Getting the Most Out of Leakage Detection

Statistical tools and measurement setups hand in hand

Santos Merino Del Pozo and François-Xavier Standaert.

ICTEAM/ELEN/Crypto Group, Université catholique de Louvain, Belgium.

Abstract. In this work, we provide a concrete investigation of the gains that can be obtained by combining good measurement setups and efficient leakage detection tests to speed up evaluation times. For this purpose, we first analyze the quality of various measurement setups. Then, we highlight the positive impact of a recent proposal for efficient leakage detection, based on the analysis of a (few) pair(s) of plaintexts. Finally, we show that the combination of our best setups and detection tools allows detecting leakages for a noisy threshold implementation of the block cipher PRESENT after an intensive measurement phase, while either worse setups or less efficient detection tests would not succeed in detecting these leakages. Overall, our results show that a combination of good setups and fast leakage detection can turn security evaluation times from days to hours (for first-order secure implementations) and even from weeks to days (for higher-order secure implementations).

1 Introduction

State-of-the-art. The concrete evaluation of cryptographic hardware and software against *side-channel analysis* (SCA) attacks is a complex and expensive process. This is especially true in the case of implementations protected with (combinations of) countermeasures, for which cryptographic engineers aim to ensure that the leakages are noisy and carry little sensitive information. In this context, minimizing the evaluation time (which we assume proportional to the evaluation cost) generally benefits from a combination of three ingredients:

1. Obtaining good measurements, with high signal and minimum noise.
2. Simplifying the evaluation goals, e.g. from key recovery to simpler detections.
3. Optimizing the distinguishers, in particular their data and time complexity.

Despite its practical relevance, the first problem is rarely the primary focus in the literature. Yet, several recent papers have highlighted the significant impact of good setups for the evaluation of masked implementations [12], especially when it comes to devices running at higher frequencies [2] or for the investigation of static leakages [11]. The second problem typically illustrates the tradeoff between the evaluation time and the accuracy of the conclusions in SCA. That is, the ultimate goal of an evaluator is to obtain accurate evaluations of the worst-case security

level of an implementation. Yet, in view of the higher cost of such worst-case analyzes, an increasingly popular approach in SCA evaluations is to start with simpler leakage detection tests, i.e., the *test vector leakage assessment* (TVLA) methodology introduced by Goodwill et al. [9]. In this case, the goal is not to estimate a key recovery success rate, but to detect whether the leakages depend on the data manipulated by the device: an arguably easier task. Following, such leakage detections can become the sole goal of the evaluation (which is then limited to qualitative conclusions: see [18] for a recent discussion), or serve as a preliminary step for more advanced (quantitative) investigations. Eventually, the last problem has been the daily bread of researchers in SCA for the last fifteen years, with various proposals of distinguishers optimized for efficiency or genericity, in profiled or non-profiled attack settings, e.g. [5, 6].

Our contributions. In this paper, we investigate the extent to which a combination of good measurement setups and state-of-the-art leakage detection tools can speed up the evaluation time for cryptographic implementations.

For this purpose, we start by comparing various measurement setups with TVLA, taking advantage of the optimizations by Schneider et al. [17]. Our experiments allow us to exhibit the significant impact of bad measurements (i.e. with limited signal and a lot of noise), which can lead to incorrectly conclude about the (in)security of a threshold implementation (TI) of the PRESENT block cipher.

Next, we study the gains that can be obtained by exploiting a more recent proposal of TVLA, based on a partition of the measurements in two classes corresponding to two fixed plaintexts, next denoted as a *fixed vs. fixed* TVLA [8], rather than a partition of the measurements in two classes where one class corresponds to a fixed plaintext and the other class to random plaintexts, as originally proposed in [9] and next denoted as the *fixed vs. random* TVLA. We show that the fixed vs. fixed test allows a consistent reduction of the data complexity needed for successful detections, with different factors depending on the signal and noise of the target implementation.

Eventually, we pushed the investigation of our first-order TI in a very high noise regime, typically corresponding to signal-to-noise ratios below 0.01, where TIs are supposed to lead to high security levels. This experiment allows us to exhibit a concrete case where 100 million measurements (corresponding to one day of sampling) were not sufficient to spot any leakage with the fixed vs. random test, whereas the fixed vs. fixed test leads to clear detections. It also leads to an interesting change of the most informative statistical moment in the leakage traces, from the third-order moment in low noise contexts to the second-order one in high noise contexts, as predicted by theory [7].

Overall, these examples highlight that as the security level of an implementation increases, the impact of the gains due to a good measurement setup and selection of optimized statistical tools is magnified. Put very simply, reducing the evaluation time from 100 seconds to 1 second is a gain by a much larger factor than reducing the evaluation time from 5 days to 1 day. Yet the second improvement is much more relevant for evaluation laboratories for which time

and cost are absolute (rather than relative) metrics. The sound combination of state-of-the-art tools in this work therefore contribute to this important goal of minimizing evaluation time (and cost).

2 Preliminaries

In order to make the paper self-contained, in this section we introduce notations and provide background information about the statistical tools and measurement setups considered throughout the paper.

2.1 Notations

We use capital letters for random variables and lower-case letters for their realization. We denote vectors and matrices with bold notations and sets with calligraphic ones.

We refer to the targeted cryptographic device storing a private key as the *device under test* (DUT). In order to assess the vulnerability of the DUT to (higher-order) SCA attacks, we consider an evaluator with full knowledge of the implementation and the ability to measure the current across the DUT. We denote by $\mathcal{T}$ the set comprising m so-called SCA traces $\boldsymbol{t}_{i\in\{1,\ldots,m\}}$, each of them made of n time samples $t_i^{j\in\{1,\ldots,n\}}$, collected while the DUT is fed with the associated plaintexts $\boldsymbol{x}_{i\in\{1,\ldots,m\}}$.

2.2 Leakage Assessment Methodology

The two versions of TVLA proposed in [9] (i.e., *specific* and *non-specific*) aim to detect the existence of (possibly) exploitable leakages at a certain statistical moment on the DUT. Following the guidelines in [17], since our DUT is equipped with a masking countermeasure (i.e., a first-order TI), we start our practical investigations by considering the non-specific fixed vs. random TVLA. Though this tool does not bring information about the hardness of mounting successful attacks against the DUT, it comes in handy when examining the existence of leakages at higher-order moments (e.g., during prototyping) where, in comparison with higher-order attacks, the number of required measurements can be significantly reduced. To this end, the evaluator records two sets of measurements $\mathcal{T}_0$ and $\mathcal{T}_1$, respectively associated to fixed and randomly generated inputs (i.e., fixed vs. random), while keeping the device key constant. The test does not require any prior knowledge of the DUT or assumption about how it leaks (i.e., non-specific). More recently, with the aim of speeding up the detection of leakages, the so-called non-specific fixed vs. fixed TVLA has been proposed by Durvaux et al. [8] as a tweak of the aforementioned fixed vs. random version. To that end, $\mathcal{T}_0$ and $\mathcal{T}_1$ are now associated to two fixed plaintexts (i.e., fixed vs. fixed), an approach that has been shown to improve the signal, remove the algorithmic noise intrinsic to the set of SCA traces associated with random

plaintexts, and thereby reduce the measurement complexity of security evaluations. We should note that in order to guarantee a non-deterministic internal state of the DUT at the beginning of a new measurement, and therefore to avoid false-positives, in both methodologies SCA traces must be collected in a randomly-interleaved fashion. Then the two sets of traces are compared by computing the Welch's (two-tailed) t-test in a univariate fashion (i.e., individually at each time sample $j \in \{1, \ldots, n\}$):

$$\mathrm{t} = \frac{\mu(\mathcal{T}_0) - \mu(\mathcal{T}_1)}{\sqrt{\frac{\sigma^2(\mathcal{T}_0)}{|\mathcal{T}_0|} + \frac{\sigma^2(\mathcal{T}_1)}{|\mathcal{T}_1|}}},$$

where $|.|$ is the sample size. The result determines if the samples have been drawn from the same population, i.e., the *null hypothesis*. A typical significance threshold to reject the null hypothesis in SCA evaluations, and therefore to pinpoint the existence of first-order leakages in the DUT, is $|\mathrm{t}| \geq 4.5$. To enable the detection of leakages at higher orders, SCA traces need to be preprocessed accordingly (we omit the details but refer an interested reader to [17]).

2.3 Measurement Setups

In the following sections, the platform employed to conduct our practical experiments is a SAKURA-G [1] featuring two Spartan-6 FPGAs (*target* and *control*) built in a 45 nm technology and which has been especially designed for research on hardware security, e.g., SCA attacks. The board provides three built-in attack points (i.e., the two heads of a resistor and the output of an embedded amplifier) to measure the voltage drop over the 1Ω shunt resistor placed in the Vdd path of the target FPGA that, by means of the corresponding voltage regulator, was supplied at 1.2 V. Since running the DUT at high frequencies, e.g., 24 MHz, is known to harden the detection (and exploitation) of SCA leakages (i.e., due to the intrinsic windowing effect), we clocked the target FPGA at 3 MHz.

In all the experiments, SCA traces were collected by means of a Teledyne Lecroy HRO66Zi WaveRunner 12-bit digital oscilloscope (DSO) at a sampling rate of 500 MS/s and a bandwidth limit of 20 MHz to reduce the environmental noise. Besides, a passive probe (i.e., a SMA-to-BNC coaxial cable) that avoids the additional noise induced by, e.g., active components in differential probes, was used in all the experiments as well. Since practical investigations in this work involved the analysis of millions of SCA leakages, our acquisition framework was designed following the guidelines in [17], allowing us to perform millions of measurements per hour by exploiting the sequence mode of the employed DSO. It should be noted that the UART communication channel between the PC and the control FPGA also contributes to increase the noise level in the SCA traces. Therefore, we made sure that before triggering the DSO, the UART channel was closed and remained in such state until the completion of the measurement.

Typically, when measuring the dynamic power consumption, it is advantageous to remove the DC shift that, from the evaluator's perspective, can be seen

Table 1: Summary of the different setups considered in this work.

	Coupling	Amplifier	Low Pass	High Pass	Pass Band
setup 1	AC 1 MΩ	✗	✗	✗	N/A
setup 2	DC 50 Ω	✓	✗	✗	N/A
setup 3	DC 50 Ω	✓	✓	✓	0.1 to 5MHz
setup 4	DC 50 Ω	✓	✓	✓	1.2 to 5MHz

as an additional source of noise. To this end, the most straightforward way is to perform current measurements using the AC coupling mode of the DSO (from now on termed *setup 1*). Further, quantization noise due to a low peak-to-peak signal amplitude can exacerbate the measurement complexity of TVLA. To cope with it, amplifiers, such as the ADI AD8000 embedded on the SAKURA-G board (in the following called *setup 2*), can be employed to increase the amplitude of the signal. In fact, security evaluators may use more elaborate setups featuring a combination of AC amplifiers, DC blockers and/or hardware filters to maximize the signal (e.g., when targeting ultra low-power designs) and reduce the noise (e.g, for masking schemes whose security level relies on having sufficiently noisy leakages). To assess their benefits (but also the repercussions of capacitive elements), we considered two more setups. In the so-called *setup 3*, we employed an AC amplifier (i.e., `ZFL-1000LN+` from Mini-Circuits), a DC to 5 MHz filter (i.e., `SLP-5+` from Mini-Circuits) and a DC blocker (i.e., `BLK-89-S+` from Mini-Circuits), whereas in *setup 4* the DC blocker was replaced by a 1.2 to 800 MHz filter (i.e., `ZFHP-1R2+` from Mini-Circuits) that together with the `SLP-5+` formed a band pass filter around 3 MHz (the clock frequency of the DUT). A high-level overview of all the aforementioned setups is given in Table 1.

3 Case Studies

After describing the tools employed in our analysis, in this section we provide an overview of the two (hardware-oriented) countermeasures deployed in our designs. More precisely, we start by describing a fully-serialized architecture of a first-order TI of PRESENT. Next we detail the Gaussian noise engine that we implemented on our target FPGA.

3.1 Threshold Implementations

In this work we considered the first-order TI technique introduced in [15], which was designed to prevent any first-order leakage even in a glitchy hardware implementation. Following the principles of multi-party computation and secret sharing, the sensitive variables and functions are implemented using at least $s \geq d + 1$ shares, where d is the algebraic degree of the targeted function. As in any other (e.g., Boolean) masking scheme, the intermediate value x is represented using a vector of s shares $\boldsymbol{x} = (x_1, \ldots, x_s)$ such that $x = \bigoplus_{i=1}^{s} x_i$. While

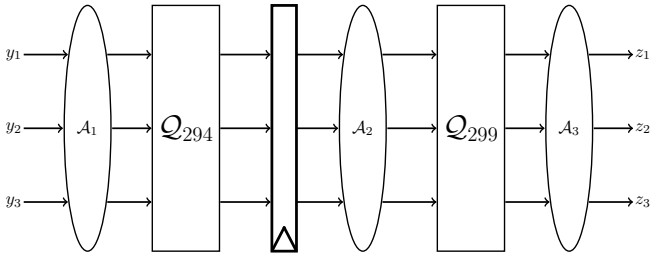


Fig. 1: Uniform first-order TI of the PRESENT Sbox in [14]

a linear function can be easily applied on each share with s instances in parallel, the implementation of non-linear functions is not a trivial task. TI implements a target non-linear function f as a vector of component functions $\boldsymbol{f} = (f_1, \ldots, f_s)$ such that every component function f_i, where $i \in \{1, \ldots, s\}$, is independent of at least one input share. Such a property is referred as the *non-completeness* property and it is the main contribution of [15]. When the sharing is correct, so-called *correctness* property, and the input shares are uniformly distributed, so-called *uniformity* property (and which indeed are standard properties in masking schemes), then the non-completeness property provides probable security against first-order SCA (even in the presence of glitches). It is noteworthy that, if the outputs of the component functions are used as inputs in next parts of the implementation, which is usually the case in symmetric-key algorithms, then the uniformity of the shared functions and their outputs must be carefully examined. Whenever this property is not satisfied, different techniques to repair the problem have already been proposed in the literature (see e.g., [13, 16]).

First-order TI of PRESENT-80. For our practical evaluations, we implemented a serialized architecture of PRESENT-80, and more concretely, *profile 2* as given in [16], where the 80-bit key is not represented in a shared form. First, the plaintext (resp. the secret key) is loaded in parallel mode into the corresponding state (resp. key) register which is made of 16 (resp. 20) 4-bit wide registers, and that also behaves as a shift register. During the 16 first clock cycles within a encryption round, the state and key registers provide the shared Sbox with the corresponding 4-bit chunks (so-called nibbles). Eventually, when the 16 Sbox computations are done, the `PLayer` and Key Schedule are performed during the last clock cycle. Following the minimum settings of a first-order TI, the datapath is represented using a 3-share Boolean masking, so all state (i.e., shift) registers and `PLayer` instances have to be tripled.

For the TI representation of the PRESENT Sbox $S(x)$, we exploited the decomposition proposed by Moradi et al. [14]. In their work, the Sbox is decomposed into two quadratic bijections $\mathcal{Q}_{294} \times \mathcal{Q}_{299}$ such that $S(x) = \mathcal{A}_3 \circ \mathcal{Q}_{299} \circ \mathcal{A}_2 \circ \mathcal{Q}_{294} \circ \mathcal{A}_1$ where $\mathcal{A}_1$, $\mathcal{A}_2$ and $\mathcal{A}_3$ are affine functions. As it can be seen in Figure 1, where we provide a graphical representation of the resulting shared Sbox, three intermediate registers (i.e., one per share) must be placed between

the component functions of the two quadratic bijections. By doing so, it is possible to disallow the propagation of glitches, which is required to ensure that the non-completeness property is satisfied. As a result, each Sbox lookup now takes two clock cycles. For more details we refer the interested reader to [14].

We used Xilinx ISE version 14.7 for design synthesis, implementation and configuration of the board. Following the recommendations in [3] to satisfy the non-completeness property, we used the `KEEP HIERARCHY` constraint when generating the bitstream of the crypto module. Note that this is needed to make sure that the assumption of component functions leaking independently is not violated (which might lead to undesired first-order leakages).

3.2 Gaussian Noise Engine

In [10], the authors investigated different FPGA-dedicated techniques to achieve maximum levels of noise in SCA traces (i.e., to hide data-dependent leakages) by configuring unused available logic. Xilinx FPGAs contain n-to-m Look-Up Tables (LUTs) that, besides being used as a Boolean function generators, can also be configured as 16-bit shift registers. This so-called Shift Register LUT (SRL) mode is exploited by the authors of [10] to create r cycling registers made of s LUTs in SRL mode that are initialized with the pattern `01...0101`. The `enable` signal of each cycling register is driven by a PRNG, in our case, a LFSR with enough period to record up to 100 million traces. Only when both the `clock` and `enable` signals are high, the power consumption will increase due to the additional bit flips in the registers. The r and s parameters are used by the designer to set respectively the variance and amplitude of the noise. Concretely, we implemented a noise engine made up of $r = 16$ cycling registers, each of them consisting of $s = 100$ LUTs in SRL mode. Since this module did not have an output, we prevented the synthesizer from removing this unconnected component by using `SAVE NET FLAG` and `KEEP` constraints. Further, in order to not introduce algorithmic noise in the measurements, the PRNG (needed for mask generation and TVLA) was implemented on the control FPGA as the realization of AES-128 in CTR mode.

4 Comparing setups with CRI's Fixed vs. Random TVLA

In this section we evaluate the ability of the aforementioned measurement setups (see Section 2.3) to ease the detection of (higher-order) SCA leakages.

In the last years, the non-specific fixed vs. random TVLA has emerged as a very popular technique for the SCA evaluation of cryptographic devices. For this reason, and because recent academic works had used it to assess the security level of higher-order TIs [4], we based our preliminary investigations on this technique.

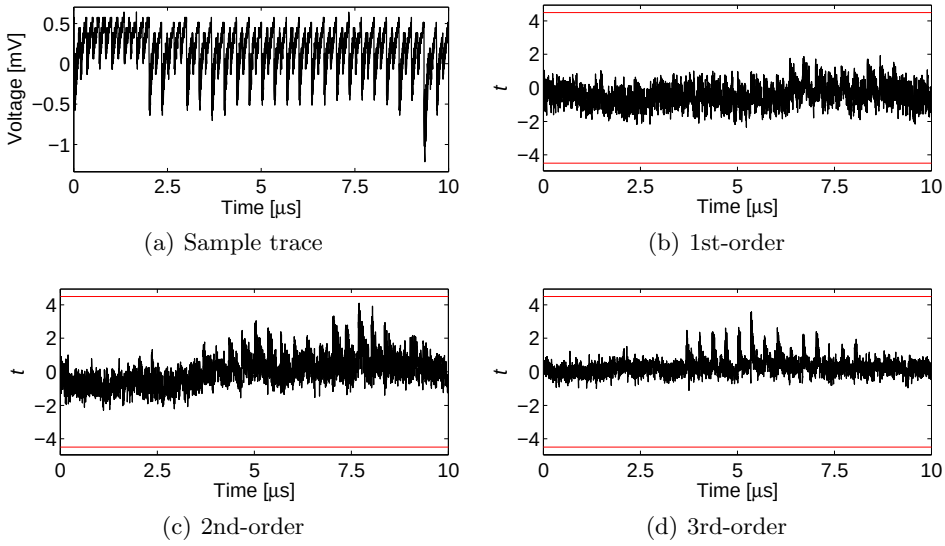
As explained in the previous section, our DUT features a fully-serialized architecture with small combinatorial circuits, and so with negligible algorithmic noise. Therefore a relatively small number of measurements was expected to suffice for our purposes, so we performed the analysis up to third-orders using

a set of 1 million SCA traces collected with fixed and random plaintexts in arbitrarily interleaved order. Note that, because we controlled the PRNG, we systematically repeated this process for each setup.

A sample power trace (comprising the first encryption round of the targeted design) recorded with *setup 1* is shown by Figure 2(a). In this setting, even limiting the bandwidth of the DSO to 20 MHz and using its maximum vertical accuracy (i.e., 1mV/div), the traces were very noisy due to the small amplitude of the signal. First, in order to verify the correct behavior of the measurement framework and the DUT, we turned the PRNGs (in charge of mask generation) off. As illustrated by Figure 15(a) in Appendix A, TVLA reported clear first-order leakages using 10 000 traces. As expected, when masks were enabled there was no easy to detect first-order leakage using up to 1 million traces, see Figure 2(b). Further, extending the analysis to second- (security in combinational logic) and third-orders (security in the memory elements) showed that such a number of measurements was not high enough to detect second- (Figure 2(c)) and third-order leakages (Figure 2(d)). By using traces collected from the amplified output provided by the SAKURA-G board (i.e., *setup 2*), and so with less quantization noise due to the higher peak-to-peak signal (see Figure 3(a)), made the detection of third-order leakages feasible (Figure 3(d)). Yet, second-order ones still remained undetectable (Figure 3(c)). Despite leading to a greater signal amplitude, similar results were obtained by considering *setup 3* (see Figure 4). The strong windowing effect induced by such setup is obvious. This is due to the DC blocker and the AC amplifier that, according to the authors of [12], makes consecutive power peaks overlap. Hence, such a behavior was expected for *setup 4* as well. As we can see in Figure 5(a), the inclusion of the high pass filter changed the polarity of the signal. Interestingly, in this case the existence of third-order leakages was pinpointed with a greater level of confidence than before (see Figure 5(d)). Moreover, and despite being featuring and AC amplifier, the aforementioned windowing effect became negligible, as shown by Figure 15(d) in Appendix A.

Figure 6 summarizes the results of these experiments. Figure 6(a) shows that, due to the register-oriented architecture of the DUT, the non-specific fixed vs. random TVLA cannot spot second-order leakages with up to 1 million SCA traces with any of the setups. On the other hand, Figure 6(b) highlights the importance of having signals with enough peak-to-peak amplitude, as shown by the results of *setup 1* where the quantization noise led to unsuccessful results. To conclude with this section, we should note that in comparison with *setup 2* and *3*, *setup 4* reduced by a factor of ≈ 3 the measurement complexity for detecting third-order leakages.

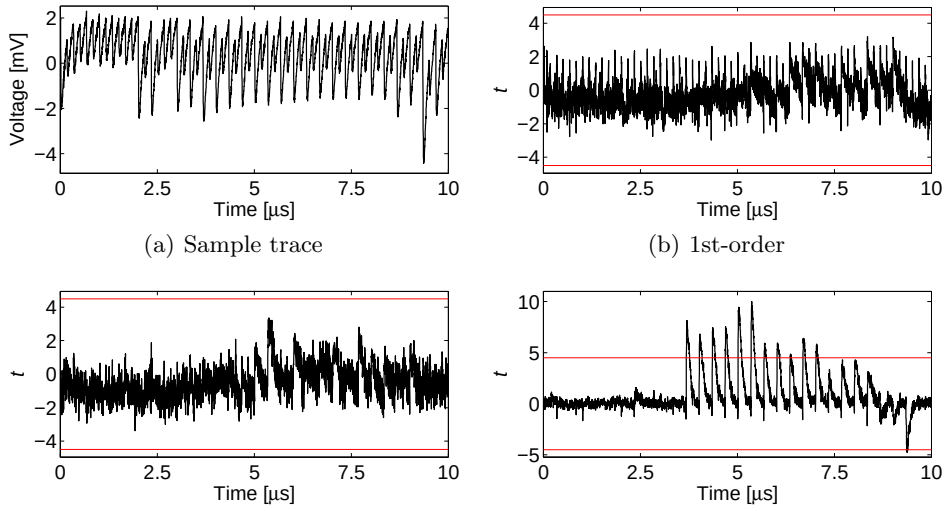
As a side-note, we finally mention that preprocessing the traces with digital filters can be used to get rid of (a part of) the noise in the measurements. By contrast, it cannot be used to improve the signal as enabled by the analog amplifiers in our setups. Furthermore, in our experiments such a preprocessing turned out to be only marginally useful for the best setups, confirming that good measurements significantly simplify the statistical evaluation of leaking devices.



(a) Sample trace (b) 1st-order

(c) 2nd-order (d) 3rd-order

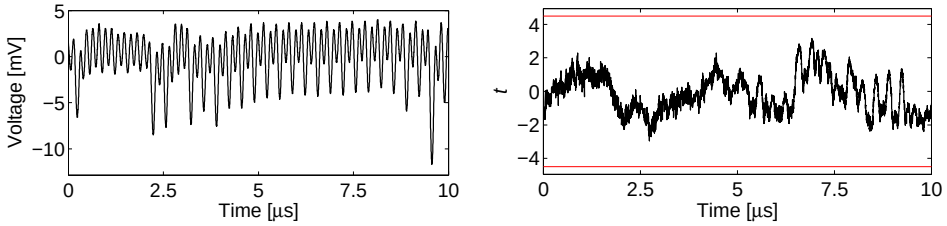
Fig. 2: Setup 1, fixed vs. random TVLA using 1 million traces



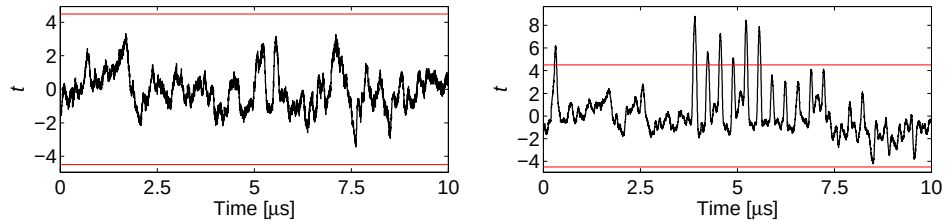
(a) Sample trace (b) 1st-order

(c) 2nd-order (d) 3rd-order

Fig. 3: Setup 2, fixed vs. random test TVLA using 1 million traces

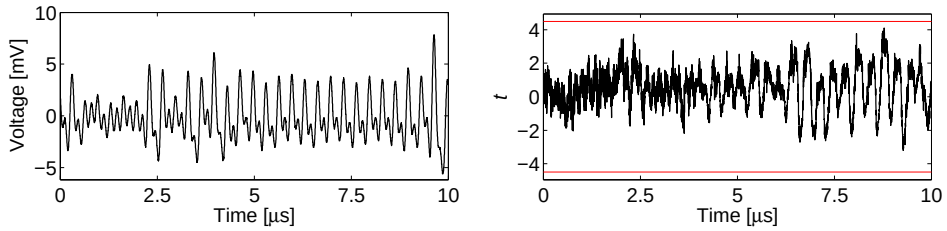


(a) Sample trace (b) 1st-order

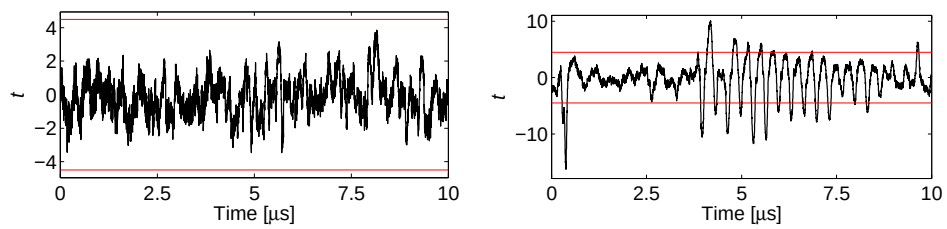


(c) 2nd-order (d) 3rd-order

Fig. 4: Setup 3, fixed vs. random TVLA using 1 million traces

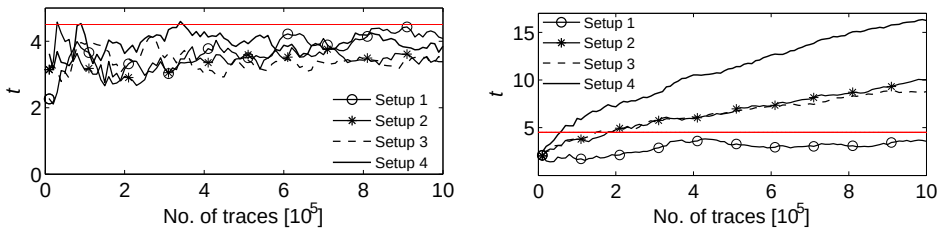


(a) Sample trace (b) 1st-order



(c) 2nd-order (d) 3rd-order

Fig. 5: Setup 4, fixed vs. random TVLA using 1 million traces



(a) 2nd-order (b) 3rd-order

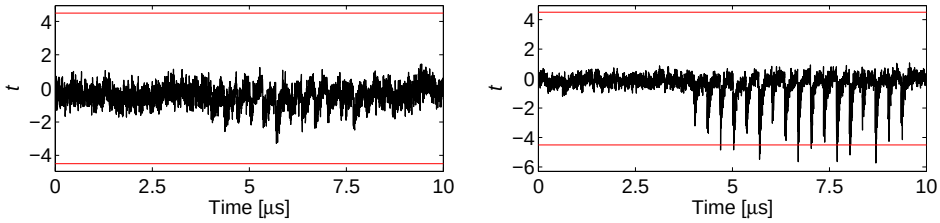
Fig. 6: Fixed vs. random TVLA in function of the number of traces (absolute values)

5 Faster leakage detection with Fixed vs. Fixed TVLA

In the last section, we have shown the impact of choosing fine-tuned setup. Now, we focus on the recently proposed non-specific fixed vs. fixed TVLA to evaluate the extent to which it can improve the speed of convergence of the original non specific fixed vs. random TVLA.

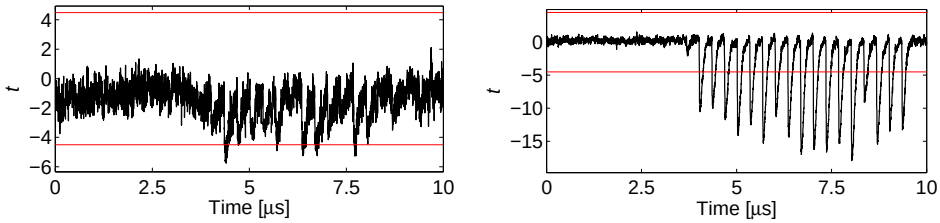
Following the same approach as in previous section, we collected 1 million measurements with each setup. The only difference with previous experiments is the partitioning, now based on two fixed classes. In order to get the best of both TVLAs, we carefully selected the plaintexts such that they led to an optimal signal. For the fixed vs. fixed TVLA, this amounts to select inputs that maximize the Hamming distance (HD) between each two consecutive values in the register (that we both control). For the fixed vs. random case, we only control one of the values, and therefore can only guarantee a halved HD on average. Of course, such an approach assumes a strong adversary with full knowledge and control over the DUT. However, we should note that this is a natural assumption for, e.g., secure hardware designers assessing the security level of a cryptographic device they have engineered. At this point we should also note that, even in the case where the two fixed plaintexts cannot be carefully selected (e.g., if the evaluator uses two randomly selected plaintexts), the fixed vs. fixed TVLA (on average) allows to double the signal in comparison with the fixed vs. random partitioning [8]. So our careful selection of plaintext indeed helps to fasten the evaluations (i.e., the goal of this paper) but does not affect the comparison between the two tests.

The results of the fixed vs. fixed TVLA at second- and third-orders are shown in Figure 7, Figure 8, Figure 9 and Figure 10. For completeness, the corresponding results at first-orders are provided by Figure 16, Figure 17, Figure 18 and Figure 19 in Appendix A. It is noteworthy that the fixed vs. fixed approach not only exhibited third-order leakages with a higher level of confidence (even with *setup 1*, which turned out to be unsuccessful in the previous section), but it also enabled the detection of second-order leakages by means of *setup 2*, *setup 3* and *setup 4*. Hence, these results further confirm the negative impact of having



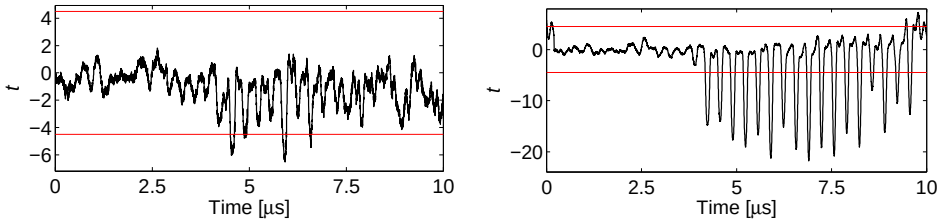
(a) 2nd-order (b) 3rd-order

Fig. 7: Setup 1, fixed vs. fixed TVLA using 1 million traces



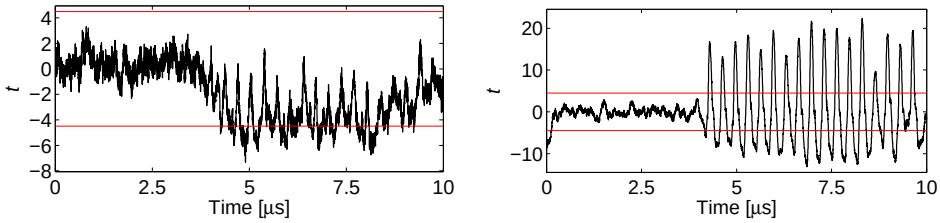
(a) 2nd-order (b) 3rd-order

Fig. 8: Setup 2, fixed vs. fixed TVLA using 1 million traces



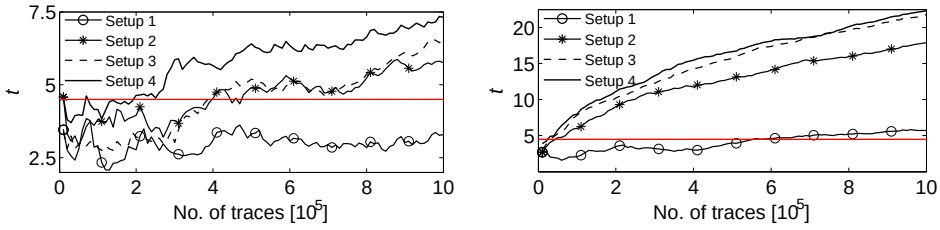
(a) 2nd-order (b) 3rd-order

Fig. 9: Setup 3, fixed vs. fixed TVLA using 1 million traces



(a) 2nd-order (b) 3rd-order

Fig. 10: Setup 4, fixed vs. fixed TVLA using 1 million traces



(a) 2nd-order (b) 3rd-order

Fig. 11: Fixed vs. fixed TVLA in function of the number of traces (absolute values)

signals with limited peak-to-peak amplitude, which can be mitigated thanks to the fixed vs. fixed TVLA.

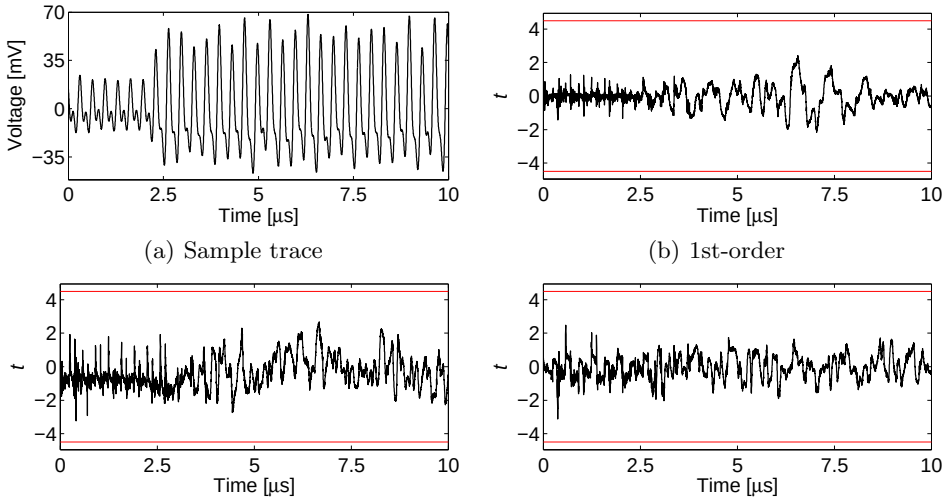
Finally, in order to compare the speed of convergence of both tests, we summarize the aforementioned results in Figure 11 as a function of the number of traces. As clear from Figure 11(a), the improvements at second-orders are significant, and they also serve to reaffirm *setup 4* as the best candidate among the different setups considered in this work. Besides, we should also highlight the results obtained for the third-order moments in Figure 11(b), where we can notice a reduction by a factor ≈ 4 on the number of required leakages for detections with *setup 2* and *setup 3*.

6 Noisy implementations and intensive evaluations

The last empirical results have shown the significant gains that we can obtain by relying on the non-specific fixed vs. fixed TVLA. Yet, it remains largely unclear whether such gains are due to increasing the signal or reducing the noise. Motivated by this question, in this section we move towards a more challenging scenario featuring a combination of masking and noise. In this new context the noise is synchronous with the crypto core, and so filtering it out looks a hard engineering task. All in all, in this setting all the gains coming from using the fixed vs. fixed TVLA will be due to the improved signal.

Since we expected the combination of masking and noise to exacerbate the required number of SCA traces to observe data dependencies at higher orders, we only considered *setup 4*. Indeed, preliminary results with the PRNG disabled already showed a reduction by a factor of ≈ 10 in the confidence to detect first-order leakages (see Figure 20 in Appendix A). Hence, when the PRNG was enabled, we decided to collect up to 100 million measurements following both leakage assessment techniques. As exemplified by the results in Figure 12, the fixed vs. random approach was not able to spot any sort of leakage. In order to make sure that our results were not biased by a poor choice of the fixed class, we tested different plaintexts that also led to analogous results. By contrast,

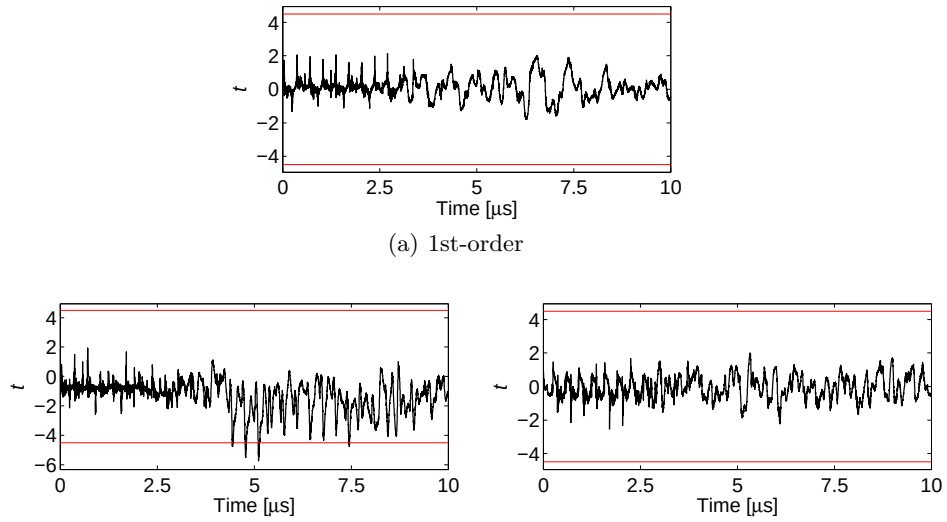
Figure 13 shows that relying on the fixed vs. fixed TVLA, such amount of measurements suffices to detect second-order leakages. Indeed, results in Figure 14 reveal that ≈ 60 million SCA traces can spot second-order leakages.



(a) Sample trace (b) 1st-order

(c) 2nd-order (d) 3rd-order

Fig. 12: Setup 4, fixed vs. random TVLA using 100 million traces effected by noise



(a) 1st-order

(b) 2nd-order (c) 3rd-order

Fig. 13: Setup 4, fixed vs. fixed TVLA using 100 million traces effected by noise

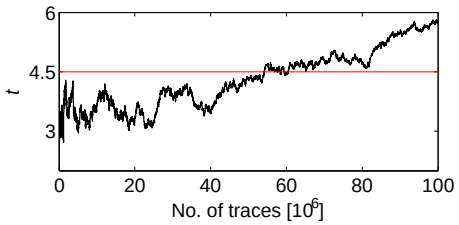


Fig. 14: Second-order fixed vs. fixed TVLA in function of the number of traces effected by noise (absolute values)

These results are well in line with theoretical expectations, which state that lower-order moments become more informative when the level of noise is sufficiently high [7]. They also confirm the effectiveness of generating additive noise on top of a masked implementation to harden the detection of higher-order leakages. More importantly, they show that even when the noise cannot be canceled out, by increasing the signal (i.e., with the fixed vs. fixed TVLA) the number of traces required to spot a higher-order leakage can be significantly reduced. In such a context, where millions of traces must be acquired, even small gain factors can save a lot of time and storage requirements to security evaluators.

7 Conclusion and open problems

In this work we have investigated up to which extent the time required to perform SCA evaluations can be reduced by means of enhanced measurement setups and statistical tools. Our preliminary investigations based on the popular fixed vs. random TVLA have highlighted the necessity of using well-designed setups. Besides, we have presented a fair comparison between it and its more recent counterpart, the so-called fixed vs. fixed TVLA. Our results have shown the benefits of the latter technique for reducing the measurement complexity of TVLA and therefore its time and storage requirements. In this regard, we refer to our last case study where a first-order TI was combined with a noise engine to harden the detection of higher-order leakages. In such a scenario, where millions of traces are first collected and analyzed afterwards, even small gain factors can save critical measurement time. In our concrete case, we were able to collect up to 100 million traces in 20 hours. So, a multiplication by a factor 4 (that we would expect if a successful detection had to be based on the fixed vs. random TVLA) would already correspond to several days of computation. This impact will be further amplified for higher-order masked implementations, e.g., multiplying the evaluation time respectively by 8 and 16 for third- and fourth-order secure implementations, and so turning it from days to weeks. In this respect we finally note that if the evaluator is able to control the masks during the acquisition, he can average the samples before raising them to the some power and therefore mitigate the noise amplification due to masking, as detailed in [18].

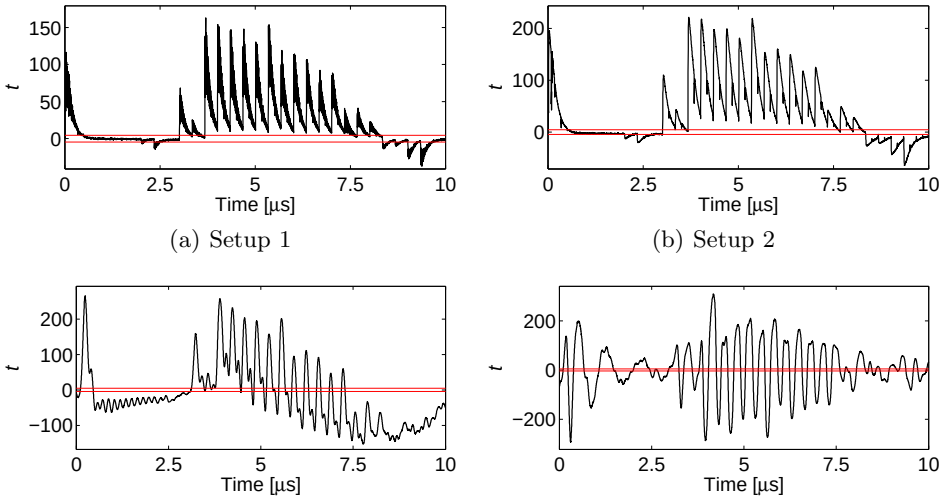
Acknowledgments. François-Xavier Standaert is an associate researcher of the Belgian Fund for Scientific Research (FNRS-F.R.S.). This work was funded in parts by the European Commission through the project REASSURE.

References

1. Side-channel AttacK User Reference Architecture. `http://satoh.cs.uec.ac.jp/SAKURA/index.html`.
2. J. Balasch, B. Gierlichs, O. Reparaz, and I. Verbauwhede. DPA, Bitslicing and Masking at 1 GHz. In *CHES 2015*, volume 9293 of *LNCS*, pages 599–619. Springer, 2015.
3. B. Bilgin, B. Gierlichs, S. Nikova, V. Nikov, and V. Rijmen. A More Efficient AES Threshold Implementation. In *AFRICACRYPT 2014*, volume 8469 of *LNCS*, pages 267–284. Springer, 2014.
4. B. Bilgin, B. Gierlichs, S. Nikova, V. Nikov, and V. Rijmen. Higher-Order Threshold Implementations. In *ASIACRYPT 2014*, volume 8874 of *LNCS*, pages 326–343. Springer, 2014.
5. E. Brier, C. Clavier, and F. Olivier. Correlation Power Analysis with a Leakage Model. In *CHES 2004*, volume 3156 of *LNCS*, pages 16–29. Springer, 2004.
6. S. Chari, J. R. Rao, and P. Rohatgi. Template Attacks. In *CHES 2002*, volume 2523 of *LNCS*, pages 13–28. Springer, 2002.
7. A. Duc, S. Faust, and F.-X. Standaert. Making Masking Security Proofs Concrete - Or How to Evaluate the Security of Any Leaking Device. In *EUROCRYPT 2015*, volume 9056 of *LNCS*, pages 401–429. Springer, 2015.
8. F. Durvaux and F.-X. Standaert. From Improved Leakage Detection to the Detection of Points of Interests in Leakage Traces. In *EUROCRYPT 2016*, volume 9665 of *LNCS*, pages 240–262. Springer, 2016.
9. G. Goodwill, B. Jun, J. Jaffe, and P. Rohatgi. A testing methodology for side channel resistance validation. In *NIST non-invasive attack testing workshop*, 2011.
10. T. Güneysu and A. Moradi. Generic Side-Channel Countermeasures for Reconfigurable Devices. In *CHES 2011*, volume 6917 of *LNCS*, pages 33–48. Springer, 2011.
11. S. Merino Del Pozo, F.-X. Standaert, D. Kamel, and A. Moradi. Side-channel attacks from static power: when should we care? In *DATE 2015*, pages 145–150. ACM, 2015.
12. A. Moradi and O. Mischke. On the Simplicity of Converting Leakages from Multivariate to Univariate - (Case Study of a Glitch-Resistant Masking Scheme). In *CHES 2013*, volume 8086 of *LNCS*, pages 1–20. Springer, 2013.
13. A. Moradi, A. Poschmann, S. Ling, C. Paar, and H. Wang. Pushing the Limits: A Very Compact and a Threshold Implementation of AES. In *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 69–88. Springer, 2011.
14. A. Moradi and A. Wild. Assessment of Hiding the Higher-Order Leakages in Hardware - What Are the Achievements Versus Overheads? In *CHES 2015*, volume 9293 of *LNCS*, pages 453–474. Springer, 2015.
15. S. Nikova, V. Rijmen, and M. Schläffer. Secure Hardware Implementation of Nonlinear Functions in the Presence of Glitches. *J. Cryptology*, 24(2):292–321, 2011.
16. A. Poschmann, A. Moradi, K. Khoo, C. Lim, H. Wang, and S. Ling. Side-Channel Resistant Crypto for Less than 2, 300 GE. *J. Cryptology*, 24(2):322–345, 2011.

17. T. Schneider and A. Moradi. Leakage Assessment Methodology - A Clear Roadmap for Side-Channel Evaluations. In *CHES 2015*, volume 9293 of *LNCS*, pages 495–513. Springer, 2015.
18. F. Standaert. How (not) to use Welch's T-test in side-channel security evaluations. *IACR Cryptology ePrint Archive*, 2017:138, 2017.

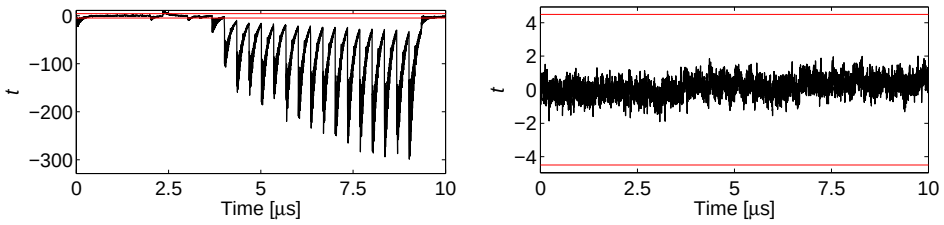
A Additional Figures



(a) Setup 1 (b) Setup 2

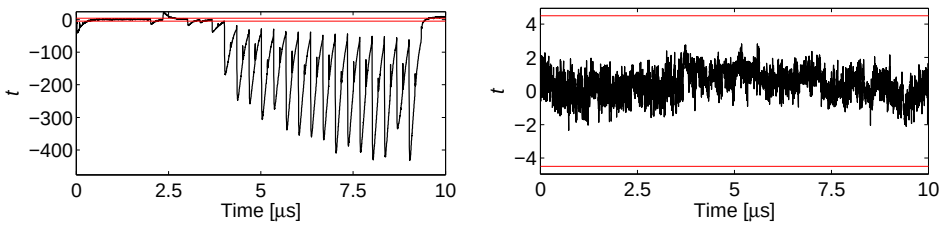
(c) Setup 3 (d) Setup 4

Fig. 15: (PRNG OFF) fixed vs. random TVLA using 10k traces



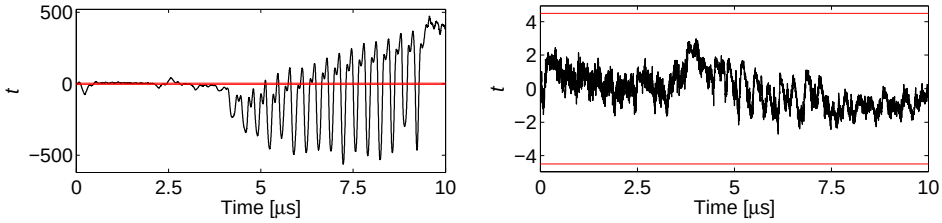
(a) PRNG OFF (10k traces) (b) PRNG ON

Fig. 16: Setup 1, first-order fixed vs. fixed TVLA using 1 million traces



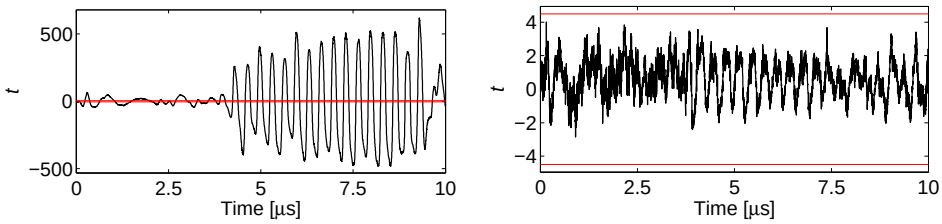
(a) PRNG OFF (10k traces) (b) PRNG ON

Fig. 17: Setup 2, first-order fixed vs. fixed TVLA using 1 million traces



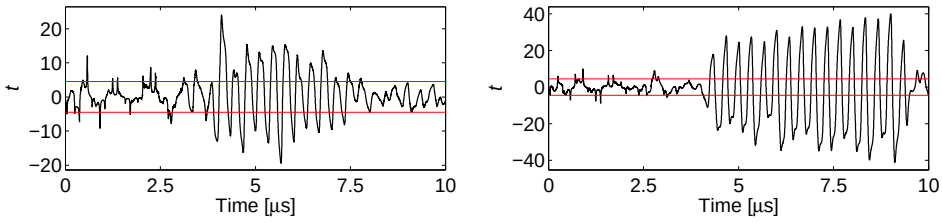
(a) PRNG OFF (10k traces) (b) PRNG ON

Fig. 18: Setup 3, first-order fixed vs. fixed TVLA using 1 million traces



(a) PRNG OFF (10k traces) (b) PRNG ON

Fig. 19: Setup 4, first-order fixed vs. fixed TVLA using 1 million traces



(a) Fixed vs. random (b) Fixed vs. fixed

Fig. 20: (PRNG OFF) setup 4, TVLA using 10k traces effected by noise