

A Framework for Evaluation and Analysis on Infection Countermeasures Against Fault Attacks

Jingyi Feng, Hua Chen, Yang Li, ZhiPeng Jiao, and Wei Xi

Abstract—Infection is a fault attack countermeasure that aims to destroy the dependency of the faulty ciphertexts on the secret key. However, current security evaluations on infection countermeasures are either tailored for the specific attack scenario, or not general enough to apply to various infection instances. They cannot come to the convincing results, let alone make comparison between different countermeasures. Based on information theory, this paper presents a generic evaluation framework that is feasible for various infection countermeasures and attack scenarios. The framework is constructed with the idea to separate the infection function from the unprotected cipher, yet consider the fault injection effect on the unprotected cipher in the infection function evaluation. Firstly, the security judging criteria for the infection function under different attack scenarios are personalized according to the injection-caused security loss of the unprotected cipher. Then, a universal method of security quantitative analysis on infection function is proposed with two important steps: the prior knowledge collection and the infection operation decomposition analysis. Because the analysis results of the simple infection operation can be reused within the infection function under various attack scenarios, the security quantifications are efficient. Based on this framework, the paper also reviews some existing infection countermeasures for their fault attack resistances. The result shows that our analysis can expose more infection vulnerabilities than previous works. Besides, the security quantification and judgment on these countermeasures give us a new insight into their security applicable scopes. They are instructive for the countermeasure selection when the implementation costs are very close. Further, the framework provides an efficient way to evaluate future infection countermeasures.

Index Terms—fault attack, infection countermeasure, security evaluation.

I. INTRODUCTION

IN recent years, implementation vulnerabilities rather than mathematical weaknesses have become significant threats to cryptographic devices. These vulnerabilities enable the adversary to observe the side-channel information leakage of the devices, perturb their normal behaviors and infer the secret

intermediates of the cipher. Among the existing implementation attacks, fault attacks (FA) [1] are powerful ones. By injecting faults into cryptographic devices with clock glitches, electromagnetic radiation or laser radiation, the attacks corrupt the normal encryption in the intermediate variable or flow sequence. Then, they acquire additional information from the faulty ciphertexts. So far, fault attacks have been successfully applied to various cryptosystems including AES [2], [3], [4], ECC [5], RC4 [6] and so on. Though the values of faulty ciphertexts are helpful for key retrieval, there still exist attacks that do not use them. For instance, ineffective fault attack (IFA) exploits the non-uniform fault models to reveal the value (distribution) of injected intermediates [7], [8]. Fault sensitivity analysis (FSA) exploits the critical intensity of the effective injection to identify the value of the processed data [9]. These attacks are also threatening to cryptographic devices.

Up to now, there are mainly two kinds of countermeasures against fault attacks: detection and infection. Both countermeasures can apply to the attacks that rely on the intermediate-oriented injection and the faulty ciphertexts exploitation. In the detection countermeasure, the cipher computation is first performed redundantly in the form of space, time or information, followed by a consistency check between the redundant and original cipher computation results [10], [11], [12]. If the results are different, the fault will be detected. Then, the countermeasure will stop its output to avoid the faulty ciphertext exploitation in fault attacks. However, if the adversary can bypass the consistency check with an additional fault injection, the detection countermeasure will be broken. To avoid the risk, Joye *et al.* proposed a new defensive strategy called infection [13]. By doubling the algorithm and scrambling the data paths of the two encryptions, the countermeasure corrupts the relationship between the secret key and the faulty output. Finally, it makes the faulty output useless in the original fault attack. As a result, there is no need for any consistency check. The ciphertext can always be returned. Infection function is the core component of the infection countermeasure. It decides how the data paths are scrambled. The early infection functions are constructed with deterministic operations. They were soon proved insecure because their scramble effects can be easily removed when the construction details are known. To address the problem, the later proposed countermeasures have tried to induce randomness into infection functions in various ways, including multiplicative masking [14], dummy computation [15], [16], system configuration parameters [17] and so on.

This work was supported by the National Key R&D Program of China under Grant 2018YFB0904900 and 2018YFB0904901. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Yiorgos Makris. (Corresponding author: Hua Chen.)

J. Feng and Z. Jiao are with TCA laboratory, Institute of Software, Chinese Academy of Sciences, Beijing 100190, China, and also with State Key Laboratory of Cryptology, P. O. Box 5159, Beijing 100878, China, and also with the University of Chinese Academy of Sciences, Beijing 100049, China (e-mail: fengjingyi, jiaozhipeng@tca.iscas.ac.cn).

H. Chen is with TCA laboratory, Institute of Software, Chinese Academy of Sciences, Beijing 100190, China (e-mail: chenhua@tca.iscas.ac.cn).

Y. Li is with The University of Electro-Communications, Chofu-shi, Tokyo 182-8585, Japan (e-mail: liyang@ice.uec.ac.jp).

W. Xi is with Electric Power Research Institute, China Southern Power Grid, Guangzhou, 510663, China (e-mail: xiwei@csg.cn).

A. Motivation and Related Works

Although random infection countermeasures overcome the weakness of detection countermeasures, their security still cannot be evaluated in a general and provable way. Most of the infection countermeasures are first evaluated internally by their designers. These evaluations only focus on some particular fault injection scenarios and base on some specific analytical strategies. Yet they ignore the diversity of fault attacks. As a result, many seemingly secure countermeasures are found vulnerable under other attack scenarios in later researches [18], [19], [20], [21]. Besides, the evaluation based on analytical strategies can hardly lead to a fair comparison between different countermeasures. Based on these facts, a natural question is: given an infection countermeasure, whether there exists a general security evaluation framework that is applicable to various fault injection scenarios and independent of the specific analytical strategy. Further, if the answer is yes, can the security be evaluated quantitatively?

In [19] and [21], Patranabis *et al.* have tried to make the evaluation more general from the perspective of information theory. They took the mutual information between the secret key and the difference of the output ciphertexts as the security evaluation metric. Then, they quantified the security of an AES infection implementation [16] formally under three different fault injection scenarios. However, their works only focus on this specific implementation and its specific infection function. They did not discuss how to generalize their methods to other infection countermeasures, especially for those with complicated infection functions. Therefore, gaps still exist in the infection countermeasure analysis and mutual information quantification. Besides, the evaluation in [19] and [21] targeted at the integrated cipher implementation. Once the unprotected cipher or the fault model changes, it always requires a new security quantification from scratch. Therefore, their method may not be the best choice for numerous evaluations under different attack scenarios. Furthermore, their quantification results reflect the combined effect of the infection together with the other parts of this implementation, including the unprotected cipher algorithm and some other defensive measures. Therefore, the security comparison between different infection countermeasures is still a problem.

In [22], Ghosh *et al.* have tried to separate the security analysis of the infection function from the analysis of the integrated cipher algorithm. They have proved that if the infected faulty ciphertexts are totally random, then no secret key can be retrieved from them. This work provides an identification criterion for the perfect secure infection countermeasure against all the faulty ciphertexts exploitation attacks. However, the criterion is too strict for the evaluation under each specific attack scenario. It is likely to result in false negative security judgments. Besides, this work did not mention how to quantify the randomness of the infected faulty ciphertexts.

B. Contribution and Organization

In this paper, we propose the first comprehensive security evaluation framework for infection countermeasures. The framework covers a large portion of fault attacks that

rely on the exploitation of faulty ciphertexts. It is feasible for both the existing and future infection functions. It is also suitable for security comparison between different infections. The framework is constructed from the point of information theory just as [19] and [21], but with the idea to separate the security requirement of the infection function from the integrated cipher implementation. The main principle is to formalize the fault injection effects on the unprotected cipher into the attack scenarios faced by the infection function, and take them into account in the evaluation. Another principle is to decompose the complicated analysis into multiple simple ones, and reuse their results for the efficient evaluation.

- Firstly, we construct a security judging criterion for infection functions based on the security requirement of the integrated cipher implementation and the security loss of unprotected cipher in the given attack scenario. In this case, the security boundaries are set more precise and personalized under different attack scenarios.
- Secondly, we put forward a universal method of security quantitative analysis with two important steps: the prior knowledge collection and the simple infection operation decomposition analysis. The analysis takes not only the infection output but also the attack-scenario-related information of the infection input as the prior knowledge. Therefore, it enables a thorough exploration of infection vulnerabilities. On the other hand, the analysis decomposes the infection function into a series of repetitive and simple infection operations that are related to the random bits. Therefore, the security quantification result of the infection function can be easily obtained through the information-theoretical analysis of the simple infection operations.
- Finally, we make security judgment on the infection function under the given attack scenario, according to the corresponding judging criterion and quantification result.

To verify the feasibility of the framework, we take the intermediate-oriented fault injections on AES-128 as the sample attack scenarios and make evaluations on the existing infection countermeasures. The result shows that most existing countermeasures are not as secure as expected. Through the analysis based on all possible prior knowledge, we find not only their well-known vulnerabilities, but also new flaws that have never been taken as threats. We summarize all these vulnerabilities in Table I and highlight the new ones in bold. This work confirms the comprehensiveness of our framework. On the other hand, the quantitative evaluation result gives us a new insight into the application of infection countermeasures. It is instructive for the countermeasure selection when the implementation costs are close.

The paper is organized as follows. Section II provides the necessary preliminaries on fault attacks and infection countermeasures. Section III presents the proposed evaluation framework. Section IV takes the fault injections on AES-128 as the sample attack scenarios to demonstrate the security judging criteria for infection functions. Then, the security quantification analyses on RIMBEN, ICDR, and LDRNM are detailed in Section V to VII, respectively. Section VIII

TABLE I: Infection countermeasure and security judgment (for FAs on AES-128)

Countermeasure	Vulnerability	Security Evaluation Result ^[†]
RIMBEN[17]	<ul style="list-style-type: none"> predictable Hamming weight of infection output predictable output of random 2-state switches 	insecure ^[‡] insecure for the deterministic, repeatable and completely diffused fault ^[§]
ICDR[15]	<ul style="list-style-type: none"> linearly dependent output bytes of MixColumns operation non-uniform differential distribution of Sbox in random infection under the repeatable fault model 	secure for the unrepeatable fault injection with the complete fault diffusion
LDRNM[22]	<ul style="list-style-type: none"> non-uniform distribution of infection outputs under the repeatable fault model distinguishable infection input for the incomplete diffusion 	secure for the unrepeatable fault injection
MM[14]	<ul style="list-style-type: none"> distinguishable power consumption 	secure
RR[16]	<ul style="list-style-type: none"> FA-vulnerable 1-bit judgment condition 	secure

^[†] The results here only cover the intermediate-oriented injections. They cover the attacks that rely on the exploitation of faulty ciphertexts.

^[‡] RIMBEN without 4-state switches. When the initial configurations of the fixed switches are open to the public.

^[§] RIMBEN without 4-state switches. When the initial configurations of the fixed switches are kept secret.

generalizes the evaluation and makes comparison between different countermeasures. Section IX concludes the paper and highlights the future work.

II. PRELIMINARIES

Notations

- C : the correct ciphertext
- ΔC : the output difference between the original and the redundant cipher computation, which is also the input of the infection function in most countermeasures
- $I(\Delta C)$: the output of the infection function
- ΔC_p : the prior knowledge of infection input ΔC that can be derived from the fault injection in the unprotected cipher. It records the number and relative location of zero (non-zero) bytes in ΔC . It also records whether ΔC is the same in different infections.
- ΔF : the difference between the infected and the fault-free (correct) ciphertext. For a series ΔF derived from the same infection input, ΔF_i denotes the one worked out from the i -th infection
- e : the induced fault, which is equal to the difference of the target intermediate before and after the fault injection
- \mathcal{K} : the distinguishable secret key involved in the fault propagation
- l_o : the information leakage of \mathcal{K} resulted from each injection
- $H(X)$: the information entropy of X measured in bits

Only a few countermeasures do not take the output difference ΔC as the infection input. They take a ΔC -related variable as their infection input. The prior knowledge of this variable can also be derived from the fault injection in the unprotected cipher. For simplicity, we generally denote ΔC as the infection input, ΔC_p as the prior knowledge of infection input, and $I(\Delta C)$ as the infection function output. For the exceptions, We will make another explanation.

A. Fault Attack

Fault attacks (FA) are an aggressive secret information retrieval method. In this attack, the adversary injects faults into cryptographic devices and makes the analysis on the faulty encryptions. The attacks are flexible owing to various injection targets, fault models and analytical strategies.

The fault injection can change either the intermediate variable or the flow sequence of the algorithm. Usually, only one injection is allowed during each encryption. In the intermediate-oriented injection, the induced fault e is described with its fault model in terms of width, location and value. The width shows the sphere of influence of the fault, such as bit-level, byte-level or word-level. The location and the value explain which part of the intermediate is corrupted and what difference the fault brings about, respectively. Both of them can either be known to the adversary or random unknown. Compared to the random unknown byte-oriented fault model, the specified known bit-oriented one requires more precise control on the fault injection. However, it allows an easier secret information retrieval analysis, especially when the fault diffusion in the ciphertext is incomplete.

So far, many traditional cryptanalysis methods have been borrowed by fault attacks to retrieve the secret key, including differential fault analysis (DFA) [23], algebraic fault analysis [24], integral fault analysis [25] and statistical fault analysis [26]. All these attacks rely on the exploitation of faulty ciphertexts. Among them, the optimal DFA [27] is the only one proposed from the information theoretical perspective. It makes full use of the information leakage about the fault model, the correct and faulty ciphertexts. Therefore, it reveals the strongest security threat of each injection. Base on this fact, we take the optimal DFA as a necessary attack scenario in our evaluation framework.

There are also attacks that do not use the values of faulty ciphertexts, such as safe-error attack (SEA) [28], ineffective fault attack (IFA) [7], [8] and fault sensitivity attack (FSA) [9]. These attacks rely on the information of the fault injection and focus on whether the injection has an effect on the encryption. They are less efficient than the attacks that exploit the values of faulty ciphertexts. However, they could perform against common countermeasures, such as detection and infection.

B. Infection Countermeasure

Infection is a fault attack countermeasure that aims to render the faulty ciphertext useless. The countermeasure consists of two parts: repetitive encryption and infection function. Fig. 1 illustrates the generic construction of infection countermeasures. The encryption is made twice with the same plaintext

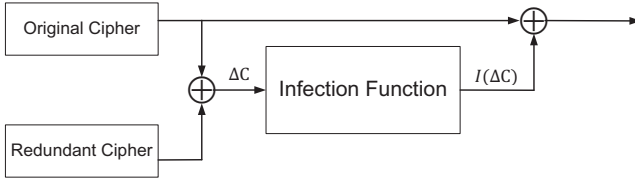


Fig. 1: Generic construction of infection countermeasures

and the same key. One is called the original cipher computation and the other one is called the redundant cipher computation. The countermeasure takes their output difference ΔC as the input of the infection function I . Then it masks the original cipher computation output with the infection function output $I(\Delta C)$. Let C denote the cipher computation output free of the fault. The infected output will either be $C \oplus I(\Delta C)$ or $C \oplus \Delta C \oplus I(\Delta C)$ depending on whether the fault is injected in the redundant or the original cipher computation. Infection functions are usually constructed with simple crypto structures and existing cryptographic modules. For $\Delta C = 0$, $I(0)$ should be equal to 0 to ensure the correctness of the fault-free encryption output. For $\Delta C \neq 0$, the infection function introduces a secret random number R to make the fault diffusion in the infected output unpredictable. In most of the existing infection countermeasures, R is updated every encryption.

$$I(\Delta C) = \begin{cases} 0, & \Delta C = 0 \\ F(\Delta C, R), & \text{otherwise} \end{cases} \quad (1)$$

Due to the randomness of infection functions, it is almost impossible to change a faulty ciphertext into the correct one through infection. Infection countermeasures cannot resist the attacks that do not use the values of faulty ciphertexts. They are also vulnerable to the double-fault injections that induce two identical faults in the original and redundant computations. Besides, the infection itself cannot resist the flow sequence-oriented injections that skip its execution. Therefore, the follow-up framework mainly focuses on the intermediate-oriented single fault injection scenarios. It evaluates how well the infections perform against the faulty ciphertexts exploitation attacks. An extended discussion about the evaluation under other fault injection scenarios is given briefly in Section VIII-C.

According to where the infections take place, we divide the integrated cipher implementations into two groups: built-in infection and built-out infection. The built-in infections are inserted between the iterations of the cipher rounds. They infect the subsequent intermediates immediately after the injection. The built-out infection is performed only once after the entire cipher computation. It enables the loose coupling between the countermeasure and the cipher algorithm. Because the built-in infection does not provide more resistance than the built-out one in terms of the worst case attack (the last round fault injection), the follow-up evaluations mainly focus on the cipher implementations with built-out infection.

III. INFECTION COUNTERMEASURE EVALUATION FRAMEWORK

In this section, first, we give an intuitive description of our evaluation framework and explain its application condition via the terminology definition. Then we detail how to construct the security judging criteria and how to make the quantitative analysis on the infection countermeasure.

A. Terminology and Design Consideration

1) *Terminology*: We show the generic evaluation framework for infection countermeasures in Fig. 2 and explain it as follows. Firstly, as shown in Fig. 1, the integrated *cipher implementation* is made up together by the *unprotected cipher* and the *infection function*. The structures of them are open to the public. The key of the cipher, the output difference of the original and the redundant cipher computation, and the random number of the infection function are kept secret. In this framework, the cipher implementation is allowed to encrypt the same plaintext many times. It means the difference between the infected and the correct ciphertext, ΔF , is available to the adversary. Secondly, the *fault injection* is characterized with the *target intermediate* and the *fault model*. Both of them are open to the public. In this framework, only one temporary intermediate-oriented injection is allowed during each encryption. Thirdly, the *evaluator* for infection countermeasures consists of two parts: *security judging criteria* and *security quantitative analysis*. Only if the quantification results satisfy the corresponding judging criteria, the countermeasure is considered secure. The framework currently only covers the attacks that rely on the exploitation of faulty ciphertexts.

2) *Design consideration*: The framework is constructed based on entropy analysis. Firstly, the evaluation result should have the same meaning for any infection countermeasure. The *evaluator* has to be independent of analytical strategies. According to the standard approach in information theory, Shannon's conditional entropy is a good choice for such evaluation metric. Given the attack parameters and the ciphertexts, it reflects the remaining uncertainty of the secret variable. Secondly, the entropy analysis is efficient for security quantifications. We can work out the uncertainty loss in the complicated infection function easily, by integrating the uncertainty losses in its more granular infection operations, without constructing a concrete key retrieval strategy. During the entire evaluation, we only have to analyze one infection operation. Then we can reuse its security quantification results many times, not only within the infection function but also under different fault injection scenarios. Last but not least, the entropy analysis always requires full use of prior knowledge. Therefore, it can ensure a more accurate security quantification and a more thorough vulnerability exploration of the infection countermeasure.

Thanks to the decoupling between the unprotected cipher and the built-out infection function, we can recover the infection input ΔC and the corresponding secret key in sequence during the fault attack. It means that the FA-resistance of the integrated cipher implementation is jointly decided by the *protective effect of the infection function* on ΔC and the

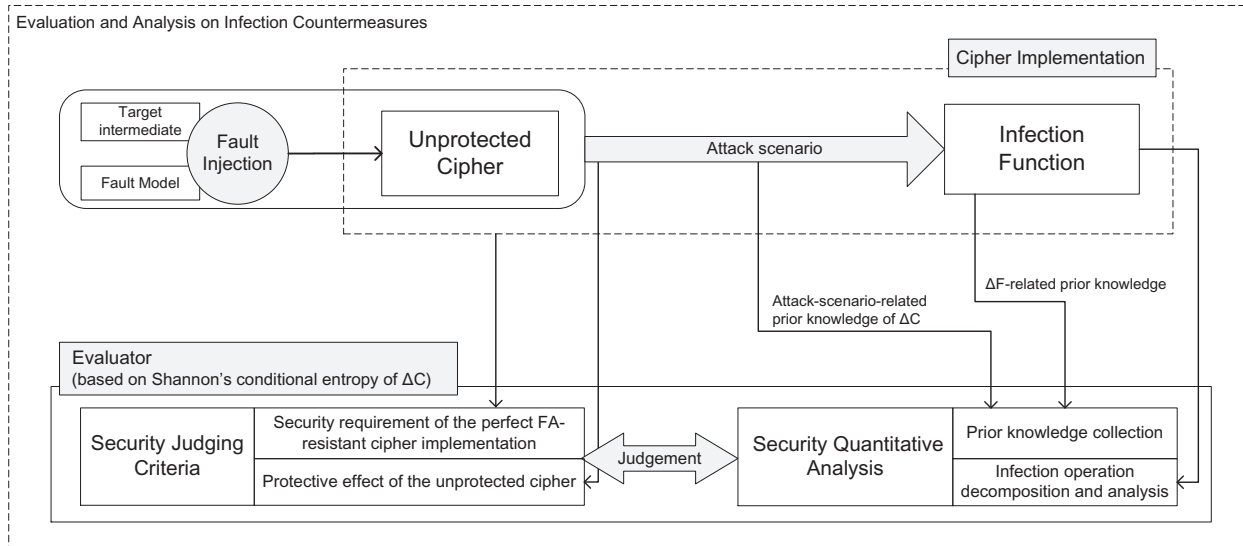


Fig. 2: Overview of security evaluation and analysis on infection countermeasure

protective effect of the unprotected cipher on the secret key. Given ΔF and the prior knowledge of infection input ΔC_p , the *protective effect of the infection function* indicates the remaining uncertainty of ΔC . Given ΔC and C , the *protective effect of the unprotected cipher* indicates the remaining uncertainty of the secret key. For simplicity, the framework focuses on the evaluation of the infection function rather than the integrated cipher implementation. The *fault injection* in the *unprotected cipher* decides the *attack scenario* faced by the infection function.

The *security judging criteria* define the lower bounded uncertainty of ΔC that should be maintained by secure infection functions under the given *attack scenario*. It is developed from the point that the ΔC recovered from a secure infection function cannot lead to an effective secret key distinguishment. First, we take the *security requirement of the perfect FA-resistant cipher implementation* as the principle. Then, we personalize the security requirement of the infection function according to the *protective effect of the unprotected cipher* on the secret key with the known ΔC .

As for *security quantitative analysis*, the goal is to work out the uncertainty of ΔC maintained by the given infection function. Different from the existing works which take ΔC a totally unknown variable, we conduct the *prior knowledge collection* in advance to figure out what can be used in the analysis. The collection is independent of the specific operations in the infection function. It focuses on the information of the infection input and output that can be directly derived from the *attack scenario* and ΔF . Then, consider the fact that most infection functions are constructed with a series of random-bit-related infection operations which are simple, identical and independent of each other. We put forward a universal security quantification method for the infection function, based on the *infection operation decomposition and analysis*. Finally, by integrating the analysis results according to the collected prior knowledge, we can obtain the remaining uncertainty of ΔC efficiently under the given attack scenario.

B. Security Judging Criteria

Security requirement of the perfect FA-resistant cipher implementation: In the fault attack on the cipher implementation, \mathcal{K} denotes the distinguishable secret key bytes involved in the fault propagation. Given C , ΔF and the attack-scenario-related prior knowledge ΔC_p , the remaining uncertainty of the secret key can be expressed as $H(\mathcal{K}|C, \Delta F, \Delta C_p)$. For the perfect FA-resistant cipher implementation,

$$H(\mathcal{K}|C, \Delta F, \Delta C_p) = H(\mathcal{K}). \quad (2)$$

Protective effect of the unprotected cipher on the secret key with the known ΔC : Consider the attack on the unprotected cipher with the known C and ΔC . The injection-caused information leakage of the secret key can be expressed as

$$l_0 = H(\mathcal{K}) - H(\mathcal{K}|C, \Delta C). \quad (3)$$

With the knowledge of the fault injection, the uncertainty of the secret keys, $H(\mathcal{K})$, can be derived through the fault propagation analysis on the unprotected cipher. Then, $H(\mathcal{K}|C, \Delta C)$ can be worked out from the differential distribution property of the nonlinear cipher components and the uncertainty of the induced fault e . For instance, for the unprotected ciphers which adopt the bijective Sboxes as the nonlinear components, $H(\mathcal{K}|C, \Delta C) \approx H(e)$ (refer to the information-theoretical analysis on the optimal DFA in [27] for details). The leakage l_0 reflects the contribution of the unprotected cipher to the FA-resistance of the integrated cipher implementation.

Judging criteria for the secure infection function: The knowledge of ΔC is necessary for the optimal DFA on the unprotected cipher. So we intuitively believe that, in the perfect FA-resistant cipher implementation, the uncertainty of ΔC guaranteed by the secure infection function has to be no less than the injection-caused information leakage of \mathcal{K} in the unprotected cipher. That is,

Proposition 1: If $H(\mathcal{K}) = H(\mathcal{K}|C, \Delta F, \Delta C_p)$, then

$$H(\Delta C|\Delta C_p, \Delta F) \geq l_0 = H(\mathcal{K}) - H(\mathcal{K}|C, \Delta C). \quad (4)$$

Because the original proposition is equivalent to its converse-negative proposition, we will prove its converse-negative proposition that:

If $H(K) > H(K|C, \Delta C) + H(\Delta C|\Delta C_p, \Delta F)$, then $H(K) > H(K|C, \Delta F, \Delta C_p)^1$.

Proof:

$$\begin{aligned} & H(K|C, \Delta C_p, \Delta F) \\ &= H(K, C, \Delta C_p, \Delta F) - H(\Delta C, C, \Delta C_p, \Delta F) \\ & \quad + H(\Delta C, C, \Delta C_p, \Delta F) - H(C, \Delta C_p, \Delta F) \quad (5) \\ &= H(K, C, \Delta C_p, \Delta F) - H(\Delta C, C, \Delta F) \\ & \quad + H(\Delta C|C, \Delta C_p, \Delta F) \end{aligned}$$

Since ΔC contains all the information in ΔC_p , then $H(K, C, \Delta C_p, \Delta F) \leq H(K, C, \Delta C, \Delta F)$.

$$\begin{aligned} & H(K|C, \Delta C_p, \Delta F) \\ & \leq H(K|C, \Delta C, \Delta F) + H(\Delta C|C, \Delta C_p, \Delta F) \quad (6) \\ & \leq H(K|C, \Delta C) + H(\Delta C|\Delta C_p, \Delta F) \end{aligned}$$

Therefore, if $H(K) > H(K|C, \Delta C) + H(\Delta C|\Delta C_p, \Delta F)$, then $H(K) > H(K|C, \Delta F, \Delta C_p)$. ■

The converse-negative proposition is proved. The original proposition is also true, which confirms our inference. Obviously, (4) is a necessary but not sufficient condition for the secure infection to achieve the perfect cipher implementation. But it offers a precise and flexible criterion to identify the insecure infection under each specific attack scenario.

On the other hand, if no information about ΔC except for its prior knowledge can be retrieved from the infection, the key retrieval attack on the unprotected cipher will be infeasible. Therefore, (7) is a sufficient condition for the secure infection.

$$H(\Delta C|\Delta C_p, \Delta F) = H(\Delta C|\Delta C_p) \quad (7)$$

C. Security Quantitative Analysis

The security quantitative analysis aims to figure out the expression of $H(\Delta C|\Delta C_p, \Delta F)$ for the given infection function under all possible attack scenarios. In this formula, ΔF can either be $I(\Delta C)$ or $\Delta C \oplus I(\Delta C)$, depending on where the fault is injected. When $\Delta F = I(\Delta C)$, ΔC can only be retrieved through the infection function inversion. When $\Delta F = \Delta C \oplus I(\Delta C)$, ΔC can also be retrieved through $I(\Delta C)$ prediction. Taking both cases into account, we first formalize the public information, including the attack scenario and ΔF into the prior knowledge of infection function. Then, we make the infection operation decomposition and analysis based on all possible prior knowledge.

1) Prior knowledge collection:

Attack-scenario-related prior knowledge of ΔC : ΔC is the secret intermediate protected by the infection function, but it is not totally unknown. With the knowledge of the fault injection and the unprotected cipher, its repeatability and the fault diffusion property can be easily revealed. They form the prior knowledge of infection function input, ΔC_p .

- **Repeatability:** Consider the case that the cipher implementation encrypts the same plaintext many times. If

¹Note that $H(K|C, \Delta F, \Delta C_p)$ can be no larger than $H(K)$. $H(K) > H(K|C, \Delta F, \Delta C_p)$ is the negative of $H(K) = H(K|C, \Delta F, \Delta C_p)$.

the adversary can inject M identical faults in these encryptions, then ΔC is repeatable in the corresponding infections. In this case, we substitute the ΔF in $H(K|C, \Delta C_p, \Delta F)$ with the infection differences $(\Delta F_1, \dots, \Delta F_M)$ derived from the identical ΔC . We can also work out the output difference of infection function as $\Delta F_i \oplus \Delta F_j$ ($i \neq j$).

- **Fault diffusion:** Fault diffusion describes the number and the relative location of the zero and non-zero bytes in ΔC . For the given unprotected cipher, it could be derived from the target intermediate and the fault model via the fault propagation analysis.

Although ΔC_p does not contribute to the secret key retrieval directly, it is useful for the retrieval of the exact value of ΔC . Moreover, ΔC_p has nothing to do with the concrete infection function. It can be reused in the evaluations of different infections.

ΔF -related prior knowledge of $I(\Delta C)$ and ΔC : In the cipher implementation, ΔF reveals either $I(\Delta C)$ or $\Delta C \oplus I(\Delta C)$. For the second case, it is necessary to check whether the distributions of the non-zero bytes in ΔC and $I(\Delta C)$ are consistent, according to the diffusion properties of the infection function. If not, the non-zero byte in ΔC (or $I(\Delta C)$) which is XORed with zero byte in $I(\Delta C)$ (or ΔC) can be revealed directly from ΔF . This operation further expands the prior knowledge of ΔC and $I(\Delta C)$.

2) *Infection operation decomposition and analysis:* The core components of the infection function are random-bit-related infection operations. They confuse the relationship between the infection function input and output. Usually, these operations appear more than once in each infection function. Their prior knowledge is of limited types. Therefore, we suggest to conduct security quantification as follows:

- Decompose the infection function into a series of identical random-bit-related operations.
- Take one of these operations as the example and enumerate all the possible types of its prior knowledge. For each type of prior knowledge, quantify the remaining uncertainty of the operation input (or output). We denote the quantification result as the protective effect of the infection operation.
- Deduce how the prior knowledge of the infection function decides the prior knowledge of the infection operations, according to the relationships between their inputs (or outputs). Then, integrate the protective effects of all the infection operations by type to construct the formula expression of $H(\Delta C|\Delta C_p, \Delta F)$.

A proper decomposition of the infection function is the key to efficient analysis. It follows three guidelines. Firstly, the infection operation should be independent of each other. Secondly, the operation should be as simple as possible to enable an easy quantitative analysis. Last but not least, there should exist available prior knowledge about this operation to support the analysis.

According to the prior knowledge of the infection function about ΔC and $I(\Delta C)$, the prior knowledge of the random-bit-related infection operation includes three aspects: input value,

output value, and output difference. In the infection function, different infection operations are connected with different bits of ΔC and $I(\Delta C)$. Therefore, their prior knowledge may belong to different types. For instance, if certain bits of $I(\Delta C)$ are known in advance, then the corresponding infection operations should be analyzed with the prior knowledge of the output values, while the others should not. For each infection operation, we take *PRIOR* to denote one possible type of its prior knowledge. *IN* and *OUT* denote its input and output to be protected. Then, its protective effect can be represented as $H(IN|PRIOR)$ (or $H(OUT|PRIOR)$). Because different infection functions adopt distinct infection operations, it requires the concrete analysis to work out $H(IN|PRIOR)$ (or $H(OUT|PRIOR)$) for different countermeasures.

According to the common construction of infection function, many operations in one infection function share the same type of prior knowledge. The probability distributions of their prior knowledge are almost the same. Therefore, we can quantify the protective effects of the infection operation based on a few types of prior knowledge and reuse the results within the infection function. However, the relationships between the infection function inputs (or outputs) and the infection operation inputs (or outputs) are distinct in different countermeasures. For each countermeasure, it requires the concrete analysis for a general understanding of how the prior knowledge of the infection function decides that of infection operations. To be specific, how many operations can work with type-1 prior knowledge, how many can work with type-2 prior knowledge, etc. Based on this analysis, the security quantification results of the infection operations can be further reused under various attack scenarios.

Let *num* denote the number of independent random infection operations in the given infection function. $PRIOR^i$ denotes the prior knowledge of the *i*-th infection operation ($i = 1, \dots, num$). IN^i and OUT^i denote its input and output, respectively. If the relationships between IN^i and ΔC , and the relationships between OUT^i and $I(\Delta C)$ are deterministic and linear, the uncertainty of ΔC maintained by this infection function can be integrated as

$$H(\Delta C|\Delta C_p, \Delta F) = \begin{cases} \sum_i H(IN^i|PRIOR^i), & \Delta C \text{ inversion} \\ \sum_i H(OUT^i|PRIOR^i), & I(\Delta C) \text{ prediction.} \end{cases} \quad (8)$$

So far, the quantification analysis only focuses on the infection function itself. It involves no concrete attack-scenario-related prior knowledge. Therefore, it can be applied to any attack scenario.

D. Security quantification and judgment

Finally, by configuring (8) with the specific prior knowledge of ΔC and $I(\Delta C)$ collected from the given attack scenario and ΔF , we get the protective effect of the infection function. Note that if the prior knowledge of ΔC or $I(\Delta C)$ is not fully utilized in the infection operation analysis, we can take the unused knowledge to filter the remaining ΔC candidates. In this case, the uncertainty of ΔC in (8) will be further reduced.

For the given attack scenario, if the security quantification result of $H(\Delta C|\Delta C_p, \Delta F)$ does not satisfy the requirement in (4), the infection function is theoretically insecure under this scenario. On the other hand, if it satisfies the requirement in (7), the infection function is secure for the scenario.

IV. EVALUATION OF INFECTION FUNCTIONS AGAINST THE FAULT ATTACK ON AES-128

To verify the feasibility of the framework, we take 8 different fault injections on AES-128 as the sample attack scenarios (refer to Table II) and evaluate several existing infection functions. In these attacks, the injection target and the width of the faults are referenced from [2], [3] and [4]. The location and the value of the faults are chosen according to the common settings. In this section, we figure out the judging criteria for the secure infection under each attack scenario. We also collect the attack-scenario-related prior knowledge of ΔC . In the following sections, we make quantification analysis and security judgment one by one on each specific infection function. Note that our evaluation results in Section III to Section VIII-A only cover the encryption intermediate-oriented single injection scenarios.

A. Security judging criteria

According to the differential distribution property of the bijective Sbox in AES-128 [27], the lower bounded uncertainty of ΔC that should be maintained by the secure infection function is

$$H(\Delta C|\Delta C_p, \Delta F) \geq l_o \approx H(\mathcal{K}) - H(e). \quad (9)$$

B. Fault diffusion in ΔC and the specific value of l_o

In the attack scenario (1.1) and (1.2), a bit-oriented fault is induced into the input of the 10-th round SubBytes (*SB*) [2]. Then, one byte in ΔC is corrupted and one key byte is involved in the fault propagation. It indicates that 15 zero bytes in ΔC are available for the infection function analysis and one secret key byte is distinguishable in the analysis of the unprotected AES-128, $H(\mathcal{K}) = 8$ bits. Since the value of the single bit-oriented fault must be 1, it brings no uncertainty to $H(e)$. The random location of the fault brings about $\log_2(\frac{128}{1}) = 7$ bits uncertainty to $H(e)$. According to (9), $l_o = 8 - 7 = 1$ bits in the attack scenario (1.1) and $l_o = 8$ bits in the scenario (1.2).

In the attack scenario (2.1), (2.2) and (2.3), a byte-oriented fault is induced into the input of the 9-th round MixColumns (*MC*) [3]. It results in 4 discrete non-zero bytes in ΔC and 4 distinguishable key bytes, $H(\mathcal{K}) = 32$ bits. The random location and the random value of the induced fault bring about $\log_2(\frac{128}{8}) = 4$ bits and 8 bits uncertainty to $H(e)$, respectively. Therefore, $l_o = 20, 24$ and 32 bits for the scenario (2.1), (2.2) and (2.3), respectively.

When further increase the width of the fault or bring forward the timing of injection, more ΔC bytes will be corrupted, and more key bytes will be involved. In the scenario (3.1), (3.2) and (3.3), a byte-oriented fault is induced before the 8-th round MixColumns (*MC*) [4]. It results in 16 non-zero bytes in ΔC

TABLE II: Fault diffusion in ΔC and Security judging criteria for infection function(against the FAs on AES-128)

Index	Attack Scenario						Fault Diffusion			$l_o = H(K) - H(e)$
	Ref	Target	Width	Location	Value	$H(e)$	ΔC	K	$H(K)$	
1.1	[2]	10R before SB	1-bit	Random	Known	7	1 Byte	1 Byte in K10	8	1
1.2				Known	Known	0				8
2.1	[3]	9R before MC	1-byte	Random	Random	12	4 Bytes	4 Bytes in K10	32	20
2.2				Known	Random	8				24
2.3				Known	Known	0				32
3.1	[4]	8R before MC	1-byte	Random	Random	12	16 Bytes	16 Bytes in K10 16 Bytes in K9	128	116
3.2				Known	Random	8				120
3.3				Known	Known	0				128

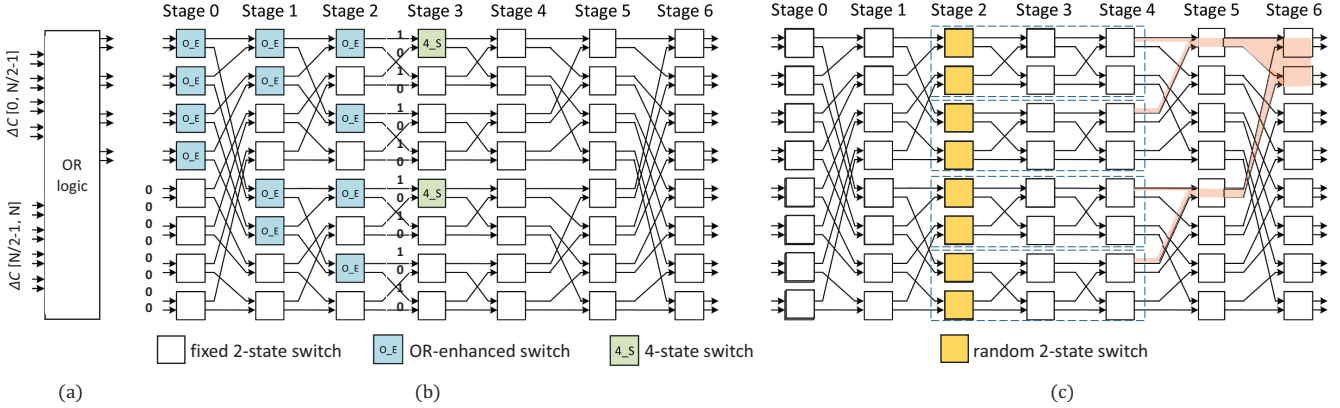


Fig. 3: Basic structure of infection function in RIMBEN (N=16). (a) Preprocessing structure. (b) HW balanced network (with 2 random 4-state switches). (c) Output confusion network (with random 2-state switches in stage 2).

and makes $H(K) = 128$ bits. Therefore, $l_0 = 116, 120$ and 128 bits for the scenario (3.1), (3.2) and (3.3), respectively.

V. QUANTIFICATION ANALYSIS ON RIMBEN

RIMBEN (Random Infection based on Modified Benes Network) [17] is a built-out infection countermeasure. As shown in Fig. 3, it employs a preprocessing OR logic and 2 modified benes networks in sequence to construct the infection function. The $N \times N$ benes network adopts a $(2 \log_2 N - 1)$ -stage symmetrical recursive structure with $\frac{N}{2}$ switches in each stage. In the first network, half of switches in stage 0 to $\log_2 N - 2$ are 2-state switches (refer to Fig. 4(a)). The control bit of the switch decides whether it performs straight or cross operation. The other half are OR-enhanced switches that perform OR operations. They, together with the preprocessing logic, make the input of the switches in stage $\log_2 N - 1$ equal to "101010..." for any $\Delta C \neq 0$. Therefore, the first network is also called HW-balanced network. The subsequent switches in RIMBEN are all 2-state switches. Most of them are configured with the fixed control bits that are randomly generated in the initialization process, except for the switches in one stage. This special stage is randomly selected from the second benes network in each infection. All the 2-state switches in this stage are configured with secret random control bits. This stage-based randomization can further confuse the infection output. As an additional option, the infection function can upgrade the 2-state switches in the middle stage of the first network to

random 4-state switches, by adding more random control bits to enable the lower/upper broadcast operations.

A. Analysis on RIMBEN without 4-state Switches

ΔF -related prior knowledge: Owing to the HW-balanced network that results in the identical intermediate, the repeatability of ΔC makes no additional contribution to the infection vulnerability analysis. Moreover, it makes the infection function non-invertible. ΔC can only be retrieved through $I(\Delta C)$ prediction under the condition that $\Delta F = \Delta C \oplus I(\Delta C)$. When the fault diffusion in ΔC is incomplete, the distributions of non-zero bytes in ΔC and $I(\Delta C)$ are different. Some certain bytes in $I(\Delta C)$ can be revealed directly from ΔF .

Infection operation decomposition and analysis: In RIMBEN without 4-state switches, the random 2-state switches in the random stage are the smallest infection operations related to the secret random bits. Fig. 4(b) illustrates the transition probability between the switch input (IN_1, IN_2) and output (OUT_1, OUT_2) under different cases. Consider the fact that all the control bits in RIMBEN are randomly generated, the values of the known input and output of the random 2-state switches obey uniform probability distribution. The protective effect on its output can be concluded as (10). In this equation, the conditional entropy is smaller than $H(OUT_1, OUT_2)$ or $H(OUT_1)$, which reveals a new vulnerability of RIMBEN.

$$\begin{aligned}
 & H(OUT_1, OUT_2 | PRIOR) \\
 &= \begin{cases} \frac{1}{2}, & PRIOR = (IN_1, IN_2) \\ 0, & PRIOR = (IN_1, IN_2, OUT_2). \end{cases} \quad (10)
 \end{aligned}$$

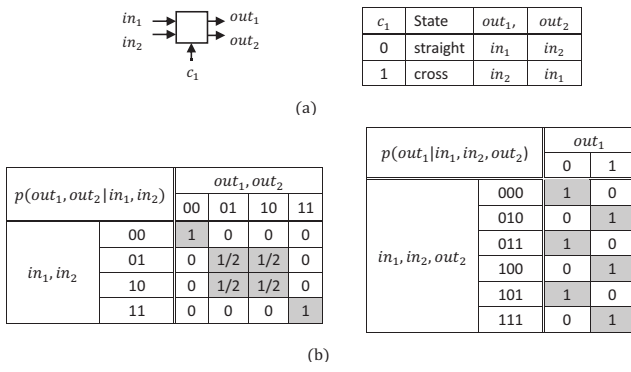


Fig. 4: 2-state switch in RIMBEN. (a) Control logic of 2-state switch. (b) Transition probability of random 2-state switch.

Because it is not specified in [17] whether the initial configurations of the fixed 2-state switches are made public or kept secret, we make the analysis under both conditions.

1) *When the initial configurations of the fixed 2-state switches are open to the public:* the relationship between the constant intermediate "101010..." of the first network and the inputs of the random 2-state switches is clear. The prior knowledge of the switch outputs can also be derived from the recovered bytes in $I(\Delta C)$. In this case, the vulnerable random 2-state switches can lead to a new attack on RIMBEN². According to the recursive topology of the network, the prior knowledge of the random 2-state switches changes with the random stage. For illustration, we take $|\Delta C|$ as the bit length of the non-zero bytes in ΔC . It is equal to the length of the unknown bits in $I(\Delta C)$.

- Consider the case that the unknown bits in $I(\Delta C)$ are continuous between the $(i \cdot |\Delta C| + 1)$ -th and the $((i + 1) \cdot |\Delta C|)$ -th bit, where $i \in [0, \frac{N}{|\Delta C|}]$. If the random 2-state switches are located between the stage $\log_2 |\Delta C|$ and $2 \log_2 N - 2 - \log_2 |\Delta C|$, then each random switch has at most one unknown output bit. So, there are $|\Delta C|$ random 2-state switches working with the prior knowledge (IN_1, IN_2, OUT_2) . (For instance, in Fig. 3c, the 4 continuous output bits marked in red are derived from 4 different 3-stage subnetworks highlighted with dash boxes. It indicates that any two output bits marked in red cannot come from the same random switches in these 3 stages.)
- If the unknown bits in $I(\Delta C)$ are not continuous, or the random switches are located in other stages, there are

²For each random 2-state switch, we first derive all its input bits from "101010...", and some of its output bits from the recovered bytes in $I(\Delta C)$. Then, we retrieve its unknown output bits based on the non-zero table items (marked in gray) in Fig. 4(b). To be specific, we select the row index of the table according to the derived bits, then find the non-zero table items in this row, and finally take the column indexes of these items as the candidates of the unknown output bits. For instance, if the switch input is "01" and one of its output bit is "0", then we find the row "010" in the right table and recover another output bit as "1". If the switch input is "01" and there is no prior knowledge of its output, then the output can either be "01" or "10", according to the left table. Finally, we transform the outputs of random 2-state switches into the bits in $I(\Delta C)$ and work out the candidate of ΔC as $I(\Delta C) \oplus \Delta F$.

at most $\frac{|\Delta C|}{2}$ random 2-state switches working with the prior knowledge (IN_1, IN_2) and the unknown outputs.

In RIMBEN, the random stage configured with random 2-state switches follows a uniform distribution. According to whether the unknown bits of infection function output are continuous, we summarize the upper bound uncertainty of ΔC as (11), in which *stage* denotes the random stage.

$$\begin{aligned}
 H(\Delta C | \Delta C_p, \Delta F) &= \sum_{stage} P(stage) \sum_{N/2} H(OUT_1, OUT_2 | PRIOR) \\
 &\leq \begin{cases} \frac{2 \log_2 |\Delta C|}{2 \log_2 N - 1} \cdot \frac{|\Delta C|}{2} \cdot \frac{1}{2} + 0, & \text{continuous} \\ \frac{2 \log_2 N - 1}{2 \log_2 N - 1} \cdot \frac{|\Delta C|}{2} \cdot \frac{1}{2}, & \text{otherwise} \end{cases} \quad (11)
 \end{aligned}$$

2) *When the initial configurations of the fixed 2-state switches are kept secret:* we cannot obtain enough prior knowledge about the random 2-state switches to support the further analysis. In this case, we decompose the infection function into more coarse-grained operation with clear input and output characteristics. The operation is the permutation network that consists of all the elements in RIMBEN after the HW-balanced process. The operation take "101010..." as input IN . It indicates the Hamming weight of the operation output OUT is $\frac{N}{2}$. Then we separate the operation output into two parts: OUT_k denotes the recovered bits and OUT_u denotes the others. Let $HW(\cdot)$ denote the Hamming weight of the variable. The protective effect of the operation can be derived as (12), in which I_{max} is the maximum value of $HW(OUT_u)$ that can be reached.

$$\begin{aligned}
 H(OUT_u | PRIOR) &\leq H(OUT_u | HW(OUT_u)) \\
 &= \sum_{i=0}^{I_{max}} P(i) H(OUT_u | HW(OUT_u) = i) \quad (12)
 \end{aligned}$$

For RIMBEN, OUT_k and OUT_u in (12) correspond to the recovered and unrecovered bytes in $I(\Delta C)$, respectively. $I_{max} = \min\{\frac{N}{2}, |\Delta C|\}$. Consider the fact that all the control bits in RIMBEN are randomly generated. Each output bit of RIMBEN obeys the identical probability distribution. Let $C(a, b)$ denote the number of b -combinations in a set of a elements. Then, $H(OUT_u | HW(OUT_u) = i) = \log_2 C(I_{max}, i)$. $P(HW(OUT_u) = i) = \frac{C(\frac{N}{2}, i) C(\frac{N}{2}, i) C(\frac{N}{2}, I_{max} - i)}{C(N, I_{max})}$. The uncertainty of ΔC can be worked out as

$$\begin{aligned}
 H(\Delta C | \Delta C_p, \Delta F) &= H(OUT_u | PRIOR) \\
 &\leq \sum_{i=0}^{I_{max}} \frac{C(\frac{N}{2}, i) C(\frac{N}{2}, i) C(\frac{N}{2}, I_{max} - i)}{C(N, I_{max})} \log_2 C(I_{max}, i) \quad (13)
 \end{aligned}$$

To sum up, the random 2-state switch is not a good choice for infection operation, especially when its inputs are known. Its predictable outputs can lead to a great uncertainty loss to ΔC . Besides, the predictable Hamming weight of the unknown bytes in $I(\Delta C)$ is another significant vulnerability of the countermeasure. However, the internal evaluation by the designers ignored all the prior knowledge about the random infection operations. It leads to an overestimation of the security of RIMBEN.

So far, our analysis is all based on the single infection. C_p

does not contain the knowledge of the repeatability of ΔC . For the repeatable infections with the identical ΔC , the protective effect of the infection function can be derived from (11) and (13) as

$$\begin{aligned} H(\Delta C|\Delta C_p) - H(\Delta C|\Delta C_p, \Delta F_1, \dots, \Delta F_M) \\ = M \cdot (H(\Delta C|\Delta C_p) - H(\Delta C|\Delta C_p, \Delta F)) \end{aligned} \quad (14)$$

B. Security Quantification and Judgment

According to the prior knowledge of ΔC and the security judging criteria in Section IV, the security of RIMBEN without 4-state switches is quantified and judged as follows.

1) *When the initial configurations of the fixed switches are open to the public:*

- For the scenario (1.1) and (1.2) with 1 non-zero byte in ΔC : $|\Delta C| = 8$ bits and the unknown bits in $I(\Delta C)$ are continuous. According to (11), $H(\Delta C|\Delta C_p, \Delta F) \leq 0.92$ bits. It is far from the security requirements (1 and 8) under the unrepeatable fault model.
- For the attack scenario (2.1), (2.2) and (2.3) with 4 non-zero bytes in ΔC : $|\Delta C| = 32$ bits and the unknown bits in $I(\Delta C)$ are not continuous. $H(\Delta C|\Delta C_p, \Delta F) \leq 16$ bits. It is not enough for the security requirements (20, 24 and 32) under the unrepeatable fault model.
- For the attack scenario (3.1), (3.2) and (3.3) with 16 non-zero bytes in ΔC : $|\Delta C| = 128$ bits and the unknown bits in $I(\Delta C)$ are continuous. $H(\Delta C|\Delta C_p, \Delta F) \leq 34.47$ bits. It is far from the security requirements (116, 120 and 128) under the unrepeatable fault model.

2) *When the initial configurations of the fixed switches are kept secret:*

- For the attack scenario (1.1) and (1.2): $|\Delta C| = 8$ bits. According to (13), $H(\Delta C|\Delta C_p, \Delta F) \leq 5.49$ bits. It is larger than 1 but smaller than 8. The countermeasure cannot resist the unrepeatable deterministic fault model.
- For the attack scenario (2.1), (2.2) and (2.3): $|\Delta C| = 32$ bits. $H(\Delta C|\Delta C_p, \Delta F) \leq 28.63$ bits. It is larger than 20 and 24, but smaller than 32. The countermeasure cannot resist the unrepeatable deterministic fault model.
- For the attack scenario (3.1), (3.2) and (3.3): $|\Delta C| = 128$ bits. $H(\Delta C|\Delta C_p, \Delta F) \leq 60.31$ bits. It is far from the security requirements (116, 120 and 128) under the unrepeatable fault model.

Above all, RIMBEN with public initial configurations does little to the security enhancement. It cannot reach the security requirements under the unrepeatable fault model, let alone the repeatable fault model. RIMBEN with secret initial configurations shows better resistance than the former one, especially for the scenarios with small $|\Delta C|$ and the random fault model. However, it cannot resist the deterministic fault. Since $H(\Delta C|\Delta C_p, \Delta F)$ is always smaller than $H(\Delta C|\Delta C_p)$, RIMBEN can hardly resist the repeatable fault injections.

C. Discussion

The case study on RIMBEN shows that different from the existing evaluations, the framework can apply to almost all the

intermediate-oriented attack scenarios. Firstly, the comprehensive collection and utilization of the prior knowledge make the evaluation result much closer to the real FA-resistance of the countermeasure. Then, the infection operation is analyzed based on all possible forms of prior knowledge. It enables efficient security quantifications under different attack scenarios. Besides, the analysis is helpful to construct efficient analytical strategies. For instance, in (12), $H(OUT_u|HW(OUT_u) = 0) \leq H(OUT_u|HW(OUT_u) = i, i \neq 0)$. By selecting the case that the Hamming weight of the unknown bytes in $I(\Delta C)$ is 0, we can recover ΔC more efficiently than in other cases.

Consider the RIMBEN with 4-state switches. The random 4-state switch only takes "10" as the input. Then, it outputs "00", "01", "10" and "11" with the equal probabilities. Therefore, t such switches will result in $2t$ unknown input bits for random 2-state switches, and $2t+1$ possible Hamming weight of $I(\Delta C)$. They reduce the available prior knowledge for the analysis. Given the information above, the FA-resistance of such RIMBEN can also be quantified under our framework.

VI. QUANTIFICATION ANALYSIS ON ICDR

ICDR (Infective Computation and Dummy Rounds) is a built-in infection countermeasure [15]. It consists of three different kinds of round computations: the original cipher computations, the redundant cipher computations and the dummy computations. The dummy computations are randomly inserted between the others except the one in the last round. Let (R_0, R_1, R_2) and (C_0, C_1, C_2) denote the corresponding round states and round intermediates, respectively. RF denotes the round function of the cipher. Algorithm 1 shows how ICDR is applied to an S-P network.

Algorithm 1: ICDR in LatinCrypt 2012[15]

Input: Plaintext P , round key k_i for $i \in \{1, \dots, n\}$, dummy round parameters (β, k_0)

Output: Ciphertext $C = \text{BlockCipher}(P, K)$

- 1 Original cipher state $R_0 \leftarrow P$, Redundant cipher state $R_1 \leftarrow P$, Dummy state $R_2 \leftarrow \beta$;
- 2 $C_0 \leftarrow 0, C_1 \leftarrow 0, C_2 \leftarrow \beta, i \leftarrow 1$;
- 3 **while** $i \leq 2n$ **do**
- 4 $\lambda \leftarrow \text{RandmBit}()$; /* $\lambda = 0$ implies a dummy round */
- 5 $\kappa \leftarrow (i \wedge \lambda) \oplus 2(\neg \lambda)$;
- 6 $\zeta \leftarrow \lceil i/2 \rceil \lambda$; /* ζ is actual round counter, 0 for dummy round */
- 7 $R_\kappa \leftarrow \text{RF}(R_\kappa, k_\zeta)$;
- 8 $C_\kappa \leftarrow R_\kappa \oplus (C_2 \oplus \beta)$; /* infect C_κ to propagate a fault */
- 9 $\mu \leftarrow \lambda(\neg(i \wedge 1)) \cdot \text{SNLF}(C_0 \oplus C_1)$; /* check if i is even */
- 10 $R_2 \leftarrow R_2 \oplus \mu$;
- 11 $R_0 \leftarrow R_0 \oplus \mu$;
- 12 $i \leftarrow i + \lambda$;
- 13 $R_0 \leftarrow R_0 \oplus \text{RF}(R_2, k_0) \oplus \beta$;
- 14 **return** R_0

ICDR is not in accordance with the general construction in Fig. 1 strictly. When the fault is injected between the penultimate and the last dummy computations, the infection effect of ICDR can be classified into two types:

1) The first one is the deterministic and non-diffused infection shown in step 9-11. It takes the deterministic nonlinear function SNLF in finite field $\mathbf{GF}(2^8)$ as the infection function.

Once the fault is injected, the deterministic infections will be executed in the subsequent original cipher rounds. Let ΔC^i denote the difference between the original and the redundant cipher intermediates in the i -th round. Then, $\text{SNLF}(\Delta C^i)$ is the deterministic mask of the i -th original cipher state. It also masks the current dummy state.

2) The second one is the random and diffused infection shown in step 13. It takes the difference between the dummy computation result and its reference value β as the infection output. In this infection, the variables β and k_0 are randomly selected and kept secret. They satisfy $\beta = \text{RF}(\beta, k_0)$. Therefore, the infection function can be represented as $\text{RI}(\Delta_\Sigma) = \text{RF}(\beta, k_0) \oplus \text{RF}(\beta \oplus \Delta_\Sigma, k_0)$, where Δ_Σ is the sum of the deterministic masks of different original cipher states. $\Delta_\Sigma = \sum_i \text{SNLF}(\Delta C^i)$.

Above all, the ciphertext is masked with $\text{SNLF}(\Delta C^n) \oplus \text{RI}(\Delta_\Sigma)$. The deterministic infection does not introduce randomness into the cipher state. Therefore, $\text{SNLF}(\Delta C^n)$ can be taken as a part of the output difference between the original and the redundant cipher computation, ΔC . The security of the countermeasure relies on the random infection. Because the random infection is executed only once at the end of the countermeasure, the built-in infection degrades into the built-out one. In this case, the difference between the correct and the infected ciphertext is $\Delta F = \Delta C \oplus \text{RI}(\Delta_\Sigma)$.

A. Analysis on ICDR with one built-out random infection

Attack-scenario-related prior knowledge: RI corresponds to the infection function I in the generic infection construction (Fig. 1). It takes $\Delta_\Sigma = \sum_i \text{SNLF}(\Delta C^i)$ rather than $\Delta C = \Delta C^n \oplus \text{SNLF}(\Delta C^n)$ as input. Because SNLF does not change the fault diffusion, the prior knowledge of ΔC is consistent with the study in Section IV. On the other hand, ΔC^i is the difference between the original and the redundant cipher intermediate of the i -th round. The prior knowledge of Δ_Σ in terms of zero (non-zero) bytes can also be worked out through the fault propagation analysis on AES-128. We collectively denote the prior knowledge of ΔC and Δ_Σ as ΔC_p .

ΔF -related prior knowledge: When the fault diffusion in ΔC is incomplete, the number of non-zero bytes in Δ_Σ is no smaller than that in ΔC . Besides, RI inherits the diffusion property of the original round function. It further makes the distributions of non-zero bytes in ΔC and $\text{RI}(\Delta_\Sigma)$ different. As a result, some bytes in ΔC and $\text{RI}(\Delta_\Sigma)$ can be revealed directly from $\Delta F = \Delta C \oplus \text{RI}(\Delta_\Sigma)$. The number of known bytes in $\text{RI}(\Delta_\Sigma)$ is equal to the number of zero bytes in ΔC .

Infection operation decomposition and analysis: The construction of the random infection function is completely decided by the cipher algorithm. For AES-128, $\text{RI}(\Delta_\Sigma) = \text{MC}(\text{SR}(\text{SB}(\beta) \oplus \text{SB}(\beta \oplus \Delta_\Sigma)))$. The differential SubBytes $\text{SB}(\beta) \oplus \text{SB}(\beta \oplus \Delta_\Sigma)$ works directly on the random number β . It can be decomposed into 16 independent random-bit-related infection operations, which are represented as $\Delta S_X(IN) = S(X) \oplus S(IN \oplus X)$. In this operation, S is the Sbox. IN is a secret input byte in Δ_Σ . X is a secret random byte in β that changes with infections. Obviously, the operation with zero input has no protective effect on its output. When the

operation input and output are both non-zero and unknown, its protective effect can be concluded as:

- Based on a single infection, no prior knowledge of this operation can be obtained for the further analysis.
- Based on M infections with the identical Δ_Σ , we denote the output difference of this operation between the 1-st and the j -th infection as $\text{OUT}_{1j} = \Delta S_{X_1}(IN) \oplus \Delta S_{X_j}(IN)$ ($j = 2, \dots, M$). It can be obtained from $\Delta F_1 \oplus \Delta F_j$, through the inversion of MC and SR . According to the non-uniform differential distribution of the AES Sbox, a random candidate combination of (X_1, IN) can result in the given difference OUT_{1j} with a probability³ of $\frac{127}{255}$. Therefore, the prior knowledge of OUT_{1j} can be used to distinguish the (X_1, IN) candidates⁴. The non-uniform differential distribution of the Sbox exposes a new vulnerability of ICDR under the repeatable fault model. It can lead to the retrieval of $\Delta S_{X_1}(IN)$, $\text{RI}(\Delta_\Sigma)$ and finally ΔC . Here we derive the remaining uncertainty of $\Delta S_{X_1}(IN)$ as

$$\begin{aligned} & H(\Delta S_{X_1}(IN) | \text{PRIOR}) \\ & \leq H(X_1, IN | \text{OUT}_{12}, \dots, \text{OUT}_{1M}) \quad (15) \\ & \leq 8 + (M - 1) \log_2 \frac{127}{255}. \end{aligned}$$

MixColumns (MC) connects the infection operation outputs with the infection function output. It consists of 4 independent mc operations. Each mc takes the outputs of 4 infection operations as its input. Then, it outputs 4 linearly dependent bytes and reserves them in the same column of $\text{RI}(\Delta_\Sigma)$. For the i -th mc related to the i -th column, let O^i denote its known output bytes in $\text{RI}(\Delta_\Sigma)$. Let I^i denote the non-zero bytes in Δ_Σ that form its input. $|O^i|$ and $|I^i|$ denote the byte length of O^i and I^i , respectively. According to the diffusion property of mc , $|O^i|$ and $|I^i|$ decide how many infection operations can work with the above mentioned prior knowledge. For instance, if $|I^i| < |O^i|$, we can recover the non-zero input bytes of the i -th mc from its known output bytes. It means no infection operation works without the prior knowledge of the output value. For general cases, given $|O^i|$ and $|I^i|$, there are at most $\max\{|I^i| - |O^i|, 0\}$ infection operations working with the non-zero inputs and unknown outputs. The other infection operations do not contribute to the security of the infection function.

For ICDR based on AES-128, the remaining uncertainty of $\text{RI}(\Delta_\Sigma)$ is the sum of the uncertainty of $\Delta S_X(IN)$ in each

³To study the dependence between the prior knowledge OUT_{1j} and the infection operation input (X_1, IN) , we construct an advanced differential distribution table for the 8-bit Sbox in AES. The table has $256 * 255$ rows and 256 columns. Each row corresponds to a possible combination of (X_1, IN) . We denote it as (x, in) , where $in \neq 0$. Each column corresponds to a possible value of $\Delta S_x(in) \oplus \Delta S_{x'}(in)$ for any x, x' and $in \neq 0$. We denote it as out . The cell in row (x, in) , column out records the amount of different x' that satisfies $out = \Delta S_x(in) \oplus \Delta S_{x'}(in)$. It has been worked out in the previous research that, for every non-zero in , the 256 possible values of x can only result in 127 different values of $\Delta S_x(in) = S(x) \oplus S(in \oplus x)$. Therefore, for every $(x, in \neq 0)$, we can get no more than 127 different out from 256 possible values of x' . It means half of cells in each row of the table are non-zero. A random candidate of (X_1, IN) can result in the given OUT_{1j} with a probability no larger than $\frac{127}{255}$.

⁴For the attack with M repeatable injections, if all the cells in row (x, in) , column OUT_{1j} ($j = 2, \dots, M$) are non-zero, then (x, in) can be taken as a candidate of (X_1, IN) .

infection operation. Since $\Delta C = \Delta F \oplus \text{RI}(\Delta \Sigma)$, the protective effect of the infection function can be worked out as

$$\begin{aligned} & H(\Delta C | \Delta C_p, \Delta F_1, \dots, \Delta F_M) \\ & \leq \sum_{i=1}^4 \max\{|I^i| - |O^i|, 0\} \cdot (8 + (M-1) \log_2 \frac{127}{255}). \end{aligned} \quad (16)$$

To sum up, when the fault diffusion in ΔC is incomplete, the diffusion property of MixColumns leads to the major uncertainty loss of ΔC . It is consistent with the studies in [18] and [16]. Meanwhile, the non-uniform differential distribution of Sbox is another significant vulnerability. It is first found in this paper. It provides evidence that even if the fault diffusion in ΔC is complete, the countermeasure is still not as secure as expected for the repeatable fault model.

B. Security Quantification and Judgment

For the speciality of ICDR infection input, we reanalyze the fault propagation in AES-128 to work out the number of non-zero bytes in $\Delta \Sigma$. We also work out the number of known bytes in $\text{RI}(\Delta \Sigma)$ according to the knowledge of zero bytes in ΔC . Then, we make the security quantification and judgment.

- For the attack scenario (1.1) and (1.2) with 1 non-zero byte in ΔC : there are at least 3 known bytes in each column of $\text{RI}(\Delta \Sigma)$ and at most 1 non-zero bytes in $\Delta \Sigma$ that are related to this column. According to (16), $H(\Delta C | \Delta C_p, \Delta F) = 0$ bit. The infection function is of no protective effect on ΔC .
- For the attack scenario (2.1), (2.2) and (2.3) with 4 non-zero bytes in ΔC : there are at least 3 known bytes in each column of $\text{RI}(\Delta \Sigma)$ and at most 2 non-zero bytes in $\Delta \Sigma$ that are related to this column. According to (16), $H(\Delta C | \Delta C_p, \Delta F) = 0$ bit. The infection function is of no protective effect on ΔC .
- For the attack scenario (3.1), (3.2) and (3.3) with 16 non-zero bytes in ΔC : the fault diffusion in ΔC is complete. There is no known byte in each column of $\text{RI}(\Delta \Sigma)$ and there are 4 non-zero bytes in $\Delta \Sigma$ related to this column. According to (16), $H(\Delta C | \Delta C_p, \Delta F) = 128$ bits. It is not smaller than 116, 120 and 128. The countermeasure may be secure for the unrepeatable fault injection. However, ΔC can be uniquely identified when the fault injections can be repeated for 17 times.

Above all, ICDR with a built-out random infection can resist neither the attack with incomplete fault diffusion nor the one with repeatable injections in theory. However, thanks to the randomly inserted dummy rounds, it reduces the success rate of the repeatable fault injections. The attack based on the repeatable ΔC is sometimes infeasible in practice.

C. Discussion

Things are complex when the fault is injected before the penultimate dummy computation. The deterministic and the additional random infections (refer to Algorithm 1, step 7-8) appear alternately. The built-in infection countermeasure is no longer equivalent to the built-out one. The built-in random infections make ΔC^n and $\Delta \Sigma$ change under the

repeatable fault injections. They destroy the prior knowledge in (15). Therefore, the cipher implementation requires at least 2 random infections to resist the attack based on the repeatable injections.

VII. QUANTIFICATION ANALYSIS ON LDRNM

LDRNM (Linear Diffusion and Randomized Nonlinear Mixing) is a built-out infection countermeasure proposed in ACISP 2015 for AES-128 [22]. The infection function first employs a deterministic linear diffusion function \mathcal{D} to disperse the fault into most output bits. Then, it modifies a bent function called NIMX to construct the nonlinear mixing function \mathcal{S} . Let $X = (x^{127}, \dots, x^0)$ denote the output of the diffusion function. $R = (r^{127}, \dots, r^0)$ denotes the random string. The output of the nonlinear mixing $S = \mathcal{S}(\mathcal{D}(\Delta C), R) = \mathcal{S}(X, R) = (s^{127}, \dots, s^0)$ is represented as follows. For $1 \leq i \leq 128$,

$$s^i = \bigoplus_{j=0}^{i-1} x^j r^j \oplus x^{i-2} x^{i-1} \oplus x^i \quad (17)$$

where x^i , r^i and s^i denote the i -th bit of X , R and S , respectively. $s^0 = s^{128}$, $x^0 = x^{128}$ and $x^{-1} = 0$.

A. Analysis on LDRNM

Infection operation decomposition and analysis: In LDRNM, $x^i r^i$ is the smallest random-bit-related infection operation. Its protective effect can be concluded as:

- Based on a single infection, we find from (17) that the input and the output of $x^i r^i$ are both related to the unknown bits in X . There is no prior knowledge for further analysis.
- Based on M infections with the repeatable ΔC and X , we denote the output difference of $x^i r^i$ in the 1-st and j -th infection as $OUT_{1j}^i = x^i r_1^i \oplus x^i r_j^i$ ($j = 2, \dots, M$). It can be obtained from the output difference of the infection function as

$$OUT_{1j}^i = \begin{cases} s_1^i \oplus s_j^i, & i = 0 \\ (s_1^i \oplus s_1^{i+1}) \oplus (s_j^i \oplus s_j^{i+1}), & 1 \leq i \leq 127. \end{cases} \quad (18)$$

Let \mathbf{eM} denote the event that $OUT_{1j}^i = 0$ for $j = 2, \dots, M$. It means $x^i r_j^i$ keeps same in all these infections. Obviously, $P(\mathbf{eM} | x^i = 0) = 1$ and $P(\mathbf{eM} | x^i = 1) = \frac{1}{2^{M-1}}$. With enough OUT_{1j}^i , the value of x^i can be determined. According to the diffusion function \mathcal{D} , $P(x^i = 0) = P(x^i = 1) = \frac{1}{2}$. Then, $(P(\mathbf{eM}), P(\overline{\mathbf{eM}}))$ and $(P(x^i | \mathbf{eM}), P(x^i | \overline{\mathbf{eM}}))$ can be respectively derived from total probability formula and Bayes formula. The remaining uncertainty of x^i can be estimated as

$$\begin{aligned} & H(x^i | \text{PRIOR}^i) \\ & = H(x^i | OUT_{12}^i, \dots, OUT_{1M}^i) \\ & = P(\mathbf{eM}) H(x^i | \mathbf{eM}) + P(\overline{\mathbf{eM}}) H(x^i | \overline{\mathbf{eM}}) \\ & = \frac{2^{M-1} + 1}{2^M} \log_2(2^{M-1} + 1) - \frac{M-1}{2}. \end{aligned} \quad (19)$$

In LDRNM, all 128 infection operations are working with the above-mentioned type of prior knowledge. Consider the fact that the fault diffusion in ΔC (about the zero bytes) has

not been taken into account in infection operation analysis. We derive the protective effect of the countermeasure as

$$H(\Delta C|\Delta C_p, \Delta F_1, \dots, \Delta F_M) \\ = \max_{i=1}^{128} \{H(x^i|PRIOR^i) - H(\Delta C_p), 0\}. \quad (20)$$

To sum up, the differential distribution of $x^i r^i$ under the repeatable fault model results in the major security loss. It is consistent with the study in [29]. However, [29] ignores the prior knowledge of ΔC under the incomplete fault diffusion. It leads to an overestimation of security.

B. Security Quantification and Judgment

- For the attack scenario (1.1), (1.2), (2.1), (2.2) and (2.3): the fault diffusion in ΔC is incomplete. According to (19) and (20), $H(\Delta C|\Delta C_p, \Delta F) = H(\Delta C|\Delta C_p)$. The countermeasure is secure under the unrepeatable fault model. Whereas, $H(\Delta C|\Delta C_p, \Delta F_1, \Delta F_2) = 0$ bit. It is of no protective effect as long as the identical injection can be repeated twice.
- For the attack scenario (3.1), (3.2) and (3.3): the fault diffusion in ΔC is complete. $H(\Delta C|\Delta C_p, \Delta F) = H(\Delta C|\Delta C_p)$. It satisfies the security requirement in (7). $H(\Delta C|\Delta C_p, \Delta F_1, \Delta F_2) = 88.15$ bits, which is smaller than 116, 120 and 128. The countermeasure is insecure when the identical injection can be repeated twice.

Above all, LDRNM shows strong resistance to the attack with unrepeatable fault injection. However, it cannot resist the repeatable fault injection attack.

VIII. EXTENDED DISCUSSION AND COMPARISON

In this section, we first evaluate two other infection countermeasures briefly. Then, we discuss the evaluation metrics for the practical security. We also extend the framework to other attack scenarios. Finally, we make a comparison between different countermeasures.

A. Quantification Analysis on Other Infections

1) *Multiplicative Masking (MM)*: Multiplicative masking is a random infection countermeasure. It multiplies the N -bit ΔC by a fresh N -bit random number R in $GF(2^N)$. Then, it masks the final ciphertext with $R\Delta C$. In this countermeasure, $R\Delta C$ is also the smallest random-bit-related infection operation. When $\Delta C \neq 0$, it generates every element in the finite field with an equal probability. Therefore, the countermeasure is secure under all the scenarios listed in Table II, no matter the fault model is repeatable or not. However, when $R = 0$, the power consumption of the multiplication $R\Delta C$ is significantly lower than when $R \neq 0$. This phenomenon enables the identification of $R = 0$ and leads to a straightforward recovery of ΔC from $\Delta F = \Delta C \oplus 0$.

2) *Random Replacement (RR)*: Different from the generic infection construction in Fig. 1, [16] proposed a new structure based on replacement. The countermeasure adopts a boolean function BLFN that maps $\Delta C = 0$ to 0 and $\Delta C \neq 0$ to 1, to decide whether replacing the original cipher output C_0 with a

random number R . The output of the cipher implementation is presented as $(\neg \text{BLFN}(\Delta C) \cdot C_0) \oplus (\text{BLFN}(\Delta C) \cdot R)$. Because BLFN is not invertible for the non-zero ΔC , and R is irrelevant to both C and ΔC , then $H(\Delta C|\Delta C_p, \Delta F) = H(\Delta C|\Delta C_p)$ under all the scenarios listed in Table II. Therefore, the countermeasure is theoretical secure against the intermediate-oriented single fault attacks. It is consistent with the evaluation in [19] and [21]. However, similar to the consistency check in the detection countermeasure, BLFN brings about a 1-bit judgment condition. Therefore, the malicious modification on the result of BLFN may lead to infection failure.

B. Practical security evaluation

An effective countermeasure should render the fault attack infeasible in practice. The attack complexities, including the number of fault injections and the computation complexity, are candidates for the practical security metrics. For the fault injection, the security judging criterion is determined by the lifetime of the cipher implementation and the adversary's ability to monitor it. For the computation complexity, the judging criterion is decided by the present computer technologies. The complexity quantification results are greatly affected by the analytical strategy adopted in the attack. Therefore, they are more like the evaluation metrics for different adversaries than the unified metrics for infection countermeasures.

Compared to the complexities of the concrete attack, the minimum attack complexities are better choices for the countermeasure evaluation. The minimum number of injections α and the minimum computation complexity η are related to the conditional entropy of ΔC and \mathcal{K} . If each fault model can be repeated for M times in the retrieval of a N -bit secret key,

$$\alpha = \lceil \frac{MN}{H(\mathcal{K}) - H(\mathcal{K}|\Delta C, \Delta C_p, \Delta F_1 \dots \Delta F_M)} \rceil \quad (21)$$

$$\eta = \frac{\alpha t_o}{M} \cdot 2^{H(\Delta C|\Delta C_p, \Delta F_1, \dots, \Delta F_M)} + \alpha t \quad (22)$$

where t_o denotes the computation complexity for the retrieval of \mathcal{K} with the known ΔC . t denotes the computation complexity for the retrieval of ΔC with the known ΔF . According to the infection operation decomposition and analysis, if ΔC can be retrieved in γ separate parts, then $t = \gamma \cdot 2^{\frac{H(\Delta C)}{\gamma}}$.

The countermeasures proved secure in theory according to (7) must be secure in practice. The ones proved insecure according to (4) are always insecure in practice, especially when the fault diffusion is incomplete and the ΔC can be retrieved in separate parts. In other cases, it requires specific attack strategies to figure out whether the key could be retrieved within the complexity restrictions.

C. Evaluation under other attack scenarios

The extension of the current framework to other fault attack scenarios is discussed briefly.

1) *Flow sequence-oriented injection with instruction skip fault model*: The injection skips some procedures of the encryption. It has both the numerical and logical effects on the integrated cipher implementation.

TABLE III: Comparison of infection countermeasures in security and cost (based on attack scenarios in Table II)

				RIMBEN ^[†]	RIMBEN ^[‡]	ICDR		LDRNM		MM	RR
$H(\Delta C \Delta C_p, \Delta F_1, \dots, \Delta F_M)$	Index	$H(\Delta C C_p)$	l_0	$M = 1$	$M = 1$	$M = 1$	$M \geq 17$	$M = 1$	$M \geq 2$	M	M
	1.1	8	1	$\leq \mathbf{0.92}$	≤ 5.49	0	0	8	0	8	8
	1.2		8	$\leq \mathbf{0.92}$	$\leq \mathbf{5.49}$	0	0	8	0	8	8
	2.1	32	20	$\leq \mathbf{16}$	≤ 28.63	0	0	32	0	32	32
	2.2		24	$\leq \mathbf{16}$	≤ 28.63	0	0	32	0	32	32
	2.3		32	$\leq \mathbf{16}$	$\leq \mathbf{28.63}$	0	0	32	0	32	32
	3.1	128	116	$\leq \mathbf{34.47}$	$\leq \mathbf{60.31}$	128	0	128	$\leq \mathbf{88.15}$	128	128
	3.2		120	$\leq \mathbf{34.47}$	$\leq \mathbf{60.31}$	128	0	128	$\leq \mathbf{88.15}$	128	128
	3.3		128	$\leq \mathbf{34.47}$	$\leq \mathbf{60.31}$	128	0	128	$\leq \mathbf{88.15}$	128	128
	Num of random bits in each infection				64	$64 + \epsilon^{[§]}$	128		128		128

[†] RIMBEN without 4-state switches. The initial configurations of the fixed switches are open to the public.

[‡] RIMBEN without 4-state switches. The initial configurations of the fixed switches are kept secret.

[§] For the random bits that can be reused in infections, ε denotes the average requirement of them in each infection before the update.

On the one hand, the instruction skip injection can lead to the faulty inputs of the following procedures. It corrupts the intermediate (or ciphertext) of the unprotected cipher. Therefore, by transforming its numerical effect into an equivalent intermediate-oriented fault model, we can work out the security judging criteria for the infection function with the current framework. For instance, if the last cipher round encryption of AES-128 and the following infection are all skipped by the injection [19], [21], the 9-th cipher round result will become the ciphertext of the unprotected cipher. It is equivalent to injecting a 128-bit fault e in the 10-th round before SubBytes (SB) with the known location and the known value, $H(e) = 0$ bit. The key bytes involved in fault propagation can be approximately represented with the last round key, $H(K) = 128$ bits. Then, the equivalent uncertainty of ΔC that should be maintained by the secure infection is $l_0 \approx H(K) - H(e) = 128$ bits. In another instance, the injection targets at the 8-th cipher round and only skips its AddRoundKey (ADK) operation on the first byte. The following encryption will be executed with a faulty ADK output. If the faulty output byte is totally random, it is equivalent to injecting a byte-oriented fault e with the known location and unknown value, which is the same as scenario (3.2) in Table II. $H(e) = 8$ bits, $H(K) = 128$ bits and $l_0 \approx H(K) - H(e) = 120$ bits. If it takes the input byte of ADK as the faulty output byte, then e is equal to the first byte of the 8-th round key. Since e is available for the key retrieval, $H(e)$ can be taken as 0. $H(K) = 128$ bits and $l_0 \approx 128$ bits.

On the other hand, the injection decides whether the infection will be executed or not. It affects the security quantification on infection functions. If the infection is skipped, the equivalent uncertainty of ΔC maintained by the infection function is 0 bit. If it is not skipped, the remaining uncertainty of ΔC can be worked out in the same way as the current framework. Finally, with the security judging criteria and quantification results, we can make security judgments on infection countermeasures in the same way as the current framework.

2) *Intermediate-oriented double-fault injections*: The injections also affect the integrated cipher implementation at two levels. On the one hand, they corrupt the original and redundant intermediates of the unprotected cipher in pairs with two identical faults. Therefore, the equivalent security judging criteria for infection function about ΔC can be worked out

with either of these faults, according to the current framework. On the other hand, the double-fault injections make the infection input equal to zero. In this case, the equivalent uncertainty of ΔC maintained by the infection function can only be 0 bit.

D. Comparison

For all the infection countermeasures analyzed in this paper, Table III summarizes their security quantification results under the attack scenarios in Table II. It also summarizes their requirements for random bits. From the table, $H(\Delta C | \Delta C_p, \Delta F_1, \dots, \Delta F_M)$ grows with the rising fault diffusion in ΔC , which leads to an exponential increment in the computation complexity of the attack (refer to (22)). It indicates that the infections are effective countermeasures especially for the attack scenarios with complete fault diffusion in ΔC . In this table, we highlight the remaining uncertainty of ΔC that is no larger than the lower bound security requirement l_0 in bold. It indicates that the countermeasure is theoretically infeasible for the specific attack scenario.

The evaluation results are instructive in countermeasure selection. To be specific, RIMBEN with public initial configurations is never recommended. RIMBEN with secret initial configurations, ICDR and LDRNM are applicable in some unrepeatable injection scenarios. Among them, RIMBEN requires the least number of random bits. It is a cost-effective choice against the scenarios with the random fault model and the incomplete fault diffusion. ICDR and LDRNM require the same amount of random bits in each infection. Because LDRNM ensures a more comprehensive fault attack resistance, it is a better choice than ICDR. However, all of them are theoretically infeasible in the repeatable fault injection scenarios, even if ΔC can only be repeated for a few times. Thanks to the existence of the random fault model, it makes injecting identical faults difficult. The repetitive infections on the identical ΔC can seldom happen in practice. Multiplicative masking and random replacement have the widest security applicable scopes. Random replacement is more cost-effective because the random bits can be reused in different infections. Nevertheless, their enhancements of FA-resistance are at the expense of introducing new vulnerabilities under other implementation attack conditions. They are still not the best choice for cipher protection.

IX. CONCLUSION AND FUTURE WORK

The paper proposes a framework for security evaluation on infection countermeasures. It makes the evaluation not only generic for different countermeasures but also efficient under various attack scenarios. The framework is constructed with the idea to separate the infection function from the integrated cipher implementation. Firstly, we personalize the security judging criteria for infection functions according to the fault injection effects on the unprotected cipher. The criteria enable precise and individual security judgments under different attack scenarios. Secondly, based on the fact that the infection function usually consists of a series of simple and identical infection operations, we propose a universal security quantification method through infection operation analysis. The analysis enables a thorough vulnerability exploration and an efficient security quantification of the infection function.

The current framework mainly focuses on the evaluation against the intermediate-oriented single injection and the DFA-like attacks. So one possible future work is to extend the framework to cover more fault injection models and more attack scenarios. Moreover, as mentioned in Section VIII, a FA-resistant infection construction may result in new vulnerabilities in terms of other implementation attacks. Hence, how to involve other side-channel information leakages into the security evaluation on infection countermeasure is another problem to be solved.

REFERENCES

- [1] D. Boneh, R. A. DeMillo, and R. J. Lipton, "On the importance of checking cryptographic protocols for faults," in *Advances in Cryptology - EUROCRYPT'97: International Conference on the Theory and Application of Cryptographic Techniques Konstanz, Germany, May 11-15, 1997 Proceedings*, W. Fumy, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1997, pp. 37-51.
- [2] C. Giraud, "Dfa on aes," in *Advanced Encryption Standard - AES*, H. Dobbertin, V. Rijmen, and A. Sowa, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 27-41.
- [3] P. Dusart, G. Letourneux, and O. Vivolo, "Differential fault analysis on a.e.s.," in *Applied Cryptography and Network Security*, J. Zhou, M. Yung, and Y. Han, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 293-306.
- [4] 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 2003: 5th International Workshop, Cologne, Germany, September 8-10, 2003. Proceedings*, C. D. Walter, Ç. K. Koç, and C. Paar, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 77-88.
- [5] I. Biehl, B. Meyer, and V. Müller, "Differential fault attacks on elliptic curve cryptosystems," in *Advances in Cryptology - CRYPTO 2000: 20th Annual International Cryptology Conference Santa Barbara, California, USA, August 20-24, 2000 Proceedings*, M. Bellare, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 131-146.
- [6] E. Biham, L. Granboulan, and P. Q. Nguyen, "Impossible fault analysis of rc4 and differential fault analysis of rc4," in *Fast Software Encryption: 12th International Workshop, FSE 2005, Paris, France, February 21-23, 2005, Revised Selected Papers*, H. Gilbert and H. Handschuh, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 359-367.
- [7] C. Clavier, "Secret external encodings do not prevent transient fault analysis," in *Cryptographic Hardware and Embedded Systems - CHES 2007*, P. Paillier and I. Verbauwhede, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 181-194.
- [8] C. Dobraunig, M. Eichlseder, T. Korak, S. Mangard, F. Mendel, and R. Primas, "Sifa: Exploiting ineffective fault inductions on symmetric cryptography," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2018, no. 3, pp. 547-572, Aug 2018. [Online]. Available: <https://tches.iacr.org/index.php/TCHES/article/view/7286>
- [9] Y. Li, K. Sakiyama, S. Gomisawa, T. Fukunaga, J. Takahashi, and K. Ohta, "Fault sensitivity analysis," in *Cryptographic Hardware and Embedded Systems, CHES 2010*, S. Mangard and F.-X. Standaert, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 320-334.
- [10] R. Karri, K. Wu, P. Mishra, and Y. Kim, "Concurrent error detection schemes for fault-based side-channel cryptanalysis of symmetric block ciphers," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 21, no. 12, pp. 1509-1517, Dec 2002.
- [11] K. Wu, R. Karri, G. Kuznetsov, and M. Goessel, "Low cost concurrent error detection for the advanced encryption standard," in *2004 International Conference on Test*, Oct 2004, pp. 1242-1248.
- [12] M. Karpovsky, K. J. Kulikowski, and A. Taubin, "Robust protection against fault-injection attacks on smart cards implementing the advanced encryption standard," in *International Conference on Dependable Systems and Networks, 2004*, June 2004, pp. 93-101.
- [13] M. Joye, P. Manet, and J.-B. Rigaud, "Strengthening hardware aes implementations against fault attacks," *IET Information Security*, vol. 1, no. 3, pp. 106-110, Sept 2007.
- [14] V. Lomné, T. Roche, and A. Thillard, "On the need of randomness in fault attack countermeasures - application to aes," in *2012 Workshop on Fault Diagnosis and Tolerance in Cryptography*, Sept 2012, pp. 85-94.
- [15] B. Gierlichs, J.-M. Schmidt, and M. Tunstall, "Infective computation and dummy rounds: Fault protection for block ciphers without check-before-output," in *Progress in Cryptology - LATINCRYPT 2012: 2nd International Conference on Cryptology and Information Security in Latin America, Santiago, Chile, October 7-10, 2012. Proceedings*, A. Hevia and G. Neven, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 305-321.
- [16] H. Tupsamudre, S. Bisht, and D. Mukhopadhyay, "Destroying fault invariant with randomization," in *Cryptographic Hardware and Embedded Systems - CHES 2014: 16th International Workshop, Busan, South Korea, September 23-26, 2014. Proceedings*, L. Batina and M. Robshaw, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 93-111.
- [17] B. Wang, L. Liu, C. Deng, M. Zhu, S. Yin, Z. Zhou, and S. Wei, "Exploration of benes network in cryptographic processors: A random infection countermeasure for block ciphers against fault attacks," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 2, pp. 309-322, Feb 2017.
- [18] A. Battistello and C. Giraud, "Fault analysis of infective aes computations," in *2013 Workshop on Fault Diagnosis and Tolerance in Cryptography*, Aug 2013, pp. 101-107.
- [19] S. Patranabis, A. Chakraborty, and D. Mukhopadhyay, "Fault tolerant infective countermeasure for aes," in *Security, Privacy, and Applied Cryptography Engineering: 5th International Conference, SPACE 2015, Jaipur, India, October 3-7, 2015, Proceedings*, R. S. Chakraborty, P. Schwabe, and J. Solworth, Eds. Cham: Springer International Publishing, 2015, pp. 190-209.
- [20] A. Battistello and C. Giraud, "A note on the security of ches 2014 symmetric infective countermeasure," in *Constructive Side-Channel Analysis and Secure Design: 7th International Workshop, COSADE 2016, Graz, Austria, April 14-15, 2016, Revised Selected Papers*, F.-X. Standaert and E. Oswald, Eds. Cham: Springer International Publishing, 2016, pp. 144-159.
- [21] S. Patranabis, A. Chakraborty, and D. Mukhopadhyay, "Fault tolerant infective countermeasure for aes," *Journal of Hardware and Systems Security*, vol. 1, no. 1, pp. 3-17, Mar 2017. [Online]. Available: <https://doi.org/10.1007/s41635-017-0006-1>
- [22] S. Ghosh, D. Saha, A. Sengupta, and D. Roy Chowdhury, "Preventing fault attacks using fault randomization with a case study on aes," in *Information Security and Privacy: 20th Australasian Conference, ACISP 2015, Brisbane, QLD, Australia, June 29 - July 1, 2015, Proceedings*, E. Foo and D. Stebila, Eds. Cham: Springer International Publishing, 2015, pp. 343-355.
- [23] E. Biham and A. Shamir, "Differential fault analysis of secret key cryptosystems," in *Advances in Cryptology - CRYPTO '97*, B. S. Kaliski, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1997, pp. 513-525.
- [24] N. T. Courtois, D. Ware, and K. M. Jackson, "Fault-algebraic attacks on inner rounds of des," in *eSmart 2010*, 2010, pp. 22-24.
- [25] R. C. W. Phan and S.-M. Yen, "Amplifying side-channel attacks with techniques from block cipher cryptanalysis," in *Smart Card Research and Advanced Applications*, J. Domingo-Ferrer, J. Posegga, and D. Schreckling, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 135-150.
- [26] M. Rivain, "Differential fault analysis on des middle rounds," in *Cryptographic Hardware and Embedded Systems - CHES 2009: 11th International Workshop Lausanne, Switzerland, September 6-9, 2009*

Proceedings, C. Clavier and K. Gaj, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 457–469.

- [27] K. Sakiyama, Y. Li, M. Iwamoto, and K. Ohta, “Information-theoretic approach to optimal differential fault analysis,” *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 1, pp. 109–120, Feb 2012.
- [28] J. Blömer and J.-P. Seifert, “Fault based cryptanalysis of the advanced encryption standard (aes),” in *Financial Cryptography*, R. N. Wright, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 162–181.
- [29] S. Banik and A. Bogdanov, “Cryptanalysis of two fault countermeasure schemes,” in *Progress in Cryptology - INDOCRYPT 2015: 16th International Conference on Cryptology in India, Bangalore, India, December 6-9, 2015, Proceedings*, A. Biryukov and V. Goyal, Eds. Cham: Springer International Publishing, 2015, pp. 241–252.



Wei Xi received the M.S. degree in electrical and electronic engineering from Huazhong University of Science and Technology, Wuhan, China, in 2003.

He is currently a Senior Engineer in Electric Power Research Institute, China Southern Power Grid. His research interests include smart grid and electric power chip development.



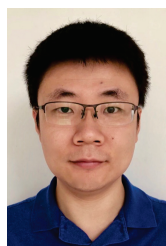
Jingyi Feng received the B.S. and M.S. degrees in electronic engineering from the University of Science and Technology of China, Hefei, China, in 2013 and 2016, respectively.

She is currently pursuing the Ph.D. degree in Institute of Software, Chinese Academy of Sciences. Her major research interests include security evaluation and improvement for cryptographic devices.



Hua Chen received the Ph.D. degree from Institute of Software, Chinese Academy of Sciences, Beijing, China, in 2005.

She is currently a Research Professor with Trusted Computing and Information Assurance Laboratory, Institute of Software, Chinese Academy of Sciences. She is a Senior Member of Chinese Association for Cryptologic Research. Her research interests include side-channel cryptanalysis, automatic cryptanalysis and randomness test.



Yang Li received the Ph.D. degree in engineering from the University of Electro-Communications, Tokyo, Japan, in 2012.

He is currently an Associated Professor in the Department of Informatics at the University of Electro-Communications, Japan. His main research interests include security evaluation and improvement for cryptographic hardware and IoT devices.



Zhipeng Jiao received the B.S. degree in computer science from Zhengzhou University, Zhengzhou, China, in 2014.

He is currently pursuing the Ph.D. degree in Institute of Software, Chinese Academy of Sciences. His current research interests include side-channel attack and protection.