IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 14 September 2023, accepted 30 September 2023, date of publication 4 October 2023,
date of current version 18 October 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3321696

RESEARCH ARTICLE

Fault Probability Correlation Analysis Based on Secondary Filtering

TONG WU[1], DAWEI ZHOU[1], LEI DU[2], AND SHIWEI WANG[2]
[1]Department of Information Security, Naval University of Engineering, Wuhan, Hubei 430033, China
[2]DPLS Laboratory, Beijing 102300, China

Corresponding author: Dawei Zhou (zdw_xp@163.com)

This work was supported by the National Natural Science Foundation of China under Grant 11202239.

ABSTRACT Correlation power analysis (CPA) is a classical method in side-channel attacks. Based on the power consumption model, the correlation between the power consumption of cryptographic devices and the assumed intermediate value is analyzed to recover the key. Theoretically, only a few power traces are required to recover the key when the noise hypothesis is known. However, in the high-frequency and high-noise environment, the completion of CPA requires more power traces, and the computational complexity also increases. Therefore, this paper proposes a fault probability correlation analysis method based on secondary filtering (2F-FPCA), which selects the fault probability traces according to the Hamming Weight of the intermediate value and reduces the number of sampling points by selecting points of interest. This method does not need to access ciphertext and is little affected by noise. Moreover, it can recover the key with fewer fault probability traces and lower computational complexity, improving the attack efficiency of CPA. In this paper, 2F-FPCAs are carried out based on the AES-128 algorithm of the Micro Controller Unit (MCU). The key can be recovered successfully using 10 fault probability traces, and the computational complexity is reduced by 10^4 times.

INDEX TERMS Correlation power analysis, side-channel attacks, fault injection attacks, AES.

I. INTRODUCTION

Side-channel attacks play an essential role in evaluating the performance of cryptographic devices. There are many classic side-channel methods, such as attacking power consumption [1], running time [2], and electromagnetic leakage [3]. Power analysis is a representative type of side-channel attack, including Simple Power Analysis (SPA) [4], Correlation Power Analysis [5], and Differential Power Analysis (DPA) [6]. Brier et al. [7] first proposed CPA, which uses the power model to calculate the correlation coefficient between the power sample and the hypothetical intermediate value to identify the correct key. Heuser et al. [8] show that CPA is almost optimal when the leakage model is known under the assumption of affine transformation and Gaussian noise.

In practical applications, in high-frequency and high-noise environments, CPA often needs more power traces to recover the key, which means more computing time and computation. In addition, when the candidate key space is too large to traverse the search, CPA is often used to recover some bytes of the key. However, under the local leakage model, the correlation coefficient of CPA is significantly reduced, and the demand for the number of power traces is greater. In order to reduce the number of power traces required for attacks, key exhaustion [9], [10] and ranking evaluation [11] algorithms are proposed to estimate the ranking of correct keys. However, the problems of computational complexity and excessive memory requirements still need to be solved.

Recently, research on the computational efficiency of CPA has received widespread attention. An enhancement technique of CPA is proposed in [12], which classifies traces by Hamming distance and combines DPA, multi-bit DPA, and CPA. Kim et al. [13] proposed a preprocessing technique to select a subset with higher correlation factors from the power trace set and then conduct CPA. In [14], the plaintext is selected in both non-adaptive and adaptive ways, and the original optimization of the standard CPA is carried out, which reduces the number of power traces needed to recover the key.

The associate editor coordinating the review of this manuscript and approving it for publication was Pedro R. M. In'acio.

All the above methods improve the computational efficiency of CPA, but these papers do not analyze the computational complexity.

A method is proposed in [15] to index the vector template with plaintext values and then associate it with the power vector model. Compared with the original CPA calculation, the calculation speed of this method is 200 times faster, and it is especially effective when there are many traces. Reference [16] proposed to find points of interest (POI) to reduce the computing time of CPA. The above methods improve the computing speed of CPA and time but need many power traces as data support.

In order to solve the problems that CPA is greatly affected by noise, significant data demand, and high computational complexity, this paper applies CPA to fault attacks and proposes a fault probability correlation analysis attack method based on secondary filtering (2F-FPCA). This method uses the data dependence of the fault probability of cryptographic equipment under the fault injection attack and establishes the fault probability trace in the operation process. The traces are classified and filtered according to the Hamming Weight of the intermediate value, which reduces the number of fault probability traces needed to calculate the assumed intermediate value. At the same time, the number of samples and iterations are significantly reduced by finding POIs on the fault probability trace. 2F-FPCA does not need to access ciphertext and is little affected by noise. It can recover the key with less fault probability trace and lower computational complexity, improving the efficiency of CPA.

The paper is organized as follows. Section II starts with an overview of CPA and the experimental platform. In Section III, we propose a fault correlation analysis attack method based on fault probability (FPFCA), carry out experiments, and analyze the attack results. In Section IV, we improve the FPFCA to 2F-FPCA and evaluate the attack results. Finally, Section V concludes this paper.

II. PRELIMINARIES AND PRACTICAL IMPLEMENTATION

A. CORRELATION POWER ANALYSIS

CPA mainly uses the correlation between the actual power consumption and the power consumption model. CPA has a general attack strategy, which is divided into 5 steps [17]:

- **Step 1:** Select an intermediate value of the executed algorithm. First, select the power consumption model and establish the leakage function. The intermediate value must be a function that depends on the small part of the key and known non-constant data value, which is usually the plaintext or the ciphertext. The most widely used power consumption model is the Hamming Distance (HD) between two corresponding values in the same register or Hamming Weight (HW) of a specific value.
- **Step 2:** Measure the power consumption. Measure the power consumption of cryptographic devices when encrypting or decrypting different data blocks.
- **Step 3:** Calculate hypothetical intermediate values. For each key hypothetical, the corresponding hypothetical intermediate value is calculated.
- **Step 4:** Map the intermediate values to the power consumption value.
- **Step 5:** Compare the hypothetical power consumption values with the power traces. The index corresponding to the maximum correlation coefficient reveals the correct key index and time. The correlation is calculated as follows.

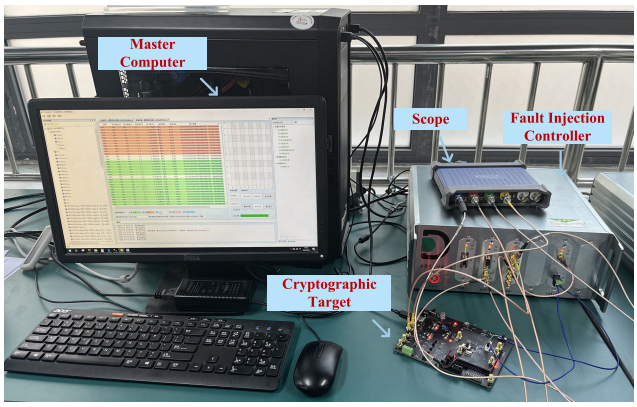

FIGURE 1. Voltage glitch injection experiment layout.

Pearson correlation coefficient is denoted as c. Suppose the attacker gets M power traces, each with N sampling points. The power consumption value corresponding to the i^{th} sampling point on the m^{th} trace is denoted as $t_{m,i}$ ($1 \leq m \leq M$, $1 \leq i \leq N$). Based on the power consumption model, the power consumption hypothesis value corresponds to the m^{th} trace under the key assumption k is denoted as $h_{m,k}$ ($1 \leq m \leq M$, $1 \leq k \leq K$). Under the key assumption k, the correlation coefficient of the i^{th} sampling point is calculated as follows:

$$c_{k,i} = \frac{\sum_{m=1}^{M} \left(h_{m,k} - \bar{h}_k\right)\left(t_{m,i} - \bar{t}_i\right)}{\sqrt{\sum_{m=1}^{M} \left(h_{m,k} - \bar{h}_k\right)^2} \sqrt{\sum_{m=1}^{M} \left(t_{m,i} - \bar{t}_i\right)^2}}$$

where $\bar{h}_k$ and $\bar{t}_i$ are the average of the power consumption model and the actual power consumption at the i^{th} sampling point respectively.

B. EXPERIMENT LAYOUT

The experimental platform mainly includes an attack target, fault injection controller, oscilloscope monitoring module, and data acquisition and processing module [18], as shown in Fig. 1.

We conduct a large number of fault injection attacks and establish fault probability traces [18] which are used in subsequent analyses.

III. FAULT CORRELATION ANALYSIS BASED ON FAULT PROBABILITY

A. SCHEME DESIGN

For the Advanced Encryption Standard algorithm (AES), attackers usually choose the sampling points near the S-box transformation operation as attack points, showing strong correlations between the actual power consumption and the power consumption model.

Taking the AES-128 algorithm as an example to implement FPFCA, as shown in Fig. 2. We conduct mass fault injection attacks on the output of the first S-box in the first round of encryption and establish the fault probability traces. If the first byte of the key is recovered, the attack is successful. The recovery of other bytes of the key is similar; attack the output of other S-boxes in the first round of encryption.

FPFCA attacks take advantage of the data dependency of cryptographic device faults. Many fault probability traces are used to analyze the fault probability at specific points, which is regarded as a function of the data being processed.

In all subsequent discussions, it is assumed that the processed data are subject to a uniform distribution.

The steps of FPFCA are as follows:

- **Step 1:** Select an intermediate value of the executed algorithm. The intermediate value must be a function *f(d, k)*, where d represents non-constant data, and k represents a small part of the key. We choose the plaintext as d and the first byte of the key as k.
- **Step 2:** Establish the fault probability traces. We calculate the fault probability of cryptographic devices when encrypting or decrypting M different data blocks. These plaintexts are recorded as vectors $\boldsymbol{m} = (m_1, m_2, \cdots, m_M)'$, where m_i represents a value corresponding to the i^{th} encryption or decryption operation. Many fault injection attacks are carried out on each m_i, and the corresponding fault probability trace is established as $\boldsymbol{p}_i' = (p_{i,1}, p_{i,2}, \cdots, p_{i,N})$, where N represents the length of the fault probability trace. The attacker establishes a fault probability trace for each of the M data. These fault probability traces are denoted as a matrix $\boldsymbol{P}_{M\times N}$.
- **Step 3:** Calculate the hypothetical intermediate value. For each possible key k, the corresponding hypothetical intermediate value is calculated, denoted as $\boldsymbol{k} = (k_1, k_2, \cdots, k_K)$, where K represents the number of all possible values of k. Given the data $\boldsymbol{m}$ and the key hypothesis $\boldsymbol{k}$, the hypothetical intermediate value can be calculated for all M encryptions and all K key assumptions: $v_{i,j} = f\left(m_i, k_j\right)(1\leq i \leq M, 1\leq j \leq K)$.
  It is obtained that the matrix $\boldsymbol{V}_{M\times K}$. The j^{th} column of $\boldsymbol{V}$contains all the intermediate values calculated by the key hypothesis k_j. In fact, the cryptographic device uses only one element in $\boldsymbol{k}$, which is denoted as k_c. Our target is to find the k_c, that is, to determine which column of $\boldsymbol{V}$ the device is dealing with during the M encryption or decryption.
- **Step 4:** Map the intermediate value to the fault probability. The hypothetical intermediate value $\boldsymbol{V}$ is mapped to the hypothetical fault probability matrix $\boldsymbol{H}$ by the HW model.
- **Step 5:** Compare the hypothetical fault probability value and the fault probability trace. Each column h_i of matrix $\boldsymbol{H}$ and each column p_j of matrix $\boldsymbol{P}$ are compared. The attacker compares the hypothetical fault probability value corresponding to each key hypothesis with the fault probability trace recorded at each location. The result of the comparison is a matrix $\boldsymbol{C}_{K\times P}$, where each element $c_{i,j}$ contains the comparison of the columns h_i and p_j. The higher the value of $c_{i,j}$, the higher the matching degree of the columns h_i and p_j.

The j^{th} sampling point on the m^{th} fault probability trace is denoted as $p_{m,j}$ $(1\leq m \leq M, 1\leq j \leq N)$. Based on the FPHW model [18], the hypothetical value of fault probability corresponding to the m^{th} trace under the key assumption i is denoted as $h_{m,i}$:

$$h_{m,i} = HW(Sbox(plaintext_m \oplus i)) \tag{1}$$

where HW represents the HW of the S-box output.

Under the key assumption i, the correlation coefficient of the j^{th} sampling point is denoted as $c_{i,j}$:

$$c_{i,j} = \frac{\sum_{m=1}^{M}\left(h_{m,i}-\bar{h}_i\right)\left(p_{m,j}-\overline{p_j}\right)}{\sqrt{\sum_{m=1}^{M}\left(h_{m,i}-\bar{h}_i\right)^2}\sqrt{\sum_{m=1}^{M}\left(p_{m,j}-\overline{p_j}\right)^2}} \tag{2}$$

where $\bar{h}_i$ and $\overline{p_j}$ represents the average values of the FPHW model and the actual fault probability at the i^{th} sampling point, respectively.

By finding the maximum value of matrix $\boldsymbol{C}$, the attacker can determine the correct key index k_c and time index t_c. The index of the maximum value is the result of the CPA attack.

B. ATTACK RESULT

We set the initial key to 0x04, conduct many fault injection attacks on 1000 random plaintexts, and establish fault probability traces. In the FPHW model [18], there is a negative correlation between the fault probability and HW, so the closer the correlation coefficient in the attack result is to -1, the stronger the correlation is.

We randomly select 6 plaintexts to attack and get 6 fault probability traces. The correlation coefficient results of sampling points on the traces calculated by Eq.(1) are shown in Fig. 3. When the candidate keys are 4,67,127, the correlation coefficient is the minimum value of -0.98. For that the candidate key is not unique, the key fails to recover.

Next, we randomly select 8 plaintexts to attack, and get 8 fault probability traces. The correlation coefficient results

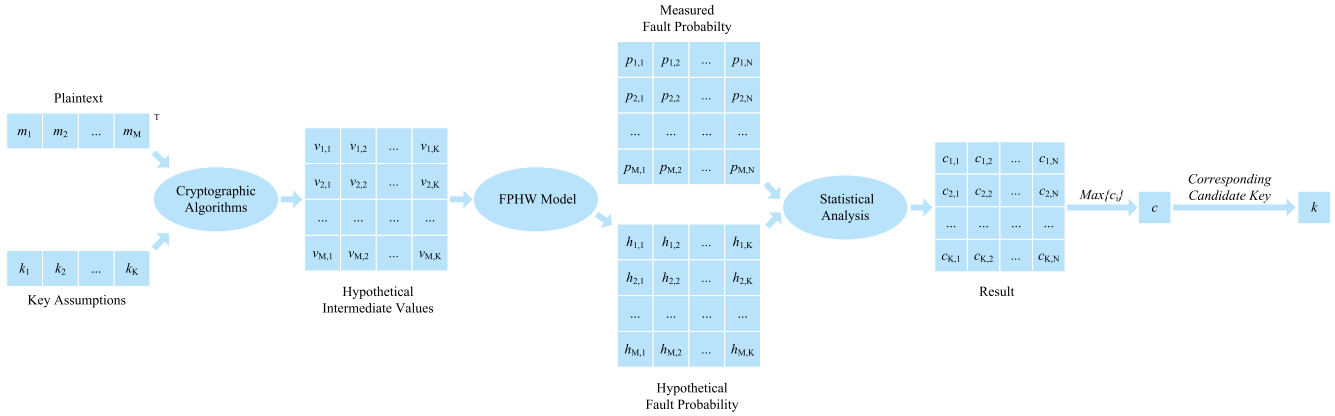

FIGURE 2. Attack flow of FPFCA.

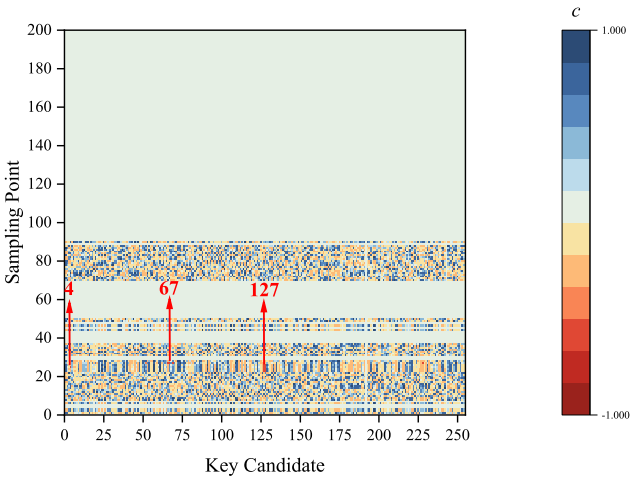

FIGURE 3. Attack under 6 random plaintexts.

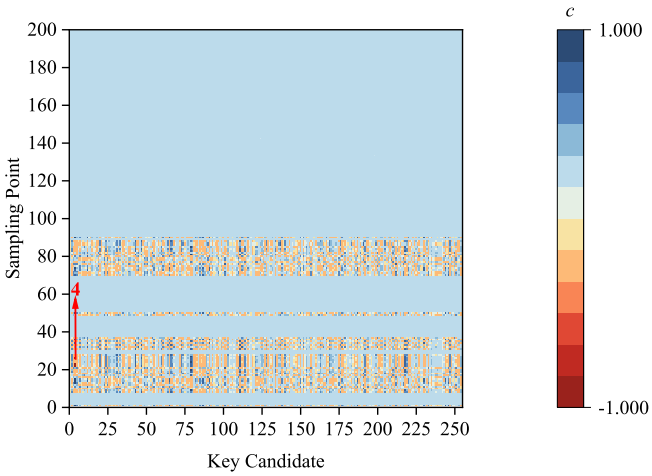

FIGURE 4. Attack under 8 random plaintexts.

of sampling points on the traces and the correlation coefficient results calculated by Eq.(1) are shown in Fig. 4. When the candidate key is 0x04, the correlation coefficient is the minimum value of -1, and the key is recovered successfully.

Under this method, the relationship between the success rate of key recovery and the number of fault probability traces is shown in Fig. 5. It can be seen that when the number of fault probability traces is 13 or more, the attacker has a 100% chance of recovering the key.

C. ANALYSIS

In the attack process, we establish the fault probability trace based on the FPHW model and align multiple data groups by introducing trigger signals. Therefore, there is no need for re-alignment and noise reduction, significantly reducing the difficulty of data processing.

The computational complexity of the FPFCA algorithm is denoted as λ:

$$\lambda = 16 \times 256 \times M \times N \tag{3}$$

According to the results of attack experiments, the computational complexity of the FPFCA algorithm is 1.06×10^7 by substituting M= 200 and N=13.

Compared with the existing CPA methods, the computational complexity of FPFCA has been significantly reduced, but the number of iterations is still significant. Therefore, we consider improving the calculation efficiency by lowering the values of M and N.

The calculation complexity is too high, which is mainly caused by a large amount of data redundancy in the selection of sampling points (M) and random plaintext (N) on the fault probability trace:

First, there are too many sampling points in the fault probability trace. The subkey only affects the fault probability in a few moments, so not all sampling points are essential in calculating correlation. Combined with the data dependence of fault probability, we need to locate the output of the S-box more accurately and reduce the search range of sampling points by selecting the sampling points with the prominent peak of fault probability trace as POIs.

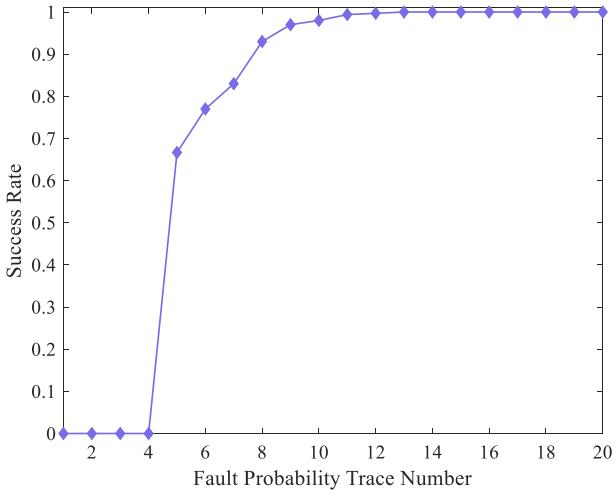

FIGURE 5. The relationship between the success rate and the number of fault probability traces.

TABLE 1. Hw distribution of single btye data.

HW	Probability
0	1/256
1	8/256
2	28/256
3	35/256
4	70/256
5	35/256
6	28/256
7	8/256
8	1/256

Second, there is much redundancy in selecting random plaintexts. All the sampling points in the fault probability traces are required to calculate the hypothetical intermediate value. Hence, the number of iterations in the calculation is enormous.

The HW of single-byte data obeys the binomial distribution. Tab. 1 shows the probability distribution of HW for uniformly distributed 8-bit data. The data with HW of 0 and 8 have the slightest probability of occurrence, and those with HW of 4 have the highest probability of occurrence. Therefore, there must be much repetition in the randomly selected plaintexts.

The experiments show that the difference in fault probability corresponding to the S-box output with the same HW is almost the same, as shown in Tab. 2.

If we select the plaintexts whose S-box outputs are of the same HW, column vectors of the hypothetical intermediate value mapped by the HW model are the same, and the corresponding fault probability traces are almost the same. In this case, it is impossible to recover the key. Therefore, we consider selecting plaintexts whose S-box outputs are of different HW, which can reduce the computational redundancy significantly.

TABLE 2. Fault probability of the same Hw data.

(A) HW = 3

Delay	Data1	Data2	Data3
	…		
2900	0.4611	0.4607	0.4626
2904	0.5711	0.5719	0.5707
2908	0.6500	0.6521	0.6518
2912	0.5756	0.5749	0.5742
2916	0.4033	0.4012	0.4029
	…		

(B) HW = 4

Delay	Data1	Data2	Data3
	…		
2900	0.4567	0.4573	0.4562
2904	0.5689	0.5681	0.5692
2908	0.6411	0.6409	0.6403
2912	0.5744	0.5741	0.5739
2916	0.3856	0.3859	0.3863
	…		

IV. FAULT CORRELATION ANALYSIS BASED ON SECONDARY FILTERING

A. SCHEME DESIGN

Based on the above considerations, we use fewer fault probability traces and more accurate POIs to reduce the number of iterations. We improve the FPFCA scheme and propose a fault correlation analysis attack based on secondary filtering (2F-FPCA). This method classifies and selects plaintexts based on HW and selects POIs more accurately, which dramatically reduces the calculation amount. The specific steps are shown in Fig. 6.

The steps of 2F-FPCA are as follows:

- **Step 1:** Select the output value of the first S-box in the first round of encryption in the AES-128 algorithm as the intermediate value.
- **Step 2:** Select plaintexts randomly and inject many voltage glitches in the encryption process to establish the fault probability traces.
  For each possible key k_i ($1 \leq i \leq K$), perform **steps 3-5**:.
- **Step 3:** Calculate the hypothetical intermediate value. Given $\boldsymbol{m} = (m_1, m_2, \cdots, m_M)'$, calculate the corresponding hypothetical intermediate value $v_{i,j} = f\left(m_i, k_j\right)(1 \leq i \leq M,\ 1 \leq j \leq K)$.
- **Step 4:** Map the intermediate value to the fault probability. The hypothetical intermediate value $\boldsymbol{v}_i = (v_{1,\mathrm{i}}, v_{2,\mathrm{i}}, \ldots, v_{M,i})$ is mapped to the hypothetical fault probability $\boldsymbol{h}_i = (h_{1,\mathrm{i}}, h_{2,\mathrm{i}}, \ldots, h_{M,i})$ by the

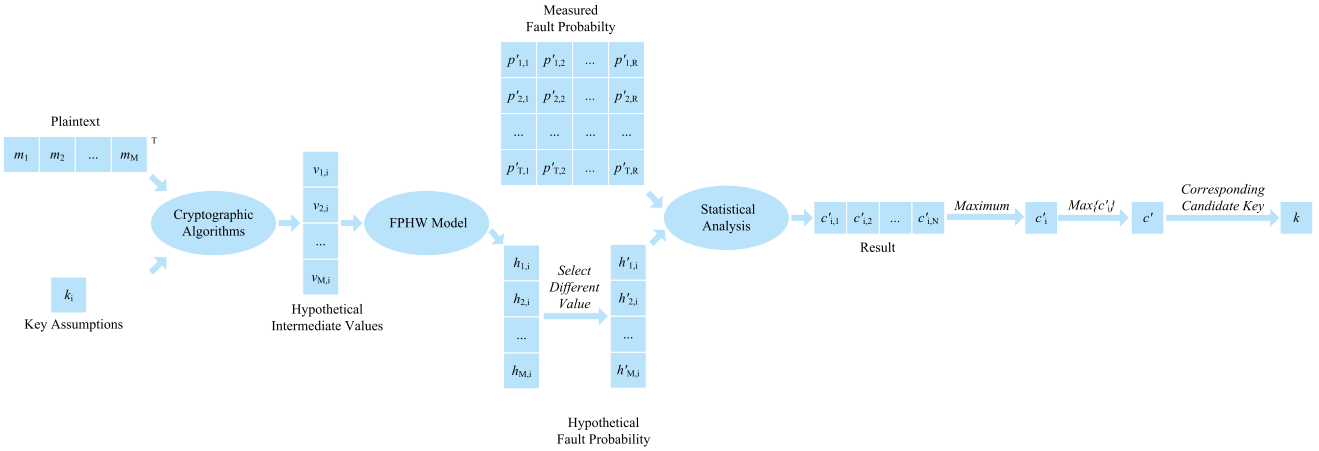

FIGURE 6. Attack flow of 2F-FPCA.

FPHW model. Remove the same value in h_i and get $\boldsymbol{h}'_i = (h'_{1,i}, h'_{2,i}, \ldots, h'_{T,i})$.

- **Step 5:** Compare the hypothetical fault probability value and the fault probability trace. Select the set of fault probability traces $P'_i = (p'_1, \boldsymbol{p}'_2, \ldots, \boldsymbol{p}'_T)$ corresponding to $\boldsymbol{h}'_i$, where $\boldsymbol{p}'_i = (p'_{i,1}, p'_{i,2}, \ldots, p'_{i,R})$ is the fault probability trace after POI selection. Compare $\boldsymbol{h}'_i$ and every column of $\boldsymbol{P}'_i$ and get $c'_i = (c'_{i,1}, c'_{i,2}, \ldots, c'_{i,R})$, which maximum value is denoted as c'_i. The maximum is denoted as $c' = max\{c'_1, c'_2, \ldots, c'_K\}$ for all the c'_i,. The key k'_i corresponding to c' is the correct key.

B. ATTACK PROCESS

According to the above scheme, we attack the implementation of the AES-128 algorithm based on MCU. The attack process is described in detail with the initial key of 0x03.

1) POI SELECTION

To reduce the computational complexity, we select the POIs in the fault probability trace to find the sampling points with the strongest correlation with the attack point to carry out subsequent attacks. From the above model verification results, we can see that the fault probability traces have a high degree of discrimination, so we first classify the attack traces based on the HW of the output data. Then, the summary of the difference method (SOD) is used to select the POIs.

For the fault probability traces $\mathrm{Trac}_i\ (i = 0, \cdots, 9)$ with HW $i (i = 0, 1, \cdots, 8)$, calculate

$$SOD = \sum_{i_1=0}^{7} \sum_{i_2=i_1+1}^{8} \left(\mathrm{Trac}_{i_1} - \mathrm{Trac}_{i_2}\right) \tag{4}$$

The SOD values of each sampling point are shown in Fig. 7. The higher the SOD value, the more significant the correlation between the sampling point and HW, so we selected 2900, 2904, 2908, 2912, and 2916ns as POIs.

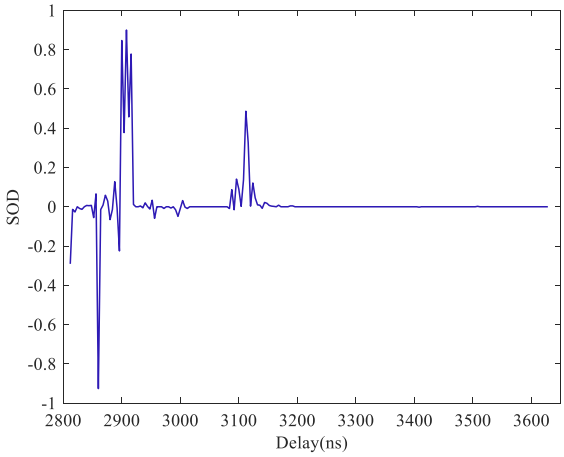

FIGURE 7. Selection of POIs.

2) RANDOM PLAINTEXTS SELECTION

We still choose the output of the first S-box in the first round of encryption as the intermediate value and inject many voltage glitches near the attack point. Find its maximum c'_i for each possible key hypothesis value $k_i\ (1 \leq i \leq K)$.

Take $k_4 = 0$x03 as an example to illustrate the attack process. The hypothetical fault probability is $h_4 = (5,5,5,4,3,5,3,3)$, and thus $h'_4 = (5,4,3)$. The set of fault probability traces after filtering is

$$\mathbf{p}'_3 = \begin{pmatrix} 0.4533 & 0.5689 & 0.6389 & 0.5733 & 0.3844 \\ 0.4567 & 0.5689 & 0.6411 & 0.5744 & 0.3856 \\ 0.4611 & 0.5711 & 0.6500 & 0.5756 & 0.4033 \end{pmatrix}.$$

3) ATTACK RESULT

Calculate that $\boldsymbol{c}'_3 = (-0.9973, -0.8660, -0.09443, -0.9997, -0.8930)$, $c'_3 = -0.9997$.

For all the $k_i (1 \leq i \leq 256)$, the corresponding c'_i calculated by Eq.(1) is shown in Fig. 8. As can be seen that the correlation coefficient of key 0x03 is closest to -1, 0x03 is supposed to be the correct key. Thus, key recovery is successful.

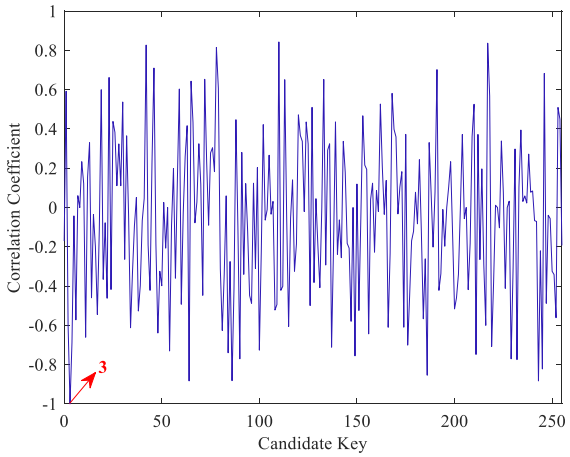

FIGURE 8. Correlation coefficient of candidate keys.

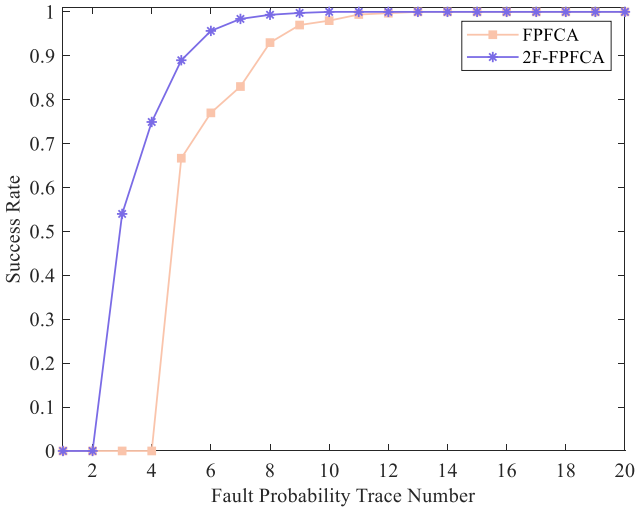

FIGURE 9. The relationship between the success rate and the number of fault probability traces.

C. ANALYSIS

Under the 2F-FPCA method, the relationship between the success rate of key recovery and the number of fault probability traces is shown in Fig. 9. The number of attack traces required for the proposed 2F-FPCA is less than that needed for FPFCA. When the number of fault probability traces is 10 or more, the attacker has a 100% chance of recovering the correct key.

We select 10 plaintexts randomly for key recovery. Under each key assumption, only 4.45 traces are needed by selecting plaintexts to complete the key recovery on average. By substituting $M = 4.45$ and $N = 5$ into Eq.(2), the algorithm's computational complexity is calculated as 9.11×10^4.

In recent years, some CPAs and improvement methods for the AES-128 algorithm have been continuously proposed, such as CPA with biased power traces [13], CPA based on CPA with multiple filtering [19], and block CPA based on artificial intelligence [20].

TABLE 3. Comparison of Cpa schemes.

(A)NUMBER OF TRACES REQUIRED FOR SUCCESS ATTACK

Scheme	Trace Number of Key Recovery
Biasing Power Traces [13]	35
MS-CPA with KE [19]	30
Hilling-climbing-based approach [20]	28
CPA [7]	13
2F-FPCA	10

(B)COMPUTATIONAL COMPLEXITY

Scheme	Computational complexity
Biasing Power Traces [13]	1.22×10^9
CPA [7]	9.75×10^8
FPFCA	1.06×10^7
2F-FPCA	9.11×10^4

Next, as shown in Tab.3, we compare the above CPA attack methods with the FPFCA and 2F-FPCA attack methods proposed in this paper. We propose two criteria for performance evaluation: the number of traces required to recover the full key and computational complexity.

As described in Tab.3, the number of traces and sampling points required for the attack directly affects the computational complexity, consistent with Eq.(2). Therefore, the attacker must select the appropriate number of traces and sampling points to achieve high execution efficiency while considering the accuracy of key recovery.

V. CONCLUSION

This paper proposes a fault correlation analysis method based on secondary filtering. Based on the AES-128 algorithm of the MCU, we conducted experiments and achieved key recovery using 10 fault probability traces.

Compared with the existing CPA methods, 2F-FPCA has the following advantages:

1) It is less affected by noise. We choose the FPHW model, which does not need to collect ciphertext and only pays attention to the response of the attack.

2) The correct key is of apparent characteristics. The correlation coefficient between the actual fault probability corresponding to the correct key and the hypothetical fault probability value based on the FPHW model is as high as -1.

3) The computational complexity is low. We select plaintexts with different HW of S-box output and POIs to reduce the number of traces and sampling points. Therefore, the computational complexity is significantly reduced, improving the attack efficiency.

Due to the difference in cryptographic algorithms and selection methods, the selection results of POIs are different. Next, we will use Deep Learning to realize the automatic selection of POIs. At the same time, we consider expanding the range of cryptographic algorithms targeted by attacks. Testing new and masked cryptographic algorithms is the critical research content for the next step.

REFERENCES

[1] M. Devi and A. Majumder, "Side-channel attack in Internet of Things: A survey," in *Applications of Internet of Things*. Singapore: Springer, 2021, pp. 213–222.

[2] Y. Kulah, B. Dincer, C. Yilmaz, and E. Savas, "SpyDetector: An approach for detecting side-channel attacks at runtime," *Int. J. Inf. Secur.*, vol. 18, no. 4, pp. 393–422, Aug. 2019.

[3] D. Poggi, P. Maurine, T. Ordas, and A. Sarafianos, "Protecting secure ICs against side-channel attacks by identifying and quantifying potential EM and leakage hotspots at simulation stage," in *Proc. Int. Workshop Constructive Side-Channel Anal. Secure Design*, Lugano, Switzerland, Oct. 2021, pp. 129–147.

[4] I. Kabin, Z. Dyka, D. Klann, and P. Langendoerfer, "EC P-256: Successful simple power analysis," 2021, *arXiv:2106.12321*.

[5] F. Schellenberg, D. R. E. Gnad, A. Moradi, and M. B. Tahoori, "An inside job: Remote power analysis attacks on FPGAs," *IEEE Des. Test*, vol. 38, no. 3, pp. 58–66, Jun. 2021.

[6] F. Schuhmacher, "Canonical DPA attack on HMAC-SHA1/SHA2," in *Proc. Int. Workshop Constructive Side-Channel Anal. Secure Design*, Leuven, Belgium, Apr. 2022, pp. 193–211.

[7] E. Brier, C. Clavier, and F. Olivier, "Correlation power analysis with a leakage model," in *Proc. 6th Int. Workshop Cryptograph. Hardw. Embedded Syst. (CHES)*, pp. 16–29, Aug. 2004, doi: 10.1007/978-3-540-28632-5_2.

[8] A. Heuser, O. Rioul, and S. Guilley, "Good is not good enough: Deriving optimal distinguishers from communication theory," in *Proc. Int. Workshop Cryptograph. Hardw. Embedded Syst.*, Busan, South Korea, Sep. 2014, pp. 55–74.

[9] L. David and A. Wool, "A bounded-space near-optimal key enumeration algorithm for multi-subkey side-channel attacks," in *Proc. Cryptographers' Track RSA Conf.* San Francisco, CA, USA, Feb. 2017, pp. 311–327.

[10] R. Poussier, F.-X. Standaert, and V. Grosso, "Simple key enumeration (and rank estimation) using histograms: An integrated approach," in *Proc. Int. Conf. Cryptograph. Hardw. Embedded Syst.*, Santa Barbara, CA, USA, Aug. 2016, pp. 61–81.

[11] D. P. Martin, J. F. O'connell, E. Oswald, and M. Stam, "Counting keys in parallel after a side channel attack," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.*, Auckland, New Zealand, Nov. 2015, pp. 313–337.

[12] T.-H. Le, J. Clédiére, C. Canovas, B. Robisson, C. Servière, and J.-L. Lacoume, "A proposition for correlation power analysis enhancement," in *Proc. Int. Workshop Cryptograph. Hardw. Embedded Syst.*, Yokohama, Japan, Oct. 2006, pp. 174–186.

[13] Y. Kim, T. Sugawara, N. Homma, T. Aoki, and A. Satoh, "Biasing power traces to improve correlation power analysis attacks," in *Proc. 1st Int. Workshop Constructive Side-Channel Anal. Secure Design (COSADE)*, 2010, pp. 77–80.

[14] M. Ouladj, P. Guillot, and F. Mokrane, "Chosen message strategy to improve the correlation power analysis," *IET Inf. Secur.*, vol. 13, no. 4, pp. 304–310, Jul. 2019.

[15] Q. L. Meunier, "FastCPA: Efficient correlation power analysis computation with a large number of traces," in *Proc. 6th Workshop Cryptogr. Secur. Comput. Syst.*, Jan. 2019, pp. 7–12.

[16] N.-T. Do and V.-P. Hoang, "An efficient side channel attack technique with improved correlation power analysis," in *Proc. Int. Conf. Ind. Netw. Intell. Syst.*, Hanoi, Vietnam, Aug. 2020, pp. 291–300.

[17] S. Mangard, E. Oswald, and T. Popp, *Power Analysis Attacks: Revealing the Secrets of Smart Cards*, vol. 31. Cham, Switzerland: Springer, 2008.

[18] T. Wu, D. Zhou, L. Du, and S. Wang, "Fault template attack based on fault probability," *IEEE Access*, vol. 11, pp. 71705–71713, 2023.

[19] Y. Ding, L. Zhu, A. Wang, Y. Li, Y. Wang, S. M. Yiu, and K. Gai, "A multiple sieve approach based on artificial intelligent techniques and correlation power analysis," *ACM Trans. Multimedia Comput., Commun., Appl.*, vol. 17, no. 2s, pp. 1–21, Jun. 2021.

[20] Y. Ding, Y. Shi, A. Wang, Y. Wang, and G. Zhang, "Block-oriented correlation power analysis with bitwise linear leakage: An artificial intelligence approach based on genetic algorithms," *Future Gener. Comput. Syst.*, vol. 106, pp. 34–42, May 2020.

TONG WU was born in Ma'anshan, Anhui, China, in 1996. She received the B.S. degree in information security from the Naval University of Engineering, Wuhan, China, in 2018, where she is currently pursuing the M.S. degree in cyberspace security. Her current research interest includes cryptographic chip security evaluation.

DAWEI ZHOU was born in Xuzhou, Jiangsu, China, in 1980. He received the B.S. degree from the China University of Mining and Technology, Xuzhou, in 2002, and the M.S. degree from the Naval University of Engineering, Wuhan, China, in 2008. He is currently an Associate Professor with the Naval University of Engineering. His current research interest includes information security.

LEI DU received the B.S. and M.S. degrees from Beihang University, Beijing, China, in 2002 and 2005, respectively. From 2005 to 2015, he was the General Manager of the Security Department, BCTC. He is currently the General Manager of the DPLS Laboratory, Beijing. His current research interest includes chip security design and testing.

SHIWEI WANG received the B.S. and Ph.D. degrees from the Beijing University of Posts and Telecommunications, Beijing, China, in 2013 and 2019, respectively. Since 2019, he has been a Senior Security Specialist and the Research and Development Department Manager of the DPLS Laboratory. His current research interests include side-channel analysis, fault injection, and security tests.