Mind the Gap: Towards Secure 1st-order Masking in Software *

Author list removed for submission

Institute list removed for submission

Abstract. Software-based cryptographic implementations are vulnerable to side-channel analysis. To protect against it, implementors often opt for masking countermeasures, which ensure theoretical protection against value-based leakages. However, the practical effectiveness of masking is halted by physical effects such as glitches and distance-based leakages, which violate the *independent leakage assumption* (ILA) and result in security order reductions. This paper aims to address this gap between masking theory and practice in the following threefold manner. First, we perform an in-depth investigation of the device-specific effects that invalidate ILA in the AVR microcontroller ATMega163. Second, we provide an automated tool, capable of detecting ILA violations in AVR assembly code. Last, we craft the first (to our knowledge) *"hardened"* 1st-order ISW-based, masked Sbox implementation, which is capable of resisting 1st-order, univariate side-channel attacks. Enforcing the ILA in the masked RECTANGLE Sbox requires <> clock cycles, i.e. a <>-fold increase compared to a naive 1st-order ISW-based implementation.

Keywords: Masking, AVR, independent leakage assumption, distance-based leakage, RECTANGLE, SCA

1 Introduction

Nowadays, the explosive growth of the "Internet of Things" (IoT) is reshaping modern society, pervading its infrastructure and communications. The rapid price drop in IoT components has transformed everyday products, enhancing them with network connectivity and information exchange capabilities. Amidst this new status quo, devices ranging from cheap sensors to expensive vehicles, are required to maintain a heightened level of theoretical and physical security.

For instance, side-channel attacks (SCA) allow adversaries to recover sensitive data, by observing and analyzing the physical characteristics and emanations of a cryptographic implementation [13]. Such physical attacks motivated research towards countermeasures that perform noise amplification, thus hindering the adversary's recovery capabilities. A common choice for provably secure, noise-amplifying software countermeasure is masking [5, 11]. Masking employs

* Removed for submission

secret-sharing techniques that establish theoretical security against the value-based leakage model. Rephrasing, masking secures implementations against adversaries that can only extract information about the value being processed at a given time and cannot perceive any leakage from idle values at the same time. This underlying assumption is often referred to as the *independent leakage assumption* (ILA) [17]. Unfortunately, such a limited adversarial model is not applicable in many practical, software-based scenarios. For instance, devices often exhibit *distance-based leakages*, which can reduce the security of the masking countermeasure [1, 8]. Likewise, coupling effects [17] and glitches [14] can pose similar security hazards.

This work attempts to bridge this gap between theory and practice in the masking countermeasure with the following threefold contribution. First, we investigate several effects that violate ILA in an ATMega163 microcontroller and subsequently we establish solutions. Second, we use this knowledge in order to build an assembly-oriented tool that is capable of detecting ILA violations in AVR-based masked implementations. Third, assisted by the developed tool, we craft the first (to our knowledge) 1st-order masked implementation in ATMega163 that is capable of resisting 1st-order, univariate attacks. In other words, we enforce the ILA in order to severely limit the informativeness of 1st-order leakages, forcing the adversary to resort to 2nd-order attacks. As a proof of concept, we develop a "hardened" 1st-order, ISW-based [11], bitsliced Sbox for the RECTANGLE cipher [24]. The "hardened" implementation requires <> clock cycles, a <>-fold increase compared to a naive 1st-order, ISW-based, bitsliced Sbox of the same cipher.

The paper is organized as follows. In Section 2, we provide preliminaries w.r.t. masking, the experimental setup and the evaluation techniques we employ. In Section 3 we offer a detailed description all the ILA-breaching effects that we have identified in ATMega163. Section 4 discusses the development of the assembly checking tool. Section 5 details the construction of a "hardened" RECTANGLE, 1st-order masked Sbox for ATMega163. We conclude and discuss future work in Section 6.

2 Background

2.1 Boolean Masking & Order Reduction

Chari et al., Goubin et al. and Messerges [5,9,15] were among the first to suggest splitting intermediate values with a secret sharing scheme, in order to force attackers to analyze higher-order statistical moments. Analytically, a dth-order masking scheme splits a sensitive value x into $d+1$ shares $(x_0, x_1, \ldots, x_d)$, as shown below.

$$x = x_0 \oplus x_1 \cdots \oplus x_d \tag{1}$$

The shares $(x_0, x_1, \ldots, x_d)$ are also referred to as the $(d+1)$-family of shares corresponding to x [18]. Given that the ILA holds and assuming sufficient noise, it has been shown that the number of traces required for a successful attack

grows exponentially w.r.t. the order d [5, 16]. Several implementation options have been suggested for the masking countermeasure, ranging from lookup-table techniques [6, 22] to GF-based circuits [10, 11, 18].

In parallel with the development of masked implementations, side-channel research focused on the practical evaluation of the countermeasure. Balasch et al. [1] put forward the concepts of value-based and distance-based leakages, as well as the notion of order reduction. We briefly restate their definitions below.

Value/Distance-based leakage function: A leakage function $L(.)$ consists of a deterministic part $L_d(.)$ and random additive noise N. The leakage function is value-based if $L_d(.)$ can only take arguments from the set of intermediate values produced by the masking scheme. The leakage is distance-based if $L_d(.)$ can take arguments from the set that contains all possible combinations of intermediate values. The combination can imply any type of transition e.g. XOR, concatenation, etc.

Order-reduction theorem: A dth-order secure masking scheme under value-based leakages is $\lfloor \frac{d}{2} \rfloor$th-order secure under distance-based leakages.

The order-reduction theorem has been verified experimentally for orders $d = 1, 2$ by Balasch et al. [1] in AVR-based and 8051-based devices. De Groot et al. [8] have verified experimentally the theorem for orders $d = 1, 2$ in the ARM Cortex-M4.

2.2 Experimental Setup & Evaluation

The implementation and evaluation is performed on the 8-bit AVR-based ATMega163 smartcard[1]. The device features a 4.4 MHz clock, <> bytes of SRAM and <> bytes of Flash memory. The traceset acquisition is carried out using the Riscure PowerTracer[2] and the Picoscope 5203 oscilloscope. The sampling rate is set at 31.5 MSamples/sec and the only post-processing applied is signal alignment.

The evaluation of the actual security order of a masking scheme is, in general, an open problem. We often face the *limited attack scope*, i.e. a given attack may not exploit the available leakage due to e.g. an unsuitable choice of intermediate values or an incorrect power model. To address this problem, generic side-channel distinguishers and extensive profiling techniques have been developed [2, 20, 23]. In this work, we opt for the leakage detection methodology [12] which prioritizes leakage detection over leakage exploitation, speeding up certain evaluation aspects. In detail, we employ the random vs. fixed, non-specific, 1st-order t-test. We perform a random vs. fixed acquisition and obtain two distinct tracesets S_{fixed} and S_{random}, under the same encryption key. The input plaintext for S_{fixed} is set to a fixed value, while for S_{random}, the input is uniformly

[1] `http://www.atmel.com/images/doc1142.pdf`
[2] `https://www.riscure.com/security-tools/hardware/power-tracer`

random. The implementation receives the fixed or random plaintext in a non-deterministic and randomly-interleaved way (as recommended by Shneider et al. [19]). Following the data acquisition, the 1st-order t-test will assess whether the two sets S_{fixed}, S_{random} stem from the same population.

$$\begin{aligned} H_{null}: &\ \mu_{fixed} = \mu_{random} \\ H_{alt}: &\ \mu_{fixed} \neq \mu_{random} \end{aligned} \tag{2}$$

$$w = \frac{\mu_{fixed} - \mu_{random}}{\sqrt{\frac{\sigma^2_{fixed}}{n} + \frac{\sigma^2_{random}}{m}}} \tag{3}$$

$$\upsilon = \frac{(\frac{\sigma^2_{fixed}}{n} + \frac{\sigma^2_{random}}{m})^2}{\frac{\sigma^4_{fixed}}{n^2(n-1)} + \frac{\sigma^4_{random}}{m^2(m-1)}} \tag{4}$$

The null hypothesis H_{null} is rejected at a given level of significance α (often set to 0.99999), if $|w| > t_{\alpha/2,\upsilon}$, where $t_{\alpha/2,\upsilon}$ is the value of the Student t distribution with υ degrees of freedom. In the evaluation context, rejecting H_{null} implies leakage detection, i.e. potential evidence of an ineffective masking scheme.

In this paper, we will use the t-test as a detection tool w.r.t. ILA effects and their solutions (see Section 3). Still, we will also employ 1st-order CPA methods [4] in order to demonstrate the exploitability of such effects. In order to reduce the computational cost of the evaluation, we use the memoryless formulas suggested by Schneider et al. [19] and the incremental approach for CPA by Botinelli et al. [3].

3 ILA-Breaching Effects

In this section, we present <> effects identified in the ATMega163 microcontroller that breach ILA and pose a hazard to any masking scheme's security. Analytically, the effects below demonstrate that independent computations *do not* lead to independent leakages and thus the order-reduction theorem can become applicable.

Every effect (Sections 3.1, 3.2, 3.3 <>) is described as a standalone, assembly-based scenario that manipulates two 4-bit shares x_0, x_1 originating from the sensitive, key-dependent, 4-bit value x, s.t. $x = x_0 \oplus x_1$. The shares x_0, x_1 are always manipulated in a theoretically sound manner, adhering to the masking scheme's requirements, i.e. we never combine the shares directly (e.g. via `eor x0, x1`).

For all these theoretically sound scenarios described, we show experimentally that ILA is not fulfilled by employing 1st-order, univariate techniques. Namely, we perform correlation-based analysis [4], computing ρ between the Hamming weight of the sensitive, key-dependent value x and the experimentally acquired traceset. To maintain a wide attack scope, we also use the leakage detection methodology [12, 19] and compute the 1st-order, random vs. fixed t-test. We conclude every scenario by suggesting possible solutions that enforce ILA and

verify all the proposed solutions experimentally, using a large number of traces. Restating Balasch et al. [1], as we are always limited by the traces at hand, we cannot rule out the existence of 1st-order leakages, yet we establish that their informativeness is limited compared to 2nd-order leakages in the target device. Note that extra care is taken in order to assess all effects independently, i.e. we use the suggested solutions so as to isolate the effect under discussion from the rest.

The effects under discussion can manifest in several data storage units (e.g. registers, SRAM/Flash memory cells, I/O buffers, etc.) and may relate to different instructions of the AVR ISA[3], leading to a very large number of potential scenarios. In order to maintain a feasible scope, we limit our discussion to instructions and storage that are often encountered in the context of cryptographic implementation, i.e. logical instructions and SRAM memory accesses.

3.1 Overwrite Effect

The overwrite effect is observable when a share gets overwritten by a different share from the same family. For instance, if share x_0 in a data storage unit (register, memory cell, etc.) gets overwritten by share x_1, then the power consumption correlates with the number of bits switched i.e. $x_0 \oplus x_1$. This effect was observed by Daemen et al. [21] and later revisited by Coron et al. [7].
Below, we address the most common situations in which overwriting arises during a cryptographic implementation. We perform two experiments: a register-based overwrite via the instruction `mov x0, x1`, and a memory-based overwrite via `st SRAM_x0 , x1`.

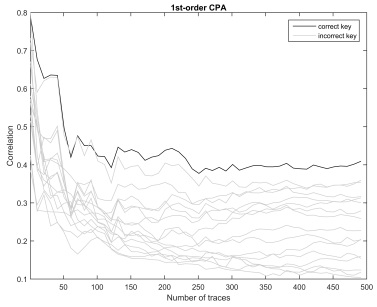
[3] `http://www.atmel.com/images/Atmel-0856-AVR-Instruction-Set-Manual.pdf`

```
;share x0 in r17
;share x1 in r23
mov r17,r23
```

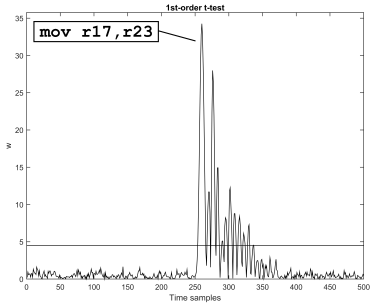
(a) Register overwrite experiment.

```
;share x0 in SRAM 0x0080
;share x1 in r17
ldi r27,0x00
ldi r26,0x80
st X,r17
```

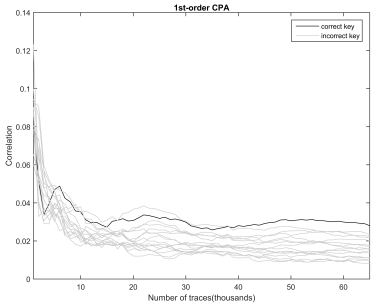
(b) Memory overwrite experiment.



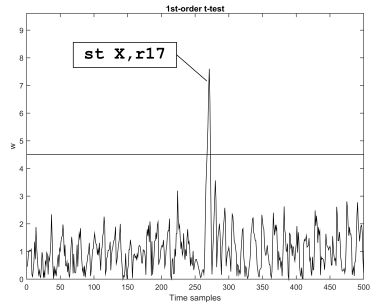
(c) Register overwrite, 1st-order CPA, HW model, 500 traces.



(d) Register overwrite, 1st-order t-test, 5k random vs. 5k fixed.



(e) Memory overwrite, 1st-order CPA, HW model, 65k traces.



(f) Memory overwrite, 1st order t-test, 50k random vs. 50k fixed.

Fig. 1: Register/memory-based overwrite effects

We confirm that overwriting is indeed an ILA-breaching effect, manifesting both in registers and SRAM memory. Note that the exploitability of the effect varies according to the data storage unit: in ATMega163, register-based overwriting can be exploited with roughly 500 traces, while memory-based requires at least 40k traces. Preventing register/memory-based overwrites is straightforward: the corresponding register/cell needs to be cleared in advance. We confirm experimentally that this solution remains secure against a <>k random vs. <>k fixed, 1st-order t-test (see Appendix A).

3.2 Memory Remnant Effect

The memory remnant effect is a type of leakage originating from consecutive SRAM accesses to different shares of the same family. For instance, assume that shares x_0, x_1 are stored in SRAM cells and get accessed sequentially. Naturally, the first access leaks share x_0 (value-based leakage), yet it also creates a "remnant" of x_0. The second memory access will leak share x_1 (value-based leakage) *and* the remnant x_0 concurrently, reducing the security order of the scheme.

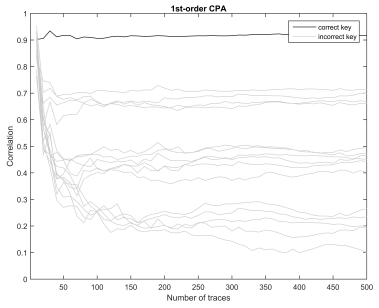
We address this scenario with the following two experiments. First, we demonstrate how two consecutive SRAM accesses `ld rA, SRAM_x0` and `ld rB, SRAM_x1` produce the remnant effect. Second, we show how clearing the register and inserting a dummy SRAM access can remove the remnant.

```
;share x0 in 0x0080
;share x1 in 0x0090
ldi r27, 0x00
ldi r26, 0x80
ld r17, X
ldi r27, 0x00
ldi r26, 0x90
ld r20, X
```

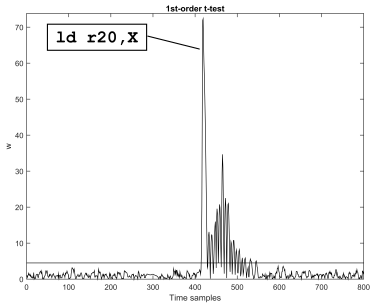
(a) Memory remnant experiment.

```
;share x0 in 0x0080
;share x1 in 0x0090
ldi r27, 0x00
ldi r26, 0x80
ld r17, X
ldi r17, 0x00
ldi r26, 0x85
ld r17, X
ldi r26, 0x90
ld r20, X
```

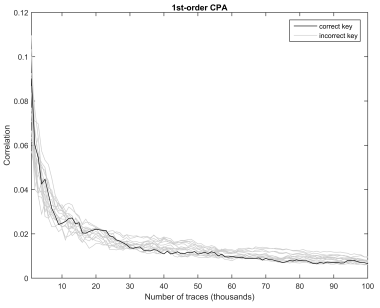
(b) Clearing remnant experiment.



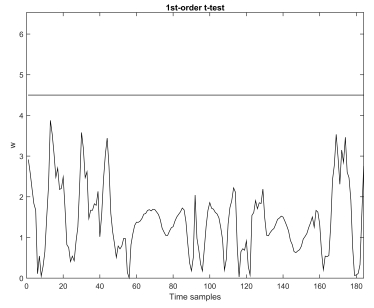
(c) Memory remnant effect,1st-order CPA, HW model, 500 traces.



(d) Memory remnant effect, 1st-order t-test, 5k random vs. 5k fixed.



(e) Clearing remnant effect,1st-order CPA, HW model, 100k traces.



(f) Clearing remnant effect, 1st-order t-test, 100k random vs. 100k fixed.

Fig. 2: Memory remnant effect

As shown above, consecutive SRAM accesses can potentially lead to ILA violations. Preventing this requires the clearing of the register and the insertion of dummy SRAM. Alternatively, the implementor could ensure that same-family shares are not accessed sequentially. Exploiting (in a univariate manner) the memory remnant effect in ATMega163 requires less than 500 traces. The `st` instruction produces a similar effect (see Appendix A).

3.3 Combined Leakage Effect

The combined leakage effect implies that accessing/processing the contents of a data storage unit will cause leakage in another unit as well. For example, assume that share x_0 is stored in register `rB` and share x_1 is being processed in register `rA`. Assume also that the registers `rA, rB` are subject to the combined leakage effect. Processing `rB` will produce a value-based leakage of x_1. At the same time, the combined leakage effect will cause `rA` to leak the value of x_0, resulting in the concurrent leakage of both shares, revealing the sensitive value x.

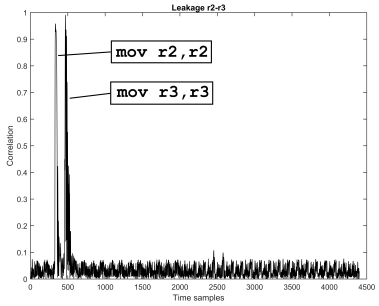
The following two experiments verify the combined leakage effect between registers `r2, r3`, i.e. that a share stored in `r2` leaks when manipulating `r2` *or* `r3` and vice-versa.

```
;clear all registers
;store x0 in r2
mov r0, r0
NOPs
mov r1, r1
NOPs
mov r2, r2
NOPs
mov r3, r3
NOPs
...
mov r31, r31
```

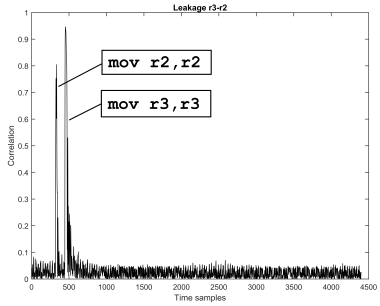
(a) Combined leakage `r2-r3`.

```
;clear all registers
;store x0 in r3
mov r0, r0
NOPs
mov r1, r1
NOPs
mov r2, r2
NOPs
mov r3, r3
NOPs
...
mov r31, r31
```

(b) Combined leakage `r3-r2`.



(c) Correlation `r2-r3` $\rho(HW(x_0), traceset)$, 5k traces.



(d) Correlation `r3-r2` $\rho(HW(x_0), traceset)$, 5k traces.

Fig. 3: Combined leakage effect.

As shown above, we have identified a pair of data storage units (`r2,r3`) that exhibit the combined leakage effect. Note that in this case the effect is symmetrical, i.e. `r2` triggers `r3` and vice-versa. We run the same experiment in order to identify all possible combined leakages in the register file. The results

are available in Appendix A, matrix R. The combined leakage mostly affects neighboring registers, although exceptions exist, e.g. register `r0`. We did not identify a similar effect in SRAM memory cells, yet our experiments were limited to a small region of neighboring cells

3.4 Other Leakage Effects

<> PENDING

Summing up, we stress the following focal points regarding the ILA-breaching effects and their solutions:

- All identified effects are device-dependent, i.e. there is no hard guarantee that they are observable/reproducible in different AVR-based microcontrollers, let alone different architectures such as ARM, TI, PIC etc. Both intra-AVR and inter-architectural observability of the effects remains open.
- The effects are often counter-intuitive when viewed in the assembly layer of abstraction. They originate from the hardware and/or the physical layer, thus can only be detected via experimental evaluation. Linking the assembly ILA-breaching effects to a particular hardware component or physical phenomenon is non-trivial [**?**, **?**], especially without knowledge of the underlying chip architecture and properties.
- Since the effect's detection requires experimental evaluation, different instructions or code arrangements can potentially lead to additional, unidentified ILA-breaching effects. Still, we maintain that it is possible to construct "hardened" masked operations in ATMega163 by removing the identified effects (see Section **??**). It remains open whether the suggested solutions are computationally optimal or more efficient clearing techniques can be identified.

The takeaway message of this section is that assembly-level soundness cannot enforce ILA and hence 1st-order security, due to the nature of the breaching effects. However, it is possible to acquire sufficient knowledge about effects and solutions in a particular device. These non-intuitive checks discussed above can be subsequently integrated into a code-checking tool which can identify such effects in assembly code.

4 Detection Tool

5 Hardened 1st-order Masked Sbox: A Case Study Using RECTANGLE

6 Conclusions

References

1. Josep Balasch, Benedikt Gierlichs, Vincent Grosso, Oscar Reparaz, and François-Xavier Standaert. On the cost of lazy engineering for masked software implemen-

tations. In *Smart Card Research and Advanced Applications - 13th International Conference, CARDIS 2014, Paris, France, November 5-7, 2014. Revised Selected Papers*, pages 64–81, 2014.
2. Lejla Batina, Benedikt Gierlichs, Emmanuel Prouff, Matthieu Rivain, François-Xavier Standaert, and Nicolas Veyrat-Charvillon. Mutual information analysis: a comprehensive study. *J. Cryptology*, 24(2):269–291, 2011.
3. Paul Bottinelli and Joppe W. Bos. Computational aspects of correlation power analysis. *IACR Cryptology ePrint Archive*, 2015:260, 2015.
4. Eric Brier, Christophe Clavier, and Francis Olivier. Correlation power analysis with a leakage model. In *Cryptographic Hardware and Embedded Systems - CHES 2004: 6th International Workshop Cambridge, MA, USA, August 11-13, 2004. Proceedings*, pages 16–29, 2004.
5. Suresh Chari, Charanjit S. Jutla, Josyula R. Rao, and Pankaj Rohatgi. Towards sound approaches to counteract power-analysis attacks. In *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, pages 398–412, 1999.
6. Jean-Sébastien Coron. Higher order masking of look-up tables. In *Advances in Cryptology - EUROCRYPT 2014 - 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, May 11-15, 2014. Proceedings*, pages 441–458, 2014.
7. Jean-Sébastien Coron, Christophe Giraud, Emmanuel Prouff, Soline Renner, Matthieu Rivain, and Praveen Kumar Vadnala. Conversion of security proofs from one leakage model to another: A new issue. In *Constructive Side-Channel Analysis and Secure Design - Third International Workshop, COSADE 2012, Darmstadt, Germany, May 3-4, 2012. Proceedings*, pages 69–81, 2012.
8. Wouter de Groot, Kostas Papagiannopoulos, Antonio de la Piedra, Erik Schneider, and Lejla Batina. Bitsliced masking and ARM: friends or foes? *IACR Cryptology ePrint Archive*, 2016:946, 2016.
9. Louis Goubin and Jacques Patarin. DES and differential power analysis (the "duplication" method). In Çetin Kaya Koç and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems, First International Workshop, CHES'99, Worcester, MA, USA, August 12-13, 1999, Proceedings*, volume 1717 of *Lecture Notes in Computer Science*, pages 158–172. Springer, 1999.
10. Dahmun Goudarzi and Matthieu Rivain. How fast can higher-order masking be in software? Cryptology ePrint Archive, Report 2016/264, 2016. `http://eprint.iacr.org/2016/264`.
11. Yuval Ishai, Amit Sahai, and David Wagner. Private circuits: Securing hardware against probing attacks. In Dan Boneh, editor, *Advances in Cryptology - CRYPTO 2003, 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 2003, Proceedings*, volume 2729 of *Lecture Notes in Computer Science*, pages 463–481. Springer, 2003.
12. Gilbert Goodwill Joshua Jaffe Gary Kenworthy Jeremy Cooper, Elke DeMulder and Pankaj Rohatgi. Test vector leakage assessment (tvla) methodology in practice.
13. Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, pages 388–397, 1999.
14. Stefan Mangard and Kai Schramm. Pinpointing the side-channel leakage of masked AES hardware implementations. In *Cryptographic Hardware and Embedded Systems - CHES 2006, 8th International Workshop, Yokohama, Japan, October 10-13, 2006, Proceedings*, pages 76–90, 2006.

15. Thomas S. Messerges. Securing the AES finalists against power analysis attacks. In Bruce Schneier, editor, *Fast Software Encryption, 7th International Workshop, FSE 2000, New York, NY, USA, April 10-12, 2000, Proceedings*, volume 1978 of *Lecture Notes in Computer Science*, pages 150–164. Springer, 2000.
16. Emmanuel Prouff and Matthieu Rivain. Masking against side-channel attacks: A formal security proof. In *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, pages 142–159, 2013.
17. Mathieu Renauld, François-Xavier Standaert, Nicolas Veyrat-Charvillon, Dina Kamel, and Denis Flandre. A formal study of power variability issues and side-channel attacks for nanoscale devices. In *Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings*, pages 109–128, 2011.
18. Matthieu Rivain and Emmanuel Prouff. Provably secure higher-order masking of AES. In *Cryptographic Hardware and Embedded Systems, CHES 2010, 12th International Workshop, Santa Barbara, CA, USA, August 17-20, 2010. Proceedings*, pages 413–427, 2010.
19. Tobias Schneider and Amir Moradi. Leakage assessment methodology - A clear roadmap for side-channel evaluations. In *Cryptographic Hardware and Embedded Systems - CHES 2015 - 17th International Workshop, Saint-Malo, France, September 13-16, 2015, Proceedings*, pages 495–513, 2015.
20. François-Xavier Standaert, Tal Malkin, and Moti Yung. A unified framework for the analysis of side-channel key recovery attacks. In *Advances in Cryptology - EUROCRYPT 2009, 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cologne, Germany, April 26-30, 2009. Proceedings*, pages 443–461, 2009.
21. Keccak team. *Note on side-channel attacks and their countermeasures.*
22. Junwei Wang, Praveen Kumar Vadnala, Johann Großschädl, and Qiuliang Xu. Higher-order masking in practice: A vector implementation of masked AES for ARM NEON. In *Topics in Cryptology - CT-RSA 2015, The Cryptographer's Track at the RSA Conference 2015, San Francisco, CA, USA, April 20-24, 2015. Proceedings*, pages 181–198, 2015.
23. Carolyn Whitnall, Elisabeth Oswald, and Luke Mather. An exploration of the kolmogorov-smirnov test as a competitor to mutual information analysis. In *Smart Card Research and Advanced Applications - 10th IFIP WG 8.8/11.2 International Conference, CARDIS 2011, Leuven, Belgium, September 14-16, 2011, Revised Selected Papers*, pages 234–251, 2011.
24. Wentao Zhang, Zhenzhen Bao, Dongdai Lin, Vincent Rijmen, Bohan Yang, and Ingrid Verbauwhede. RECTANGLE: a bit-slice lightweight block cipher suitable for multiple platforms. *SCIENCE CHINA Information Sciences*, 58(12):1–15, 2015.