Experimental Evaluations of Fault Sensitivity Imbalance for Masked AES Using t-Test and Entropy

Haruka Hirata*, Koki Furuno*, Daiki Miyahara*,
Mitsufu Iwamoto*, Kazuo Sakiyama*, Yang Li*
*The University of Electro-Communications, Tokyo, Japan
Email: h.haruka@uec.ac.jp

***Abstract*—Physical attacks on electronic devices that involve cryptographic functions, such as IC cards and credit cards, pose a significant threat to security. Side-channel and fault attacks are typical threats, and in recent years, deep learning-based analysis methods have also been proposed to effectively extract encryption keys. Threshold implementation is one of the countermeasures to thwart side-channel attacks, which conceals physical leakage by splitting the processed data into multiple values termed shares. The protected circuit must be evaluated both theoretically and practically. However, while evaluation methodologies for side-channel analysis have been well established, those for fault analysis have not yet been thoroughly explored. In this paper, we propose two practical evaluation methods for fault analysis utilizing the fault sensitivity. The first detects data-dependent fault occurrences using Welch's t-test, and the second assesses fault sensitivity on each share using Shannon entropy. We then conduct fault injection experiments on a TI-protected AES implementation to demonstrate the efficacy of our proposed methods. The results show that the t-test evaluation can successfully detect data-dependent faults, and the entropy-based evaluation reveals differences in fault sensitivity among the shares.**

***Index Terms*—AES, Fault Sensitivity Analysis, TI, S-box, entropy, t-test.**

I. Intorduction

Side-channel attacks exploit physical information unintentionally leaked from hardware or software during cryptographic operations—referred to as side-channel information—to recover secret keys. Well-known examples include power analysis attacks such as Differential Power Analysis (DPA) and Correlation Power Analysis (CPA) [1], [2], which analyze power consumption during encryption, as well as electromagnetic analysis attacks [3], which utilize electromagnetic emissions from the circuit. In contrast, fault attacks deliberately inject faults into cryptographic circuits and analyze the resulting behavior to extract secret information. Representative techniques include Differential Fault Analysis (DFA) [4], which leverages the difference between correct and faulty ciphertexts to recover the key, and Fault Sensitivity Analysis (FSA) [5], which exploits data-dependent fault behavior, namely the *fault sensitivity*.

To evaluate side-channel resistance, both theoretical and practical approaches have been studied. Theoretical approaches include programming-based leakage detection tools [6]–[10] while practical evaluations often rely on statistical tests such as Welch's t-test, the χ^2-test [11], and the Kolmogorov–Smirnov test [12]. A well-established method, known as Test Vector Leakage Assessment (TVLA) [13], [14], leverages test vectors and has proven effective in assessing the leakage resilience.

On the other hand, methodologies for evaluating resilience against fault attacks are limited, although some verification tools for fault analysis [15]–[17] have been proposed. Moreover, these verification tools inspect a circuit's security at the gate level; thus, they might overlook a data-dependent flaw, namely, an algorithmic-level issue.

For example, the zero-value attack on a masked AES implementation, pointed out by Hirata et al. [18], cannot be detected by VerFI [15], a simulation-based verification tool. Furthermore, the use of fixed and random plaintext datasets in traditional TVLA may fail to reveal data-dependent behavior, making it difficult to determine which input values cause erroneous outputs. Moreover, an assessment methodology for shared form implementation under fault injection attacks has not been explored yet due to the difficulty of precisely measuring fault sensitivity on each share.

To address these challenges, this paper proposes two practical evaluation methodologies for masked cryptographic implementations against fault attacks, leveraging Welch's t-test and Shannon entropy. As a case study, we investigate a masked AES implementation. First, we design a specific test vector to diagnose data-dependent issues, which the t-test can successfully detect. Next, the second evaluation assesses the share-wise security and vulnerability of the implementation by using an additional setup acting as an oracle, revealing an alternative attack surface.

The rest of this paper is organized as follows. Section II provides background information. Section III introduces the proposed evaluation methodologies, and Section IV presents the experimental results. Finally, Section V concludes the paper.

II. Background

This section describes preliminaries for the protected implementation.

The final published version is available at IEEE Xplore: https://doi.org/10.1109/CANDAR68384.2025.00024

A. Masking

Masking [19] is a widely used countermeasure against side-channel analysis, particularly probing attacks, where an attacker observes precisely the wire(s) on a circuit. In this scheme, a sensitive value x is split into multiple values, called *shares*, using random values, such that

$$x \mapsto (x_0, x_1, \ldots, x_d), \quad \text{where} \quad x = \bigoplus_{i=0}^{d} x_i. \tag{1}$$

All computations during encryption are performed in shared form and then are recombined after the operation to obtain the unmasked output value.

In the d-probing attacker model [19], the attacker is allowed to observe up to d internal wires (i.e., shares). A masked implementation is said to be d-th-order secure under the adversary model if any subset of (at most) d observed shares reveals no information regarding the sensitive value x. Formally, the security condition is given by:

$$\Pr[X = x \mid \text{observations}] = \Pr[X = x], \tag{2}$$

where the observations refer to the values obtained by the probing.

B. Implementation of AES S-box

An S-box, namely a non-linear function, is a crucial component in protecting cipher implementation against side-channel analysis. Canright proposed a compact calculation for the AES S-box [21] which performs inversion over $GF(2^8)$ and then an affine transformation. The computation can be easily pipelined and is compatible with the masked circuit; therefore, it is often used for protected AES implementations.

For the inversion over $GF(2^8)$, the given input $x \in GF(2^8)$ is mapped to an isomorphic field $GF((2^4)^2)$, i.e., $\phi(x) = (a, b) \in GF((2^4)^2)$. The multiplicative inverse of (a, b), i.e., $(c, d) = (a, b)^{-1}$, is computed as follows:

$$(c, d) = (tb,\, ta), \ \text{with} \ t = \left[ab + (a+b)^2\nu\right]^{-1}, \tag{3}$$

where, $\nu \in GF(2^4)$ is a constant value. Finally, we obtain the inverse $y = x^{-1}$ by $\phi^{-1}((c, d)) = y$.

Figure 1 illustrates these computations, and we utilize an ASIC board whose implementation is based on the calculation above in experiments described in Section IV.

The implementation is proposed by De Cundde et al. [20] and has three shares, ensuring the second-order security. Moreover, it is based on the consolidating masking schemes [22] approach, which extends threshold implementation (TI) [23], but we hereafter refer to the implementation as TI-AES for simplicity.

III. Leakage Evaluation Metrics

This section introduces evaluation metrics, Welch's t-test, and entropy, which are performed in Section IV.

A. Welch's t-test

The t-test is a statistical method used to determine whether two groups have significantly different means.

Given two sets of observations with sample means M_1, M_2, variances V_1, V_2, and sample sizes N_1, N_2, the t-value is computed as:

$$t = \frac{M_1 - M_2}{\sqrt{\frac{V_1}{N_1} + \frac{V_2}{N_2}}}. \tag{4}$$

The result is evaluated using the p-value derived from the t-distribution. If the p-value is below (or the t-value surpasses) a predefined threshold α (e.g., $\alpha = 10^{-5}$ for 99.999% confidence), then the null hypothesis—the two means are equal—is rejected, indicating a statistically significant difference. The corresponding t-value is 4.5 for $\alpha = 10^{-5}$, and this value is commonly used in power side-channel evaluations. Accordingly, we adopt a threshold of 4.5 in our assessment, following this common practice.

B. Entropy

Entropy [24] is a fundamental concept that quantifies the uncertainty associated with a random variable. When describing the uncertainty of a probability distribution, Shannon entropy is typically used. For a discrete random variable X taking values in a finite set $\mathcal{X}$, the Shannon entropy is defined as:

$$H(X) = -\sum_{x \in \mathcal{X}} p(x) \log_2 p(x), \tag{5}$$

where $p(x)$ denotes the probability that X takes the value x, i.e., $\Pr[X = x]$.

Another commonly used metric is the minimum entropy, or *min-entropy*, which captures the amount of uncertainty based on the most probable outcome. It is defined as:

$$H_\infty(X) = -\log_2 \max_{x \in \mathcal{X}} p(x). \tag{6}$$

Min-entropy reflects the predictability of the most likely value of X, and therefore serves as an indicator of information leakage through guessing. From an attacker's perspective, entropy quantifies the uncertainty about the secret value and thus represents the amount of information that remains secure.

These evaluation methods are applied to the fault sensitivity behavior of the TI-AES implementation, as described in the following section.

IV. Experiments

This section demonstrates fault evaluation experiments with the method described in the previous section.

We use the SAKURA-G board [25], which is designed for side-channel leakage experiments and incorporates two FPGAs, including a main FPGA for an encryption circuit and a control FPGA for data communication to the main FPGA and PC via a serial port. We have two setups in the experiment: one uses an ASIC board and the main FPGA mounted on SAKURA-G, and the other uses SAKURA-G only. Thus, the latter setup substitutes the ASIC board with the main FPGA. The physical setup is shown in Figure 2(a)

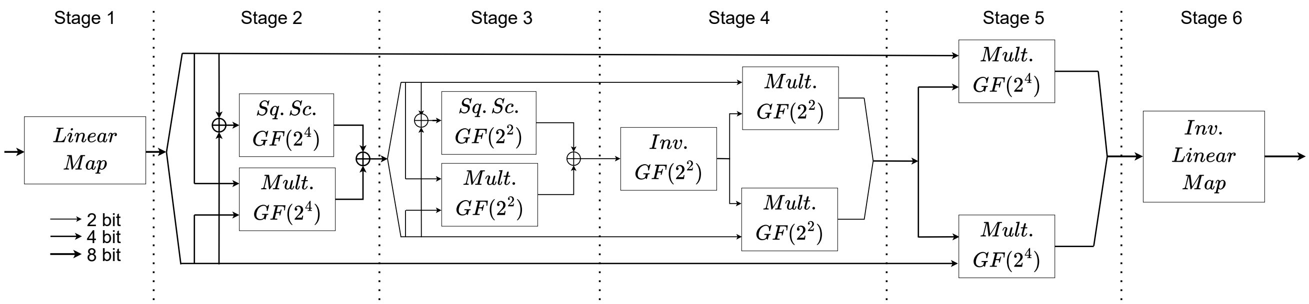
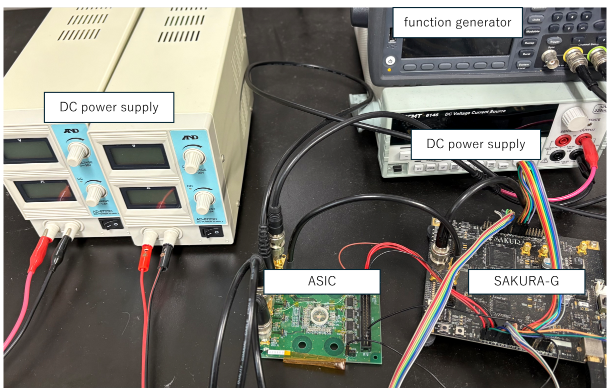
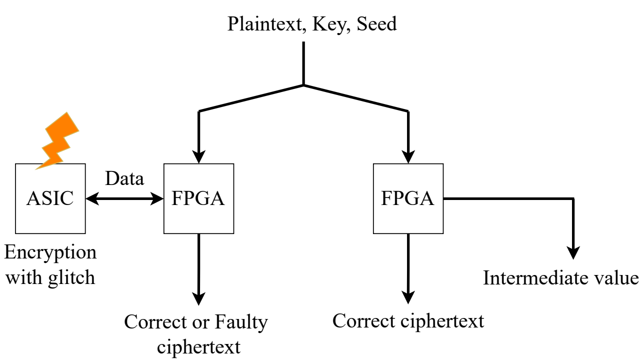


Fig. 1: Structure of the pipelined S-box proposed by De Cnudde et al. [20]



(a) Physical setup.



(b) Overview of our experimental setup for obtaining intermediate values.

Fig. 2: Experimental setup and fault injection overview.

We inject faults via a clock glitch, which causes a timing setup violation during the encryption. The glitchy clock, namely, a high-frequency clock, is generated by a Phase Locked Loop module built in the FPGA, and the timing of the glitch injection is located in the last round of the AES encryption. We utilize the clock glitch to induce the faulty operation in both the t-test and the entropy evaluation.

A. Setup

1) t-test evaluation: This section demonstrates value-dependent security evaluation using Welch's t-test, and Table I summarizes the experimental setup.

TABLE I: Setup for the t-test experiment

Target boards	SAKURA-G (Spartan-6)
Clock frequency	3.075 MHz
Glitch frequency	76.875 MHz

We generate specific plaintexts whose internal state, namely, the input to the last round, consists of a 16-byte vector of the same byte value v. In other words, the intermediate input value to the last round is as follows:

$$
\begin{aligned}
S_0^{10} &= 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00. \\
S_1^{10} &= 01\ 01\ 01\ 01\ 01\ 01\ 01\ 01\ 01\ 01\ 01\ 01\ 01\ 01\ 01\ 01, \\
&\vdots \\
S_{255}^{10} &= \mathrm{ff\ ff\ ff\ ff\ ff\ ff\ ff\ ff\ ff\ ff\ ff\ ff\ ff\ ff\ ff\ ff}.
\end{aligned}
$$

In this way, the entire 16-byte input to the last round is set to v; thus, the inputs to all stages of pipelined S-boxes are v.

Let M be a set of plaintexts $M = \{M_0, M_1, \ldots, M_{255}\}$, where the internal state appears as the value S_i^{10}, the input to the last round, with the given plaintext M_i. We here define the fault ratio R as the number of fault occurrences divided by the number of executions:

$$R = \frac{\#\text{fault occurences}}{\#\text{executions}}.$$

Algorithm 1 shows a data collecting procedure for the t-test evaluation. We use two SAKURA-G boards, denoted as G_1 and G_2, in the experiment for comparison purposes; thus, we perform the procedure twice for the two boards. Moreover, we make all conditions identical between the two boards, including the provided data and the mounted FPGA bitstream files.

Algorithm 1 Collecting fault ratios for the t-test evaluation

Select a fixed plaintext P_1 from M
for $i = 1$ to 200 **do**
    Select randomly a plaintext P_2 from M
    Obtain the fault ratio R_i^{fixed} for P_1
    Obtain the fault ratio R_i^{rand} for P_2
end for
Compute t-value from $(R_1^{\text{fixed}}, R_2^{\text{fixed}}, \ldots R_{200}^{\text{fixed}})$ and $(R_1^{\text{rand}}, R_2^{\text{rand}}, \ldots R_{200}^{\text{rand}})$
return t-value

TABLE II: Setup for the entropy experiment

Target board	ASIC (28 nm CMOS process)
Sub-board	SAKURA-G (Spartan-6)
Clock frequency	3.075 MHz
Glitch frequency	332.1 MHz
Supply voltage to the ASIC	0.63 V

For obtaining the fault ratios R^{fixed} (lines 4), the seed value used for making shares is refreshed in every execution; thus, a combination of shares varies for each encryption, resulting in different path delays although the unmased value is identical.

2) Entropy evaluation: This section evaluates the fault sensitivity on each share via Shannon and min-entropy.

In general, ASICs cannot be modified once they have been implemented; for example, it is not possible to add wires connecting to registers. Therefore, it is impossible to know intermediate values processed during encryption computation. Moreover, masking tequnique makes it more difficult as the circuit is implemented with masking and the processed values are calculated in shared form. To tackle this, we build the FPGA-only setup as an oracle to obtain additional data, specifically the intermediate values of the shares in the S-box computation.

The experimental overview and data collection procedure are illustrated in Figure 2(b) and Algorithm 2. The same data—specifically, the plaintext and the seed used for masking—is provided to the FPGA-only setup as in the ASIC–FPGA setup, simulating the intermediate values processed in the ASIC. However, to evaluate key dependency, we supply two variants of the key to the FPGA-only setup: the correct key ($K_2 = K_1$) and an incorrect key ($K_2 \neq K_1$). The operation is repeated one million times ($N = 1,000,000$), with the key fixed for all executions.

The experimental setup is summarized in Table II. The glitch frequency is higher than that of the t-test evaluation ($\approx 4.32\times$) due to the difference in critical path delay between an FPGA and an ASIC, while the normal clock frequency is the same as the t-test evaluation. Moreover, we supply a lower power voltage $0.63\,V$ than the intended value $0.90\,V$ to the ASIC core to enhance the fault intensity.

Algorithm 2 Collecting faulty and intermediate values

Require: Keys K_1, K_2, Number of iterations N
Ensure: Ciphertexts C_i, C'_i, Intermediate values I_i
1: **for** $i = 1$ to N **do**
2: Randomly generate P
3: $C'_i \leftarrow \text{AES}(K_1, P)$ // with glitch (ASIC-FPGA)
4: C_i, $I_i \leftarrow \text{AES}(K_2, P)$ // without glitch (FPGA-only)
5: **end for**
6: **return** C_i, C'_i, I_i

In this work, we mainly focus on stage 2, which exhibits the most significant critical path delay among all stages of the S-box. We thus can control the glitch intensity more precisely and minimize its effect on other modules, such as key scheduling, rather than the S-box.

TABLE III: The t-values for the fixed plaintexts on each stage of the S-box.

		Stage					
	Plaintexts	1	2	3	4	5	6
G_1	P_0	−0.438	730	39.4	173	0.276	−0.831
	P_{255}	1.12	−2.28	−0.240	0.932	−1.34	1.07
G_2	P_0	135	177	224	28.1	26.3	1.02
	P_{255}	8.43	0.214	−2.12	0.562	−0.861	−0.584

B. Results

1) t-test: We demonstrate two specific values P_0 and P_{255} for a case study, namely the all-zero S_0^{10} and the all-one S_1^{10} case. Table III shows the t-values on each stage of the pipelined S-box for G_1 and G_2.

The process on stages 2, 3, and 4 corresponds to the calculation inside the bracket in Eq. (3), i.e., $\left[ab + (a+b)^2\nu\right]^{-1}$ described in Section II-B. Even if the error occurs during this calculation, the resulting output is always correct for zero-value input since the mapped value on $GF((2^4)^2$ is zero-value as well, i.e., $\phi(0) = (0,0)$ as pointed out by Hirata et al. [18].

Therefore, the implementation has the value-dependent issue that nullifies the error, and the t-values on stages 2, 3, and 4 considerably surpass the threshold 4.5 on both G_1 and G_2. This result indicates that the value-dependent error is detectable by our evaluation method.

However, although the given data to boards and the mounted bitstream files are identical, the t-values on stages 1 and 5 have a distinguishable difference on the boards for P_0; the value of G_1 is under the threshold, while that of G_2 surpassed it. The same result is observed on stage 1 for P_0. The device's inherent characteristics might cause this difference, and this result implies the importance of the threshold to determine whether data-dependent error appears.

2) Entropy: We observed $12,281$ faulty outputs out of $1,000,000$ operations ($\approx 1.23\,\%$), and first checked the distribution of incorrect values in unmasked form. Figure 3 shows the histogram of faulty values, and the Shannon and min-entropy of the distribution shown in the graph are 7.980 and 7.497, respectively. To confirm the correctness of these values, we generate a simulated distribution, and those entropies are displayed in Table IV. It shows Shannon and min-entropy of the simulated distribution with data $N = 1,000,000$ and $N = 12,281$. The distribution does not contain zero-value since the faulty value never appears, as already mentioned above; the entropies are thus lower than eight. Therefore, the values obtained from the distribution in Figure 3 are reasonable.

Then, the histograms of faulty values and the corresponding entropies for each share are shown in Figure 4 and Table V. A clear bias is observed in the second and third shares, as indicated by both the Shannon and min-entropy metrics, when the oracle is configured with the correct key, i.e., $K_1 = K_2$. In contrast, when the incorrect key ($K_1 \neq K_2$) is used in the

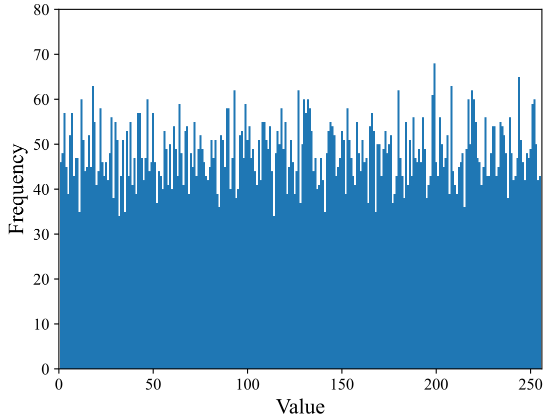


Fig. 3: Histogram of faulty value.

TABLE IV: Shannon and min-entropy values computed from simulated distribution.

Entropy Type	$\#data = 10^6$	$\#data = 12,281$
Shannon	7.994	7.981
Min	7.918	7.562

oracle, the entropies are close to the ideal values presented in Table IV, except for a slight deviation in the min-entropy of the third share. These results confirm that the observed bias is indeed key-dependent. Furthermore, as shown in Figure 5, the entropy values become stable when the number of collected data points reaches $N \geq 8,000$, confirming that the size of the measured data is sufficient for the evaluation.

These biases in entropy due to the imbalance of fault sensitivity affect the feasibility of fault injection and probing attacks, known as combined attacks. In such attacks, the attacker injects a fault during the encryption operation and then obtains the first share x_0, which follows an unbiased distribution. However, as the remaining shares are biased—as shown in Figures 4(b) and 4(c)—the conditional probability of Eq. (2) with the observed x_0 no longer satisfies, leading to

$$\Pr[X = x \mid \text{observations} = \{x_0\}] \neq \Pr[X = x]. \tag{7}$$

As a result, the share x_0 leaks information about the unmasked value x.

To ensure security against such attacks, the fault sensitivity, specifically, the critical path delay, must be balanced as much as possible. Additionally, the entropy of each share should be close to its maximum value (e.g., eight for an AES implementation). However, determining a precise entropy threshold that guarantees security remains an open challenge.

TABLE V: Shannon and min-entropy under the correct and incorrect keys.

Entropy Type	Key Condition	Share 1	Share 2	Share 3
Shannon	Correct	7.984	5.529	7.682
	Incorrect	7.986	7.986	7.985
Min	Correct	7.414	4.852	6.465
	Incorrect	7.518	7.518	7.375

V. Conclusion

In this paper, we proposed novel methodologies for evaluating the resilience of masked AES implementations against fault attacks, using Welch's t-test and entropy as evaluation metrics. We demonstrated the efficacy of these methods through fault injection experiments. In the first evaluation, we designed a specific test vector to detect data-dependent faults, which was successfully identified by the t-test. Then, the entropy analysis revealed a new attack surface in the presence of a combined adversary, namely, fault injection followed by probing.

As future work, we aim to investigate precise threshold values for these metrics to determine whether an implementation can be considered secure under both evaluation methodologies.

Acknowledgment

This work was supported by JSPS Research Fellows (grant number JP25KJ1291) and.. We would like to thank Taiga Sato for his assistance in collecting the experimental data.

References

[1] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Advances in Cryptology — CRYPTO' 99*, ser. LNCS, vol. 1666. Springer, 1999, pp. 388–397.

[2] E. Brier, C. Clavier, and F. Olivier, "Correlation power analysis with a leakage model," in *Cryptographic Hardware and Embedded Systems — CHES 2004*, vol. 3156. Springer, 2004, pp. 16–29.

[3] J.-J. Quisquater and D. Samyde, "Electromagnetic analysis (EMA): Measures and counter-measures for smart cards," in *Smart Card Programming and Security*. Springer, 2001, pp. 200–210.

[4] E. Biham and A. Shamir, "Differential fault analysis of secret key cryptosystems," in *Advances in Cryptology - CRYPTO '97*, ser. LNCS, vol. 1294. Springer, 1997, pp. 513–525.

[5] Y. Li, K. Sakiyama, S. Gomisawa, T. Fukunaga, J. Takahashi, and K. Ohta, "Fault sensitivity analysis," in *Cryptographic Hardware and Embedded Systems, CHES 2010*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 320–334.

[6] G. Barthe, S. Belaïd, G. Cassiers, P. Fouque, B. Grégoire, and F. Standaert, "maskverif: Automated verification of higher-order masking in presence of physical defaults," in *Computer Security - ESORICS 2019*, ser. LNCS, vol. 11735. Springer, 2019, pp. 300–318.

[7] D. Knichel, P. Sasdrich, and A. Moradi, "SILVER – statistical independence and leakage verification," in *Advances in Cryptology – ASIACRYPT 2020*, ser. LNCS, vol. 12491. Springer, 2020, pp. 787–816.

[8] N. Müller and A. Moradi, "PROLEAD A probing-based hardware leakage detection tool," *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, vol. 2022, no. 4, pp. 311–348, 2022.

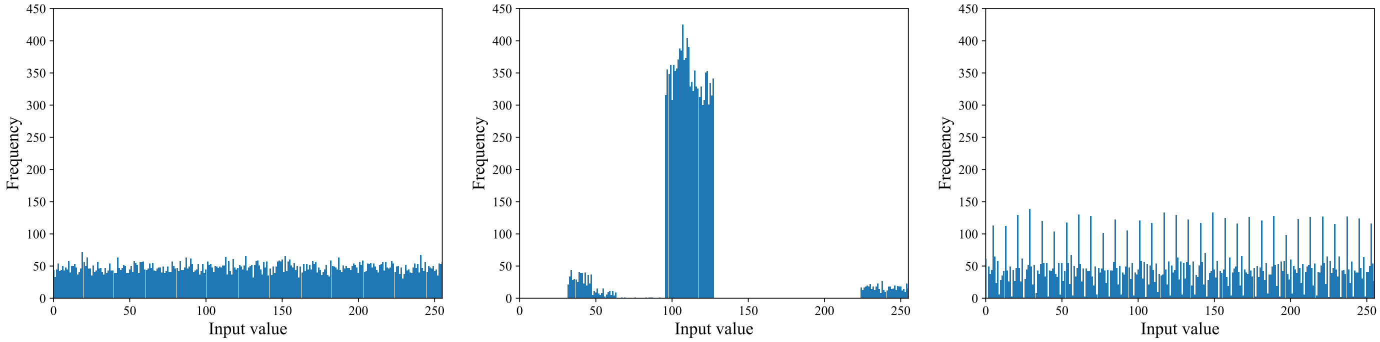
[9] J. Zeitschner, N. Müller, and A. Moradi, "Prolead_sw probing-based software leakage detection for ARM binaries," *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, vol. 2023, no. 3, pp. 391–421, 2023.

[10] J. Richter-Brockmann, J. Feldtkeller, P. Sasdrich, and T. Güneysu, "VERICA – verification of combined attacks automated formal verification of security against simultaneous information leakage and tampering," *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, vol. 2022, no. 4, pp. 255–284, 2022.

[11] A. Moradi, B. Richter, T. Schneider, and F. Standaert, "Leakage detection with the x2-test," *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, vol. 2018, no. 1, pp. 209–237, 2018.

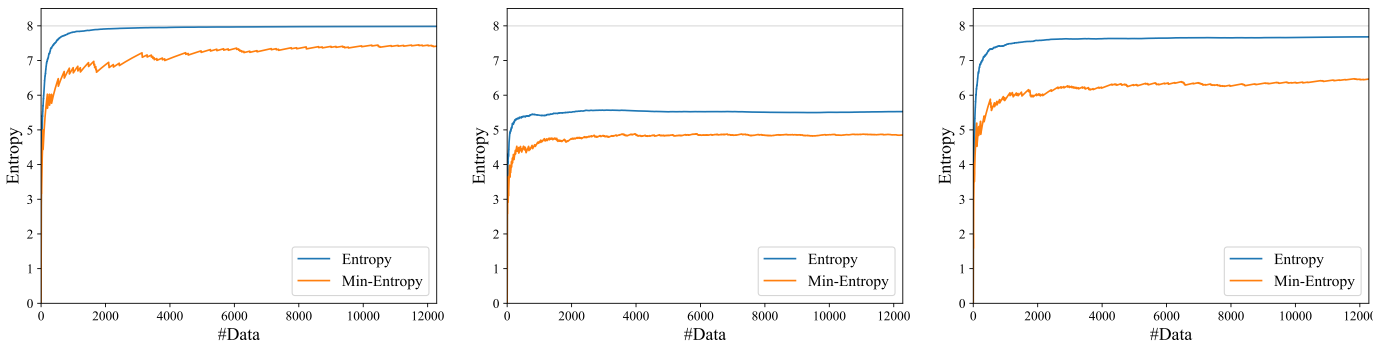
[12] X. Zhou, K. Qiao, and C. Ou, "Leakage detection with kolmogorov-smirnov test," *IACR Cryptol. ePrint Arch.*, p. 1478, 2019.

[13] G. T. Becker, J. Cooper, E. K. DeMulder, G. Goodwill, J. Jaffe, G. Kenworthy, T. Kouzminov, A. J. Leiserson, M. E. Marson, P. Rohatgi, and S. Saab, "Test vector leakage assessment (TVLA) methodology in practice," in *International Cryptographic Module Conference*, 2013.



(a) The first share (b) The second share (c) The third share

Fig. 4: Frequency histograms of faulty values.



(a) The first share (b) The second share (c) The third share

Fig. 5: Evolutions of the Shannon and Min-entropy.

[14] F. Bache, J. Wloka, P. Sasdrich, and T. Güneysu, "Multivariate TVLA - efficient side-channel evaluation using confidence intervals," *IEEE Trans. Computers*, vol. 74, no. 3, pp. 790–804, 2025.

[15] V. Arribas, F. Wegener, A. Moradi, and S. Nikova, "Cryptographic fault diagnosis using verfi," in *2020 IEEE International Symposium on Hardware Oriented Security and Trust, HOST 2020, San Jose, CA, USA, December 7-11, 2020*. IEEE, 2020, pp. 229–240.

[16] J. Richter-Brockmann, A. R. Shahmirzadi, P. Sasdrich, A. Moradi, and T. Güneysu, "FIVER - robust verification of countermeasures against fault injections," *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, vol. 2021, no. 4, pp. 447–473, 2021.

[17] F. Uhle, N. Müller, and A. Moradi, "Fault injection evaluation with statistical analysis - how to deal with nearly fabricated large circuits," Cryptology ePrint Archive, Paper 2025/1287, 2025.

[18] H. Hirata, D. Miyahara, V. Arribas, Y. Li, N. Miura, S. Nikova, and K. Sakiyama, "All you need is fault: Zero-value attacks on AES and a new λ-detection m&m," *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, vol. 2024, no. 1, pp. 133–156, 2024.

[19] Y. Ishai, A. Sahai, and D. A. Wagner, "Private circuits: Securing hardware against probing attacks," in *Advances in Cryptology — CRYPTO 2003*, ser. LNCS, vol. 2729. Springer, 2003, pp. 463–481.

[20] T. D. Cnudde, O. Reparaz, B. Bilgin, S. Nikova, V. Nikov, and V. Rijmen, "Masking AES with $d+1$ shares in hardware," in *Cryptographic Hardware and Embedded Systems – CHES 2016*, ser. LNCS, vol. 9813. Springer, 2016, pp. 194–212.

[21] D. Canright, "A very compact S-Box for AES," in *Cryptographic Hardware and Embedded Systems — CHES 2005*, ser. LNCS, vol. 3659. Springer, 2005, pp. 441–455.

[22] O. Reparaz, B. Bilgin, S. Nikova, B. Gierlichs, and I. Verbauwhede, "Consolidating masking schemes," in *Advances in Cryptology — CRYPTO 2015*, ser. LNCS, vol. 9215. Springer, 2015, pp. 764–783.

[23] S. Nikova, V. Rijmen, and M. Schläffer, "Secure hardware implementation of nonlinear functions in the presence of glitches," *J. Cryptol.*, vol. 24, no. 2, pp. 292–321, 2011.

[24] T. M. Cover and J. A. Thomas, "Entropy, relative entropy and mutual information," in *Elements of Information Theory*, 2nd ed. Hoboken, NJ: Wiley-Interscience, 2006, ch. 2, pp. 13–55.

[25] H. Guntur, J. Ishii, and A. Satoh, "Side-channel attack user reference architecture board sakura-g," in *2014 IEEE 3rd Global Conference on Consumer Electronics (GCCE)*, 2014, pp. 271–274.