Practical Differential Fault Attacks on the GPRS Standard Ciphers

Zhengting Li, Lin Ding, An Wang, Haotong Xu, Zheng Liu, Zheng Wu, Xinhai Wang and Jiang Wan

***Abstract*—GEA-1 and GEA-2 are two standard stream ciphers used in GPRS (General Packet Radio Service) to protect against eavesdropping GPRS between the base station and the phone. Now, a range of current phones still support them. In this paper, a differential fault attack on the GEA-like stream ciphers under the random fault model is proposed for the first time. In this attack, an efficient dedicated algorithm for identifying the exact fault location is proposed. By using this dedicated algorithm, the attacker can succeed in determining the exact fault location. As applications, practical differential fault attacks on the GPRS standard ciphers (i.e., GEA-1 and GEA-2) are presented, which recover the 64-bit secret keys of GEA-1 and GEA-2 with time complexities of $2^{33.807}$ and $2^{33.858}$, respectively. We validate the cryptanalytic results by simulating the whole attacks on the platform ChipWhisperer Lite. The experimental results show that both GEA-1 and GEA-2 can be broken within sixteen minutes on a common laptop. Finally, the possible countermeasures are presented to protect the processed data of massive GPRS devices.**

***Index Terms*—Differential fault attack, Countermeasure, Standard cipher, GEA-1, GEA-2, General Packet Radio Service.**

I. Introduction

GPRS (General Packet Radio Service) is a packet oriented wireless data communication service for mobile communications on 2G and 3G cellular communication systems. It is a mobile data standard established by the European Telecommunications Standards Institute (ETSI), and was widely deployed during the late 1990s and the early 2000s. To protect the data processed by the GPRS devices against eavesdropping between the base station and the phone, a proprietary stream cipher named GEA-1 was used for this purpose. It was initially designed by ETSI Security Algorithms Group of Experts (SAGE) in 1998. A technical report on the design process of GEA-1 is available at [1]. GEA-2, designed in 1999, is an improvement of GEA-1 [2]. Both GEA-1 and GEA-2 take a 64-bit secret key, a 32-bit initial vector (IV) and a flag bit to indicate the transfer direction as input and output a keystream no more than 1600 bytes (i.e., 12800 bits).

The ETSI prohibited the implementation of GEA-1 in mobile phones in 2013, while it is still mandatory to implement GEA-2 today [3]. In [4], Beierle et al. tested a range of current phones if they follow the specification regarding disabling GEA-1, such as Apple iPhone XR, Apple iPhone 8, Samsung Galaxy S9, HMD Global Nokia 3.1, Huawei P9 lite and OnePlus 6T. Astonishingly, all tested phones disrespect the specification and support the vulnerable cipher GEA-1. As described in [4], [5], this could have severe implications as it opens the door for various types of downgrade attacks. As for GEA-2, the ETSI followed the mid-term goal to prohibit the implementation of GEA-2 in mobile phones. However, this usually requires the consent of several parties and may take a long time. In other words, GEA-2 will still be implemented in current phones to protect the processed data for a long time. Thus, the security of widely-deployed GEA-2 and GEA-1 directly affects the security of massive GPRS devices' communication.

Related works. In EUROCRYPT 2021, Beierle et al. [4] presented the first public cryptanalytic analysis of GEA-1 and GEA-2. They discovered a weakness in the initialization process of GEA-1, i.e., the joint state of two of three LFSRs has the joint entropy of only 40 bits, whereas their joint state has a size up to 64 bits. Based on this weakness, a practical meet-in-the-middle (MITM) attack on GEA-1 was proposed with a time complexity of 2^{40}, requiring a memory space of 44.5 GiB and 65 bits of known keystream. Since their attack can recover the 64-bit secret key of GEA-1, it makes it possible to decrypt the entire GPRS session. Through in-depth analysis, the unusual weakness of GEA-1 is believed to have been intentionally introduced and hidden by the designers, presumably due to the strict export regulations on cryptography that were in effect when GEA-1 was designed. In contrast, the weakness of GEA-1 is not present in GEA-2. However, they still proposed an attack on GEA-2 with a time complexity of $2^{45.1}$, requiring a memory space of 32 GiB and 12800 bits of known keystream used to encrypt a full GPRS frame. Since it is impractical or even impossible to implement the attacks on GEA-1 and GEA-2 on a common laptop due to the high memory consumption, Beierle et al. executed their attacks on a cluster made up of four AMD EPYC 7742 64-Core Processors. The experimental results show that it has to take about 25 minutes to recover the secret key of GEA-1, while roughly four months are required for GEA-2. In the follow-up works [6], [7], Beierle et al. analyzed the design of GEA-1 further and answered the open question that how to construct a GEA-1-like cipher with such a reduced security.

In EUROCRYPT 2022, Amzaleg and Dinur [8] improved and complemented the previous cryptanalytic analysis of

Manuscript received 16 September 2025; revised 13 January 2026; accepted 5 March 2026. This work was supported by the National Natural Science Foundation of China under Grant 62472439. (*Corresponding author*: *Lin Ding*.)

Zhengting Li, Lin Ding, Zheng Wu, Xinhai Wang and Jiang Wan are with the Information Engineering University, Zhengzhou 450001, China (e-mails: lizhengting0225@163.com, dinglin_cipher@163.com, g7001162@163.com, wxh1066559569@126.com, wanjiang0626@163.com).

An Wang, Haotong Xu and Zheng Liu are with the School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing 100081, China (e-mails: wanganl@bit.edu.cn, haotongxu@bit.edu.cn, liuzheng98@bit.edu.cn).

GEA-1 and GEA-2 in [4]. For GEA-1, an improved attack was proposed, where the memory space required is reduced from 44.5 GiB to 4 MiB, while the time complexity remains 2^{40}. Benefitting from significantly reduced memory consumption, this attack on GEA-1 can be implemented on a common laptop in average time of 2.5 hours. For GEA-2, two new attacks were presented. The first attack has a time complexity of $2^{64}/(l-62)$, requiring a memory space of 64 GiB and l bits of consecutive keystream. The second attack has a time complexity of 2^{55}, requiring a memory space of 32 MiB and 11320 bits of fragmented keystream. Although the required keystream bits in these two attacks on GEA-2 are less than 12800 bits, but the time complexities are so large that cannot be implemented on a common laptop.

Recently, Ding et al. [5] presented time-memory-data trade-off attacks on GEA-1 and GEA-2, respectively. The attack on GEA-1 has an online/offline time complexity of $2^{26}/2^{32}$, requiring a memory space of 8 MiB. However, this attack needs 2^{38} bits of known keystream generated by the fixed unknown key and 2^{32} IVs. This data complexity is so huge that makes it impractical or even impossible for the attacker to collect from the GPRS devices. The attack on GEA-2 has an online time complexity of $2^{48.4}$, requiring a memory space of 42 TiB. However, this attack has an offline time complexity of 2^{65} that is clearly impractical to implement on a common laptop. Besides, it needs $2^{38.6}$ bits of known keystream generated by the fixed unknown key and 2^{32} IVs, which is also impractical or even impossible for the attacker to collect from the GPRS devices. Besides, they analyzed the slide properties of GEA-1 and GEA-2 and then proposed practical related key attacks on them. However, as remarked in [9], '*related-key attacks on stream cipher are often of limited practical significance*', since they are based on a unrealistic assumption that the encryption is performed using two different keys that have a relationship known to the attacker, but the values of these keys are unknown. Thus, these related-key attacks cannot be considered as real threats on GEA-1 and GEA-2, due to the unrealistic attack assumption.

Differential fault attack. Cryptanalysis is the field of analyzing cryptographic systems to detect weaknesses or information leaks. It is generally thought of as exploring the weaknesses of the underlying mathematics of a cryptographic system but it also includes looking for weaknesses in implementation, such as side channel attacks [10]. Side-channel attacks represent a powerful category of attacks against cryptographic devices, and include a variety of attacks, such as simple timing attack [11], differential power attack [12], correlation power attack [13], template attack [14] and fault attack [15]. In a fault attack, the attacker can inject faults during a cryptographic operation and exploit the faulty results to infer private information (e.g., the secret key). Fault attack was first introduced by Boneh et al. [15] by describing an attack on the RSA public key system. Subsequently, Biham and Shamir [16] extended such attack on symmetric ciphers and called it differential fault attack. Differential fault attack can be thought of as a combination of the classical differential attack [17] and fault attack, and had been successfully applied to the well-known block ciphers DES [16], [18], [19] and AES [20]–[25].

In the domain of stream ciphers, the first fault attack was introduced by Hoch and Shamir in [26]. After then, differential fault attacks on various well-known stream ciphers have been proposed, such as RC4 [27], Grain Family [28]–[31], Trivium family [32], [33], MICKEY family [34], SOSEMANUK [35], HC-128 [36], Salsa [37], ChaCha [37], WG family [38], Plantlet [39], SNOW 3G [40], Kreyvium [41], FLIP [41] and Rasta [42]. These cryptanalytic results show that differential fault attack is very efficient against stream ciphers and has become one of the potent practical threats to modern cryptographic implementations. In the meantime, the fault-injection techniques for practical attacks were also developed in many ways, and there are many ways to inject faults [43], e.g., laser shots, voltage glitches, clock glitches, ionizing radiation, chip underpowering and temperature increase. The most common technique of injecting faults is by using laser shots or voltage glitches. Recently, to answer the question "How practical are fault injection attacks", Breier and Hou [44] analyzed the practicality of fault attack by testing the most commonly used techniques of injecting faults. The result shows that fault attacks can be successfully mounted on most commonly used architectures (e.g., ARM, Intel and AMD), by utilizing injection devices that are often below the thousand dollar mark. Thus, fault attacks can be considered practical in many scenarios, especially when the attacker can physically access the target device.

Recently, besides the physical fault injection techniques, some modern techniques for remote fault injection have been proposed, e.g., Rowhammer [45], ZK-Hammer [46], Achilles [47] and HyperHammer [48]. These modern techniques offer new choices for the fault injection and make the differential fault attack more powerful.

Our contributions. This paper for the first time analyzes the security of the GPRS standard ciphers (i.e., GEA-1 and GEA-2) against differential fault attack under the random fault model, which is the most widely accepted model of fault attack. In this attack model, the attacker is allowed to induce faults into the internal state bits, while she does not know the exact location during fault injection.

Firstly, we give a high-level description of the GEA-like stream ciphers and introduce the attack model used in this paper. Secondly, a differential fault attack on the GEA-like stream ciphers is proposed. The proposed differential fault attack consists of two processes, i.e., identifying the exact fault location and recovering the internal state. To identify the exact fault location, an efficient dedicated algorithm is proposed for the GEA-like stream ciphers. By using this algorithm, the attacker can succeed in determining the exact fault location. As applications, practical differential fault attacks on GEA-1 and GEA-2 are proposed for the first time, which enable the attacker to recover the 64-bit secret keys of GEA-1 and GEA-2 with time complexities of $2^{33.807}$ and $2^{33.858}$, respectively. To validate the cryptanalytic results, the whole attacks have been simulated on the platform ChipWhisperer Lite. The experimental results show that both GEA-1 and GEA-2 can be broken within sixteen minutes on a laptop.

Thus, GEA-1 and GEA-2 are indeed weak against differential fault attacks. To the best of our knowledge, this is the first practical demonstration of differential fault attacks on the GPRS standard ciphers to be reported in the literature. Finally, to mitigate our attacks on the GPRS standard ciphers, the possible countermeasures are presented. At the cipher design level, a new variant of GEA-2 named GEA-2M is proposed as a concrete countermeasure. The new variant GEA-2M has significantly enhanced the resistance against all existing attacks (including our differential fault attacks) and provides a better security margin than GEA-2.

The key reason why our attacks on the GPRS standard ciphers can be successful is that the feedback shift registers are updated linearly and independently in the keystream generation process. This prevents the differential propagations from one feedback shift register to another, which significantly weakens the resistance of the GPRS standard ciphers against differential fault attacks. Thus, our cryptanalytic attacks and the design of GEA-2M could give some new insights into how to design a more secure GEA-like stream cipher, and thus might be helpful for the cipher designers.

Technical novelty of our attacks. The proposed attacks in this paper follow the basic idea of differential fault attack. The attacks are not completely new, since some of their ideas were also used in previous attacks on various cryptosystems, but we believe that this is the first time that the signature vector is introduced to identify the exact fault location algorithmically and then an efficient dedicated algorithm for a kind of stream ciphers is proposed. Based on this algorithm, the attacker can succeed in determining the exact fault location with a high probability. Since successfully identifying the exact fault location is a precondition for an efficient differential fault attack, this paper could give some new insights into how to mount an efficient differential fault attack. More importantly, the idea is general and may be applied to other cryptosystems.

Practicality of our attacks. It is important to note that both attacks on GEA-1 and GEA-2 do not rely on strong assumptions about the attacker's capabilities, and achieve practical cryptanalytic results, i.e., recovering the secret keys with practical time complexities and high success rates. The practicality of our differential fault attacks on the GPRS standard ciphers can be confirmed from the following three aspects.

- Firstly, as tested in [4], a range of current phones support the GPRS standard ciphers GEA-1 and GEA-2. This provides a great convenience for the attacker to physically access the target device. In fact, even if the attacker can not physically access the target device and the physical fault injection techniques do not work, the attacker could consider the remote fault injection techniques proposed in [45]–[48], which enhance the capability of the attacker to execute a practical differential fault attack.
- Secondly, as analyzed in [44], injecting faults into a target device can be successfully achieved by utilizing some commonly used techniques and inexpensive injection devices. It indicates that injecting faults is completely feasible to mount differential fault attacks on the GPRS standard ciphers.
- Finally, as simulated on the platform ChipWhisperer Lite, recovering the 64-bit secret keys of GEA-1 and GEA-2 can be done within sixteen minutes on a common laptop.

Summarily, considering the three aspects above, the differential fault attacks on the GPRS standard ciphers are indeed practical on a common laptop.

Impact of our attacks. As Beierle et al. [4] stated, "*While some countries are about to sunset 2G technology (or have already done so), other countries rely on GPRS as a fallback data connection. Consequently, the security of those connections was and still is relevant for a large user base.*" Since our attacks can effectively recover the 64-bit secret keys of GEA-1 and GEA-2 practically, the impact of our attacks on the massive phone users should be taken seriously. That is because, '*once the secret key is recovered, the attacker can decrypt all traffic for the complete GPRS session until the key gets invalid, which happens in the GPRS authentication and ciphering procedure triggered by the network*,' as analyzed in [4]. This applies that the attacker can eavesdrop on the communications between the phone and the base station by the proposed attacks, which clearly poses a serious threat to the privacy of massive phone users. Thus, since the GPRS technology is still used in large scale, the security of the GPRS standard ciphers is still a widely concerned problem. This motivates us to thoroughly analyze the resistance of the GPRS standard ciphers against differential fault attacks, which can lead to a better understanding of the practical security of these ciphers. Due to the significant effectiveness of our attacks, some necessary countermeasures should be adopted when implementing GEA-1 and GEA-2 to protect the processed data of massive GPRS devices.

Outline. The organization of the paper is as follows. A brief description of the GPRS standard ciphers is given in Section II. In Section III, a high-level description of the GEA-like stream ciphers is given, the attack model is described and a differential fault attack on the GEA-like stream ciphers is proposed. The practical differential fault attacks on GEA-1 and GEA-2 are presented in Sections IV and V, respectively. The countermeasures to mitigate our attacks on the GPRS standard ciphers are presented in Section VI. Section VII concludes the paper.

II. A Brief Description of the GPRS Standard Ciphers

In this section, we will briefly describe the GPRS standard ciphers. It should be noted that the details of the initialization phases of these ciphers are omitted here, since they are irrelevant to our attacks. For the complete descriptions, we refer the reader to [4]. Since GEA-2 is a simple successor of GEA-1, we first introduce GEA-1 and then describe the modifications made in GEA-2. An overview of the keystream generators of GEA-1 and GEA-2 is depicted in Fig. 1 [4]. Note that the register D is only present in GEA-2.

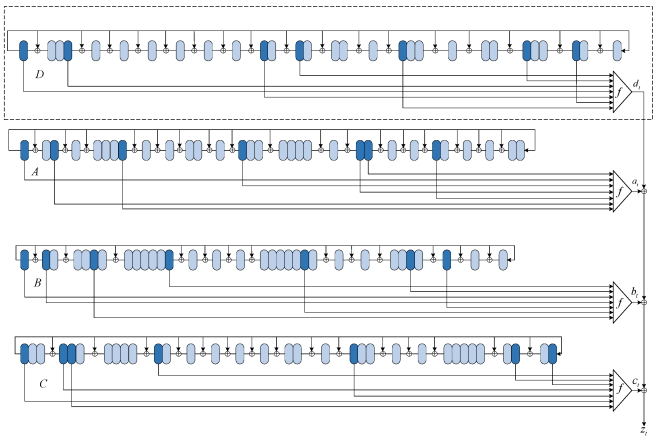

Fig. 1. The keystream generators of GEA-1 and GEA-2. The register D is only present in GEA-2.

A. The GEA-1 Stream Cipher

As depicted in Fig. 1, the keystream generator of GEA-1 logically consists of three linear feedback shifts registers (LFSRs) and a linear keystream output function. It supports a 64-bit secret key and a 32-bit IV, and has an internal state size of 96 bits.

All of the three LFSRs of GEA-1 named A, B and C are defined over the finite field $\mathbb{F}_2$, and consist of 31, 32 and 33 bits in length, respectively. Let $\left(a_0^{(t)}, \cdots, a_{30}^{(t)}\right)$, $\left(b_0^{(t)}, \cdots, b_{31}^{(t)}\right)$ and $\left(c_0^{(t)}, \cdots, c_{32}^{(t)}\right)$ denote the states of the LFSRs A, B and C at time t, respectively. The recurrence relations of these three LFSRs are given as follows.

LFSR A:

$$a_i^{(t+1)} = \begin{cases} a_{i+1}^{(t)} \oplus a_0^{(t)}, & if\ i \in T_A \\ a_{i+1}^{(t)}, & if\ i \in \{0, \cdots, 29\} - T_A \\ a_0^{(t)}, & if\ i = 30 \end{cases}$$

LFSR B:

$$b_i^{(t+1)} = \begin{cases} b_{i+1}^{(t)} \oplus b_0^{(t)}, & if\ i \in T_B \\ b_{i+1}^{(t)}, & if\ i \in \{0, \cdots, 30\} - T_B \\ b_0^{(t)}, & if\ i = 31 \end{cases}$$

LFSR C:

$$c_i^{(t+1)} = \begin{cases} c_{i+1}^{(t)} \oplus c_0^{(t)}, & if\ i \in T_C \\ c_{i+1}^{(t)}, & if\ i \in \{0, \cdots, 31\} - T_C \\ c_0^{(t)}, & if\ i = 32 \end{cases}$$

where the three tap sets T_A, T_B, T_C used above are given as follows.

$$T_A = \{0, 2, 3, 7, 8, 9, 11, 12, 15, 19, 20, 22, 23, 24, 26, 27, 28\}$$
$$T_B = \{0, 2, 6, 12, 13, 14, 15, 22, 23, 24, 28, 29, 30\}$$
$$T_C = \{2, 5, 9, 11, 12, 13, 14, 15, 17, 18, 21, 22, 23, 28, 30\}$$

After the initialization process, GEA-1 starts generating the keystream bits $\{z_t\}_{t\geq 0}$ by

$$z_t = a_t \oplus b_t \oplus c_t,\ t \geq 0$$

where a_t, b_t, c_t are three output bits from the same nonlinear filter function f by taking seven specified bits in each LFSR as input. Here, f is a Boolean function from $\mathbb{F}_{2^7}$ to $\mathbb{F}_2$, and has an algebraic degree of 4. The algebraic normal form (ANF) of f is given as

$$\begin{aligned} f(x_0, \cdots, x_6) = {} & x_0x_1x_5x_6 \oplus x_0x_2x_3x_6 \oplus x_0x_2x_5x_6 \oplus x_0x_3x_5x_6 \\ & \oplus x_1x_2x_5x_6 \oplus x_1x_3x_4x_6 \oplus x_1x_3x_5x_6 \oplus x_0x_1x_3 \\ & \oplus x_0x_1x_4 \oplus x_0x_1x_6 \oplus x_0x_2x_3 \oplus x_0x_2x_4 \oplus x_0x_2x_6 \\ & \oplus x_0x_3x_5 \oplus x_1x_2x_5 \oplus x_1x_2x_6 \oplus x_1x_4x_6 \oplus x_2x_5x_6 \\ & \oplus x_0x_2 \oplus x_0x_3 \oplus x_0x_5 \oplus x_1x_3 \oplus x_1x_5 \oplus x_1x_6 \\ & \oplus x_2x_3 \oplus x_2x_5 \oplus x_2x_6 \oplus x_4x_5 \oplus x_5x_6 \oplus x_1 \oplus x_2 \\ & \oplus x_3 \oplus x_5 \end{aligned}$$

As shown in Fig. 1, the three output bits a_t, b_t, c_t can be generated as follows.

$$\begin{aligned} a_t &= f\left(a_{22}^{(t)}, a_0^{(t)}, a_{13}^{(t)}, a_{21}^{(t)}, a_{25}^{(t)}, a_2^{(t)}, a_7^{(t)}\right) \\ b_t &= f\left(b_{12}^{(t)}, b_{27}^{(t)}, b_0^{(t)}, b_1^{(t)}, b_{29}^{(t)}, b_{21}^{(t)}, b_5^{(t)}\right) \\ c_t &= f\left(c_{10}^{(t)}, c_{30}^{(t)}, c_{32}^{(t)}, c_3^{(t)}, c_{19}^{(t)}, c_0^{(t)}, c_4^{(t)}\right) \end{aligned}$$

B. The GEA-2 Stream Cipher

GEA-2 is a simple extension of GEA-1 and has similar structure with GEA-1, as depicted in Fig. 1. GEA-2 adds a fourth LFSR named D of length 29. Let $\left(d_0^{(t)}, \cdots, d_{28}^{(t)}\right)$ denote the state of LFSR D at time t, and the recurrence relation of the fourth LFSR is given as follows.

$$d_i^{(t+1)} = \begin{cases} d_{i+1}^{(t)} \oplus d_0^{(t)}, & if\ i \in T_D \\ d_{i+1}^{(t)}, & if\ i \in \{0, \cdots, 27\} - T_D \\ d_0^{(t)}, & if\ i = 28 \end{cases}$$

where the tap sets T_D used above is given as follows.

$$T_D = \{0, 3, 4, 5, 6, 7, 8, 9, 11, 13, 15, 16, 19, 20, 22, 25, 27\}$$

After the initialization process, GEA-2 starts generating the keystream bits $\{z_t\}_{t\geq 0}$ by

$$z_t = a_t \oplus b_t \oplus c_t \oplus d_t,\ t \geq 0$$

where a_t, b_t, c_t are the same as in GEA-1, and the bit d_t is generated as follows.

$$d_t = f\left(d_{12}^{(t)}, d_{23}^{(t)}, d_3^{(t)}, d_0^{(t)}, d_{10}^{(t)}, d_{27}^{(t)}, d_{17}^{(t)}\right)$$

III. Differential Fault Attack on the GEA-Like Stream Ciphers

In this section, a differential fault attack on the GEA-like stream ciphers is proposed. Before presenting our attack, a high-level description of the GEA-like stream ciphers is introduced as follows.

A. A High-Level Description of the GEA-Like Stream Ciphers

The GEA-like stream cipher consists of n LFSRs defined over the finite field $\mathbb{F}_2$, denoted as $L_1, \cdots, L_n$. They contain $l_1, \cdots, l_n$ bits, respectively. The internal state of L_i at time t can be described as $X_i^{(t)} = (x_{i,0}^{(t)}, \cdots, x_{i,l_i-1}^{(t)})$.

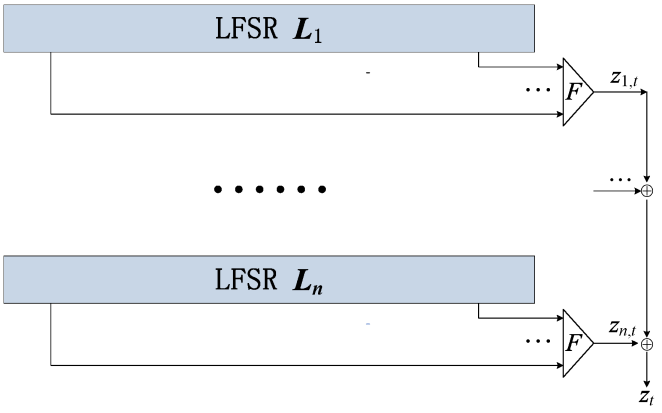

Fig. 2. The keystream generators of the GEA-like stream cipher

As shown in Fig. 2, the keystream bit z_t is generated by XORing n output bits $z_{1,t}, \cdots, z_{n,t}$ from n LFSRs, i.e.,

$$z_t = z_{1,t} \oplus \cdots \oplus z_{n,t}$$

where $z_{i,t}$ is the output bit generated by the nonlinear filter function F by taking several bits in the LFSR L_i as input, i.e., $z_{i,t} = F(X_i^{(t)})$. Since our attack is irrelevant to the algebraic expression of F, we do not make any limitation to this nonlinear filter function.

B. The Attack Model

As with every differential fault attack, certain capabilities are required for the attacker, such as the ability to restart the encryption device. Obviously, the fewer capabilities are made by the attacker the better, since the differential fault attack will become unrealistic if the attacker is granted too many capabilities. In this paper, the differential fault attack on the GEA-like stream ciphers follows a standard random fault model, which is the most commonly fault attack model used in the literature. The faults can be injected by using voltage glitches, laser shots, clock glitches, ionizing radiation, etc. In this attack model, the attacker is assumed to have the following capabilities which are generally acceptable in cryptanalytic literature:

(a) the attacker can reinitialize the encryption operation multiple times with the same key-IV pair.
(b) the attacker can inject a fault that flips a state bit from 0 to 1 or 1 to 0, while she does not know the exact location during fault injection.

The practicality of our attack model. We will illustrate the practicality of the attack model used in our paper from the following three aspects:

- The random fault model is one of the most widely recognized fault models and considered to be the most relaxed fault model to achieve a successful differential fault attack [49]. Although the random fault attack model appears to be more complicated to implement in a practical scenario compared with the differential attack, several well-known ciphers like DES [16], [18], [19], AES [20]–[25] and RSA [15] had been successfully broken by the attack. Besides, differential fault attacks on several popular stream ciphers such as Grain-128, Trivium, ChaCha and SNOW 3G had also been successfully proposed in [28]–[33], [37], [40]. Most of these attacks follow the same attack model with this paper. There are some attacks (e.g., the attacks on Grain-128 in [28]–[30]) that adopt a rather optimistic attack model, assuming that the attacker has the ability to inject a single bit fault in the same location over and over again. Compared with this kind of attack model, the random fault attack model used in this paper is clearly more practical.
- Reinitializing the encryption operation multiple times with the same key-IV pair is realistic in our attack model. As noted in [50], the adversary under the random fault model is certainly more powerful than a normal passive attacker and has the ability to access the keystream generator as a physical device. For a stream cipher that utilizes a resynchronization mechanism where the keystream frame is generated from a master key and a public IV, the adversary can feed the same IV into the device several times, which could successfully reinitialize the encryption operation multiple times with the same key-IV pair. In particular, this is completely realistic for the GEA-like standard ciphers, as multiple keystream frames generated from the same key are undoubtedly normal. This is because the length of one keystream frame is strictly limited to 1600 bytes for one key-IV pair, and multiple keystream frames should be generated when the plaintext contains more than 1600 bytes in length. Since the IV is public, the adversary that has the ability to access the GPRS device can easily control it by feeding the same IV into the device several times. Thus, the adversary can practically reset the GPRS device with the same key-IV pair as often as he needs.
- Injecting a fault that flips a state bit is also realistic in our attack model. Since the LFSRs of the GEA-like standard cipher are regularly clocked and one keystream bit is generated per clock cycle, thus the adversary can easily identify steps in the keystream generation and have a complete control over the exact timing of the fault injection. To obtain a bit fault injection, the adversary can execute the fault injection several times and check whether it is a bit fault injection. This has been successfully implemented for the the GEA-like standard ciphers on the platform ChipWhisperer Lite, as shown in Sect. IV and V. Thus, the adversary can practically inject a fault that flips a state bit of the GEA-like standard cipher.

As illustrated above, the practicality of our attack model is fully guaranteed. Moreover, the experimental verifications of our attacks implemented on the platform of ChipWhisperer Lite further confirm the practicality of our attack model.

The idea of our differential fault attack is generic and can be appliable to any GEA-like stream cipher. We first introduce the differential fault attack on the GEA-like stream cipher in this section, and then apply it to GEA-1 and GEA-2 in the following sections.

C. Identifying the exact fault location

Under the attack model above, the attacker can inject a fault at some random location r at any time t. By random location, we mean that the attacker cannot explicitly choose the location where the fault is to be injected. Since the GEA-like stream cipher consists of n LFSRs with lengths $l_1, \cdots, l_n$, respectively, which leads to an internal state size of $\mathcal{L}_n = l_1 + \cdots + l_n$ bits totally, the location r satisfies $0 \leq r \leq \mathcal{L}_n - 1$. More specifically, injecting a fault at the state bit $x_{1,r}^{(t)}$ of LFSR L_1 corresponds to the location $0 \leq r \leq l_1 - 1$. Similarly, injecting a fault at the state bit $x_{i,r-\mathcal{L}_i}^{(t)}$ of LFSR L_i corresponds to the location $\mathcal{L}_i \leq r < \mathcal{L}_{i+1}$, where $\mathcal{L}_i = l_1 + \cdots + l_{i-1}$. Without loss of generality, here we assume $t = 0$. However, it should be noted that the attack can work for any $t \geq 0$.

By injecting a fault at some random location r, certain changes in the keystream bits can be achieved. Let z_t and $z_t^{[r]}$ denote the fault-free keystream bit and faulty keystream bit for the random location r, respectively. Denote by $\Delta z_t^{[r]}$ the differential of z_t and $z_t^{[r]}$, i.e., $\Delta z_t^{[r]} = z_t \oplus z_t^{[r]}$. To make the attack successful, it is very important to identify the exact location where the fault has been induced. To achieve this goal, a signature vector $Sign_L^{[r]}$ is introduced and defined as follows.

$$Sign_L^{[r]} = \left(s_0^{[r]}, s_1^{[r]}, \cdots, s_{L-1}^{[r]}\right)$$

where

$$s_t^{[r]} = \begin{cases} 0\,, & if\ \Pr\left(\Delta z_t^{[r]} = 0\right) = 1 \\ *\,, & if\ 0 < \Pr\left(\Delta z_t^{[r]} = 0\right) < 1 \\ 1\,, & if\ \Pr\left(\Delta z_t^{[r]} = 0\right) = 0 \end{cases}$$

In the signature vector $Sign_L^{[r]}$, the parameters r and L denote the random location where the fault is injected and the length of required keystream, respectively. Before identifying the fault location, the attacker has to pre-compute all $\mathcal{L}_n$ signature vectors $Sign_L^{[0]}, Sign_L^{[1]}, \cdots, Sign_L^{[\mathcal{L}_n-1]}$ for GEA-1, and store them for comparisons. In the following, an algorithm called **Algorithm 1** is proposed to pre-compute all these $\mathcal{L}_n$ signature vectors for the GEA-like stream cipher.

At the beginning of **Algorithm 1**, the initial differentials of all $\mathcal{L}_n - 1$ internal state bits are set to be zeros, and then one fault is injected into one of these $\mathcal{L}_n - 1$ state bits, as in the steps 2-8 of **Algorithm 1**. To obtain the signature vector $Sign_L^{[r]}$ of length L, the steps 10-17 are executed L times. In the steps 10-11, the probability $\Pr(\Delta z_{i,t} = 0)$ is obtained by looking up the table D_F. In this table, the differential of the input bits of F are inputted, and the probability $\Pr(\Delta F = 0)$ is taken as output. Supposing there are u input bits ($u = 7$ for both GEA-1 and GEA-2), then there are 2^u possible values for the u differential bits and thus the table D_F can be easily constructed with complexity 2^{2u}. In the steps 12-19, **Algorithm 1** calculates the value of the signature bit $s_t^{[r]}$ according to the obtained values of $\Pr(\Delta z_{i,t} = 0)$ ($1 \leq i \leq n$). In the steps 20-21 of **Algorithm 1**, the differentials of n LFSRs are updated using $L_{i,update}$ in a linear manner, where $L_{i,update}$ denotes the linear update function of the LFSR L_i. To obtain all $\mathcal{L}_n$ signature vectors, the Steps 2-22 should be executed $\mathcal{L}_n$ times, in each of which one signature vector $Sign_L^{[r]}$ is obtained. Finally, all $\mathcal{L}_n$ signature vectors $Sign_L^{[0]}, Sign_L^{[1]}, \cdots, Sign_L^{[\mathcal{L}_n-1]}$ are stored by **Algorithm 1** as output. The time cost of this algorithm mainly depends on the possible values of r and t, which leads to a time complexity of $\mathcal{L}_n \cdot L$.

Algorithm 1: Pre-computing all $\mathcal{L}_n$ signature vectors for the GEA-like stream cipher

Output: All $\mathcal{L}_n$ signature vectors for the GEA-like stream cipher.

1 **for** $r = 0$ **to** $\mathcal{L}_n - 1$ **do**
2 **for** $i = 1$ **to** n **do**
3 **for** $j = 0$ **to** $l_i - 1$ **do**
4 $\Delta x_{i,j}^{(0)} \leftarrow 0$;
5 $\mathcal{L}_0 \leftarrow 0$;
6 **for** $i = 1$ **to** n **do**
7 **if** $r \in [\mathcal{L}_{i-1}, \mathcal{L}_i - 1]$ **then**
8 $\Delta x_{i,r-\mathcal{L}_{i-1}}^{(0)} \leftarrow 1$;
9 **for** $t = 0$ **to** $L - 1$ **do**
10 **for** $i = 1$ **to** n **do**
11 $\Pr(\Delta z_{i,t} = 0) \leftarrow D_F\left(\Delta X_i^{(t)}\right)$;
12 **if** *there exists* i *such that* $0 < \Pr(\Delta z_{i,t} = 0) < 1$ **then**
13 $s_i^{[r]} \leftarrow *$;
14 **else**
15 **if** $\Pr(\Delta z_{i,t} = 0) = 0$ *holds for all* $1 \leq i \leq n$ **then**
16 $s_i^{[r]} \leftarrow 1$;
17 **else**
18 **if** $\Pr(\Delta z_{i,t} = 0) = 1$ *holds for all* $1 \leq i \leq n$ **then**
19 $s_i^{[r]} \leftarrow 0$;
20 **for** $i = 1$ **to** n **do**
21 $\Delta X_i^{(t+1)} \leftarrow L_{i,update}(\Delta X_i^{(t)})$;
22 Output and store the obtained signature vector $Sign_L^{[r]}$.

After obtaining $\mathcal{L}_n$ signature vectors by **Algorithm 1**, the attacker can identify the location where the fault has been induced. To achieve this goal, the attacker can collect the fault-free keystream of length L (i.e., $z_0, \cdots, z_{L-1}$) and the L-bit faulty keystream (i.e., $z_0^{[r]}, \cdots, z_{L-1}^{[r]}$), and then calculate the differential vector of them as $\Delta Z_L^{[r]} = \left(\Delta z_0^{[r]}, \cdots, \Delta z_{L-1}^{[r]}\right)$, where $\Delta z_t^{[r]} = z_t \oplus z_t^{[r]}$ for $t = 0, \cdots, L-1$. After then, the attacker can make comparisons of the differential vector $\Delta Z_L^{[r]}$ with each of $\mathcal{L}_n$ signature vectors $Sign_L^{[0]}, Sign_L^{[1]}, \cdots, Sign_L^{[\mathcal{L}_n-1]}$ to identify the exact fault location. For any given signature vector, say $Sign_L^{[h]}$ ($0 \leq h \leq \mathcal{L}_n - 1$), if the two conditions $\Delta z_t^{[r]} = 1$ and $s_t^{[h]} = 0$ hold simultaneously for some t ($0 \leq t \leq L-1$), a contradiction certainly exists between the given signature vector $Sign_L^{[h]}$ and the differential vector $\Delta Z_L^{[r]}$, and then $r \neq h$ automatically holds. For the convenience of description, a new definition is given as follows.

Definition 1. We call the relation between the signature vector $Sign_L^{[h]}$ and the differential vector $\Delta Z_L^{[r]}$ a *mismatch*, if there

exists at least one t ($0 \leq t \leq L-1$) such that $\Delta z_t^{[r]} = 1$ and $s_t^{[h]} = 0$ hold simultaneously.

By making comparisons using the obtained $\mathcal{L}_n$ signature vectors and the differential vector $\Delta Z_L^{[r]}$, the attacker can filter the signature vectors that achieve mismatches and obtain the remaining signature vectors which indicate the possible fault locations. The detailed process of identifying the exact fault location is described as an algorithm called **Algorithm 2**, as follows.

Algorithm 2: Identifying the exact fault location for the GEA-like stream cipher

Input: the fault-free keystream of length L (i.e., $z_0, \cdots, z_{L-1}$), the L-bit faulty keystream (i.e., $z_0^{[r]}, \cdots, z_{L-1}^{[r]}$), and the $\mathcal{L}_n$ signature vectors $Sign_L^{[0]}, Sign_L^{[1]}, \cdots, Sign_L^{[\mathcal{L}_n-1]}$

Output: All possible fault locations.

1 Calculate the differential vector $\Delta Z_L^{[r]} = \left(\Delta z_0^{[r]}, \cdots, \Delta z_{L-1}^{[r]}\right)$, where $\Delta z_t^{[r]} = z_t \oplus z_t^{[r]}$ for $t = 0, \cdots, L-1$;
2 **for** $h = 0$ **to** $\mathcal{L}_n - 1$ **do**
3 **for** $t = 0$ **to** $L-1$ **do**
4 Check whether $\Delta z_t^{[r]} = 1$ and $s_t^{[h]} = 0$ hold simultaneously:
5 **if** *Yes* **then**
6 return to step 2 and try the next value of h;
7 **else**
8 return to step 3 and try the next value of t.
9 If the check in step 4 always fails, go to step 10.
10 Output the value of h such that the check in step 4 always fails for $t = 0, \cdots, L-1$.

It is easy to see that the time cost of **Algorithm 2** is mainly dominated by the possible values of h and t, which leads to a time complexity of $\mathcal{L}_n \cdot L$. The probability that **Algorithm 2** succeeds in determining the exact fault location is obviously determined by the parameter L. More specifically, when L is chosen to be small, there may be several values of h outputted as possible fault locations, and thus the attacker cannot identify the exact fault location. In contrast, when L is chosen to be large enough, **Algorithm 2** can succeed in filtering all signature vectors that achieve mismatches and output the exact fault location with a high probability. Thus, the attacker can achieve a high success probability by choosing a reasonable value of L.

D. Recovering the Internal State

After determining the exact fault location, the attacker can start recovering the $\mathcal{L}_n$-bit internal state of the GEA-like stream cipher. Since our attack is under the random fault model, it means that the attacker cannot explicitly choose the location where the fault is injected. In other words, any value of r ($0 \leq r \leq \mathcal{L}_n - 1$) may be the exact fault location. Here, there are mainly two approaches that can be used to recover the internal state, i.e., algebraic attack and Guess and Determine attack. Algebraic attack aims at constructing the differential equations involving the internal state bits and solving them by the algebraic techniques, e.g., linearization or Gröbner Bases. Guess and Determine attack aims at recovering the remaining internal state bits after guessing some internal state bits. To achieve an efficient attack on the GEA-like stream cipher, the attacker should try both of these two approaches to see which is better.

IV. A Differential Fault Attack on GEA-1

In this section, we will apply the differential fault attack described above to the GEA-1 stream cipher. The attack follows the generally acceptable random fault model, where the attacker can inject a fault at some random location but does not know the exact location during the fault injection.

A. Identifying the exact fault location for GEA-1

For GEA-1, $n = 3$, $l_1 = 31, l_2 = 32, l_3 = 33$ and $\mathcal{L}_3 = 96$. By applying **Algorithm 1** to GEA-1, the attacker can obtain all 96 signature vectors for GEA-1. The time cost of pre-computing these signature vectors is about $96 \cdot L$. In step 11 of **Algorithm 1**, the probability $\Pr(\Delta z_{i,t} = 0)$ is obtained by looking up the table D_F. For GEA-1, $F = f$ and f is the nonlinear Boolean function from $\mathbb{F}_{2^7}$ to $\mathbb{F}_2$ with an algebraic degree of 4. In the table D_f, seven differential bits $\Delta x_0, \cdots, \Delta x_6$ are inputted, and the probability $\Pr(\Delta f = 0)$ is taken as output. Since there are only $2^u = 2^7 = 128$ possible values for the seven differential bits, the table D_f can be easily constructed with a practical time complexity of 2^{14}. See Appendix for full details of the table D_f.

After obtaining 96 signature vectors by **Algorithm 1**, the attacker can identify the location where the fault has been induced by **Algorithm 2**. Since the time cost of **Algorithm 2** is mainly dominated by the possible values of h and i, which leads to a time complexity of $96 \cdot L$ for GEA-1. The probability that **Algorithm 2** succeeds in determining the exact fault location is obviously determined by the parameter L. To obtain a reasonable value of L, we make a series of experiments. In each experiment, we set L to be a fixed value, and execute **Algorithm 2** 2^{16} times for GEA-1. Before executing **Algorithm 2**, the fault location is randomly selected for each time. The results obtained in these experiments are summarized in Table I. As shown in Table I, when the length of keystream used to identify the exact fault location is no less than 2048, the fault location can be correctly identified with a probability of about 0.995 which is quite near 1. To further confirm the success probabilities obtained, we make another series of experiments and enlarge the number of samples from 2^{16} to 2^{20} in each experiment. The results show that the success probabilities obtained in Table I are accurate. Thus, we choose $L = 2048$ to achieve a success probability of 0.995 for GEA-1. Since the time cost of applying **Algorithm 2** (or **Algorithm 1**) to GEA-1 is $96 \cdot L$, thus **Algorithm 2** (or **Algorithm 1**) can be easily done on a common laptop due to its low time cost.

TABLE I
THE PROBABILITY THAT **ALGORITHM 2** SUCCEEDS IN IDENTIFYING THE EXACT FAULT LOCATION FOR GEA-1

The number of executions of **Algorithm 2**	L	The number of times identifying the exact fault location correctly	Success probability
2^{16}	512	8189	0.125
	1024	34835	0.532
	1536	63916	0.975
	2048	65186	**0.995**
2^{20}	512	130909	0.125
	1024	560099	0.534
	1536	1023388	0.976
	2048	1043047	**0.995**

B. Recovering the Internal State of GEA-1

After determining the exact fault location for GEA-1, the attacker can start recovering the 96-bit internal state of GEA-1. Since our attack is under the random fault model, it means that the attacker cannot explicitly choose the location where the fault is injected. In other words, any value of $r\,(0 \le r \le 95)$ may be the exact fault location. In the following, we will show how to recover the internal state of the LFSR where the fault is identified to be injected.

Recovering One of Three LFSRs Using One Random Fault. Now, we first take the LFSR A as an example, and then tackle the issue similarly in the cases of LFSRs B and C, respectively. When the fault is identified to be injected into LFSR A, by comparing the fault-free keystream and faulty keystream, a series of differential equations can be obtained as follows.

$$\begin{aligned}\Delta z_t^{[r]} &= z_t \oplus z_t^{[r]} \\ &= z_{1,t} \oplus z_{1,t}^{[r]} \oplus z_{2,t} \oplus z_{2,t}^{[r]} \oplus z_{3,t} \oplus z_{3,t}^{[r]},\ t = 0, \cdots, N-1\end{aligned}$$

where $\{z_t\}$ and $\left\{z_t^{[r]}\right\}$ denote the N-bit fault-free keystream and N-bit faulty keystream, respectively. In the case that the exact fault location r has been identified to satisfy $0 \le r \le 30$, the internal states of LFSRs B and C do not contain any differentials, since they are updated independently of the LFSR A. Thus, $z_{2,t} = z_{2,t}^{[r]}$and $z_{3,t} = z_{3,t}^{[r]}$ always hold simultaneously. Now, the differential equations above can be simplified as follows.

$$\Delta z_t^{[r]} = z_{1,t} \oplus z_{1,t}^{[r]},\ t = 0, \cdots, N-1$$

Considering the update function $L_{1,update}$ of LFSR A and the nonlinear function f, the differential equations above can be rewritten as follows.

$$\begin{aligned}\Delta z_t^{[r]} = &f\left(L_{1,update}^t\left(a_0^{(0)}, \cdots, a_{30}^{(0)}\right)\right) \oplus \\ &f\left(L_{1,update}^t\left(a_0^{(0)}, \cdots, a_{r-1}^{(0)}, a_r^{(0)} \oplus 1, a_{r+1}^{(0)}, \cdots, a_{30}^{(0)}\right)\right)\end{aligned}$$

where $t = 0, \cdots, N-1$, and $a_0^{(0)}, \cdots, a_{30}^{(0)}$ denote the 31-bit internal state of LFSR A. After obtaining the differential equations above, a straight idea is to solve these equations to recover the 31-bit internal state $a_0^{(0)}, \cdots, a_{30}^{(0)}$. Now, we try to achieve this goal using a combination of algebraic techniques and a guessing strategy. More specifically, we can guess some internal state bits of LFSR A and recover the remaining bits by solving a system of nonlinear equations. Without loss of generality, we assume that the attacker guesses $m\,(0 \le m \le 30)$ internal state bits of LFSR A. Since the nonlinear function f has an algebraic degree of 4 and an algebraic immunity of also 4, the number of monomials present in the system of nonlinear equations is upper bounded by $\mathcal{N}_m = 1 + \sum_{j=1}^{4} \binom{31-m}{j}$, where $31-m$ denotes the number of remaining internal state bits except the guessed m bits. By using a linearization technique, a linear system that consists of $\mathcal{N}_m$ equations at most can be obtained. Thus, to recover the 31-bit internal state of LFSR A, the attacker needs to solve a linear system for each possible value of the guessed m internal state bits. This leads to a time complexity of 2^m times the cost of solving a linear system of size $\mathcal{N}_m$. The fastest practical algorithm we are aware of to solve a linear system in M unknowns is Strassen's algorithm [51], which has a time complexity of $7 \cdot M^{\log_2 7}$. Thus, the time complexity of recovering the 31-bit internal state of LFSR A can be calculated as

$$\begin{aligned}T_{[A,m]} &= 2^m \cdot 7 \cdot \mathcal{N}_m^{\log_2 7} \\ &= 2^m \cdot 7 \cdot \left(1 + \sum_{j=1}^{4} \binom{31-m}{j}\right)^{\log_2 7}\end{aligned}$$

We have calculated the time complexity $T_{[A,m]}$ for each possible value of m, the results show that the minimum value $\min\left\{T_{[A,m]}, 0 \le m \le 31\right\}$ is obtained when $m = 30$ satisfies. It is easy to calculate that $T_{[A,30]} \approx 2^{35.615}$, which is bigger than 2^{31} that indicates the time complexity of making an exhaustive search of the 31-bit internal state of LFSR A. In other words, to achieve the lowest time complexity, the attacker should guess all 31 internal state bits directly, and then filter out the wrong guesses and obtain the correct internal state using the differential equations above. Thus, the attacker can recover the 31-bit internal state of LFSR A with a time complexity of $T_A = 2^{31}$.

Similar to the case when the fault is identified to be injected into the LFSR A, when the fault is identified to be injected into the LFSR B (or C), the attacker can recover the 32-bit (or 33-bit) internal state of LFSR B (or C) with a time complexity of $T_B = 2^{32}$ (or $T_C = 2^{33}$).

Recovering All Three LFSRs Using More Random Faults. The subsection above shows that when the fault is identified to be injected into one of three LFSRs, the internal state of this LFSR can be recovered. However, the other two LFSRs have to be exhausted, which will lead to a quite high time complexity. If more random faults are injected, more internal state bits of GEA-1 will be recovered, and then the time complexity can be significantly reduced.

Suppose that there are $w\,(\ge 2)$ random faults used in the differential fault attack on GEA-1. To achieve a low time complexity, we expect that these w random faults are injected into at least two of three LFSRs. This consists of the following four cases.

- **Case One:** The w random faults are injected into LFSRs

A and B. In this case, the attacker can recover the internal states of the LFSRs A and B independently, as the subsection above analyzes. This leads to a time complexity of $T_A + T_B = 2^{31} + 2^{32}$. After then, the attacker has to make an exhaustive search of the 33-bit internal state of LFSR C to recover all 96-bit internal state of GEA-1, which has a time complexity of 2^{33}. Thus, the total time complexity of this case can be calculated as $2^{31} + 2^{32} + 2^{33} = 2^{33.807}$.

- **Case Two:** The w random faults are injected into LFSRs A and C. Similar to **Case One**, the total time complexity of this case can be calculated as $2^{31}+2^{33}+2^{32} = 2^{33.807}$.
- **Case Three:** The w random faults are injected into LFSRs B and C. Similar to **Case One**, the total time complexity of this case can be calculated as $2^{32} + 2^{33} + 2^{31} = 2^{33.807}$.
- **Case Four:** The w random faults are injected into all three LFSRs. In this case, the attacker can recover the internal states of all three LFSRs independently, as the subsection above analyzes. This directly leads to a total time complexity of $T_A + T_B + T_C = 2^{31} + 2^{32} + 2^{33} = 2^{33.807}$.

Taking into account the four cases above, once w random faults are injected into at least two of three LFSRs, the attacker can recover all 96-bit internal state of GEA-1 with a total time complexity of $2^{33.807}$. Since the fault-free keystream and faulty keystream are only used to filter out the wrong guesses and obtain the correct internal state, 96 bits are enough theoretically to recover all 96-bit internal state of GEA-1. Thus, when $L = 2048$ satisfies, the available fault-free and faulty keystream bits are more than enough to make the attack on GEA-1 succeed. Let p_w denote the probability that w random faults are injected into at least two of three LFSRs, and then it can be calculated as follows.

$$p_w = 1 - \left(\tfrac{31}{96}\right)^w - \left(\tfrac{32}{96}\right)^w - \left(\tfrac{33}{96}\right)^w$$

To achieve a sufficiently high success probability of the attack on GEA-1, it is easy to obtain that $w = 4$ is a reasonable choice, which implies $p_4 = 0.963$.

C. Complexity Analysis of the Attack on GEA-1

The differential fault attack on GEA-1 consists of two processes, i.e., identifying the exact fault location and recovering the internal state. As shown in **Algorithm 2**, the process of determining the exact fault location has a time complexity of $96 \cdot L$. When $L = 2048$ is chosen, the differential fault attack on GEA-1 has a total time complexity of $T_{GEA-1} = 96 \cdot 2048 + 2^{33.807} \approx 2^{33.807}$. It shows that the time cost of determining the exact fault location is so small that can be ignored, compared with the time cost of recovering the internal state. Futhermore, the number of keystream bits required by the attack is dominated by the process of determining the exact fault location. As analyzed above, when $L = 2048$ satisfies, the available fault-free and faulty keystream bits are more than enough to make the attack on GEA-1 succeed. Since each keystream in GEA-1 is limited to 1600 bytes (i.e., 12800 bit), a keystream of length 2048 is undoubtedly valid.

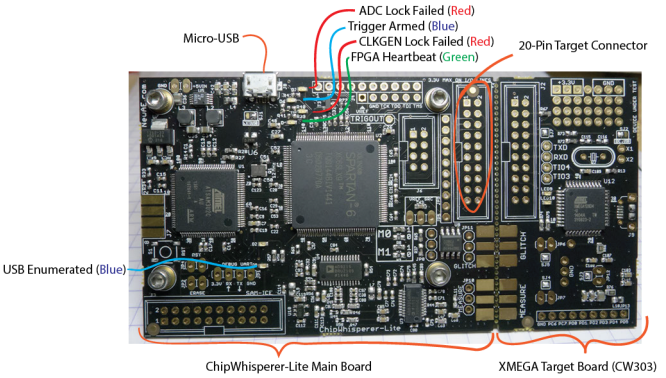

Fig. 3. The platform ChipWhisperer Lite used in the experimental verification of the attack on GEA-1 (and also GEA-2).

Now, we analyze the success probability of the attack on GEA-1 above. To make the attack succeed, there are two conditions which have to be satisfied. The one is that w random faults are injected into at least two of three LFSRs. The other is that the fault locations are correctly identified. Thus, the success probability of the attack on GEA-1, denoted as p_{GEA-1}, can be calculated as follows.

$$p_{GEA-1} = p_w \cdot (0.995)^2 = p_4 \cdot (0.995)^2 = 0.953$$

Thus, the attack on GEA-1 above can succeed in recovering all 96-bit internal state of GEA-1 with a probability of 0.953, which is sufficiently high to make the attack practical.

D. Experimental Verification of the Attack on GEA-1

To verify the attack on GEA-1 above, we have validated the cryptanalytic result by implementing the whole attack process. Firstly, we implement the GEA-1 stream cipher on the platform ChipWhisperer Lite, as shown in Fig. 3. Secondly, we inject random faults by voltage glitches, and use **Algorithm 2** to check whether it is a random bit fault. If it is a random bit fault, the exact fault location is identified. When no less than two of three LFSRs are injected random bit faults, it is enough to guarantee the success of the attack on GEA-1. We implement this process 64 times, the experimental result shows that the random bit faults can be successfully injected into no less than two of three LFSRs within about 343.330 seconds on average. Finally, after determining the exact fault location, we recover the internal state bits of all three LFSRs on a common laptop with 1.19 GHz Intel(R) Core(TM) i5-1035G1 processor. The experimental result shows that the processes of recovering three LFSRs A, B, C can be completed with about 46.414, 82.323 and 180.131 seconds, respectively. Thus, all 96-bit internal state of GEA-1 can be recovered within about 308.868 seconds.

Considering the time costs of determining the exact fault location and recovering all internal state bits, the GEA-1 stream cipher can be broken on a common laptop within $343.330 + 308.868 = 652.198$ seconds (approximately 10.870 minutes) by our differential fault attack.

TABLE II
THE PROBABILITY THAT **ALGORITHM 2** SUCCEEDS IN IDENTIFYING THE EXACT FAULT LOCATION FOR GEA-2

The number of executions of **Algorithm 4**	L	The number of times identifying the exact fault location correctly	Success probability
2^{16}	512	5514	0.084
	1024	35607	0.543
	1536	63609	0.971
	2048	65241	**0.995**
2^{20}	512	86868	0.083
	1024	570797	0.544
	1536	1017292	0.970
	2048	1043411	**0.995**

V. A DIFFERENTIAL FAULT ATTACK ON GEA-2

In this section, we will apply the differential fault attack on the GEA-2 stream cipher. The attack is similar to the attack on GEA-1 and also follows the random fault model.

A. Identifying the exact fault location for GEA-2

Since GEA-2 utilizes a fourth LFSR D of length 29, it has $n = 4$, $l_1 = 31, l_2 = 32, l_3 = 33, l_4 = 29$ and $\mathcal{L}_4 = 125$ for GEA-2. the location r satisfies $0 \leq r \leq 124$ for GEA-2. By implying **Algorithm 1** to GEA-2, the attacker can obtain all $\mathcal{L}_4 = 125$ signature vectors $Sign_L^{[0]}, Sign_L^{[1]}, \cdots, Sign_L^{[124]}$ for GEA-2. After then, the attacker can identify the exact fault location by applying **Algorithm 2** to GEA-2, using the obtained 125 signature vectors.

To evaluate the probability that **Algorithm 2** succeeds in determining the exact fault location for GEA-2, a series of experiments have been made. In each experiment, the length of available keystream L is chosen to be a fixed value (such as $L = 512, 1024, 1536, 2048$), and **Algorithm 2** is executed 2^{16} or 2^{20} times. Before executing **Algorithm 2**, the fault location is randomly selected for each time. The experimental results obtained are summarized in Table II. As shown in Table II, when the length of available keystream used to identify the exact fault location is no less than 2048, the fault location can be correctly identified with a probability of about 0.995.

B. Recovering the Internal State of GEA-2

After determining the exact fault location, the attacker can start recovering the 125-bit internal state of GEA-2. The attacker can apply the strategy of recovering the 96-bit internal state of GEA-1 to GEA-2. More specifically, the attacker can recover three of four LFSRs by using a certain amount of random faults, and then recover the remaining LFSR by making an exhaustive search. Suppose that there are $w\,(\geq 2)$ random faults used in the differential fault attack on GEA-2. To propose a practical differential fault attack on GEA-2, we expect that these w random faults are injected into at least three of four LFSRs. Let q_w denote the probability that n random faults are injected into at least three of four LFSRs, and then it can be calculated as follows.

$$q_w = 1 - \left(\tfrac{31+32}{125}\right)^w - \left(\tfrac{31+33}{125}\right)^w - \left(\tfrac{31+29}{125}\right)^w - \left(\tfrac{32+33}{125}\right)^w - \left(\tfrac{32+29}{125}\right)^w - \left(\tfrac{33+29}{125}\right)^w + 2\left(\tfrac{31}{125}\right)^w + 2\left(\tfrac{32}{125}\right)^w + 2\left(\tfrac{33}{125}\right)^w + 2\left(\tfrac{29}{125}\right)^w$$

To achieve a sufficiently high success probability of the attack on GEA-2, it is easy to obtain that $n = 6$ is a reasonable choice, which implies $q_6 = 0.907$. Thus, by using 6 random faults, the random faults can be injected into at least three of four LFSRs with a high probability of about 0.907.

C. Complexity Analysis of the Attack on GEA-2

In the internal state process, the attacker can recover these three LFSRs with a time complexity of $T_\alpha + T_\beta + T_\delta$, where $\{\alpha, \beta, \delta\} \subset \{A, B, C, D\}$. Considering the time cost of making an exhaustive search of the remaining LFSR, the time complexity of recovering the 125-bit internal state of GEA-2 can be calculated as $T_A + T_B + T_C + T_D = 2^{31} + 2^{32} + 2^{33} + 2^{29} \approx 2^{33.858}$. Taking into account the time cost of determining the exact fault location, when $L = 2048$ is chosen for GEA-2, the differential fault attack on GEA-2 has a total time complexity of $T_{GEA-2} = 125 \cdot 2048 + 2^{33.858} \approx 2^{33.858}$. Similar to the attack on GEA-1 above, the success probability of the attack on GEA-2, denoted as p_{GEA-2}, can be calculated as follows.

$$p_{GEA-2} = q_6 \cdot (0.995)^3 = 0.893$$

Up to now, we have proposed a practical differential fault attack on GEA-2. The attack can succeed in recovering all 125-bit internal state of GEA-2 with a probability of 0.893, which is sufficiently high to make the attack practical.

D. Experimental Verification of the Attack on GEA-2

Similar to the attack on GEA-1, we have validated the cryptanalytic result by implementing the whole attack process, to confirm the theoretical cryptanalytic result of GEA-2 above. Firstly, we implement the GEA-2 stream cipher on the platform ChipWhisperer Lite, as shown in Fig. 3. Secondly, we inject random faults by voltage glitches, and use **Algorithm 2** to check whether it is a random bit fault. If it is a random bit fault, the exact fault location is identified. When no less than three of four LFSRs are injected random bit faults, it is enough to guarantee the success of the attack on GEA-2. We implement this process 32 times, the experimental result shows that the random bit faults can be successfully injected into no less than three of four LFSRs within about 587.625 seconds on average. Finally, after determining the exact fault location, we recover the internal state bits of all four LFSRs on a common laptop with 1.19 GHz Intel(R) Core(TM) i5-1035G1 processor. The experimental result shows that the process of recovering all 125-bit internal state of GEA-2 can be completed within about 320.473 seconds.

Considering the time costs of determining the exact fault location and recovering all internal state bits, the GEA-2 stream cipher can be broken on a common laptop within $587.625 + 320.473 = 908.098$ seconds (approximately 15.135 minutes) by our differential fault attack.

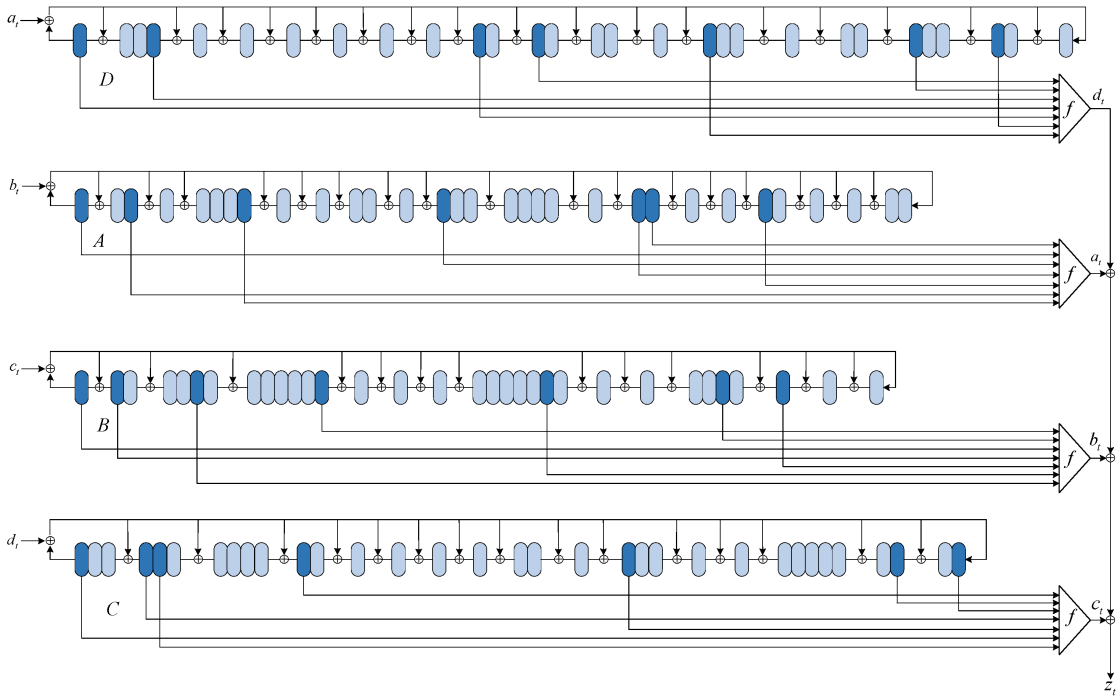

Fig. 4. The structure of the new variant GEA-2M.

VI. The Countermeasures to Mitigate Our Attacks on the GPRS Standard Ciphers

Since the GPRS standard ciphers are still widely used in many applications, it is crucial to discuss possible countermeasures that could neutralize our attacks, ideally with minimal impact on the deployed services to maintain availability. Generally, the fault attacks may be prevented at the cipher design level or by the countermeasures separately from the ciphers. For the GPRS standard ciphers, the possible countermeasures are presented as follows.

The Countermeasure at the Cipher Design Level. The key reason that our attacks on the GPRS standard ciphers are successful is the linear updation mechanism and mutual independence of the feedback shift registers. To address this problem, a feasible way is to bring in nonlinearity and remove mutual independence of the feedback shift registers in the GPRS standard ciphers. Since GEA-2 is an alternative version of GEA-1 and provides a better security margin than GEA-1, we will take GEA-2 as a basis and propose a new variant of GEA-2 named GEA-2M to enhance the resistance against the differential fault attacks.

The new variant GEA-2M has a similar structure to GEA-2, as depicted in Fig. 4. The only difference is that the four output bits of the nonlinear filter function f in GEA-2M is XORed onto the feedback bits of four LFSRs. It should be noted that the four output bits are used in the interlaced form, i.e., the output bits a_t, b_t, c_t, d_t corresponding to four LFSRs A, B, C, D are XORed onto the feedback bits of the LFSRs D, A, B, C, respectively. By the slight modification, the four LFSRs in GEA-2 become nonlinear in GEA-2M. The recurrence relations of four NFSRs in GEA-2M are given as follows.

NFSR A:

$$a_i^{(t+1)} = \begin{cases} a_{i+1}^{(t)} \oplus a_0^{(t)} \oplus b_t, & if\ i \in T_A \\ a_{i+1}^{(t)}, & if\ i \in \{0, \cdots, 29\} - T_A \\ a_0^{(t)} \oplus b_t, & if\ i = 30 \end{cases}$$

NFSR B:

$$b_i^{(t+1)} = \begin{cases} b_{i+1}^{(t)} \oplus b_0^{(t)} \oplus c_t, & if\ i \in T_B \\ b_{i+1}^{(t)}, & if\ i \in \{0, \cdots, 30\} - T_B \\ b_0^{(t)} \oplus c_t, & if\ i = 31 \end{cases}$$

NFSR C:

$$c_i^{(t+1)} = \begin{cases} c_{i+1}^{(t)} \oplus c_0^{(t)} \oplus d_t, & if\ i \in T_C \\ c_{i+1}^{(t)}, & if\ i \in \{0, \cdots, 31\} - T_C \\ c_0^{(t)} \oplus d_t, & if\ i = 32 \end{cases}$$

NFSR D:

$$d_i^{(t+1)} = \begin{cases} d_{i+1}^{(t)} \oplus d_0^{(t)} \oplus a_t, & if\ i \in T_D \\ d_{i+1}^{(t)}, & if\ i \in \{0, \cdots, 27\} - T_D \\ d_0^{(t)} \oplus a_t, & if\ i = 28 \end{cases}$$

where the four tap sets T_A, T_B, T_C, T_D are the same with that in GEA-2.

Unlike GEA-2, the four feedback shift registers in GEA-2M are nonlinear, and any one of them is influenced by another one. Overall, the four NFSRs in GEA-2M becomes a mutually influenced whole, and thus we name the new variant GEA-2M. The security margin of GEA-2M is analyzed as follows.

- The slight modification can significantly enhance the resistance of GEA-2M against the differential fault attacks. Since four NFSRs in GEA-2M becomes a mutually influenced whole, any fault injection into the internal state of GEA-2M will propagate to all internal state bits, and thus the signature vectors can not be constructed for GEA-2M. Without the signature vectors, it becomes quite difficult or even impossible to identify the exact fault location. Even if the signature vectors are constructed for GEA-2M and the exact fault location is determined, recovering the four NFSRs that consist of 125 bits in total will certainly be extremely time-consuming, which would make the differential fault attack worse than the exhaustive key search. Therefore, the new variant GEA-2M is immune to our differential fault attack.
- All existing attacks on GEA-2 utilize a weakness of GEA-2, i.e., the four LFSRs are mutually independent. By the slight modification, the independence is removed in GEA-2M. The cryptanalyst has to consider the four NFSRs together when evaluating the security margin of GEA-2M, which will greatly increase the difficulty of proposing a valid attack on GEA-2M. Thus, the security margin of GEA-2M against the existing attacks is also enhanced.

As for the implementation cost, only four XOR operations are added in GEA-2M, compared with the original GEA-2. The incremental implementation cost is so small that can be ignored, compared with the implementation cost of four feedback shift registers and four filter functions. Thus, the implementation cost of GEA-2M is almost the same with that of GEA-2, but GEA-2M provides a better security margin than GEA-2, as analyzed above.

The Countermeasures Separately from the Cipher. The differential fault attack can also be prevented separately from the cipher. Up to now, a wide range of fault injection countermeasures has been developed and well studied to thwart fault attacks, e.g., error detection, correction techniques, repeated computations and a combination of error detecting codes and threshold implementations, as reviewed in [49]. Overall, the countermeasures can be divided into two main categories. In the first category, the countermeasures are introduced to prevent the fault happening through a specialized device. In the second category, the countermeasures aim at eliminating the effect of the fault by means of redundancy or protocol. To mitigate our attacks on the GPRS standard ciphers, the following two countermeasures that are reasonable for the various application scenarios of the GPRS standard ciphers can be deployed.

- **Deploying a separate and dedicated device that is not related to the GPRS standard cipher**. Generally, the deployed devices can be divided into two main categories. In the first category, a passive device where a shield is used to block external interference can be used. As analyzed in [52], it may be possible to peel off the passive layer using this kind of dedicated device like Ultra Tec ASAP-1. In the second category, an active device (such as a sensor that detects any potential stress) also works to mitigate differential fault attacks, as confirmed in [53], [54].
- **Using redundancy in computation**. Generally, all the countermeasures that mitigate fault effect utilize some form of redundancy to achieve protection against the fault attacks. The most common choice for this kind of countermeasures is duplicating the circuit fully or partially. The duplication can be implemented in the spatial or temporal domain. As for the GPRS standard ciphers, a technique called *double modular redundancy* [55] can be used. *Double modular redundancy* is a technique that is used to detect fault injection attacks, and is the most common form of modular redundancy for error detection. As depicted in Fig. 5, the basic idea of *double modular redundancy* is that by duplicating the module a fault injection attack can be detected by comparing the outputs of the modules. After then, some predefined procedure to handle the situation of a fault detection is deployed, such as not producing any output or producing a random output.

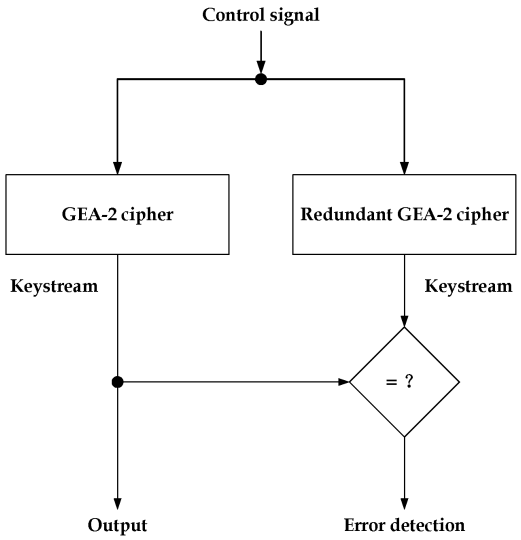

Fig. 5. The countermeasure for GEA-2 using the *double modular redundancy* technique.

Using the countermeasures above, our differential fault attacks can be successfully mitigated.

VII. Conclusions

GEA-1 and GEA-2 are two proprietary stream ciphers used in GPRS to protect against eavesdropping GPRS between the phone and the base station. Up to now, differential fault attacks on the GPRS standard ciphers have not yet been studied. In this paper, a differential fault attack on the GEA-like stream ciphers under the random fault model is proposed for the first time. As applications, practical differential fault attacks on

the GPRS standard ciphers are presented. The cryptanalytic results show that the 64-bit secret keys of GEA-1 and GEA-2 can be recovered with time complexities of $2^{33.807}$ and $2^{33.858}$, respectively. We validate the cryptanalytic results by simulating the whole attacks on the platform ChipWhisperer Lite. The experimental results show that both GEA-1 and GEA-2 can be broken within sixteen minutes on a common laptop. To mitigate our attacks on the GPRS standard ciphers, the possible countermeasures are presented. At the cipher design level, a new variant of GEA-2 named GEA-2M is proposed as a concrete countermeasure. The new variant GEA-2M has significantly enhanced the resistance against all existing attacks (including our differential fault attacks) and provides a better security margin than GEA-2. To protect the security of the massive GPRS devices all over the world, we recommend using GEA-2M to replace GEA-2.

ACKNOWLEDGMENTS

The authors would like to thank the associate editor and anonymous reviewers for their valuable comments and suggestions.

REFERENCES

[1] ETSI: Security algorithms group of experts (sage); report on the specification, evaluation and usage of the gsm gprs encryption algorithm (gea). Technical Report [Online]. Available: https://www.etsi.org/deliver/etsi_tr/101300_101399/101375/01.01.01_60/tr_101375v010101p.pdf

[2] C. Brookson. GPRS Security [Online]. Available: https://web.archive.org/web/20120914110208/

[3] ETSI: Security algorithms group of experts (sage); report on the specification, evaluation and usage of the gsm gprs encryption algorithm (gea). Technical Report [Online]. Available: https://www.etsi.org/deliver/etsi_tr/101300_101399/101375/01.01.01_60/tr_101375v010101p.pdf

[4] C. Beierle et al., "Cryptanalysis of the GPRS Encryption Algorithms GEA-1 and GEA-2," in *Proc. EUROCRYPT 2021*, Zagreb, Croatia, Oct. 2021, pp. 155-183.

[5] L. Ding, Z. Wu, X. Wang, Z. Guan, and M. Li, "New Attacks on the GPRS Encryption Algorithms GEA-1 and GEA-2," *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 2878-2889, Aug. 2022.

[6] C. Beierle, P. Felke, and G. Leander, "To Shift or Not to Shift: Understanding GEA-1," Cryptology ePrint Archive: Report 2021/829 [Online]. Available: https://eprint.iacr.org/2021/829

[7] C. Beierle, T. Beyne, P. Felke, G. Leander, "Constructing and Deconstructing Intentional Weaknesses in Symmetric Ciphers," in *Proc. CRYPTO 2022*, Santa Barbara, USA, August 13-18, 2022, pp. 748-778.

[8] D. Amzaleg, I. Dinur, "Refined Cryptanalysis of the GPRS Ciphers GEA-1 and GEA-2," in *Proc. EUROCRYPT 2022*, Trondheim, Norway, May 30-June 3, 2022, pp. 57-85.

[9] E. Pasalic, Y. Wei, "Generic related-key and induced chosen IV attacks using the method of key differentiation," Cryptology ePrint Archive: Report 2013/586 [Online]. Available: https://eprint.iacr.org/2013/586

[10] FX. Standaert, "Introduction to Side-Channel Attacks," in *Secure Integrated Circuits and Systems*, pp. 27-42, Jan. 2009, Springer, Boston, MA. https://doi.org/10.1007/978-0-387-71829-3_2

[11] P. Kocher, "Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS and Other Systems," in *Proc. CRYPTO 1996*, Santa Barbara, USA, August 18-22, 1996, pp. 104-113.

[12] P. Kocher, J. Jaffe, B. Jun, "Differential power analysis," in *Proc. CRYPTO 1999*, Santa Barbara, California, USA, August 15-19, 1999, pp. 388-397.

[13] E. Brier, C. Clavier, F. Olivier, "Correlation power analysis with a leakage model," in *Proc. CHES 2004*, Cambridge, MA, USA, August 11-13, 2004, pp. 16-29.

[14] S. Chari, J.R. Rao, P. Rohatgi, "Template attacks," in *Proc. CHES 2002*, Redwood Shores, CA, USA, August 13-15, 2002, pp. 13-28.

[15] D. Boneh, R. A. DeMillo and R. J. Lipton, "On the importance of checking cryptographic protocols for faults," in *Proc. EUROCRYPT 1997*, Konstanz, Germany, May 11-15, 1997, pp. 37-51.

[16] E. Biham, A. Shamir, "Differential Fault Analysis of Secret Key Cryptosystems," in *Proc. CRYPTO 1997*, Santa Barbara, California, USA, August 17-21, 1997, pp. 513-525.

[17] E. Biham, A. Shamir, "Differential cryptanalysis of DES-like cryptosystems," in *Proc. CRYPTO 1990*, Santa Barbara, CA, USA, August 11-15, 1990, pp. 2-21.

[18] M. Rivain, "Differential fault analysis on DES middle rounds," in *Proc. CHES 2009*, Lausanne, Switzerland, Sep. 6-9, 2009, pp. 457-469.

[19] F. Zhang, S. Guo, X. Zhao, T. Wang, J. Yang, F.-X. Standaert, D. Gu, "A Framework for the Analysis and Evaluation of Algebraic Fault Attacks on Lightweight Block Ciphers," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 5, pp. 1039-1054, May 2016.

[20] J. Blmer, J.P. Seifert, "Fault based cryptanalysis ofthe advanced encryption standard (AES)," in *Proc. FC 2003*, Guadeloupe, Guadeloupe, January 27-30, 2003, pp. 162-181.

[21] C.N. Chen, S.M. Yen, "Differential fault analysison AES key schedule and some countermeasures," in *Proc. ACISP 2003*, Wollongong, NSW, Australia, July 9-11, 2003, pp. 118-129.

[22] 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, Feb. 2012.

[23] S. Patranabis, A. Chakraborty, D. Mukhopadhyay, P.P. Chakrabarti, "Fault Space Transformation: A Generic Approach to Counter Differential Fault Analysis and Differential Fault Intensity Analysis on AES-Like Block Ciphers," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 5, pp. 1092-1102, May 2017.

[24] T. E. Pogue, N. Nicolici, "Incremental Fault Analysis: Relaxing the Fault Model of Differential Fault Attacks," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 28, no. 3, pp. 750-763, March 2020.

[25] N. Vafaei, S. Zarei, N. Bagheri, M. Eichlseder, R. Primas, H. Soleimany, "Statistical Effective Fault Attacks: The Other Side of the Coin," *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 1855-1867, May 2022.

[26] J.J. Hoch, A. Shamir, "Fault Analysis of Stream Ciphers," in *Proc. CHES 2004*, Cambridge, MA, USA, August 11-13, 2004, pp. 240-253.

[27] E. Biham, G. Louis, Q.N. Phong, "Impossible fault analysis of RC4 and differential fault analysis of RC4," in *Proc. FSE 2005*, Paris, France, February 21-23, 2005, pp. 359-367.

[28] S. Banik, M. Subhamoy, S. Santanu, "A differential fault attack on the grain family of stream ciphers," in *Proc. CHES 2012*, Leuven, Belgium, September 9-12, 2012, pp. 122-139.

[29] A. Berzati, C. Canovas, G. Castagnos, B. Debraize, L. Goubin, A. Gouget, P. Paillier, S. Salgado, "Fault Analysis of Grain-128," in *Proc. HOST 2009*, San Francisco, CA, USA, July 27-27, 2009, pp. 7C14.

[30] S. Karmakar, D. Roy Chowdhury, "Fault analysis of Grain-128 by targeting NFSR," in *Proc. AFRICACRYPT 2011*, Dakar, Senegal, July 5-7, 2011, pp. 298C315.

[31] S. Sarkar, S. Banik, S. Maitra, "Differential Fault Attack against Grain Family with Very Few Faults and Minimal Assumptions," *IEEE Transactions on Computers*, vol. 64, no. 6, pp. 1647-1657, 1 June 2015.

[32] M. Hojsłk, B. Rudolf, "Differential fault analysis of Trivium," in *Proc. FSE 2008*, Lausanne, Switzerland, Feb. 10-13, 2008, pp. 158-172.

[33] Y. Hu, J. Gao, Q. Liu, Y. Zhang, "Fault analysis of Trivium," *Designs, Codes and Cryptography*, vol. 62, pp. 289-311, May 2011.

[34] S. Banik, S. Maitra, "A differential fault attack on MICKEY 2.0" in *Proc. CHES 2013*, Santa Barbara, California, USA, August 20-23, 2013, pp. 215-232.

[35] Y.E. Salehani, A. Kircanski, A.M. Youssef, "Differential fault analysis of sosemanuk," in *Proc. AFRICACRYPT 2011*, Dakar, Senegal, July 5-7, 2011, pp. 316-331.

[36] A. Kircanski, A.M. Youssef, "Differential fault analysis of HC-128," in *Proc. AFRICACRYPT 2010*, Stellenbosch, South Africa, May 3-6, 2010, pp, 261-278.

[37] A. Beckers, B. Gierlichs, I. Verbauwhede, "Fault Analysis of the ChaCha and Salsa Families of Stream Ciphers," in *Proc. CARDIS 2017*, Lugano, Switzerland, November 13-15, 2017, pp. 196-212.

[38] M.A. Orumiehchiha, S. Rostami, E. Shakour, J. Pieprzyk, "A differential fault attack on the WG family of stream ciphers," *Journal of Cryptographic Engineering*, vol. 10, no. 2, pp. 189-195, March 2020.

[39] S. Maitra, A. Siddhanti, S. Sarkar, "A Differential Fault Attack on Plantlet," *IEEE Transactions on Computers*, vol. 66, no. 10, pp. 1804-1808, Oct. 2017.

[40] B. Debraize, I.M. Corbella, "Fault Analysis of the Stream Cipher Snow 3G," in *Proc. 2009 Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC)*, Lusanne, Switzerland, February 17, 2009, pp. 103-110.

[41] D. Roy, B. Bathe, S. Maitra, "Differential Fault Attack on Kreyvium & FLIP," *IEEE Transactions on Computers*, vol. 70, no. 12, pp. 2161-2167, Dec. 2021.

[42] R. Radheshwar, M. Kansal, P. Maux, D. Roy, "Differential Fault Attack on Rasta and FiLIP$_{DSM}$," *IEEE Transactions on Computers*, vol. 72, no. 8, pp. 2418-2425, Aug. 2023.

[43] A. Barenghi, L. Breveglieri, I. Koren, D. Naccache, "Fault Injection Attacks on Cryptographic Devices: Theory, Practice, and Countermeasures," *Proceedings of the IEEE*, vol. 100, no. 11, pp. 3056-3076, Nov. 2012.

[44] J. Breier, X. Hou, "How Practical Are Fault Injection Attacks, Really?," *IEEE Access*, vol. 10, pp. 113122-113130, Nov. 2022.

[45] Y. Kim, R. Daly, J. Kim, C. Fallin, J. Lee, D. Lee, et al., "Flipping bits in memory without accessing them: An experimental study of DRAM disturbance errors," in *Proc. 2014 ACM/IEEE 41st International Symposium on Computer Architecture (ISCA)*, Minneapolis, MN, USA, June 14-18, 2014, pp. 361-372.

[46] J. Liang, X. Zhang, D. Hu, Q. Shen, Y. Fang, Z. Wu, "ZK-Hammer: Leaking Secrets from Zero-Knowledge Proofs via Rowhammer," in *Proc. the 62nd Annual ACM/IEEE Design Automation Conference (DAC 2025)*, San Francisco, CA, USA, June 22-25, 2025, pp. 1-7.

[47] J. Liang, Z. Zhang, X. Zhang, Q. Shen, Y. Gao, X. Yuan, et al., "Achilles: a formal framework of leaking secrets from signature schemes via Rowhammer," in *Proc. the 34th USENIX Conference on Security Symposium*, Seattle, WA, USA, August 13-15, 2025, pp. 6757-6776.

[48] W. Chen, Z. Zhang, X. Zhang, Q. Shen, Y. Yarom, D. Genkin, et al., "HyperHammer: Breaking Free from KVM-Enforced Isolation," in *Proc. the 30th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS 2025)*, Rotterdam, Netherlands, March 30-April 3, 2025, pp. 545-559.

[49] A. Baksi, S. Bhasin, J. Breier, D. Jap, D. Saha, "A Survey on Fault Attacks on Symmetric Key Cryptosystems," *ACM Computing Surveys*, vol. 55, no. 4, pp. 1-34, Nov. 2022.

[50] F. Armknecht, W. Meier, "Fault Attacks on Combiners with Memory," in *Proc. SAC 2005*, Kingston, ON, Canada, August 11-12, 2025, pp. 36-50.

[51] D.H. Brailey, K. Lee, H.D. Simon, "Using Strassen's Algorithm to Accelerate the Solution of Linear Systems," *Journal of Softcomputing*, vol. 4, no. 35, pp. 357-371, Jan. 1990.

[52] H. Bar-El, H. Choukri, D. Naccache, M. Tunstall and C. Whelan, "The Sorcerer's Apprentice Guide to Fault Attacks," in *Proceedings of the IEEE*, vol. 94, no. 2, pp. 370-382, Feb. 2006.

[53] L. Zussa et al., "Efficiency of a glitch detector against electromagnetic fault injection," in *Proc. 2014 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, Dresden, Germany, 2014, pp. 1-6.

[54] H. Wei, B. Jakub, B. Shivam, "Cheap and cheerful: A low-cost digital sensor for detecting laser fault injection attacks," in *Proc. SPACE 2016*, Hyderabad, India, December 14-18, 2016, pp. 27-46.

[55] J.M. Schmidt, M. Medwed, "Countermeasures for Symmetric Key Ciphers," in M. Joye, M. Tunstall (Eds.), "Fault Analysis in Cryptography," Springer, Berlin, Heidelberg, 2012, pp. 73-87.