HAL
open science

From theory to practice of private circuit: A cautionary note

Debapriya Basu Roy, Shivam Bhasin, Sylvain Guilley, Jean-Luc Danger, Debdeep Mukhopadhyay

To cite this version:

Debapriya Basu Roy, Shivam Bhasin, Sylvain Guilley, Jean-Luc Danger, Debdeep Mukhopadhyay. From theory to practice of private circuit: A cautionary note. 33rd IEEE International Conference on Computer Design (ICCD), Oct 2015, New York, United States. 10.1109/ICCD.2015.7357117 . hal-02412245v2

HAL Id: hal-02412245

https://telecom-paris.hal.science/hal-02412245v2

Submitted on 12 Dec 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

From Theory to Practice of Private Circuit: A Cautionary Note

Debapriya Basu Roy*, Shivam Bhasin†, Sylvain Guilley†, Jean-Luc Danger† and Debdeep Mukhopadhyay*
* IIT Kharagpur, India
† TELECOM-ParisTech & Secure-IC S.A.S., France

***Abstract*—Private circuits, from their publication, have been really popular among the researchers. They also form the basis for provable masking schemes. There are several works which try to improve the results of bit-level private circuits based on 2-input gates for the combinational logic. However, strangely, no practical side-channel analysis of private circuits has been presented so far, which is the focus of the present paper. In this paper, we have tried to identify the 'ambush' or hidden dangers in the implementation of private circuits, which can compromise its security in practical scenarios. We have implemented block cipher *SIMON* with private circuit and have performed side-channel analysis on it. The result shows that, in practice, there is significant amount of information leakage which can be exploited by adversaries. Some leakage comes from practical optimization applied by standard CAD tools, if they restructure the netlists. But even with immutable netlists, we identify leakage caused by a kind of *glitch* known as *early evaluation*. Lastly, we demonstrate how to translate theoretically secure private circuit to practically secure private circuit with added overhead, by clocking every combinational gate. Leakage detection tests are applied to attest the security of considered variants of private circuits.**

Keywords: Provably secure masking, ISW, Boolean logic optimization, glitches, side-channel attacks, synchronized logic, SIMON.

I. Introduction

The value of information has increased significantly in the last decade and with this, the need of protecting it from malicious adversaries has also increased. Cryptography enables us to secure this valuable information from malicious agents. However a mathematically secure cryptographic algorithm does not necessarily guarantee security as most of the implementations of cryptographic algorithms can be cracked through *Side-Channel Attacks (SCA)*. In a standard SCA, apart from the plaintext and ciphertext, an adversary has the access to physical information like power consumption, electromagnetic (EM) radiation, timing information etc. of the device. This information acts like unwanted (or side) channels through which sensitive data can leak. Designing countermeasures against SCA is therefore an absolute necessity and a popular research topic to ensure proper security of the sensitive information.

In their seminal work [1], Ishai, Sahai and Wagner, proposed a theoretical countermeasure against probing attack, which is strongest form of SCA. Probing attack considers a strong adversary who is capable of observing exact values of one or several internal nets of the circuit including the nets containing sensitive information. Although the proposed countermeasure [1], here after referred to as *ISW scheme* or *t*-private circuits, is suited for probing attack, it can be used to prevent power or EM SCA. In case of SCA, the adversary cannot observe the exact value of the sensitive information (key bits or key-dependent nets) but a linear or non-linear transformation of sensitive values like Hamming weight or Hamming distance. Private circuits also form the basis of much studied countermeasures against SCA known as masking [2]. In particular, Rivain and Prouff [2] had proposed a dth order provably secure masking scheme for AES. This masking scheme was derived by optimizing the hardware-oriented *t*-private circuits for software implementations.

The *t*-private circuits are secure against an adversary capable of observing any t nets of the circuit at any given time instant. Construction of *t*-private circuits involves $2t$ number of random bits for each input bit of the circuit. Moreover, it requires $2t+1$ random bits for each 2-input *AND* gate present in the circuit. The overall complexity of the design is $\mathcal{O}(nt^2)$ where n is the number of gates in the circuit.

The complexity of $\mathcal{O}(nt^2)$ is often considered impractical for several practical implementations. After the publication of [1], there have been many works which try to improve its result, in particular the area overhead. In [3], Park and Tyagi have improved the complexity of private circuit from $\mathcal{O}(nt^2)$ to $\mathcal{O}(nt)$. Moreover, in their recent work [4], the authors have further improved it to $\lceil t/2 \rceil$ for private circuits. They have also provided theoretical analysis and improvement of private circuit in context of power based side-channel attack and glitch [5], [6]. On the other hand, Rivain and Prouff have developed similar methodology like private circuits for masking countermeasure in software applications of crypto-algorithms [2]. In another work similar to private circuits [7], Faust et al. have provided a general circuit transformation for two different leakage models:- constant depth circuit leakage model and noisy leakage model. Difference between countermeasures of [7] and [1] is that private circuit considers leakage model as local measurement function, whereas in [7], leakage model is considered as global measurement function.

Further optimizations to private circuits motivates the minimal use of *AND* gates in the circuit which has maximum complexity in private circuit methodology. Some examples contain recent works in masking friendly block ciphers like *PICARO* [8] and *Zorro* [9]. In similar direction, researchers are trying to rewrite existing block ciphers like *AES* and *DES* with reduced number of non-linear operations [10], [11], [12].

FPGA, as an implementation platform has become hugely popular due to its features like programmability and re-

configurability. Moreover, the in-house development facility of FPGA, makes it an attractive choice for implementing cryptographic algorithms. In [3], the authors have proposed a design methodology of private circuits on FPGA.

However, strangely till now, very little practical evaluation of private circuit is present in the literature. By practical evaluation, we mean any standard cipher, implemented by private circuit and tested against SCA, which is the focal point of the present paper. In a recent work [13], t-private implementation of PRESENT with $t = 1$ was tested. Authors analyze a straightforward implementation of t-private PRESENT to fail against against CPA and correlation-enhanced collision attacks. Although the practical analysis presented in [13] is detailed, authors fail to explain the phenomena which causes side-channel leakage. The countermeasure, proposed in [1] is based on sound theoretical proof but with some inherent assumptions which may not be valid in practical scenario. In this paper, we will try to identify the practical scenarios in which private circuit may fail to provide us the desired security. We would like to state that we are not making any claim against security of private circuit, but we are pointing out the dangerous situations where expected security can be compromised. In the work [14], Balasch et al. have shown how *lazy engineering* can affect security of masking system in software. We actually try to identify the *lazy engineering* practices in hardware which can rattle the security of private circuit.

For this we have implemented a lightweight block cipher *SIMON* [15] using private circuit methodology on SASEBO-GII board. As we evaluate private circuits in a pure hardware setting, we choose to use the original ISW scheme as presented in [1] to be fair towards various schemes branched out of it. Moreover, the implementation in [3] is primarily proposed for FPGA target. However, we intend to study the security of private circuits in general. The choice of block cipher is motivated by its compact design and simplistic construction using basic gates like *AND* & *XOR*. The implemented private circuits are analyzed against SCA using EM traces and correlation power analysis [16]. Moreover, we have used *Test Vector Leakage Assessment (TVLA)* methodology [17], [18] based leakage detection to classify our design as side-channel secure or not. We primarily have implemented three different versions of *SIMON* on FPGA using private circuit, which are as follows:-

- *Optimized SIMON*: In this case, SIMON is implemented according to the ISW scheme [1], but the design tool is free to optimize the circuit. As our implementation platform is a Virtex-5 FPGA which has six-input Look-up table (LUT), design tool (in our case Xilinx ISE) will optimize the circuit to reduce the resource requirement of the design. This is an example of *lazy engineering* approach, where designer is allowing the tool to do modification on the design without being aware of its possible impact on the security of private circuits.
- 2-*input LUT based SIMON*: Here, to mimic the private circuit methodology exactly on the FPGA, we have constrained the design tool to map each two-input gate to a single LUT. In other words, though a LUT has six inputs, it is modeled as two-input gate and gate-level optimization is minimized.
- *Synchronized 2-input LUT based SIMON*: This is nearly similar to the previous methodology. The only difference is that each gate or LUT is preceded and followed by flip-flops so that each and every input to the gates is synchronized and glitches are minimized (if not suppressed, see [19]).

We will show that among these three, *Optimized SIMON* can be broken using CPA whereas, 2-*input LUT based SIMON* is resistant against CPA, but fails *TVLA* test. Finally we will show that *Synchronized* 2-*input LUT based SIMON* is not only resistant against CPA, but also passes *TVLA* test [17], [18].

We could have based our study on more popular ciphers like AES or PRESENT, but with the FPGA in consideration, it was not possible to fit protected designs when using constraints $LOCK_PINS$ and $KEEP$. In fact, we had to switch from SIMON64/96 to SIMON32/64, to be able to fit all the designs on the FPGA.

The rest of the paper is organized as follows: in section II we will give a very brief description of SIMON block cipher, SCA and private circuits. In the next section we will discuss various factors for which theoretically sound private circuit may leak in practical scenarios, which will be followed by our case-study on SIMON. Finally we will show our experimental results, discuss about provably secure logic styles in hardware, and conclude the paper.

II. Preliminaries

In this section, we will very briefly provide some background information on SIMON, side-channel analysis and t-private circuits.

A. SIMON block cipher

In 2013, NSA had introduced two ultra-lightweight block cipher SIMON and SPECK [15] with a Feistel construction. Out of the two block ciphers, SIMON is more suited for hardware implementations. SIMON can encrypt a block of $2k$ bits, with a key of $m \cdot k$ bits. Here k is the word size and m is the number of words in the key. In the following, we consider the smallest version of SIMON (*SIMON 32/64*) i.e 32-bits of block and 64-bits of key. The encryption process involves 32 identical rounds for *SIMON 32/64*. SIMON uses a simple bitwise AND between selected bits after left circular shift $S(i)$, where i is the shift value. The rest of the sub-operations are bitwise XOR. We choose the smallest version because protected SIMON can have huge overheads and we wanted to test and compare all versions of protected SIMON on the same FPGA.

B. Physical Attacks

Physical attacks or SCA try to extract secret information from a device by exploiting unintentional leakage Y (power consumption, electromagnetic radiation,...). A typical attack

is performed as follows. An attacker predicts an intermediate leakage value $L(X, K)$, for a known part of the plaintext (or ciphertext) X and key hypothesis K. Next, the attacker uses a distinguisher like Pearson's Correlation in Correlation Power Analysis (CPA [16]), to distinguish the correct key $k^\star$ from other false key hypotheses.

Alternately, it is not always possible to mount a successful attack, specially in protected implementations. In such cases, to assess the security, experts rely on leakage detection test like TVLA. TVLA consists in operating the device under test with *a fixed and chosen key*. Thereafter two sets of measurements of considerable size are acquired, one with fixed input message and the other with varying messages. Then, a T-test is applied on both sets of measurements. Similar difference testing can be performed on intermediate values of the block cipher and also on each bit of that intermediate value.

C. t-*Private Circuit*

In *t*-private circuit approach, a circuit is transformed in such a way that any adversary, having capability of observing *t* nets, cannot get access to a single bit of sensitive information. The minimum number of probes required by an adversary to extract one bit of information is *t+1*. This section provides a brief description of such transformation.

- *Input Encoding*:- Any input bit a is transformed into a vector $\grave{a}$ of $2t+1$ bits. The first $2t$ bits are random values $(a_1, a_2, \ldots, a_{2t})$ and the last bit (a_{2t+1}) is computed by the following way:

$$a_{2t+1} = a \oplus \bigoplus_{i=1}^{2t} a_i \ . \tag{1}$$

- *AND gate*: Inputs a and b of *AND* gate are transformed into vectors $\grave{a} = (a_1, a_2, \ldots, a_{2t+1})$ and $\grave{b} = (b_1, b_2, \ldots, b_{2t+1})$. Output of the *AND* gate is also a vector $\grave{c} = (c_1, c_2, \ldots, c_{2t+1})$, which is calculated by following steps:
  1) Generate random bits $r_{i,j}$, where $i \neq j$ and $1 \leq i \leq j \leq 2t+1$.
  2) Compute $r_{j,i} = (r_{i,j} \oplus a_i b_j) \oplus a_j b_i$, where $i \neq j$ and $1 \leq i \leq j \leq 2t+1$.
  3) Compute $c_i = a_i b_i \oplus \bigoplus_{j \neq i} r_{i,j}$, where $1 \leq i \leq 2t$ and $1 \leq j \leq 2t$.
- *NOT gate*: Input a is transformed into a vector $\grave{a} = (a_1, a_2, \ldots, a_{2t+1})$. Output $\grave{\bar{a}}$ is computed by inverting any bit of $\grave{a}$. E.g., $\grave{\bar{a}} = (\overline{a_1}, \overline{a_2}, \ldots, \overline{a_{2t+1}})$.
- *XOR gate*: Like *AND* gate; inputs a and b of *XOR* gate are transformed into vectors $\grave{a} = (a_1, a_2, \ldots, a_{2t+1})$ and $\grave{b} = (b_1, b_2, \ldots, b_{2t+1})$. Output $\grave{c} = (c_1, c_2, \ldots, c_{2t+1})$ is calculated in the following way [2]. Perform:

$$c_i = a_i \oplus b_i, 1 \leq i \leq 2t \ . \tag{2}$$

Using these transformations, one can easily transform any digital circuit to a *t*-private circuit, because this set of gates is universal.

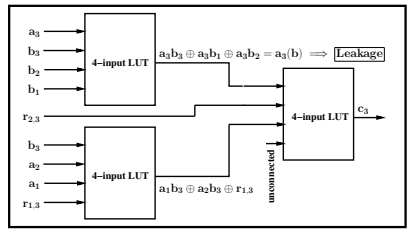


Fig. 1: $t = 1$ private circuit for AND third coordinate on 4-input LUTs

In the following section, we will deal with private circuits qualitatively. More precisely we will look into the security pitfalls of *t*-private circuits which may easily exist in a real implementation.

III. Optimization and Delay: Ambushes for Private Circuit

Private circuits were proposed as a countermeasure against physical attacks, with a sound theoretical proof. However, much alike cryptographic algorithms, physical countermeasures suffer a risk of implementation pitfalls. These pitfalls can arise from various reasons. In the rest of the section, we highlight two main reasons, that could cause failure of private circuit in practice i.e. become susceptible to side-channel leakage. These two factors are *CAD optimization* and *delay in random variables*. Please note that it is possible for an experienced designer to implement private circuits properly without any optimizations. This non-optimized implementation would incur significant overheads in area and performance as shown later.

A. CAD Optimizations

Circuit optimization is the primary objective of any CAD design tool. This optimization is all the more serious where a generic netlist is mapped to FPGA building blocks like configurable logic blocks (CLBs). Given a design, any CAD tool for FPGA will try to reduce the CLB utilization and improve its timing performance. Though the optimization is a highly useful property of CAD tools, it can be a disaster for private circuits from security point of view. We will highlight this phenomenon with the following example.

Let us consider an *AND* gate in t-private circuit for $t = 1$. Inputs of the *AND* gate are two vectors $\grave{a} = (a_1, a_2, a_3)$ and $\grave{b} = (b_1, b_2, b_3)$, encoded according to equation (1). Output $\grave{c} = (c_1, c_2, c_3)$ is calculated as follows:

$$c_1 = a_1 b_1 \oplus r_{1,2} \oplus r_{1,3} \tag{3}$$

$$c_2 = a_2 b_2 \oplus (r_{1,2} \oplus a_1 b_2) \oplus a_2 b_1 \oplus r_{2,3} \tag{4}$$

$$c_3 = a_3 b_3 \oplus (r_{1,3} \oplus a_1 b_3) \oplus a_3 b_1 \oplus (r_{2,3} \oplus a_2 b_3) \oplus a_3 b_2 \tag{5}$$

In the equations (3), (4) and (5), the order of the computation is very important, which is protected using parentheses. If the order of the computation is not maintained, information leakage can occur, defeating the very purpose of private circuits. However, when an FPGA design tool maps the above equations in the LUTs of FPGA, it will try to minimize

the LUT utilization ratio and may not necessarily follow the correct order. For example equation (5) can be mapped into the 4-input LUTs without maintaining the desired order of computation, as shown in Fig. 1.

This is one of the many possible ways in which equation (5) can be mapped into LUTs, but with wrong computation order. This leads to a leakage: consider for instance the output $x = a_3b_3 \oplus a_3b_1 \oplus a_3b_2$ of the top LUT. Clearly, x simplifies to a_3b, and this intermediate variable depends on b.

$$\begin{cases} p(b=0|x=0) & = 2/3, \\ p(b=1|x=0) & = 1/3, \end{cases} \quad \text{and} \quad \begin{cases} p(b=0|x=1) & = 0, \\ p(b=1|x=1) & = 1. \end{cases} \tag{6}$$

So, when observing the value carried by x, some information about the clear (unmasked) bit b is recovered. Namely, when x is measured equal to 0, it is more likely that b is equal to 0. Moreover, when x happens to be 1, then the attacker knows that b is 1 (with 100% probability). Similar observations can be found for 6-input LUTs also. The optimization that is shown in the above example can be removed by modeling the LUTs as 2-input gates only, however with added overhead. As an example, we set Xilinx ISE to maximum area optimization and synthesized an *AND* for $t = 1$ *AND* gate. The *AND* gate can be implemented using 4 *LUT6* i.e. one to compute c_1, another to compute c_2 and two *LUT6* to compute c_3. However, if we just look at the equations, we can easily count that 21 gates or LUTs are required to implement an *AND* gate without any security pitfalls.

FPGA CAD tools are designed to achieve a desired area or timing optimization in a design, whereas for private circuits, security is the primary concern rather than area or performance. As of now, the user cannot constrain the FPGA CAD tools to respect orders in the combinational variables processing (actually, such feature would make little sense for most applications). So, in the sequel, we manually generate netlists.

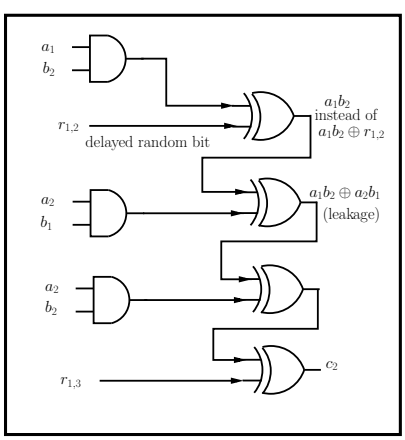


Fig. 2: Impact of Delay on Private Circuit

B. Delay in Random Variables

Another threat on private circuits is delays and glitches in the random variablesRandom variables are extremely important for security of private circuits. In the analysis of private circuit in [1], it is assumed that all the inputs along with the random variables are synchronized, which may not be valid in practical situations. There are two ways in which random variables can be provided to the private circuit: as external input or from a *Random Number generator (RNG)*. Generally, random numbers are provided to the circuit from an *RNG*, as handling large number of inputs is difficult on FPGAs due to limited number of available I/O pins. Now, RNG itself is a complex circuit like a stream cipher. Therefore the random numbers generated by *RNG* may not arrive at the same time as the masked inputs of the circuit and lead to information leakage. For example, in equation (5), delay in the arrival of random bits $r_{1,3}$, $r_{2,3}$, a_1 and a_2 lead to information leakage. A delay in arrival of random inputs is equivalent to absence of that signal initially. In the absence of random variables, the security of private circuits is not proven. This phenomena is illustrated in Fig. 2. Indeed, the evaluation of equation (5) with, say, $r_{1,3} = r_{2,3} = a_1 = a_2 = 0$ (since late and keeping previous values) can lead to the evaluation of $x = a_3b_3 \oplus a_3b_1 \oplus a_3b_2$, which discloses the value of (unmasked) bit b (recall equation (6)).

The crucial idea is that, even if gates are (*statically*) placed in an *unoptimized* order that respects the parentheses (recall equation (5)), they may well evaluate (*dynamically*) in a different order due to delays in the inputs arrival. In next section, we are going to experimentally validate the security of private circuits in a real implementation. For this we have implemented block cipher *SIMON* with private circuit methodology. We will show how *optimization* and *delay in random variables* can disrupt the security of private circuit and make *SIMON* vulnerable to side-channel attack by performing side-channel analysis on the implemented *SIMON*.

We would like to emphasize that the two presented pitfalls are not the only pitfalls failing private circuits. However, as digital designers we clearly recognize these two pitfalls which can impact the security of private circuits and thus propose some tricks to fix the presented issues.

IV. Experimental Analysis and Result

In this section, we perform practical evaluation of *private circuit* by executing SCA on block cipher *SIMON*. We start with the experimental setup, followed by the analysis of the obtained result on various implementation of private circuits.

A. Experimental Setup

A parallel implementation SIMON32/64 crypto-core, running at clock frequency of 24-MHz, along with a simple UART interface is used to test our design on the Xilinx Virtex `XC5-VLX30` FPGA of the SASEBO-GII platform. Both plaintext and key are coded according to the equation (1). *Round function*, along with *Key schedule* are coded as per *t*-private circuits as shown in Fig. 3. As we mentioned previously, t-private circuits need $2t$ bits of random bits for each input and $2t + 1$ bits for each and gate in the circuit. Thus for implemented SIMON with $t = 1$, total number of random bits required by SIMON is 272, whereas for $t = 2$ and

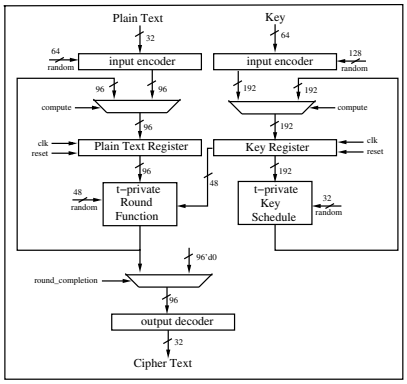


Fig. 3: Architectural Overview of SIMON

$t = 3$, number of required random bits become 608 and 1008. Random numbers are generated by a maximal length LFSR.

To perform SCA, side-channel traces are acquired using a EM antenna of the HZ–15 kit from Langer, placed over a power decoupling capacitor. The traces are captured on a 54855 Infiniium Agilent oscilloscope at a sampling rate of 2 GSample/s. In the following subsections we will discuss the result of SCA on different variants of *t-private SIMON*. The main objective of the following analysis is that the proposed tricks result is reduction of side-channel leakage.

B. Optimized SIMON

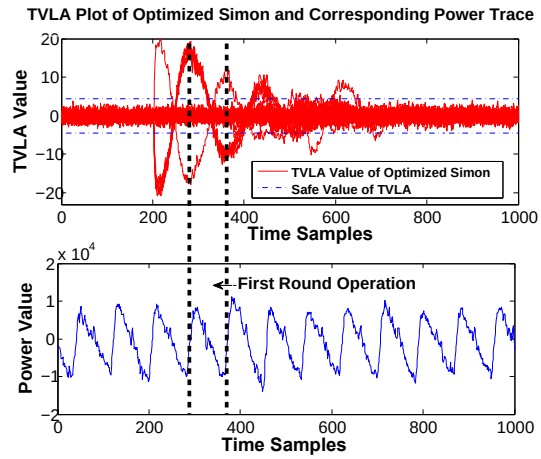
In this setting, we implemented SIMON using private circuit methodology with $t = 1$. At this stage, the FPGA tool were not constrained and were allowed to do all optimizations if possible. The area and performance figures are presented in Tab. II. We collected 100000 EM traces and performed *TVLA* test, along with *CPA*. *TVLA* test is performed on the first round output of *SIMON* for each bit of plaintext. Basically, the traces are partitioned into 2 sets on the basis of value of concerned bit of round output i.e. 1 or 0. Thereafter, we compute the means (μ_1, μ_0) and standard deviations (σ_1, σ_0) of the two sets. The T-test is calculated as $T = \frac{\mu_1 - \mu_0}{\sqrt{\frac{\sigma_1^2}{N_1} + \frac{\sigma_0^2}{N_0}}}$, where N_1 and N_0 are the number of traces in each set.

The corresponding result can be seen in Fig. 4(a). The first few peaks in the plot are due to the plaintext loading, however after that it can be seen that for some bits, value of TVLA is much larger than the safe value of TVLA i.e. ± 4.5 for secure system [17], [18]. Thus this implementation of SIMON is vulnerable to SCA though it is designed by private circuit. The leakage comes due to the optimizations applied by CAD tools. To validate our claim further, we carried out CPA around the sample points where TVLA peaks can be observed. We have chosen Hamming-distance as our attack model and we targeted first round of *SIMON32/64*. The leakage model can be written as follow:

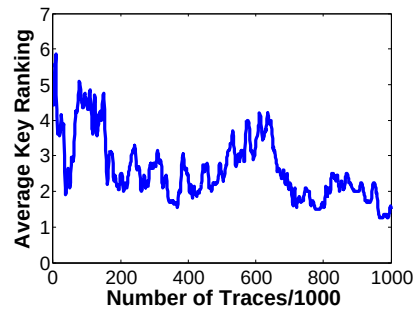
$$L_{HD} = \mathsf{HW}[R(x_{i+1}, x_i) \oplus key_i \oplus x_{i+1}],$$

where x_i, x_{i+1} are parts of plaintext, key_i is the round key and R is the round function. The first round operation of

(a) TVLA Plot



(b) Average Key Ranking



(c) Correlation Value of Key Guesses

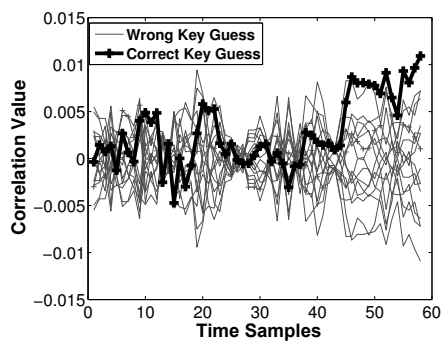


Fig. 4: Side-Channel Analysis of Optimized SIMON

SIMON32/64 involves 16 bits of key and we try to recover key nibble by nibble. Now, generally for a nibble, key space is 16; however, as SIMON has no non-linear operation on the key in first round, for each key guess there is another key candidate which has exactly same value of correlation coefficient with opposite polarity. Due to this symmetry, total key space reduces from 16 to 8. To measure the success rate, we have calculated *average key ranking* over all the nibbles at intervals of 1000 power traces and the corresponding result is shown in Fig. 4(b). At the end of 100000 traces, we have been able to recover the correct key for two nibbles and for the rest of the two nibbles, ranking of correct key is 2,

which clearly shows successful attack. Fig. 4(c) shows the correlation values of different key guesses. Though the gap between the correct key and wrong key guesses is marginal, it is consistent as indicated by Fig. 4(b), providing us enough evidence to conclude the attack as successful. The reason for this small nearest rival distance is the Hamming-distance attack model which does not incorporate randomization of private circuit. The key ranking curve is not very smooth. Theoretically addition of traces should lower the key ranking, but the reverse phenomenon can happen in practical scenario, as it depends upon the quality of the acquired traces. CPA is a statistical analysis and this phenomenon is statistical artifact. Moreover, the average key ranking will depends on the combined progression of all the 4 nibbles. Nevertheless, due to the CAD tool optimization, it is possible to successfully retrieve secret information from private circuit.

C. 2-input LUT based SIMON

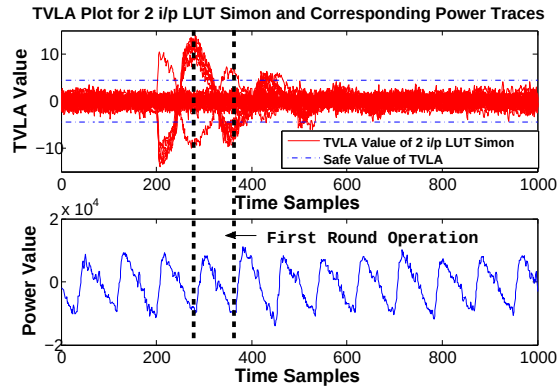
In the previous subsection, we provided experimental validations of the disastrous effect of CAD tool optimization on private circuits. Designer can use various attributes and constraints to disable the optimization property of CAD tools, and hence can mimic the private circuit more efficiently (when the input HDL file is described structurally with LUTs). In this implementation, we have constrained the Xilinx ISE tool to treat each LUT as a 2-input gate using `KEEP` and `LOCK_PIN` attributes. *SIMON 32/64* is then designed using this 2-input LUT gates so that no optimizations can be applied by the CAD tools. The area and performance figures are given in Tab. II. Similar SCA is carried out for this implementation also and corresponding result is shown in Fig. 5. As we can see, TVLA plot still shows occurrence of information leakage, though it is less significant compared to information leakage observed for *Optimized SIMON*. Average key ranking plot (Fig. 5(b)) also shows improvement in terms of more resistance against SCA. However, as there is still information leakage, as shown in TVLA plot, the system could be broken by attack which employs better model. In other words, we cannot be absolute confident about the side-channel resistance of the implementation.

D. Synchronized 2-input LUT based SIMON

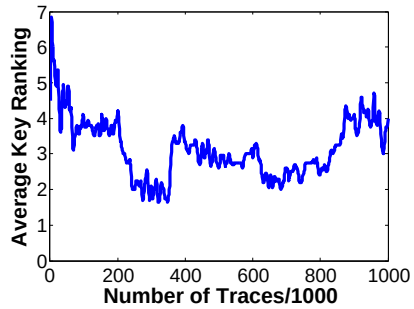
In the previous subsection, we have shown how side-channel attack is resisted by 2-input LUT based SIMON. However, TVLA plot of 2-input LUT based SIMON still shows some information leakage. In this subsection, we tackle this issue.

We have shown in section III how delay in random variables can disrupt the security of private circuit. In our implementation, random variables are generated using a maximum length LFSR and we suspect that asynchronous random variables as the reason of the information leakage. To analyze this, we have made a new implementation where each 2-input LUT is followed by a flip-flop[1], so that if there is any delay or glitches on random variables, it will be handled locally and will not propagate across the circuit. This is a resource greedy implementation as shown in Tab. II. We have carried out similar side-channel analysis on this implementation and the result can be seen in Fig. 6. Now before analyzing the result we will like to state that this implementation has more clock cycle requirement compared to previous two implementations due to the presence of flip-flops after each 2-input LUT. For example, in the previous two implementations, first round operation occurs just after the start of encryption, whereas in this implementation first round output is obtained in the 9-th clock cycle after the start of encryption. As we are doing TVLA test on the first round output of SIMON, TVLA peak

(a) TVLA Plot



(b) Average Key Ranking



(c) Correlation Value of Key Guesses

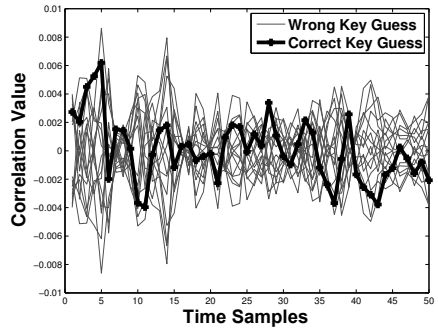
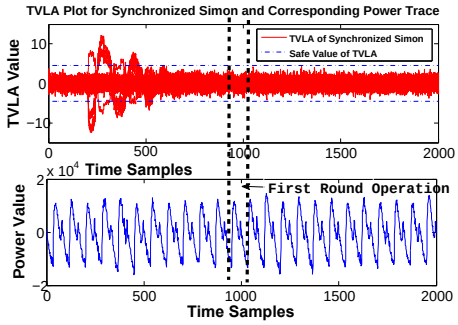


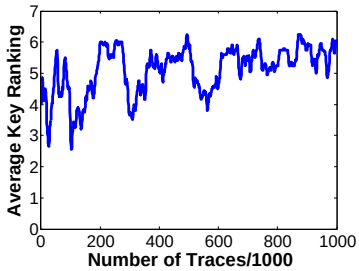
Fig. 5: Side-Channel Analysis of 2-input LUT SIMON

[1]In the *ISW* scheme [1], logic gates of *stateless circuits* are combinational instances, which, by design, evaluate as soon as one input changes. Our addition of the flip-flop after each combinational gate ensures that they evaluate only once, which is the legal mode of operation for *ISW* to be sound.

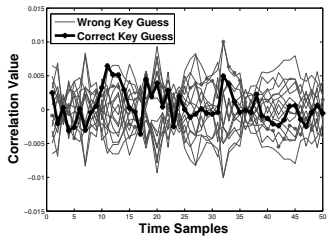
(a) TVLA Plot



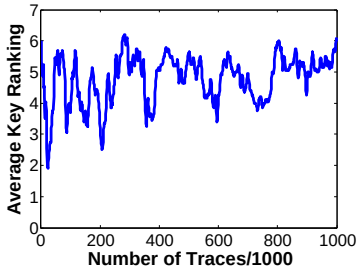
(b) Avg. Key Rank (TVLA Peak)



(c) Correlation Value at TVLA Peak



(d) Avg. Key Rank (first round)



(e) Correlation Value at first round

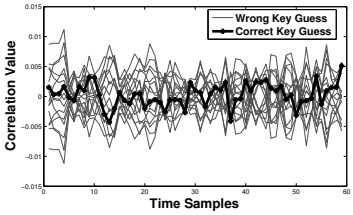


Fig. 6: Side-Channel Analysis of Synchronized 2-input LUT SIMON

TABLE I: Summary of Side-Channel Analysis

Design Name	TVLA Test	Max. TVLA Leakage	Avg. Key Ranking	Remarks
Optimized SIMON	Fails, significant information leakage	18	Key ranking is low,successful attack	Not secure
2-input LUT based SIMON	Fails, but less information leakage	12	Key ranking is high,attack fails	Secure against CPA with HD model
Synchronized 2-input LUT based Simon	Passes: no leakage at first round	3.5	Key ranking is high, attack fails	Secure

TABLE II: Performance and Resource Requirement Comparison

Name	LUTs	Registers	Slices	Freq. (MHz)	Clock Cycles
Unprotected SIMON	218 (1×)	165 (1×)	107 (1×)	625 (1×)	32 (1×)
Optimized SIMON	761 (3.49×)	805 (4.88×)	595 (5.56×)	147 (0.23×)	32 (1×)
2-input LUT based SIMON	1305 (5.99×)	805 (4.88×)	1241 (11.60×)	88 (0.14×)	32 (1×)
Synchronized 2-input LUT based SIMON	1309 (6.00×)	17.70 (3.62×)	4090 (38.22×)	104 (0.16×)	288 (9×)

should appear near the clock-cycle where first round output is obtained. As we can see, in TVLA plot (Fig. 6(a)), there are few peaks near the start of encryption which indicates plaintext loading, but no peak at the first round operation. However, to eliminate any ambiguity, we have performed CPA around both the first round operation and TVLA peak. Result shows that in both cases, side-channel attack is not working and implementation under attack can be considered secure against side-channel attack. Thus in this case, theoretically secure private circuit is translated into practically secure private circuit implementation. The summary of our experimental Analysis is shown in Tab. I.

E. Resource Comparison

In this subsection, we will compare our three implementation from the perspective of resource requirement and performance. The comparison is shown in Tab. II. First of all, we notice that the overhead is larger than regular masking ([20] announces $\approx$ 50% more area and $\approx$ 15% less frequency). As we can see, among the three implementations, *Optimized SIMON* requires least resources and provides better timing performance. On the other hand, 2-*input LUT based SIMON* and has a moderate resource overhead and exhibits little poor timing performance. But *Synchronized* 2-*input LUT based SIMON* has a huge area overhead along with extremely poor timing performance. The reason of its poor timing performance is the routing constraints that have been applied to preserve the integrity of the private circuit. Higher version of SIMON can not be mapped in the FPGA due to this reason. Though *Synchronized* 2-*input LUT based SIMON* shows poor performance, it is most secure against side-channel attacks.

V. Cautionary Note / Discussion

In this case-study about SIMON implemented using private circuits, we have shown that it is feasible to achieve the expected security level in practice. However, in the meantime, we have come across two ambushes:

1) optimization by synthesizers, which basically break the countermeasure, and
2) misinterpretations of the specifications, for instance by letting gates evaluate in *early*, as soon as one input changes, instead of waiting for all inputs.

Clearly, such issues may show up in any secure design flow. They might be undetected unless suitable verification tools check the correctness of the implementation. For first order masking schemes, the TVLA approach allows a methodical leakage detection on the running implementation. Static verification is a complementary approach, which should ideally be applied before verification in hardware (FPGA or ASIC). Verification based on SAT-solvers [21], [22] and formal methods [23] have shown up for software implementations of masking. However, similar efforts applied to hardware would definitely help. Hardware verification is more tricky because the evaluation order of the combinational gates is not statically known.

Besides, we also emphasize that secure compilation of circuits is definitely another direction to explore. In our case study, the netlist has been hand-written and mapped onto secure gates. However, for larger designs, one would badly need an automatic countermeasure insertion tool. Initial works exist for application of masking to software applications [24], [25]. Transposition to hardware masking would definitely be welcomed.

But, in summary, we learned from our case-study that secure masking in hardware is possible both in theory and in practice, provided the implementation is generated rigorously, so as to avoid ambushes (which are all well known, now), and carefully checked for pitfalls.

VI. Conclusion

Private circuits, proposed as a countermeasure against strongest side-channel adversaries, is based on sound theoretical proof. In this paper, we analyzed private circuits at an implementation level on a SIMON crypto-processor. Our results show that it is very easy for a CAD tool to override the basic requirements of private circuits. Therefore proper framework and if possible specific CAD tools are required to implement such strong but complex countermeasures.

Practical evaluations indicate that with proper constraints the leakage can be reduced. Moreover, by synchronizing each gate, we remove glitches and delay and approach much closer to theoretical evaluation of private circuits, but at a huge overhead. Further works would focus on more efficient implementation of private circuit without compromising on security.

References

[1] Y. Ishai, A. Sahai, and D. Wagner, "Private Circuits: Securing Hardware against Probing Attacks," in *In Proceedings of CRYPTO 2003*. Springer-Verlag, 2003, pp. 463–481.

[2] M. Rivain and E. 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*, 2010, pp. 413–427.

[3] J. Park and A. Tyagi, "t-Private logic synthesis on FPGAs," in *Hardware-Oriented Security and Trust (HOST), 2012 IEEE International Symposium on*, June 2012.

[4] "t-Private Systems: Unified Private Memories and Computation," in *Security, Privacy, and Applied Cryptography Engineering*, ser. Lecture Notes in Computer Science, R. Chakraborty, V. Matyas, and P. Schaumont, Eds., vol. 8804, 2014.

[5] M. Gomathisankaran and A. Tyagi, "Glitch resistant private circuits design using HORNS," in *IEEE Computer Society Annual Symposium on VLSI, ISVLSI 2014, Tampa, FL, USA, July 9-11, 2014*, 2014, pp. 522–527.

[6] J. Park and A. Tyagi, "Towards making private circuits practical: DPA resistant private circuits," in *IEEE Computer Society Annual Symposium on VLSI, ISVLSI 2014, Tampa, FL, USA, July 9-11, 2014*, 2014, pp. 528–533.

[7] "Protecting circuits from leakage: the computationally-bounded and noisy cases," in *Advances in Cryptology — EUROCRYPT 2010*, ser. Lecture Notes in Computer Science, H. Gilbert, Ed., vol. 6110, 2010.

[8] G. Piret, T. Roche, and C. Carlet, "PICARO - A block cipher allowing efficient higher-order side-channel resistance," in *Applied Cryptography and Network Security - 10th International Conference, ACNS 2012, Singapore, June 26-29, 2012. Proceedings*, 2012, pp. 311–328.

[9] B. Gérard, V. Grosso, M. Naya-Plasencia, and F. Standaert, "Block ciphers that are easier to mask: How far can we go?" in *Cryptographic Hardware and Embedded Systems - CHES 2013 - 15th International Workshop, Santa Barbara, CA, USA, August 20-23, 2013. Proceedings*, 2013, pp. 383–399.

[10] C. Carlet, L. Goubin, E. Prouff, M. Quisquater, and M. Rivain, "Higher-order masking schemes for s-boxes," in *Fast Software Encryption - 19th International Workshop, FSE 2012, Washington, DC, USA, March 19-21, 2012. Revised Selected Papers*, 2012, pp. 366–384.

[11] V. Grosso, E. Prouff, and F. Standaert, "Efficient masked s-boxes processing - A step forward -," in *Progress in Cryptology - AFRICACRYPT 2014 - 7th International Conference on Cryptology in Africa, Marrakesh, Morocco, May 28-30, 2014. Proceedings*, 2014, pp. 251–266.

[12] J. Coron, J. Großschädl, and P. K. Vadnala, "Secure Conversion between Boolean and Arithmetic Masking of Any Order," in *Cryptographic Hardware and Embedded Systems - CHES 2014, Busan, South Korea, September 23-26, 2014. Proceedings*, 2014, pp. 188–205.

[13] Z. N. Goddard, N. Lajeunesse, and T. Eisenbarth, "Power Analysis of the t-Private Logic Style for FPGAs," in *HOST*, ser. IEEE Computer Society, June 2-3 2015, .

[14] J. Balasch, B. Gierlichs, V. Grosso, O. Reparaz, and F.-X. Standaert, "On the Cost of Lazy Engineering for Masked Software Implementations," *IACR Cryptology ePrint Archive*, vol. 2014, p. 413, 2014.

[15] R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks, and L. Wingers, "The SIMON and SPECK Families of Lightweight Block Ciphers," Cryptology ePrint Archive, Report 2013/404, 2013.

[16] É. Brier, C. Clavier, and F. Olivier, "Correlation Power Analysis with a Leakage Model," in *CHES*, ser. LNCS, vol. 3156. Springer, August 11–13 2004, pp. 16–29, Cambridge, MA, USA.

[17] G. Goodwill, B. Jun, J. Jaffe, and P. Rohatgi, "A testing methodology for side-channel resistance validation," September 2011, NIST Non-Invasive Attack Testing Workshop, http://csrc.nist.gov/news_events/non-invasive-attack-testing-workshop/papers/08_Goodwill.pdf.

[18] J. Cooper, G. Goodwill, J. Jaffe, G. Kenworthy, and P. Rohatgi, "Test Vector Leakage Assessment (TVLA) Methodology in Practice," Sept 24–26 2013, International Cryptographic Module Conference (ICMC), Holiday Inn Gaithersburg, MD, USA.

[19] A. Moradi and O. Mischke, "Glitch-free Implementation of Masking in Modern FPGAs," in *HOST*, ser. IEEE Computer Society, June 2-3 2012, pp. 89–95.

[20] S. Bhasin, T. Graba, J. Danger, and Z. Najm, "A look into SIMON from a side-channel perspective," in *2014 IEEE International Symposium on Hardware-Oriented Security and Trust, HOST 2014, Arlington, VA, USA, May 6-7, 2014*. IEEE Computer Society, 2014, pp. 56–59.

[21] A. G. Bayrak, F. Regazzoni, D. Novo, and P. Ienne, "Sleuth: Automated Verification of Software Power Analysis Countermeasures," in *Cryptographic Hardware and Embedded Systems - CHES 2013 - 15th International Workshop, Santa Barbara, CA, USA, August 20-23, 2013. Proceedings*, 2013, pp. 293–310.

[22] H. Eldib, C. Wang, M. Taha, and P. Schaumont, "QMS: Evaluating the Side-Channel Resistance of Masked Software from Source Code," in *Proceedings of the The 51st Annual Design Automation Conference on Design Automation Conference*, ser. DAC '14, 2014, pp. 209:1–209:6.

[23] G. Barthe, S. Belaïd, F. Dupressoir, P.-A. Fouque, B. Grégoire, and P.-Y. Strub, "Verified Proofs of Higher-Order Masking," EUROCRYPT 2015, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I

[24] A. Moss, E. Oswald, D. Page, and M. Tunstall, "Compiler Assisted Masking," in *CHES*, 2012, pp. 58–75.

[25] A. G. Bayrak, F. Regazzoni, D. Novo, P. Brisk, F. Standaert, and P. Ienne, "Automatic Application of Power Analysis Countermeasures," *IEEE Trans. Computers*, vol. 64, no. 2, pp. 329–341, 2015.