analysis," *J. Cryptograph. Eng.*, vol. 5, no. 3, pp. 153–169, Sep. 2015. doi: 10.1007/s13389-014-0092-8.
[17] K. Sakiyama, Y. Li, M. Iwamoto, and K. Ohta, "Information-theoretic approach to optimal differential fault analysis," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 1, pp. 109–120, Feb. 2012.
[18] G. Piret and J.-J. Quisquater, "A differential fault attack technique against SPN structures, with application to the AES and KHAZAD," in *Cryptographic Hardware and Embedded Systems—CHES*, C. D. Walter, Ç. K. Koç, and C. Paar, Eds. Berlin, Germany: Springer, 2003, pp. 77–88.

QIANG LIU (M'14) received the Ph.D. degree from the Department of Electrical and Electronic Engineering, Imperial College London, London, U.K., in 2008, where he was a Research Associate with the Department of Computing, from 2009 to 2011. He is currently an Associate Professor with the School of Microelectronics, Tianjin University, with research interests in hardware security, VLSI design optimization, and reconfigurable computing.

BO NING received the B.Eng. degree from Tianjin University, Tianjin, China, in 2017. She is currently pursuing the master's degree with the School of Microelectronics, Tianjin University. Her current research interest includes hardware security.

PENGJIE DENG received the B.Eng. degree from Tianjin University, Tianjin, China, in 2015, and the master's degree from the School of Microelectronics, Tianjin University, in 2018. His research interest includes hardware security.