IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received January 10, 2022, accepted February 7, 2022, date of publication February 10, 2022, date of current version February 17, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3150833

Low Complexity Correlation Power Analysis by Combining Power Trace Biasing and Correlation Distribution Techniques

NGOC-TUAN DO[1], VAN-PHUC HOANG[1], (Member, IEEE), AND CONG-KHA PHAM[2], (Member, IEEE)
[1]Institute of System Integration, Le Quy Don Technical University, Hanoi 100000, Vietnam
[2]Department of Information and Network Engineering, The University of Electro-Communications (UEC), Chofu-shi, Tokyo 182-8585, Japan

Corresponding author: Van-Phuc Hoang (phuchv@lqdtu.edu.vn)

This work was supported by the Vietnam National Foundation for Science and Technology Development (NAFOSTED) under Grant 102.02-2020.14.

ABSTRACT Side channel attack (SCA) is a class of crypt-analytic attacks for security evaluation of cryptographic and embedded microprocessor implementations. Among several SCA approaches, the correlation power analysis (CPA) is an efficient way to recover the secret key of the specific cryptographic algorithms running on the target devices such as embedded microprocessors. However, the evaluation process is time-consuming since a large number of traces are required to overcome the impact of noise. Hence, this paper proposes new methods to reduce the computation time by using Point of Interest (POI) extractor with the power trace biasing technique and the correlation distribution for the low complexity correlation power analysis (CPA). The theoretical explanations are provided and the experiments on different platforms such as ASCAD and RISC-V processor based databases are conducted to justify the proposed techniques. Especially, our experiments are performed with different protected schemes such as masking, hiding and combined hiding-masking techniques. The experimental results indicate that our proposed methods provide reliable results in comparison with the standard CPA. By using only a half of the power traces for taking the POIs, our first proposal not only decreases the execution time approximately by half but also enhances the success rate of the attack. Moreover, the second method based on power trace biasing technique is proposed in order to achieve better results and reduce the number of traces needed for selecting the POIs. With only 28.9% of given power traces needed, our second proposed technique reduces the execution time to only 2.6 times of the standard CPA.

INDEX TERMS Correlation power analysis, side channel attack, embedded microprocessor security, power trace biasing, correlation distribution.

I. INTRODUCTION

Cryptography is one of the key solutions to provide confidentiality, integrity, and authenticity of data. However, the implementation of cryptographic algorithms is facing various security threats. Side channel attack (SCA) is a realistic thread that maliciously deciphers the secret information by exploiting the side channel information such as the power consumption or electromagnetic radiation emanation [1]. In addition, SCA can be used as a promising tool for hardware Trojan detection [2], [3]. Therefore, utilizing unintentional leakage from a cryptographic device is critical in security evaluation for cryptographic implementations. Several evaluation/certification implementations are running SCA daily on devices under test to verify the security level. To perform the SCA evaluation, the person usually switches between the role of a designer and the role of an attacker. For a wide range of SCA applications, in this paper, we consider SCA as a tool of crypt-analysis which reveals the secret key and we also refer to the word "attacker" for either attacker or evaluator in the rest of this paper.

The evaluation process is time-consuming since a large number of traces are required to overcome the impact of noise. Especially, it becomes less efficient when the SCA

The associate editor coordinating the review of this manuscript and approving it for publication was Diana Gratiela Berbecaru.

countermeasures are applied to improve the desired security level. In general, these countermeasures can be divided into two groups. The first one is the so-called algorithmic countermeasures [4], [5]. The main purpose of these techniques is to randomize the intermediate values that are the target of attackers. Masking is the most popular technique for this type of countermeasures. This method exponentially increases the number of samples on each power trace as shown in [6]. In other words, the evaluation process must implement the high-order DPA. The second technique is the hiding countermeasure. The goal of this method is to bury the leakage position of the power consumption in a different kind of noise. Several hiding countermeasures have been proposed, such as a correlated power noise generator [7], randomly changing clock frequency and supply voltage of circuits [8], wave dynamic differential logic (WDDL) [9] or recently, random dynamic back-gate biasing technique (RDBB) [10]. These techniques lead to the reduction of the SNR metric or the disarrangement of time at which the attacked intermediate result is processed. Consequently, the number of power traces for evaluation is increased significantly [11]. To accelerate the certification process, a method to reduce the computation complexity of the SCA technique is highly required.

A. RELATED WORKS

In literature, several techniques have been proposed to reduce the computation complexity of the standard SCA method. The most common method is selecting a small subset of points where the leakage prevails. This issue has been dealt previously by some researchers. In [12], Chari *et al.* proposed to use the templates to profile the interesting time samples. This method requires a great number of power traces to create the templates T on n different sensitive values, i.e., Hamming weight (HW) values or simply, the values of a subkey. Then, the points of interest (POIs) are found as the points which maximize $D_i = \sum_{k_1,k_2=1}^{n} \left(T_{k_1,i} - T_{k_2,i}\right)$ where T_k is calculated as $T_{k,i} = \frac{1}{n_c}\sum_{j=1}^{n_c} t_{j,i}$ and n_c is the number of power traces which belong to the class k. Two other improvements of Chari's method called *SOSD* and *SOST* were introduced by Gierlichs *et al.* [13]. Recently, Rioja *el al.* [14] presented a method based on Estimation of Distribution Algorithms called EDA based profiled attack (EDA based PA) for SCA. The authors have proved that EDA-PA outperforms all state-of-the-art pre-processing and POI selection methods. Even though the mentioned techniques are considered the most powerful from the side channel attacks, they have a practical problem of requiring the access to a reference device. Indeed, an evaluation process using a clone device is not always available in practice. Furthermore, a huge number of power traces with random inputs (plaintexts and keys) must be collected and processed. Therefore, the evaluation process based on profiled attacks may cause an increasing of the cost (a reference device needed) and time-consuming. Thus, to ease the evaluation process in general, a low complexity SCA method without using a reference device needs to be developed.

Fortunately, the non-profiled attack is suitable for SCA evaluation in the case of reference device is not available. Indeed, by computing the correlation of the ground-truth models and the power traces recorded from the target device, the non-profiled SCA can recover the secret key. Therefore, evaluation process based on non-profiled attacks do not require any reference device. To reduce the computation complexity of the SCA technique in the non-profiled context, several methods have proposed. Kim *et al.* [15] proposed a method called power trace extractor. This method extracts a small set of traces with a high signal-to-noise ratio (SNR) distributed in both tails of the distribution range to enlarge the variance of the exploitable consumption component in the power trace. Similarly, an empirical method uses the adaptive chosen-plaintext CPA attacks (ACP-CPA) [16]. The authors tried to resolve the drawback of discarding too many traces in the extractor proposed by Kim *et al.* [15]. However, this technique requires many requests to chose adaptive plaintexts for all bytes. This leads to the time-consuming computations. Recently, an improvement of power traces extractor presented by Ou *et al.* [17] called Maximizing Estimated SNR First (MESF). Unlike the proposed in [15], the novelty of this work is that this technique extracts the subset of power traces with the smallest estimated noise and maximize the variance of the data-dependent power consumption. Consequently, by using the high SNR samples, the computation complexity of the attacks can be reduced instead of increasing the success rate (SR). However, Ou's techniques are based on the mean power consumption of plaintext byte values. This constraint limits these techniques to the scenarios where the attacker has a large number of traces to estimate the mean values. Hence, it leads to the high computational cost to perform an attack with a large number of power traces. When it comes to powerful SCA techniques, it is indispensable to mention deep learning (DL) based SCA. They can be used to break the cryptographic implementation without any pre-processing methods. However, most of studies have only focused on applying DL to perform the profiled SCA [18]–[20]. Only few DL based techniques have been proposed in the non-profiled context [21], [22]. One of them is the state-of-the-art non-profiled based technique called differential deep learning analysis (DDLA). However, we have already shown that it is sensitive with the additional Gaussian noise [23]. Despite attacking successfully the masking countermeasure, this technique is less effective than the conventional second-order DPA which can overcome the hiding countermeasure by using a large enough number of power traces. In addition, the DL based attack often requires a huge number of power traces. This leads to a very high computation time for attacking the combined hiding-masking method.

In summary, we provide Table 1 to compare use cases of state-of-the-art SCA techniques. As stated previously, the profiled SCA requires a copy version of the target device to

TABLE 1. The state-of-the-art techniques for reducing complexity of SCA.

Method	Scope of method			Device requirements	
	Pre-processing	POI selection	Key recovery	Copy of target	Target
SOSD	✗	✓	✗	✓	✓
SOST	✗	✓	✗	✓	✓
SNR	✗	✓	✗	✓	✓
EDA-Based PA	✗	✓	✓	✓	✓
DL based profiled SCA	✗	✗	✓	✓	✓
PCA	✓	✓	✗	✗	✓
Correlation	✗	✓	✓	✗	✓
ACP-CPA	✓	✗	✗	✗	✓
MESF	✓	✗	✗	✗	✓
DDLA	✗	✓	✓	✗	✓

complete the evaluation process. Therefore, all methods using reference device such as SOSD, SOST, SNR, EDA-based PA and DL based profiled SCA, are out of the scope of this paper. In the non-profiled scenario, the evaluation process requires many steps to find the POI, the pre-processing to create a new set of traces, and using a distinguisher such as CPA on new traces to reveal the correct subkey. These steps must be done for every subkey of the secret key. Hence, it leads to a more expensive and time-consuming evaluation process. From Table 1, we are more interested in methods like correlation technique and DDLA. They can be used for both POI selection and key recovery. However, due to the limitation of DDLA for cases with the presence of the Gaussian noise, this work focuses on providing an improved distinguished CPA that automatically perform the tuning of POIs, together with key recovering steps. We consider the attack scenario where the attacker has enough power traces to guarantee successful CPA attacks.

B. OUR CONTRIBUTIONS

To the best of our knowledge, there is not any paper in the literature mentioning the implementation and optimization of the POIs selection/tuning step together with key recovery steps automatically. We will validate the efficiency of the proposed method on the popular countermeasure techniques including hiding, masking and combined masking-hiding methods. For convenience in simulating multiple levels of hiding countermeasure, we consider noise-generation techniques as hiding countermeasures for the rest of this paper. We will also demonstrate that our proposed method outperforms the non-profiled machine learning based attacks for cases with the presence of the Gaussian noise. In particular, the experimental results will clarify the efficiency of our proposed on various countermeasures such as masking, hiding, and combined hiding-masking methods with the dataset collected by the emerging open reduced instruction set computer (RISC-V) processor platform and the ASCAD database [18].

The main contributions of this paper are as follows:

- We provide a theoretical analysis of the correlation distribution and propose a POI extractor method named partial correlation power analysis (P-CPA). We perform a standard CPA on a half of the given power traces to take all candidates of the correct samples which have the highest correlation value. The number of POIs is very small compared to the length of the original power trace (i.e., 2446.5 times and 99.19 times smaller than in the cases of ASCAD database and our RISC-V based data, respectively).

- We propose a novel POI extractor based on both correlation distribution and power trace biasing technique called power trace biasing based P-CPA (BP-CPA) in order to improve the probability of taking the correct samples and reduce the number of needed traces. It is highly noted that in the proposed technique, the extractor requires only 28.9% of the given power traces compared with 50% in the P-CPA technique.

- By using the proposed POI extractors, we provide two auto-CPA algorithms that automatically perform the POI selection, together with key recovering. As results, the computation time is reduced by approximately 2.0 times with P-CPA and 2.6 times with BP-CPA, compared to the conventional CPA.

The rest of this paper is organized as follows. The leakage characteristic of samples, CPA and the correlation distribution are briefly introduced in Section II. Next, in Section III, we will present the computation complexity of CPA and the proposed ideas to reduce the computational complexity and an efficient CPA technique that allows recovering secret key bytes faster than the conventional one. Then, the experimental results clarifying the efficiency of the proposed methods with different kinds of countermeasures will be shown in Section IV. Finally, Section V concludes the paper.

II. GENERAL BACKGROUND

A. LEAKAGE CHARACTERISTICS OF SAMPLES

Let $p_{i=\{1,n\}}$ denote the i^{th} plain-text encrypted in AES-128 algorithm, where n is number of plain-texts. We denote

$k_{j=\{1,16\}}$ as the subkey number j of secret key. The power traces that are recorded during the encryption are denoted as $t_{1\ldots n,1\ldots l}$, where l is the number of samples recorded per encryption. According to [24], the power consumption of a single sample $t(\tau)$ can be express as the sum of a data-dependent component $t_d(\tau)$, an operation-dependent component $t_o(\tau)$, switching noise $t_{sw.noise}(\tau)$, electronic noise $t_{el.noise}(\tau)$, and the constant component $t_{const}(\tau)$. All components are independent with each other as shown in (1).

$$t(\tau)=t_o(\tau)+t_d(\tau)+t_{sw.noise}(\tau)+t_{el.noise}(\tau)+t_{const}(\tau) \tag{1}$$

Let $t_{exp}(\tau)$ denote the exploitable component including the operation-dependent component $t_o(\tau)$ and data-dependent component $t_d(\tau)$, see 2. Let $t_{noise}(\tau)$ denote the noise component consist of the switching noise $t_{sw.noise}(\tau)$ and the electronic noise $t_{el.noise}(\tau)$, as in (3).

$$t_{exp}(\tau) = t_d(\tau) + t_o(\tau) \tag{2}$$

$$t_{noise}(\tau) = t_{sw.noise}(\tau) + t_{el.noise}(\tau) \tag{3}$$

The SNR of the time sample $t(\tau)$ is the ratio of the variance of exploitable power consumption $t_{exp}(\tau)$ to the variance of noise component $t_{noise}(\tau)$. Therefore, the formula of SNR can be simplified as:

$$SNR = \frac{\sigma^2\left(t_{exp}(\tau)\right)}{\sigma^2\left(t_{noise}(\tau)\right)} \tag{4}$$

As explained in [24], in term of one-bit DPA, the attacker considers one of the 8-bit intermediate values. Therefore, the power consumption of the other 7 bits is switching noise and its variance is larger than 0. In term of CPA, all bits of intermediate values are considered, the variance of the switching noise is $t_{sw.noise}(\tau) = 0$. The constant component $t_{const}(\tau)$ is also zero for a sample $t(\tau)$.

B. CORRELATION POWER ANALYSIS (CPA)

Based on the plaintext and all possible values for the subkey, hypothesis values for the intermediate results are calculated. This lead to a matrix $I_{1\ldots K,1\ldots n}$ of hypothetical intermediate values, where K is the number of possible values for the subkey k.

According to [25], CPA exploits the correlation between the real power consumption t and the power consumption model H of the running device at some certain points of time which must depend on the fixed secret key k and the plaintext p changed for each trace. In CPA attack, the Pearson correlation coefficient is the common measure to determine the linear relationship between two variables. The define of the Pearson correlation coefficient r is shown in equation (5). r estimates the correlation ρ between two variables based on n power traces. $\bar{h}_k$ and $\bar{t}_i$ are the average values of the power consumption model and real power consumption at the instant i $(1 \leq i \leq l)$, respectively.

$$r_{k,i} = \frac{\sum_{j=1}^{n}(h_{j,k}-\bar{h}_k)(t_{j,i}-\bar{t}_i)}{\sqrt{\sum_{j=1}^{n}(h_{j,k}-\bar{h}_k)^2\sum_{j=1}^{n}(t_{j,i}-\bar{t}_i)^2}} \tag{5}$$

The Pearson correlation between the power consumption model and the power trace is calculated for every value of k and t. It results in the matrix $R = r_{1\ldots K,1\ldots l}$ of correlation coefficients. There is an alternative form of the correlation equation for online calculations and it allows us to add one trace per time without re-summing all of the past data. This form is presented in (10), as shown at the bottom of the page.

Further analysis of the correlation between the real power consumption value and the power consumption model is presented in [26]. The important Formula 6.5 given in this work [26] can be expressed as:

$$\rho(h_k, t) = \frac{\rho\left(h_k, t_{exp}\right)}{\sqrt{1+\frac{1}{SNR}}} \tag{6}$$

For a conventional CPA, the correlation $\rho\left(h_k, t_{exp}\right)$ between the power consumption model and the data-dependent component is a constant for a time sample. From equation (6), it is clear that the SNR determines the correlation $\rho(h_k, t)$, and $\rho(h_k, t)$ approaching a constant when the number of power traces used in attack is large enough.

C. DISTRIBUTION OF CORRELATION COEFFICIENTS IN CPA

In the previous section, we have shown that the correlation coefficients are calculated at every point of every power trace. Using the same notation in Section I.A, we can see that l different values of r correspond to l points are calculated using n power traces. Therefore, the distribution of values of r on each point after repeated samples of n power traces is the sampling distribution. According to [24], in SCA context, if a sufficiently large number of traces can be measured $(n \geq 30)$, the Fisher's transformation can be used to map the random variable R to a random variable Z that has a normal distribution, as in the equation (7). The mean of Z is then given by μ in (8) and the variance in (9).

$$R \mapsto Z = \frac{1}{2}\ln\frac{1+R}{1-R} \tag{7}$$

$$\mu = E(Z) = \frac{1}{2}\cdot\ln\frac{1+\rho}{1-\rho} \tag{8}$$

$$\sigma^2 = Var(Z) = \frac{1}{n-3} \tag{9}$$

$$r_{k,i} = \frac{n\sum_{j=1}^{n}h_{j,k}t_{j,i}-\sum_{j=1}^{n}h_{j,k}\sum_{j=1}^{n}t_{j,i}}{\sqrt{\left(\left(\sum_{j=1}^{n}h_{j,k}\right)^2-n\sum_{j=1}^{n}h_{j,k}^2\right)\left(\left(\sum_{j=1}^{n}t_{j,i}\right)^2-n\sum_{j=1}^{n}t_{j,i}^2\right)}} \tag{10}$$

In [24], the authors aim to calculate the lower bound for the number of power traces needed to attack successfully. Their observation depends on the $\rho_{\max}$ which occurs between the correct hypothesis power consumption and one of the devices at the correct sample time $t(\tau_{kc,tc})$. The number of traces that are needed to take the correct key in practice is mainly determined by the distance between the sampling distribution with $\rho = 0$ and $\rho = \rho_{\max}$. All values of R are drawn from one of these two sampling distributions. From (8) and (9), we can clarify that the estimator values $\rho = 0$ and $\rho = \rho_{\max}$ are normally distributed with $\mu_{\rho_0} = 0$ and $\mu_{\rho_{\max}} = \frac{1}{2}\ln\frac{1+\rho_{\max}}{1-\rho_{\max}}$, respectively. The more overlap there is between these distributions, the less clearly a significant peak is in R.

In order to measure the distance between the distributions, the authors calculate the probability that a value drawn from the distribution with $\rho = \rho_{\max}$ is bigger than the one that is drawn from the distribution with $\rho = 0$. This calculation is given by formula (11) and can be transformed to formula (12).

$$\alpha = \Phi\left(\frac{\frac{1}{2}\ln\frac{1+\rho_{\max}}{1-\rho_{\max}} - \frac{1}{2}\ln\frac{1+0}{1-0}}{\sqrt{\frac{2}{n-3}}}\right) \tag{11}$$

$$n = 3 + 8\left(\frac{Z_\alpha}{\ln\left(\frac{1+\rho_{\max}}{1-\rho_{\max}}\right)}\right)^2 \tag{12}$$

In [26], the authors performs the further analysis and provide several comprehensive examples of DPA attacks. Especially, the author elaborates on issues like the simulation of DPA attacks and the calculation of the number of power traces that are needed to perform DPA attacks successfully. Table 6.1 in [26] provides the results to illustrate the relationship between $\rho_{\max}$ and the calculated number of traces according to (12). Especially, the authors indicate that since $\sigma = 1\big/\sqrt{n-3} \approx 1\big/\sqrt{n}$ with the large enough value of n, the estimators for all correlation coefficients before and after the attacked intermediate results is processed are essentially located in the interval $\pm 4\sigma = 4\big/\sqrt{n}$. Moreover, the authors in [26] also indicated that $n \approx \frac{28}{\rho_{ck,ct}^2}$. The purpose of this work is not to find the lower bound of the number of power traces. We assume that the n is already sufficient and we use n power traces in total to implement the CPA efficiently. Our proposed technique will be discussed more detail in the next section.

III. PROPOSED METHODS FOR REDUCING THE COMPUTATION COMPLEXITY OF CPA

For the computation complexity of the CPA technique, firstly, we present the complexity of the attack on a protected device using hiding countermeasure. Then, we describe the computation cost on masking protected device. As described in Section II.A, the attackers need n power traces to perform a standard DPA successfully. Therefore, the number of computations is proportional to $l * n * K$. In addition, hiding countermeasure will bury the intermediate results by adding different kinds of noise. Consequently, the number of correlation coefficients is reduced and the number of power traces is increased, according to (12).

In term of the second-order DPA attack, the authors in [6] have shown that the complexity is the square order in comparison to a standard DPA attack on a interval with the length of l. This means that the length of each power trace will increase from l to $\frac{l(l-1)}{2}$. Therefore, the complexity of the second-order technique is proportional to $\frac{l(l-1)}{2} * n * K$. In fact, the more noise in the power trace, the more measurements are needed for a successful DPA attack. Therefore, cryptographic devices are typically protected by a combination of hiding and masking countermeasures. This leads to a very time consuming task to perform an attack on the highly protected devices.

A. PARTIAL CORRELATION POWER ANALYSIS

It is clear that reducing the length l of each power trace can reduce significantly the complexity of the DPA attack. In this section, we describe the process of our proposed method and explain the way to reduce the number of samples of a power trace in computing process. As mentioned in the previous section, all values of matrix R are drawn from one of two sampling distributions. By using Fisher's transformation, we assume that all values of matrix $R \rightarrow Z$ will be drawn from two normal distributions $N_1\,(0, \sigma_n)$ and $N_2\left(\mu_{\rho_{\max}}, \sigma_n\right)$, where $\sigma_n = \frac{1}{\sqrt{n}}$ as illustrated in Fig. 1. From the equation (9), it is clear that an attacker can decrease the overlap between these two normal distributions by increasing the number of traces.

As explained in [26], the recorded traces are quite long compared to the interval during which the attacked intermediate results are processed. Usually, many operations are executed during the recording and completely independent of the attacked intermediate values. Therefore, most of correlation values $\rho_{ck,1} \ldots \rho_{ck,l}$ are usually zero in practice. Indeed, we assume that all values of ρ except $\rho_{ck,tc}$ will be distributed in the form of $N_1(0, \sigma_n)$, and $\rho_{ck,tc}$ at the sample $t(\tau_{kc,tc})$ will be distributed in the form of $N_2\left(\mu_{\rho_{\max}}, \sigma_n\right)$. Hence, we have some observations as follows.

Firstly, if we reduce the number of traces from n to $n/2$ and $n/3$, the standard deviation of the normal distribution $N_1(0, \sigma_n)$ will be changed from σ_n to $\sigma_{n/2}$, $\sigma_{n/3}$. The shape of distribution $N_1(0, \sigma_n)$ will be changed to $N_3(0, \sigma_{n/2})$, $N_5(0, \sigma_{n/3})$, respectively, as illustrated in Fig. 1.

Secondly, from the equation (6), the correlation $\rho_{ck,tc}$ will be a stable value when the number of traces is large enough. In this case, we assume that the $\rho_{ck,tc}$ will distribute around and near the mean $\mu_{\rho_{\max}} = \frac{1}{2}\ln\frac{1+\rho_{\max}}{1-\rho_{\max}}$ and then keeps consistent. Therefore, the value of mean $\mu_{\rho_{\max}}$ will be used to determine the interval of POI.

Thirdly, if n is replaced by $n/2$, we can obtain:

$$\frac{4\sqrt{2}}{\sqrt{n}} > \frac{\sqrt{28}}{\sqrt{n}} \tag{13}$$

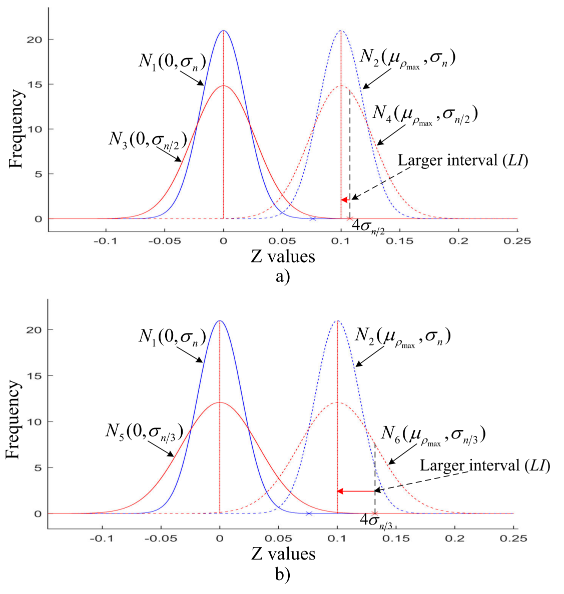


FIGURE 1. **The probability density of Z values on different numbers of used power traces: a) n and $n/2$; b) n and $n/3$.**

Consequently, $4\sigma_{n/2} > \rho_{\max}$. In this context, $\rho_{\max}$ is small ($\rho_{\max} < 0.2$), therefore $\mu_{\rho_{\max}} \approx \rho_{\max}$. It means that the right tail of distribution $N_3(0, \sigma_{n/2})$ is larger than $\mu_{\rho_{\max}}$. This larger interval (LI) can be calculated as $LI = 4\sigma_{\rho_0} - \mu_{\rho_{\max}}$, where σ_{ρ_0} denotes standard deviation of zero mean normal distribution in Fig. 1. The same calculations can be done for $n/3$. As a result, if we take the LI containing the samples which have the highest correlation, we can take the correct samples $t(\tau_{kc,tc})$. In other words, we can take the POI based on the right tail of $N_3(0, \sigma_{n/2})$, $N_5(0, \sigma_{n/3})$.

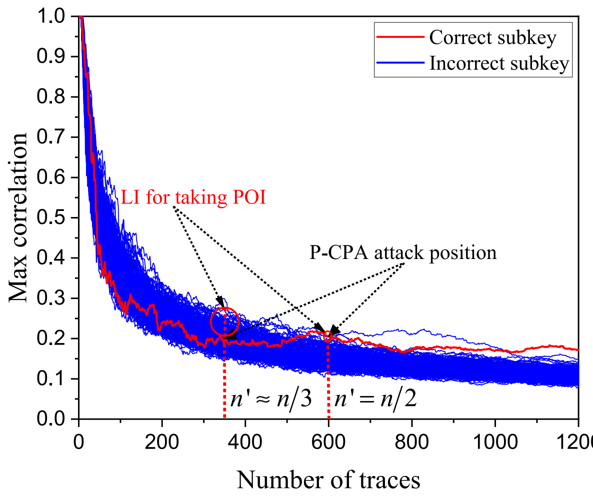


FIGURE 2. **The remarkable positions to perform P-CPA.**

Let's consider an intuitive example to illustrate our observation about the correlation. We perform an online 2-order CPA on the first 1200 power traces of the public ASCAD database. This is a software masked AES implementation obtained from an ATMega8515 micro-controller [18]. Then, we take the maximum correlation of each hypothesis key. In this context, we assume that the secret key is known and illustrated as a red line in Fig. 2. Two remarkable positions in which the numbers of traces are $\frac{n}{2}$ and $\frac{n}{3}$ are determined, respectively. In addition, the diameter of the red circle is used to illustrate the LI for taking POI. It is clear that at the position $n' = \frac{n}{2}$, the red circle is very small, the maximum correlation of the correct key is higher than almost all other hypothesis keys. It means that, if we take the top-down α POIs which have the highest correlation at $n' = \frac{n}{2}$, we can take the correct samples $t(\tau_{kc,tc})$. In contrast, at the position $n' = \frac{n}{3}$, the red circle is larger and the correlation is very low compared with other hypothesis keys. Therefore, a larger number of POIs must be taken to reveal the correct samples $t(\tau_{kc,tc})$. It is noted that the correlation matrix has the size of $key \times l$. Therefore, taking the sample $t(\tau_{kc,tc})$ when $n' = \frac{n}{3}$ is very challenging.

With α selected POIs where α is very small in comparison to l, the CPA computation complexity can be reduced proportionally with $\frac{l}{\alpha}$. We complete our first proposal by Algorithm 1. Suppose that we employ it to perform an auto-CPA on a set of $n \times l$ power traces corresponding to set of plain-texts which has size of $n \times 16$, we chose $n' = 1/2n$ and $\alpha = 100$. Algorithm 1 is divided into two phases. Firstly, we take a subset of power trace which has size of $n' \times l$ (Steps 2, 3). Then, we perform the standard CPA using the function *StandardCPA*() on this subset of power traces. As a result, the matrix correlation $R_{key,l}$ is taken and it is also the end of phase 1 (Step 5). Next, we take a top-down 100 samples corresponding to first 100 highest correlation values of matrix $R_{key,l}$ (Steps 8, 9, 10). After that, we take the second subset of power traces with the size of $n \times 100$ and the corresponding subset of plain-texts (Steps 12, 13). Finally, the phase 2 is performed (Step 15). The standard CPA is implemented again on the newest subset of power traces and plaintexts. The output of Algorithm 1 is the correct byte of the secret key.

B. PARTIAL CORRELATION POWER ANALYSIS BASED ON POWER TRACE BIASING

As indicated previously, taking the sample $t(\tau_{kc,tc})$ when $n' = \frac{n}{3}$ is very challenging. Therefore, we propose another method based on the power trace biasing technique.

For each sample $t\ (\tau)$, the operation on all power traces is usually the same. Therefore, the variance of the operation-dependent power consumption $\sigma^2(t_o(\tau)) = 0$. Consequently, the SNR value in (4) can be further simplified as:

$$SNR = \frac{\sigma^2(t_d(\tau))}{\sigma^2(t_{noise}(\tau))} \tag{14}$$

In the conventional CPA, it is clear that if the plaintexts are chosen randomly, the 8-bit intermediate values are uniformly distributed. In addition, each bit of the intermediate value is independent with the other bits and the probability of each bit is 0.5. Therefore, HW follows a binomial distribution. In practice, the probability of each HW is distributed as

Algorithm 1 Auto-CPA Based on Partial Correlation Power Analysis: P-CPA

```
    Input: $trace^{n \times l}$, $plaintext^{n \times 16}$, $n' = 1/2n$, $\alpha = 100$
    Output: $CorrectKey_{Byte}$
1:  for Byte from 1 to 16 do
2:      $Plt_0 = plaintext^{n' \times 16}$
3:      $Tr_0 = trace^{n' \times l}$
4:      for key from 0 to 255 do                         ▷ Phase 1
5:          $R_{key,l} \leftarrow StandardCPA(Plt_0, Tr_0)$
6:      end for
7:      while $\alpha \leq 100$ do                       ▷ Taking POIs
8:          $\alpha = \alpha + 1$
9:          $\bar{l}_\alpha = l_i \leftarrow argmax(R_{key,l})$
10:         $l_i = 0$
11:     end while
12:     $Plt_1 = plaintext^{n \times 16}$
13:     $Tr_1 = trace^{n \times \bar{l}}$                ▷ $\bar{l}$ has size of $(1 \times \alpha)$
14:     for key from 0 to 255 do                         ▷ Phase 2
15:         $R_{key,\bar{l}} \leftarrow StandardCPA(Plt_1, Tr_1)$
16:     end for
17:     $CorrectKey_{Byte} = key \leftarrow argmax(R_{key,\bar{l}})$
18: end for
```

TABLE 2. Probability distribution of the *HW* of a uniformly distributed 8-bit value.

HW	0	1	2	3	4	5	6	7	8
Probability	$\frac{1}{256}$	$\frac{8}{256}$	$\frac{28}{256}$	$\frac{56}{256}$	$\frac{70}{256}$	$\frac{56}{256}$	$\frac{28}{256}$	$\frac{8}{256}$	$\frac{1}{256}$

expressed in Table 2. It is clear that the HW with big deviation appears with low probability. In this case, we assume that $Var(t_d(\tau)) \propto Var(HW)$. It means that we can take the high values of the variance $\sigma^2(t_o(\tau))$ if the adaptive plaintexts are chosen. Based on this observation, we proposed a method for selecting the adaptive plain-texts which give the high SNR values. However, unlike the previous works [16], our novelty is that we take the plain-text from a given set of plain-texts. If we take n' plain-texts which have the intermediate values corresponding to $HW = \{0, 1, 2, 6, 7, 8\}$, n' is approximately 28.9% of n. In other words, if we calculate the CPA on $n' = 1/3n$ for taking POI as the phase 1 of P-CPA, the probability of taking the correct sample $t(\tau_{kc,tc})$ is very high.

The previous intuitive example is used again with further experiments. Fig. 3.(a) illustrates the correlation at the first 350 power traces of a given set of power traces (one third of previous example). Meanwhile, Fig. 3.b presents the correlation of 350 power traces selected by biasing technique. It clearly shows that the correlation of correct key in Fig. 3.(b) is higher and more consistent than that in Fig. 3.(a). Therefore, it is easy to take the number of POIs which consist of the correct sample $t(\tau_{kc,tc})$. Our second proposed method is presented by Algorithm 2.

Similar to Algorithm 1, Algorithm 2 is divided into two phases. However, Algorithm 2 is different with Algorithm 1 in the process of the phase 1 and the value of $\alpha = 250$. We also do further steps to take out $n' \approx \frac{n}{3}$ power traces based on the biasing technique (Step 5, 7, 8). The rest of Algorithm 2 is processed the same as Algorithm 1.

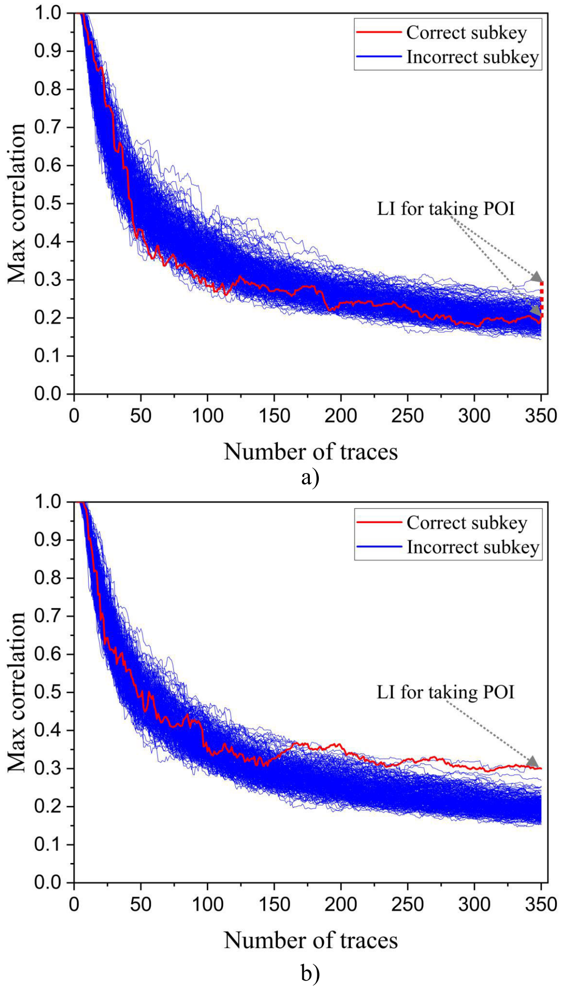


FIGURE 3. The correlation of the correct key for two CPA methods: (a) Standard CPA; (b) CPA with power trace biasing technique.

An important practical aspect is the number of POIs that have to be taken in order to perform a successful attack. However, calculating the exact number of POIs that contain the correct sample is very challenging in practice because of several reasons. Firstly, $\rho_{\max}$ is a random variable and it is difficult to find the certain values of $\rho_{\max}$. Secondly, it is clear that LI have different lengths corresponding to different values of n. Finally, there are so many correct samples that can be used to reveal the correct keys. Indeed, the authors in [26] have indicated that in their experiments, the intermediate result is used in several instructions. This is very typical for software implementations. Each time the micro-controller performs an operation that involves the attacked intermediate result, there is at least one peak in the matrix R. Due to the above mentioned reasons, we focus only on finding the number of traces for the POI extractor. The theory of

Algorithm 2 Auto-CPA Based on Power Trace Biasing Based Partial Correlation Power Analysis: BP-CPA

```
    Input: $trace^{n\times l}$, $plaintext^{n\times 16}$, $n' = 0$, $\alpha = 250$
    Output: $CorrectKey_{Byte}$
 1: $Plt_0 = plaintext^{n\times 16}$
 2: for Byte from 1 to 16 do
 3:     for $key$ from 0 to 255 do                          ▷ Phase 1
 4:         for i from 1 to $n$ do
 5:             $h \leftarrow HW(SBOX((plaintext_{i,16}, key))$
 6:             if $h = 0, 1, 2, 6, 7, 8$ then
 7:                 $n' = n' + 1$
 8:                 $trace_{n',l} = trace_{i,l}$
 9:             end if
10:         end for
11:         $Tr_0 = trace^{n'\times l}$
12:         $R_{key,l} \leftarrow StandardCPA(Plt_0, Tr_0)$
13:     end for
14:     while $\alpha \leq 250$ do                          ▷ Taking POIs
15:         $\alpha = \alpha + 1$
16:         $\bar{l}_\alpha = l_i \leftarrow argmax(R_{key,l})$
17:         $l_i = 0$
18:     end while
19:     $Plt_1 = plaintext^{n\times 16}$
20:     $Tr_1 = trace^{n\times \bar{l}}$
21:     for $key$ from 0 to 255 do                          ▷ Phase 2
22:         $R_{key,\bar{l}} \leftarrow StandardCPA(Plt_1, Tr_1)$
23:     end for
24:     $CorrectKey_{Byte} = key \leftarrow argmax(R_{(key,\bar{l})})$
25: end for
```

calculating exactly the number of POIs is out of the scope of this work. However, based on several practical attacks and simulations, we have determined the values of POI = 100 and POI = 250 for $n/2$ and $n/3$, respectively, can be enough to reduce n for POI extraction with a small number of POIs. This reasonable choice will be proven by our experiments in the next section.

IV. EXPERIMENTAL RESULTS

To prove the efficiency of our proposed on different types of SCA countermeasures, the data of two platforms including RISC-V processor and public ASCAD database were chosen. All experiments were performed with MATLAB software on a personal computer with Intel Core i5-9500 CPU, DDR4 24GB memory. In our experiments, the average of the success rate and the computation time are used as the metrics to evaluate the efficiency of the proposed methods.

A. MASKING

Our first experiment is performed on the ASCAD database containing 60,000 traces in which each trace consists of 700 samples. The protected AES core was implemented by the masking countermeasure with a different masking value for each bye of the secret key. This technique leads to

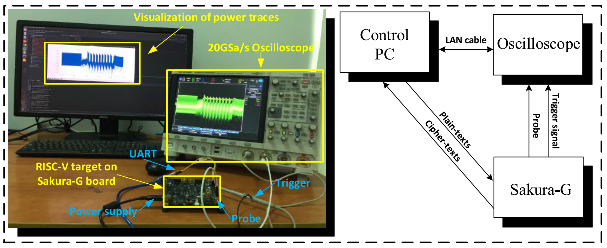


FIGURE 4. Test platform: RISC-V power traces acquisition on Sakura-G board.

TABLE 3. The parameter of traces for our experiments on noise added RISC-V power traces.

$\sigma(noise)$	0.004	0.006	0.008	0.01
n(Std-CPA)	400	800	1000	1600
n'(P-CPA)	200	400	500	800
n'(BP-CPA)	≈ 115.6	≈ 231.2	≈ 289	≈ 462.4

an increased computational complexity. In our experiments, we perform the standard 2-order CPA (Std-CPA) to determine the number of power traces needed for the attack. P-CPA and BP-CPA techniques are then performed to investigate the efficiency in terms of the computation time. In the pre-processing state of Std-CPA, we calculate a pre-processed trace that contains all values $|l_a - l_b| \forall l_a, l_b \in l$. As a result, the new dataset with 244,650 samples on each power trace is collected. Next, the Std-CPA is performed on the pre-processing traces. As depicted in Fig. 5(a), the Std-CPA was implemented successfully with 1200 ASCAD traces. The highest correlation is 0.16605 and the execution time of Std-CPA is 1188.4 seconds. Then, these traces are used to implement P-CPA and BP-CPA. Fig. 5.(b) shows that P-CPA can reveal the correct key on power traces that used by Std-CPA. Especially, P-CPA allows to take exactly the same sample which has the highest correlation as Std-CPA ($\rho = 0.16605$). For BP-CPA, it is clear that this method works better than P-CPA and Std-CPA. BP-CPA only used 350 power traces for phase 1 (nearly 1/3 of total power traces needed in Std-CPA). In addition, the POIs of BP-CPA are taken accurately, the position of the correct sample ($X = 3$) is lower than P-CPA ($X = 13$). It means that the power trace biasing technique makes the values of the correct key higher and leads to the higher probability of the successful attack. Since the partial correlation is used, the computation time values of P-CPA and BP-CPA decline considerably from $1,188.4$ seconds to 583.54 seconds and 446.94 seconds, respectively. These results have clarified the efficiency of our proposed methods on masking protected devices.

B. HIDING

This experiment aims to evaluate the proposed technique in different contexts of hiding countermeasures. Therefore, several noise levels were built by changing the

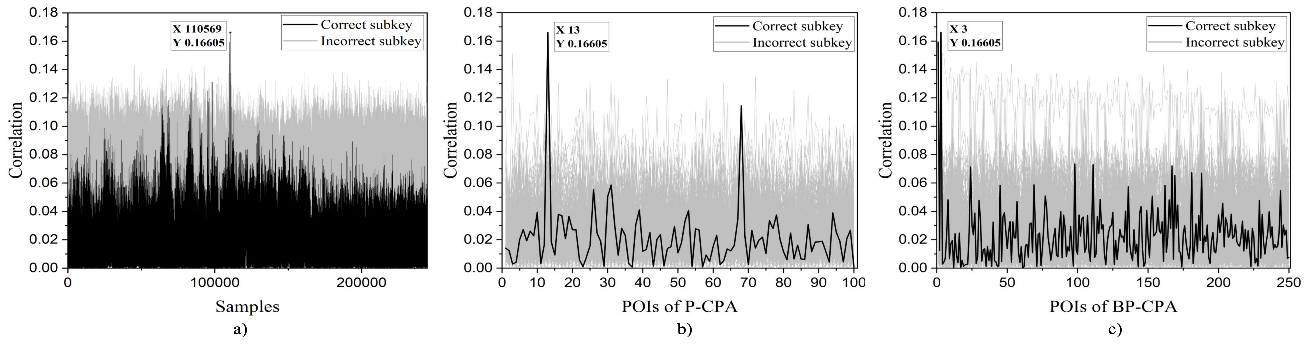


FIGURE 5. Attack results of standard CPA, P-CPA and BP-CPA methods on masking countermeasure: (a) Standard CPA; (b) P-CPA; (c) BP-CPA.

standard deviation of Gaussian noise on clean power traces. We have performed the power traces acquisition for the unprotected AES-128 on Sakura-G board. To obtain the power trace, we use an oscilloscope at the sampling rate of 250 MS/s. The target acquisition contains a Murax 32-bit RISC-V MCU operating at the clock frequency of 48 MHz, as illustrated in Fig. 4. We have collected 10,000 power traces, each power trace contains 9,919 samples. We then add the Gaussian noise centered in zero with several values of standard deviation to simulate different levels of hiding countermeasure. As same as the previous experiment, we perform CPA attacks by using three techniques of Std-CPA, P-CPA, BP-CPA to compare the efficiency in the computation time. Table 3 presents the details of parameters on each experiment in this work. We mount the attacks and repeat these attacks in each case of noise 100 times, the power traces of each attack are randomly taken from 10,000. The results of the experiments are then averaged. As depicted in Fig. 6.(a), the computation times of three techniques on each level of noise are different. It is clear that the average computation time of Std-CPA is highest on all experiments. The results indicate that the computation time values of P-CPA and BP-CPA are the 2^{nd} lowest and the lowest ones, respectively. These results demonstrate that BP-CPA is the most effective in terms of the computation time and it can reduce approximately two times compared with Std-CPA. However, it is worth noting that the time consumption values of P-CPA and BP-CPA methods when $\sigma = 0.004$ are nearly the same, and the reduction of the computation time is 1.5 times. This is an unexpected result because the number of traces for the attack is small and phase 1 of BP-CPA needs some extra computation time to filter the high SNR power traces. Comparing the reliability of our proposed techniques, the averages of success attack is taken. As illustrated in Fig. 6.(b), P-CPA achieves the highest success rate and the success rate of BP-CPA is lower than Std-CPA in cases of $\sigma = 0.006$ and $\sigma = 0.008$. However, in general, the results of the BP-CPA success rate are acceptable.

C. COMBINED HIDING AND MASKING

In this experiment, we consider the combination of hiding and masking techniques. To simulate this scenario, we simply add

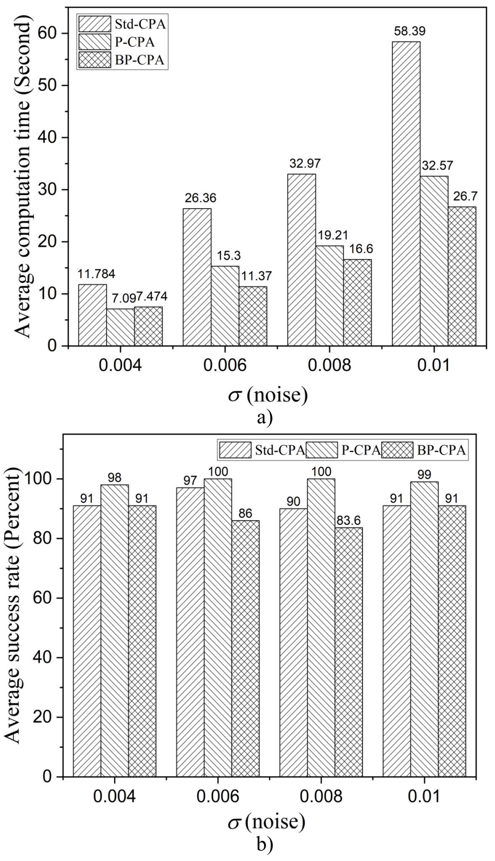


FIGURE 6. Average computation time and success rate of one subkey on noise added RISC-V power traces. (a) Average computation time; (b) Average success rate.

the Gaussian noise with different levels to ASCAD database. In TCHES-2019, Benjamin *et al.* have demonstrated that DDLA can break the masked cryptographic devices without any advance knowledge about the masking implementation [21]. In contrast, a 2-order CPA attack needs to be pre-processed and performed in a squared computation complexity compared to the 1-order CPA. However, the standard 2-order CPA can overcome the noise-generation-based hiding countermeasure. Therefore, in this experiment, we mount the attacks to compare the possibility of DDLA and CPA on

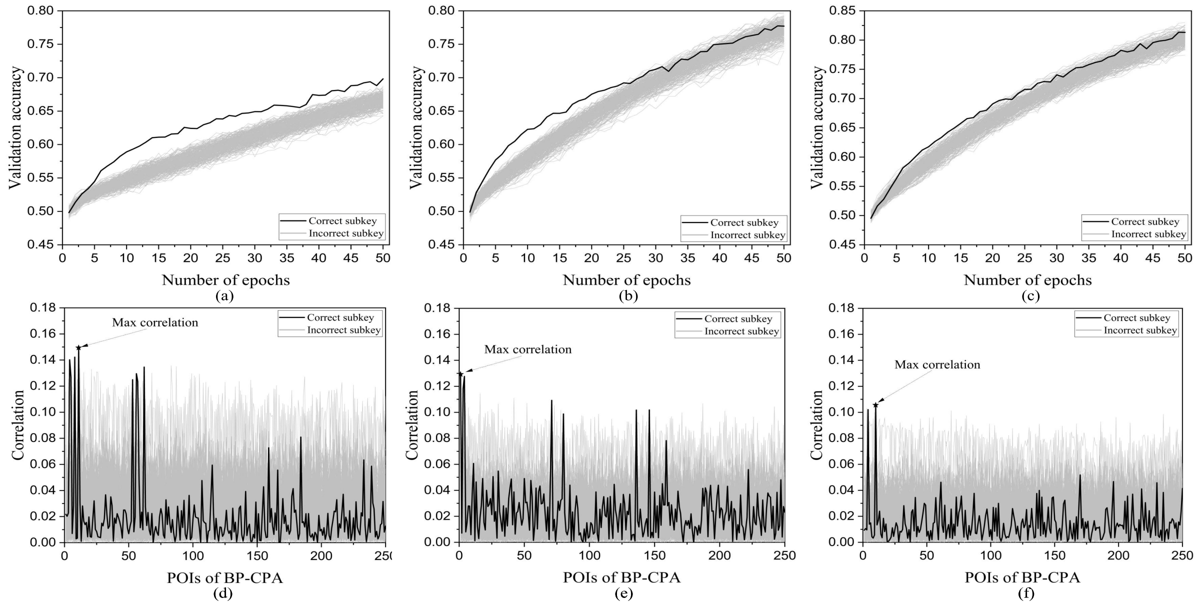


FIGURE 7. The experimental results on differential levels of noise added ASCAD database. (Left column) $\sigma = 0.5$; (Center column) $\sigma = 1.0$; (Right column) $\sigma = 1.5$; (a), (b), (c) DDLA attack; (d), (e), (f) BP-CPA attack.

a combined countermeasure. We reconstruct the same model of the MLP network on TCHES-2019 with 30,000 power traces as the original work [21]. In terms of CPA, we use a maximum 2,700 and minimum 1,200 power traces for the highest and the lowest levels of noise, respectively. It is noted that the original traces in each attack are the same, but the Gaussian noise is re-initialized and added on each implementation. Table 4 shows the parameter of our experiments in detail. For the convenience, we choose the most effective proposed technique BP-CPA for this experiment. The attack results are shown in Fig. 7. As shown clearly in Fig. 7.(a, b, c), it is difficult to discriminate the correct subkey by DDLA when $\sigma = 1.0$ and it can not distinguish the correct subkey in the case of $\sigma = 1.5$. In contrast, our proposed technique can reveal the correct subkey in all cases. The black lines in Fig. 7.(d, e, f) present the correlation of the correct subkey. It is worth noting that the correct samples which have the highest correlation are taken at the beginning of the POI axis. It means that the power trace biasing technique makes the correlation values of candidate samples higher than the rest. In addition, our algorithm takes the POI following the top-down strategy. Therefore, we can reduce the number of POIs to the value of less than 250. More interestingly, the maximum correlation value decreases when the number of traces increases as explained in (12). Despite having the highest values, the black lines are not clearly distinguished from the rest (gray lines). This is because we take the minimum number of traces for successful attacks. These experimental results have indicated that the proposed method could outperform the non-profiled DL based attack (DDLA) on hiding-masking protected devices.

TABLE 4. The parameter of traces for the experiments on the noise added ASCAD database.

$\sigma(noise)$	0.5	1.0	1.5
n(Std-CPA)	1500	2200	2700
n'(BP-CPA)	≈ 433.5	≈ 635.8	≈ 780.3
TCHES-2019 [21]	30,000	30,000	30,000

After proving the efficiency of our proposed method, we decide to perform further experiments to evaluate the execution time of DDLA, Std-CPA and our proposed techniques. To achieve the reliable results, we repeat 50 times the experiment of CPA in Section IV-C. The computation time is then averaged and presented in Fig. 8.(a). The DDLA columns show the execution time of the DDLA technique. They are almost the same because we fixed the number of traces for all different levels of noise. The Std-CPA columns illustrate the computation time of Std-CPA. We increase the number of traces corresponding to the increase of the deviation of noise. As a result, the execution time of Std-CPA increases significantly and it reaches the highest value when $\sigma = 1.5$. In addition, the execution time of Std-CPA is higher than DDLA in all cases. These results indicate that the drawback of Std-CPA is time-consuming. Fortunately, the goal of this work is to resolve the limitation of Std-CPA. As expected, the BP-CPA columns present the time consumption of BP-CPA which is lower than those of both DDLA and Std-CPA. Especially, the execution time of BP-CPA reduces approximately by 2.6 times compared to Std-CPA in all cases.

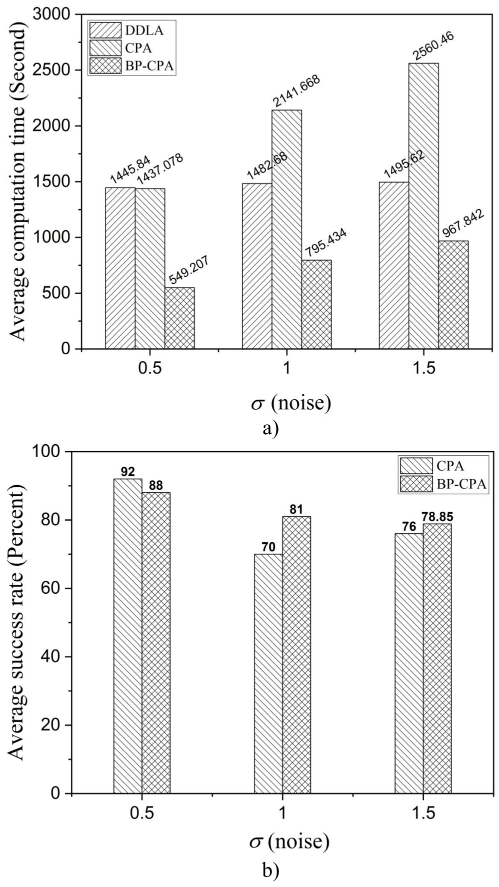


FIGURE 8. Average of computation time and success rate on different levels of Gaussian noise added ASCAD: (a) Average of computation time; (b) Average of success rate.

The results also indicate that our proposed technique outperforms DDLA even though DDLA uses a huge number of power traces (967.842 seconds compared to 1495.62 seconds in the case of $\sigma = 1.5$).

Consider the reliability, the comparison of the success rate between Std-CPA and BP-CPA is shown in Fig. 8.(b). In terms of the low noise level, the success rate of our proposal is slightly lower than Std-CPA. This is the limitation of our proposed technique. Our proposal uses power trace biasing to increase the standard deviation of $t_d(\tau)$ on $n' = n/3$ power traces. However, the attack phase is still based on standard CPA. Therefore, the success rate of the proposed technique is less than or the same as Std-CPA in such cases. In contrast, our proposal has better results than Std-CPA in the remaining cases. From (14), it clearly shows that the higher the noise, the smaller the SNR value. Therefore, Std-CPA needs to use more power traces in order to discriminate the correct samples as obtained from (6). By using the POI extractor, our proposed technique has eliminated most of the noise affected samples that are not related to the correct key or the operation. Consequently, the proposed technique increases the probability of the successful attack in comparison to Std-CPA: 81% compared to 70% in the case of $\sigma = 1$ and 78.85% compared to 76% in the case of $\sigma = 1.5$.

V. CONCLUSION

This paper has proposed two novel POI extractors based on the power trace biasing technique and the correlation distribution to reduce the computation time. Our first proposal (P-CPA) exploits the correlation distribution. On the other hand, the second proposal (BP-CPA) is a new power trace biasing based technique which requires only 28.9% of power traces. Using proposed POI extractors, we presented two auto-CPA algorithms that perform automatically the POI selection, together with key recovering. The effectiveness of our algorithms was proved by conducting various experiments based on different platforms, such as the ASCAD database and RISC-V processor based dataset. The results indicate that when compared with the standard CPA, by using only a half of the power traces needed for taking the POIs, our proposed method not only decreases the execution time by approximately 2.0 times but also enhances the success rate of attack. In addition, based on power trace biasing technique, our proposed technique reduces the execution time to approximately 2.6 times compared to the standard CPA. However, the success rate of BP-CPA is lower than the standard CPA. We assume that this is the trade-off between the number of traces needed for taking POI and the attack success. In the future work, we plan to combine our proposal with other preprocessing techniques to further enhance the performance of the SCA evaluation.

REFERENCES

[1] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Advances in Cryptology*, M. Wiener, Ed. Berlin, Heidelberg: Springer, 1999, pp. 388–397.

[2] S. Bhunia, M. S. Hsiao, M. Banga, and S. Narasimhan, "Hardware Trojan attacks: Threat analysis and countermeasures," *Proc. IEEE*, vol. 102, no. 8, pp. 1229–1247, Aug. 2014.

[3] L. Liu, T. Wang, X. Wang, and T. He, "A method of implanting combinational hardware trojan based on evolvable hardware," *Comput. Electr. Eng.*, vol. 93, Jul. 2021, Art. no. 107229. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0045790621002184

[4] J.-S. Coron, "Resistance against differential power analysis for elliptic curve cryptosystems," in *Cryptographic Hardware and Embedded Systems*, Ç. K. Koç and C. Paar, Eds. Berlin, Heidelberg: Springer, 1999, pp. 292–302.

[5] J. D. Golić and C. Tymen, "Multiplicative masking and power analysis of aes," in *Cryptographic Hardware and Embedded Systems*, B. S. Kaliski, Ç. K. Koç, and C. Paar, Eds. Berlin, Heidelberg: Springer, 2003, pp. 198–212.

[6] E. Oswald, S. Mangard, C. Herbst, and S. Tillich, "Practical second-order DPA attacks for masked smart card implementations of block ciphers," in *Topics in Cryptology*, D. Pointcheval, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 192–207.

[7] N. Kamoun, L. Bossuet, and A. Ghazel, "Correlated power noise generator as a low cost DPA countermeasures to secure hardware AES cipher," in *Proc. 3rd Int. Conf. Signals, Circuits Syst. (SCS)*, Nov. 2009, pp. 1–6.

[8] S. Yang, W. Wolf, N. Vijaykrishnan, D. N. Serpanos, and Y. Xie, "Power attack resistant cryptosystem design: A dynamic voltage and frequency switching approach," in *Design, Automation and Test in Europe*, vol. 3, Munich, Germany: IEEE, Mar. 2005, pp. 64–69.

[9] K. Tiri and I. Verbauwhede, "Secure logic synthesis," in *Field Programmable Logic and Application*, J. Becker, M. Platzner, and S. Vernalde, Eds. Leuven, Belgium: Springer, Aug. 2004, pp. 1052–1056.

[10] B.-A. Dao, T.-T. Hoang, A.-T. Le, A. Tsukamoto, K. Suzaki, and C.-K. Pham, "Exploiting the back-gate biasing technique as a countermeasure against power analysis attacks," *IEEE Access*, vol. 9, pp. 24768–24786, 2021.

[11] S. Mangard, "Hardware countermeasures against dpa – a statistical analysis of their effectiveness," in *Topics in Cryptology*, T. Okamoto, Ed. Berlin, Heidelberg: Springer, 2004, pp. 222–235.

[12] S. Chari, J. R. Rao, and P. Rohatgi, "Template attacks," in *Cryptographic Hardware and Embedded Systems*, B. S. Kaliski, Ç. K. Koç, and C. Paar, Eds. Berlin, Heidelberg: Springer, 2003, pp. 13–28.

[13] B. Gierlichs, K. Lemke-Rust, and C. Paar, "Templates vs. stochastic methods," in *Cryptographic Hardware and Embedded Systems*, L. Goubin and M. Matsui, Eds. Berlin, Heidelberg: Springer, 2006, pp. 15–29.

[14] U. Rioja, L. Batina, J. L. Flores, and I. Armendariz, "Auto-tune POIs: Estimation of distribution algorithms for efficient side-channel analysis," *Comput. Netw.*, vol. 198, Oct. 2021, Art. no. 108405.

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

[16] W. Hu, L. Wu, A. Wang, X. Xie, Z. Zhu, and S. Luo, "Adaptive chosen-plaintext correlation power analysis," in *Proc. 10th Int. Conf. Comput. Intell. Secur.*, Nov. 2014, pp. 494–498.

[17] C. Ou, S.-K. Lam, D. Sun, X. Zhou, K. Qiao, and Q. Wang, "SNR-centric power trace extractors for side-channel attacks," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 40, no. 4, pp. 620–632, Apr. 2021.

[18] R. Benadjila, E. Prouff, R. Strullu, E. Cagli, and C. Dumas, "Deep learning for side-channel analysis and introduction to ASCAD database," *J. Cryptograph. Eng.*, vol. 10, no. 2, pp. 163–188, Jun. 2020.

[19] T. Kubota, K. Yoshida, M. Shiozaki, and T. Fujino, "Deep learning side-channel attack against hardware implementations of AES," *Microprocessors Microsyst.*, vol. 87, Nov. 2021, Art. no. 103383. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0141933120305408

[20] B. Hettwer, S. Gehrer, and T. Güneysu, "Applications of machine learning techniques in side-channel attacks: A survey," *J. Cryptograph. Eng.*, vol. 10, no. 2, pp. 135–162, Jun. 2020, doi: 10.1007/s13389-019-00212-8.

[21] B. Timon, "Non-profiled deep learning-based side-channel attacks with sensitivity analysis," *IACR Trans. Cryptograph. Hardw. Embedded Syst.*, vol. 2019, no. 2, pp. 107–131, Feb. 2019. [Online]. Available: https://tches.iacr.org/index.php/TCHES/article/view/7387

[22] D. Kwon, H. Kim, and S. Hong, "Non-profiled deep learning-based side-channel preprocessing with autoencoders," *IEEE Access*, vol. 9, pp. 57692–57703, 2021.

[23] N.-T. Do, V.-P. Hoang, and V.-S. Doan, "Performance analysis of non-profiled side channel attacks based on convolutional neural networks," in *Proc. IEEE Asia Pacific Conf. Circuits Syst. (APCCAS)*, Dec. 2020, pp. 66–69.

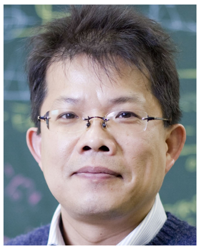
[24] *Stat. Characteristics Power Traces*, Springer, Boston, MA, USA, 2007, pp. 61–99, doi: 10.1007/978-0-387-38162-6_4.

[25] E. Brier, C. Clavier, and F. Olivier, "Correlation power analysis with a leakage model," in *Cryptographic Hardware and Embedded Systems*, M. Joye and J.-J. Quisquater, Eds. Berlin, Heidelberg: Springer, 2004, pp. 16–29.

[26] *Differential Power Analysis*, Springer, Boston, MA, USA, 2007, pp. 119–165, doi: 10.1007/978-0-387-38162-6_6.

NGOC-TUAN DO received the master's degree in electronic engineering from Le Quy Don Technical University, Hanoi, Vietnam, in 2017, where he is currently pursuing the Ph.D. degree in electronic engineering with Le Quy Don Technical University. His research interest includes hardware security and integrated circuit design.

VAN-PHUC HOANG (Member, IEEE) received the Ph.D. degree in electronic engineering from The University of Electro-Communications, Tokyo, Japan, in 2012. He has worked as a Postdoctoral Researcher and a Visiting Scholar at The University of Electro-Communications; Telecom Paris, France; and the University of Strathclyde, Glasgow, U.K., during the period of 2012–2018. He is currently working as an Associate Professor and the Director of the Institute of System Integration, Le Quy Don Technical University, Hanoi, Vietnam. He is also serving as the Vice Chair in International Affairs & Conferences, Radio-Electronics Association of Vietnam (REV). His research interests include hardware security, digital circuits and systems, embedded systems for the Internet of Things, and VLSI architecture for digital signal processing. He was the PI of 02 NAFOSTED funded projects and one World Bank funded project in hardware security. He was the Technical Program Chair of several IEEE international conferences, such as ICDV 2017, MCSoC 2018, SigTelCom 2019, APCCAS 2020, and ATC 2020.



CONG-KHA PHAM (Member, IEEE) received the B.S., M.S., and Ph.D. degrees in electronics engineering from Sophia University, Tokyo, Japan. He is currently a Professor with the Department of Information and Network Engineering, The University of Electro-Communications (UEC), Tokyo. His research interests include hardware security, open source processors, and the design of analog and digital systems using integrated circuits.