

Scatter: a Missing Case?

Yuanyuan Zhou^{1,2}, Sébastien Duval¹, and François-Xavier Standaert¹

¹ UCLouvain, Crypto Group, Belgium

² Brightsight BV, Delft, The Netherlands

Abstract. **Scatter** is a multivariate transform proposed in combination with the Chi² and MIA distinguishers at COSADE 2018. Its primary motivation is to inherently deal with the misalignment and synchronization issues that may decrease the efficiency of concrete side-channel attacks. In this paper, we first show empirically that when compared to natural competitors for first-order multivariate attacks (e.g., exploiting linear regression on-the-fly), it does not bring improvements in the (simulated and actual) implementation settings studied by its authors. We then show that the same holds in the higher-order case: in most practically-relevant settings, **Scatter** works best when combined with a combination function mixing the leakage samples in a non-linear manner, bringing it back to a situation where it does not improve standard distinguishers.

Keywords: Side-Channel Analysis · Scatter Transform · Shuffling · Masking

1 Introduction

Side-channel attacks are an important threat to the security of modern embedded devices [MOP07]. Masking [CJRR99,GP99] and shuffling [HOM06,VMKS12] are among the most investigated solutions to mitigate these attacks.

Informally, masking can be viewed as a data randomization which aims at forcing the adversary to estimate higher-order statistical moments of the leakage distributions; similarly, shuffling can be viewed as a time randomization which aims at forcing the adversary to deal with information spread in multivariate distributions. As a result, evaluating a masked and/or shuffled implementation boils down to a quest for simple and efficient tools enabling the analysis of higher-order and multivariate statistical distributions. The literature typically divides such distinguishers as profiled ones, like Template Attacks (TAs) [CRR02], where the adversary can use a device he controls to build a leakage model, and non-profiled ones, like Correlation Power Analysis (CPA) [BCO04], where the adversary uses a hypothetical model based on engineering intuition.

The **Scatter** transform was introduced at COSADE 2018 [TGWC18]. Roughly, it is a multivariate pre-processing to use in combination with “generic-emulating” distinguishers [WOS14], such as Mutual Information Analysis (MIA) [GBTP08] or the Chi² test [MRSS18]. Its main motivation comes from the observation that the efficiency of concrete side-channel attacks can be significantly reduced in

case of misaligned traces, which may be due to jitter in the measurements or to dedicated countermeasures such as shuffling (or random delays [CK10]). **Scatter** is claimed to efficiently deal with such synchronization issues, while having potential for improving higher-order side-channel attacks (e.g., against masked implementations) [TVW19]. Preliminary experiments showed good features in these directions, but a comparison with competing distinguishers is missing.

In this paper, we complete this research in two directions.

We start by investigating the basic potential of **Scatter** for an efficient exploitation of first-order multivariate leakages. For this purpose, our seed observation is that the COSADE 2018 paper mostly compared **Scatter** with univariate CPA-based attacks. In this context, it appears natural that **Scatter** resists better to misaligned traces, since the misalignment will typically spread the informative samples over multiple time dimensions (i.e., a multivariate distribution). We therefore compare the efficiency of **Scatter** with a more natural competitor for first-order multivariate attacks, namely the on-the-fly regression-based distinguisher described in [DPRS11]. We performed experiments against a simulated shuffled implementation and a concrete jittery implementation, both similar to the settings investigated in [TGWC18]. Our results suggest that the on-the-fly regression always outperforms **Scatter** in these contexts.

We follow by studying the applicability of **Scatter** to masked implementations where computations are performed on secret-shared data.

In this respect, we first show that **Scatter**'s basic (univariate) probabilistic transform is inherently unable to characterize the higher-order multivariate statistical leakages of a masked implementation. Hence, the only possible option to deal with such cases is (as usual) to generalize **Scatter** to multivariate distributions, either by estimating these distributions directly or by combining the leakage samples in a non-linear manner (see for example [PRB09,SVO⁺10]). Note that the latter implies that **Scatter** cannot avoid the combinatorial explosion of the number of samples to test in order to detect Points-of-Interest (POIs).

Our experiments next confirm the findings of Thiebeauld et al. that non-linear combination functions are beneficial for the efficiency of higher-order **Scatter** from a data complexity viewpoint [TVW19]. Since we are then back to a situation where on-the-fly regression is applicable, we finally compare both distinguishers and show empirically that, as in the first-order context, **Scatter** is outperformed by linear regression in the simulated cases we studied.

Overall, we cannot preclude another useful application of **Scatter**. But in the absence of theoretical or empirical arguments highlighting its interest over other established distinguishers, we conclude that it currently lacks a use case.

Note that our study is limited to the investigation of **Scatter** in combination with side-channel distinguishers (as it was proposed so far). One possible scope for further investigation is the study of this probabilistic transform as a pre-processing before leakage detection (e.g., with the Chi² test [MRSS18]).

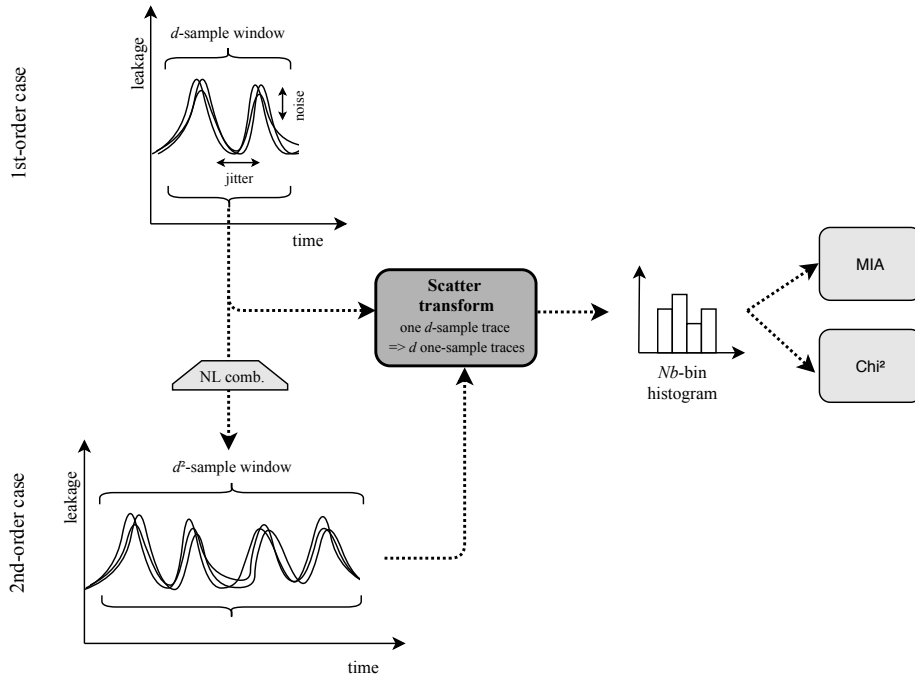


Fig. 1: Scatter transform with MIA and Chi^2 distinguishers: high-level view.

2 Background

2.1 Scatter transform with Chi^2 /MIA distinguishers

The Scatter transform applied to first-order leakages and combined with the Chi^2 and MIA distinguishers is illustrated on the top of Figure 1. The basic idea is to consider each d -dimension trace as d (one-dimension) samples, and to estimate the distribution of these d samples thanks to histograms. The histograms are then partitioned according to the key guess and a hypothetical leakage model (e.g., the Hamming weight of an S-box output). The Chi^2 or MIA distinguishers are finally used to search for the correct key guess. More precisely:

1. We estimate histograms based on the amplitude of the sample points within a window of size d . For each measured trace, we convert the d sample points to an N_b -bin histogram. For an 8-bit oscilloscope, the max N_b is 256.
2. The “histogram traces” are then partitioned based on the key guess and hypothetical leakage model. In this work we consider the Hamming weight leakages of an AES S-box output. As a result, we obtain $N_b \times 9 \times 256$ partitioned histogram traces (i.e., 9 Hamming weights, 256 key candidates).

3. We compute the distributions $\text{pdf}_{g,h}[u]$ using the partitioned histogram traces, for each key guess g and corresponding Hamming weight hypothetical leakage h , where u denotes the histogram value:

$$\text{pdf}_{g,h}[u] = \frac{\text{Acc}_{g,h}[u]}{\sum_{u'=0}^{N_b-1} \text{Acc}_{g,h}[u']}, \quad (1)$$

in which $\text{Acc}_{g,h}[u]$ is the total number of occurrences of value u for a key guess g and its corresponding Hamming weight hypothetical leakage h .

4. The correct key guess g_{correct} is distinguished by applying a generic-emulating side-channel distinguisher to the estimated distributions $\text{pdf}_{g,h}[u]$.

Both the Chi^2 and MIA distinguishers can be used in combination with the **Scatter** transform in order to search for the correct key candidate.

The Chi^2 distinguisher is based on Pearson's \mathcal{X}^2 -test to perform a partition-based DPA [SGV08]. When successful, the partition based on the correct key guess should lead to the highest confidence level to reject the null hypothesis. In the **Scatter** context, it estimates how much a distribution differs from a general distribution (e.g., in our case study, the mean distribution of all 9 Hamming weight leakage distributions for a key guess g and a value u) — the correct key guess being expected to show the most significant difference. The Chi^2 value is computed according to the following formula:

$$\mathcal{X}_{g,h}^2 = \sum_{u=0}^{N_b-1} \frac{(\text{pdf}_{g,h}[u] - \frac{1}{9} \cdot \sum_{h'=0}^8 \text{pdf}_{g,h'}[u])^2}{\frac{1}{9} \cdot \sum_{h'=0}^8 \text{pdf}_{g,h'}[u]}. \quad (2)$$

For each key guess, there are 9 Chi^2 scores corresponding to 9 Hamming weights. The logarithm sum of all 9 scores is then used as the final score.

The MIA distinguisher was introduced by Gierlichs et al. [GBTP08]. It is based on estimating the mutual information between a hypothetical leakage model and the actual leakages. Under a correct partitioning (i.e., the correct key guess), it is expected that the largest mutual information should be observed for the correct key candidate to distinguish the correct key guess from the wrong ones. The MIA value is computed according to the following formula:

$$\text{MI}_g = \sum_{h=0}^8 \text{P}(Y = h) \cdot (\nabla_1(g, h) - \nabla_2(h)), \quad (3)$$

in which:

$$\text{P}(Y = h) = \frac{1}{n \cdot 9} \cdot \sum_{u=0}^{N_b-1} \text{Acc}_{g,h}[u], \quad (4)$$

with n the number of traces collected and:

$$\nabla_1(g, h) = \sum_{u=0}^{N_b-1} \text{pdf}_{g,h}[u] \cdot (\text{pdf}_{g,h}[u]), \quad (5)$$

$$\nabla_2(h) = \sum_{u=0}^{N_b-1} \left(\frac{1}{9} \cdot \sum_{h'=0}^8 \text{pdf}_{g,h'}[u] \right) \cdot \log \left(\frac{1}{9} \cdot \sum_{h'=0}^8 \text{pdf}_{g,h'}[u] \right). \quad (6)$$

2.2 On-the-fly linear regression

The use of linear regression for (profiled) side-channel attacks was introduced by Schindler et al. [SLP05]. It was then extended to non-profiled key-recovery attacks in [DPRS11]. We next denote this non-profiled extension as LRA.

Let us denote the leakage measurement as \mathbf{L} . The target m -bit intermediate value v (e.g., the S-box output in our case) is first decomposed according to some basis. In the following, we will use the usual (linear) basis made of the 8 bits of v ($v[m-1], v[m-2], \dots, v[0]$). LRA then simply tests the linear relation between the actual leakages and their approximation with this basis, thanks to the coefficient of determination R^2 . More precisely:

1. We first compute $(v_{\hat{g}}[m-1], v_{\hat{g}}[m-2], \dots, v_{\hat{g}}[0])$ for each key guess \hat{g} and each input plaintext & measurement $\mathbf{L}_i, i = 0, 1, \dots, n-1$.
2. We then estimate the linear regression model between the measurement \mathbf{L} and the following approximation:

$$\mathbf{L}_{\text{app}} = \beta_{\hat{g},0} + \beta_{\hat{g},1} \cdot v_{\hat{g}}[0] + \dots + \beta_{\hat{g},m} \cdot v_{\hat{g}}[m-1], \quad (7)$$

using ordinary least square method to estimate the parameter $\beta_{\hat{g},j}$.

3. We finally compute the coefficient of determination $R_{\hat{g}}^2$ for each key guess. The correct key guess g_{correct} is supposed to show the highest R^2 value.

2.3 Selection of parameters

The efficiency of the three aforementioned distinguishers is quite dependent on the good selection of their parameters: number of bins for the Chi² and MIA distinguishers, size of the basis for LRA. As already mentioned, our experiments are based on LRA with a linear 9-element basis (the eight S-box output bits and a constant), which is a standard choice for this distinguisher [SLP05]. For the Chi² and MIA distinguishers, choosing the optimal number of bins is usually tricky. We selected 9 and 25 bins in our experiments: 9 since it naturally corresponds to Hamming weight leakages, 25 to assess the impact of more bins. We note that this choice is expected to be slightly detrimental to the LRA distinguisher (since under a Hamming weight assumption, a 2-element basis with the Hamming weight of the S-box output should be even faster to estimate).

3 First-order experiments

We first investigate a simulated shuffled implementation, since this was the case study put forward in the COSADE 2018 paper on `Scatter`. We continue by targeting a real device of which the measurements are affected by a strong jitter, preventing the good alignment of the traces around the leaking part.

3.1 Setting #1: a simulated shuffled implementation

Shuffling is a widely-used side-channel countermeasure [HOM06,VMKS12]. Its main principle is to execute sensitive operations in a random order so that their leakages are spread over a multivariate distribution. As a result, each single point in time can correspond to the execution of various operations.

Shuffled implementations are the typical context in which `Scatter`'s multivariate transform was claimed to be a useful tool at COSADE 2018.

Implementation settings. The main parameter influencing the security of a shuffled implementation is the number of parallel operations which are randomized. We next consider a default size of 16 (corresponding to the AES case) and additionally experimented with a permutation of size 64, which could correspond to the execution of 48 dummy S-boxes. In our default setup, a single POI is leaking (corresponding to the target S-box execution) but we also considered a case with four POIs (which does not reflect a concrete AES implementation and was just aimed to understand the impact of a denser leakage in the `Scatter` window). Finally, we used a Signal-to-Noise Ratio (SNR) of 10, 1 and 0.1, reflecting low-noise, medium-noise and high-noise contexts [Man04]. The way we generated simulated traces is similar to the `Scatter` paper. For a window size d (i.e., the shuffling size in our experiments), we:

1. Choose the number of informative points (n_i),
2. Pick up their location in the d possible positions uniformly at random,
3. Put random leakages (of the same shape) in all the other points,
4. Add Gaussian noise to the entire trace based on the chosen SNR.

Our simulations focus on Hamming weight leakages for the first-round first S-box of an AES-128 encryption, namely $\text{HW}(\text{Sbox}(p[0] \oplus k[0]))$, where $p[0]$ and $k[0]$ correspond to the first bytes of the 16-byte AES input and key, respectively.

For each simulation setting, we estimated the Success Rate (SR) of the different attacks under investigation based on 100 independent experiments [SMY09].

Attack results. The results of our experiments are in Figure 2. We analyzed a wide range of parameters reflecting the various settings in which the `Scatter` transform could be exploited. As previously mentioned, we also evaluated this transform with both the Chi^2 and MIA distinguishers, using 9 and 25 bins. Those are systematically compared with the LRA distinguisher (9-element basis).

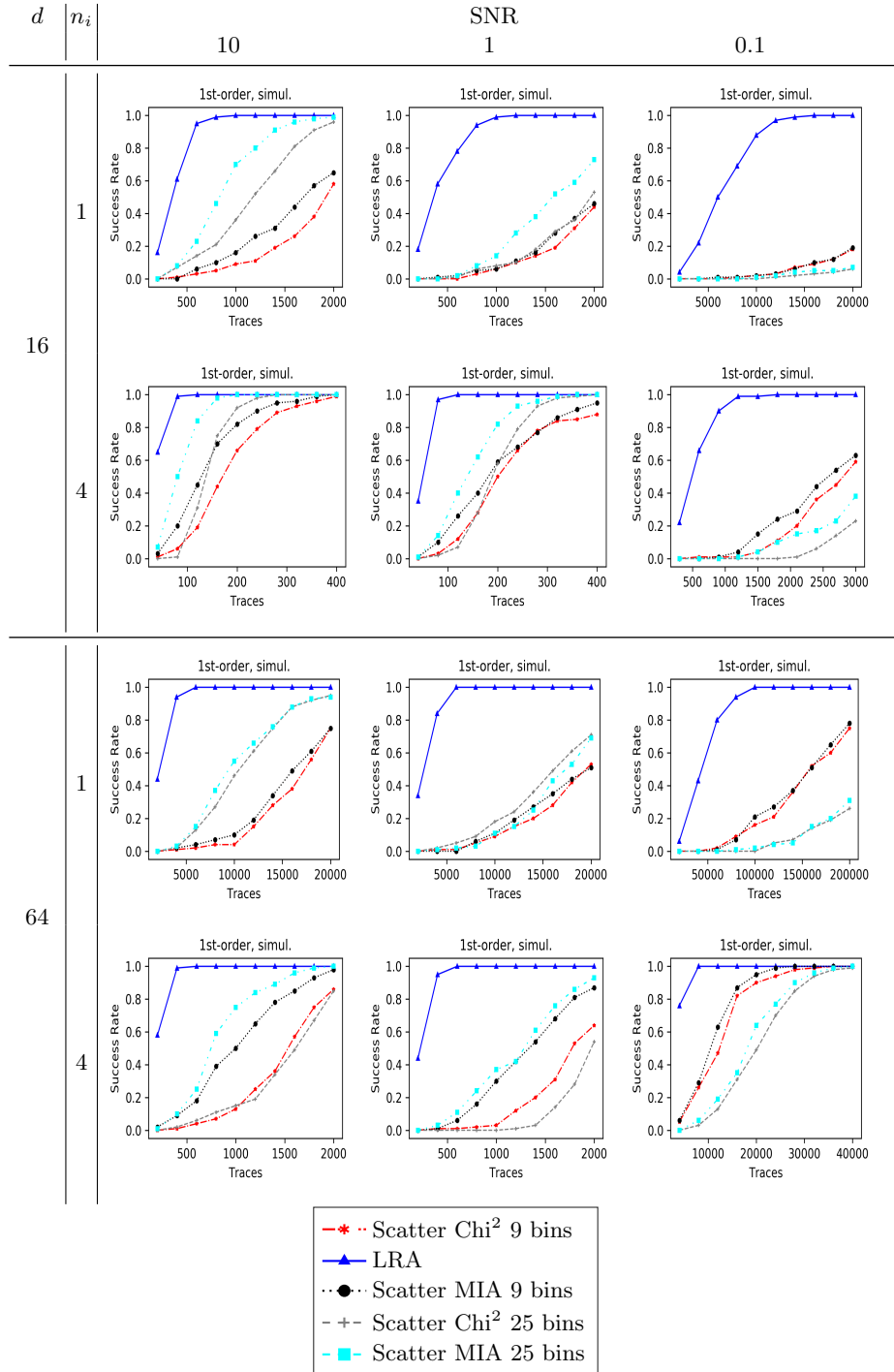


Fig. 2: Success rate on simulated shuffled implementations, with d the window size, n_i the number of POIs per window and various SNR values.

In general, these experiments carry the expected intuitions regarding the impact of our different parameters: decreasing the SNR makes the attacks more difficult (i.e., when moving from the left of the figure to the right of the figure); increasing d (i.e., the permutation size) makes the attacks more difficult (i.e., when moving from lines 1 and 2 to lines 3 and 4); increasing the number of POIs makes the attack easier (i.e., when moving from line 1 to line 2 and from line 3 to line 4). More specifically related to **Scatter**:

- LRA always outperforms **Scatter** with both the Chi^2 and MIA distinguishers, no matter the permutation size, noise level and number of POIs;
- The performance gap between LRA and **Scatter** is getting bigger as the attacks become more difficult (i.e., when the permutation size increases, the noise level increases and the number of POIs decreases);
- **Scatter** with the MIA distinguisher performs slightly better than **Scatter** with the Chi^2 distinguisher (which is in line with the COSADE 2018 results).
- As for the impact of the number of bins for **Scatter**: more bins generally show better results with lower noise and less bins generally works better with higher noise. The latter is in line with the findings of [GBTP08].

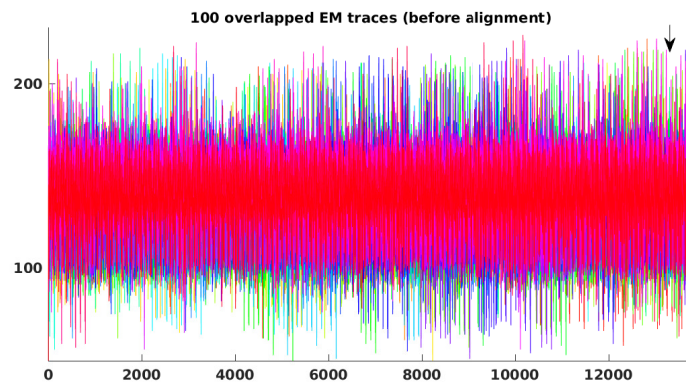
3.2 Setting #2: a concrete jittery implementation

We now extend our investigations to a real device, namely a software AES implementation using a secure processor. Due to the variable internal clock, the inserted random instructions during AES calculations, and the interrupts caused by the running Android-like operating system (OS), the measured traces are very jittery and we cannot really align the traces at the leaking time interval. We study how well **Scatter** can handle this challenging scenario.

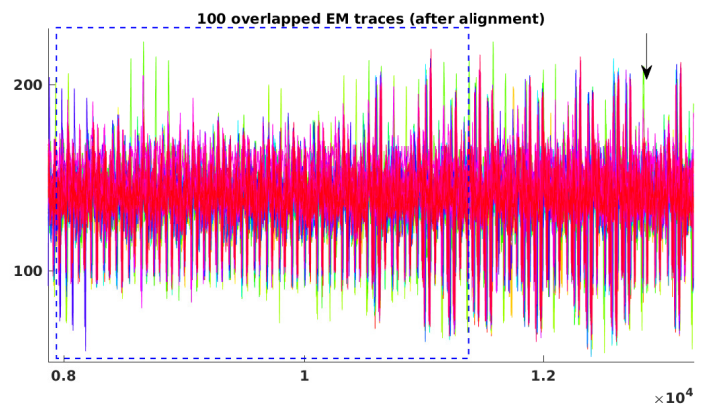
Implementation settings. The target secure processor is a Cortex-M4 chip running at 50 MHz next to a Qualcomm MSM8998 general processor which is running an Android-like OS. The secure processor is used for cryptography calculations and it communicates with the MSM8998 processor via UART (Universal Asynchronous Receiver/Transmitter) interface. The AES implementation is unprotected except for the random instructions inserted during the AES execution. Interrupts are additionally caused by the running Android-like OS and make the measured traces more noisy and hard-to-align. We measured 100,000 ElectroMagnetic (EM) traces on top of the secure processor using an EM probe, with a LeCroy Waverunner 620Zi oscilloscope, at a sampling rate of 5 GHz.

During the measurements, we triggered the oscilloscope at the end of the entire AES encryption command processing. The raw EM traces are noisy and hardly show distinct patterns that can be used for alignment, as can be seen in Figure 3(a). We therefore used a simple correlation-based pre-processing in order to better synchronize these EM traces, working as follows:

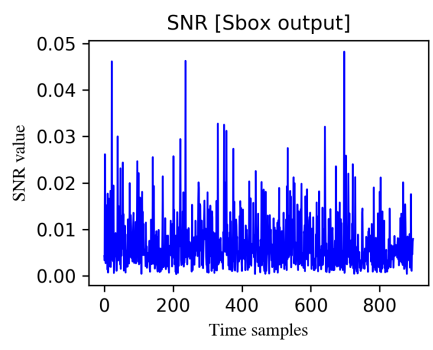
- Two intervals are chosen. First, a searching interval A that contains the operation to be synchronized is manually selected among all the traces. Next, a smaller reference interval B_q specific to each trace q is also chosen.



(a)



(b)



(c)

Fig. 3: 100 overlapped EM traces before alignment (a) and after alignment (b), and target S-box estimated SNR (c).

- For each trace, we find the portion to be synchronized by using the second window B_q to search over the whole interval A . The right portion is selected as the one having the maximum correlation with the reference interval. If the correlation is lower than a given threshold (chosen by the attacker/evaluator), the trace is assumed not good enough and discarded.

After performing such an alignment, we were able to determine where the AES computations occur by means of SEMA (Simple Electro Magnetic Analysis) and CEMA (Correlation Electro Magnetic Analysis), as shown in Figure 3(b).

Attack results. Our comparisons are based on 99,902 aligned EM traces focusing on the leaking part (the other traces were discarded). As a first note, none of the investigated distinguishers directly succeeded in recovering key bytes by exploiting the leakage in the time domain. We then applied a Fast Fourier Transform (FFT) in order to convert the traces into the frequency domain and to mitigate the impact of misalignment. After this pre-processing, LRA was able to recover all 16 key bytes of an AES state, but Scatter was not (neither with the Chi^2 nor with the MIA distinguishers). These results are illustrated in Figure 4 where the success rate is estimated based on 100 independent experiments.

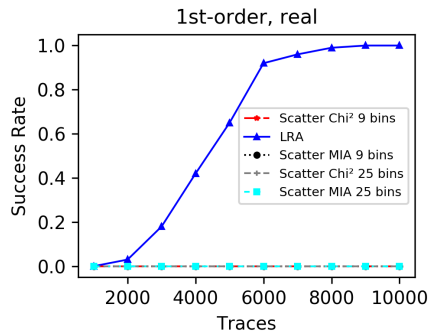


Fig. 4: Attacks against a real implementation with strong jitter.

4 Higher-order Scatter

4.1 The need of a combination function

We start with a simple negative result highlighting the need to generalize the Scatter transform before application to higher-order side-channel attacks. For this purpose, let us imagine that the two “clock cycles” represented at the top of Figure 1 correspond to the two shares of a masked sensitive variable x . Let us further consider that this sensitive variable is one bit and can be written as $x = x_1 \oplus x_2$ with x_1 picked up uniformly at random. Let us finally assume that the

adversary can obtain the leakage of the two shares x_1 and x_2 , denoted as l_1 and l_2 : under Hamming weight leakages, we have $l_1 = x_1$ and $l_2 = x_2$, meaning that the adversary can directly observe the shares. In this context, 1st-order probing security is guaranteed because the observation of either l_1 or l_2 does not reveal anything about x . By contrast, a second-order probing attack is trivial since $l_1 \oplus l_2 = x$. More interestingly, a second-order statistical attack is also successful since the distribution of (l_1, l_2) when $x = 0$ is $(0,0)$ with probability $\frac{1}{2}$ and $(1,1)$ with probability $\frac{1}{2}$, while this distribution becomes $(0,1)$ with probability $\frac{1}{2}$ and $(1,0)$ with probability $\frac{1}{2}$ when $x = 1$ (which has a different variance).

If we now apply the **Scatter** transform, each bivariate trace (l_1, l_2) is split into two univariate traces l_1 and l_2 , and histograms are built from these two traces. As a result, the two traces $(0,0)$ and $(1,1)$ that correspond to the case $x = 0$ are turned into four traces 0, 0, 1, 1. Their histogram gives 0 with probability $\frac{1}{2}$ and 1 probability $\frac{1}{2}$. Similarly, the two traces $(0,1)$ and $(1,0)$ that correspond to the case $x = 1$ are turned into four traces 0, 1, 1, 0, leading to exactly the same histogram. So directly applying the first-order **Scatter** transform to a masked implementation cancels the differences between these distributions that can be used to mount a successful second-order attack. The same example generalizes to any number of shares and probing/statistical security order.

As usual in side-channel analysis, the solution to prevent this issue is to generalize the transform to higher-orders. There are essentially two solutions for this purpose: either one considers all the pairs (and triples, quadruples, ...) of samples and applies a multivariate (e.g., Chi^2 or MIA) distinguisher to it, or one uses a combination function (e.g., the normalized product in the context of Hamming weight leakages [PRB09,SVO⁺10]) and applies a (univariate in the case of LRA or multivariate in the case of **Scatter**) distinguisher to its output. As discussed for example in [BGP⁺11,MRSS18], directly considering all the pairs (and triples, quadruples, ...) of samples and applying a multivariate distinguisher is usually more expensive, due to the curse of dimensionality when estimating multivariate distributions in a non-parametric manner. Our experiments showed the same trend and so do the experiments of Thiebauld et al. in [TVW19].

As a result, we next consider higher-order attacks based on a combination function illustrated at the bottom of Figure 1. That is, in the second-order case we will concretely investigate, we start by extending the original d -sample window to a d^2 -sample window containing all the normalized product samples and then apply the **Scatter** transform combined with the Chi^2 or MIA distinguishers, or LRA. Note that this solution suffers from the usual drawback that the cost of finding the POIs in the traces grows exponentially in the number of shares (and exactly the same would hold for the first aforementioned solution where a multivariate distinguisher is applied to all the tuples of samples).³

³ In the report on the second-order application of **Scatter** [TVW19], an optional projection of the histogram traces is considered. In our experiments, this projection (just as the direct bivariate attacks) did not exhibit any improvement. This seems natural in a simulated setting where the normalized product combination function is known to be optimal [PRB09]. So we next ignore this optional projection.

4.2 Second-order simulated experiments

Implementation settings. We now complete the previous first-order experiments with second-order simulations. We consider a 2-share implementation where the adversary obtains the two Hamming weights corresponding to the two shares of a target S-box’s output. Based on the previous observation that `Scatter` tends to behave better in less challenging scenarios (and in order to limit the cost of our simulations, which increases with the security levels), we selected the following parameters: a permutation of size $d = 4$ with a single POI (i.e., $n_i = 1$) and a SNR of 10, 1 and 0.25. For completeness, we also report results with $d = 16$, 4 POIs (i.e., $n_i = 4$) and a SNR of 10 and 1. We focused on `Scatter` with the MIA distinguisher that was the best in class for our first-order experiments. (This is also similar to what has been done in [TVW19]).

Attack results. The results of our second-order experiments based on a normalized product combination function are displayed in Figure 5.

Observations are essentially similar to the first-order case: again, LRA systematically outperforms attacks based on `Scatter` and the more “challenging” the implementation (e.g., the lower the SNR), the bigger the gap. This can be explained by the fact that the product combining pre-processing generates traces that can be exploited in a very similar way as an unprotected implementation (up to the noise level that is amplified by the product operation).

5 Conclusion

Exploiting the side-channel leakages of an implementation protected with jitter, shuffling and/or masking is a challenging problem. The `Scatter` transform was introduced at COSADE 2018 as a novel transform aimed to make such attacks more efficient. In this work, we nailed down its specificity and compare it with a natural competitor for unprofiled multivariate side-channel analysis.

It turns out a standard on-the-fly application of linear regression leads to more efficient attacks in many practically-relevant contexts, including all the ones initially put forward by the `Scatter` authors. It is therefore an interesting open problem to determine whether this transform can sometimes be a useful ingredient in a side-channel security evaluation. The exhibition of a simulated case where such a gain can be observed appears as a natural next to answer this question. Without such a case, evaluators are left with the conclusion that it currently does not improve over existing solutions.

Acknowledgments. François-Xavier Standaert is a senior research associate of the Belgian Fund for Scientific Research (F.R.S.-FNRS). This work has been funded in parts by the ERC project SWORD (Grant Number 724725), the H2020 project REASSURE and the UCLouvain ARC project NANOSEC.

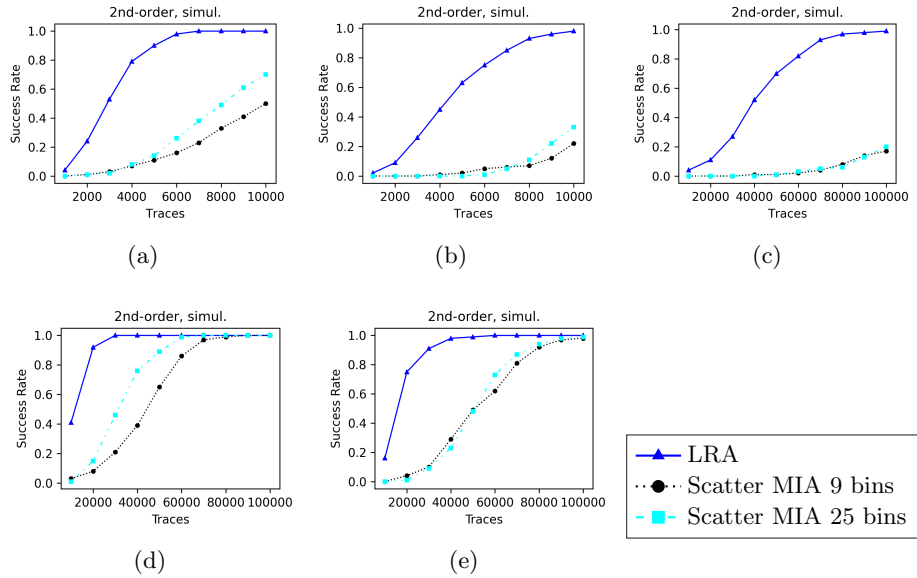


Fig. 5: Simulated shuffled & masked implementation: (a) $d = 4$, $n_i = 1$, $\text{SNR}=10$; (b) $d = 4$, $n_i = 1$, $\text{SNR}=1$; (c) $d = 4$, $n_i = 1$, $\text{SNR}=0.25$; (d) $d = 16$, $n_i = 4$, $\text{SNR}=10$; (e) $d = 16$, $n_i = 4$, $\text{SNR}=1$.

References

- BCO04. Eric Brier, Christophe Clavier, and Francis Olivier. Correlation power analysis with a leakage model. In *CHES*, volume 3156 of *Lecture Notes in Computer Science*, pages 16–29. Springer, 2004.
- BGP⁺11. Lejla Batina, Benedikt Gierlich, Emmanuel Prouff, Matthieu Rivain, François-Xavier Standaert, and Nicolas Veyrat-Charvillon. Mutual information analysis: a comprehensive study. *J. Cryptology*, 24(2):269–291, 2011.
- CJRR99. Suresh Chari, Charanjit S. Jutla, Josyula R. Rao, and Pankaj Rohatgi. Towards sound approaches to counteract power-analysis attacks. In *CRYPTO*, volume 1666 of *Lecture Notes in Computer Science*, pages 398–412. Springer, 1999.
- CK10. Jean-Sébastien Coron and Ilya Kizhvatov. Analysis and improvement of the random delay countermeasure of CHES 2009. In *CHES*, volume 6225 of *Lecture Notes in Computer Science*, pages 95–109. Springer, 2010.
- CRR02. Suresh Chari, Josyula R. Rao, and Pankaj Rohatgi. Template attacks. In *CHES*, volume 2523 of *Lecture Notes in Computer Science*, pages 13–28. Springer, 2002.
- DPRS11. Julien Doget, Emmanuel Prouff, Matthieu Rivain, and François-Xavier Standaert. Univariate side channel attacks and leakage modeling. *J. Cryptographic Engineering*, 1(2):123–144, 2011.
- GBTP08. Benedikt Gierlich, Lejla Batina, Pim Tuyls, and Bart Preneel. Mutual information analysis. In *CHES*, volume 5154 of *Lecture Notes in Computer Science*, pages 426–442. Springer, 2008.

- GP99. Louis Goubin and Jacques Patarin. DES and differential power analysis (the "duplication" method). In *CHES*, volume 1717 of *Lecture Notes in Computer Science*, pages 158–172. Springer, 1999.
- HOM06. Christoph Herbst, Elisabeth Oswald, and Stefan Mangard. An AES smart card implementation resistant to power analysis attacks. In *ACNS*, volume 3989 of *Lecture Notes in Computer Science*, pages 239–252, 2006.
- Man04. Stefan Mangard. Hardware countermeasures against DPA ? A statistical analysis of their effectiveness. In *CT-RSA*, volume 2964 of *Lecture Notes in Computer Science*, pages 222–235. Springer, 2004.
- MOP07. Stefan Mangard, Elisabeth Oswald, and Thomas Popp. *Power analysis attacks - revealing the secrets of smart cards*. Springer, 2007.
- MRSS18. Amir Moradi, Bastian Richter, Tobias Schneider, and François-Xavier Standaert. Leakage detection with the x2-test. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2018(1):209–237, 2018.
- PRB09. Emmanuel Prouff, Matthieu Rivain, and Régis Bevan. Statistical analysis of second order differential power analysis. *IEEE Trans. Computers*, 58(6):799–811, 2009.
- SGV08. François-Xavier Standaert, Benedikt Gierlichs, and Ingrid Verbauwhede. Partition vs. comparison side-channel distinguishers: An empirical evaluation of statistical tests for univariate side-channel attacks against two unprotected CMOS devices. In *ICISC*, volume 5461 of *Lecture Notes in Computer Science*, pages 253–267. Springer, 2008.
- SLP05. Werner Schindler, Kerstin Lemke, and Christof Paar. A stochastic model for differential side channel cryptanalysis. In *CHES*, volume 3659 of *Lecture Notes in Computer Science*, pages 30–46. Springer, 2005.
- SMY09. François-Xavier Standaert, Tal Malkin, and Moti Yung. A unified framework for the analysis of side-channel key recovery attacks. In *EUROCRYPT*, volume 5479 of *Lecture Notes in Computer Science*, pages 443–461. Springer, 2009.
- SVO⁺10. François-Xavier Standaert, Nicolas Veyrat-Charvillon, Elisabeth Oswald, Benedikt Gierlichs, Marcel Medwed, Markus Kasper, and Stefan Mangard. The world is not enough: Another look on second-order DPA. In *ASIACRYPT*, volume 6477 of *Lecture Notes in Computer Science*, pages 112–129. Springer, 2010.
- TGWC18. Hugues Thiebauld, Georges Gagnerot, Antoine Wurcker, and Christophe Clavier. SCATTER: A new dimension in side-channel. In *COSADE*, volume 10815 of *Lecture Notes in Computer Science*, pages 135–152. Springer, 2018.
- TVW19. Hugues Thiebauld, Aurélien Vasselle, and Antoine Wurcker. Second-order scatter attack. *IACR Cryptology ePrint Archive*, 2019:345, 2019.
- VMKS12. Nicolas Veyrat-Charvillon, Marcel Medwed, Stéphanie Kerckhof, and François-Xavier Standaert. Shuffling against side-channel attacks: A comprehensive study with cautionary note. In *ASIACRYPT*, volume 7658 of *Lecture Notes in Computer Science*, pages 740–757. Springer, 2012.
- WOS14. Carolyn Whitnall, Elisabeth Oswald, and François-Xavier Standaert. The myth of generic dpa...and the magic of learning. In *CT-RSA*, volume 8366 of *Lecture Notes in Computer Science*, pages 183–205. Springer, 2014.