Hindawi
Security and Communication Networks
Volume 2021, Article ID 6614702, 13 pages
https://doi.org/10.1155/2021/6614702

Research Article

Side-Channel Leakage Detection with One-Way Analysis of Variance

Wei Yang and Anni Jia

School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China

Correspondence should be addressed to Wei Yang; generalyzy@njust.edu.cn

Received 1 January 2021; Revised 7 February 2021; Accepted 25 February 2021; Published 5 March 2021

Academic Editor: Liguo Zhang

Side-channel analysis (SCA) is usually used for security evaluation to test the side-channel vulnerability of a cryptographic device. However, in practice, an analyser may need to cope with enormous amounts of side-channel measurement data to extract valuable information for SCA. Under the circumstances, side-channel leakage detection can be used to identify leakage points which contain secret information and therefore improve the efficiency of security assessment. This investigation proposes a new black-box leakage detection approach on the basis of the one-way analysis of variance (ANOVA). In accordance with the relevance between leakage points and inputs of a cryptographic algorithm, the proposed method divides side-channel samples into multiple classes and tests the difference among these classes by taking advantage of the one-way ANOVA. Afterwards, leakage points and nonleakage points can be distinguished by determining whether the null hypothesis is accepted. Further, we extend our proposed method to multichannel leakage detection. In particular, a new SCA attack with a F-statistic-based distinguisher is capable of developing if the input of the leakage detection approach is replaced by a sensitive intermediate variable. Practical experiments show the effectiveness of the proposed methods.

1. Introduction

Side-channel analysis (SCA) applies the physical leakage signals (e.g., sound, power consumption, electromagnetic radiation, and the like) of a running cryptographic device for retrieving the secret information [1–4]. Due to the efficiency and easy achievement, SCA has been a serious threat to cryptographic devices, but at the same time, SCA can also be used to evaluate the side-cannel resistance of these devices. In practice, SCA usually needs numerous leakage traces for security evaluation. That may lead to a fairly high computation complexity and significantly reduce the efficiency of the assessment. Therefore, side-channel leakage detection is proposed to reduce the computation complexity of security evaluation and improve the efficiencies of SCA attacks [5, 6].

Points in a side-channel signal can be grouped into two sets: leakage points and nonleakage points [6]. Leakage points contain sensitive information related to the key used in a crypto device and can be utilized by SCA. By contrast, nonleakage points include useless information for SCA. The task of leakage detection is to find the leakage points in side-channel measurements. Generally, the existing leakage detection methods view leakage points as the outliers in the measurement data and use differing statistics to winnow them. A simple way is to exploit the SCA distinguisher output as the statistic; any point with a high score is assigned to the leakage point set [7]. Additionally, a major category of leakage detection is based on the hypothesis test and identifies the “outliers” through computing a test statistic and a significance level. For example, the typical Test Vector Leakage Assessment (TVLA) methodology searches the leakage points by exploiting *Welch*’s t-test. The TVLA checks the t-statistic over the same time instants. If the value of the statistic is greater than the threshold determined by a prespecified significance level, the corresponding point will be regarded as a leakage point. Some similar leakage detection approaches are also proposed, such as leakage detection based on mutual information [8], the correlation-based ρ-test [9], the paired t-test [10], leakage detection with χ^2-test [11], and the rest. Besides, a method based on normalized interclass variance and a new approach based on the constant parameter channel model [12] are investigated as well.

Nevertheless, some of the aforementioned methods need numerous traces to highlight the difference between leakage points and nonleakage points [11, 13], some methods need the knowledge of leakage models [7, 13], some require the low-noise measurements [11] or carefully chosen inputs [5, 9, 10], some need complicated calculation [8, 12], and so forth.

In sight of the limitations of the previous work, this paper proposes a black-box leakage detection approach based on the one-way analysis of variance (ANOVA). The main idea of the proposed method is that side-channel leakage samples at one point depend on the corresponding inputs of a cryptographic algorithm. That is, the same input yields the same leakage, while two differing inputs lead to two differing leakages. Consequently, according to the relevance between the leakage points and inputs of the algorithm, side-channel samples can be divided into multiple groups. These groups corresponding to a leakage point are supposed to have a statically significant difference because that variations of the inputs and the corresponding side-channel leakages are synchronous. On the contrary, the groups corresponding to a nonleakage point should have a statistically insignificant difference. Then, leakage points can be detected by the one-way ANOVA which is used for testing the difference of multiple groups.

The proposed method relies only on the inputs (i.e., plaintext or ciphertext) of the cryptographic algorithm and the corresponding side-channel samples. Hence, it is unnecessary to acquire any in-depth knowledge of the cryptographic algorithm implementation and the leakage model of a device. Moreover, it has no special requirements for the inputs and measurements and can be performed rapidly.

Further, we develop a new SCA distinguisher by changing the input of the proposed leakage detection approach. The effectiveness of the SCA attack using the distinguisher is verified by two practical experiments.

The rest of the paper consists of four sections, including the preliminaries (Section 2), the proposed method and its extensions (Section 3), the experimental results and analysis (Section 4), and the conclusion (Section 5).

2. Preliminaries

2.1. Side-Channel Leakage Detection. Side-channel leakage detection is used to detect the information leakage in a specific side-channel [5]. Unlike SCA attacks, leakage detection needs not to distinguish the correct key used in a cryptographic implementation from the other incorrect key guesses. Its task is to find key-dependent sample points (i.e., leakage points) in a side-channel leakage trace. That is to say, the output of leakage detection is a set of points of interest, which are corresponding to the formative moments of leakages related to secret information. Leakage detection is an important part of the side-channel evaluation and provides a preliminary result for further analysis or advanced assessment.

2.2. Test Vector Leakage Assessment. Test Vector Leakage Assessment (TVLA) methodology is a classical and extensively used technology for side-channel leakage detection. The basic assumption of TVLA is that leakage points and nonleakage points belong to two normally distributed populations, respectively. Then, an analyser can distinguish leakage points from nonleakage points by using *Welch*'s t-test to compare the parameters of these two populations.

The TVLA methodology includes two methods, that is, specific t-test and nonspecific t-test [5, 13, 14]. For the specific t-test, the side-channel traces are divided into two sets in accordance with the value of a single bit of a targeted intermediate key-dependent variable. If the two trace sets are considered as two normally distributed populations, *Welch*'s t-test can be applied to detect side-channel information leakage by testing the difference of the two population means over each time instant in the traces. The t-statistic is shown as follows:

$$t = \frac{\left(\overline{T_A} - \overline{T_B}\right)}{\sqrt{\left(S_A^2/N_A\right) + \left(S_B^2/N_B\right)}}, \tag{1}$$

where T_A and T_B are the two trace sets, $\overline{T_A}$, $\overline{T_B}$, S_A, S_B, N_A, and S_B are the means, variances, and size of T_A and T_B, respectively. If the value of t-statistic at one time instant exceeds the threshold determined by the degrees of freedom and a prespecified significance level, the point will be categorized as a leakage point. Otherwise, the point will be categorized as a nonleakage point.

For the nonspecific t-test, it needs to acquire two groups of side-channel measurements with the same size to apply for the pairwise t-test. One set of traces corresponds to a fixed input, and the other corresponds to random inputs. If one point in time for which the value of t-statistic is beyond that threshold, the point will be selected as a leakage point. If the corresponding t-statistic is within the bounds determined by that threshold, the point will be marked as a nonleakage point.

2.3. Leakage Detection with χ^2-Test. The leakage detection with the χ^2-test can be regarded as a complement to TVLA [11]. The core idea of this approach is to apply the χ^2-test of independence to check the association between the input groups (i.e., plaintext or ciphertext) and the measured leakage points.

The χ^2-test of independence is also named the χ^2-test of association. It is a nonparametric test to determine whether two categorical variables are related or independent. The statistic for the χ^2-test of association is computed as

$$\chi^2 = \sum_{i=1}^{r} \sum_{j=1}^{s} \left[\frac{\left(V_{ij} - NP_{ij}\right)^2}{NP_{ij}} \right], \tag{2}$$

where r represents the number of the input groups (i.e., plaintext or ciphertext), s represents the number of the measured leakage points, N denotes the total number of the observations of two categorical variables, V_{ij} and P_{ij} denote the number and the frequency of concurrence of the observations of two categorical variables, respectively.

Similarly, if one point in time for which the value of χ^2-test exceeds the threshold determined by the degrees of freedom and a prespecified significance level, the point will be incorporated into the leakage points. Otherwise, the point will be incorporated into the nonleakage points.

2.4. One-Way Analysis of Variance. One-way analysis of variance (one-way ANOVA) can be seen as a generalization of *Welch*'s t-test. Where *Welch*'s t-test would be applied for comparison of group means of two normally distributed populations, one-way ANOVA is applied for the comparison of multiple group means [15, 16]. It assumes that samples in each group are independently drawn from a normally distributed population, and all populations have the same variance. The test statistic is

$$F = \frac{(N-1)SS_b}{[(r-1)SS_w]}, \tag{3}$$

where r represents the number of the input groups, N denotes the total number of the samples in all groups, SS_b denotes the sum of squares between groups, that is, the *between-group variation*, and SS_w denotes the residual variance within groups, that is, the *within-group variation*. If the value of F-statistic is beyond the threshold, it states that all r populations own the same mean value.

3. The Proposed Method

In this section, one-way ANOVA is applied for leakage detection, including monochannel and multichannel leakage detection. In addition, a novel SCA distinguisher is developed to serve for key recovery attack as well.

3.1. Side-Channel Leakage Detection with One-Way ANOVA. What the one-way ANOVA investigates is the influence of a numerical or categorical input variable on a numerical dependent variable. For a cryptographic implementation, the side-channel leakage samples depend on the inputs and the secret key of the crypto algorithm (i.e., plaintext or ciphetext). Since the key is usually changeless in a short period of time, the leakage data can be viewed as the numerical response data of a variable which relies on the inputs. Additionally, the possible values of the plaintext or ciphertext of a crypto algorithm are limited and can be categorized into multiple classes. Therefore, it is possible to detect leakage points by exploiting the one-way ANOVA.

Suppose that there are N side-channel leakage traces. They are collected from a running crypto device and correspond to N random plaintext (or ciphertext) with the same key. At one point in time, these N traces can be grouped into r groups according to the inputs of the crypto algorithm, and the ith group consists of n_i samples, scilicet $l_{i1}, l_{i2}, \ldots, l_{in_1}$, $i = 1, 2, \ldots, r$. Let $l_1, l_2, \ldots, l_r$denote the normally distributed populations corresponding to the r groups, respectively. That is,

$$l_{ij} \sim \phi(\mu_i, \sigma^2), \tag{4}$$

where $\phi(\mu_i, \sigma^2)$ represents the normal distribution with mean μ_i and variance σ^2, $i = 1, 2, \ldots, r$, $j = 1, 2, \ldots, n_i$.

For a leakage point in time, the corresponding leakage should be varied with the input variable of the crypto algorithm. Consequently, at this point, the leakage samples in all r groups drawn from the r populations should be statistically distinguishable. In other words, the means of these populations should have a significant difference. On the contrary, for a nonleakage point, the corresponding samples contain no information about the secret key. As a result, the variation of the samples should be random at this point. In this case, the samples in all r groups drawn from the r populations should be statistically indistinguishable. Namely, the means of these populations should have a statistically insignificant difference.

On the basis of the above-mentioned inference, we can perform leakage detection by using the one-way ANOVA. Table 1 depicts the data organization of the one-way ANOVA. Hence, the between-group variation SS_b and the within-group variation SS_w in equation (3) can be, respectively, rewritten as

$$SS_b = \sum_{i=1}^{r} \left(\overline{l_i} - \sum_{i=1}^{r} \frac{n_i \overline{l_i}}{n} \right)^2, \tag{5}$$

$$SS_w = \sum_{i=1}^{r} \sum_{j=1}^{n_i} \left(l_{ij} - \overline{l_i} \right)^2, \tag{6}$$

where $\overline{l_i}$ means the sample mean of the ith group. Obviously, the expectation of $\overline{l_i}$ is equal to μ_i. It can be viewed that the between-group variation SS_b indicates the difference between each group mean and the total sample mean, and the within-group variation SS_w indicates the difference between the samples in a group and the group mean. If SS_b is much greater than SS_w, it states that there is a significant difference among all groups, and these r populations are unlikely from the same normal population. From this viewpoint, it is reasonable to apply the one-way ANOVA for leakage detection, because the samples corresponding to leakage points contain information about multiple input data and can be deemed to come from different populations, while the samples corresponding to nonleakage points include random data information and can be considered to come from the same population.

The threshold for the hypothesis test is

$$\mathrm{Th} = F_{1-\alpha}(r-1, N-1), \tag{7}$$

where α denotes the significance level and F and $(r-1, N-1)$ represent the F distribution and the degrees of freedom, respectively. Points which exceed the threshold Th are acceptable as the leakage points, while the other points within the bound

Table 1: One-way ANOVA data organization.

Group	Observations of samples	Sample mean	Sample variance
1	$l_{11}, l_{12}, \ldots, l_{1n_1}$	$\overline{l_1}$	S_1^2
2	$l_{21}, l_{22}, \ldots, l_{2n_2}$	$\overline{l_2}$	S_2^2
3	$l_{31}, l_{32}, \ldots, l_{3n_3}$	$\overline{l_3}$	S_3^2
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$r-2$	$l_{r-2,1}, l_{r-2,2}, \ldots, l_{r-2,n_{r-2}}$	$\overline{l_{r-2}}$	S_{r-2}^2
$r-1$	$l_{r-1,1}, l_{r-1,2}, \ldots, l_{r-1,n_{r-1}}$	$\overline{l_{r-1}}$	S_{r-1}^2
r	$l_{r1}, l_{r2}, \ldots, l_{rn_r}$	$\overline{l_r}$	S_r^2

determined by the threshold are classified as the nonleakage points. The whole test procedure is described as follows:

Initially, for each point in time, divide the side-channel samples at this point into multiple groups according to the corresponding plaintext or ciphertext.

Secondly, compute the value of F-statistic in accordance with equations (3), (5), and (6).

Finally, compare the F-statistic with the threshold Th in equation (7), and mark the point as a leakage point ($F > \mathrm{Th}$) or a nonleakage point ($F \leq \mathrm{Th}$).

The proposed leakage detection algorithm (LD_ANOVA for short) is elaborated in Algorithm 1. The symbol L denotes a matrix constructed by side-channel traces in row, X denotes the corresponding plain text or cipher text set, and α denotes the significance level. Further, L can be represented by the following formula:

$$L = \bigcup_{m=1}^{M} \bigcup_{i=1}^{r_m} \bigcup_{j=1}^{n_i} l_{ij}. \tag{8}$$

3.2. Extension to Multichannel Leakages. Generally, a crypto device in operation leaks multiple types of side information simultaneously. Consider that multichannel leakages usually contain much more potential key-dependent information than monochannel leakage [17], and it is feasible for an analyser to collect these leakages from multiple channels in practice; we also investigate how to enhance the leakage detection result by utilizing multichannel leakages.

To make the proposed detection approach in the previous section apply to the case of multichannel leakages as well, a preprocessing step aiming at transforming the multichannel leakage sets into a fused leakage set is needed. Based on the types of crypto implementations, there are different methods to perform the transform [17]. For instance, it is suitable to use the treatment of *Simple Fusion Attack* (SFA) to perform transform, if multichannel measurements are acquired from a crypto implementation with a few leakage points. And the fusion way of Fusion Attack Based Singular Value Decomposition (SVD_FA) may be a better fit for an implementation with multiple leakage points [17].

Algorithm 2 describes the proposed leakage detection algorithm for multichannel leakage detection. In the algorithm, the pseudocode in the first row means that, according to the leakage characteristic of a crypto implementation, applying a function Tr to transform the multichannel leakage sets *ML* into a monochannel leakage set L and transform the multichannel input sets *MX* into a monochannel input set X corresponding to L. The operator $\mathrm{Tr}(\cdot)$ indicates the corresponding transform.

3.3. SCA Distinguisher Based on One-Way ANOVA. Interestingly, we find that if we group the side-channel leakage samples in Table 1 in accordance with the values of a targeted intermediate variable, we can develop a novel SCA distinguisher to perform key recovery.

The idea is that the difference of means among all groups is supposed to be the most significant when the leakage data is categorized correctly. As compared to the correct key, the wrong key candidates will make the samples in the same group scattered in disparate groups and therefore make the difference of all populations indistinguishable and the leakage information related to the correct key hidden. Hence, the SCA distinguisher can be defined as follows:

$$\Delta \xrightarrow{\text{yields}} \widehat{k} \rightleftharpoons F(\widehat{k}) = \max_{k} F(k). \tag{9}$$

In equation (9), k and $\widehat{k}$ denote the candidates of key and the key guess, respectively. In addition, $F(\cdot)$ represents the F-statistic function in equation (3).

Naturally, a new SCA method yields. Its details are shown in Algorithm 3.

4. Experimental Results and Discussion

4.1. Monochannel Leakage Detection. To access the effectiveness of the proposed detection approach (i.e., LD_ANOVA), we collected two sets of side-channel traces to perform monochannel leakage detection at first. These two side-channel leakage sets include 30, 000 power traces acquired from an unprotected AES-128 encryption algorithm running on FPGA (Xilinx Kintex-7) and 5, 000 power traces acquired from a masked AES-128 encryption algorithm running on an embedded 8-bit MCU. The FPGA implementation belongs to a parallel hardware implementation, while the MCU version is a serial software implementation. The traces are all corresponding to random inputs. Parameters of the leakage acquisition are listed in Table 2.

Figures 1 and 2 show the detection results of the FPGA implementation by the proposed method and the other two typical detection methods. The two selected approaches used for comparison are the TVLA technology [13] and its complement, namely, the leakage detection with the χ^2-test (LD_CS for short) [11]. Compared with the nonspecific t-test, the specific t-test can offer higher confidence of detection at a lower cost [18]; the paper uses the latter t-test as the comparison. All values of the significance level α in the three aforementioned methods for leakage detection are set to 0.0001 to obtain results at a confidence of 99.99% [5]. Note that, the red-dashed lines in all figures represent the thresholds determined by the significance level α.

It can be seen that our proposed approach performs better than the other methods. For instance, LD_ANOVA is capable of finding the first leakage point by exploiting 10, 000 traces. In this case, both the TVLA and LD_CS fail to

```
     Input: $L$, $X$, and $\alpha$
     Output: leakage point set $LP$
 (1) $LP \longleftarrow \varnothing$
 (2) For $m = 1, 2, \ldots, M$ // $M$ denotes the number of points per trace
 (3)   $SS_b \longleftarrow \varnothing$, $SS_w \longleftarrow \varnothing$, $F \longleftarrow \varnothing$, $\mathrm{Th} \longleftarrow \varnothing$, $\overrightarrow{\mu} \longleftarrow \varnothing$
 (4)   Divide the $m^{\text{th}}$ column of $L$ into $r_m$ groups according to the samples of $X$
 (5)   For $i = 1, 2, \ldots, r_m$
 (6)     $\overrightarrow{\mu} \longleftarrow \overrightarrow{\mu} \cup (\sum_{j=1}^{n_i} l_{ij}/n_i)$
 (7)   End
 (8)   $SS_b \longleftarrow \sum_{i=1}^{r_m} (\overrightarrow{\mu_i} - \sum_{i=1}^{r_m} n_i\overrightarrow{\mu_i}/n)^2$ $SS_w \longleftarrow \sum_{i=1}^{r_m} \sum_{j=1}^{n_i} (l_{ij} - \overline{l_i})^2$ // $\overrightarrow{\mu_i}$ denotes the $j^{\text{th}}$ element of the vector $\overrightarrow{\mu}$.
 (9)   $F \longleftarrow (N-1)SS_b/[(r-1)SS_w]$
(10)   $\mathrm{Th} \longleftarrow F_{1-\alpha}(r-1, N-1)$
(11)   $LP \longleftarrow LP \cup t_m$, iff $F > \mathrm{Th}$
(12) End
(13) Return: $LP$.
```

Algorithm 1: Leakage detection based on one-way ANOVA.

```
     Input: $ML$, $MX$, and $\alpha$
     Output: leakage point set $LP$
 (1) $L \leftarrow \mathrm{Tr}(ML)$, $X \leftarrow \mathrm{Tr}(MX)$
 (2) On the basis of $X$ and $\alpha$, apply Algorithm 1 to detect leakage points
     Return: $LP$.
```

Algorithm 2: Leakage detection with multichannels leakages.

```
     Input: $L$, $X$, and $K$
     Output: key guess $\widehat{k}$
 (1) Select a targeted intermediate variable
 (2) For $i = 1, 2, \ldots, |K|$
 (3)   Calculate the intermediate values in accordance with the samples of $X$
 (4)   $F_0 \longleftarrow \varnothing$
 (5)   For $m = 1, 2, \ldots, M$
 (6)     $SS_b \longleftarrow \varnothing$, $SS_w \longleftarrow \varnothing$, $\overrightarrow{\mu} \longleftarrow \varnothing$
 (7)     Divide the $m^{\text{th}}$ column of $L$ into $r_m$ groups according to the target intermediate values
 (8)     For $i = 1, 2, \ldots, r_m$
 (9)       $\overrightarrow{\mu} \longleftarrow \overrightarrow{\mu} \cup (\sum_{j=1}^{n_i} l_{ij}/n_i)$
(10)     End
(11)     $SS_b \longleftarrow \sum_{i=1}^{r_m} (\overrightarrow{\mu_i} - \sum_{i=1}^{r_m} n_i\overrightarrow{\mu_i}/n)^2$ $SS_w \longleftarrow \sum_{i=1}^{r_m} \sum_{j=1}^{n_i} (l_{ij} - \overline{l_i})^2$
(12)     $F_0 \longleftarrow F_0 \cup (N-1)SS_b/[(r-1)SS_w]$
(13)   End
(14)   $F(k) \longleftarrow \max\{F_0\}$
(15) End
(16) $\widehat{k} \longleftarrow \max_k F(k)$
     Return: $\widehat{k}$
```

Algorithm 3: SCA distinguisher based on one-way ANOVA.

detect any leakage point. Moreover, LD_ANOVA always identifies more leakage points than the TVLA and LD_CS. A comparison for the number of leakage points detected by these three methods is listed in Table 3 (rows 2, 6, and 10). In order to accurately evaluate the efficiency of the proposed method, the false positive points (i.e., the detected non-leakage points, denoted as N_{fp}) and the false negative points (i.e., the undetected leakage points, denoted as N_{fn}) are shown in Figure 3. Likewise, the number of the detected "real" leakage points are investigated (denoted as N_c).

Table 2: Parameters of the leakage acquisition.

Crypto implementation	Sample rate	Number of samples	Number of traces	Leakage model
FPGA	5G Sa/s	2,500	30,000	Hamming Distance
MCU	1G Sa/s	20,000	5,000	Hamming Weight

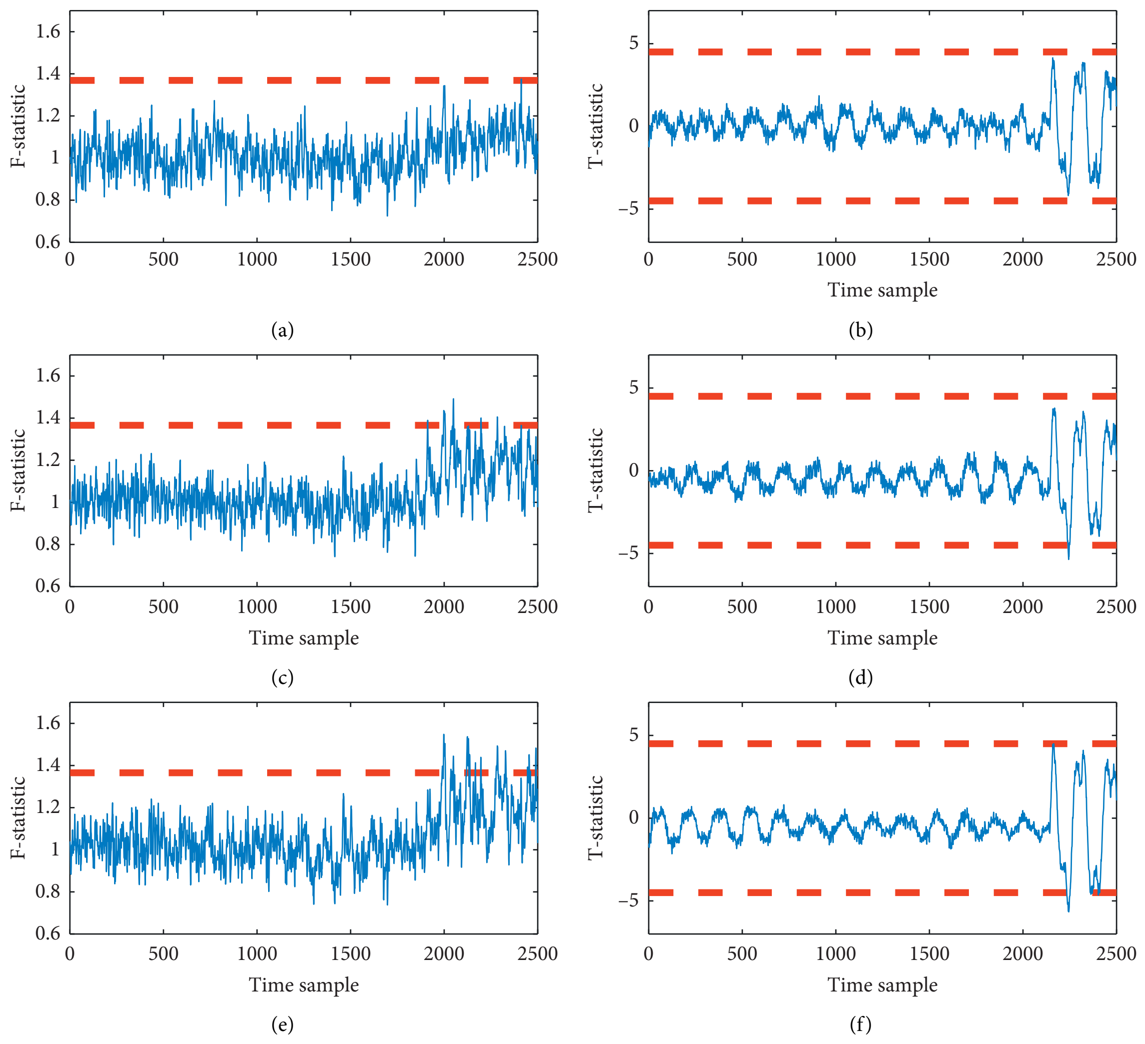


Figure 1: Detection results of TVLA and LD_ANOVA for the FPGA implementation. Left column: results of LD_ANOVA. Right column: results of TVLA. Upper row: 10,000 used traces. Middle row: 20,000 used traces. Lower row: 30,000 used traces.

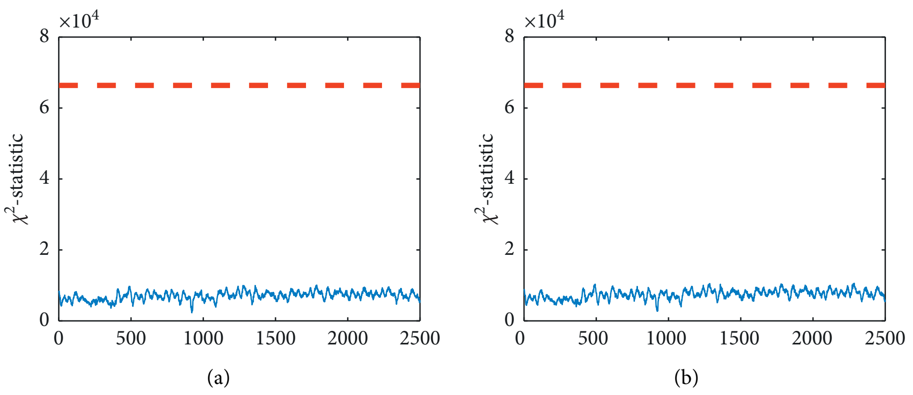
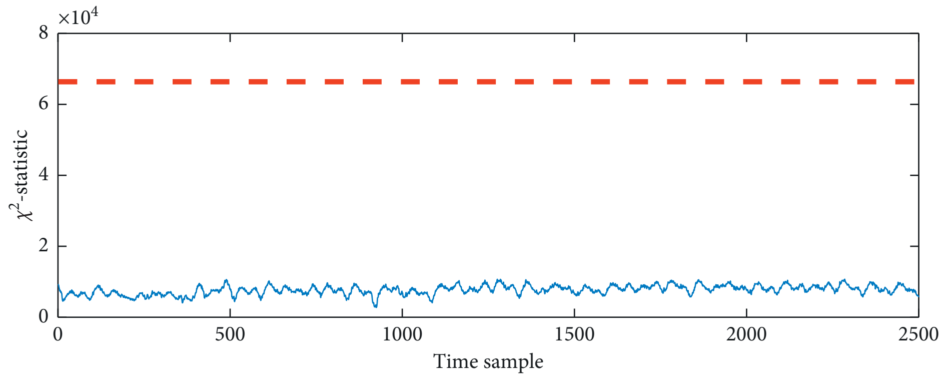


Figure 2: Continued.



(c)

FIGURE 2: Detection results of LD_CS for the FPGA implementation. Upper row (left column): 10,000 used traces. Upper row (right column): 20,000 used traces. Lower row: 30,000 used traces.

TABLE 3: Detection results of the FPGA implementation.

Detection method		10,000 traces	20,000 traces	30,000 traces
LD_ANOVA	D	1 point	21 points	68 points
	N_{fp}/R_{fp}	0/0%	4/2.72%	7/4.76%
	N_{fn}/R_{fn}	146/99.32%	130/88.44%	86/58.50%
	N_c/R_c	1/0.68%	17/11.56%	61/41.50%
TVLA	D	0 points	7 points	21 points
	N_{fp}/R_{fp}	0/0%	3/2.04%	3/2.04%
	N_{fn}/R_{fn}	147/100%	143/97.28%	129/87.76%
	N_c/R_c	0/0.0%	4/2.72%	18/12.24%
LD_CS	D	0 points	0 points	0 points
	N_{fp}/R_{fp}	0/0%	0/0%	0/0%
	N_{fn}/R_{fn}	147/100%	147/100%	147/100%
	N_c/R_c	0/0%	0/0%	0/0%

Naturally, the coverage rate of the “real” leakage points, the false positive rate, and the false negative rate are also calculated and illustrated in Figure 3. These three metrics are defined as follows:

$$R_{fp} = \frac{|D - D \cap P|}{|P|} \times 100\%, \tag{10}$$

$$R_{fn} = \frac{|P - D \cap P|}{|P|} \times 100\%, \tag{11}$$

$$R_c = 100\% - R_{fn} = \frac{|D \cap P|}{|P|} \times 100\%, \tag{12}$$

where P means the leakage point set of an implementation; D means the detected leakage point set; “$|\cdot|$” denotes the cardinality of a set; and R_{fp}, R_{fn}, and R_c represent the false positive rate, the false negative rate, and the coverage rate, respectively. It is noted that an efficient detection method should lead to lower R_{fp} and R_{fn} and higher R_c as compared to an inefficient method. From the data in Table 3, it can draw a conclusion that the proposed method is more efficient than the other two.

For the masked implementation, before applying for the detection approaches, the side-channel samples are recombined by the normalized product function to eliminate the influence of Boolean masking [19]. The preprocessing function computes the product of the leakages corresponding to the masked intermediate variable and the masking variable as the leakage of the key-dependent variable so that leakage points can be matched with the inputs. Figures 3 and 4 illustrate that LD_ANOVA still exhibits the best performance. For example, it marks 38 points as leakage points when the TVLA finds one leakage point by using 1,000 traces. Although these points contain plenty of nonleakage points, there are still 20 leakage points. Due to the limited number of traces, the detection result is changeable; for example, only 4 leakage points are discovered when we use 3,000 traces to execute LD_ANOVA. The number of the detected leakage points increases with the increasing traces; for example, 39 leakage points are identified when 5,000 traces are used. The information of leakage points detected by the three methods is described in Table 4. The three rates defined in equations (10)–(12) are demonstrated in Table 4 as well. The results lead to a similar inference to the first experiment.

Besides, the coverage rate in Table 4 becomes higher than that in Table 3 when all the traces are used. The cause of this phenomenon is that the FPGA implementation is a parallel implementation which makes the collected traces include more switching noise [20] as compared to the 8-bit MCU implementation.

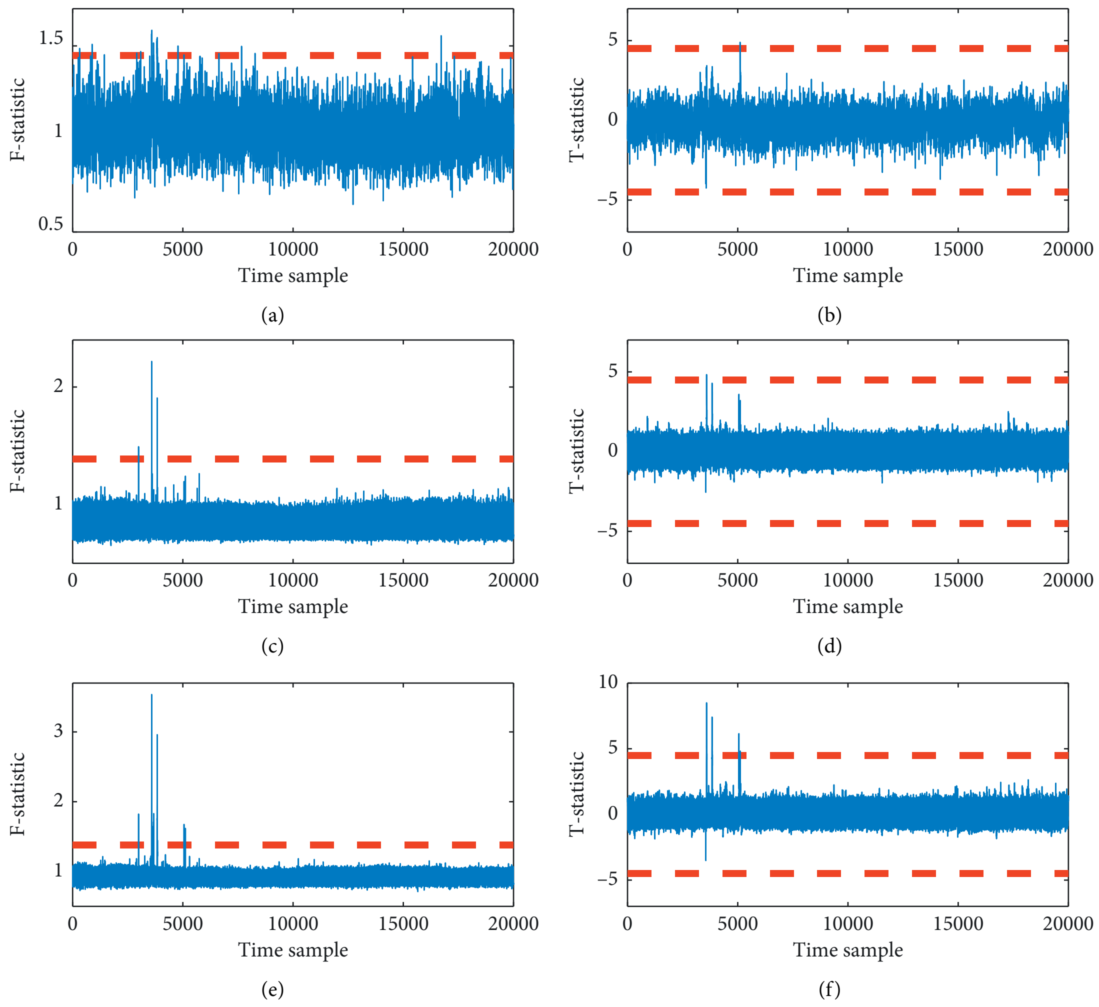


Figure 3: Detection results of TVLA and LD_ANOVA for the 8-bit MCU implementation. Left column: results of LD_ANOVA. Right column: results of TVLA. Upper row: 1,000 used traces. Middle row: 3,000 used traces. Lower row: 5,000 used traces.

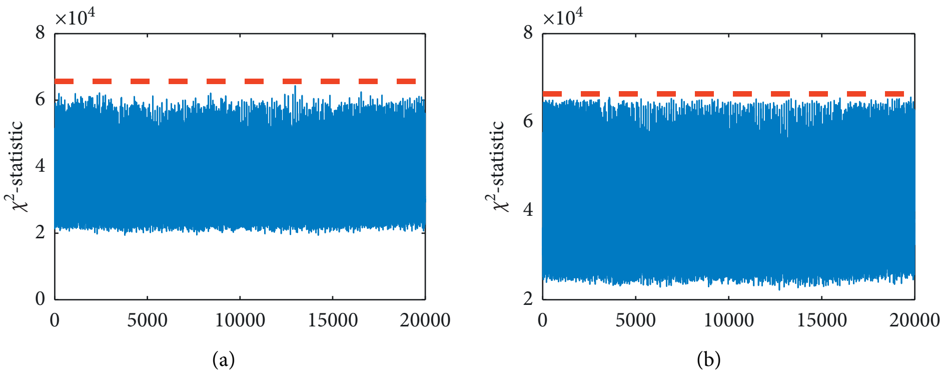
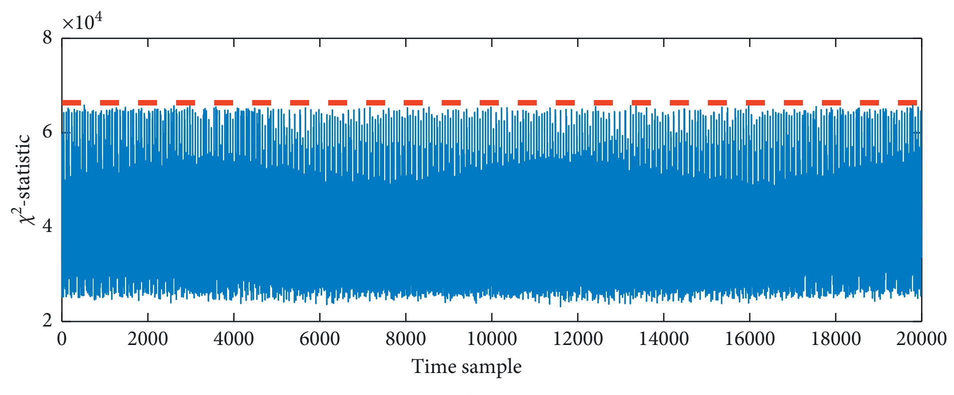


Figure 4: Continued.



(c)

Figure 4: Detection results of LD_CS for the 8-bit MCU implementation. Upper row (left column): 1,000 used traces. Upper row (right column): 3,000 used traces. Lower row: 5,000 used traces.

Table 4: Detection results of the 8-bit MCU implementation.

Detection method		1,000 traces	3,000 traces	5,000 traces
LD_ANOVA	D	38 points	4 points	39 points
	N_{fp}/R_{fp}	18/36.73%	0/0%	0/0%
	N_{fn}/R_{fn}	29/59.18	45/91.84%	10/20.41%
	N_c/R_c	20/41.82%	4/8.16%	39/79.59%
TVLA	D	1 point	1 point	28 points
	N_{fp}/R_{fp}	0/0%	0/0%	0/0%
	N_{fn}/R_{fn}	48/97.96%	48/97.96%	21/42.86%
	N_c/R_c	1/2.04%	1/2.04%	28/57.14%
LD_CS	D	0 points	0 points	0 points
	N_{fp}/R_{fp}	0/0%	0/0%	0/0%
	N_{fn}/R_{fn}	49/100%	49/100%	49/100%
	N_c/R_c	0/0%	0/0%	0/0%

It is noted that LD_CS fails to distinguish leakage points from nonleakage points in the above two experiments. The reason is that the method needs to divided leakage samples into multiple bins before performing the test of independence, and the process is vulnerable to the noise in the side-channel traces, which will influence the detection efficiency [12]. The reason that LD_ANOVA shows a better performance than the TVLA is because the former divides side-channel samples into more than two classes according to the input classes and obtains much more information about the relevance between the two.

4.2. Multichannel Leakage Detection. Consider that multiple types of leakages of a running cryptographic device can be collected from multiple side channels, which may expose much more information that monochannel SCA cannot utilize; the proposed method is used for multichannel leakage detection as well. We simultaneously collected 30, 000 and 5,000 electromagnetic (EM) traces from the FPGA and MCU implementations in the previous section (Section A, Section 4), respectively. Due to the fact that the oscilloscope we used in the experiments can only collect the power traces and the EM traces with the same sample rate, the sample rates of these two EM trace sets are set to be the same as the sample rates of the corresponding power traces (viz., 5G Sa/s and 1G Sa/s). The sample rates are experimentally selected to ensure both the power and EM trace sets contain adequate leakage information for side-channel analysis.

Figure 5 depicts an example of multichannel leakage detection for the FPGA implementation. The fused trace set is obtained by the nonnegative matrix factorization-based leakage fusion [17]. Two monochannel detection results by utilizing the power leakage and the EM leakage are also illustrated in the figure. As shown in Figure 5, multichannel leakage detection performs better than any monochannel leakage detection.

Figure 6 compares the union of the detected power leakage points and the detected EM leakage points to the result of the multichannel leakage detection by exploiting fused traces. It can be viewed that the fusion of monochannel leakages makes a mass of potential monochannel leakage points exposed. For instance, 341 leakage points are identified when the multichannel detection uses all fused traces, while the monochannel detection only identifies 50 and 61 leakage points, respectively. Furthermore, the fusion of the differing channel makes the influence of the switching noise

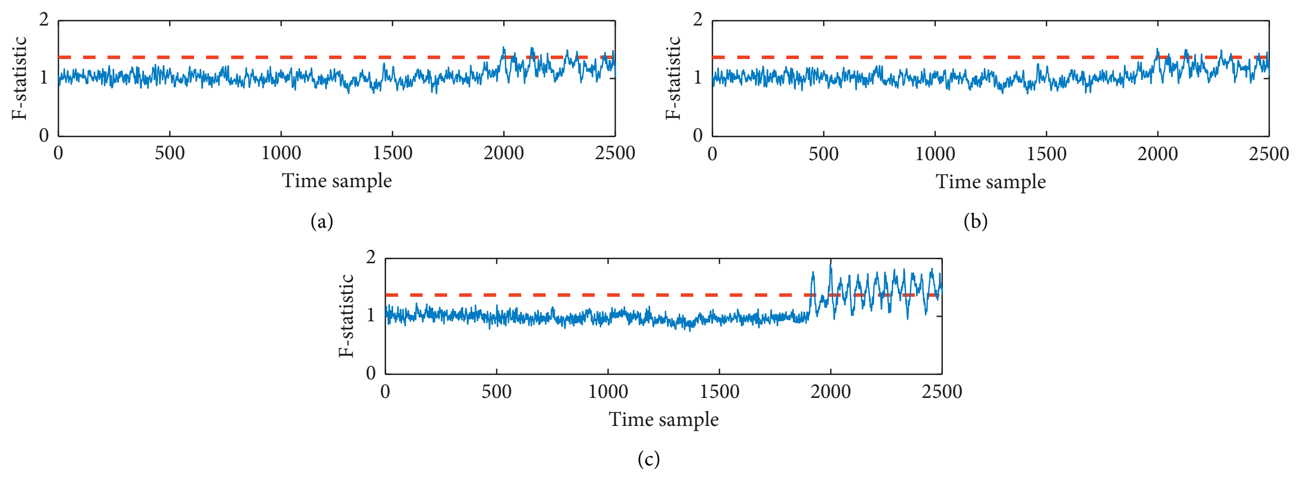


Figure 5: Results of monochannel leakage detection and multichannel leakage detection by LD_ANOVA for the FPGA implementation. Upper row: detected power leakage points (10,000 used traces). Middle row: detected EM leakage points (20,000 used traces). Lower row: detected fused leakage points (30,000 used traces).

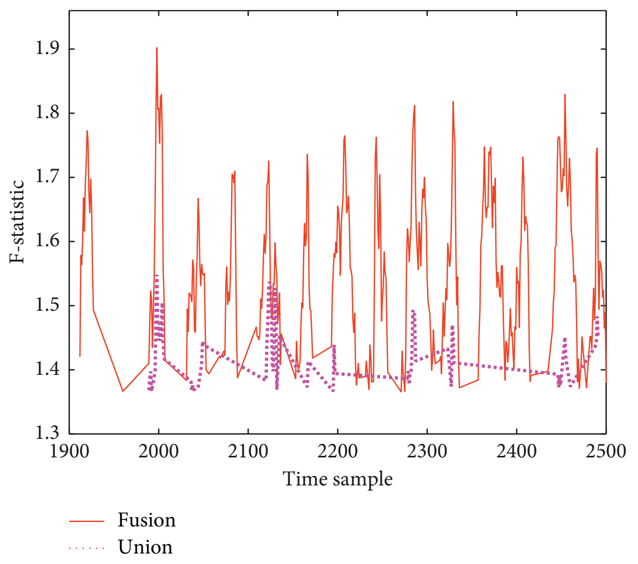


Figure 6: Detected leakage points of fused traces and union of detected monochannel leakage points (the FPGA implementation). "Union": union of detected power and EM leakage point sets. "Fusion": detected leakage points of fused traces.

in the traces significantly reduce and the coverage rate significantly increase. More detailed data is illustrated in Table 5.

The detection results for the 8-bit MCU implementation draw a similar conclusion. Similarly, the corresponding details are described in Table 6.

4.3. Runtime and Memory. To ensure that our proposed method can be efficiently performed for practical evaluation, the runtime and memory of the proposed approach are investigated experientially. We perform a monochannel leakage detection by applying TVLA, LD_CS, and LD_ANOVA, respectively.

This experiment is executed on the HP laptop with the Intel Core i7-7500U CPU (Dual-Core, 2.70 GHz and 2.90 GHz), 8 GB RAM, and MATLAB R2017a software platform. The three detections are repeated 10 times with 30,000 power traces acquired from the FPGA implementation in Section A, and the averages of the runtimes and the memories are plotted in Figure 7.

Note that, the code of the TVLA approach is performed in parallel. As a result, it can be seen that TVLA is the most efficient in terms of time. Moreover, LD_ANOVA is the most efficient in terms of memory, while LD_CS is the most inefficient—in both time and memory. On the whole, the computation cost of a practical security evaluation applying the proposed leakage detection approach is able to afford an analyser.

4.4. New Attack. Finally, two practical experiments are also carried out to evaluate the effectiveness of the proposed new

Table 5: Monochannel and multichannel leakage detection results for the FPGA implementation.

Leakage		10,000 traces	20,000 traces	30,000 traces
Power traces	D	1 point	21 points	68 points
	N_{fp}/R_{fp}	0/0%	4/2.72%	7/4.76%
	N_{fn}/R_{fn}	146/99.32%	130/88.44%	86/58.50%
	N_c/R_c	1/0.68%	17/11.56%	61/41.50%
EM traces	D	0 points	19 points	52 points
	N_{fp}/R_{fp}	0/0%	0/0%	2/1.64%
	N_{fn}/R_{fn}	122/100%	103/84.43%	72/59.02%
	N_c/R_c	0/0%	19/15.57%	50/40.98%
Fused traces	D	4 points	168 points	355 points
	N_{fp}/R_{fp}	0/0%	8/2.35%	14/4.11%
	N_{fn}/R_{fn}	337/98.83%	181/53.08%	0/0%
	N_c/R_c	4/1.17%	160/46.92%	341/100%

Table 6: Monochannel and multichannel leakage detection results for the 8-bit MCU implementation.

Leakage		1,000 traces	3,000 traces	5,000 traces
Power traces	D	38 points	4 points	39 points
	N_{fp}/R_{fp}	18/36.73%	0/0%	0/0%
	N_{fn}/R_{fn}	29/59.18	45/91.84%	10/20.41%
	N_c/R_c	20/41.82%	4/8.16%	39/79.59%
EM traces	D	24 points	3 points	24 points
	N_{fp}/R_{fp}	16/43.24%	0/0%	0
	N_{fn}/R_{fn}	29/78.38%	34/91.89%	13/35.14%
	N_c/R_c	8/21.62%	3/8.11%	24/64.86%
Fused traces	D	15 points	54 points	62 points
	N_{fp}/R_{fp}	2/3.13%	2/3.13%	0/0%
	N_{fn}/R_{fn}	51/79.69%	12/18.75%	2/3.13%
	N_c/R_c	13/20.31%	52/81.25%	62/96.87%

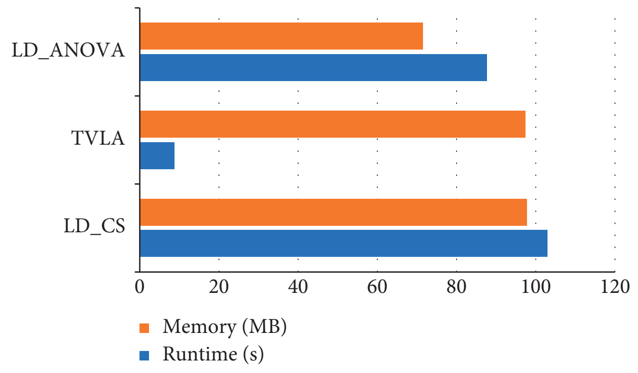
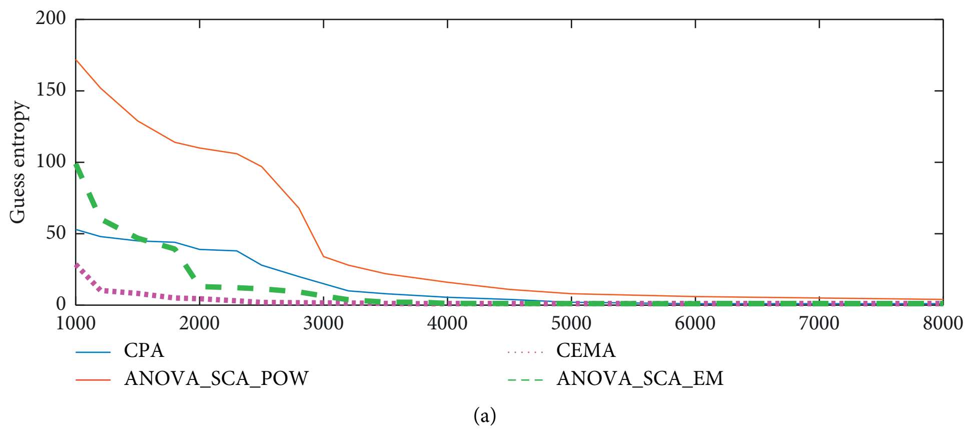


Figure 7: Runtimes and memories of LD_ANOVA, TVLA, and LD_CS.

SCA attack based on ANOVA (ANOVA_SCA for short) in Algorithm 3. Two power trace sets and two EM trace sets of the two aforementioned implementations in Section A are exploited to launch the key attack recovery, respectively. The leakage models of the FPGA and MCU implementations are approximate to the Hamming Distance and Hamming Weight models, respectively. Besides, the randomly selected attack targets are the XOR of the input and output of the 9th S-box in the last round of AES-128 and the output of the 16th S-box in the last round of the encryption algorithm, respectively. The other classical SCA attack, namely, correlation analysis attack (CAA) [17, 21, 22], is also used for comparison. The security metric, guess entropy [23], serves as the evaluation metric of the key recovery attacks. The attack results depicted in Figure 8 confirm the efficacy and efficiency of ANOVA_SCA. In particular, ANOVA_SCA is even much more efficient than CAA for the masked implementation.

We also estimate the average runtimes and the average memories when performing ANOVA_SCA and CAA, respectively. These two attacks are repeated 10 times with 30,000 power traces acquired from the FPGA implementation in Section A. This experiment is executed on the same HP laptop and MATLAB software platform mentioned in Section C. The statistical results are plotted in Figure 9. It can be viewed that it is feasible to apply the proposed attack for security assessment because of its practical utility and high efficiency.



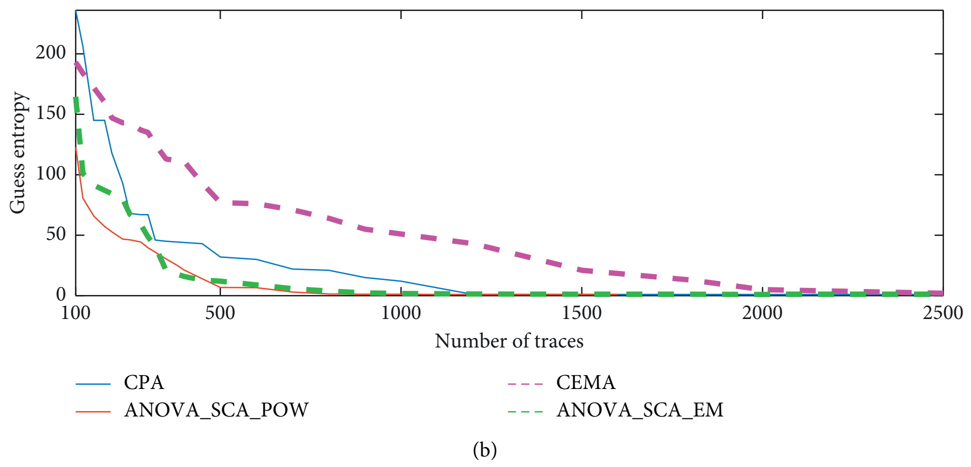


Figure 8: Guess entropy curves of CAA and ANOVA_SCA for the FPGA implementation and the 8-bit MCU implementation. Upper row: FPGA implementation. Lower row: MCU implementation. “CPA”: CAA attack exploiting power traces. “ANOVA_SCA_POW”: ANOVA_SCA attack exploiting power traces. “CEMA”: CAA attack exploiting EM traces. “ANOVA_SCA_EM”: ANOVA_SCA attack exploiting with EM traces.

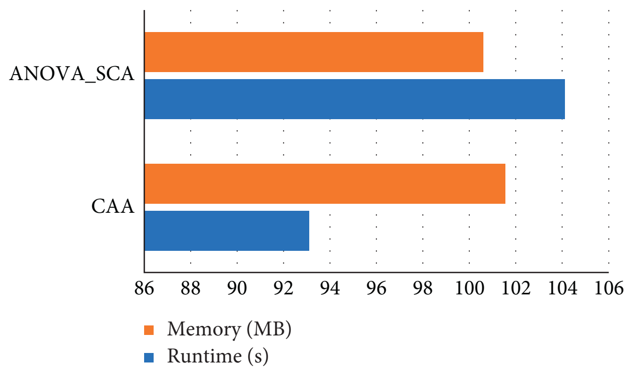


Figure 9: Runtimes and memories of ANOVA_SCA and CAA.

5. Conclusions

The paper develops a black-box side-channel leakage detection approach, which is designed for the security assessment of cryptographic devices. The proposed detection method is based on the one-way ANOVA, which ensures that the evaluation is highly efficient—in terms of both time and memory. On the basis of monochannel leakage detection, the proposed method is extended to detect multichannel leakages as well. Compared with utilizing monochannel leakage individually, the use of the fused leakage is beneficial in finding much more leakage points efficiently. Furthermore, the proposed detection method is capable of applying for the key recovery attack, when a key-dependent immediate variable serves as the input of the method. In the future, the extension of our proposed approach to enhance the performance of multichannel leakage detection will be investigated further.

Data Availability

The side-channel measurement data and codes used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by National Natural Science Foundation of China (nos. 61802186, 61472189, 61572255, and 62002167).

References

[1] P. C. Kocher, "Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems," in *Proceedings of the CRYPTO 1996*, pp. 104–113, Santa Barbara, CA, USA, August 1996.

[2] P. C. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Proceedngs of the CRYPTO 1999*, pp. 388–397, Santa Barbara, CA, USA, August 1999.

[3] K. Gandolfi, C. Mourtel, and F. Olivier, "Electromagnetic analysis: concrete results," in *Proceedings of the Cryptographic Hardware and Embedded Systems -2001*, pp. 251–261, Paris, France, May 2001.

[4] D. Genkin, A. Shamir, and E. Tromer, "RSA key extraction via low-bandwidth acoustic cryptanalysis," in *Proceedings of the CRYPTO 2014*, pp. 444–461, Santa Barbara, CA, USA, August 2014.

[5] G. Goodwill, B. Jun, J. Jaffe et al., "A testing methodology for side-channel resistance validation," in *Proceedings of the NIST NIAT 2011*, pp. 115–136, Nara, Japan, September 2011.

[6] S. Bhasin, J. Danger, S. Guilley et al., "NICV: normalized interclass variance for detection of side-channel leakage," in *Proceedings of the EMC'14*, pp. 310–313, Tokyo, Japan, May 2014.

[7] J. Liu, Z. Guo, D. Gu et al., "Enhanced side-channel leakage detection method by considering combinational logic," *China Communications*, vol. 12, no. 6, pp. 1–10, 2015.

[8] L. Mather, E. Oswald, J. Bandenburg et al., "Does my device leak information? an a priori statistical power analysis of leakage detection tests," in *Proceedings of the ASIACRYPT 2013*, pp. 486–505, Bengaluru, India, December 2013.

[9] F. Durvaux and F. Standaert, "From improved leakage detection to the detection of points of interests in leakage traces," in *Proceedings of the EUROCRYPT 2016*, pp. 240–262, Vienna, Austria, May 2016.

[10] A. A. Ding, C. Chen, and T. Eisenbarth, "Simpler, faster, and more robust t-test based leakage detection," in *Proceedings of the COSADE 2016*, pp. 163–183, Graz, Austria, April 2016.

[11] A. Moradi, B. Richter, T. Schneider et al., "Leakage detection with the χ^2-test," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2018, no. 1, pp. 209–237, 2018.

[12] W. Yang, H. Zhang, Y. Gao et al., "Side-Channel leakage detection based on constant parameter channel model," in *Proceedings of the International Conference on Computer Design 2020*, pp. 553–560, Hartford, CT, USA, October 2020.

[13] G. Becker, J. Cooper, E. DeMulder, G. Goodwill et al., "Test vector leakage assessment (TVLA) methodology in practice," in *Proceedings of the International Catholic Migration Commission 2013*, Gaithersburg, MD, USA, September 2013.

[14] D. B. Roy, S. Bhasin, S. Guilley, A. Heuser, S. Patranabis, and D. Mukhopadhyay, "CC meets FIPS: a hybrid test methodology for first order side channel analysis," *IEEE Transactions on Computers*, vol. 68, no. 3, pp. 347–361, 2019.

[15] S. McKillup, *Statistics Explained: An Introductory Guide for Life Scientists*, Cambridge University Press, Cambridge, UK, 2nd edition, 2011.

[16] W. Snedecor and C. George, *Statistical Methods*, Iowa State University Press, Sioux, IA, USA, 8th edition, 1989.

[17] W. Yang, Y. Zhou, Y. Cao, H. Zhang, Q. Zhang, and H. Wang, "multi-channel fusion attacks," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 8, pp. 1757–1771, 2017.

[18] T. Schneider and A. Moradi, "Leakage assessment methodology–a clear roadmap for side-channel evaluations," in *Proceedings of the Cryptographic Hardware and Embedded Systems - 2015*, pp. 495–513, SaintMalo, France, September 2015.

[19] P. Bottinelli and J. W. Bos, "Computational aspects of correlation power analysis," *Journal of Cryptographic Engineering*, vol. 7, no. 3, pp. 167–181, 2017.

[20] S. Mangard, E. Oswald, and T. Popp, *Power Analysis Attacks: Revealing the Secrets of Smart Cards*, Springer, New York, NY, USA, 2007.

[21] E. Brier, C. Clavier, and F. Olivier, "Correlation power analysis with a leakage model," in *Proceedings of the Cryptographic Hardware and Embedded Systems-2004*, pp. 16–29, Boston, MA, USA, August 2004.

[22] E. Peeters, *Advanced DPA Theory and Practice*, Springer, New York, NY, USA, 2013.

[23] F.-X. Standaert, T. G. Malkin, and M. Yung, "A unified framework for the analysis of side-channel key recovery attacks," in *Proceedings of the EUROCRYPT 2009*, pp. 443–461, Cologne, Germany, April 2009.