TVLA, Correlation Power Analysis and Side-Channel Leakage Assessment Metrics

William Unger, *Boise State University*
Liljana Babinkostova, *Boise State University*
Mike Borowczak, *University of Wyoming*
Robert Erbes, *Idaho National Laboratory*
Aparna Srinath, *Boise State University*

***Abstract*—We provide insights into the data leakage of GIFT-COFB [1] by performing Correlation Power Analysis (CPA) on GIFT-64 based on the Hamming Weight model. We assess the reliability of several existing theoretical measures in identifying resiliency to a CPA attack by making a quantitative comparison with the outcomes of these metrics for the SBoxes of PICCOLO and PRESENT, as well as with several other SBox variants that demonstrated sufficient weaknesses against cryptanalysis.**

The existing theoretical metrics that we analyze include transparency order, revisited transparency order, signal-to-noise ratio, DPA signal-to-noise ratio and non-linearity aiming to characterize the CPA resistance of the above mentioned SBoxes.

We utilize the LWC and CAESAR hardware implementation [2] of GIFT-COFB by Rezvani et. al [1] and apply TVLA using the Flexible Open-source workBench fOr Side-channel analysis (FOBOS)[2] test architecture [3].

***Index Terms*—Correlation Power Analysis, Transparency Order, Non-linearity, Signal-to-Noise Ratio**

I. Introduction

Side-channel attacks, introduced in 1996 by P. Kocher [4], exploit side-channel leakages such as power consumption from a device to extract secret information. Since Kocher's original paper, a large number of very efficient side-channel attacks has been reported on a wide variety of cryptographic implementations [5]–[8].

A more general approach for determining if the power consumption of a device relates to the data it is manipulating, referred to as Test Vector Leakage Assessment (TVLA), was proposed by Goodwill et al. [9]. Although TVLA itself is not sufficient for a comprehensive security evaluation, it is often considered as a first test to guide further evaluations.

Our Contributions: Our contributions in this paper are three-fold: First, we provide insights on the data leakage of GIFT-COFB [1] by performing Correlation Power Analysis (CPA) on GIFT-64 based on the Hamming Weight model.

Second, we relate the CPA results with existing theoretical metrics such as Transparency Order (TO) [7], Revisited Transparency Order (RTO) [10], Signal-To-Noise Ratio [11], DPA Signal-To-Noise Ratio (DPA-SNR) [12], and Non-Linearity [13].

Work supported through the INL Laboratory Directed Research & Development (LDRD) Program under DOE Idaho Operations Office Contract DE-AC07-05ID14517.

[1] SAL: NIST Lightweight Cryptography Implementations (Jan 2019), https:// github.com/vtsal

[2] FOBOS: https://cryptography.gmu.edu/fobos/

Third, we utilize the LWC and CAESAR hardware implementation [2] of GIFT-COFB by Rezvani et. al [3] and apply TVLA using the Flexible Open-source workBench fOr Side-channel analysis (FOBOS)[4] test architecture [3].

The paper is organized as follows. Section II provides background on several theoretical metrics used. Section III defines our test environment and describes our specific implementation of the CPA attack. In Section III-D we provide the results related to CPA and the above mentioned metrics. The paper concludes with Section IV where we provide a description of the experimental setup for the TVLA analysis and provide the experimental results of this analysis.

II. Background

Let $\mathbb{F}_2^n$ be the vector space that contains all the n-bit binary vectors, where n is a positive integer. We shall use the symbol $\oplus$ to denote the addition operation in the two-element field $\mathbb{F}_2$, and the symbol $+$ for the standard addition operation on real numbers or integers. The inner product of two binary vectors $\mathbf{a}, \mathbf{b} \in \mathbb{F}_2^m$ is defined as $\mathbf{a} \cdot \mathbf{b} = \bigoplus_{i=1}^{m} a_i \cdot b_i$. Formally, an S-Box can be viewed as (n, m)- Boolean function $F : \mathbb{F}_2^n \mapsto \mathbb{F}_2^m$ that maps n input bits to m output bits. When $m = 1$ an (n, m) Boolean function is simply called a Boolean function.

The *Walsh transform* of the Boolean function F is denoted by W_F and is an integer valued function over $\mathbb{F}_2^n$ which is defined as follows: For each $u \in \mathbb{F}_2^n$,

$$W_F(u) = \sum_{x \in \mathbb{F}_2^n} (-1)^{F(x) \oplus x \cdot u}.$$

The *auto-correlation transform*, $\mathcal{A}_F$, of the function F with respect to u is defined as

$$\mathcal{A}_F(u) = \sum_{x \in \mathbb{F}_2^n} (-1)^{F(x) \oplus F(x \oplus u)}.$$

The *cross-correlation spectrum* between two Boolean functions F_1, F_2 is denoted $\mathcal{C}_{F_1,F_2}$, and is defined as the value

$$\mathcal{C}_{F_1,F_2}(u) = \sum_{x \in \mathbb{F}_2^n} (-1)^{f_1(x) \oplus f_2(x \oplus u)}$$

for every $u \in \mathbb{F}_2^n$ [5].

[3] SAL: NIST Lightweight Cryptography Implementations (Jan 2019), https:// github.com/vtsal

[4] FOBOS: https://cryptography.gmu.edu/fobos/

[5] Note that $\mathcal{C}_{f,f}(u) = \mathcal{A}_f(u)\big)$.

The Walsh transform of an (n, m)-Boolean function F is the function which maps any ordered pair $(u, v) \in \mathbb{F}_2^n \times \mathbb{F}_2^m$ to the value at u of the Walsh transform of the "component" function $v \cdot F$, that is, $\sum_{x \in \mathbb{F}_2^n} (-1)^{v \cdot F(x) \oplus u \cdot x}$.

Transparency Order (TO) [7]. Let n, m be positive integers and let F be an (n, m)-Boolean function. The transparency order of F, is defined as

$$TO(F) = \max_{\beta \in \mathbb{F}_2^m} (|\mathrm{m} - 2\mathrm{H}(\beta)| - \frac{1}{2^{2\mathrm{n}} - 2^{\mathrm{n}}} \sum_{\mathrm{a} \in \mathbb{F}_2^{\mathrm{n}*}} |\sum_{\mathrm{i}=1}^{\mathrm{m}} (-1)^{\beta_{\mathrm{i}}} \mathcal{A}_{\mathrm{F_i}}(\mathrm{a})|) \quad (1)$$

where $\beta \in \mathbb{F}_2^m$ denotes the register initial state which is assumed to be constant.
In order to provide a better quantitative security criterion another form of TO was proposed in [10].

Revisited Transparency Order (RTO) [10]. For positive integers n, m let F be an (n, m)-Boolean function (balanced). The revisited transparency order of F is defined as

$$RTO(F) = \max_{\beta \in \mathbb{F}_2^m} (m - \frac{1}{2^{2n} - 2^n} \sum_{a \in \mathbb{F}_2^{n*}} |\sum_{j=1}^{m} \sum_{i=1}^{m} (-1)^{\beta_i \oplus \beta_j} \mathcal{C}_{F_i, F_j}(a)|) \quad (2)$$

The *Non-Linearity* of (n, m)-Boolean function F [13] is defined as

$$NL(F) = 2^{n-1} - \frac{1}{2} \max_{u \in \mathbb{F}_2^n, v \in \mathbb{F}_2^{n*}} |W_F(u, v)| \quad (3)$$

This metric is used to measure resistance against linear cryptanalysis [14] and has a relationship to the SCA success rate [14].
The *Signal to Noise Ratio (SNR)* [11] is a probabilistic measurement (commonly expressed in decibels as $20 \log(SNR)$) of the quotient of the signal and noise in a cryptographic implementation

$$SNR = \frac{Var(Signal)}{Var(Noise)} \quad (4)$$

The *DPA Signal-To-Noise Ratio (DPA-SNR)* [12] of an (n, m)-Boolean function F is defined as

$$DPA - SNR(F) = n2^n \left(\sum_{a \in \mathbb{F}_2^n} \left(\sum_{i=0}^{n-1} \left(\sum_{x \in \mathbb{F}_2^n} (-1)^{F_i(x) + x \cdot a} \right) \right) \right) \quad (5)$$

III. Implementation of Correlation Power Analysis on GIFT-64

A. Correlation Power Analysis (CPA)

Correlation Power Analysis (CPA) [6] aims at recovering the secret key k given a set of N power traces and the corresponding sets of the intermediate values $(V_{i,k})_{1 \leq i \leq N}$, using a correlation factor between the measured power samples $(H_{i,k})1 \leq i \leq N$ and the power model of the computed sensitive values. The Hamming weight (HW) model and the Hamming distance (HD) model are the most common power models used to describe the hypothetical power consumption of the target device as a function of the intermediate value. The HW model is more common for software implementations, whereas the HD model is generally used for hardware implementations. The Success Rate (SR) metric is used in order to quantify the amount of effort required to recover the correct key. The general framework consists of the following steps [15]:

- Identify Point of Interest (POI)
- Capture Power Traces
- Compute Intermediary Values
- Classify Hypothetical Power Consumption
- Compare Measurements with Predictions using the Pearson's Correlation Coefficient

B. Experimental Setup

The point of interest (POI), also known as the leakage point, is chosen to be after the S-Box function in rounds 2, 3, 4, and 5 of the cipher. The S-Box output in rounds 2, 3, 4, and 5 is chosen as the leakage point (i.e. point of interest).

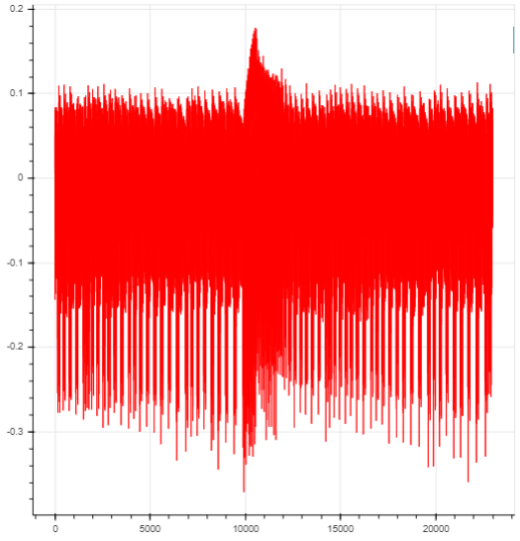

Fig. 1. A sample voltage trace.

A correlation is computed for each possible value of a targeted sub-key used in the round, and the sub-key with the highest correlation becomes the predicted value for that sub-key. As the secret key content is introduced at the end of each round, we target the first round key by predicting the Hamming Weight of the SBox output in the second round. This is repeated for all of the sub-keys of a given round key, and for as many round keys as needed in order to recover the 128-bit private key.

A sample voltage capture for execution of a portion of GIFT-64 on the XMEGA is shown in Figure 1. The x-axis

indicates time increments and the y-axis indicates the voltage reading at each point in time. In this figure the spike in the trace corresponds to the ending of one round and the beginning of another. By choosing sub-key values we can then predict what the hypothetical values will be at the point of interest and use the Hamming weight of those values for the correlation calculations to rank the possible sub-keys.

C. Execution Environment

We used the ChipWhisperer ecosystem [16] to provide the device under test (DUT), control board, and oscilloscope. The DUT was hosted on the ChipWhisperer CW308 UFO board and consisted of an ATXMEGA128D4 8-bit RISC micro-controller. The hardware environment used for data collection in shown in Figure 2.

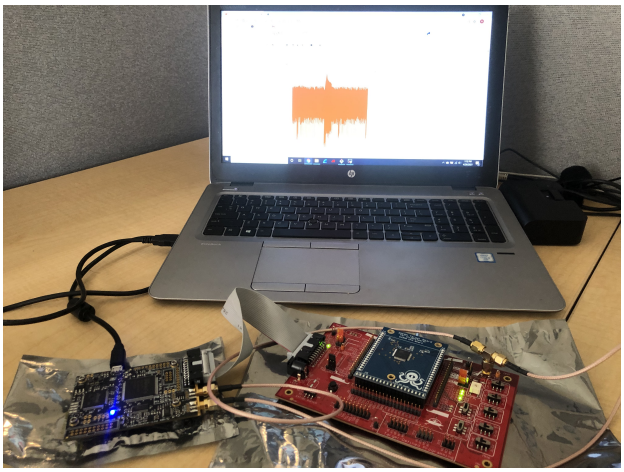

Fig. 2. Execution Hardware - Left:ChipWhisperer Lite - Right:XMEGA on CW308 UFO Board

We incorporated a C-language implementation of GIFT-64 using 8-bit data types into the "simple serial" firmware provided with the ChipWhisperer. All experimental data was collected using this implementation and hardware, apart from the S-Box constants which were modified with each test case. The CPA attack and the data analysis was performed using Python.

Experiment. As an experiment we consider a loop in which each iteration adds a randomly chosen plaintext/voltage array pair to a data-set, and then the CPA attack is executed using that data-set. The result of the CPA attack, i.e. the SR metric, is stored along with the count of plaintext/voltage array pairs used to perform the attack. The experiment continues looping until either the size of the data-set reaches the threshold cap or until a success limit of 5 consecutive successful CPA attacks are observed. In our study we used a threshold cap of 150 iterations and a pool of plaintext/voltage array pairs containing 2,000 entries. Example pseudocode is shown below.
The success limit was implemented in part to speed up computation and in part through recognition that in most cases, once the CPA attack is successful within a given

Algorithm 1: Experiment Pseudocode

```
Result: Success rate for each trace count
count = 0;
successCount = 0;
list initialized;
results structure initialized;
while count < Threshold do
    Add random plaintext-voltage array pair to list;
    Conduct CPA attack using list;
    count++;
    results[count] = CPA Success Rate (1/0);
    if Successful then
        successCount++;
        if successCount == 5 then
            Mark remaining results successful;
            return results;
        end
    end
    else
        successCount = 0;
    end
end
return results;
```

experiment it is likely to continue being successful with continued addition of more information. We chose the constant 5 to be a success limit based upon early experimental data, but the best parameter to ensure success stability requires further investigation.

A threshold cap of 150 was chosen based on trial and error. The goal was to choose a cap which would halt execution of an experiment but also allow each experiment to capture the full progression to a 100% mean success rate. Each of the experiments achieved 100% mean success rate before reaching the 150^{th} iteration.

The output of an experiment is a set of ordered pairs (x, y) stored in a *Results* array in which the 'x' value is the size of the data-set used for the CPA attack and the 'y' value is either 0 or 1 depending on if the CPA attack was successful or not. This means that the Results from an experiment is an array in which the index is the trace/count size and the element in the array is the success rate.

Trial. In this study we consider a *trial* to be a collection of 100 experiments in which the trial results output has ordered pairs similar to the output of the experiment, but the 'y' values hold the mean success rate of the 100 executions of the experiments.
For each SBox we computed several trials and our data will be explored in the Experimental Results section.

D. Experimental Results

We evaluated the GIFT-64 SBox against several additional SBoxes, some of which are well-known. More precisely, we

Algorithm 2: Trial Pseudocode

Result: Mean Success Rate for Each Trace Count
count = 0;
results structure initialized;
while *count* $<$ *100* **do**
    Execute Experiment;
    Store results of experiment in the results structure;
    count++;
end
Average the results of the experiments for each trace count;
Return the average mean success rates;

used the SBoxes from PRESENT, PICCOLO and the following 4x4 SBoxes (generated to purposely vary the metric values of the other SBoxes used in this study):

$$S_1 = [0, E, 7, 6, 4, 5, 2, 1, 3, F, A, B, 8, 9, C, D]$$
$$S_2 = [5, 1, 7, 6, 4, 0, 2, E, 3, F, B, A, 8, 9, C, D]$$
$$S_3 = [0, F, E, D, C, B, A, 9, 8, 7, 6, 5, 4, 3, 2, 1]$$
$$S_4 = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, A, B, C, D, F, 0]$$

In order to have a fair comparison of all SBoxes, we implement them in the same way by using look-up tables. Table I reports the theoretical metrics for each of the SBoxes.

TABLE I
VALUES OF THE METRICS APPLIED ON SBOXES

S-Box	Non-Linearity	SNR	DPA-SNR	TO	RTO
PICCOLO	4	39.401	3.108	3.666	3.333
GIFT-64	4	39.348	2.399	3.466	3.066
PRESENT	4	34.665	2.129	3.533	3.266
S_2	2	39.968	2.946	3.4	3.266
S_4	0	39.252	2.484	2.933	2.933
S_1	0	38.582	2.579	3.266	3.133
S_3	0	34.148	2.484	2.933	2.933

Note that all of the metric scores are deterministic except for the classical SNR for which we took the average of 100 executions of the SNR measurement. For each of the SBoxes we conducted several trials and there was negligible deviation on repeating trials on the same SBox. The graphs of the mean SR metric values of the analyzed SBoxes are presented in Figure 3.

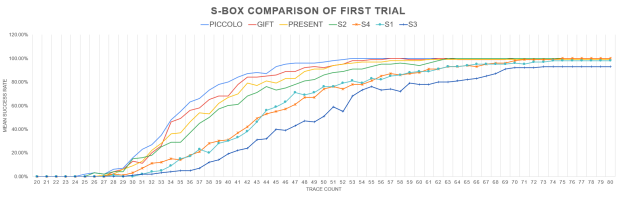

Fig. 3. Mean success rate (SR) metric values of the CPA attacks

Non-linearity characterizes the resistance of SBox against linear cryptanalysis [17]. The higher the non-linearity of an SBox, the more resistant the SBox is to linear cryptanalysis. However, it is widely accepted that SBoxes with higher non-linearity, more information leak through side channels [14].

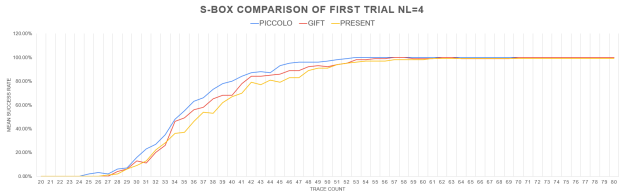

Fig. 4. Mean success rate of CPA on the SBoxes with non-linearity $NL = 4$

Figure 4 illustrates that the PRESENT SBox is the most CPA resistant among the SBoxes with non-linearity NL=4. Also note that the PRESENT SBox has a lower SNR value than the GIFT-64 or the PICCOLO SBox.

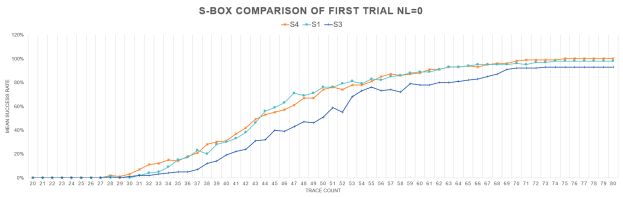

Fig. 5. Mean success rate of CPA on the S-Boxes with non-linearity $NL = 0$

Figure 5 illustrates a clear distinction that the mean SR value of S_3 is sufficiently lower among the SBoxes with non-linearity $NL = 0$. We also know that the SNR value of S_3 is significantly lower than that of S_1 and S_4 leading to a correlation between the SNR value and the mean SR value assuming their non-linearity values are equal.

IV. TEST VECTOR LEAKAGE ASSESSMENT OF LWC AND CAESAR HARDWARE IMPLEMENTATION OF GIFT-COFB

In this section we provide a useful preliminary assessment of the LWC and CAESAR hardware implementation's leakage [2] of GIFT-COFB by Rezvani et. al [6] implementation's leakages, before carrying out more elaborate evaluations and attacks.

We perform Test Vector Leakage Assessment (TVLA) using the Flexible Open-source workBench fOr Side-channel analysis (FOBOS)[7] test architecture [3]. TVLA identifies differences between two sets of side channel measurements, such as power and traces, by computing Welch's t-test for the two sets of measurements. Two sets of measurements are taken, the first with fixed inputs and the second with random inputs, which we will refer to as T_A and T_B, respectively. At each time step a pass/fail decision is made by testing for a null hypothesis such that the means of the two sets is

[6] SAL: NIST Lightweight Cryptography Implementations (Jan 2019), https:// github.com/vtsal
[7] FOBOS: https://cryptography.gmu.edu/fobos/

equivalent. The TVLA at each time step is calculated as follows:

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

where T_A and T_B are the two trace sets, $\overline{T_A}, \overline{T_B}, S_A, S_B, N_A$, and N_B are the means, variances, and size of T_A and T_B, respectively.
The null hypothesis is rejected with a confidence level of 99.9999% if the absolute value of the t-test is greater than 4.5 [9]. A rejected null hypothesis (fail decision) suggests that the two trace sets T_{rnd} and T_{fix} are different and might leak some information.

A. Experimental Setup.

The experiments were conducted using the Flexible Open-source workBench fOr Side-channel analysis (FOBOS) [3], [18] setup on a machine equipped with an Intel Core i5 CPU @ 2.4 GHZ running the Ubuntu 20.04.1 operating system. The setup of FOBOS is comparable with CAESAR hardware API [19] and LWC hardware API [20].

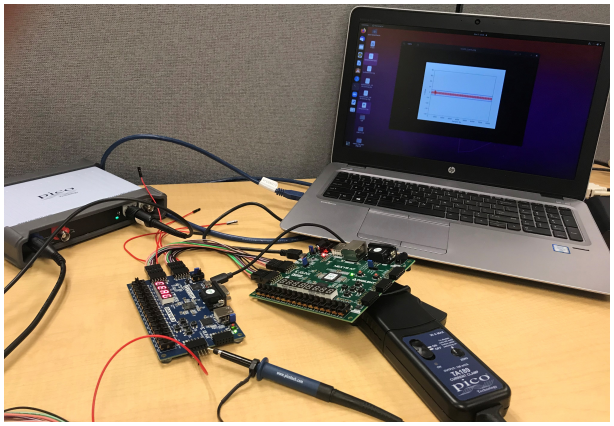

Fig. 6. FOBOS experimental setup

The FOBOS hardware setup, i.e., the control board, DUT board and oscilloscope used are Digilent Basys3, Digilent Nexys A7 and Picoscope 5000 series (5244D) respectively. To measure the power consumed by the device, the DUT board was modified so that we have a jumper on the power line (core FPGA voltage) and several capacitors on the voltage rail are removed to make the side-channel attack easier.
Once the experiment has been set up as shown in the Figure 6, we used the CAESAR and LWC hardware implementation of GIFT-COFB [8] and generated the input fixed-vs-random test vector by using the same key, nonce and associated data.

B. Experimental Results

A power leakage detection assessment was performed on a cryptographic implementation. The DUT clock was set to 10 MHz and the oscilloscope sampling rate was set at 12000 samples/sec.
A collection of 2000 traces were made using fixed-vs-random test vectors. Figure 7 and 8 shows the TVLA results on Nexys A7 for the unprotected GIFT-COFB hardware implementation on LWC and CAESAR respectively. Sample numbers are shown on x-axis; t-values are shown on y-axis. Leakage where $t > |4.5|$ denotes a t-test failure.

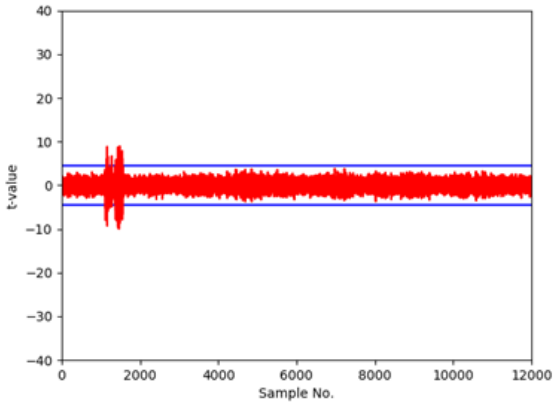

Fig. 7. TVLA results on LWC hardware implementation of GIFT-COFB.

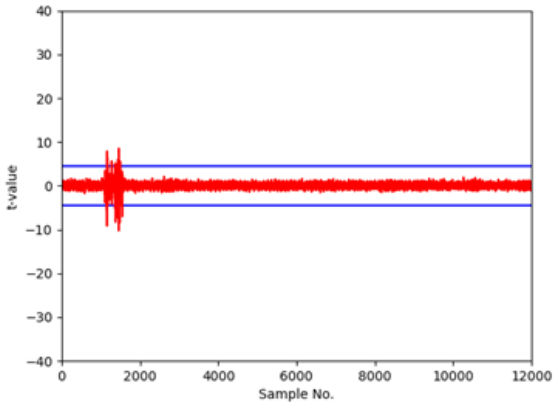

Fig. 8. TVLA results of CAESAR hardware implementation of GIFT-COFB.

As observed from Figure 7 and Figure 8, there are some spikes visible between the sampling number 0 and 2000. It can be deduced that there is significant first order power leakage in the unprotected versions.

V. Conclusion and Next Steps

In this paper, we analysed several well known theoretical metrics aiming to characterise the resistance of GIFT-64 and several other SBoxes against adversaries exploiting physical leakages. We then compared the outcome of these metrics with the real leakages measured on a software cryptographic device. All the experiments indicate that it is still unclear how an evaluator can sort SBoxes in terms of resiliency against side-channel attacks.

[8] https://cryptography.gmu.edu/athena

Our analysis shows that the versions of transparency order (TO and RTO) combined with the signal-to-noise ratio (SNR and DPA-SNR) variants are not sufficient to ensure practical security of the implementation of GIFT-COFB (and the other analyzed ciphers). Furthermore, the results from Table 2 suggest that among SBoxes with equal non-linearity (e.g., GIFT-COFB, PRESENT and PICCOLO), there is no significant differences in terms of TO and SNR variants.

Given the traces from inputs on the LWC and CAESAR hardware implementations of GIFT- COFB, TVLA has shown the presence of some leakage at one location on the traces in both implementations.

We are in the process of applying TVLA on hardware implementations of other LWC finalists to demonstrate the presence of leakages and identify their specific sources. Moreover, we plan to incorporate deep learning techniques and neural networks to side channel attacks (particularly on protected software/hardware implementations) in a non-profiled context and investigate whether such attack methods can outperform some classic non-profiled attacks such as CPA.

ACKNOWLEDGMENT

We would like to acknowledge high-performance computing support of the R2 compute cluster (DOI: 10.18122/B2S41H) provided by Boise State University's Research Computing Department.

REFERENCES

[1] S. Banik, A. Chakraborti, T. Iwata, M. Minematsu, M. Nandi, T. Peyrin, Y. Sasaki, S. M. Sim, and Y. Todo, "Gift-cofb," *Submission to NIST Competition*, vol. 1, 2021.

[2] B. Rezvani and W. Diehl, "Hardware implementations of nist lightweight cryptographic candidates: A first look," *IACR Cryptol. ePrint Arch.*, vol. 2019, p. 824, 2019.

[3] A. Abdulgadir, W. Diehl, and J.-P. Kaps, "An open-source platform for evaluation of hardware implementations of lightweight authenticated ciphers," in *2019 International Conference on ReConFigurable Computing and FPGAs (ReConFig)*, 2019, pp. 1–5.

[4] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," Springer, pp. 388–397, 1999.

[5] C. Clavier, J. S. Coron, and N. Dabbous, "Differential power analysis in the presence of hardware countermeasures," *CHES 2000*, vol. 1965, pp. 252—-263, 2000.

[6] E. Brier, C. Clavier, and F. Olivier, "Correlation power analysis with a leakage model," in *International workshop on cryptographic hardware and embedded systems*, 2004, pp. 16–29.

[7] E. Prouff, "Dpa attacks and s-boxes," in *International Workshop on Fast Software Encryption*. Springer, 2005, pp. 424–441.

[8] S. Mangard, E. Oswald, and T. Popp, *Power Analysis Attacks: Revealing the Secrets of Smart Cards (Advances in Information Security)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2007.

[9] G. Goodwill, B. Jun, J. Jaffe, and P. A. Rohatgi, "A testing methodology for side-channel resistance validation," *NIST non-invasive attack testing workshop*, vol. 7, pp. 115–136, 2011.

[10] H. Li, Y. Zhou, J. Ming, G. Yang, and C. Jin, "The notion of transparency order, revisited," *The Computer Journal*, vol. 63, no. 12, pp. 1915–1938, 2020.

[11] R. E. Ziemer and W. H. Tranter, *Principles of Communication Systems, Modulation and Noise*, ser. ohn Wiley & Sons. New York, 1995.

[12] S. Guilley, P. Hoogvorst, , and R. Pacalet, "Differential power analysis model and some results," in *Smart Card Research and Advanced Applications Vi*. Springer, 2004, pp. 127–142.

[13] K. Nyberg, "Differentially uniform mappings for cryptography," *EUROCRYPT'93, Lect. Notes Comput. Sci.*, vol. 765, no. 1, pp. 55–64, 1993.

[14] A. Biryukov, D. Dinu, and J. Großschädl, "Correlation power analysis of lightweight block ciphers: From theory to practice," in *International Conference on Applied Cryptography and Network Security*. Springer, 2016, pp. 537–557.

[15] Z. Wang, P. Zhang, C. Chen, and H. Hu, "Pre-processing of power traces in power analysis," *Chin. J. Commun. Technol.*, vol. 50, no. 4, pp. 765–770, 2017.

[16] C. O'Flynn and D. Chen, "Chipwhisperer: An open-source platform for hardware embedded security research," in *International Workshop on Constructive Side-Channel Analysis and Secure Design*. Springer, 2014, pp. 243–260.

[17] M. Matsui, "Linear cryptanalysis method for des cipher," in *Workshop on the Theory and Application of of Cryptographic Techniques*. Springer, 1993, pp. 386–397.

[18] R. Velegalati, J.-P. Kaps *et al.*, "Towards a flexible, opensource board for side-channel analysis (fobos)," *Cryptographic architectures embedded in reconfigurable devices, CRYPTARCHI*, 2013.

[19] E. Homsirikamol, W. Diehl, A. Ferozpuri, F. Farahmand, P. Yalla, J.-P. Kaps, and K. Gaj, "Caesar hardware api," *Cryptology ePrint Archive*, 2016.

[20] J.-P. Kaps, W. Diehl, M. Tempelmeier, E. Homsirikamol, and K. Gaj, "Hardware api for lightweight cryptography," *URL https://cryptography. gmu. edu/athena/index. php*, pp. 1–26, 2019.