Improved Fault Analysis on SIMECK Ciphers

Duc-Phong Le, Rongxing Lu, and Ali A. Ghorbani

Canadian Institute for Cybersecurity, University of New Brunswick,
Fredericton, Canada E3B 5A3.
{ le.duc.phong,rlu1,ghorbani}@unb.ca

Abstract. The advances of Internet of Things (IoT) have had a fundamental impact and influence in sharping our rich living experiences. However, since IoT devices are usually resource-constrained, lightweight block ciphers have played a major role in serving as a building block for secure IoT protocols. In CHES 2015, SIMECK, a family of block ciphers, was designed for resource-constrained IoT devices. Since its publication, there have been many analyses on its security. In this paper, under the one bit-flip model, we propose a new efficient fault analysis attack on SIMECK ciphers. Compared to those previously reported attacks, our attack can recover the full master key by injecting faults into only a *single round* of all SIMECK family members. This property is crucial, as it is infeasible for an attacker to inject faults into different rounds of a SIMECK implementation on IoT devices in the real world. Specifically, our attack is characterized by exercising a deep analysis of differential trail between the correct and faulty immediate ciphertexts. Extensive simulation evaluations are conducted, and the results demonstrate the effectiveness and correctness of our proposed attack.

Keywords: SIMECK Ciphers, Lightweight Cryptography, Cryptanalysis, Differential Fault Analysis, Bit-flip Model.

1 Introduction

One of the key technologies transforming our living experiences into much smarter ones is the Internet of Things (IoT). However, since IoT devices are usually resource-constrained, it is pressing to design lightweight block ciphers as building blocks to secure IoT protocols, *i.e.*, providing basic security requirements including confidentiality, data integrity, and source authentication for IoT devices.

By now, there have been numerous lightweight block ciphers introduced. A statistic performed by researchers of the University of Luxembourg [1] lists 36 lightweight block ciphers proposed by 2016. In order to evaluate and standardize lightweight cryptographic algorithms, NIST is supporting a Lightweight Cryptography project. By now, 32 candidates have been selected and are evaluating in Round 2. Readers can find more information about the project in this link `https://csrc.nist.gov/projects/lightweight-cryptography`.

In CHES 2015, Yang *et al.* [1] introduced SIMECK, which is a family of lightweight block ciphers based upon Feistel structure and combines the good design principles

[1] https://www.cryptolux.org/index.php/Lightweight˙Block˙Ciphers

of the SIMON and SPECK block ciphers [2]. Precisely, SIMECK consists of three members with block sizes of 32, 48, and 64, and the corresponding key sizes are 64, 96, and 128, respectively. As demonstrated in [1], SIMECK allows a smaller and more efficient hardware implementation in comparison to SIMON. Due to its nice property in efficiency, SIMECK has been significantly analyzed since its publication.

As is known to all, fault analysis is a very efficient implementation attack against cryptographic protocols, which essentially tries to influence the behavior of a cryptographic device and determine sensitive information by examining the effects. Today, there have been several mechanisms to inject faults into microprocessors. Examples include changes in the power supply, the clock frequency [3], or intensive lighting of the circuit [4]. In most cases, injecting faults will force a change in the data located in one of the registers. The first fault attack against RSA-CRT implementation was reported in a Bellcore press in 1996 and subsequently analyzed by Boneh, DeMillo and Lipton in [5]. Concretely, Boneh *et al.* [5] showed that many implementations of RSA signatures and other public-key algorithms are vulnerable to a certain transient fault occurring during the processing phase, and the RSA-CRT implementation is at extreme risk to be compromised by using one erroneous result. After that, Biham and Shamir [6] introduced the Differential Fault Analysis (DFA) against symmetric cryptosystems such as the DES [6]. They assume that an attacker can disturb DES computations by using the same - but *unknown* - plaintext at the last (three) DES round(s). The wrong ciphers provide a system of equations for the unknown last round key bits that finally reveals the correct key value. Since then, there have been many DFA attacks carried out on other block ciphers, including attacks against AES [7,8], Triple DES [9], IDEA [10], SIMON and SPECK [11, 12] or KLEIN lightweight block ciphers [13]. These attacks are performed on the key schedule [8], S-box [13], or intermediate inputs [11]. They are also carried out under various fault models (see more details in Section 2.3.2).

In 2016, Nalla *et al.* [14] presented two fault analysis attacks against SIMECK ciphers. While the former is under the one bit-flip fault model, the latter is under the random byte fault model. Both of the two attacks aim at recovering the last round key by injecting faults into the second last round. In this paper, we would like to present an improved fault analysis attack against SIMECK block ciphers under the one bit-flip model, which is more practical than Nalla *et al.*'s attacks. Specifically, the main contribution of this paper is three-fold.

- First, we show that the whole master key of SIMECK block ciphers could be recovered by injecting faults into a single round of the ciphers, which makes our attack more *practical* than Nalla *et al.*'s attacks [14], as their attacks require faults from 4 different rounds.
- Second, by deducing more key bits with one fault, our attack also requires fewer faults than those previously reported attacks.
- Third, we conduct extensive simulation evaluations, and the results demonstrate the effectiveness and correctness of our attack.

The remainder of this paper is organized as follows. Section 2 briefly recalls SIMECK ciphers, security analyses, and differential fault analysis on this family of ciphers. Section 3 discusses some facts and observations that will be used in our attack. Section 4

describes our differential fault analysis under the one bit-flip model. Then, we present our simulation evaluations in Section 5. Finally, we draw our conclusions in Section 6.

2 Preliminaries

In this section, we briefly recall the specification of the SIMECK family of lightweight block ciphers and differential fault analysis on this family of ciphers. For the sake of consistency, we make use of the following notations listed in Table 1 in the rest of the paper. In principles, we use capital letters for vectors or strings while small letters are used to represent individual bits.

Table 1. Definition of Notations

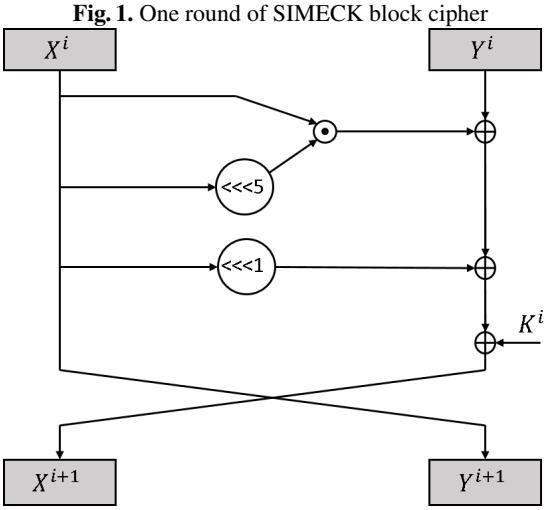
Notation	Definition
T	the total number of rounds of the cipher.
(X^i, Y^i)	input of round i, $i = 0, 1, \ldots, T-1$.
(X^T, Y^T)	ciphertext.
$(\hat{X}^i, \hat{Y}^i)$	faulty input of round i, $i = 0, 1, \ldots, T-1$.
$(\hat{X}^T, \hat{Y}^T)$	faulty ciphertext.
$X^i + Y^i$	'XOR' operation of X^i and Y^i.
$X^i Y^i$	'AND' operation of X^i and Y^i.
(x^i_j, y^i_j)	variables denoting bit j of the input of the round i, $i = 0, 1, \ldots, T$, $j = 0, 1, \ldots, n-1$.
$(\hat{x}^i_j, \hat{y}^i_j)$	variables denoting bit j of the faulty input of the round i, $i = 0, 1, \ldots, T$, $j = 0, 1, \ldots, n-1$.
$x^i_j + y^i_j$	bitwise 'XOR' operation of x^i_j and y^i_j.
$x^i_j y^i_j$	bitwise 'AND' operation of x^i_j and y^i_j.
K^i	the round key in round i, $i = 0, 1, \ldots, T-1$.
k^i_j	j^{th}, $0 \leq j \leq n-1$ bit round key of round i.
Δ^i	the difference between correct and faulty left inputs of round i, where $0 \leq i < T$.
δ^i_j	the difference at bit j of Δ^i, where $0 \leq i < T$, and $0 \leq j < n$.
$S^a(X)$	Circular left rotation of a n bit word X by a bits.

2.1 SIMECK Specification

The SIMECK family of lightweight block ciphers was introduced in CHES 2015 [1] and was optimized for hardware implementations. Similar to SIMON, SIMECK is based on a typical Feistel design and comprises three simple operations, namely, the bit-wise 'AND', 'rotation' and 'XOR' operations. Let n denote the word size. Then, SIMECK$2n/4n$ refers to perform encryptions or decryptions on $2n$-bit message blocks using a $4n$-bit key, where $n = 16, 24$ or 32 is called the word size of SIMECK$2n/4n$.

Fig. 1 shows a round function of SIMECK. Basically, the design of this family is based on the balanced Feistel network. There are in total $32, 36$ and 44 encryption

Fig. 1. One round of SIMECK block cipher



rounds for SIMECK32/64, SIMECK48/96, an SIMECK64/128, respectively. In the following, let us consider encrypting a plaintext P. At each round i, the input message is divided into two words X^i and Y^i, where $P = X^0 \| Y^0$ is the $2n$-bit plaintext. The round function of SIMECK is defined as follows:

$$(X^{i+1}, Y^{i+1}) = (Y^i + F(X^i) + K^i, X^i) \tag{1}$$

where $F(X) = X \cdot S^5(X) + S^1(X)$, K^i is the round key at the round i, and S^a denotes a circular left rotation by a bits. Let $(x^i_{n-1}, \ldots, x^i_0)$, and $(y^i_{n-1}, \ldots, y^i_0)$ denote the input of round i for $0 \le i \le T-1$. The following relations hold for any $j = 0, \ldots, n-1$:

$$\begin{aligned} x^{i+1}_j &= f(x^i_j) + y^i_j + k^i_j, \\ y^{i+1}_j &= x^i_j, \end{aligned} \tag{2}$$

where $f(x^i_j) = (x^i_j \,\&\, x^i_{(j-5) \bmod n}) + x^i_{(j-1) \bmod n}$. As all indices are computed modulo n, in what follows, x^i_j will denote $x^i_{j \bmod n}$, $x^i_j x^i_{j'}$ will denote $x^i_{j \bmod n} \,\&\, x^i_{j' \bmod n}$, and $x^i_j + x^i_{j'}$ will denote $x^i_{j \bmod n} + x^i_{j' \bmod n}$.

Key Schedule: In SIMECK ciphers, the round key is generated from the master K as follows. Firstly, the master is segmented into four words $K = (K^3, K^2, K^1, K^0)$, then for $i \ge 0$:

$$K^{i+4} = K^i + F(K^{i+1}) + C + (z_j)_i,$$

where C is a constant value defined by $C = 2^n - 4$, $(z_j)_i$ denotes the i-th bit of the sequence z_j. SIMECK32/64 and SIMECK48/96 use the same sequence z_0, which can be generated by the primitive polynomial $X^5 + X^2 + 1$ with the initial state $(1, 1, 1, 1, 1)$, whereas SIMECK64/128 uses the sequence z_1, which can be generated by the primitive

polynomial X^6+X+1 with the initial state $(1, 1, 1, 1, 1, 1)$. Compared to SIMON ciphers, SIMECK ciphers are more efficient and compact hardware implementation due to its reuse of the round function with the round constant $C + (z_j)_i$ in the key scheduling algorithm.

2.2 Security Analysis on SIMECK Ciphers

Since SIMECK is based on SIMON and SPECK ciphers [2], the security level of SIMECK ciphers is expected to be *comparable* to the those of SIMON ciphers.

Since their publication, there have been a number of academic works analyzing the security of SIMECK ciphers [15–21]. We briefly recall a few attacks in this section. Kolbl and Roy [16] performed differential and linear cryptanalysis [2] [22, 23], the two most widely used attacks on block ciphers. They managed to break up to 19/32, 26/36, and 33/44 rounds of SIMECK32/64, SIMECK48/96, and SIMECK64/128 ciphers, respectively, with higher probability. These results cover more rounds compared to those against SIMON ciphers (see [16, Table 6]). Then, an improved differential attack using the dynamic key-guessing technique [17] was able to break up to 22, 28, and 35 rounds of SIMECK32/64, SIMECK48/96, and SIMECK64/128 ciphers. Even better, using the same technique with linear hull cryptanalysis, Qin et al. [18] were able to break 1 or 2 more rounds compared to results in [17].

2.3 Differential Fault Analysis (DFA) on SIMECK Ciphers

Differential Fault Analysis The *Differential Fault Analysis* or *DFA* was first proposed by Biham and Shamir in [6]. Unlike the statistical attacks mentioned above such as differential and linear attacks, which requires a large amount of data to perform an attack, a DFA attack is able to recover the full master key by using just a few pair of plaintexts/ciphertexts.

In the Biham-Shamir's DFA attack against DES block cipher [24], an attacker will inject a fault in some rounds when executing a cryptographic implementation to obtain a faulty ciphertext. By analyzing the difference between the correct and faulty ciphertexts, the last round key could be revealed. Then, with the knowledge of the last round key, the attacker decrypts the correct ciphertext to obtain the input of the last round, which is the output of the second last round. After that, the attacker repeats this procedure to obtain more round keys until the master key can be deduced by the key schedule. Subsequently, various DFA attacks on block ciphers have also been carried out, including attacks against AES [7], Triple DES [9], IDEA [10], SIMON and SPECK [11, 12], and SIMECK [14].

Fault Models There exist various techniques to inject a fault into a computing device during its execution. Low-cost techniques include power glitch, clock tempering, or

[2] Differential cryptanalysis is a chosen-plaintext attack that studies how differences in input can affect the resultant difference at the output. Linear cryptanalysis is a known-plaintext attack in which the attacker studies probabilistic linear relations (called linear approximations) between parity bits of the plaintext, the ciphertext, and the secret key.

temperature variation [3, 5]. Kommerling and Kuhn reported that glitch attacks at the external power and clock supply lines are the most useful in practice [3]. High-cost techniques include light/laser injections or electromagnetic (EM) disturbances [4, 25].

These techniques allow specific control of a single register, a bus, or memory.

The fault characteristics resulting from a fault injection are commonly captured in a *fault model*. A fault model will indicate the following characteristics: the location of the fault, the number of bits affected by the fault, and the fault types. In [26], Riviere *et al.* classified fault models as follows:

- *Bit-wise models*: in which faults will manipulate a single bit. There are five types of bit-wise fault models: bit-set, bit-flip, bit-reset, stuck-at, and random-value.
- *Byte-wise models*: in which faults will modify a byte at once. There are three types of byte-wise fault models: byte-set, byte-reset, or random-byte.
- *Wider models*: in which faults will manipulate an entire word that can be from 8 to 64 bits depending on the given architecture.

Theoretical works often assume that the attacker has a precise control on both timing and location, *i.e.*, he knows the attacked bit as well as the attacked operation. In practice, it would be more challenging to inject a fault into a precise bit. In this paper, we consider the bit-flip model in which a single bit will be flipped to its complementary value, either from 0 to 1 or from 1 to 0. We also assume that the location of the flipped bit is *unknown* to the attacker.

DFA attacks on SIMECK block cipher In the following, we briefly review differential fault analysis against SIMECK cipher in the *bit-flip* and *random byte* fault models.

In [14], Nalla *et al.* presented the first differential fault analysis against SIMECK ciphers. Specifically, they demonstrated two DFA attacks. The former makes use of the *bit-flip fault model* and could recover the n-bit *last round key* using $n/2$ bit faults. The latter makes use of the *random byte fault model* that is more practical and could retrieve the last round key using $n/8$ faults. The process could be repeated four times to recover 4 round keys that are enough to recover the full master key.

Basically, their attacks mainly exploit the information leaked by the 'AND' operation, which is the only non-linear function of SIMECK. Specifically, the attacker injects a fault in the intermediate left half ciphertext X^{T-2}, where T is the number of rounds of SIMECK. If one of the two input bits of the 'AND' operation is 0, then flipping the other input bit does not affect the output bit of $X^{T-1} = Y^T$. From the following equation, we can observe that one can deduce the last round key K^{T-1} if the value X^{T-2} is known.

$$K^{T-1} = X^{T-2} + F(Y^T) + X^T \tag{3}$$

Suppose that the j^{th} bit of X^{T-2} was flipped, one is able to deduce the value of $(j-5)^{th}$ and $(j+5)^{th}$ bits of X^{T-2}, and thus recover the corresponding bits of K^{T-1} using Eq. (3). In other words, 2 bits of K^{T-1} will be disclosed with every bit flipped in X^{T-2}. Therefore, by assuming that one could control the injected fault location, it requires $n/2$ faulty ciphertexts to retrieve the n-bit key. However, if the attacker has no control over

the location of the flipped bit, the average number of faults required is approximately double, namely, around n faults.

3 Preliminary Observations and Facts

In this section, some facts and observations that can be used to analyze faults on SIMECK block ciphers will be discussed. First, we consider the following lemma.

Lemma 1. *Let $X^t = \{x_0^t, x_1^t \ldots, x_{n-1}^t\}$ and $\hat{X}^t = \{\hat{x}_0^t, \hat{x}_1^t, \ldots, \hat{x}_{n-1}^t\}$ be the correct and faulty left inputs respectively of the intermediate t-th round, $0 \leq t < T$. Let $\delta_j^t = x_j^t + \hat{x}_j^t$, for $0 \leq j < n$, be the differential representation of two correct and faulty bits x_j^t and $\hat{x}_j^t$. We have:*

$$\delta_j^{t+1} = \delta_j^t x_{j-5}^t + \delta_{j-5}^t x_j^t + \delta_j^t \delta_{j-5}^t + \delta_{j-1}^t + \delta_j^{t-1} \tag{4}$$

Proof. From Eq. (2), we have:

$$\begin{aligned} x_j^{t+1} &= x_j^t x_{j-5}^t + x_{j-1}^t + y_j^t + k_j^t, \quad \text{and} \\ \hat{x}_j^{t+1} &= \hat{x}_j^t \hat{x}_{j-5}^t + \hat{x}_{j-1}^t + \hat{y}_j^t + k_j^t \end{aligned}$$

Note that $y_j^t = x_j^{t-1}$. Then, summing up the two above equations gives:

$$\delta_j^{t+1} = x_j^t x_{j-5}^t + \hat{x}_j^t \hat{x}_{j-5}^t + \delta_{j-1}^t + \delta_j^{t-1}$$

We have:

$$\begin{aligned} & x_j^t x_{j-5}^t + \hat{x}_j^t \hat{x}_{j-5}^t \\ &= (x_j^t + \hat{x}_j^t)(x_{j-5}^t + \hat{x}_{j-5}^t) + \hat{x}_j^t x_{j-5}^t + x_j^t \hat{x}_{j-5}^t \\ &= \delta_j^t \delta_{j-5}^t + (\hat{x}_j^t x_{j-5}^t + x_j^t x_{j-8}^t) + (x_j^t \hat{x}_{j-5}^t + x_j^t x_{j-8}^t) \\ &= \delta_j^t \delta_{j-5}^t + \delta_j^t x_{j-5}^t + \delta_{j-5}^t x_j^t \end{aligned}$$

As a result, $\delta_j^{t+1} = \delta_j^t x_{j-5}^t + \delta_{j-5}^t x_j^t + \delta_j^t \delta_{j-5}^t + \delta_{j-1}^t + \delta_j^{t-1}$. □

Lemma 1 tells us that each bit δ_j^{t+1} can be represented in terms of intermediate plaintext bits and input differences in the previous rounds. More particular, the value of δ_j^{t+1} depends on 2 bits of the intermediate input X^t, 3 bits of the input difference Δ^t, and one bit of Δ^{t-1}. This allows us to construct a differential trail table to record and trace the δ_j^i values. Table 2 shows the differential trail of SIMECK32/64 when we inject faults into the round 27. The other trail tables are listed in Fig. 2.

Remark 1 (Input differences at the round $T-2$.). Let T be the number of round of a SIMECK cipher, given the left input differences at the rounds $T-1$ and T (*i.e.*, Δ^{T-1} and Δ^T, resp.), and the left input of the round $T-1$, *i.e.*, X^{T-1}, then the left input differences of the round $T-2$ could be computed as follows:

Fig. 2. Differential Trails of SIMECK Ciphers. Without loss of generality, we suppose that a fault will be injected into the bit 0. The notation $*$ denotes for a complex non-linear expression of variables that is non-trivial to deduce their values.

SIMECK48/96

Δ^{31} : $(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$
Δ^{32} : $(x_{19}^{31}, 1, 0, 0, 0, x_5^{31}, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$
Δ^{33} : $(*, x_{20}^{32} + x_{19}^{31}, 1, 0, 0, *, x_6^{32} + x_5^{31}, 0, 0, 0, *, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$
Δ^{34} : $(*, *, x_{21}^{33} + x_{20}^{32} + x_{19}^{31}, 1, 0, *, *, x_7^{33} + x_6^{32} + x_5^{31}, 0, 0, *, *, 0, 0, 0, *, 0, 0, 0, 0, 0, 0, 0, 0)$
Δ^{35} : $(*, *, *, x_{22}^{34} + x_{21}^{33} + x_{20}^{32} + x_{19}^{31}, 1, *, *, *, x_8^{34} + x_7^{33} + x_6^{32} + x_5^{31}, 0, *, *, *, 0, 0, *, *, 0, 0, 0, *, 0, 0, 0)$
Δ^{36} : $(*, *, *, *, x_{23}^{35} + x_{22}^{34} + x_{21}^{33} + x_{20}^{32} + x_{19}^{31}, *, *, *, *, x_9^{35} + x_8^{34} + x_7^{33} + x_6^{32} + x_5^{31}, *, *, *, *, 0, *, *, *, 0, 0, *, *, 0, 0)$

SIMECK64/128

Δ^{39} : $(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$
Δ^{40} : $(x_{27}^{39}, 1, 0, 0, 0, x_5^{39}, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$
Δ^{41} : $(*, x_{28}^{40} + x_{27}^{39}, 1, 0, 0, *, x_6^{40} + x_5^{39}, 0, 0, 0, *, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$
Δ^{42} : $(*, *, x_{29}^{41} + x_{28}^{40} + x_{27}^{39}, 1, 0, *, *, x_7^{41} + x_6^{40} + x_5^{39}, 0, 0, *, *, 0, 0, 0, *, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$
Δ^{43} : $(*, *, *, x_{30}^{42} + x_{29}^{41} + x_{28}^{40} + x_{27}^{39}, 1, *, *, *, x_8^{42} + x_7^{41} + x_6^{40} + x_5^{39}, 0, *, *, *, 0, 0, *, *, 0, 0, 0, *, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$
Δ^{44} : $(*, *, *, *, x_{31}^{43} + x_{30}^{42} + x_{29}^{41} + x_{28}^{40} + x_{27}^{39}, *, *, *, *, x_9^{43} + x_8^{42} + x_7^{41} + x_6^{40} + x_5^{39}, *, *, *, *, 0, *, *, *, 0, 0, *, *, 0, 0, 0, *, 0, 0, 0, 0, 0, 0)$

$$\begin{aligned}\Delta^{T-2} = \Delta^{T-1} S^5(X^{T-1}) + S^5(\Delta^{T-1}) X^{T-1} \\ + \Delta^{T-1} S^5(\Delta^{T-1}) + S^1(\Delta^{T-1}) + \Delta^T\end{aligned} \tag{5}$$

Eq. (5) can be easily obtained because from Eq. (4) we have:

$$\delta_j^{T-2} = \delta_j^{T-1} x_{j-5}^{T-1} + \delta_{j-5}^{T-1} x_j^{T-1} + \delta_j^{T-1} \delta_{j-5}^{T-1} + \delta_{j-1}^{T-1} + \delta_j^T$$

Once an attacker obtains the correct ciphertext $C = (X^T, Y^T)$ and the faulty ciphertext $\hat{C} = (\hat{X}^T, \hat{Y}^T)$, s/he first computes values: $X^{T-1} = Y^T$, $\hat{X}^{T-1} = \hat{Y}^T$, $\Delta^T = X^T + \hat{X}^T$, and $\Delta^{T-1} = X^{T-1} + \hat{X}^{T-1}$. Then, s/he is able to deduce the value of Δ^{T-2} via Eq. (5).

Fault Propagation For each presence of fault in the round t at the bit position j, *i.e.*, $\delta_j^t = 1$, it will affect into 3 left input differences in the next round $t + 1$, they are values of δ_j^{t+1}, δ_{j+1}^{t+1}, and δ_{j+5}^{t+1}.

Table 2. Differential trail for the left half of the last 7 rounds of SIMECK32/64 when flipping the least significant bit 0 at the round 27. The notation $*$ denotes a non-linear expression that would not be used in our attack.

Bit	**0**	**1**	**2**	**3**	**4**	**5**	**6**
Δ^{26}	0	0	0	0	0	0	0
Δ^{27}	1	0	0	0	0	0	0
Δ^{28}	x_{11}^{27}	1	0	0	0	x_5^{27}	0
Δ^{29}	$*$	$x_{12}^{28} + x_{11}^{27}$	1	0	0	$*$	$x_6^{28} + x_5^{27}$
Δ^{30}	$*$	$*$	$x_{13}^{29} + x_{12}^{28} + x_{11}^{27}$	1	0	$*$	$*$
Δ^{31}	$*$	$*$	$*$	$x_{14}^{30} + x_{13}^{29} + x_{12}^{28} + x_{11}^{27}$	$*$	$*$	$*$
Δ^{32}	Known values						

Bit	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Δ^{26}	0	0	0	0	0	0	0	0	0
Δ^{27}	0	0	0	0	0	0	0	0	0
Δ^{28}	0	0	0	0	0	0	0	0	0
Δ^{29}	0	0	0	$*$	0	0	0	0	0
Δ^{30}	$x_7^{29} + x_6^{28} + x_5^{27}$	0	0	$*$	$*$	0	0	0	$*$
Δ^{31}	$*$	$x_8^{30} + x_7^{29} + x_6^{28} + x_5^{27}$	0	$*$	$*$	$*$	0	0	$*$
Δ^{32}	Known values								

From Lemma 1, we have:

$$\begin{aligned}
\delta_j^{t+1} &= \delta_j^t x_{j-5}^t + \delta_{j-5}^t x_j^t + \delta_j^t \delta_{j-5}^t + \delta_{j-1}^t + \delta_j^{t-1} \\
\delta_{j+1}^{t+1} &= \delta_{j+1}^t x_{j-4}^t + \delta_{j-4}^t x_{j+1}^t + \delta_{j+1}^t \delta_{j-4}^t + \delta_j^t + \delta_{j+1}^{t-1} \\
\delta_{j+5}^{t+1} &= \delta_{j+5}^t x_j^t + \delta_j^t x_{j+5}^t + \delta_{j+5}^t \delta_j^t + \delta_{j+4}^t + \delta_{j+5}^{t-1}.
\end{aligned}$$

The values are dependent on the value of δ_j^t. From Table 2, it can be seen that if one injects a fault at the bit position 0 of the round 27, that is, $\delta_0^{27} = 1$, then three bit positions $0, 1$ and 5 at the round 28 will be affected, they are δ_0^{28}, δ_1^{28}, and δ_5^{28}. These faults continue propagating into the next round in the same way.

Lemma 2. *Suppose that a fault is injected into the left input of the round t and one-bit flipped at the position ℓ is made,* i.e.*, $\delta_\ell^t = 1$ and $\delta_j^t = 0$ for $j \neq \ell$. Then, for $i \geq 1$,*

1. $\delta_{\ell+i}^{t+i} = 1$ *for* $i \leq \min\{4, \frac{n}{5}\}$
2. $\delta_{\ell+j}^{t+i} = 0$ *for* $1 \leq i \leq 3$, *and* $i < j \leq 4$
3. $\delta_{\ell+j}^{t+i} = 0$ *for* $2 \leq i \leq 4$, *and* $i + 5 \leq j \leq i + 8$
4. $\delta_j^{t+i} = 0$ *for* $\ell + 5i < j < n + \ell$.

Proof. This lemma can be easily proved due to Eq. (4). It can be seen that that the bit-flipped fault δ_ℓ^t will affect to three input differences in the next round, that is, δ_ℓ^{t+1}, $\delta_{\ell+1}^{t+1}$ and $\delta_{\ell+5}^{t+1}$. Bits at the position j, where $j \in [0, \ell)$ and $j \in (\ell + 5, n)$ will not be affected, *i.e.*, $\delta_j^{t+1} = 0$. Furthermore, $\delta_{\ell+1}^{t+1} = \delta_\ell^t = 1$ and $\delta_{\ell+j}^{t+1} = 0$, for $2 \leq j \leq 4$ due to Lemma 1.

Table 3. Differential trail for the left half of the last 6 rounds of SIMECK48/96 and SIMECK64/128 when a fault injected at the round $T-5$ causing one bit flipped. The notation $*$ denotes an algebraic expression of immediate input bit variables.

SIMECK48/96	Left Input Differences (Bit position)		
Round	0–7	8–15	16–23
Δ^{31}	10000000	00000000	00000000
Δ^{32}	*1000*00	00000000	00000000
Δ^{33}	**100**0	00*00000	00000000
Δ^{34}	***10***	00**000*	00000000
Δ^{35}	****1***	*0***00*	*000*000
Δ^{36}	********	******0*	**00**00

SIMECK64/128	Left Input Differences (Bit position)			
Round	0–7	8–15	16–23	24–31
Δ^{39}	10000000	00000000	00000000	00000000
Δ^{40}	*1000*00	00000000	00000000	00000000
Δ^{41}	**100**0	00*00000	00000000	00000000
Δ^{42}	***10***	00**000*	00000000	00000000
Δ^{43}	****1***	*0***00*	*000*000	00000000
Δ^{44}	********	******0*	**00**00	0*000000

Likewise, the rightmost bit position $\ell+5$ at the round $t+1$ will propagate faults into three bits in the next round $t+2$ (*i.e.*, $\delta^{t+2}_{\ell+5}$, $\delta^{t+2}_{\ell+6}$ and $\delta^{t+2}_{\ell+10}$) and $\delta^{t+2}_{\ell+2} = \delta^{t+1}_{\ell+1} = 1$. The process will continue in the next round and so on.

□

Observation From Lemma 1, we have:

$$\text{If } \delta^t_j = 1 \ \&\ \delta^t_{j-5} = 0, \text{ then } x^t_{j-5} = \delta^{t+1}_j + \delta^t_{j-1} + \delta^{t-1}_j$$
$$\text{If } \delta^t_j = 0 \ \&\ \delta^t_{j-5} = 1, \text{ then } x^t_j = \delta^{t+1}_j + \delta^t_{j-1} + \delta^{t-1}_j$$

Table 4 shows the possibility to recover two bits x^t_{j-5} and x^t_j based on the relation of δ^t_j and δ^t_{j-5}.

4 Differential Fault Analysis on SIMECK ciphers

This section will describe our DFA attack against SIMECK lightweight block ciphers in the bit-flip model. Specifically, there will be one bit flipped when a fault is injected. Given a plaintext P, the SIMECK encryption function outputs the corresponding ciphertext C. Assume that a fault is injected into the input X^t of an intermediate round t and causes a bit-flip at the position ℓ.

Table 4. Recover x^t_{j-5} and x^t_j from Observation 3

δ^t_j	δ^t_{j-5}	δ^{t-1}_j	x^t_{j-5}	x^t_j
0	0	known/unknown	unknown	unknown
0	1	known	unknown	$\delta^{t+1}_j + \delta^t_{j-1} + \delta^{t-1}_j$
0	1	unknown	unknown	unknown
1	0	known	$\delta^{t+1}_j + \delta^t_{j-1} + \delta^{t-1}_j$	unknown
1	0	unknown	unknown	unknown
1	1	known/unknown	unknown	unknown

Let $\hat{X}^t$ be the fault value, so X^t and $\hat{X}^t$ will be different at the ℓ^{th} bit and identical everywhere else. In other words, if $(x^t_0, \ldots, x^t_{n-1})$ and $(\hat{x}^t_0, \ldots, \hat{x}^t_{n-1})$ are n-bits of X^t and $\hat{X}^t$, respectively, then $x^t_j = \hat{x}^t_j + 1$ for $j = \ell$ and $x^t_j = \hat{x}^t_j$ for $j \neq \ell$.

4.1 Attack Description

We aim at recovering the full master key K (equivalent to 4 round keys) by injecting faults into a single round only. Faults will be injected at the round $T - 5$ and we try to obtain 4 round keys $K^{T-1}, K^{T-2}, K^{T-3}$ and K^{T-4}. For instance, in order to retrieve 4 round keys of SIMECK32/64 (K^{28}, K^{29}, K^{30}, and K^{31}), we will inject faults into the round 27. Our analytic attack based on the differential trail works as follows (Algorithm 1):

Algorithm 1: DFA attack on SIMECK ciphers

Step 1: Choose a random plaintext P to feed into the SIMECK encryption function, and get a ciphertext C as return.

Step 2: Re-run encryption with the above input P, and then inject a fault into the left input at the round $T - 5$. Without loss of generality, we suppose that this fault flips the least significant bit x^{T-5}_0.

Step 3: Find input differences of the subsequent rounds $T-5 < t < T$, *i.e.*, $\Delta^t = X^t + \hat{X}^t$. These input differences can be determined due to Lemma 1 and could be represented by $0, 1$ or algebraic expressions of other input variables. (as shown in Table 2). The differences for SIMECK32/64 when the bit x^{27}_0 flipped are listed in Table 2. Such differences for SIMECK48/96 and SIMECK64/128 are listed in Fig 2. It can be seen that each round contains 2 linear expressions to express the differences between correct and faulty intermediate ciphertexts.

Step 4: Deduce bits of X^{T-2},

- From linear expressions: As Δ^T, Δ^{T-1}, and Δ^{T-2} are known, the attacker can deduce two bits of X^{T-2} by summing up linear equations in two round $T - 2$ and $T - 1$. For example, if a fault was injected into the position 0 at the round 27 of SIMECK32/64, as shown in Table 2, the attacker can obtain the values of x^{30}_8 and x^{30}_{14} by summing

up $\delta_8^{31} + \delta_7^{30}$, and $\delta_3^{31} + \delta_2^{30}$, respectively. Let ℓ be the fault position at the round $T-5$. As the fault will propagate to the position $\ell+3$ at the round $T-2$, the attacker will be able to deduce two bits $x_{\ell-2}^{T-2}$ and $x_{\ell+8}^{T-2}$.

– From Observation 3: The attacker knows Δ^{T-1}, Δ^{T-2} and some bits of Δ^{T-3} according to Lemma 2. Thus, if $\delta_j^{T-2} = 1$ and $\delta_j^{T-3} = 0$ (resp., $\delta_{j+5}^{T-3} = 0$), then thanks to Observation 3 the attacker can deduce two bits x_{j-5}^{T-2} (and x_{j+5}^{T-2}, resp.).

Step 5: After obtaining bits of X^{T-2}, the attacker can retrieve the corresponding bits of the last round key K^{T-1} due to Eq. (3). He/she repeats steps 2–4 with different flipped bit position to recover the full round key K^{T-1}.
Step 6: Decrypt the last round using K^{T-1} to get $(X^{T-1}, Y^{T-1})(= (X^{T-1}, X^{T-2}))$ and $(\hat{X}^{T-1}, \hat{Y}^{T-1}) = ((\hat{X}^{T-1}, \hat{X}^{T-2}))$. Again, as the values Δ^{T-1}, Δ^{T-2}, and X^{T-2} are known, the attacker could compute Δ^{T-3} due to Eq. (4). He/she then repeats steps 4 and 5 to obtain the round key K^{T-2}. The attacker continues this process to retrieve two more round keys K^{T-3} and K^{T-4}.
Step 7: With 4 round keys, the attacker could deduce the master key K of a SIMECK block cipher by the key schedule.

From Step 4, it can be seen that for each bit 1 of Δ^{T-2}, the attacker may recover 2 bits of X^{T-2} (corresponding to two bits of K^{T-1}). Likewise, once K^{T-1} is recovered, that attacker knows Δ^{T-2}, Δ^{T-3} and some bits of Δ^{T-4}, he then may be able to recover two bits of X^{T-3} (corresponding to two bits of K^{T-2}) with each bit 1 of Δ^{T-3} and so on.

Table 5. Comparisons of the Fault analysis attacks on SIMECK ciphers

SIMECK$2n/4n$	Nalla *et al.* [14]		Our work		
	Locations to recover master key	No of faults * (Last round key)	Locations to recover master key	No of faults (Last round key)	No of faults (Master key)
SIMECK32/64	30, 29, 28, 27	28.32	27	12.12	25.78
SIMECK48/96	34, 33, 32, 31	41.44	31	22.88	43.56
SIMECK64/128	42, 41, 40, 39	57.06	39	30.14	89.51

4.2 Fault Position

In this section, we will discuss how the attacker determines the position of a fault.

Determining fault position in SIMECK64/128. A fault will be injected into the round 39 at the position ℓ, that is, $\delta_\ell^{39} = 1$. An attacker will use on the following facts to deduce the value of ℓ:

– From Lemma 2, the input differences at the round 42 will have 16 consecutive zeros (see Table 3).

– The fault position ℓ will be propagated and shifted right three positions at the round 42, that is $\delta^{42}_{\ell+3} = 1$.
– There is a pattern $10 * * * 00 * *000 * \underbrace{00\ldots0}_{16\,digits}$, where the value of $*$ could be 1 or 0. The position of the first 1 in this pattern will be $\ell + 3$. This pattern consists of 29 digits out of 32 digits of Δ^{42}.

Determining fault position in SIMECK48/96. A fault will be injected into the round 31 at the position ℓ, that is, $\delta^{31}_{\ell} = 1$. Likewise, an attacker will use on the following facts to deduce the value of ℓ:

– From Lemma 2, the input differences at the round 34 will have 8 consecutive zeros (see Appendix 2).
– The fault position ℓ will be propagated and shifted right three positions at the round 34, that is $\delta^{34}_{\ell+3} = 1$.
– There is a pattern $10 * * * 00 * *000 * \underbrace{00\ldots0}_{8\,digits}$, where the value of $*$ could be 1 or 0. The position of the first 1 in this pattern will be $\ell + 3$. This pattern consists of 21 digits out of 24 digits of Δ^{34}.

Determining fault position in SIMECK32/64. As there is no consecutive bit zero determined by Lemma 2, determining the fault position for SIMEC 32/64 is challenging. However, we still have this pattern $10 * * * 00 * *000$ in 16 bits of Δ^{30}, where the value of $*$ could be 1 or 0. The first 1 in this pattern will be $\ell + 3$, where j is the fault position injected at the round 27.

Based on the above facts and given that Δ^{T-2} is known, the attacker also can deduce the fault position ℓ at the round $T - 5$ with high confidence.

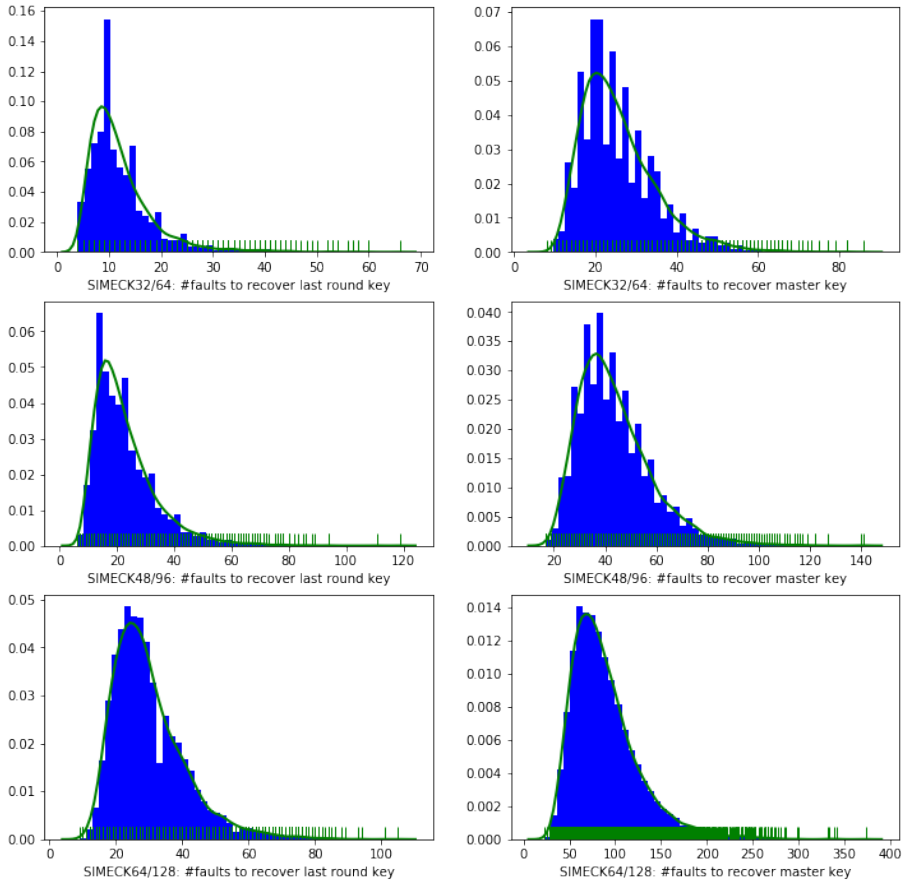
5 Experiments

5.1 Setup

To verify our proposed attack, we implemented a software simulation against all three SIMECK family members in C programming language[3]. In this simulation, we suppose that the attacker cannot control the position of the faults. He thus may inject faults into the same position many times until fully recovering one round key or the master key. Once a *random* fault is injected into the left input of the round i at the bit j, only the differential input δ^i_j will be equal to one, differential inputs at other positions will be zero. The positions of the faults were generated *randomly* and *independently*.

For each cipher member, we repeat the experiment 10,000 times and report the average number of faulty ciphers required to recover the last round key K^{T-1} and the whole master key (corresponding to the last 4 round keys K^{T-1}, K^{T-2}, K^{T-3}, and K^{T-4}). The only purpose of recovering the last key is to compare our attacks to the ones in [14].

Fig. 3. Histogram of the number of faults required to recover keys. The number of samples is 10,000. Frequency density is represented on the vertical axis and the number of fault injections is represented on the horizontal axis.



5.2 Simulation Results

Fig. 3 shows a histogram of the number of fault injections to obtain the last round key and the full master key of all three variants of SIMECK block cipher. The horizontal axis represents the number of faults, and the vertical axis represents the frequency experiments requiring that number of faults. The total number of experiments is 10,000, as mentioned in the previous section. As seen in Fig. 3, all histograms are right-skewed. In order to recover the last round key in SIMECK32/64, it requires from a few fault injections to about 70 fault injections. However, most of the experiments only required around 9-13 fault injections. Likewise, the maximum number of fault injections required to recover the full master key is about 90; however, most of the experiments only required around 19-25 fault injections.

Table 5 compares the round locations, in which faults are induced, the number of faults required between our attacks, and Nalla *et al.*'s attack under the one bit-flip model. From Table 5, it can be seen that our fault analysis attack requires much fewer faults than Nalla *et al.*'s attack to recover the last round key for all three members of SIMECK. Even more, our attack to recover the whole master key of SIMECK32/64 also requires fewer faults than theirs for only the last round key. Last but not least, in order to recover the whole master key, while our attack injects the faults into a single round only, Nalla *et al.*'s attack has to inject faults into 4 different rounds of a SIMECK cipher. As a result, our attack is more *practical*.

5.3 Discussions

The proposed attack is based on an analytic method, which analyzes the differential trail between the correct and faulty ciphertexts to deduce the intermediate inputs, and then round keys. The key point in this method is to find the differential trail that is not too complicated to analyze. Compared to theoretical attacks [15, 17, 19, 20], this method is more straightforward, but we demonstrated that the attack is *effective*. In this paper, we use only linear expressions to analyze, however, non-linear expressions (e.g., quadratic expressions denoted as $*$ in Table 2 and Fig. 2) may be useful for more-in-depth analyses. Block ciphers with more complicated round functions would be resistant to the proposed attack, e.g., using a non-linear S-box. However, this will trade-off with the performance of the implementation.

6 Conclusion

In this paper, we have proposed an improved fault attack on the SIMECK lightweight block ciphers under the one bit-flip model. In this model, we assume that a single bit will be flipped to its complementary value once a fault was successfully injected. We also assume that the attacker has no control over the location of the flipped bit. The advantage of our attack is that not only it requires less number of faults, but also faults need to be injected into only a *single round* of the ciphers in order to recover the whole master key. As a result, it makes our attack more *practical*. We demonstrated the effectiveness of our

[3] The source code could be found in the link `https://github.com/dple/DFA_Simeck`

attack by simulating in C for all three members of SIMECK ciphers. Our experimental results over 10, 000 times showed that the attack requires 25.78, 43.56, and 89.51 faults on average to recover the full master key of the SIMECK32/64, SIMECK48/96, and SIMECK64/128, respectively.

References

1. G. Yang, B. Zhu, V. Suder, M. D. Aagaard, and G. Gong, "The Simeck Family of Lightweight Block Ciphers," in *Cryptographic Hardware and Embedded Systems – CHES 2015*. Berlin, Heidelberg: Springer, 2015, pp. 307–329.
2. R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks, and L. Wingers, "The simon and speck families of lightweight block ciphers." *IACR Cryptology ePrint Archive*, 2013.
3. O. Kömmerling and M. G. Kuhn, "Design principles for tamper-resistant smartcard processors," in *Proceedings of the USENIX Workshop on Smartcard Technology on USENIX Workshop on Smartcard Technology*, ser. WOST'99. Berkeley, CA, USA: USENIX Association, 1999, pp. 2–2.
4. S. P. Skorobogatov and R. J. Anderson, "Optical fault induction attacks," in *Revised Papers from the 4th International Workshop on Cryptographic Hardware and Embedded Systems*, ser. CHES '02. London, UK, UK: Springer-Verlag, 2003, pp. 2–12.
5. D. Boneh, R. A. DeMillo, and R. J. Lipton, "On the importance of checking cryptographic protocols for faults," in *Proceedings of the 16th Annual International Conference on Theory and Application of Cryptographic Techniques*, ser. EUROCRYPT'97. Berlin, Heidelberg: Springer-Verlag, 1997, pp. 37–51.
6. E. Biham and A. Shamir, "Differential fault analysis of secret key cryptosystems," in *Proceedings of the 17th Annual International Cryptology Conference on Advances in Cryptology*, ser. CRYPTO '97. London, UK, UK: Springer-Verlag, 1997, pp. 513–525.
7. C. Giraud, "Dfa on aes," in *Proceedings of the 4th International Conference on Advanced Encryption Standard*, ser. AES'04. Berlin, Heidelberg: Springer-Verlag, 2005, pp. 27–41.
8. C. H. Kim, "Improved differential fault analysis on aes key schedule," *IEEE transactions on information forensics and security*, vol. 7, no. 1, pp. 41–50, 2011.
9. L. Hemme, "A differential fault attack against early rounds of (triple-)des," in *Cryptographic Hardware and Embedded Systems - CHES 2004*, M. Joye and J.-J. Quisquater, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 254–267.
10. C. Clavier, B. Gierlichs, and I. Verbauwhede, "Fault analysis study of idea," in *Topics in Cryptology – CT-RSA 2008*, T. Malkin, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 274–287.
11. H. Tupsamudre, S. Bisht, and D. Mukhopadhyay, "Differential fault analysis on the families of simon and speck ciphers," in *2014 Workshop on Fault Diagnosis and Tolerance in Cryptography*. IEEE, 2014, pp. 40–48.
12. D.-P. Le, S. L. Yeo, and K. Khoo, "Algebraic differential fault analysis on simon block cipher," *IEEE Transactions on Computers*, vol. 68, no. 11, pp. 1561–1572, November 2019.
13. M. Gruber and B. Selmke, "Differential fault attacks on KLEIN," in *Constructive Side-Channel Analysis and Secure Design - 10th International Workshop, COSADE 2019, Darmstadt, Germany, April 3-5, 2019, Proceedings*, ser. Lecture Notes in Computer Science, I. Polian and M. Stöttinger, Eds., vol. 11421. Springer, 2019, pp. 80–95. [Online]. Available: https://doi.org/10.1007/978-3-030-16350-1_6
14. V. Nalla, R. A. Sahu, and V. Saraswat, "Differential fault attack on simeck," in *Proceedings of the Third Workshop on Cryptography and Security in Computing Systems*, ser. CS2 '16. New York, NY, USA: ACM, 2016, pp. 45–48. [Online]. Available: http://doi.acm.org/10.1145/2858930.2858939

15. N. Bagheri, "Linear cryptanalysis of reduced-round simeck variants," in *International Conference on Cryptology in India*. Springer, 2015, pp. 140–152.
16. S. Kölbl and A. Roy, "A brief comparison of simon and simeck," in *International Workshop on Lightweight Cryptography for Security and Privacy*. Springer, 2016, pp. 69–88.
17. K. Qiao, L. Hu, and S. Sun, "Differential analysis on simeck and simon with dynamic key-guessing techniques," in *International Conference on Information Systems Security and Privacy*. Springer, 2016, pp. 64–85.
18. L. Qin, H. Chen, and X. Wang, "Linear hull attack on round-reduced simeck with dynamic key-guessing techniques," in *Australasian Conference on Information Security and Privacy*. Springer, 2016, pp. 409–424.
19. K. Zhang, J. Guan, B. Hu, and D. Lin, "Integral cryptanalysis on simeck," in *2016 Sixth International Conference on Information Science and Technology (ICIST)*. IEEE, 2016, pp. 216–222.
20. ——, "Security evaluation on simeck against zero-correlation linear cryptanalysis," *IET Information Security*, vol. 12, no. 1, pp. 87–93, 2017.
21. S. Sadeghi and N. Bagheri, "Improved zero-correlation and impossible differential cryptanalysis of reduced-round simeck block cipher," *IET Information Security*, vol. 12, no. 4, pp. 314–325, 2018.
22. E. Biham and A. Shamir, "Differential cryptanalysis of des-like cryptosystems," *Journal of CRYPTOLOGY*, vol. 4, no. 1, pp. 3–72, 1991.
23. M. Matsui, "Linear cryptanalysis method for des cipher," in *Workshop on the Theory and Application of of Cryptographic Techniques*. Springer, 1993, pp. 386–397.
24. National Institute of Standards and Technology, "FIPS-46: Data Encryption Standard (DES)," 1977, revised as FIPS 46-1:1988, FIPS 46-2:1993, FIPS 46-3:1999, available at http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf.
25. D. Samyde, S. Skorobogatov, R. Anderson, and J.-J. Quisquater, "On a new way to read data from memory," in *Proceedings First International IEEE Security in Storage Workshop(SISW)*, vol. 00, 12 2002, p. 65.
26. L. Rivière, J. Bringer, T.-H. Le, and H. Chabanne, "A novel simulation approach for fault injection resistance evaluation on smart cards," in *2015 IEEE Eighth International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*. IEEE, 2015, pp. 1–8.