The Dissertation Committee for Vishnuvardhan Iyer Certifies that this is the approved version of the following Dissertation:

Fine-grained Methods for Using EM Fields Measured Near Computing Chips to Evaluate Data Leakage

Committee:

Ali Yilmaz, Supervisor

Andreas Gerstlauer, Co-Supervisor

Michael Orshansky

Jaydeep Kulkarni

Emily Porter

Calvin Chan (CU Boulder)

Fine-grained Methods for Using EM Fields Measured Near Computing Chips to Evaluate Data Leakage

by

Vishnuvardhan Venkatramani Iyer

Dissertation

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Doctor of Philosophy

The University of Texas at Austin

December 2023

Acknowledgements

I would first like to express my sincere gratitude to my advisor Dr. Yilmaz for his constant encouragement, guidance and support throughout my graduate studies and research project.

I would also like to thank the remaining members of my dissertation committee: Dr. Andreas Gerstlauer, Dr. Michael Orshansky, Dr. Jaydeep Kulkarni, Dr. Emily Porter, and Dr. Calvin Chan, for their feedback. I would like to thank Dr. Gerstlauer and Dr. Orshansky for their inputs and support during my doctoral studies, as Co-PIs of the NSF award that funded me. I would also like to thank Dr. Kulkarni and his group for multiple collaborations, which have also contributed significantly to this work.

Further, I would like to acknowledge my fellow UT collaborators Ge Li, Meizhi Wang, and Aditya Thimmaiah who have helped in making my research and dissertation successful. I also learnt a lot from my colleagues in the CEM group, particularly Jon Kelley and Andrew Maicke. Over the past few years I have had a chance to make some great friends: Amruth Bhargav, Raghav Thyagarajan, DMS Gautham, Pawan Joshi, Sudhanva Vasishtha Dhinnessh R, Ali Farshkaran, Kartik Patel, Siddhartha Raman, and others, who have always been supportive in my endeavors.

Finally, I would like to thank my parents and extended family for their continuous support and encouragement during my time at UT.

Abstract

Fine-grained Methods for Using EM Fields Measured Near Computing Chips to Evaluate Data Leakage

Vishnuvardhan Venkatramani Iyer, Ph. D.

The University of Texas at Austin, 2023

Supervisor: Ali Yilmaz

This thesis presents novel fine-grained methods that show electromagnetic (EM) fields measured near chips during computations can be effectively used to evaluate data leakage. Several near-field measurement techniques combined with appropriate statistical analyses are introduced in the dissertation. The proposed EM side-channel analysis (SCA) methods are used to rapidly localize information leakage on the chip, identify optimal reusable measurement setups to minimize marginal cost of future evaluations, and infer the data values of interest. These methods are used to perform measurement-based evaluations of data leakage from several embedded system applications: (i) Using encryption keys of the advanced encryption standard (AES) algorithm as the data of interest, a multi-stage measurement protocol is introduced to rapidly identify chip locations which are most likely to leak the key, as well as the actual key value; the method was found to be ~2× to ~37× faster than alternatives while using them to evaluate the SCA resilience of several baseline and hardened implementations of AES; (ii) Assuming processor instructions as the data of interest, a hierarchical disassembler is developed to recover the execution trace of programs from a general-purpose micro-controller; the method was found to recover ~97%

instructions from several application benchmarks; (iii) Using Bluetooth payload as the data of interest, vulnerable locations on a Bluetooth Low Energy server implementation are isolated, and the data values of the payload are estimated; while the exact data values were not found, the Hamming Weight (HW) of test data was identified with 100% accuracy. These methods provide feasible alternatives to an exhaustive evaluation where data is recovered after measuring all possible computations at every single probe configuration. The feasibility of these methods is inherently dependent on the restrictions placed on evaluators, i.e., the threat model. Thus, a systematic study of protocols suited for different threat models are performed, which also includes the marginal cost comparisons of different SCA attack modalities. Finally, the thesis also introduces novel metrics and modelling methods that improve potency of side-channel security evaluations.

Table of Contents

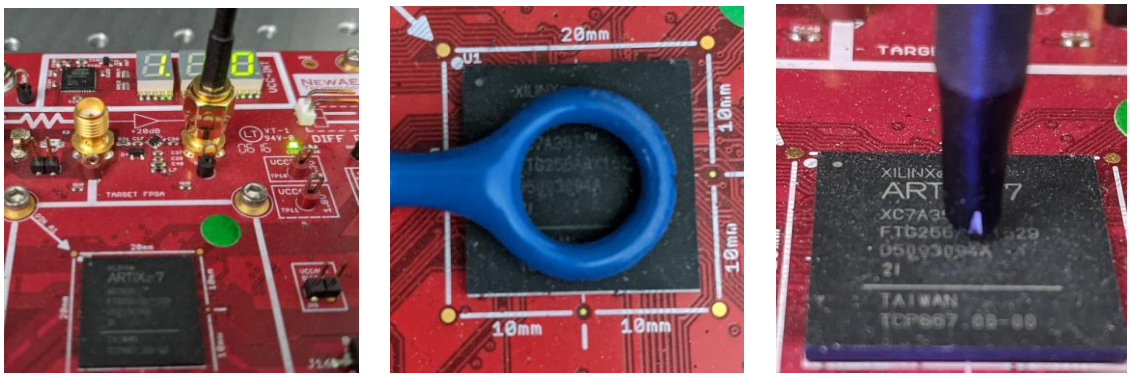
List of Tables

List of Figures

1. Introduction

Data in computing chips can be leaked unintentionally via EM fields, power consumption, temperature, etc. measured on/close to the chip [1]-[11]. These "side channels" are potential pathways that attackers can exploit to recover critical data, such as encryption keys from cryptographic implementations [5]-[21], rendering the security of these chips obsolete. In particular, measurement of EM fields represents a non-invasive pathway for attackers to recover data without tampering the device, necessitating effective security evaluations to mitigate any potential exploits. The vulnerabilities of various digital devices to EM side-channel analysis (SCA) attacks have been repeatedly demonstrated over the past decade [1]-[21]. Computations of interest in such devices unintentionally influence transitions in digital CMOS logic, resulting in data-dependent switching of transistors that affects power consumption and EM emanations [22]. Observed fields can be represented as sum of fields generated by the computations of interest, un-correlated background computations (henceforth referred to as algorithmic noise), and measurement noise [4],[6],[21]. EM SCA setups use appropriate statistical tools to relate observed fields to the computations of interest, quantify noise in signals, and deduce the data of interest.

Conventionally, EM SCA attacks have been performed using large probes that aggregate fields from a multitude of on-/off-chip sources, including algorithmic noise from those uncorrelated to the computations of interest [6], similar to power SCA attacks, where the measured signal is dictated by the aggregate current drawn by the logic blocks (Figs. 1(a) and (b)). As a result, they typically require many measurements to denoise signals, establish sufficient statistical relations with computations of interest, and recover data. Furthermore, these are memoryless attacks: previous attacks do not impact future evaluations. Such coarse-grained setups are commonly used to evaluate hardware security [8], [9], [12], [16] in part because the setups are relatively easy to implement, requiring a



(a) (b) (c)

Figure 1.1: Setups for power, coarse-grained EM, and fine-grained EM SCA attacks: (a) A sensor monitoring the aggregate power use of the chip (via the top-right port). (b) A 10-mm diameter H-field probe aggregating fields emanated by sources distributed throughout the chip. (c) A 1-mm diameter H-field probe scanning the chip surface for vulnerabilities.

single sensor configuration. In contrast, security evaluations using fine-grained EM SCA setups with relatively small probes (Fig. 1(c)) are rare, more elaborate, and potentially more potent [9]. These attacks first search for optimal configurations, e.g., locations and orientations, of probes that are most sensitive to target signals/least sensitive to noise; they then use these configurations to perform appropriate statistical analysis and recover data. Because they can localize leakage sources [5]-[7], [9]-[11], e.g., via high-resolution scans, these setups can circumvent some countermeasures that are effective against power and coarse-grained EM SCA attacks [1].

While coarse-grained EM SCA attacks are simpler to implement, fine-grained EM SCA attacks can be more efficient when used with optimal probe configurations, making them more potent than the conventional power/coarse-grained EM SCA attack methods [9],[11],[22]; moreover, once identified, these configurations can be reused to minimize the cost of future attacks on similar chips. Fine-grained EM SCA attacks' initial search for optimal probe configurations, however, can be rather costly [10] because of the large

number of probe configurations that must be evaluated. The cost of fine-grained EM SCA attacks can become intractable if chips are evaluated exhaustively with measurements corresponding to a multitude of combinations of data values in digital blocks, and these signals are collected at all possible probe configurations. This naïve approach is henceforth referred to as the "exhaustive search" for optimal configurations. For example, an exhaustive search for configurations leaking AES keys in a space of 51×51 locations, across 2 orientations (See Chapters 2 and 3), involves performing expensive correlation analysis attacks with a large set of encryptions ($>10^5$ in some cases [12], [18]) at all configurations, following which the most optimal configuration can be identified. Similarly, an exhaustive search for information leaking configurations in micro-controllers may involve probing the chip with a large configuration space (see Chapter 4) with all possible architectural states of the chip's digital components ($>10^{12}$ for an 8051 processor [54]), such as registers, counters, etc.

The acquisition costs accrued to perform fine-grained EM SCA attacks can be reduced by minimizing the number of configurations measured. Adaptive scan algorithms such as greedy [10] or gradient [11] search select probe configurations over multiple scans by introducing constraints on the resolution, search area, or the number of measurements and discarding non-optimal configurations in each scan. These search algorithms may zero in on local minima and cannot guarantee the best probe configuration will be identified, unlike the exhaustive search. Measurement costs can also be reduced by pre-supposing that information leakage is limited to certain time/frequency samples or locations [5],[23]. In [5], the information-leaking frequency was constant across the search space and a small set of initial guess configurations were used to rapidly isolate leakage to near decoupling capacitors over a test board implementing the advanced encryption standard (AES) algorithm. Similarly, in [23], both the time window and frequencies of information leakage

were identified, potentially reducing future measurement costs. Such methods are contingent on the invariance of information-leaking times/frequencies/locations. This may not be the case, however, for certain classes of countermeasures that can change signal profiles from encryption to encryption (see Section 3.3.3). Pre-supposing narrow time/ frequency/spatial windows to reduce the search space in the presence of such countermeasures can erroneously indicate that a system is resilient. Thus, these methods have limited utility for evaluating EM SCA attack vulnerabilities of hardened implementations. Further, the effectiveness of each method also depends on the restrictions placed during evaluations.

There is an inherent asymmetry between evaluators, who must ensure the module is sufficiently secure against all probe configurations, and actual attackers, who must ensure it is sufficiently vulnerable to only one probe configuration. The asymmetry is amplified when evaluators and actual attackers are subject to different constraints; in particular, on their ability to observe or control the module's inputs, outputs, or internal parameters, such as keys. These constraints are formalized in threat models: Actual attackers are often restricted to a "black-box threat model" [9], where the module's output and EM fields can be observed for a potentially unlimited number of encryptions but its input and internal parameters, such as encryption keys, cannot be accessed. In contrast, security evaluators may also be granted partial/full control over the input (a "gray-/white-box threat model" [9], [11], [19]) and internal chip parameters [21] (a "gold-box threat model" [9]). Thus, evaluators may observe the output and EM fields for specially designed test cases [21]. When evaluators face fewer restrictions, they can accelerate the security evaluation by implementing targeted tests and obtaining statistical indicators of information leakage, e.g., via test vector leakage assessment (TVLA) [24], [25] or analysis of variance (ANOVA) [4], [6], [19], [21], [26], prior to performing correlation-analysis

attacks. Therefore, instead of minimizing the number of configurations probed, the acquisition cost is reduced by minimizing the number of signals collected at each configuration.

Once the acquisition cost has been accrued, and optimal measurement configurations are identified, the configurations are then used to recover information using appropriate statistical methods. Information recovery uses only the reusable optimal configurations identified in the initial search, allowing evaluators to perform future evaluations at a small marginal cost. The statistical methods used for the initial search for configurations, as well as information recovery, depend on the chip functions and the data of interest, since the utility of fine-grained EM SCA attacks can range from identifying an encryption key from cryptographic modules (Chapters 2 and 3), to disassembling instructions implemented by a general purpose processor (Chapter 4). Therefore, fine-grained EM SCA evaluations can potentially be a powerful tool to validate a system's security and perform non-invasive data recovery, if the cost is feasible.

1.1 THESIS STATEMENT

The rich signal content present in EM side-channels leaking from embedded systems can be potently utilized by employing fine-grained methods that analyze EM fields from several on-chip locations with high scan resolutions. The feasibility and practicality of these methods is contingent on an evaluator's understanding of computations of interest, as well as the threat model during evaluation, which can enable the development of optimized measurement techniques and statistical methods that aid data recovery using these non-invasive EM side-channels.

1.2 THESIS CONTRIBUTIONS

The contributions of this thesis are as follows:

- *Comparison of different SCA attacks* [9]: We empirically evaluate the effectiveness of coarse- and fine-grained electromagnetic (EM) side-channel analysis (SCA) attacks, as well as power SCA attacks on implementations of the AES algorithm, subject to different constraints. These constraints/threat models can range from a highly restrictive red-box model, where the evaluator is only allowed access to side-channel signals, to a white-box model, where evaluators can manipulate the input data sent to the device. To compare the effects of these constraints on the fine-grained EM SCA, we develop/suitably adapt search methods for each threat model and compute their associated measurement costs.
- *Rapid Evaluation of AES vulnerabilities* [21]: We develop a multi-stage measurement protocol that identifies optimal measurement configurations—that minimize the marginal cost for repeated attacks to extract the data of interest, using far fewer measurements than previously demonstrated. We achieve this by developing a set of inexpensive characterization measurements based on a gold-box threat model, where Analysis of Variance (ANOVA) computations are performed on high-resolution scans to determine information leaking-locations rapidly. The protocol is used to test the resilience of several baseline and hardened implementations of AES.
- *EM-based Instruction Disassembler* [69]: We develop a hierarchical instruction-level disassembler that analyzes leakages created via the EM side-channel during program execution. The disassembler identifies instruction-dependent features using fine-grained EM measurements for a set of optimally designed instruction profiling codes to minimize measurement costs. These features are used in the

classification phase to disassemble the instructions from several application benchmarks, as well as predicting control-flow/ branching in the code.

- *Extensions to existing methods* [75],[76]: We improve upon the ANOVA method demonstrated for AES in [21], and also extend them to other embedded systems. As an example, we demonstrate the method as a data recovery tool from a Bluetooth server receiving data. We also present an information-leakage modelling method that can potentially predict fields from arbitrary data computations. This is achieved by representing EM information leakage as a superposition of leakages from individual sources and using a linear combination of data-dependent EM-basis functions to predict the same.

1.3 THESIS ORGANIZATION

The rest of this dissertation is organized as follows: Chapter 2 presents a comprehensive study of SCA attack modalities and compares their effectiveness in recovering AES encryption key for several threat models. Chapter 3 presents a multi-stage measurement protocol to rapidly evaluate vulnerabilities of AES modules. Chapter 4 presents a hierarchical disassembler that recovers execution traces from programs running on a general purpose micro-controller. Chapter 5 presents methods to model information leakage and demonstrates the methods on a general purpose micro-controller and a Bluetooth server implementation. Chapter 6 concludes the work.

2. Comparison of SCA Attack Modalities[1]

This chapter presents a brief background on AES and its vulnerabilities, introduces the ANOVA F-statistic for fine-grained EM SCA attacks, summarizes a previously proposed measurement protocol, and uses these tools to evaluate AES implementations under various constraints. The potency of fine-grained EM SCA attacks are also compared to coarse-grained EM and power SCA attack towards the end of the chapter.

2.1 Background on AES Operations and Vulnerabilities

This section summarizes the operations performed by an AES implementations and a known vulnerability that is exploited in SCA attacks.

2.1.1 The AES algorithm

AES, a commonly adopted standard for processor and wireless security, specifies a symmetric-key algorithm [27] that uses the same key for encryption and decryption. It is a block cipher that groups inputs into fixed 16-byte blocks and can use keys of size 128, 192, or 256 bits; the 128-bit implementation is used in this thesis (Fig. 2.1). Each encryption e by AES-128 requires 10 rounds of operations to transform the 16-byte input plaintext $\mathbf{ip}_e$ to the output ciphertext $\mathbf{oc}_e^{10}$ using the key $\mathbf{k}^0$ (Fig. 2.1). In each round $rd \in \{1, \cdots, 10\}$, a round key $\mathbf{k}^{rd}$ (generated from the key $\mathbf{k}^0$ via a key-expansion algorithm [25]) is used to update the 16-byte output to $\mathbf{oc}_e^{rd} = [oc_e^{rd,0}, \cdots, oc_e^{rd,15}]$. All AES operations are performed byte wise: In each round rd, first, each byte $b' \in \{0, \cdots, 15\}$ of the previous round's output $oc_e^{rd-1,b'}$ is replaced by an intermediate value $iv_e^{rd,b'}$ using a substitution

[1] This chapter is partly based on a previous publication: V. Iyer, M. Wang, J. Kulkarni, and A. Yilmaz, "A systematic evaluation of EM and power side-channel analysis attacks on AES implementations," in *Proc. IEEE ISI*, Nov. 2021.
The author contributed to the formulation, implementation, and measurements presented in this article, as well as the writing of this manuscript.

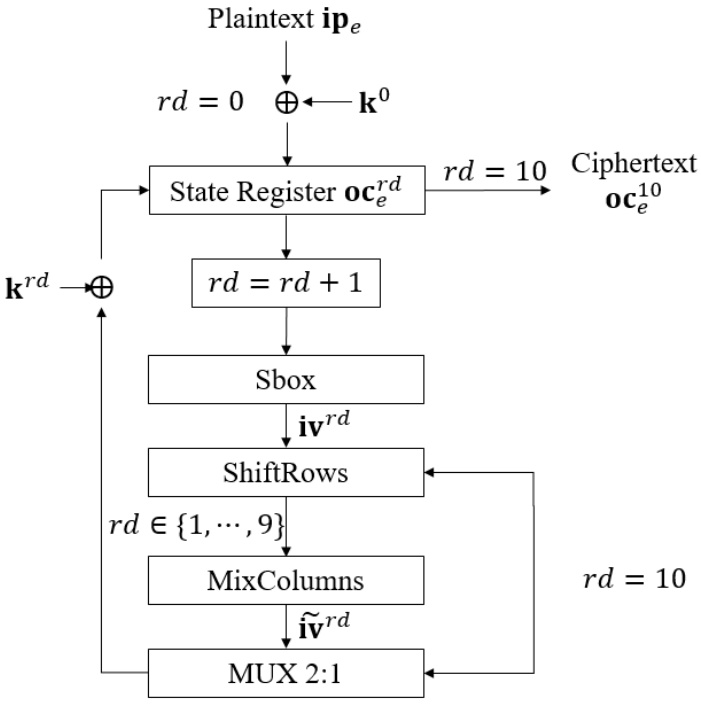


Figure 2.1: Algorithm flow of the 128-bit AES

box (*Sbox*). The *Sbox* transform replaces a byte's value using a one-to-one non-linear map defined by Rijndael's finite field [27]. This step adds "confusion" to the cipher, i.e., for fixed input plaintext, minor variations in the key result in large variations in the ciphertext. Then, the byte order of $iv_e^{rd,b'}$ is shuffled using the *ShiftRows* and *MixColumns* transforms to generate $\tilde{iv}_e^{rd,b}$, where $b \in \{0, \cdots, 15\}$ is the new position of the byte in the updated 16-byte array. These steps "diffuse" information in the cipher, i.e., different ordering of the bytes in the input plaintext causes large variations in the output ciphertext. Finally, the intermediate value is *XOR*ed with the key byte $k^{rd,b}$ to generate the output byte $oc_e^{rd,b}$. The *MixColumns* operation is skipped in the last round; thus, the last round of AES can be represented as

$$oc_e^{10,b} = ShiftRows\left(Sbox\left(oc_e^{9,b'}\right)\right) \oplus k^{10,b} \tag{2.1}$$

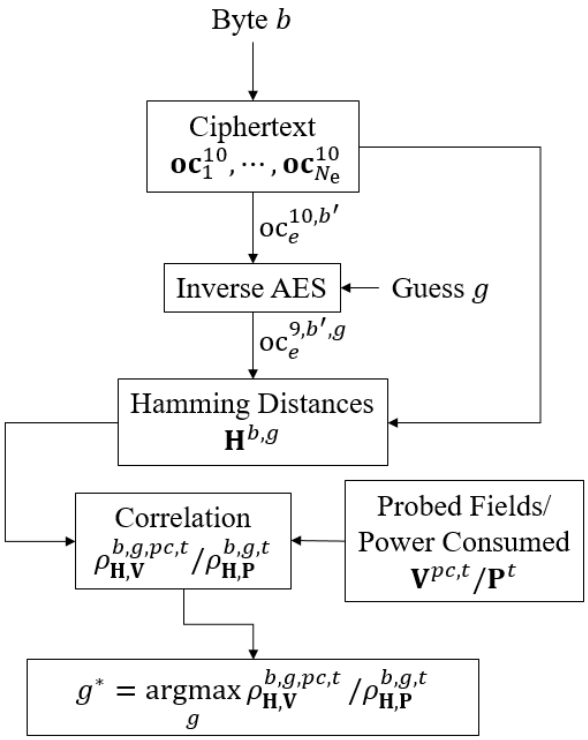


Figure 2.2: Byte-wise EM/Power SCA attack flow

If attackers have access to the output and if they know/correctly guess the 10th round key $\mathbf{k}^{10}$—the data of interest for SCA attacks on AES—they can invert Eq. (2.1) as

$$oc_e^{9,b'} = Sbox^{-1}\left(ShiftRows^{-1}\left(k^{10,b} \oplus\ oc_e^{10,b}\right)\right). \tag{2.2}$$

2.1.2 Vulnerabilities of AES to SCA attacks

The fields emanated/power consumed in the final round of AES depend on the key, which causes an EM/power side-channel vulnerability [5], [6], [10], [19]. EM/power SCA SCA attacks on AES use hypothetical leakage models [28] to correlate observed fields to the computations/ processes during the final round of AES. These models abstract the sources of emanations in the DUT, such as transistor switching, currents on clock and power traces, EM coupling, etc., using simplified quantities. This work employs a byte-

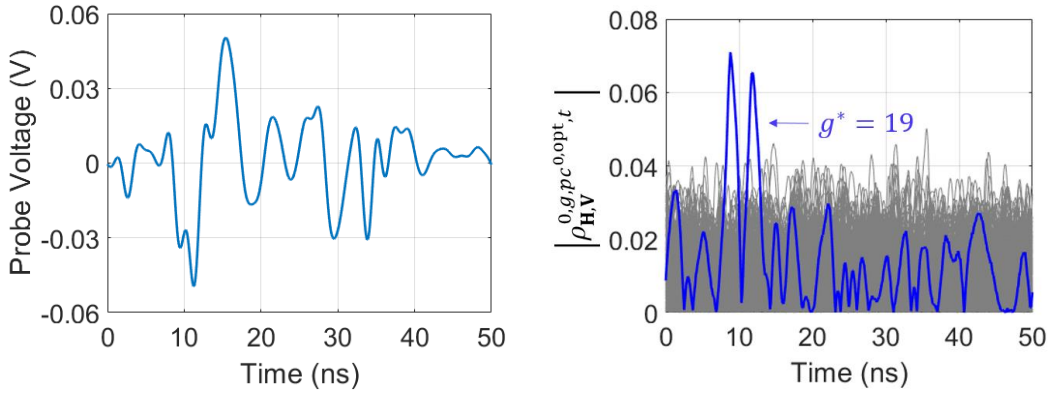


Figure 2.3: (a) Probed fields (left) at an optimal configuration observed during the last round of AES when the key and the input plaintext are set to $\mathbf{k}^0 = \mathbf{k}_1$ and $\mathbf{ip}_1$. (b) The correlation coefficients for all 256 guesses for $k^{10,0}$, when $N_e = 4000$ encryptions are observed. The coefficient corresponding to the correct guess $g^* = 19$ is shown in blue.

wise SCA attack (Fig. 2.2), which adopts a Hamming distance (HD) leakage model [21] that correlates the observed fields with the HD between $oc_e^{9,b'}$ and $oc_e^{10,b'}$ to disclose $k^{10,b}$. Byte-wise analysis significantly reduces the complexity of key search [22]. In this attack, the attackers observe N_e encryptions and for each encryption $e \in \{1, \cdots, N_e\}$, they use the observed $\mathbf{oc}_e^{10}$ together with every possible guess $g \in \{0, \cdots, 255\}$ for the key byte $k^{10,b}$ in Eq. (2.2) to compute the corresponding penultimate round value $oc_e^{9,b',g}$ for each byte $b \in \{0, \cdots, 15\}$. Let $H_e^{b,g}$ denote the HD between $oc_e^{9,b',g}$ and $oc_e^{10,b'}$ and let the integer array $\mathbf{H}^{b,g} = [H_1^{b,g}, \cdots, H_{N_e}^{b,g}]$ store the HDs for all encryptions; there are 16×256 such arrays.

In the fine-grained attack, the attackers also observe the probed fields $V_e^{pc,t}$ at times t during the last round of AES using a multitude of probe configurations pc—referring to the probe's transverse location l, height h, and orientation o above the DUT. Let the real array $\mathbf{V}^{pc,t} = [V_1^{pc,t}, \cdots, V_{N_e}^{pc,t}]$ store the probed fields for all encryptions; there are $N_l \times N_h \times N_o \times N_t$ such arrays. Attackers compute the Pearson correlation coefficient

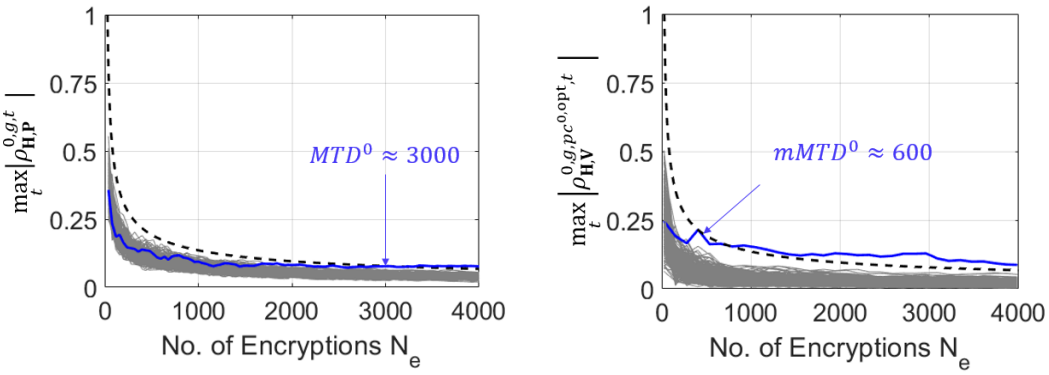
$\rho_{\mathbf{H},\mathbf{V}}^{b,g,pc,t}$ between the arrays $\mathbf{H}^{b,g}$ and $\mathbf{V}^{pc,t}$ for each key byte b, guess g, configuration pc, and time t [6], [20]:

$$\rho_{\mathbf{H},\mathbf{V}}^{b,g,pc,t/f} = \frac{\mathrm{Cov}\left(\mathbf{H}^{b,g},\mathbf{V}^{pc,t/f}\right)}{\sqrt{\mathrm{Var}\left(\mathbf{H}^{b,g}\right)\mathrm{Var}\left(\mathbf{V}^{pc,t/f}\right)}} \tag{2.3}$$

Attackers can compute the correlation coefficients in Eq. (2.3) using time samples; e.g., the probed fields $V_1^{pc,t}$ are shown in Fig. 2.3 (a) for $\mathbf{k}_1 = [0x00, 0x01, \cdots, 0x0F]$ and $\mathbf{ip}_1 = [0x00, 0x00, \cdots, 0x00]$. The largest correlation coefficient will correspond to the correct guess $g^* = k^{10,b}$ for byte b if the leakage model accurately categorizes the underlying sources of emanations (after observing a sufficient number of encryptions); e.g., the coefficients that result from observing $N_\mathrm{e} = 4000$ encryptions with randomly generated input plaintexts are shown in Fig. 2.3 (b).

In the power SCA or coarse-grained EM SCA attack, the aggregate power consumption or EM emanation is recorded during the final round of AES for each encryption; the observed signals are stored in the array $\mathbf{P}^t$ of size $N_\mathrm{t} \times N_\mathrm{e}$ for N_t time samples. Correlating $\mathbf{P}^t$ with the Hamming distances $\mathbf{H}^{b,g}$ yields the correlation coefficients $\rho_{\mathbf{H},\mathbf{P}}^{b,g,t}$, for each key-byte b, guess key g, and time instant t. The correct guess key value g^* is identified similar to the fine-grained EM SCA attack. Once all 16 bytes of $\mathbf{k}^{10}$ are disclosed, the AES key-expansion algorithm is inverted to disclose the key $\mathbf{k}^0$, which can then be used to decrypt any ciphertext $\mathbf{oc}_e^{10}$ and recover the corresponding plaintext $\mathbf{ip}_e$ from any past or future encryption.

While the correlation coefficient corresponding to the correct guess stands out in Fig. 2.3, it is important to ask if $k^{10,0}$ could be disclosed by observing fewer encryptions. Indeed, to evaluate side-channel security, *the minimum number of measurements necessary to disclose all key bytes must be quantified*. In the power SCA and coarse-grained EM SCA



(a) (b)

Figure 2.4: Maximum value of (a) power SCA and (b) fine-grained EM SCA correlation coefficients for all 256 guesses for $k^{10,0}$ as the number of encryptions increases. The value corresponding to the correct guess $g^* = 19$ (blue) crosses the null hypothesis threshold (dashed) after $MTD^0/mMTD^0$ measurements.

attack, the minimum number of encryptions needed to disclose a key-byte b is defined as the "measurements to disclosure" MTD^b (Fig. 2.4(a)), i.e., when $N_e \geq MTD^b$, the correlation coefficient corresponding to the correct guess g^* is sufficiently larger than those corresponding to the incorrect guesses. Here, and throughout this thesis, a correlation coefficient is considered sufficiently large if its maximum value over all time samples crosses the null hypothesis threshold derived from the inverse t-distribution for a confidence interval of 99.99% [10],[29]. Therefore, the power/coarse-grained EM SCA attacks require a marginal cost of

$$Marginal\ Cost^{\text{Pwr/cgEM}} = \min\left(N_e^{\max}, \max_b MTD^{b,\text{Pwr/cgEM}}\right) \tag{2.4}$$

encryptions to be observed, i.e., they observe more and more encryptions until either all bytes are disclosed or a limit on the number of observations, $N_e^{\max}$, is reached.

In contrast, fine-grained EM SCA may require a smaller marginal cost if an optimal configuration with low MTD is identified. Let $MTD^{b,pc}$ denote the minimum number of measurements to disclose key byte b when using the probe configuration pc [10]. Let

$$pc^{b,\mathrm{opt}} = \underset{pc}{\operatorname{argmin}}\, MTD^{b,pc}\,; mMTD^{b} = MTD^{b,pc^{b,\mathrm{opt}}} \tag{2.5}$$

denote the optimal probe configuration to disclose $k^{10,b}$ and the minimum number of measurements to do so; e.g., in Fig. 2.3, the correct guess for $k^{10,0}$ could be identified (Fig 2.4) by observing time-domain fields only for $mMTD^{0} \approx 600$ encryptions when using $pc^{0,\mathrm{opt}}$, while power SCA required $MTD^{0} \approx 3000$ to recover the same key byte. However, identifying the optimal configurations in fine-grained EM SCA using an exhaustive search may become infeasible since it requires

$$Acquis.Cost = N_{\mathrm{e}}^{\max} N_{\mathrm{l}} N_{\mathrm{h}} N_{\mathrm{o}} \text{ (exhaustive search)} \tag{2.6}$$

measurements, to perform expensive correlation analyses at all possible configurations.

2.1.3 Effect of Noise on Correlation Attacks

The correlation analysis is degraded and EM SCA attacks fail when noise obfuscates the target signals—originating from the computation of byte b of the output ciphertext $oc_e^{10,b}$ in Eq. (2.1)—in the probed fields $\mathbf{V}^{pc,t}$ [6]. The noise can be categorized as *measurement noise*, which arises from the environment—temperature variations, vibrations, equipment sensitivity, drift, variability of supply voltage, input clock jitter, etc. [6], [30]—and *algorithmic noise*, which arises from uncorrelated background computations/processes in the DUT [4], [19]. Measurement noise exhibits as variations in observed fields when the exact same encryption is repeated [31],[32]. For AES-128, the algorithmic noise for the byte b computation includes fields that originate from the computation of the 15 bytes other than byte b of the output ciphertext [19], [24].

To analyze the effect of noise, let's decompose the observed fields in the arrays $\mathbf{V}^{pc,t}$ into the independent and hypothetical quantities listed in the arrays $\mathbf{T}^{b,pc,t}$, $\mathbf{B}^{b,pc,t}$, $\mathbf{N}^{pc,t}$ [4], [6]. Here, target signals in **T**, algorithmic noise in **B**, and measurement noise in **N** arise from computations involving the data of interest ($k^{10,b}$), background computations in the DUT, and other EM sources, respectively. Then, the time-domain correlation coefficient in Eq. (2.3) can be expressed as [6]:

$$\rho_{\mathbf{H},\mathbf{V}}^{b,g,pc,t} = \underbrace{\frac{\mathrm{Cov}\left(\mathbf{H}^{b,g},\mathbf{T}^{b,pc,t}\right)}{\sqrt{\mathrm{Var}(\mathbf{H}^{b,g})\mathrm{Var}\left(\mathbf{T}^{b,pc,t}\right)}}}_{\rho_{\mathbf{H},\mathbf{T}}^{b,g,pc,t}} \frac{1}{\sqrt{1+\frac{\mathrm{Var}\left(\mathbf{B}^{b,pc,t}\right)}{\mathrm{Var}\left(\mathbf{T}^{b,pc,t}\right)}+\frac{\mathrm{Var}(\mathbf{N}^{pc,t})}{\mathrm{Var}\left(\mathbf{T}^{b,pc,t}\right)}}} \tag{2.7}$$

In this representation, the noise-free correlation coefficient $\rho_{\mathbf{H},\mathbf{T}}^{b,g,pc,t}$ is degraded by the variance terms. Probe configurations that have larger ratios $\mathrm{Var}(\mathbf{T}^{b,pc,t})/\mathrm{Var}(\mathbf{B}^{b,pc,t})$ and $\mathrm{Var}(\mathbf{T}^{b,pc,t})/\mathrm{Var}(\mathbf{N}^{pc,t})$ will yield correlation coefficients $\rho_{\mathbf{H},\mathbf{V}}^{b,g,pc,t/f}$ closer to the noise-free value. The variance ratios in Eq. (2.7) are often combined and represented as signal-to-noise ratio in SCA attacks [19],[24].

Because the entries in the arrays **T**, **N**, and **B** are unmeasurable hypothetical quantities, the ratios of their variances cannot be found exactly. They can be estimated, however, from measured fields via ANOVA [4], [6], [19], [24]. The ANOVA F-statistic, defined as a ratio of variances, is used for hypothesis testing to determine if a dataset is sensitive to variations in a target process. The methodology groups data based on different versions of a target process, and compares variance between groups and variance within groups, to quantify the dependence of the dataset on the target. Here, the F-statistics are used to estimate the two ratios in Eq. (2.7) as [4],[6]:

$$\frac{\mathrm{Var}\left(\mathbf{T}^{b,pc,t}\right)}{\mathrm{Var}(\mathbf{N}^{pc,t})} \approx F_N^{b,pc,t} \quad \frac{\mathrm{Var}\left(\mathbf{T}^{b,pc,t}\right)}{\mathrm{Var}\left(\mathbf{B}^{b,pc,t}\right)} \approx F_B^{b,pc,t} \tag{2.8}$$

The most accurate estimates in Eq. (2.8) require observing all possible variants in the relevant computations; e.g., to obtain $F_B^{b,pc,t}$, fields can be measured for up to

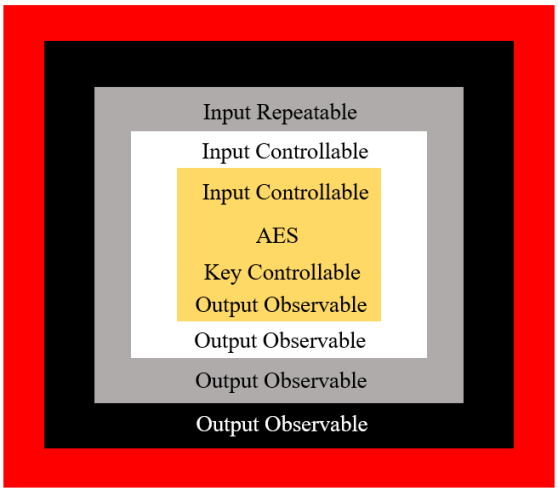


Figure 2.5: SCA threat models for AES. Unrestricted attackers (gold box) control the key and have complete access to device peripherals. The access to the DUT is progressively restricted (white, gray, and black box) until attackers have no access to inputs and outputs (red box).

256×256 possible variants in the switch from $oc_e^{9,b'}$ to $oc_e^{10,b'}$ and $256^{15} \times 256^{15}$ possible variants in background computations. Typically, far fewer samples are sufficient; e.g., the $F_B^{b,pc,t}$ statistic was previously obtained using $oc_e^{10,b'} = \{0,1,\ldots,255\}$, ignoring $oc_e^{9,b'}$ values, and 4-40 variants in background computations [19],[24].

2.2 Fine-Grained EM SCA Attacks Subject to Constraints

This section summarizes threat models under which evaluators may operate, and presents search protocol for each model. Further, the acquisition costs of each protocol is also included after each subsection.

2.2.1 Threat Models

Threat models, in security literature, define the environment for security evaluations, particularly the accessibility to various device parameters. In this chapter, all of the threat models assume that the attackers have physical access to the DUT and can observe the output ciphertext, but

(1) the most restrictive *black-box threat model* assumes attackers have no access to inputs;
(2) a less restrictive *gray-box threat model* assumes attackers have partial control over inputs, i.e., they can repeat inputs but not observe them; and
(3) the least restrictive *white-box threat model* assumes attackers have full access to inputs, i.e., they can repeat and observe them (the cipher key is unknown).
Two further threat models are considered to bound these models including (i) a red-box, which limits attackers from observing the output ciphertext, and (ii) a gold-box, where access is granted to both the input plaintext and key (Fig. 2.5). The gold-box threat model is described in detail in Chapter 4.

2.2.2 Attacking a Red Box: Pre-characterization Phase

An initial low-cost scan can discard ineffective probe configurations and reduce the search space in fine-grained EM SCA attacks. In this scan, $N_{\mathrm{e}}^{\mathrm{pre}}$ encryptions are observed with each probe configuration pc. The encryptions can potentially be all different; the only constraint is that the same encryption is not repeated $N_{\mathrm{e}}^{\mathrm{pre}}$ times for any pc. Once the observed fields are recorded, $\max_{t} \mathrm{STD}(\mathbf{V}^{pc,t})$ is computed for each pc. Probe configurations with the smallest standard deviations, close to the noise floor of measurement equipment, can be deemed insensitive to the sources of interest and discarded. This pre-characterization requires

$$Acq.Cost^{\mathrm{pre}} = N_{\mathrm{l}} N_{\mathrm{h}} N_{\mathrm{o}} N_{\mathrm{e}}^{\mathrm{pre}} \tag{2.9}$$

encryptions to be observed; here, $N_{\mathrm{e}}^{\mathrm{pre}} \ll N_{\mathrm{e}}^{\mathrm{max}}$.

As the AES input and output are not used, this phase can be considered a fine-grained EM SCA attack for the red box threat model. While configurations that give rise to the largest variations in $\mathbf{V}^{pc,t}$ are of interest, these variations can stem from not only the changes in targeted signal sources ($\mathbf{T}^{pc,t}$) but also measurement noise ($\mathbf{N}^{pc,t}$) and

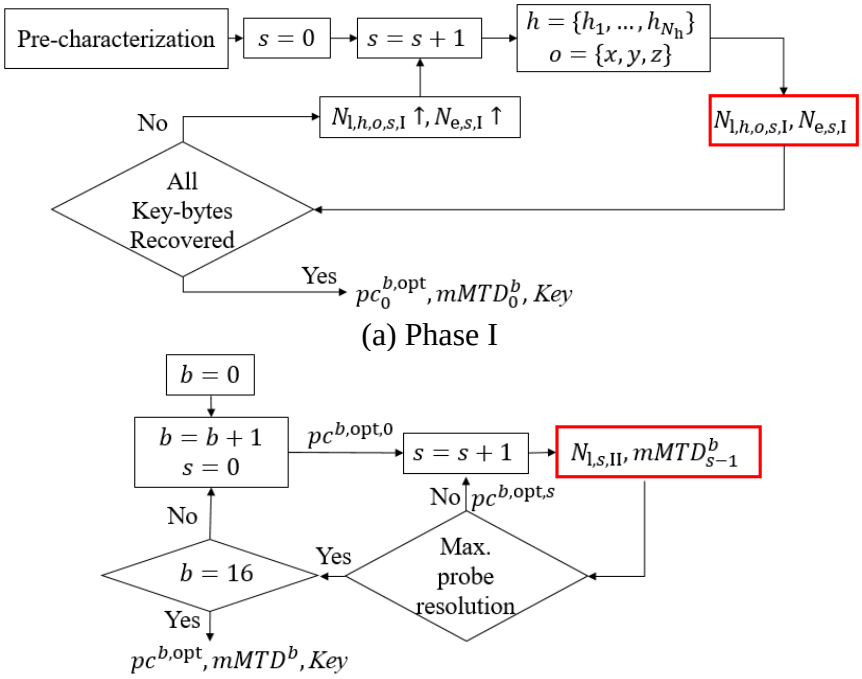


Figure 2.6: The fine-grained EM SCA measurement protocol in the black-box threat model. Scans are marked with red and the number of locations and encryptions observed in each scan are specified. Phase I scans are performed with multiple probe orientations, becoming progressively more expensive, while Phase II scans become progressively cheaper.

algorithmic noise ($\mathbf{B}^{pc,t}$). In general, attackers cannot use just the signals measured during the pre-characterization phase to perform correlation analysis. Instead, this phase enables attackers to rapidly judge if potentially exploitable signals exist, reducing the acquisition cost of the following measurement protocols.

2.2.3 Attacking a Black Box

The black-box threat model, where attackers can observe the outputs but have no access to the inputs or the key, is commonly used for side-channel security evaluation. In this threat model, statistical methods that can rapidly identify probe configurations degraded by noise are unavailable because of the restrictions on the attackers. Search

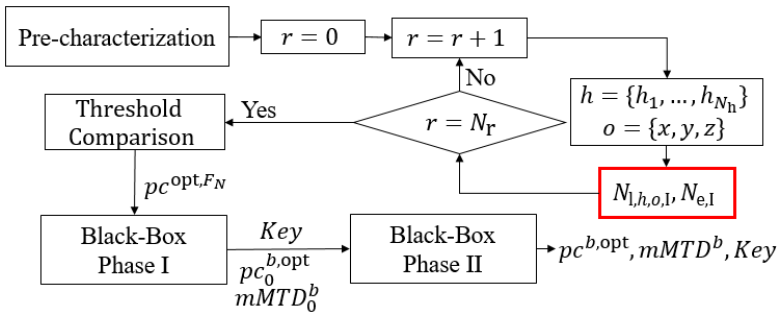


Figure 2.7: The measurement protocol in Phase I of the gray-box threat model prunes the search space by repeating scans, computing $F_N^{pc,t}$, and comparing it to a threshold $F_{N,c}$. The reduced set of configurations are then evaluated with the black-box protocol.

protocols based on correlation analysis [10],[11], including the exhaustive search, can be used; here, the method in [10] is implemented.

The measurement protocol for the black-box threat model is an adaptive scan performed in 2 phases: In Phase I (Fig. 2.6(a)), $N_{\text{scan,I}}$ progressively costlier low-resolution scans are performed to identify the probe configurations $pc_0^{b,opt}$ that disclose the key-byte b with $mMTD_0^b$ measurements. In each scan s of Phase I, either the number of locations probed $N_{l,h,o,s,\text{I}}$ or number of encryptions observed $N_{e,s,I}$ is increased [10],[29]. Then, for each key-byte, $N_{\text{scan,II}}$ progressively cheaper scans are performed in Phase II (Fig. 2.6(b)) to optimize the configurations found in Phase I. Each scan in Phase II uses only the optimal orientations $o_0^{b,opt}$ at height $h_0^{b,opt}$, restricts the area of the scan near the optimal locations in the previous scan $l_{s-1}^{b,opt}$,and observes only the minimum number of encryptions used to disclose the key byte in the previous scan. This requires

$$Acq.Cost^{\text{Bbox}} = Acq.Cost^{\text{pre}} + \sum_{s=1}^{N_{\text{scan,I}}} \sum_{h=1}^{N_h} \sum_{o=1}^{N_o} N_{e,s,\text{I}} N_{l,h,o,s,\text{I}} + \sum_{b=1}^{16} \sum_{s=1}^{N_{\text{scan,II}}} mMTD_{s-1}^b N_{l,s,\text{II}} \quad (2.10)$$

measurements. In the black-box threat model, this search protocol may converge to local minima for MTDs and not identify the most optimal probe configurations [21].

2.2.4 Attacking a Gray Box

The gray-box threat model permits attackers partial control over the input: while they cannot modify or observe the plaintexts, attackers can repeat them. This enables signal averaging to improve the signal-to-noise ratio. It also enables the use of repeatability characterizations and ANOVA F-statistics to prune the search space because probe configurations showing low signal variance for repeated encryptions and high signal variance for changing encryptions are most likely to disclose the keys [6].

The measurement protocol for the gray-box threat model is performed in 3 phases: In Phase I (Fig. 2.7), one scan per orientation is performed, where $N_{\mathrm{e,I}}$ encryptions are repeated N_{r} times at $N_{\mathrm{l},h,o,\mathrm{I}}$ locations. For each encryption e, the sample mean $\bar{x}_e^{pc,t}$ and variance $s_e^{pc,t}$ are computed across the repeated measurements and the F-statistic that quantifies the effect of measurement noise on signals is estimated as [6]

$$F_N^{pc,t} = \frac{N_{\mathrm{e,I}} \times N_{\mathrm{r}} \times \mathrm{Var}\left(\bar{x}_1^{pc,t}, \bar{x}_2^{pc,t}, \ldots, \bar{x}_{N_{\mathrm{e,I}}}^{pc,t}\right)}{\mathrm{Mean}\left(s_1^{pc,t}, s_2^{pc,t}, \ldots, s_{N_{\mathrm{e,I}}}^{pc,t}\right)} \tag{2.11}$$

The computed values are compared to a threshold $F_{N,\mathrm{c}}$ derived from F-distributions for a selected confidence level. Configurations with F-values greater than the threshold are least affected by measurement noise. This model enables attackers to identify configurations significantly degraded by measurement noise (see Eq. (2.7)) and remove them from the search after Phase I. Typically, the resolution of the Phase I scan is higher than its black-box counterpart as it requires fewer encryptions to be observed. Once configurations pc^{opt,F_N} with high F-values are isolated, phases I and II of the measurement protocol for the black-box threat method are performed (Fig. 2.6). This requires

$$Acq.Cost^{\mathrm{Gbox}} = Acq.Cost^{\mathrm{pre}} + \sum_{h=1}^{N_{\mathrm{h}}} \sum_{o=1}^{N_{\mathrm{o}}} N_{\mathrm{l},h,o,\mathrm{I}} N_{\mathrm{r}} N_{\mathrm{e,I}} + \sum_{s=1}^{N_{\mathrm{scan,II}}} \sum_{h=1}^{N_{\mathrm{h}}} \sum_{o=1}^{N_{\mathrm{o}}} N_{\mathrm{e},s,\mathrm{II}} N_{\mathrm{l},h,o,s,\mathrm{II}} + \sum_{b=1}^{16} \sum_{s=1}^{N_{\mathrm{scan,III}}} mMTD_{s-1}^{b} N_{\mathrm{l},s,\mathrm{III}} \tag{2.12}$$

2.2.5 Attacking a White Box

The white-box threat model permits attackers complete control over the inputs. The measurement protocol is performed in 4 phases (Fig. 2.8): Because the key is unknown, Phase I of the protocol for the gray-box threat model is implemented followed by Phase I of the protocol for the black-box threat model to recover the key. In these first two phases, the protocol prioritizes recovering the key over isolating optimal configurations; this allows low-resolution scans to first disclose the key and then further optimize the attack by computing the F-statistic $F_B^{pc,t}$. Because each byte of AES is targeted separately, the algorithmic noise is assumed to come from uncorrelated computations involving the remaining 15 bytes. Although each byte can potentially switch from 256 values in the penultimate round to 256 values in the final output, the Hamming distance (HD) of this transition reduces the number of combinations from 256×256 to 9 values, from HD_0 to HD_8. This simplification is consistent with the HD leakage model used in Section 2.1 for correlation analysis. For each HD_i of a target byte, $N_{\mathrm{e,III}}$ encryptions are performed, where uncorrelated bytes are chosen randomly to increase algorithmic noise. The mean $\bar{\bar{x}}_{\mathrm{HD}_{i_1}}^{pc,t}$ and

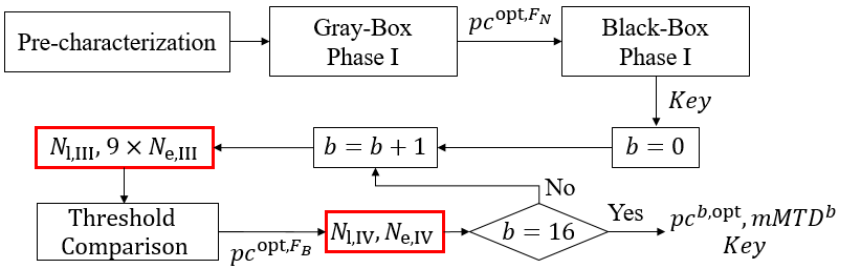


Figure 2.8: The measurement protocol in the white-box threat model initially performs Phase I of the protocols used for gray- and black-box threat models. Once the key is disclosed, the search space is pruned by computing $F_B^{pc,t}$ statistic byte-wise and comparing it to a threshold $F_{B,\mathrm{c}}$. The reduced set of configurations are then evaluated using correlation analysis.

variance $\bar{s}_{\mathrm{HD}_i}^{pc,t}$ are computed on the averaged signals across the changing encryptions, and the F-statistic $F_B^{pc,t}$ is estimated as

$$F_B^{pc,t} = \frac{9\times N_{\mathrm{e,III}}\times \mathrm{Var}(\bar{x}_{\mathrm{HD}_0}^{pc,t},\bar{x}_{\mathrm{HD}_1}^{pc,t},\cdots,\bar{x}_{\mathrm{HD}_8}^{pc,t})}{\mathrm{Mean}(\bar{s}_{\mathrm{HD}_0}^{pc,t},\bar{s}_{\mathrm{HD}_1}^{pc,t},\cdots,\bar{s}_{\mathrm{HD}_8}^{pc,t})} \tag{2.13}$$

In Phase III, $F_B^{pc,t}$ is estimated in a single high-resolution byte-wise scan using configurations identified in Phase II. Comparing the computed values with a threshold $F_{B,\mathrm{c}}$ derived from F-distributions enables attackers to remove configurations significantly degraded by algorithmic noise after Phase III. Phase IV subjects optimal configurations pc^{opt,F_B} to correlation analysis. This requires

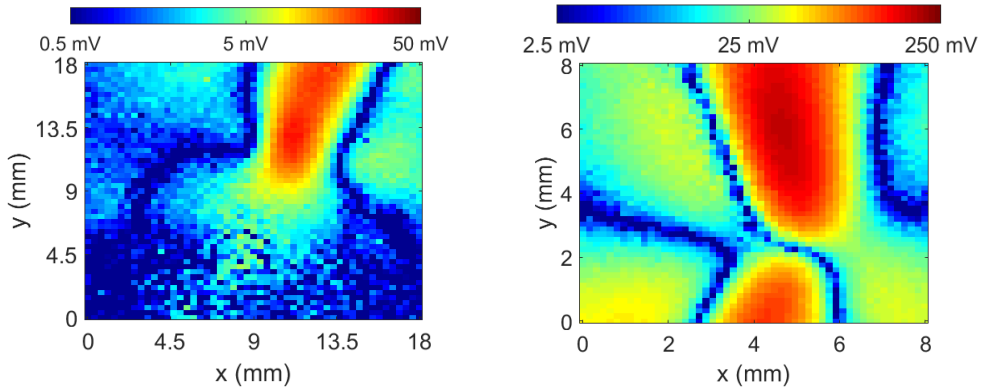
$$Acq.Cost^{\mathrm{Wbox}} = Acq.Cost^{\mathrm{pre}} + \sum_{h=1}^{N_{\mathrm{h}}}\sum_{o=1}^{N_{\mathrm{o}}} N_{\mathrm{l},h,o,\mathrm{I}} N_{\mathrm{r}} N_{\mathrm{e,I}} + \sum_{s=1}^{N_{\mathrm{scan,II}}}\sum_{h=1}^{N_{\mathrm{h}}}\sum_{o=1}^{N_{\mathrm{o}}} N_{\mathrm{e},s,\mathrm{II}} N_{\mathrm{l},h,o,,s,\mathrm{II}} + \sum_{b=1}^{16} 9N_{\mathrm{e,III}} N_{\mathrm{l,III}} + \sum_{b=1}^{16} N_{\mathrm{e,IV}} N_{\mathrm{l,IV}} \tag{2.14}$$

encryptions to be observed.

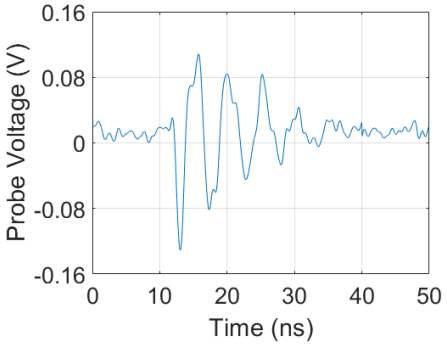
2.3 MEASUREMENT RESULTS

2.3.1 Setup

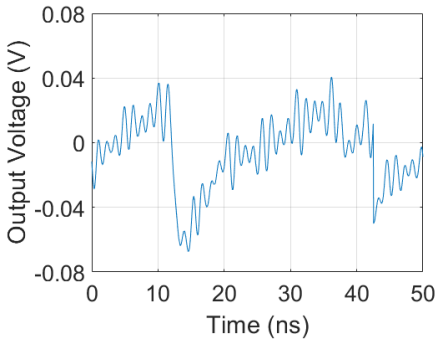
Fine-grained EM SCA attacks were implemented on AES-128 implementations using a 1-mm diameter H-field probe, at a fixed height $h_1 = 0.5$ mm, to scan an 8×8 mm^2 ASIC [12] and an 18×18 mm^2 Artix-7 FPGA [33]. Both chips operated at input clock frequency of 20 MHz and supply voltage of 1.1 V. A Keysight DSOS054A oscilloscope recorded the signals with a sampling rate of 10 GS/s. Analysis was performed locally on the oscilloscope, saving experiment time. The probe was positioned using Riscure's EM probe positioner. The setup allows scanning only in *x*- and *y*-orientation, i.e., $N_{\mathrm{o}} = 2$. The search space included $N_{\mathrm{l}} = 51 \times 51$ locations in both orientations. The spatial distributions of measured EM signals are shown in Fig. 2.9(a). Coarse-grained EM SCA attacks were performed using a 10-mm H-field probe while power attacks were performed using



(a) Fine-grained EM SCA



(b) Coarse-grained EM SCA



(c) Power SCA

Figure 2.9: (a) Spatial map of the absolute value of the measured signals using an *x*-oriented 1-mm diameter H-field probe at ~12 ns during the last round for the FPGA (left) and the secured ASIC (right). N_l= 51×51 locations were probed in both cases. (b) EM signal measured by a *z*-oriented 10-mm diameter H-field probe positioned at the center of the FPGA. (c) Supply variation of FPGA during the last round of AES operations.

available supply pins on the test boards. Signals captured for power and coarse-grained EM SCA attack are shown in Fig. 2.9(b)-(c).

In addition to an unsecured AES implementation, the ASIC also used a module hardened against power and coarse-grained EM SCA attacks by using a power delivery mechanism based on the galvanic isolation principle [9], [12]. Galvanic isolation is typically used in high-voltage power converters, where the secondary side of the converter is separated from the primary side to protect it from potentially damaging transient voltages

Marginal Cost	DUT		
	FPGA	**Baseline ASIC**	**Secured ASIC**
Power	4.20×10^{3}	1.00×10^{5}	$>2.00\times10^{6}$
Coarse-Grained EM	4.58×10^{4}	1.48×10^{5}	$>2.00\times10^{6}$
Fine-Grained EM	1.05×10^{4}	2.65×10^{4}	2.80×10^{4}

Table 2.1: Marginal costs of SCA attacks

and currents [12]. Here, the AES core is isolated from the external power supply to protect the module from power SCA attacks. Reconfigurable capacitor banks are used to supply the necessary charge to perform AES computations. Therefore current signatures and ground bounce in the external supply have minimal data-dependent variance.

2.3.2 Marginal Cost

First, the marginal costs of EM and power SCA attacks are compared (correlation analysis was performed using the optimal probe configurations or the fine-grained EM SCA attack) to judge their effectiveness. The number of observations with each attack modality was limited to 2 million encryptions; in some cases, the AES key could not be extracted within this limit. The observed marginal costs for all the implementations are listed in Table 2.1. Table 2.1 shows that the coarse-grained EM SCA attack was the least effective SCA modality against all the implementations. Surprisingly, the power SCA attack was the most effective against the FPGA (recovering the key with ~2.5× fewer

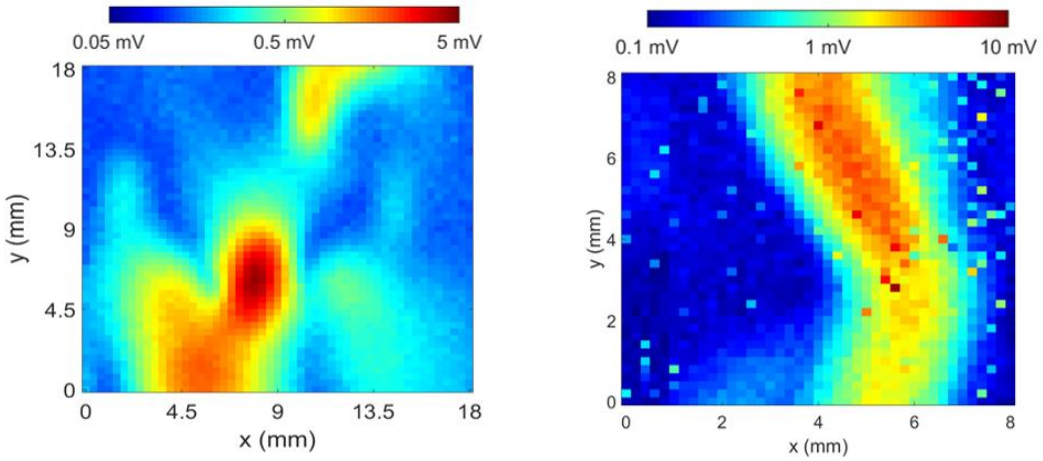


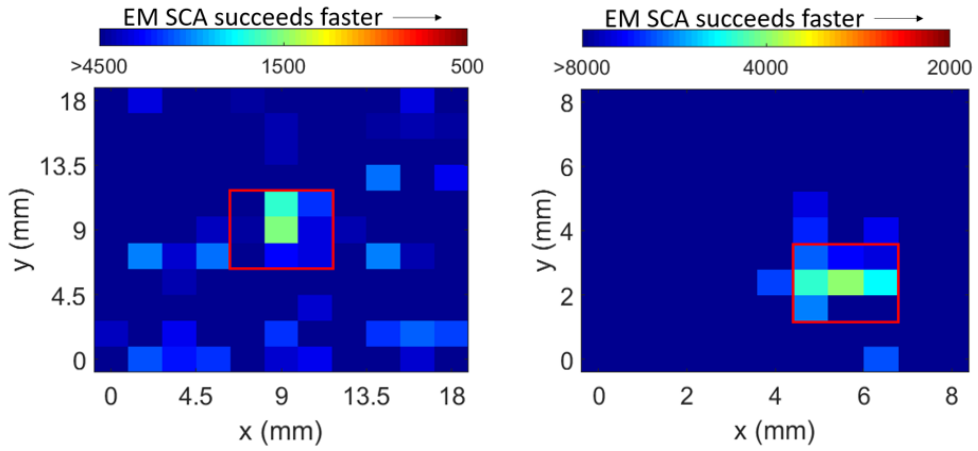
Figure 2.10: Spatial maps of $\max_t \mathrm{Std}(\mathbf{V}^{t,pc})$ obtained with the *x*-oriented probe for the FPGA (left) and ASIC (right).

encryptions than the best alternative); this may be because the FPGA and its test board are specifically designed and marketed to study power SCA attacks, i.e., they must have particularly low-noise outputs suitable for the power attack. The fine-grained EM SCA attack required ~3.7× fewer encryptions for the baseline ASIC and >70× times fewer encryptions for the secured ASIC compared to the power SCA attack.

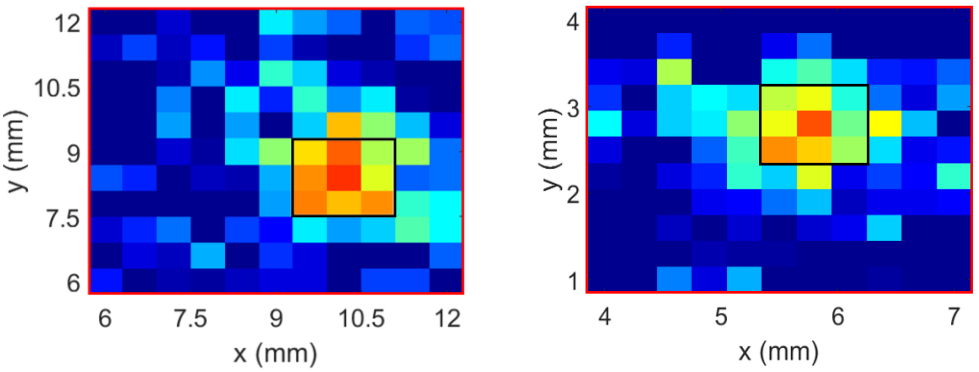
Using the exhaustive search to isolate the optimal probe configurations for the fine-grained EM SCA attack would require ~10^8 measurements for both implementations, if $N_{\mathrm{e}}^{\max} = 20\,000$. Next, the results from the search protocols to reduce this cost are reported for the FPGA and the secured ASIC (similar acquisition costs were observed for both secured and unsecured implementations).

2.3.3 Comparison of Fine-grained EM SCA Protocols

The pre-characterization (Fig. 2.10) was performed using $N_{\mathrm{e}}^{\mathrm{pre}} = 50$ encryptions for the maximum number of observers on both chips. The signal's standard deviation across the chip was computed and configurations with low variance (< 0.1 mV) were



(a) Phase I scan 2



(b) Phase II scan 1

Figure 2.11: MTD maps for byte 1 obtained from the black-box search protocol for the FPGA (left) and ASIC (right) implementations. Scans constrain area (red and black) and number of measurements progressively to reduce cost.

discarded. The pre-characterization showed a significant reduction in the initial search space for the ASIC (~40%) compared to the FPGA (~15%). Before implementing the protocols, the configurations eliminated by the pre-characterization phase were noted. If a scan included such a configuration, that measurement was skipped and the probe was positioned at the next configuration.

The protocol for the black-box threat model (Fig. 2.11) [10], [29] required $N_{scan,I} = 2$ Phase I scans for the FPGA, with the second scan requiring $N_{e,2,II} = 6000$ encryptions per configuration and probed observers on an equally spaced grid of size 11 × 11 over the

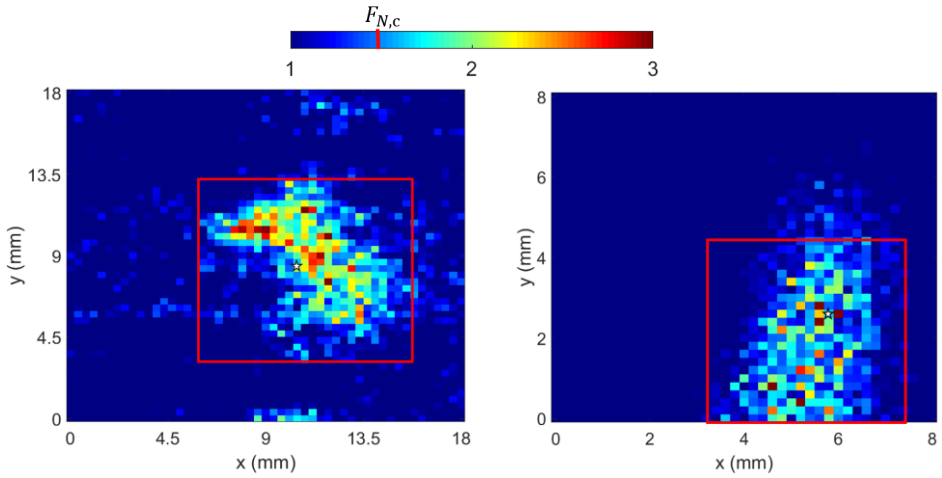


Figure 2.12: Spatial map of $\max_t F_N^{pc,t}$ and the are used in subsequent analysis (red) with an *x*-oriented probe for the FPGA (left) and ASIC (right).

chip. It required $N_{scan,I} = 2$ Phase I scans for the ASIC, where $N_{e,2,II} = 8000$ encryptions per configuration were used in the second scan. Both implementations required $N_{scan,II} = 2$ scans to disclose all bytes of the key.

Attacks using the gray-box protocol first computed F-statistic $F_N^{pc,t}$ for configurations within the search space reduced by pre-characterization. To compute the F-statistic, $N_{e,I}$ =20 encryptions were repeated $N_r = 50$ times [6]. As shown in Fig. 2.12, comparing the values with the critical threshold 1.6 (confidence level 95%), several non-optimal configurations were discarded. Phases II and III implemented the black-box search protocol over a reduced area, using $N_{scan,II} = 1$ and $N_{scan,III}$=2 scans.

Attacks using the white-box protocol started with the pre-characterization and Phase I for the gray-box model. Phase II performed a low-resolution scan with $N_{l,1,II} = 6 \times 6$, in the region marked in Fig. 2.12. Once the final round keys were identified, inputs were provided to the chip such that for each variation of Hamming distance switching of an output byte, $N_{e,III} = 20$ encryptions were generated to compute the $F_B^{pc,t}$ statistic (Fig.

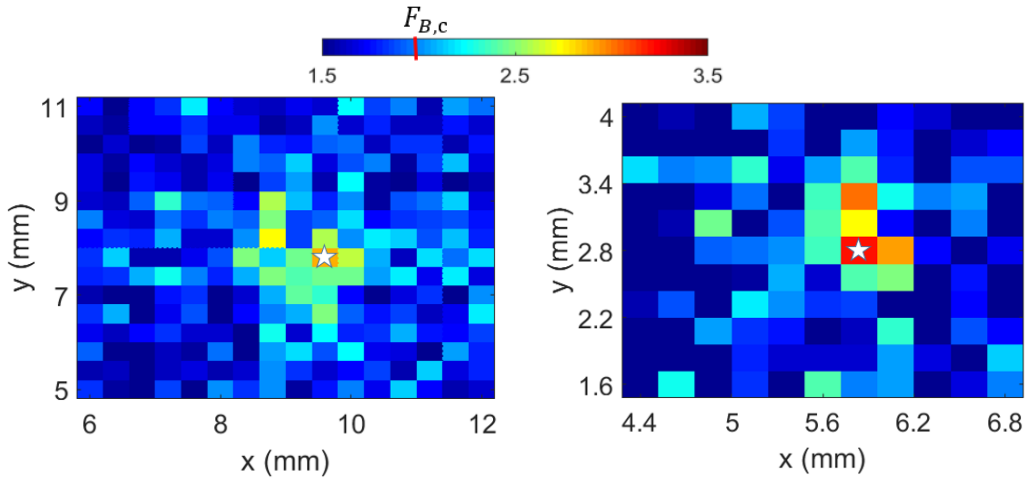


Figure 2.13: Spatial map of $\max_t F_B^{pc,t}$ compared to optimal configurations (star) for the FPGA (left) and ASIC (right).

2.13) in Phase III. The statistic was computed at a comparatively finer resolution for the FPGA since a larger region was observed to leak information in previous phases.

2.3.4 Comparison of Acquisition Costs

The pre-characterization stage required ~2.6×10^5 encryptions for both AES implementations. The acquisition costs were ~9.9×10^6, ~7.3×10^6, and ~6.9×10^6 (~1.27×10^7, ~9.8×10^6, and ~6.8×10^6) measurements for the FPGA (ASIC) when the black-, gray-, and white-box threat model was used. The number of probe configurations and the accumulation of the acquisition cost at each phase of the search protocols are shown in Figs. 2.14(a)-(c). The final acquisition costs are compared to that of the exhaustive approach in Fig. 2.14(d). Compared to the exhaustive search, the search protocols for the black-, gray-, and white-box threat models showed ~8-10×, ~10-13×, and ~14-15× cost reduction. The search protocols for the gray- and white-box threat models required ~1.3-1.35× and ~1.5-2× fewer measurements compared to that for the black-box one, respectively.

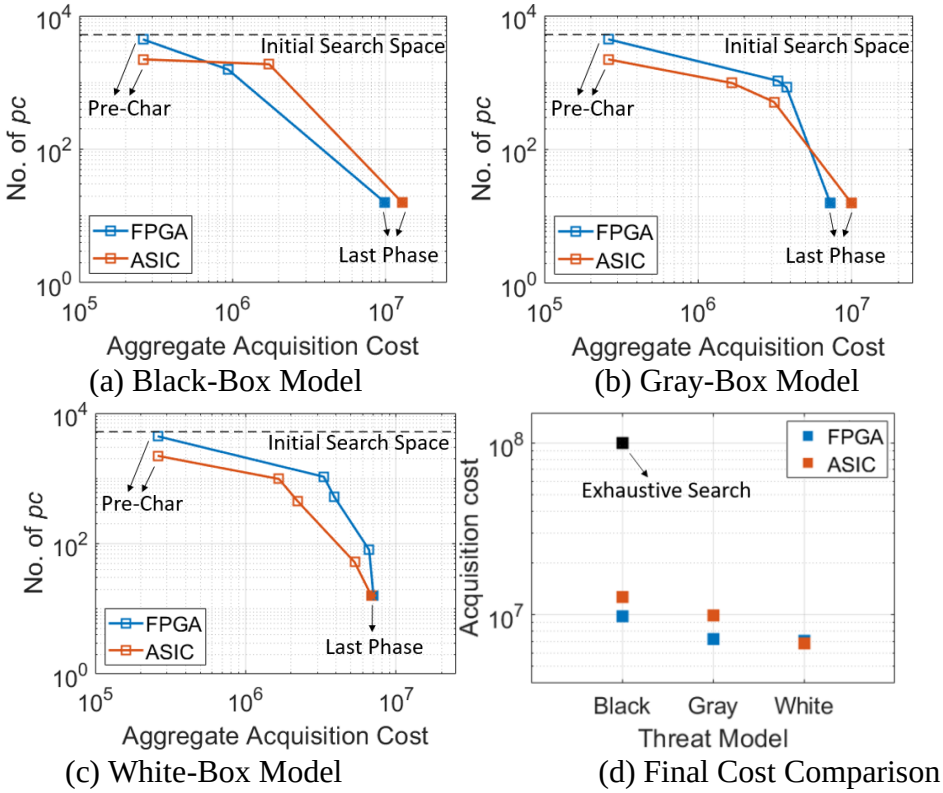


Figure 2.14: Reduction of the search space for optimal probe configurations. The optimal configurations were more rapidly isolated for less restrictive threat models.

2.4 SUMMARY

In this chapter, fine-grained EM SCA attacks were systematically compared to coarse-grained EM and power SCA attacks. Though fine-grained EM SCA attacks were found to be more than 70× effective compared to the alternatives on AES-128, they are constrained by the potentially infeasible acquisition cost of the measurements. Various threat models were introduced to categorize search protocols that can rapidly isolate optimal probe configurations in fine-grained EM SCA attacks. Experiments showed that different protocols can reduce the acquisition cost compared to an exhaustive search by ~8-15×. These protocols enable designers to rapidly evaluate the security of cryptographic modules that implement EM and power SCA countermeasures.

3. Evaluation of AES using ANOVA F-Statistics[2]

This chapter presents a 3-stage measurement protocol to rapidly evaluate fine-grained EM security for a gold-box threat model. The protocol is used to evaluate the resilience of several baseline and hardened implementations of AES in both time and frequency domain. Further, the costs of the protocol are compared to several alternatives at the end of the chapter.

3.1 MEASUREMENT PROTOCOL

This section presents the proposed 3-stage measurement protocol that uses ANOVA indicators to evaluate side-channel security. Unlike the methods in Chapter 2, here, the protocol is performed using both time samples t and frequency samples f. The $F_N^{b,pc,t/f}$ and $F_B^{b,pc,t/f}$ metrics (See Eq. (2.8)) are computed and used to reduce the search space in Stages I and II, respectively. The remaining configurations are used to perform correlation analysis in Stage III and acquire optimal probe configurations. The acquisition cost and measurement time of the proposed protocol are quantified and contrasted to the TVLA indictor. All analyses shown in this section were obtained from an attack on the first key byte of AES-128, using the Artix-7 FPGA, operated at 20 MHz clock and 1 V supply voltage, and the optimal measurement configuration in [10], [20]: a 1-mm diameter H-field probe, oriented in the *x* direction, and located at (9.7, 8, 0.5) mm from the bottom left corner of the chip.

[2] This chapter is partly based on a previous publication: V. V. Iyer and A.E. Yilmaz, "An ANOVA method to rapidly assess information leakage near cryptographic modules," *IEEE Trans. Electromagn. Compat.*, vol. 64, no. 4, Aug. 2022.
The author contributed to the formulation, implementation, and measurements presented in this article, as well as the writing of this manuscript.

3.1.1 The Gold-Box Threat Model

The proposed method assumes side-channel security evaluators have full control over the input and the encryption key of the DUT (a "gold-box threat model" [9]). This permits evaluators to not just emulate but enhance correlation-analysis attacks, which are applicable even under the highly restrictive black-box threat model [9] but quickly become infeasible for fine-grained EM SCA evaluation. In particular, fewer restrictions permit evaluators to design targeted tests, estimate the impact of noise, and rapidly identify ineffective probe configurations.

3.1.2 Choosing Test Cases to Compute F-statistics

To compute each F-statistic, a set of test cases is constructed. Because evaluators are permitted to modify the AES encryption key as well as the input plaintext, each encryption e in the set can use a potentially different plaintext $\mathbf{ip}_e$ and key $\mathbf{k}_e^0$. To construct the test cases, all 16 bytes of the ciphertext in the penultimate round are enforced to be constant and set to zero for simplicity, i.e., $\mathbf{oc}_e^9 = [0\text{x}00, \cdots, 0\text{x}00]$. Thus, the HD between $oc_e^{9,b'}$ and $oc_e^{10,b'}$ is the Hamming weight of $oc_e^{10,b'}$; e.g., $oc_e^{10,b'} = 0\text{x}00$ gives HD_0 and $oc_e^{10,b'} = 0\text{xFF}$ gives HD_8. As a result, evaluators can specify test cases (set each plaintext $\mathbf{ip}_e$ and key $\mathbf{k}_e^0$) by only setting the output ciphertext $\mathbf{oc}_e^{10}$. Once $\mathbf{oc}_e^{10}$ is set, the last round key $\mathbf{k}_e^{10}$ is found from Eq. (2.1) as:

$$k_e^{10,b'} = 0\text{x}63 \oplus oc_e^{10,b'} \tag{3.1}$$

This is because each byte in the specified $\mathbf{oc}_e^9$ (0x00) is always mapped to 0x63 by AES. Once all 16 bytes of $\mathbf{k}_e^{10}$ are deduced, the key $\mathbf{k}_e^0$ and plaintext $\mathbf{ip}_e$ corresponding to $\mathbf{oc}_e^{10}$ are extracted as detailed in Section 2.1.2. The first two stages of the proposed protocol use test cases constructed with this approach.

The test cases should be chosen based on the leakage model used in the correlation analysis; thus, in this paper, they are chosen using the HD leakage model, where the data of interest $k^{10,b}$ is disclosed by targeting the switching in the last AES round from $oc_e^{9,b'}$ to $oc_e^{10,b'}$. Other leakage models may be more suitable depending on the implementation and algorithm; e.g., test cases were constructed using Hamming weights in [4] to model the fields emanated during data transfer on a processor bus. The HD leakage model used in this paper assumes that the target signals arising from computations involving $k^{10,b}$ have only 9 instead of 256 possible variants $\{\mathrm{HD}_0, \cdots, \mathrm{HD}_8\}$ corresponding to the HD between $oc_e^{9,b'}$ and $oc_e^{10,b'}$, all test cases with the same HD yield indistinguishable target signals, and test cases that correspond to HD_0 and HD_8 are extreme variants, whose target signals differ the most.

3.1.3 Stage I: Measurement-Noise-Based Leakage Indicator

In Stage I, the $F_N^{b,pc,t/f}$ statistic is evaluated by using test cases that correspond to extreme variants for the computations of interest and minimize algorithmic noise; i.e., test cases consist of the 2 extreme variants for each byte b—corresponding to HD_0 and HD_8 between $oc_e^{9,b'}$ and $oc_e^{10,b'}$—while the other 15 bytes of $\mathbf{oc}_e^{10}$ are kept constant and set to 0x00 (HD_0). Because the test case corresponding to $\mathbf{oc}_{\mathrm{e}}^{10} = \mathbf{0}$ can be reused as one of the extreme variants for each byte and because the remaining test cases are generated by changing only one of 16 bytes of $\mathbf{oc}_{\mathrm{e}}^{10}$ to 0xFF (HD_8), a total of $N_{\mathrm{e,I}} = 17$ plaintext-key pairs are used as test cases in Stage I. The HDs for these 17 test cases can be stored in a 17×16 integer array :

$$\mathbf{H}_{\mathrm{I}} = \begin{bmatrix} \mathrm{HD}_0 & \mathrm{HD}_0 & & \mathrm{HD}_0 \\ \mathrm{HD}_8 & \mathrm{HD}_0 & \cdots & \mathrm{HD}_0 \\ \mathrm{HD}_0 & \mathrm{HD}_8 & & \mathrm{HD}_0 \\ & \vdots & \ddots & \vdots \\ \mathrm{HD}_0 & \mathrm{HD}_0 & \cdots & \mathrm{HD}_8 \end{bmatrix} \tag{3.2}$$

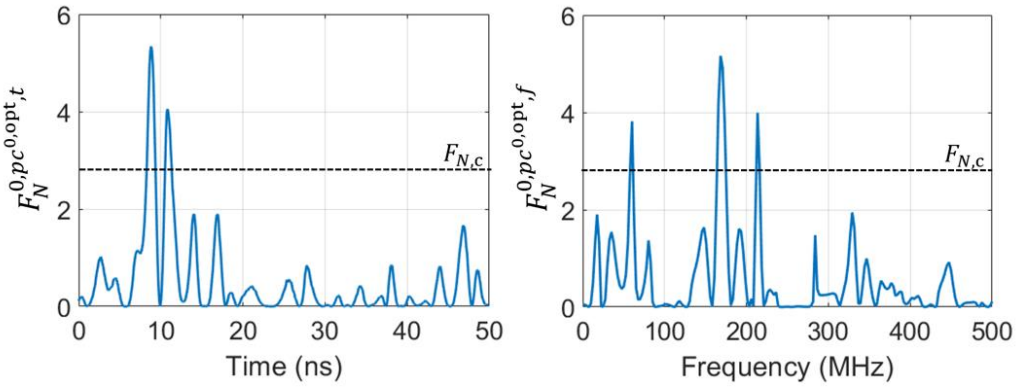


Figure 3.1: Time-domain (left) and frequency-domain (right) $F_N^{0,pc,t/f}$ metric, evaluated with the probe configuration $pc^{0,\mathrm{opt}}$.

These $N_{\mathrm{e,I}}$ encryptions are repeated $N_{\mathrm{r,I}}$ times for each possible probe configuration and the F-statistic is evaluated as:

$$F_N^{b,pc,t/f} = \frac{2N_{\mathrm{r,I}} \times \mathrm{Var}(\bar{x}_{\mathrm{HD}_0}^{b,pc,t/f}, \bar{x}_{\mathrm{HD}_8}^{b,pc,t/f})}{\mathrm{Mean}(s_{\mathrm{HD}_0}^{b,pc,t/f}, s_{\mathrm{HD}_8}^{b,pc,t/f})} \tag{3.3}$$

Here, sample means $\bar{x}_{\mathrm{HD}_{0/8}}^{b,pc,t/f}$ and variances $s_{\mathrm{HD}_{0/8}}^{b,pc,t/f}$ of the probed fields are computed across the $N_{\mathrm{r,I}}$ samples. The fields for $\bar{x}_{\mathrm{HD}_0}^{b,pc,t/f}$ and $s_{\mathrm{HD}_0}^{b,pc,t/f}$ ($\bar{x}_{\mathrm{HD}_8}^{b,pc,t/f}$ and $s_{\mathrm{HD}_8}^{b,pc,t/f}$) are observed using the test case generated by setting $oc_e^{10,b'}$ to 0x00 (0xFF) and all other bytes of $\mathbf{oc}_{\mathrm{e}}^{10}$ to 0x00, i.e., the test case in row 1 ($b' + 2$) of $\mathbf{H}_{\mathrm{I}}$.

An example of the F-statistic computed using $N_{\mathrm{r,I}} = 30$ repetitions is shown in Fig. 3.1. Comparing the data to Fig. 2.3 shows that, $F_N^{b,pc,t/f}$ is large whenever $\rho_{\mathbf{H},\mathbf{V}}^{b,g,pc,t/f}$ is large but the converse is not true, i.e., the indicator captures the information leakage but also overestimates it. The computed F-values are compared to a threshold $F_{N,c}$ to generate a leakage indicator:

$$Indicator_{\mathrm{I}}^{b,pc} = \begin{cases} 1 & \text{if } \max_{t/f} F_N^{b,pc,t/f} \geq F_{N,c} \\ 0 & \text{otherwise} \end{cases} \tag{3.4}$$

Only configurations with indicator value 1 are selected for measurements in Stage II, i.e., only $N_{\mathrm{pc,II}}^{b} = \sum_{pc} Indicator_{\mathrm{I}}^{b,pc}$ probe configurations are used.

3.1.4 Stage II: Algorithmic-Noise-based Leakage Indicator

In Stage II, the $F_B^{b,pc,t/f}$ statistic is evaluated by using test cases that correspond to extreme variants for both the computations of interest and background computations. Test cases consist of the 2 extreme variants for each byte b, while 14 of the remaining 15 bytes of $\mathbf{oc}_e^{10}$ are kept constant at 0x00 (HD_0) and 1 other byte is set to the 2 extreme variants. Consider the 32 test cases for byte $b = 0$: In half of these cases, $oc_e^{10,0}$ (byte 0 is not impacted by ShiftRows, so $b' = b$) is 0x00 (HD_0) or 0xFF (HD_8); for each half, $N_\mathrm{B} = 16$ background process variants are generated by setting all or all but one of the remaining bytes of $\mathbf{oc}_\mathrm{e}^{10}$ to HD_0. The HDs for these 32 test cases can be stored in an integer array of size 32×16:

$$\mathbf{H}_\mathrm{II}^0 = \begin{bmatrix} \mathrm{HD}_0 & \mathrm{HD}_0 & \mathrm{HD}_0 & \cdots & \mathrm{HD}_0 \\ \mathrm{HD}_0 & \mathrm{HD}_8 & \mathrm{HD}_0 & \cdots & \mathrm{HD}_0 \\ \mathrm{HD}_0 & \mathrm{HD}_0 & \mathrm{HD}_8 & \cdots & \mathrm{HD}_0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathrm{HD}_0 & \mathrm{HD}_0 & \mathrm{HD}_0 & \cdots & \mathrm{HD}_8 \\ \mathrm{HD}_8 & \mathrm{HD}_0 & \mathrm{HD}_0 & \cdots & \mathrm{HD}_0 \\ \mathrm{HD}_8 & \mathrm{HD}_8 & \mathrm{HD}_0 & \cdots & \mathrm{HD}_0 \\ \mathrm{HD}_8 & \mathrm{HD}_0 & \mathrm{HD}_8 & \cdots & \mathrm{HD}_0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathrm{HD}_8 & \mathrm{HD}_0 & \mathrm{HD}_0 & \cdots & \mathrm{HD}_8 \end{bmatrix} \tag{3.5}$$

Similar test cases and their HD arrays $\mathbf{H}_\mathrm{II}^b$ are constructed for all bytes b. The first 17 rows of each $\mathbf{H}_\mathrm{II}^b$ is a reordering of the 17 test cases in $\mathbf{H}_\mathrm{I}$; thus, only $N_{\mathrm{e,II}}^b$ =15 new plaintext-encryption key pairs are needed for each byte in Stage II. Using these test cases, the F-statistic is evaluated as

$$F_B^{b,pc,t/f} = \frac{2N_\mathrm{B} \times \mathrm{Var}(\bar{\bar{x}}_{\mathrm{HD}_0}^{b,pc,t/f}, \bar{\bar{x}}_{\mathrm{HD}_8}^{b,pc,t/f})}{\mathrm{Mean}(\bar{s}_{\mathrm{HD}_0}^{b,pc,t/f}, \bar{s}_{\mathrm{HD}_8}^{b,pc,t/f})} \tag{3.6}$$

Here, the sample means $\bar{\bar{x}}_{\mathrm{HD}_{0/8}}^{b,pc,t/f}$ and variances $\bar{s}_{\mathrm{HD}_{0/8}}^{b,pc,t/f}$ are computed across the N_B samples. The fields for $\bar{\bar{x}}_{\mathrm{HD}_0}^{b,pc,t/f}$ and $\bar{s}_{\mathrm{HD}_0}^{b,pc,t/f}$ ($\bar{\bar{x}}_{\mathrm{HD}_8}^{b,pc,t/f}$ and $\bar{s}_{\mathrm{HD}_8}^{b,pc,t/f}$) are observed using the test cases in rows 1-16 (17-32) of $\mathbf{H}_\mathrm{II}^b$. Extra bars are used above the sample means and variances because the tests are repeated $N_{\mathrm{r,II}}$ times and the probed fields are first averaged

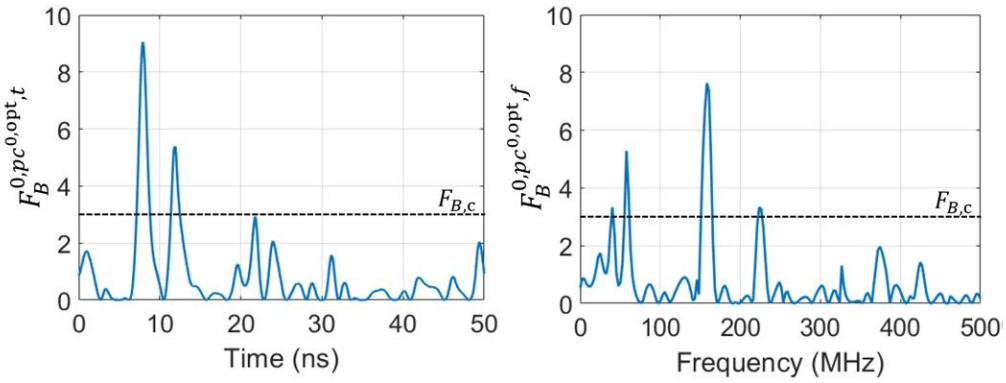


Figure 3.2: Time-domain (left) and frequency-domain (right) $F_N^{0,pc,t/f}$ metric, evaluated with the probe configuration $pc^{0,\mathrm{opt}}$.

over them. The number of repetitions per test case in Stage II can be lower than that in Stage I, i.e., $N_{\mathrm{r,II}} < N_{\mathrm{r,I}}$, in part because configurations most sensitive to measurement noise are discarded in Stage I and in part because the goal is to reduce noise rather than accurately capture variations in repeated measurements.

An example of the F-statistic computed with $N_{\mathrm{r,II}} = 10$ repetitions is shown in Fig. 3.2. Comparing Figs. 2.3, 3.1, and 3.2, it can be observed that both F-statistics must be maximized to successfully disclose the encryption key. Similar to Stage I, the computed F-values are compared to a threshold $F_{B,c}$ to generate a leakage indicator:

$$Indicator_{\mathrm{II}}^{b,pc} = \begin{cases} 1 & \text{if } \max\limits_{t/f} F_B^{b,pc,t/f} \geq F_{B,c} \\ 0 & \text{otherwise} \end{cases} \tag{3.7}$$

Configurations with indicator value 0 are eliminated at the end of Stage II, i.e., only $N_{\mathrm{pc,III}}^{b} = \sum_{pc} Indicator_{\mathrm{II}}^{b,pc}$ probe config-urations are used in Stage III. The thresholds $F_{N,c}$ and $F_{B,c}$ are derived from F-distributions for a 90% confidence level.

3.1.5 Stage III: ANOVA-Informed Correlation Analysis

In Stage III, correlation analysis is performed to identify $pc^{b,\mathrm{opt}}$ by using only the probe configurations not eliminated at the end of Stage II. One potential approach, after

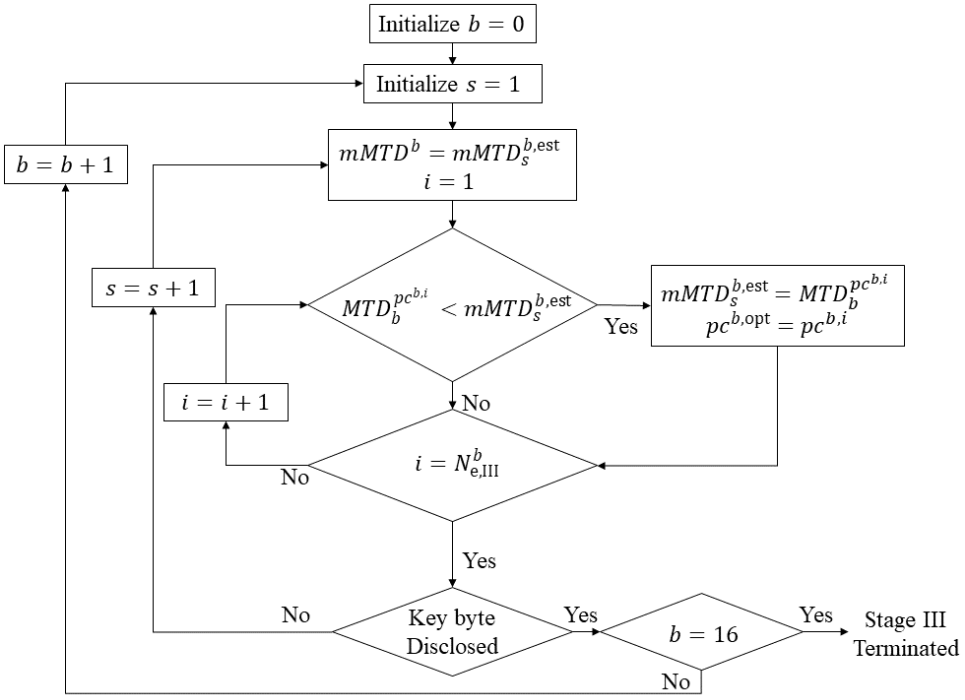


Figure 3.3: Flowchart of Stage III of the proposed protocol.

collecting N_e measurements, is to repeatedly compute the correlation coefficient in Eq. (2.3), starting with N_e encryptions, followed by $N_\text{e} - 1$ encryptions, and so on, until $MTD^{b,pc}$ is identified, i.e., where the coefficient for the correct guess drops below the null hypothesis threshold (Fig. 2.4). This requires $\text{O}(N_\text{e})$ to $\text{O}(N_e^2)$ operations; alternatively, a binary search algorithm can be implemented to identify $MTD^{b,pc}$ in $\text{O}(N_\text{e} \log N_\text{e})$ operations [29]. Stage III ends by identifying $mMTD^b$ and $pc^{b,\text{opt}}$ for each byte b.

A naïve approach to ensure $mMTD^b$ is identified in Stage III is to set $N_\text{e} = N_\text{e}^{\max}$, a large number of encryptions that ensures all key bytes are disclosed. Alternatively, the F-values found in Stage II can be used to inform the search and potentially reduce the measurement costs of Stage III (Fig. 3.3): In this approach, $N_{\text{scan,III}}^b$ scans are performed for each byte b, using all $N_{\text{pc,III}}^b$ probe configurations. Before these scans, the probe configurations are arranged in descending order of their F-values found in Stage II as

$\{pc^{b,1}, pc^{b,2}, \ldots, pc^{b,N^b_{\text{pc,III}}}\}$. In each scan $s = 1, \ldots, N^b_{\text{scan,III}}$, an initial estimate of $mMTD^b$ is chosen as $mMTD_s^{b,\text{est}}$ and $mMTD_s^{b,\text{est}}$ encryptions are observed using each configuration $pc^{b,i}$ for $i = 1, \ldots, N^b_{\text{pc,III}}$. If $MTD_b^{pc^{b,i}} < mMTD_s^{b,\text{est}}$ for any configuration, the remaining configurations are evaluated by reducing $mMTD_s^{b,\text{est}}$ to $MTD_b^{pc^{b,i}}$. The estimate is so updated throughout the scan and this process continues until all $N^b_{\text{pc,III}}$ probe configurations are tested. The scans are terminated if $MTD_b^{pc^{b,\text{i}}} < mMTD_s^{b,\text{est}}$ for any probe configuration and the key was disclosed. Otherwise, $mMTD_s^{b,\text{est}}$ is increased to $mMTD_{s+1}^{b,\text{est}}$ and the process is repeated until the key is disclosed; e.g., in this work, each scan incremented the estimate by 500 encryptions. If the number of encryptions is increased, only the additional $mMTD_{s+1}^{b,\text{est}} - mMTD_s^{b,\text{est}}$ encryptions have to be observed because the observations from the previous scan can be reused when computing the correlation coefficient. In the best-case/ minimum-cost scenario, the first configuration tested in the first scan reveals $pc^{b,\text{opt}}$ and $mMTD^b$, while in the worst-case/ maximum-cost scenario, the final configuration at the end of the final scan reveals the optimal configuration. Therefore, $mMTD^b \leq N^b_{\text{e,III}} \leq mMTD^{b,\text{est}}_{N^b_{\text{scan,III}}}$ encryptions are observed with each probe configuration in Stage III.

3.2 Measurement Costs and Alternative Methods

3.2.1 Acquisition Cost

The acquisition cost of the proposed protocol is the total number of measurements in each stage, which is the product of the number of encryptions observed per configuration, number of repetitions, and number of configurations probed, i.e.,

$$Acquis.Cost = N_{\text{e,I}} N_{\text{r,I}} N_{\text{l}} N_{\text{h}} N_{\text{o}} + \sum_{b=1}^{16} N^b_{\text{e,II}} N_{\text{r,II}} N^b_{\text{pc,II}} + \sum_{b=1}^{16} N^b_{\text{e,III}} N^b_{\text{pc,III}} \text{ (ANOVA)} \quad (3.8)$$

Once the acquisition cost is accrued, the *marginal cost* of future evaluations can be reduced by reusing probe configurations $pc^{b,\mathrm{opt}}$ and performing only the minimum number of measurements $mMTD^b$ for each byte. The marginal cost of future evaluations is [10],[9],

$$Marginal\ Cost = \sum_{b=1}^{16} mMTD^b \tag{3.9}$$

The marginal cost of evaluating a module employing a countermeasure is compared to that of a baseline module to quantify the improvement in the EM side-channel security of hardened AES modules in this paper. In the most resilient modules, some key bytes may potentially not be disclosed [20]. In these cases, to limit the measurement costs of the evaluation, the number of encryptions performed per configuration is restricted to be no more than $N_{\mathrm{e}}^{\max}$.

3.2.2 Acquisition Time and Storage

The storage requirements and acquisition time can be computed based on the acquisition cost and equipment parameters [29]. Each time sample is typically stored as a single-precision floating-point number in the oscilloscope. Therefore, the storage requirements can be estimated as $Acquisition\ Cost^{\mathrm{ANOVA}} \times N_{\mathrm{t}} \times 4$ bytes, where N_{t} is the number of samples in one clock period. Acquisition time is an equipment-dependent quantity; e.g., primitive oscilloscopes generally collect and transfer data to a computer one encryption at a time, which can take more time because of the latency associated with each transaction. On the other hand, higher-end oscilloscopes can store/transfer several N_{seg} measurements, avoiding latency-related issues associated with the acquisition [29]. Further, such oscilloscopes have sufficient processing capabilities to perform analysis, without needing to transfer data to another computer. As a result, multi-stage protocols, where measurements in each stage are decided by results of the previous stage, can be

potentially sped up [29]. If t_{cap} seconds is required to capture each sample, and t_{sav} seconds is required to save each sample, then

$$\begin{aligned} Acq.Time_{\mathrm{I}}^{\mathrm{ANOVA}} &= N_{\mathrm{t}}\left(t_{\mathrm{cap}}+t_{\mathrm{sav}}\right)\times\left\lceil\frac{N_{\mathrm{e,I}}N_{\mathrm{r,I}}N_{\mathrm{l}}N_{\mathrm{h}}N_{\mathrm{o}}}{N_{\mathrm{seg}}}\right\rceil \\ Acq.Time_{\mathrm{II}}^{\mathrm{ANOVA}} &= N_{\mathrm{t}}\left(t_{\mathrm{cap}}+t_{\mathrm{sav}}\right)\times\ \sum_{b=1}^{16}\left\lceil\frac{N_{\mathrm{e,II}}^{b}N_{\mathrm{r,II}}N_{\mathrm{pc,II}}^{b}}{N_{\mathrm{seg}}}\right\rceil \\ Acq.Time_{\mathrm{III}}^{\mathrm{ANOVA}} &= N_{\mathrm{t}}\left(t_{\mathrm{cap}}+t_{\mathrm{sav}}\right)\times\ \sum_{b=1}^{16}\left\lceil\frac{N_{\mathrm{e,III}}^{b}N_{\mathrm{pc,III}}^{b}}{N_{\mathrm{seg}}}\right\rceil \end{aligned} \tag{3.10}$$

seconds are required to perform experiments in each stage of the protocol. In addition to acquisition time, evaluators may also need to account for the speed of probe positioning [29] and analysis time in each stage. Frequency-domain fields are generated during analysis, using the FFT algorithm, which requires a processing time of $\mathrm{O}(N_{\mathrm{t}}\log N_{\mathrm{t}})$ seconds for each encryption. While time-domain fields are limited by the clock period, the frequency-domain fields can be limited to N_{BW} samples, corresponding to the limiting bandwidth of all equipment [20], to reduce analysis time. At each configuration and time/frequency sample, the analysis time needed to compute F-values in Eq. (2.8) is $\mathrm{O}(N_{\mathrm{B}}N_{\mathrm{T}})$ seconds. The MTD identification in stage III requires $\mathrm{O}(N_{\mathrm{e,III}}^{b}\log N_{\mathrm{e,III}}^{b})$ seconds per sample.

3.2.3 Alternative Methods

The proposed protocol is compared to several alternatives in Section 3.4. The exhaustive search method (Section 2.1.2) [10], [29] is one potential alternative. It performs correlation analysis by observing $N_{\mathrm{e}}^{\max}$ encryptions across the entire search space of probe configurations in a single, high-resolution scan. As a result, the exhaustive approach requires [10], [29]

$$Acquis.Cost = N_{\mathrm{e}}^{\max}N_{\mathrm{l}}N_{\mathrm{h}}N_{\mathrm{o}}\ \text{(exhaustive search)} \tag{3.11}$$

measurements to be observed. A more viable correlation-analysis approach is the greedy-search adaptive scan protocol implemented in [10], [20], [29] and briefly described in the black-box attack in Section 2.2.3. This greedy-search protocol requires [10],

$$Acq.Cost^{\text{Bbox}} = Acq.Cost^{\text{pre}} + \sum_{s=1}^{N_{\text{scan,I}}} \sum_{h=1}^{N_{\text{h}}} \sum_{o=1}^{N_{\text{o}}} N_{\text{e},s,\text{I}} N_{\text{l},h,o,s,\text{I}} + \sum_{b=1}^{16} \sum_{s=1}^{N_{\text{scan,II}}} mMTD_{s-1}^{b} N_{\text{l},s,\text{II}} \quad \text{(Greedy Search)} \tag{3.12}$$

measurements. Note that this approach can have unlimited cost, e.g., for hardened modules, if the number of scans is not bounded. In practice, the acquisition cost of this protocol should be bounded by that of the exhaustive search method in Eq. (3.11) by limiting its phase I to at most $N_{\text{e}}^{\max}$ encryptions and its phase II to have at most the same resolution as the exhaustive search.

Another alternative is the TVLA method, a commonly used leakage indicator, including in the ISO/IEC 17825 standard [34],[35], to evaluate the side-channel resilience of crypto-systems [11],[12],[25]. The TVLA method also statistic-ally characterizes the probed fields for specially constructed test cases. Here, the DUT is assumed to be a "white box" [9], where evaluators can control the inputs to the chip but not the encryption key. It uses Welch's t-test to compare the means of two sets of observed fields—a reference set (SetA), where inputs are fixed, and a test set (SetB), where the inputs are randomly generated—hypothesizing that information leakage is present if there are significant changes in the means of the two sets. In SetA, one plaintext is repeated N_{SetA} times for a fixed key; in SetB, N_{SetB} randomly generated inputs are encrypted using the same key as SetA. Computing the sample means $\bar{x}_{\text{SetA/SetB}}^{pc,t/f}$ and variances $s_{\text{SetA/SetB}}^{pc,t/f}$ across the $N_{\text{SetA}}/N_{\text{SetB}}$ samples, the Welch t-test is evaluated as:

$$T^{pc,t/f} = \frac{\bar{x}_{\text{SetB}}^{pc,t/f} - \bar{x}_{\text{SetA}}^{pc,t/f}}{\sqrt{s_{\text{SetB}}^{pc,t/f}/N_{\text{SetB}} + s_{\text{SetA}}^{pc,t/f}/N_{\text{SetA}}}} \tag{3.13}$$

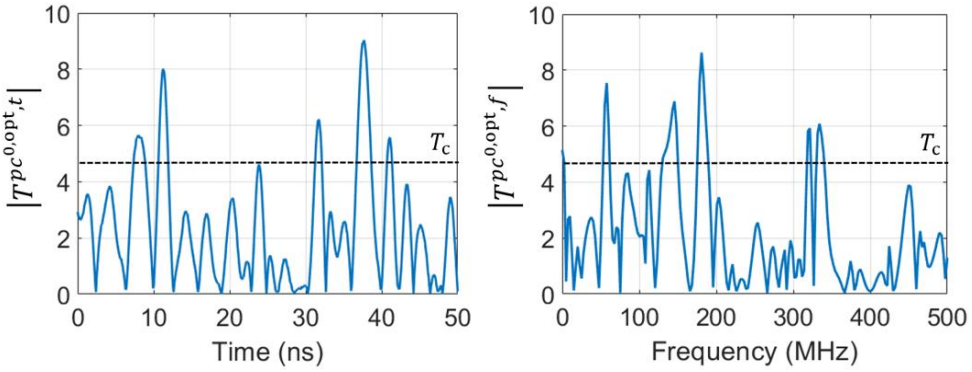


Figure 3.4: Time-domain (left) and frequency-domain (right) TVLA metric, evaluated with the probe configuration $pc^{0,\mathrm{opt}}$.

Using the parameters in [11], an example TVLA metric computed for 200 fixed plaintext and 200 random plaintext is shown in Fig. 3.4. In addition to accurately indicating leakages at ~10 ns/~200 MHz (Fig. 2.3), the TVLA also shows exaggerated leakage at ~40 ns/~300 MHz. Because test cases are randomized without any restrictions, the TVLA method is leakage-model independent and can be used as a generic approach to analyze side-channel leakage; in contrast, the ANOVA approach described in this work constructs test cases based on the leakage model used in the correlation analysis. The results of TVLA are not necessarily linked, however, to the number of measurements needed to disclose the key [11], [23], [35]. Furthermore, results of low-cost TVLA experiments using fewer encryptions ($N_{\mathrm{SetA}}, N_{\mathrm{SetB}} \approx$ 200-500) may have limited accuracy [11]. Increasing the number of encryptions (N_{SetA}, $N_{\mathrm{SetB}} \approx$ 20000) to improve accuracy [12] is infeasible for fine-grained EM SCA attacks as the acquisition cost would approach the exhaustive search. The computed T-statistic can be compared with a threshold T_c to generate another indicator:

$$Indicator_{\mathrm{TVLA}}^{pc} = \begin{cases} 1 & \text{if } \max\limits_{t/f} T^{pc,t/f} \geq T_c \\ 0 & \text{otherwise} \end{cases} \tag{3.14}$$

Once probe configurations with 0 TVLA indicator values are eliminated, correlation analysis is performed only with $N_{\mathrm{pc,TVLA}} = \sum_{pc} Indicator_{\mathrm{TVLA}}^{pc}$ configurations. An exhaustive search (TVLA+e) would observe $N_{\mathrm{e}}^{\max}$ encryptions at each configuration. Alternatively, a TVLA-informed search (TVLA+i) similar to that in Section 3.1.5 can be used to reduce the measurement costs. In this approach, $N_{\mathrm{scan,TVLA}}^{b}$ scans are performed for each byte b and $mMTD^{b} \leq N_{\mathrm{e,TVLA}}^{b}$ encryptions are observed with each probe configuration. The acquisition cost of these two protocols are

$$\begin{aligned} Acquis.\ Cost &= (N_{\mathrm{SetA}} + N_{\mathrm{SetB}}) N_{\mathrm{l}} N_{\mathrm{h}} N_{\mathrm{o}} + \\ &\quad N_{\mathrm{e}}^{\max} N_{\mathrm{pc,TVLA}}\ (\mathrm{TVLA + e}) \\ Acquis.\ Cost &= (N_{\mathrm{SetA}} + N_{\mathrm{SetB}}) N_{\mathrm{l}} N_{\mathrm{h}} N_{\mathrm{o}} + \\ &\quad \sum_{b=1}^{16} N_{\mathrm{e,TVLA}}^{b} N_{\mathrm{pc,TVLA}}\ (TVLA + i) \end{aligned} \tag{3.15}$$

3.3 DEVICES UNDER TEST

This section describes the 9 AES implementations (2 baseline and 7 hardened ones) whose vulnerability to EM SCA attacks is evaluated with the proposed protocol. The countermeasures in these implementations are separated into three categories representing different strategies to secure the chip. The countermeasures tested in this thesis are based on existing implementations in [18]-[20], [36],[37].

3.3.1 Baseline AES implementations

The first baseline AES module was implemented on an Artix-7 FPGA with 20 mm × 20 mm chip size tested on the CW305 evaluation board [33]. The evaluation board, which was specifically designed to demonstrate SCA attacks, allowed the clock frequency and supply voltage to be changed. As a baseline scenario, the chip was operated at clock frequency of $f^{\mathrm{clk}} = 20$ MHz and supply voltage of $V^{\mathrm{s}} = 1$ V. This baseline implementation is used as a reference to test 3 repeatability countermeasures (Section

3.3.2), 1 algorithmic countermeasure (Section 3.3.3), and 1 physical design strategy (Section 3.3.4), all implemented on the same FPGA.

The second baseline AES module was an ASIC with 10 mm × 10 mm chip size [36]. The chip was operated at input clock frequency $f^{\text{clk}} = 37.5$ MHz and supply voltage $V^{\text{s}} = 1.1$ V. It is used as a reference for testing 2 physical design strategies implemented on the same chip.

3.3.2 AES Implementations with Repeatability Countermeasures

Observed fields depend on the DUT's operating supply voltage and clock frequency. Randomly scaling these parameters can create temporal shifts and modify amplitudes in observed signals, reducing the repeatability of experiments and increasing measurement noise. Three such countermeasures based on EM interference reduction techniques [20] are tested in this paper:

1) Frequency Scaling (FS): Randomizing clock frequency creates delays in the circuit and misaligns measurements over multiple encryptions. While this jitter dithers time-domain signals [14], frequency-domain EM SCA attacks remain effective against this countermeasure. The FS countermeasure was implemented by varying the clock frequency in the range $f^{\text{clk}} = 20$ MHz $\pm$ 0.25 MHz.

2) Voltage Scaling (VS): Voltage scaling desensitizes peak-to-peak amplitudes of observed fields to the data being encrypted [15]. This countermeasure obfuscates both time- and frequency-domain fields. The VS countermeasure was implemented by varying the input supply in the range $V^{\text{s}} = 1$ V $\pm$ 0.05 V.

3) Voltage-Frequency Scaling (VFS): This countermeasure combines the VS and FS countermeasures to provide maximum dithering of fields in both time- and frequency-domain [16]. The VFS countermeasure was implemented by simultaneously varying the

input supply and clock frequency in the ranges selected in the VS and FS countermeasure (set 2 in [20]).

These countermeasures were implemented on the FPGA, using the programmable clock and voltage supply, such that 5 fixed states of voltage, frequency, or voltage-frequency pairs were chosen within the selected ranges. These countermeasures can be implemented with relatively low overhead [15], [16].

3.3.3 AES Implementations with Algorithmic Countermeasures

Countermeasures artificially introducing algorithmic noise typically introduce additional operations/modify data flow in the algorithm. Examples include hiding and masking [19], [36], [37], where exploitable intermediate round outputs are modified to break correlation with observed fields. A majority of countermeasures in this category for AES focus on masking non-linear Sbox operations using novel transformations or changes to existing implementations; e.g., in [19], a byte permutation (BP) network that rearranges bytes randomly was proposed as a precursor to Sbox operations and AES correctness was maintained by using an inverse BP network to re-order bytes at the end of each round. This method showed limited resilience improvement (~3.2×) for a black-box threat model [19]. Therefore, in addition to the hardened Sbox implementation in [19], a simple Boolean XOR operation for linear operation masking [36], [37] is used to hide the intermediate state register value in this paper. Here, a random "mask" variable $\mathbf{M}_m$ changes $\mathbf{oc}_e^9$ to masked value $\mathbf{oc}_{e,m}^9$ in the penultimate round. The last round begins with "unmasking" and byte-order randomization using the BP network. After Sbox operation, bytes are re-ordered with the inverse BP network, followed by the shift rows operation. The final operations of AES can be summarized as,

$$oc_{e,m}^{9,b'} = oc_e^{9,b'} \oplus M_m^{b'}$$
$$i\tilde{v}_e^{10,b} = ShiftRows\left(BP^{-1}\left[Sbox\left(BP\left[oc_{e,m}^{9,b} \oplus M_m^b\right]\right)\right]\right) \quad (3.16)$$
$$oc_{e,m}^{10,b} = i\tilde{v}_e^{10,b} \oplus k^{10,b}$$

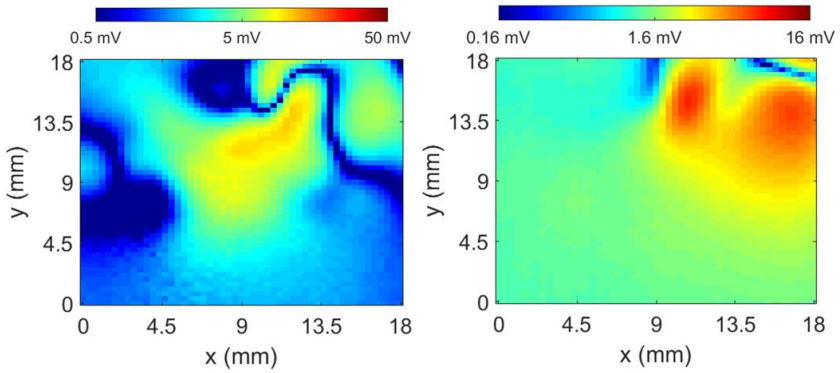
This countermeasure was implemented on the FPGA using the nominal clock frequency and input supply. While it can be an effective countermeasure, masking incurs significant area and delay overheads [19]; moreover, it can be vulnerable to higher-order attacks [36], [37] outside the scope of this paper, where the mask is attacked first, followed by the key.

3.3.4 AES Implementations with Physical Design Strategies

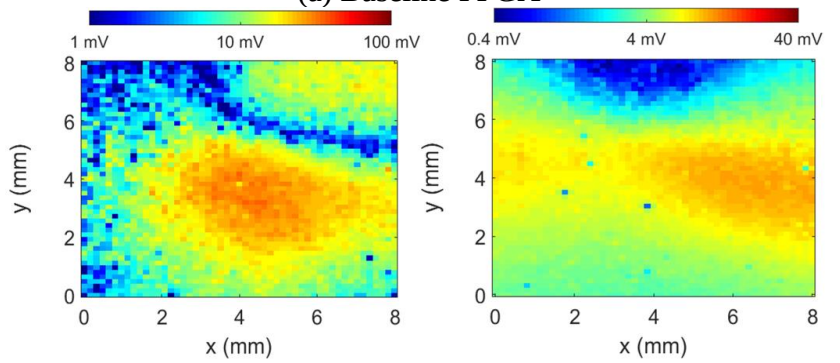
These countermeasures minimize data-dependent variations in observed fields by implementing dedicated signal attenuation hardware [13], modifying the chip's physical design [18], or shielding the module [17]. These may not be effective at all frequencies of interest, can increase packaging costs, or increase the area overhead. In this paper, 3 such countermeasures are implemented: In the first one, a 25-μm thick aluminum foil is placed over the FPGA to attenuate fields and degrade EM SCA attacks. The other two countermeasures are implemented on the ASIC and involve changes to the AES module's power grid. The first design implements a "twisted pair" grid structure [18]; the second one uses wider and thicker power rails to shield signals from lower metal layers [18].

3.4 Baseline Results

This section presents the results for the baseline FPGA and ASIC implementations of AES-128. The proposed method is compared to alternatives in terms of acquisition costs. All spatial maps of fields and computed statistics in this section were obtained with the *x*-oriented probe. The setup for the FPGA was already described in Section 2.3.1. The ASIC used an additional Arduino interface, which acts as an intermediary during the transfer of



(a) Baseline FPGA



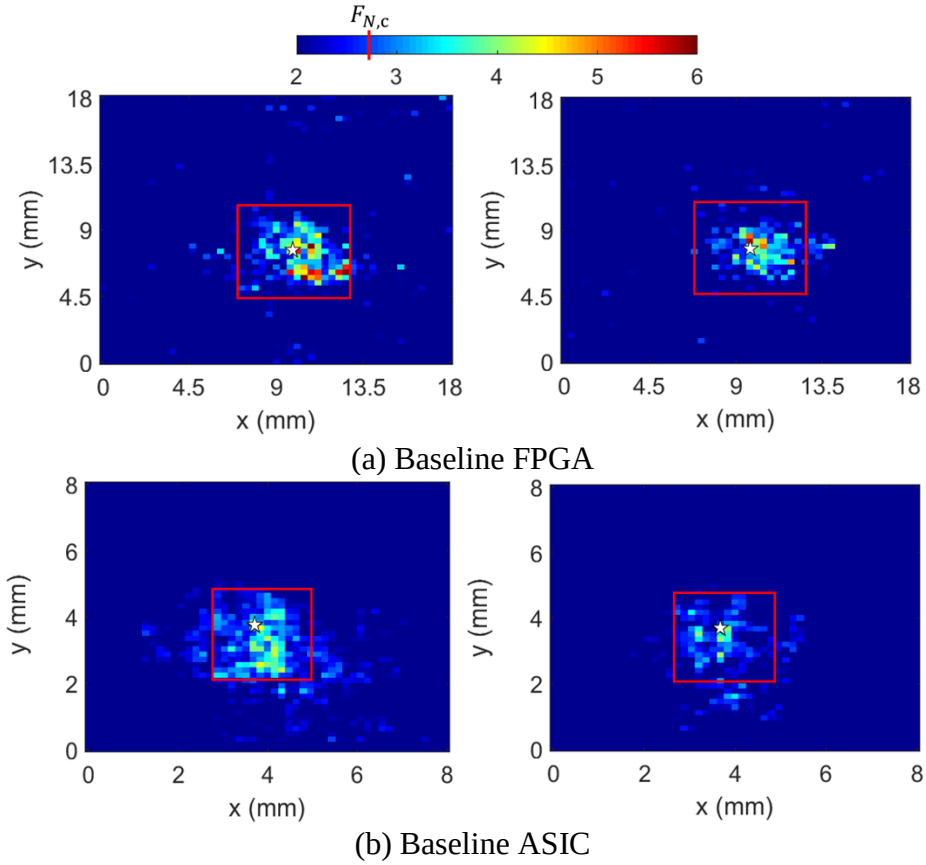
(b) Baseline ASIC

Figure 3.5: Spatial map of (a) time-domain signals at ~8 ns (left) and frequency-domain signals at ~160 MHz (right) for the FPGA module detailed in [10], and (b) time-domain signals at ~6 ns (left) and frequency-domain signals at ~100 MHz (right) for the ASIC module detailed in [18].

plaintext and keys from the main computer. To demonstrate the spatial resolution, maps of time- and frequency-domain fields at information leaking time/ frequency samples are plotted in Fig. 3.5 for the two DUTs, averaged over 30 repeated measurements. These composite images are obtained one pixel/measurement at a time by re-positioning the probe and repeating the encryption.

3.4.1 Proposed Protocol Results

The Stage I $F_N^{b,pc,t/f}$ metric was computed by repeating the $N_{\mathrm{e,I}} = 17$ encryptions detailed in Section 3.1.3 $N_{\mathrm{r,I}} = 30$ times. Spatial maps of the maximum $F_N^{0,pc,t/f}$ are plotted



(a) Baseline FPGA

(b) Baseline ASIC

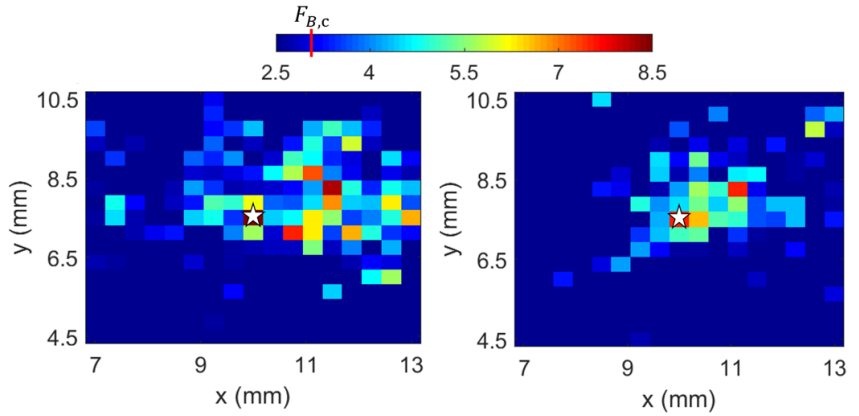
Figure 3.6: Spatial map of (a) time-domain signals at ~8 ns (left) and frequency-domain signals at ~160 MHz (right) for the FPGA module detailed in [10], and (b) time-domain signals at ~6 ns (left) and frequency-domain signals at ~100 MHz (right) for the ASIC module detailed in [18].

in Fig. 3.6; a large portion of the configurations with high F-values were located inside the areas marked with red boxes. Fig. 3.6 shows that frequency-domain analysis discarded more configurations in Stage I. The Stage II $F_B^{b,pc,t/f}$ metrics were computed by repeating the $N_{\mathrm{e,II}}^{b} = 15$ encryptions detailed in Section 3.1.4 $N_{\mathrm{r,II}} = 10$ times and averaging the signals. Spatial maps of the maximum $F_B^{0,pc,t/f}$ are shown in Fig. 3.7 only for the areas marked with red boxes in Fig. 3.6 for simplicity (high F-ratio configurations outside the red boxes were also evaluated in Stage II). Then, configurations whose maximum $F_B^{0,pc,t}$ were larger than $F_{B,c}$ were tested in Stage III to find the optimal probe

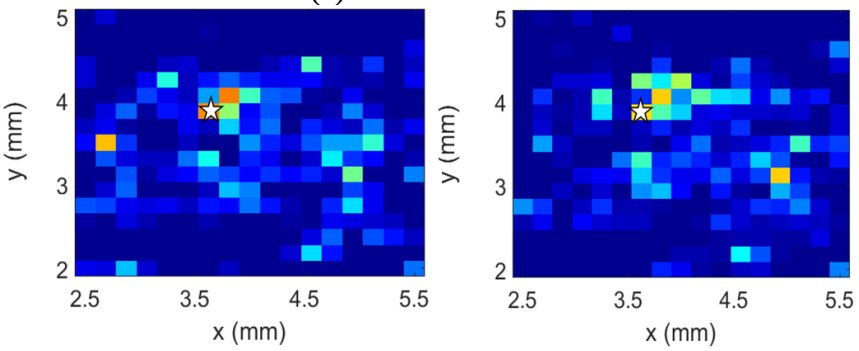
Acquisition Cost	Baseline FPGA		
	Time Domain	Frequency Domain	
Stage I ($\times 10^6$)	2.65	2.65	Stage I ($\times 10^6$)
Stage II ($\times 10^6$)	1.62	1.18	Stage II ($\times 10^6$)
Stage III ($\times 10^6$)	1.24	1.16	Stage III ($\times 10^6$)

Table 3.1: Proposed ANOVA Method's Costs

configurations, using at most $N_{\text{scan,III}}^{b} = 2/3\ (6/8)$ scans in time/frequency domain for the baseline FPGA (ASIC). Each scan incremented the estimate $mMTD_s^{b,\text{est}}$ by 500. The

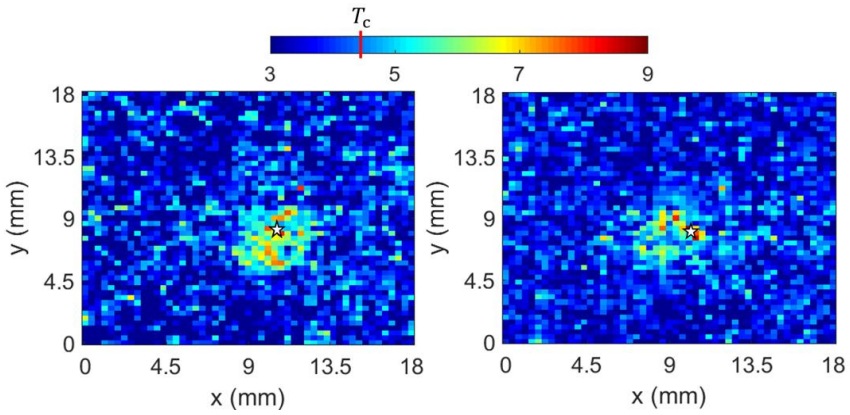


(a) Baseline FPGA

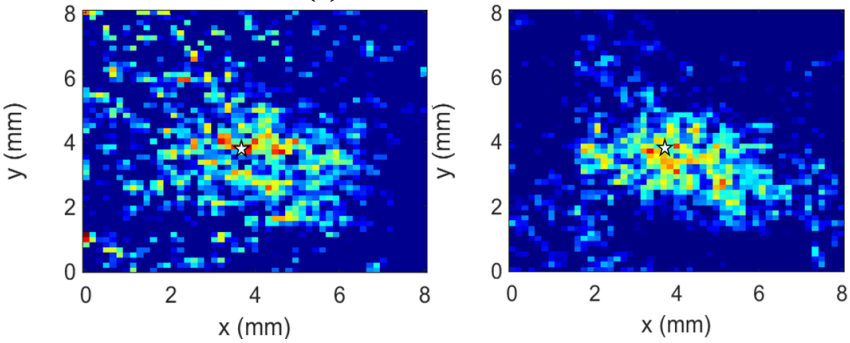


(b) Baseline ASIC

Figure 3.7: Spatial map of (a) time-domain signals at ~8 ns (left) and frequency-domain signals at ~160 MHz (right) for the FPGA module detailed in [10], and (b) time-domain signals at ~6 ns (left) and frequency-domain signals at ~100 MHz (right) for the ASIC module detailed in [18].



(a) Baseline FPGA



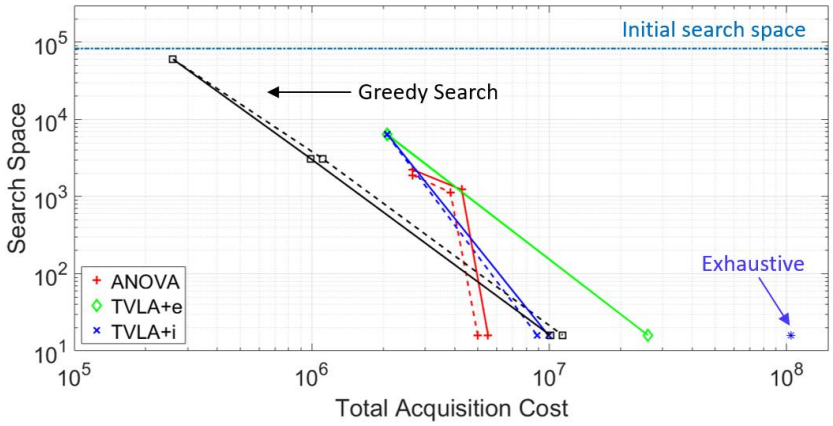
(b) Baseline ASIC

Figure 3.8: Spatial map of $\max_t T^{pc,t}$(left) and $\max_f T^{pc,f}$ (right) for the baseline (a) FPGA [10] and (b) ASIC [18]. Optimal configurations are shown with stars.

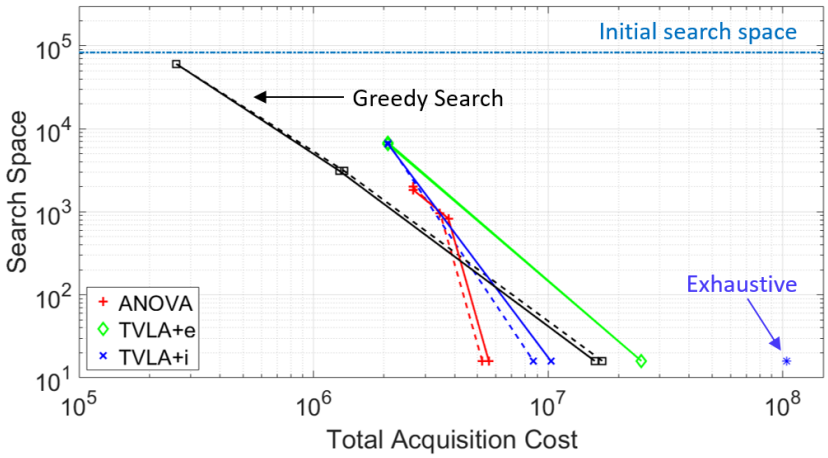
acquisition costs of the protocol are listed in Table 3.1. The table shows that the time-domain evaluation required ~1.2× (~1.1×) more measurements than the frequency-domain one for the FPGA (ASIC).

3.4.2 Cost Comparison to Alternative Methods

Let's first compare the proposed method for evaluating EM SCA vulnerability to emulating correlation-analysis attacks. Using an exhaustive scan, where $N_e^{\max} = 20000$ encryptions are observed with every probe configuration in the search space, correlation analysis would require ~10^8 measurements. The acquisition cost can be lowered with



(a) Baseline FPGA



(b) Baseline ASIC

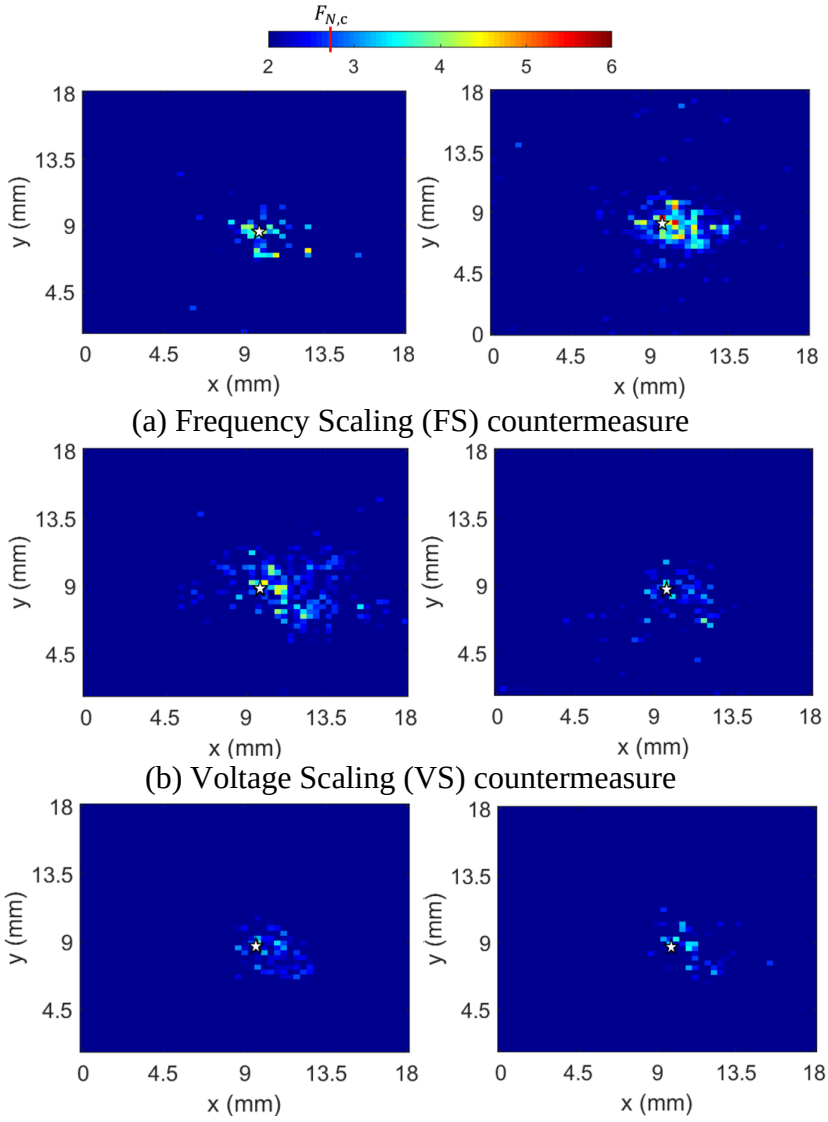
Figure 3.9: Reduction of search space for the optimal probe configuration in time (solid) and frequency domain (dashed) for the baseline (a) FPGA [10] and (b) ASIC [18]. Unlike the exhaustive- and greedy-search protocols, which emulate correlation analysis by actual attackers with restricted access, the TVLA and ANOVA protocols accelerate the process by computing statistical metrics.

adaptive scan protocols. Here, the greedy search protocol [10] was implemented with a pre-characterization stage: Every probe configuration was used to observe fields for $N_{\mathrm{e}}^{\mathrm{pre}} = 50$ random encryptions and configurations where the standard deviation was < 0.1 mV were removed from the search space. In Phase I, $N_{\mathrm{scan,I}} = 2$ (3) scans were performed for the FPGA (ASIC) and optimal configurations were identified by using $N_{\mathrm{e,2,I}} = 5000$

($N_{\mathrm{e,2,I}} = 5000$ and $N_{\mathrm{e,3,I}} = 8000$) encryptions; in phase II, $N_{\mathrm{scan,II}} = 2$ (2) scans were performed for each byte. The final costs of implementing the protocol on the baseline FPGA (ASIC) were found to be ~1.0/1.1×10^7 (~1.6/1.7×10^7) measurements in time/frequency domain. Therefore, the proposed protocol was observed to be ~17-22× cheaper than the exhaustive approach and ~2-3× cheaper than the adaptive acquisition approach for the baseline cases.

Next, let's compare the proposed ANOVA-based method to TVLA-based alternatives. Here, TVLA was implemented using $N_{\mathrm{SetA}} = N_{\mathrm{SetB}} = 200$ encryptions for both baseline implementations. Spatial maps of the maximum $T^{pc,t/f}$ are shown in Fig. 3.8. Numerous "false positives" are observed throughout the search space, especially for the FPGA. Using the TVLA+e protocol on the FPGA (ASIC) required ~2.6/2.8×10^7 (~2.5/2.4×10^7) measurements in time/frequency domain. The TVLA+i protocol required $N^{b}_{\mathrm{scan,TVLA}} = 2/3$ $(6/8)$ scans and ~9.9/10.2×10^6 (~8.8/8.6×10^6) measurements in time/frequency domain. Therefore, the proposed protocol was observed to be ~4-5× cheaper than the TVLA+e and ~1.5-2× cheaper than the TVLA+i method for the baseline cases.

All protocols identified similar information-leaking configurations and minimum MTDs, although each protocol required different acquisition cost to reach the final result. All protocols began with the same maximum search space ($N_{\mathrm{o}} \times N_{\mathrm{l}} = 2 \times 51 \times 51$ configurations), at 0 acquisition cost, and ended with 16 optimal configurations (one for each byte) after accruing the acquisition cost of the measurements. The costs of the protocols are plotted in Fig. 3.9, along with the reduction of the search space at each stage/phase. The search space size at the end of each stage is the sum of remaining possible probe configurations identified for each byte. Fig. 3.9 shows that the methods' performances were rather insensitive to whether time- or frequency-domain signals were



(a) Frequency Scaling (FS) countermeasure

(b) Voltage Scaling (VS) countermeasure

(c) Voltage-Frequency Scaling (VFS) countermeasure

Figure 3.10: Spatial map of $\max_t F_N^{0,pc,t}$ (left) and $\max_f F_N^{0,pc,f}$ (right) for the FPGA implementing three countermeasures that increase the measurement noise.

used and that the proposed protocol outperformed the alternatives for the baseline implementations. Whether the same observations apply to hardened implementations is presented next.

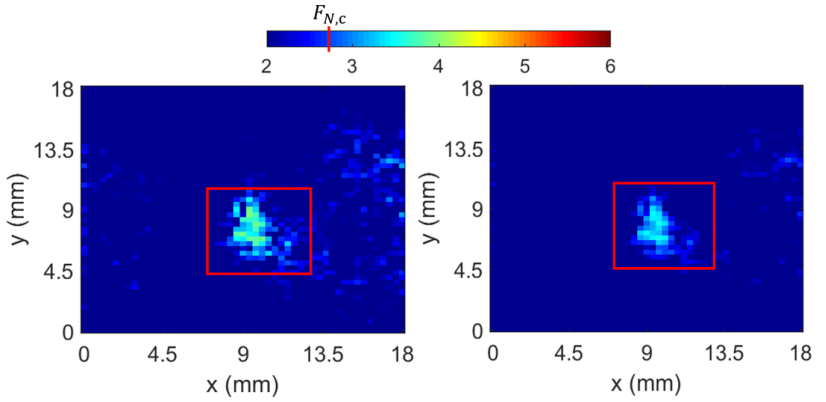
3.5 RESULTS FOR COUNTERMEASURES

This section details the results of evaluations of AES implementations hardened by the countermeasures described in Section 3.3 and the measurement setup detailed in Section 2.3.1. For each class of countermeasures, spatial maps of $F_N^{b,pc,t/f}$ and/or $F_B^{b,pc,t/f}$ are shown in Sections 3.5.1-3. Section 3.5.4 presents the costs of evaluating the counter-measures along with the improvement in resilience. For countermeasures with $mMTD^b > N_e^{\max} = 20000$, the cost of the greedy-search protocol is replaced by the cost of the exhaustive scan.

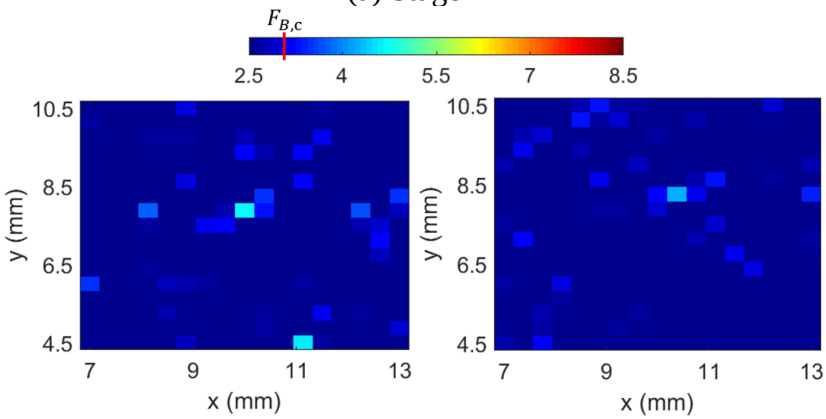
3.5.1 Countermeasures Increasing Measurement Noise

The countermeasures FS, VS, and VFS detailed in Section 3.3.2 increase the measurement noise in signals. Because they increase variance within repeated measurements, these counter-measures should degrade $F_N^{b,pc,t/f}$. Spatial maps of the maximum $F_N^{0,pc,t/f}$ are plotted in Fig. 3.10 for the 3 hardened implementations. The results can be compared to those for the baseline FPGA in Fig. 3.6; the optimal probe configurations were found to be the same in all cases.

The FS countermeasure could improve the resilience of the module against time-domain EM SCA attacks but had negligible impact on frequency-domain ones. Although shifts in time domain should not impact the magnitude of signals in frequency domain, delaying/hastening the signal still caused some minor variations in the frequency-domain EM SCA attack; this is because measurements were time-gated to the nominal clock period [20]. The VS countermeasure could improve the resilience of the module against both time- and frequency-domain EM SCA attacks, although the impact was more apparent in the frequency-domain approach. Voltage scaling affects the fields disproportionately in time domain, in particular, more variance was observed around signal peaks, while at other time intervals signals were more repeatable [20].



(a) Stage I



(b) Stage II

Figure 3.11: Spatial map of $\max_t F_N^{0,pc,t}$ (top-left), $\max_f F_N^{0,pc,f}$ (top-right), $\max_t F_B^{0,pc,t}$ (bottom-left), and $\max_f F_B^{0,pc,f}$ (bottom-right) for the FPGA implementing the masking countermeasure that increases algorithmic noise.

The VFS countermeasure could improve the resilience of the module against both time- and frequency-domain EM SCA attacks. Because this countermeasure combines the previous two countermeasures, the first two stages of the proposed ANOVA method could identify only a few promising configurations with either time- or frequency-domain signals. The proposed method required ~7.2/5.5×10^6, ~6/5.7×10^6, and ~6.9/7×10^6 measurements to identify the optimal probe configurations for the FPGA hardened with the FS, VS, and VFS countermeasure in time/frequency domain.

3.5.2 Countermeasures Increasing Algorithmic Noise

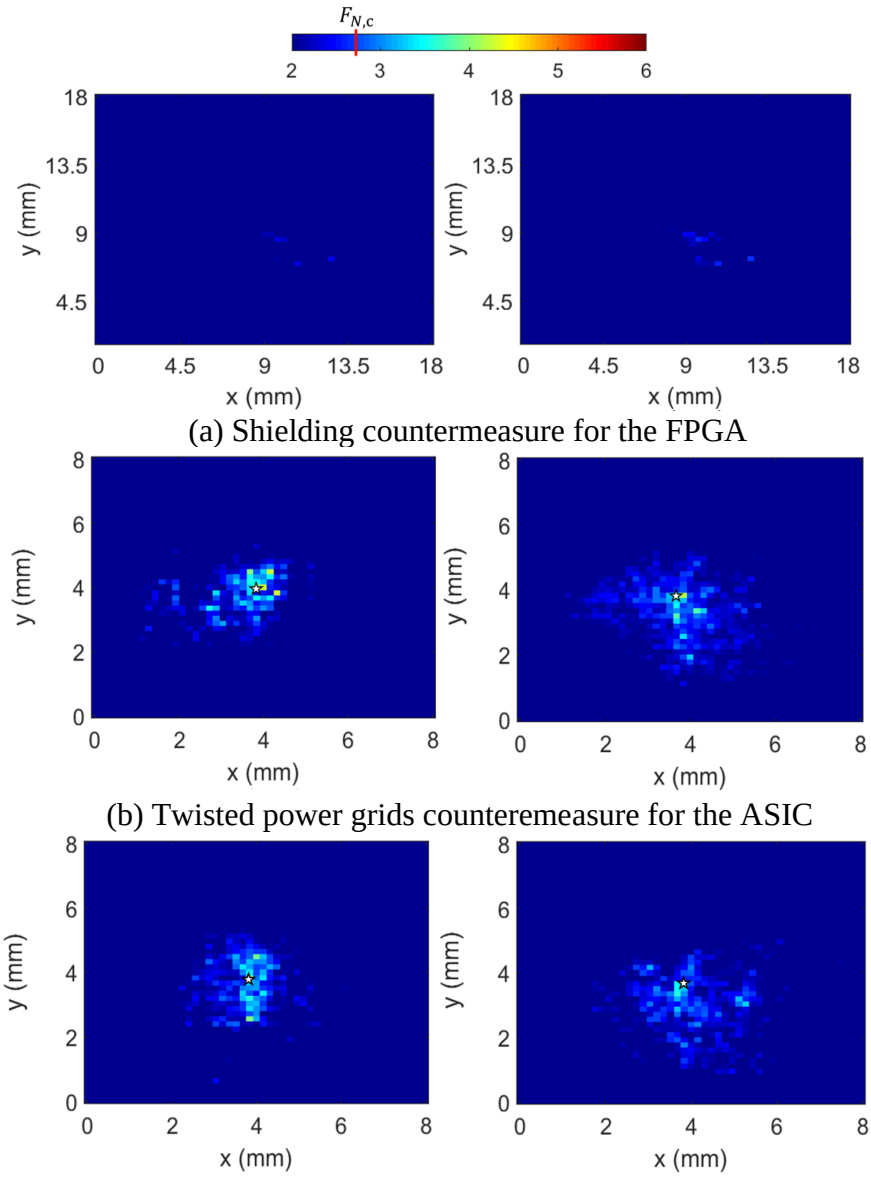
The masking countermeasure detailed in Section 3.3.3 increases the algorithmic noise. Because it performs additional uncorrelated computations, this countermeasure should primarily degrade $F_B^{b,pc,t/f}$. Spatial maps of the maximum $F_N^{0,pc,t/f}$ and $F_B^{0,pc,t/f}$ are plotted in Fig. 3.11. Comparing the results to that for the baseline FPGA in Figs. 3.6-3.7 shows that more configurations were eliminated compared to the baseline at the end of Stage I in addition to Stage II, because randomly masking the state register increases signal variance for repeated encryptions as well as increasing algorithmic noise from uncorrelated computations. More importantly, it was observed at the end of Stage III that none of the probe configurations could disclose *any* key byte after $N_{\mathrm{e}}^{\max}$ encryptions.

Unlike the adaptive scan protocols, which would potentially need the same number of measurements as an exhaustive scan (~10^8) to reach this conclusion, the proposed ANOVA method required only ~8.3/7.6×10^6 measurements in time/frequency domain.

3.5.3 Countermeasures Attenuating Target Signals

The physical design strategies detailed in Section 3.3.4 attenuate the target signals. Because they also reduce the variance of the target signals, these countermeasures should degrade both $F_N^{b,pc,t/f}$ and $F_B^{b,pc,t/f}$. Spatial maps of the maximum $F_N^{0,pc,t/f}$ are plotted in Fig. 3.12, and can be compared with baseline results in Fig. 3.6.

The shielded FPGA revealed no configurations of interest at the end of Stage I, failing to disclose the AES key; this is to be expected as the shield is 3-4 skin depths thick at the information leaking frequencies. While the physical design strategies in [18] revealed few configurations of interest, these configurations were successful in recovering the key, providing limited improvement in resilience. The dense wider power-grid structure revealed marginally fewer configurations compared to the twisted power-grid countermeasure.



(a) Shielding countermeasure for the FPGA

(b) Twisted power grids counteremeasure for the ASIC

(c) Wider power grids countermeasure for the ASIC

Figure 3.12: Spatial map of $\max_t F_N^{0,pc,t}$ (left) and $\max_f F_N^{0,pc,f}$ (right) for the three countermeasures attenuating target signals. Optimal configurations, if present, are shown with stars.

frequency-domain analysis to evaluate the shielding countermeasure. The evaluation of the twisted power-grid structure, the time-/frequency-domain analysis required acquisition cost of ~6.6/7.3×10^6 measurements. The evaluation of the dense wider power grid structure in time/frequency domain required acquisition cost of ~8/8.1×10^6 measurements.

3.5.4 Marginal and Acquisition Cost Comparison

Next, the effectiviness of the countermeasures are evaluated and the costs of the proposed ANOVA method are compared to those of the alternatives when countermeasures are present.

For the baseline FPGA (ASIC), using the optimal configurations identified in Section 3.4, the marginal cost of disclosing keys was only ~1.1/1.4×10^4 (~3.5/4.4×10^4) measurements in time/frequency domain, i.e., disclosing the AES key required ~3× more measurements for the ASIC.

When the FPGA was hardened with the FS, VS, and VFS countermeasures, using the optimal configurations identified in Section 3.4.1, the attackers could disclose the AES key with ~6.9/1.5×10^4, ~0.5/1.2×10^5, and ~1.5/1.7×10^5 measurements in time/frequency domain, respectively. Comparing these marginal costs to those of the baseline FPGA shows that these countermeasures improve the module's resilience to EM SCA attack significantly. These are easy to implement counter-measures that require relatively small design overhead.

When the FPGA was hardened with masking or shielding, because no key bytes could be disclosed by observing $N_{\mathrm{e}}^{\max}$ encryptions, the marginal cost of disclosing the key was $> 16N_{\mathrm{e}}^{\max}$, i.e., >3.2×$10^5$ measurements; thus, these counter-measures improve the module's resilience to EM SCA attack by >30/24× in time/frequency domain. Masking considerably improves the security of the chip at the cost of larger area and delay overheads [19]. While

DUT	Improve-ment over Baseline for TD/FD Attack	Most Effec-tive Attack	Acquisition Cost of Alternatives vs. ANOVA for Most Effective Attack		
			Adaptive Scan	TVLA+e	TVLA+i
FPGA Baseline	1×/ 1×	TD	1.8×	4.8×	1.8×
ASIC Baseline	1×/ 1×	TD	2.7×	4.7×	1.6×
FPGA with FS	6.3×/ 1.1×	FD	2.3×	4.8×	1.8×
FPGA with VS	4.7×/ 8.6×	TD	3.0×	6.9×	3.6×
FPGA with VFS	13.6×/ 12.1×	TD	4.3×	6.0×	4.6×
FPGA with Masking*	>30×/ >24×	-	13.1×	5.3×	5.3×
FPGA with Shielding*	>30×/ >24×	-	37.0×	3.6×	3.6×
ASIC with Twisted Power Grid	1.5×/ 1.4×	TD	1.9×	7.7×	4.3×
ASIC with Wider Power Grid	2.4×/ 2.1×	TD	1.7×	7.1×	4.2×

Table 3.2: Effectiveness of Countermeasures and the Cost of Evaluation

a very simplistic shield was used here, practical use of shielding can incur large packaging costs [17]. Additionally, incorrect shielding can block higher-frequency contributions to measurement noise and potentially *reduce* the module's resilience.

When the ASIC was hardened with twisted and dense wider power grid, using the optimal configurations identified in Section 3.5.3, the attackers could disclose the AES key with ~5.3/5.9×10^4 and ~8.5/9.2×10^4 measurements in time/ frequency domain, respectively. For

these physical design strategies, while no logic blocks were added, implying little to no power overhead, layout changes increase the module's area.

The resilience of the 9 AES implementations against fine-grained EM SCA attacks and the costs of this evaluation are shown in Table 3.2. In Table 3.2, the resilience improvement is calculated as the ratio of an implementation's marginal cost over that of the baseline module. The improvement for security evaluation is quantified by dividing the acquisition costs of the alternative methods by that of the proposed method. In each case, both time- and frequency-domain EM SCA attacks were performed but the acquisition costs are compared only for the attack that had the lower marginal cost.

Table 3.2 shows that among all countermeasures, masking and shielding countermeasures were most effective in improving the chip's security. In all 9 cases, the ANOVA method required the fewest measurements to evaluate the EM SCA security of the AES implementation. Applying the proposed method was ~1.7-37× cheaper than the adaptive scan protocol, ~3.6-7.7× cheaper than the TVLA followed by exhaustive correlation analysis, and ~1.6-5.3× cheaper than the TVLA-informed correlation analysis. The protocol was particularly efficient when evaluating the most secure implementations.

3.6 SUMMARY

In this chapter, an ANOVA-based measurement method was presented to evaluate fine-grained EM SCA vulnerability of cryptographic modules. The method was used to evaluate 2 baseline and 7 hardened implementations of the AES algorithm against fine-grained EM SCA attacks. The method is implemented in multiple stages; in the first two stages, it eliminates probe configurations posing the lowest risks by estimating the contribution of measurement and algorithmic noise in observed fields, in the last stage it applies correlation-analysis informed by the risk estimates identified in the previous stages

to actually reveal the AES key. The method assumes a gold-box threat model and uses specifically chosen inputs and encryption keys in order to evaluate measurement and algorithmic noise with few measurements. The (gold-box) ANOVA method required upto ~37×, ~7.7×, and ~5.3× fewer measurements than the (black-box) greedy-search correlation analysis, the (white-box) TVLA followed by exhaustive correlation analysis, and the (white-box) TVLA-informed correlation analysis, respectively. The proposed method is particularly efficient for evaluating the most secure chips, such as the shielded-FPGA implementation, where it discards ineffective measurement configurations at a relatively low acquisition cost. Thus, it enables rapid empirical evaluation of how effective a countermeasure is for hardening a cryptographic module against fine-grained EM SCA attacks.

The proposed method can be used with alternative methods [39]-[40] in Stage III, if the set of probe configurations can be sufficiently condensed in Stages I and II. The proposed method can also be extended to evaluating other computing systems by suitably modifying definitions of target and background processes; e.g., a related ANOVA method was used in [4] to evaluate the security of a general-purpose embedded system. Further, it can be combined with a powerful pre-characterization method, demonstrated in [41].

4. Fine-Grained EM SCA-Based Instruction Disassembler[3]

An instruction-level disassembler, based on analysis of near-field electromagnetic (EM) signals emanated during program execution, is demonstrated in this chapter, to deduce the instructions of interest and recover the execution trace of programs on general-purpose microcontrollers. The initial sections of this chapter focus on introducing SCA-based disassemblers, comparing relevant work, and briefly summarizing the fine-grained EM SCA approach. This approach is later elaborated, and demonstration of the disassembler on a general-purpose micro-controller is performed towards the end of the chapter.

4.1 Introduction to SCA-Based Disassemblers

On-chip computations impact the electromagnetic (EM) fields emanated as well as the power consumed by embedded systems [42]-[50], causing information about the operations they execute to leak through these side channels. By probing these fields and exploiting variations in the measured signals, side-channel analysis (SCA) attacks can non-invasively recover information about target processes even in embedded processors that execute general-purpose programs. At the highest fidelity, EM SCA can potentially disassemble a program's execution trace from a device under test (DUT) at the instruction level. Although such instruction-level disassemblers based on power SCA are well documented [44]-[46], only a few attempts based on EM SCA are reported in the literature [48]-[49]. Disassemblers using relatively large EM [6], or power [3]-[5] probes aggregate the fields emanated or power consumed by many/all system components throughout the

[3] This chapter is partly based on an accepted publication: V.V. Iyer, A. Thimmiah, M. Orshansky, A. Gerstlauer, A. Yilmaz, "A hierarchical classification method for high-accuracy instruction disassembly with near-field EM measurements," in publication.
The author contributed to the formulation, implementation, and measurements presented in this article, as well as the writing of this manuscript.

DUT. Thus, any potential features in the measured signals that can distinguish instructions are heavily obfuscated by algorithmic noise from uncorrelated processes in addition to measurement noise from the environment and the sensor setup [21]. Such coarse-grained EM/power SCA setups generally require extensive measurements to quantify and filter out noise [44]-[48]. Contrarily, fine-grained EM SCA setups [21], [48], which use relatively small probes, are sensitive to the fields emanated by a subset of system components near the probes because EM emanations decay rapidly with distance and are polarized. Indeed, when probes are *appropriately positioned and oriented*, fine-grained EM SCA can improve the success rate of disassembly [48]. Thus, fine-grained EM SCA attacks first scan for effective measurement configurations that have high signal-to-noise ratios and then use these low-noise configurations to actually extract information [48], [4]. However, the "acquisition cost" of finding optimal configurations in existing fine-grained approaches can be prohibitively large [21]. The efficiency of a disassembler directly relates to how well the instructions are profiled during the initial acquisition phase, which dictates the acquisition cost in terms of measurement time and storage requirements. A naïve profiling approach involves instantiating each instruction with all possible combinations of different operands, addresses, and data present in architectural registers, such as program counters, stack, etc. [44]-[46]. To feasibly profile instructions, conventional SCA-based disassemblers typically sub-sample this space of architectural states by randomly instantiating instructions several times with different operand values and machine states. This approach has limited feasibility for fine-grained EM SCA-based disassemblers because of the high acquisition cost of searching a 5-D space of potential optimal measurement configurations— the possible probe locations (3-D), orientations (1-D), and observation times (1-D) —as the DUT executes many instantiations of each instruction [21]; e.g., the setup used in this article would require $\sim 5000\times$ more signals to be collected

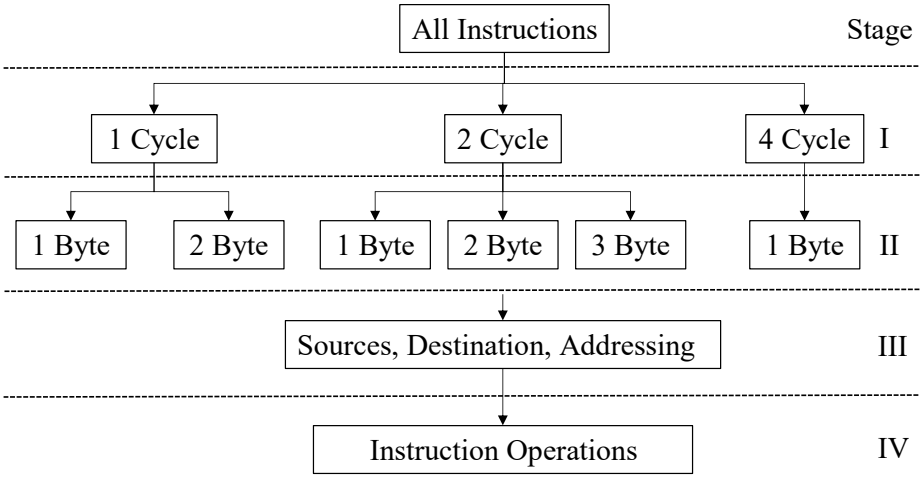


Figure 4.1: Hierarchical grouping of instructions based on length (I), size (II), operands (III), and functions (IV).

compared to using a single probe configuration. The scalability of such methods further reduces as the size of the instruction set N increases. Indeed, fine-grained EM SCA approaches using the random instantiations method for profiling instructions [48] have been limited to small instruction sets. Random instantiations may also miss critical corner cases which can lead to potential misclassifications in the classification phase.

In this work, a novel scalable and effective instruction disassembler using fine-grained EM signals is proposed. As in previous SCA-based disassemblers [44]-[48], the proposed method has 2 phases. *The feature-selection phase* identifies optimal measurement configurations and corresponding signal features. After this phase, *the classification phase* identifies instructions from signals measured as the DUT executes an arbitrary code. It collects signals using only the selected set of configurations and evaluates them according to the features identified in the first phase. To support large instruction sets, the disassembly is performed hierarchically; a 4-stage hierarchy—consisting of an instruction's cycle length, size, operands used, and functions implemented (Fig. 4.1)—is

used; and the feature-selection phase is performed bottom-up, while the classification phase is performed top-down through the hierarchy. A hierarchical classification allows evaluators to identify distinct leakage-mode informed features pertinent to each stage. Furthermore, ensuring high classification success rate in upper hierarchical levels enables evaluators to still recover key information about the executed instructions even if accuracy in separating details on lower levels is reduced.

The hierarchical classification is combined with a leakage model-informed sub-sampling of potential architectural states to profile instructions and identify optimal features for each stage in a feasible and scalable manner. The feature-selection phase uses a Hamming weight (HW) leakage model to design "profiling codes" consisting of a condensed set of test instructions such that—if there was no noise and if the leakage model was valid—the signals measured as the DUT executes these codes would min-max bound the signals that would be measured as the DUT executes all possible instantiations of the profiled instructions. The min-max signal envelopes for each instruction class are collected and stored in the hierarchical database, as the profiling codes are executed. Configurations where pairs of instruction classes can most easily be separated are identified. The signals measured at these configurations are the "features" that are used to classify instructions using binary classification with majority voting [46] in the next phase.

In addition to measured signals, this work also uses novel "differential signals" derived from them to improve success rates. These signals capture the impact of an instruction on the architectural state over multiple cycles. The capabilities of the disassembler are further augmented by assuming branches taken and not-taken as separate instruction classes, enabling control-flow prediction. The proposed method enables high-resolution measurements at a low acquisition cost, efficiently identifying highly potent features within a large search space. As a result of the leakage-model-informed feature

selection, and hierarchical classification, improved success rates are observed for application benchmarks, compared to alternative methods [45], [48].

The contributions of this work can be summarized as follows:

- Fine-grained EM SCA-based disassembly is performed by identifying optimal probe configurations and corresponding signal envelopes during the feature-selection phase.
- In addition to directly probed signals, novel differential signals derived from them are used as features.
- Control-flow leakage prediction is enabled with input-constrained analysis of branch instructions.
- Success rates of ~99% and ~97% are observed when the proposed method is used to disassemble test codes and application benchmarks from the Dalton project [14] executed by a AT89S51 microcontroller unit implementing the i8051 instruction set [51] ($N = 90$ instructions).

4.2 OVERVIEW

This section reviews previous SCA-based disassemblers and presents an overview of the proposed approach.

4.2.1 Relevant work

Various SCA-based methods exist for recovering information about target processes on embedded systems. Code-monitoring with SCA is most often used to identify fixed instruction sequences, separate basic blocks, and predict control flow [42],[43] based on some *a priori* knowledge of an evaluated benchmark. Using SCA to disassemble

	[44]	[47]	[48]	[46]	[49]	[64]	This Work
DUT	PIC16F 687	ATMega 328	PIC 16F687	ATMega 328P	PIC16F15376	Cortex M0	AT89S51
# of Instr. (N)	33	2	33	112	50	17	90
Side-Channel	Power	Coarse-grained EM	Fine-grained EM	Power	Fine-grained EM	Power	Fine-grained EM
# of Samples Measured per Instr. ($N_{\mathrm{pc}} \times N_{\mathrm{t}} \times \bar{N}_{\mathrm{inst}}$)	$\sim 2 \times 10^6$ ($1\times 1000 \times 2000$)	$\sim 2 \times 10^4$ ($1\times 100 \times 200$)	$\sim 1.2 \times 10^8$ ($20 \times 2500 \times 2350$)	$\sim 1.5 \times 10^5$ ($1\times 50 \times 3000$)	$\sim 3.2 \times 10^7$ ($400 \times 2000 \times 40$)	$\sim 1.1 \times 10^7$ ($1 \times 6000 \times 1768$)	$\sim 4.7 \times 10^7$ ($5200 \times 1000 \times 9$)
Success (test code)	~70.1%	100%	~96.2%	~99.0%	~95.0%	~99.0%	~99.3%
Success (application code)	~50.8%	–	~87.7%	–	–	~88.2%	~97.3%

Table 4.1: Comparison of relevant work

individual instructions from an arbitrary unknown code as in [44]-[48] is far more challenging in part because each instruction impacts a multitude of architectural blocks differently. Disassemblers can be compared based on their success rates and their acquisition cost. While success rate is simply the ratio of correctly identified instructions and total number of executed instructions, the acquisition cost is a function of the number of sensor configurations used during profiling N_{pc}, the number of instantiations performed to characterize each instruction $\bar{N}_{\mathrm{inst}}$, and the number of samples collected for each of these measurements N_t. The acquisition cost in this work only accounts for samples stored post measurement collection, and does not quantify repeated measurements and averaging performed by the oscilloscope software.[4]

[4] Please note that the acquisition cost here only quantifies storage requirements and not acquisition time. Acquisition time is related to several setup-dependent factors including oscilloscope features, DUT parameters, averaging method, etc., some of which are not always available in literature.

Instruction disassembly based on coarse-grained EM or power SCA setups [44]-[47] uses a single sensor configuration ($N_{\mathrm{pc}} = 1$) and requires significant post-processing of the signals measured as the DUT executes an extensive set of test instructions. In [44], a power SCA-based disassembler, using principal component analysis (PCA) for feature selection and a multivariate Gaussian classifier, was proposed to evaluate a small instruction set ($N = 33$). It correctly recognized ~71% and ~51% of instructions in test code and application benchmarks, respectively. The method in [44] assumes some *a priori* knowledge of the code, however, as it applies hidden Markov models to blocks of the executed code. In [47], a coarse-grained EM SCA-based disassembler, using PCA with frequency-domain signals for feature selection and AdaBoost, support vector machine, and other methods for classification, was proposed. It was able to distinguish 2 instructions with a 100% success rate. Unfortunately, the method's performance for the remaining instructions was not evaluated in [47]. A larger instruction set ($N > 100$) was evaluated in [46] with a power SCA-based disassembler, using Kullback-Leibler (KL) divergence for feature selection and quadratic discriminant analysis for classification. The method disassembled a test code with ~99% success rate. Although [46] used hierarchical classification, included an extra method to improve success rates for application benchmarks, and recovered 2 instructions implemented in one such code with 92% success rate, the method was not evaluated comprehensively on real-world application benchmarks. In [64], an instruction disassembler targeting a Cortex M0 processor was proposed, implementing KL divergence for feature selection and classification algorithms demonstrated in [46], which was further enhanced by using models based on multi-layer perceptron and convolutional neural network. While the method recognized ~99% and ~88% of instructions in test code and application benchmarks respectively, the disassembly was limited to a small subset of the full instruction set ($N = 17$).

Instruction disassembly based on fine-grained EM SCA was demonstrated in [48],[49]. A small instruction set ($N = 33$) was evaluated in [48] using linear discriminant analysis for feature selection and a k-Nearest Neighbor algorithm for classification. While the disassembler recognized ~96% of the instructions in a test code and ~88% of them in application benchmarks, the approach in [48] is an invasive method that requires decapsulation of the DUT to constrain the search space of configurations during feature selection. A similar fine-grained setup in [49] targeted a slightly larger instruction set ($N = 50$) by performing bit-level disassembly of opcodes, training quadrature discriminant analysis-based classifiers to identify individual bit transitions as instructions are pre-fetched. Although the disassembler recognized 95% of instructions in test codes, it was not evaluated on real benchmarks.

While the methods proposed in [44]-[49], [26] (Table 4.1) have very high success rates when disassembling test codes that follow the same structure/template as the profiling codes they use to select features, their success rates either decrease markedly or are unknown when disassembling application benchmarks; moreover, the methods in [44],[46],[48], [64] which were developed and tested with only limited number of instructions, may not scale well as N, the instruction set's size, increases. Another issue common to the methods in [44]-[49] is that they do not elaborate on the disassembly of conditional branches; such branches requires careful consideration during both phases of disassembly and can enable the detection of possible transitions to different parts of the code and the evaluation of control flow for comprehensive disassembly. Finally, the methods in [44]-[48] extensively instantiate instructions with randomized operands, in different sequences, etc.; they instantiate each instruction from 200 [47] to 3000 [46] times. These methods cannot be directly extended to fine-grained EM SCA because their acquisition costs would be infeasibly high, especially if the number of possible instructions

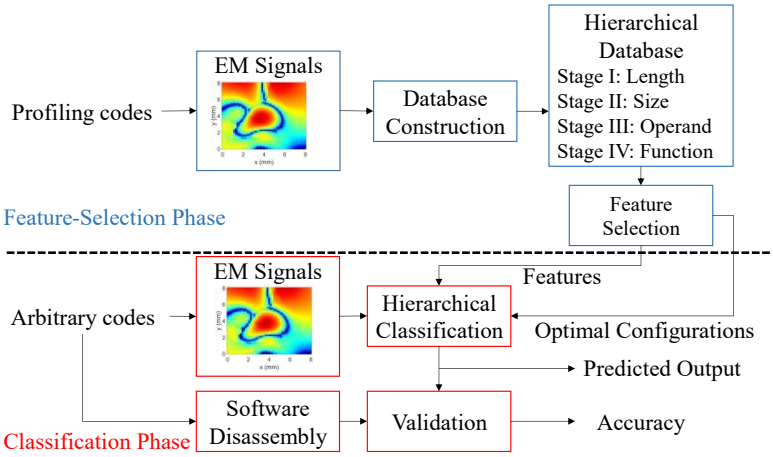


Figure 4.2: Overview of the proposed approach.

and measurement configurations is large. By contrast, our proposed method aims to (i) improve the success rate of disassembly for application codes, (ii) identify if branches were taken/not taken during execution, and (iii) maintain a feasible acquisition cost even for large instruction sets and high-resolution EM probing.

4.2.2 Proposed Approach

As mentioned in the Introduction, the proposed method consists of two phases (Fig. 4.2). In the *feature-selection phase*, EM fields emanated from the DUT are collected for all instructions by designing and using profiling codes that instantiate each instruction for multiple specific machine states, chosen according to the HW leakage model [4], [54]. The signals are collected with all measurement configurations in a 5-D search space consisting of the probe location, probe orientation, and time interval. Next, the min-max bounds of signals—directly probed fields, as well as differential signals derived from them—are found for each instruction, and these signal envelopes are compiled within a hierarchical database. The database stores for each instruction—at the bottom stage of the hierarchy—real-valued envelopes that are multivariate functions of the measurement configuration,

i.e., they are functions of 5 variables. For the upper stages of the hierarchy, instructions are grouped using certain instruction attributes (Fig. 4.1), and the database is compiled bottom-up, i.e., the envelopes for the instruction classes in the upper stages are constructed using envelopes for instruction classes compiled in the lower stages.

Once the database is constructed, it is used to identify optimal measurement configurations and features for binary classification. During feature selection, the envelopes for each instruction class are compared pairwise (one at a time) to those of other classes at the same stage; the comparison identifies M configurations, where the pair's signal envelopes are most distant; i.e., these are the optimal values of the 5 variables to distinguish the pair from each other. The signals obtained with the optimal measurement configurations, i.e., the selected features, and the envelopes of the two classes corresponding to them are recorded for use in the next phase. In the *classification phase*, signals measured while the DUT executes arbitrary codes are categorized hierarchically starting from the top stage. At each stage, candidate classes are identified given the class selected in the previous stage, using binary classification with majority voting [46].

4.3 BACKGROUND

This section describes the DUT's measurement setup, the SCA threat model, the hierarchical grouping of the instruction set, and the signals used in the proposed method.

4.3.1 Measurement Setup

To demonstrate the proposed method, this article uses the AT89S51 microcontroller, which implements 111 instructions, differing in function, size, length, addressing mode, source and destination operands, etc. [51]. The setup used for the measurements is shown in Fig. 4.3. The DUT was operated at 2 MHz. Fields were sensed

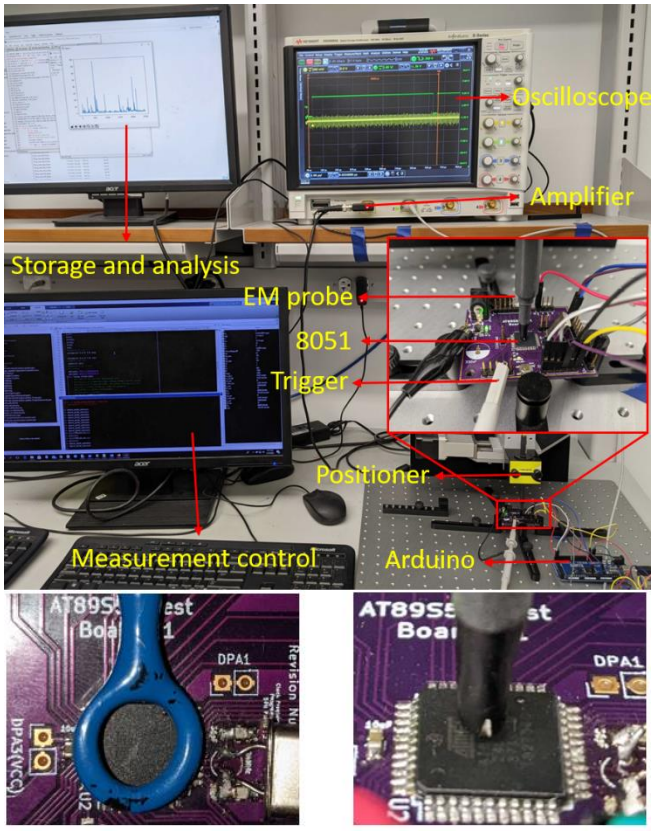


Figure 4.3: Measurement setup used for instruction disassembly (top, same as in [4]) and probes used for coarse-grained (bottom-left) and fine-grained (bottom-right) EM SCA.

using a 1-mm H-field probe, positioned at a fixed height of 0.5 mm and various points on an equally spaced 51×51 grid over the DUT's surface (area~8×8 mm^2) using Riscure's probe positioner. Measurements were performed using both *x*- and *y*- oriented probes. Therefore, N_{pc} =5202 probe configurations were used for constructing the database. Signals were collected and analyzed using a Keysight DSOS054A oscilloscope, at a sampling rate of 2 GS/s ($N_{\mathrm{t}} = 1000$ samples); the signals were collected 50 times and averaged to minimize measurement noise. For comparison and validation, measurements using the coarse-grained EM SCA setup were also performed, using a 10-mm H-field probe. HEX files for programs, generated using Keil's 8051 emulator, were uploaded to

the program memory of the chip using an Arduino as interface. These codes included start/end markers to simplify measurements, implemented via a general-purpose I/O pin. The probe positioning, data acquisition, and subsequent data storage were automated to save experiment time. To reduce storage requirements, samples were saved as single-precision floating-point numbers in binary file format. More information on the setup can be found in [54]. Only $N = 90$ instructions were considered for the following analyses; instructions that use external and indirect addressing modes were excluded because such instructions are seldom used by compilers for general-purpose codes, unless access to external memory is required, and because the focus of this article is on EM emanations arising from on-chip switching activity.

4.3.2 Threat Model

Different threat models are assumed in the feature-selection and classification phase experiments. To allow accurate profiling, limited restrictions are placed on evaluators during the first phase. As in previous works [45]-[48], the feature-selection phase assumes that evaluators have the ability to control a clone of the DUT, or the DUT itself such that they have the ability to send known profiling codes to the device and observe the internal architectural state of the microcontroller as each instruction is executed. Further, the evaluators are assumed to also have the ability to repeat such codes as many times as desired, allowing field measurements to be averaged to minimize measurement noise. In contrast to this transparent "white-box" model of the feature-selection phase, a more restrictive "gray-box" model [9] is used in the classification phase. In this model, the code being executed, the inputs, and the internal operations of the DUT are assumed to be not visible to the evaluators but the evaluators are assumed to still have the ability to repeat the codes being targeted, similar to the setup used by other fine-grained EM works that

Length	Size	Operands	Functions
1Cycle (51 ins)	1Byte (25 ins)	Acc[1]	INC; DEC; RR; RRC; RL; RLC; SWAP; DA; CPL; CLR
		Acc,Reg	ADD; ADDC; SUBB; ORL; XRL; ANL; MOV; XCH
		C-bit[2]	SETB; CLR; CPL
		Reg[3]	INC; DEC
		Reg,Acc	MOV
		No ops.	NOP
	2Byte (26 ins)	Acc, Imm[4]	ADD; ADDC; SUBB; ORL; XRL; ANL; MOV
		Acc, Dir	ADD; ADDC; ORL; ANL; XRL; SUBB;MOV; XCH
		Dir[5]	INC; DEC
		C-bit, Bit	MOV
		Bit[6]	CLR; CPL; SETB
		Reg, Imm	MOV
		Dir, Acc	ORL; ANL; XRL; MOV
2Cycle (51 ins)	1Byte(5 ins)	Acc, Dptr[7]	MOVC
		Acc, PC[8]	JMP; MOVC
		No ops.	RET;RETI
	2Byte (17 ins)	Addr[9]	ACALL; AJMP
		C, Bit	ANL; ORL
		Reg, Off[10]	DJNZ
		Off	JZ; JNZ; JC; JNC; SJMP
		C, /Bit	ANL; ORL
		Dir	PUSH;POP
		Reg, Dir	MOV
		Dir, Reg	MOV
		Bit, Cbit	MOV
	3Byte (15 ins)	Dir, Imm	MOV; ANL; ORL; XRL
		Bit, Off	JB; JBC; JNB
		Addr	LCALL;LJMP
		Acc, Imm, Off	CJNE
		Acc, Dir, Off	CJNE
		Reg, Imm, Off	CJNE
		Dir, Off	DJNZ
		Dir, Dir	MOV
		Dptr, Imm	MOV
4Cycle (2 ins)	1Byte (2 ins)	Acc, B[11]	MUL;DIV

[1]Accumulator, [2] Carry Bit, [3] General Purpose Registers, [4] Immediate Value,[5] Direct RAM Address, [6] Register Bit, [7] Data Pointer, [8] Program Counter, [9] Branch Address, [10] Branch Offset, [11] B Register

Table 4.2: Instruction Groups

combine measurements from multiple locations to increase success rates of disassembling instructions [48], [49], or identify an instruction's functional units [55].

4.2.3 Hierarchical Grouping of Instructions

Attempting to directly classify measured signals within a large set of candidate instructions increases the odds of misclassification. Hierarchical classification can decrease the misclassification risk by reducing the number of possible candidates in each stage, assuming the stages in the hierarchy are appropriately chosen for the DUT (poor groupings can result in potentially more misclassifications at the upper stages). In [46], a 2-stage hierarchy was used: the instructions were separated into 8 groups based on operands and into sub-groups based on their function. That grouping is not suitable for microcontrollers that have a large number of possible operands (>30 for AT89S51). Instead, in this article, 2 higher stages, where instructions are grouped according to length and size, are added to the hierarchy. In Stages III and IV of the hierarchy, instructions are grouped based on operands and their functions as in [46], resulting in 4 stages of hierarchy (Fig. 4.1). These 4 attributes of each instruction ins are represented with the label $\mathrm{ID}_{ins} = (L, S, Op, Fn)$. Here, L denotes the length, S the size, Op the operands, and Fn the function of the instruction i.e., how long it requires to complete execution, the number of bytes fetched from program memory for it, the memory locations of the chosen data values in it, and the operations it performs, respectively. In AT89S51, instructions require $L \in \{1,2,4\}$ cycles for execution, are of size $S \in \{1,2,3\}$ bytes, have 30 possible operands, and implement 45 functions. Table 4.2 shows the resulting hierarchy. In the following, cycle lengths and sizes are represented with the suffixes C and B; e.g., the label for the 1 cycle 1 byte instruction INC Acc is $\mathrm{ID}_{INC\ Acc} = (1C, 1B, Acc, INC)$.

4.2.4 Observed Signals' Dependence on Chip Processes

Fields Signals collected by a near-field probe above a DUT are functions of 5 variables in the measurement setup used (Fig. 4.3): The probe's configuration pc—its transverse location (x, y), height h, and orientation o relative to the DUT—and the time of observation t. Thus, the probed fields can be represented as 5-dimensional functions $V(pc, t)$. Of course, the measured signal also depends on the processes pr that the DUT is executing, i.e., the state of the microcontroller. These processes are performed at specific time-intervals within a DUT's machine cycle, localizing features temporally. The processes can be abstracted as a combination of a target process Tpr_i and one or more background processes Bpr_j, where the subscripts i and j represent versions within these processes [9]; e.g., if the entire instruction opcode is considered the target process, then the 90 target versions are $Tpr_1 \equiv$INC Acc, $Tpr_2 \equiv$ DEC Acc, …, $Tpr_{90} \equiv$DIV Acc, B and the background processes include data operations in various architectural registers. The background processes can be represented using the state of architectural registers $X \in \{\mathrm{X}_1, \ldots \mathrm{X}_{N_\mathrm{x}}\}$, where each state X_k represents a unique data value in registers (RAM, stack, program counter, etc.) and N_x is the number of combinations of register contents. Thus, the signals can also be represented as 7-dimensional functions $V(pc, t, Tpr_i, Bpr_j)$. Using the notation in [4], a signal's dependence on measurement configuration and processes executed on the DUT are highlighted with super/sub-scripts; e.g., $V^{pc,t}_{Tpr_i,Bpr_j}$.

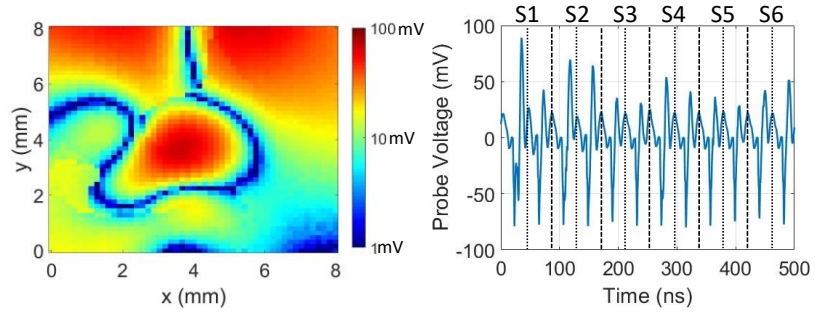
In addition to the probed fields $V^{pc,t}_{Tpr_i,Bpr_j}$, the differential signal

$$\Delta V^{pc,t}_{Tpr_i,Bpr_j} = \left| V^{pc,t+\Delta t}_{Tpr_i,Bpr_j} - V^{pc,t}_{Tpr_i,Bpr_j} \right|, \tag{4.1}$$

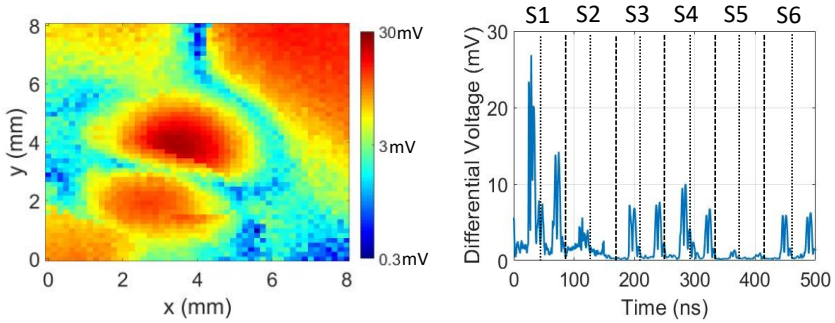
is introduced. Here, Δt is the product of cycle length L of the target process Tpr_i and clock period T_clk. In this work, the differential signals are computed between the corresponding clock cycles of adjacent instructions. While traditional differential side-channel analysis assumes observed signals in a single clock cycle represents the transition

between different machine states, the differential signal introduced in this article computes differences in fields over multiple clock cycles, i.e., it captures the change in fields measured from before an instruction is executed, to after it is executed. This is a useful quantity for separating instructions that modify contents of architectural blocks shared across the instruction set, such as program counters, or the pre-fetched architectural registers. For instance, the 8051 reserves certain sub-cycles to operate on the accumulator or certain RAM registers [51], irrespective of the executed instruction, enabling easier identification of instructions impacting these registers with differential signals. Example signals are plotted in Fig. 4.4.

If a single-stage disassembler was used, the target process would be the complete instruction opcode. Thus, each version of the target process from Tpr_1 to Tpr_{90} would represent a candidate opcode for disassembling the observed signals. The large set of candidates poses major issues in feature selection and classification; e.g., a total of ${}^{90}_{2}C$=4005 classifiers are required for binary classification [46]. In contrast, the proposed 4-stage hierarchical disassembler constructs only 281 classifiers because there are relatively small numbers of candidate classes in each stage. What constitutes target and background processes, however, changes at each stage of the hierarchy. The target process in each stage is a different attribute of the opcode, identified by the label $\mathrm{ID}_{ins} = (L, S, Op, Fn)$. Because classification in each stage distinguishes instructions based on only one attribute, the remaining attributes of the opcode are assumed to be part of the background: In Stage I, the target instruction length can take values from the set $L \in \{1\mathrm{C}, 2\mathrm{C}, 4\mathrm{C}\}$. Here Bpr for $L = 1\,\mathrm{C}$ instructions includes any combination of the architectural state X, and the 51 groups of $(1\mathrm{C}, S, Op, Fn)$ in Table 4.2. The hierarchy then enables independent analysis within each branch in the following stages; e.g., in Stage II, the instruction size is analyzed separately for 1C instructions (for which $S \in \{1\mathrm{B}, 2\mathrm{B}\}$) and



(a) Probed Fields



(b) Differential Signals

Figure 4.4: Space-time distribution of (a) probed fields, and (b) differential signals derived from them, measured by a *y*-oriented probe at 51×51 locations for MOV A, #00 instruction. Spatial maps are plotted at 25 ns and time variations are plotted at the center location. Each machine cycle is divided into 6 states and 2 sub-states [51].

2C ones (for which $S \in \{1B, 2B, 3B\}$). Although attributes (S, Op, Fn) are assumed to be "background" processes here, they are still constrained by target process versions being evaluated, unlike the state of background architectural registers that is unrestricted.

4.4 PHASE I: FEATURE SELECTION

This section details the database construction, the profiling codes, and the feature-selection method in the first phase of disassembly.

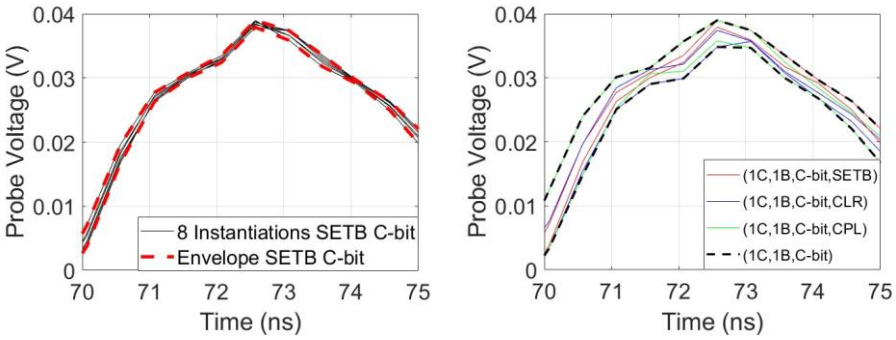


Figure 4.5: The envelopes in stage IV portion of the database (left) are the min-max bounds of the probed fields for multiple instantiations of each instruction; here, the SETB C-bit instruction. The instantiations have different initial conditions of the C-bit (0 and 1) and RAM registers (0x00 and 0xFF). The envelopes in stage III portion of the database (right) are the min-max bounds of the envelopes of all instructions that have the same operand; here, C-bit.

4.4.1 Database Construction

Each instruction class is characterized by 4 signal envelopes in the database; these envelopes are 5-dimensional functions (of pc, t). The hierarchical database is constructed as follows (see Fig. 4.1 for stage definitions). First, the Stage IV of the database is compiled for the 90 instructions. For each instruction Tpr_i, multiple instantiations are executed (see Section 4.4.2), the EM fields are probed using all possible probe configurations, and the min-max envelopes of probed fields and differential signals are stored in the database:

$$\mathbf{env}_{Tpr_i}^{pc,t} = [\min V\,, \max V\,, \min \Delta V\,, \max \Delta V] \tag{4.2}$$

Here, the minima and maxima are found among all instantiations of the instruction, i.e., $\forall Bpr_j \in Bpr$. Next, these 90 instructions are grouped according to their operand class, as per Table 4.2. The envelopes for each of the 35 operand classes in Stage III are constructed by computing the min-max bounds of the envelopes of all the instructions with that operand. Similarly, Stage II (I) portions of the database are compiled from its Stage III (II) portions. Fig. 4.5 shows an example computation of the min-max envelopes.

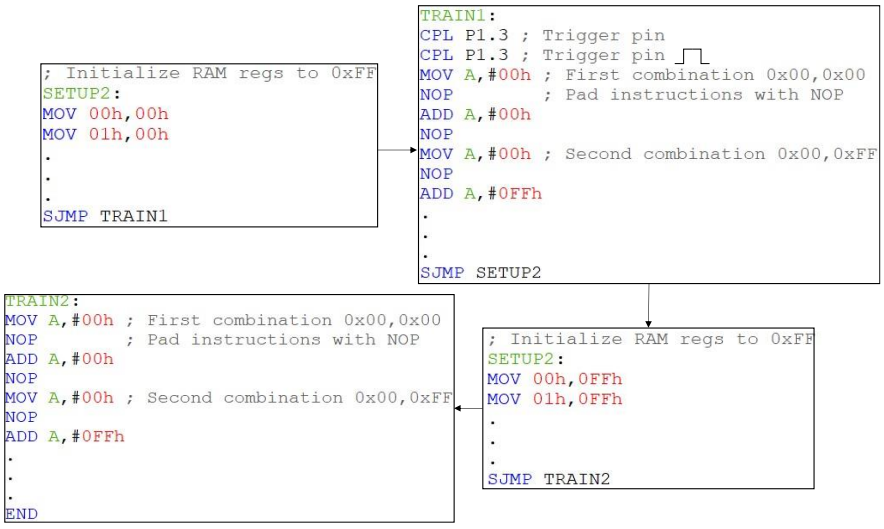


Figure 4.6: Profiling codes instantiate instructions with different operands, under different machine states. NOP instructions are introduced to keep the computation of differential signals consistent.

4.4.2 Method for Selecting Features

One approach to finding the signal envelopes is to collect an extensive set of signals, e.g., by instantiating the architectural registers X with random values. For instance, [46] used 3000 such instantiations per instruction for feature selection. While this can improve classification accuracy for coarse-grained EM/power SCA setups, the acquisition cost for fine-grained EM setups quickly becomes intractable when so many instantiations are used: For $N = 90$ instructions, if $N_{\mathrm{t}} = 50$ time samples of signals are measured as in [46] with a single probe configuration ($N_{\mathrm{pc}} = 1$), a total of 13.5×10^6 samples would be acquired. If they are measured with the fine-grained EM SCA setup in this work, with N_{pc}~5200 probe configurations (Section 4.6), a total of 70×10^9 samples would be acquired. Storing these samples as single-precision floating-point numbers would require ~50 MB of space for the former and ~280 GB for the latter setup. Additional storage may

be required during feature selection, e.g., to transform time-domain data to frequency domain.

A smaller set of signals can be collected by modeling the leakage as if it depends only on HWs of data in architectural registers, a common approach in processor security evaluations [4]; e.g., signals for 256 data values can be bound by those for extreme instantiations of data 0x00 (HW 0) and 0xFF (HW 8). Then, the data-dependency of each instruction—except conditional branch instructions—can be bound by using at most 4 instantiations, by setting operands and result to data values 0x00 and 0xFF. For example, consider the instruction ADD Acc, Imm. To bound its data dependence, the data values in the Accumulator register and the Immediate value in program memory are chosen from the set {(0x00,0x00), (0x00,0xFF), (0xFF,0x00), (0xFF,0xFF)}. Further, to improve coverage of background processes, all 128 bytes of RAM, including stack registers, are instantiated as either 0x00 or 0xFF. Therefore, 8 instantiations are used to characterize each instruction in the profiling codes. Code snippets used to profile this instruction are shown in Fig. 4.6. In addition to the instruction instantiations, extra instructions are used to support measurements, such as a general-purpose pin triggering the oscilloscope for ease of experiment.

Because conditional branches perform different functions depending on the result of the condition evaluation, branches taken and not taken for the same instruction are considered as separate classes in Stage IV, i.e., they have the same instruction length, size, and operands, but different functions. Introducing 12 additional instruction classes for the conditional branch instructions in Table 4.2, control-flow prediction is enabled in the final stage of disassembly. Using 16 instantiations for conditional branch instructions and 8 for other instructions, the proposed profiling codes contain a total of $N\overline{N}_{\mathrm{inst}} = 12 \times 16 + 78 \times 8 = 816$ specially-designed test instructions (in addition to miscellaneous

instructions used as markers for measurement, and various instructions needed to clear flag registers, data memory, or stack). These profiling codes are used to acquire the following total number of samples to construct the database:

$$N_{\text{samp}} = N\bar{N}_{\text{inst}}N_{\text{pc}}N_{\text{t}} \quad (\#\text{ of Samples Acquired}) \tag{4.3}$$

Here, N_{pc} is number of probe configurations, N_{t} is number of time samples, N is the number of instructions, and $\bar{N}_{\text{inst}}$ is the average number of instantiations used to profile each instruction. While $N_{\text{pc}}N_t$ depends on the measurement setup, $\bar{N}_{\text{inst}}$ depends on the profiling method.

4.4.3 Selecting the Features

Feature selection identifies optimal measurement configurations where envelopes (and therefore signals) are easily separable when compared pairwise. Here, as well as in Section 4.5, the process is presented for two instruction classes a and b at the same stage of the hierarchy. First, the "average distance" between the pairs' envelopes is computed:

$$Dist_{a,b}^{pc,t} = \frac{\left|(\mathbf{env}_a^{pc,t}[1]+\mathbf{env}_a^{pc,t}[2])-(\mathbf{env}_b^{pc,t}[1]+\mathbf{env}_b^{pc,t}[2])\right|}{2} \tag{4.4}$$

While feature selection in Stages II-IV directly uses this quantity, a pre-processing step is required in Stage I because signals with different time lengths are compared. It is assumed that the first cycle of multi-cycle instructions is similar to a single-cycle instruction, due to the presence of opcode fetch-related processes. Consequently, in Stage I feature selection, signals for multi-cycle instructions are partitioned into multiple single-cycle windows, similar to [44]. The partitioned windows are then compared separately to single-cycle instructions, assuming the cycles that follow the first cycle will show sufficient differences to allow their length-based classification. Fig. 4.7 shows an example of the distance between single-cycle instructions and the second cycle of two-cycle instructions. The distance $\Delta Dist_{a,b}^{pc,t}$ between the differential signal envelopes is computed similarly. As

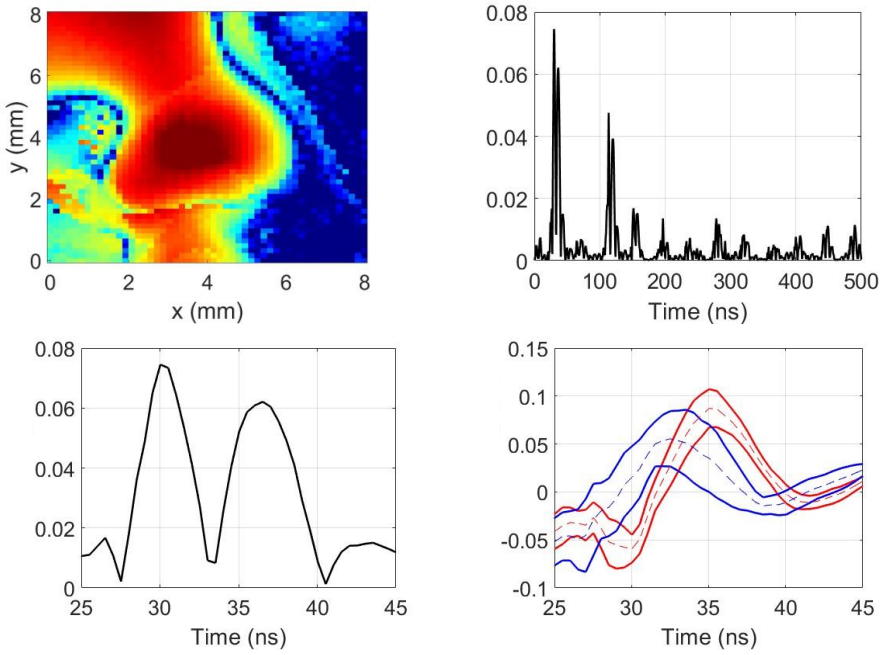


Figure 4.7: Spatial map (top-left) of $Dist_{1C,2C}^{pc,t}$ between 1-cycle and 2-cycle instructions at t~30 ns and time variation (top-right) at an optimal probe location (starred). Distance (bottom-left) and envelope (bottom-right) plots for an optimal time interval showed that instruction classes were more separable when the difference between the envelope averages (dashed) increased, particularly at t~30 and t~37 ns.

demonstrated in Fig. 4.8, some instruction classes are potentially more separable using differential signals. Prediction of a program's control flow can be achieved in Stage IV of the disassembly, as shown in Fig. 4.9.

Next, optimal measurement configurations that maximize the distance between signal envelopes are identified. For each pairwise comparison, $M = 10$ optimal probe configurations—5 each for direct and differential signals—and the corresponding 10 optimal time instances are stored in the arrays $\mathbf{pc}_{a,b}^{\text{opt}}$ and $\mathbf{t}_{a,b}^{\text{opt}}$. The signals at these optimal measurement configurations are the selected features that will be compared with the stored envelopes to classify instructions.

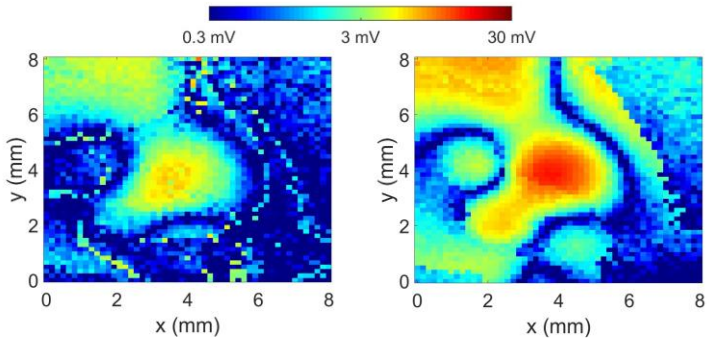


Figure 4.8: Comparing the classes $(1C, 2B, \text{Dir})$ and $(1C, 2B, [\text{Acc}, \text{Dir}])$ in stage III with $Dist_{a,b}^{pc,t}$ (left) and $\Delta Dist_{a,b}^{pc,t}$ (right) shows that they are more separable when using differential signals. Here, $t{\sim}120$ ns.

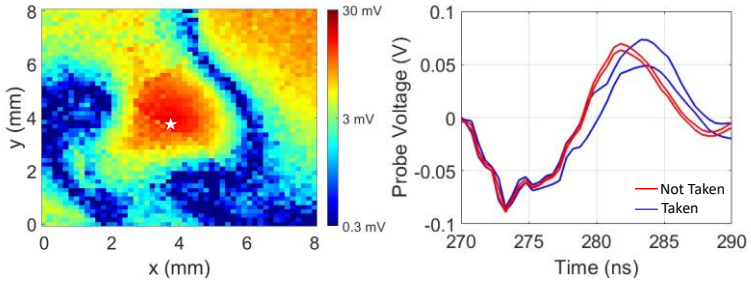


Figure 4.9: Distance between branch "taken" and "not taken" classes for instruction $(1C, 2B, \text{Off}, \text{JNZ})$ in Stage IV (left), shows that the disassembly can potentially predict program flow. The spatial map of distance is plotted at $t{\sim}285$ ns and the observed fields are plotted at an optimal configuration (starred).

4.5 PHASE II: CLASSIFICATION

During classification, the probed field $V^{pc,t}$ and differential signal $\Delta V^{pc,t}$ are compared to the signal envelopes in the database. The deviation of evaluated signals from the envelopes of candidate classes a and b in the database are computed as

$$Dev_{a/b}^{pc,t} = \text{Max}\left\{V - \mathbf{env}_{a/b}^{pc,t}[2], 0\right\} + \text{Max}\left\{\mathbf{env}_{a/b}^{pc,t}[1] - V, 0\right\} \quad (4.5)$$

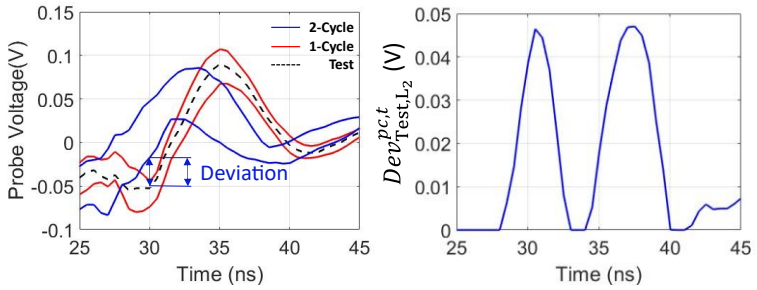


Figure 4.10: An evaluated signal for instruction (1C,1B,Acc,Inc) correctly shows large deviation from envelope of 2-cycle instructions at t~30 ns and t~37 ns.

This metric is 0 if the evaluated signal is within the stored envelope. The deviation of a probed field from the envelopes in Fig. 4.7 is shown in Fig. 4.10. A corresponding metric $\Delta Dev_{a/b}^{pc,t}$ is computed for the differential signals.

During binary classification, the net deviation of the evaluated signal from the two candidates a and b is computed only with the M optimal measurement configurations for separating them:

$$NetDev_{a/b} = \sum_{m=1}^{M/2} Dev_{a/b}^{\mathbf{pc}_{a,b}^{\mathrm{opt}}[m],\mathbf{t}_{a/b}^{\mathrm{opt}}[m]} + \sum_{m=M/2+1}^{M} \Delta Dev_{a/b}^{\mathbf{pc}_{a,b}^{\mathrm{opt}}[m],\mathbf{t}_{a,b}^{\mathrm{opt}}[m]} \tag{4.6}$$

The instruction class with the smaller net deviation is considered the more likely candidate for the evaluated signal. To classify among multiple candidates, the binary classification is implemented with a majority voting method [5]:

$$vote_{a,b} = \begin{cases} +1, \text{ if } NetDev_a \geq NetDev_b \\ -1, \text{ if } NetDev_a < NetDev_b \end{cases}$$
$$a^* = \underset{a}{\operatorname{argmax}} \sum_{b=1\ (b\neq a)}^{N_c} vote_{a,b} \tag{4.7}$$

Here, a^* is the most likely candidate class and N_c is the number of candidate classes.

4.6 EXPERIMENTS AND RESULTS

To test the proposed disassembler, first, each instruction is instantiated 100 times with random operand values. In this test set, each instruction is padded with a NOP instruction, and before the instantiations the RAM registers are cleared, similar to the profiling codes shown in Fig. 4.6. A total of 10200 instructions are evaluated in this test set. This evaluation is similar to the test sets that follow the templates of profiling codes, used in [44]-[48]. For conditional branch instructions, two separate test sets are used for the branch "taken" and "not-taken" cases. The operands in both cases are randomized with constraints, to ensure the functions are correctly executed; e.g., for the jump-if-not-zero instruction's branch "taken" case, the operand is allowed to take all values other than 0.

Second, a more robust and complete evaluation of the proposed disassembler is performed by using a set of 4 application codes from Dalton benchmarks [53], which are specifically designed to optimize the performance of 8051 cores: the greatest common divisor (GCD), Fibonacci (FIB), sort, and square root (SQRT) codes. As their names indicate, the codes compute the GCD of two numbers, generate the first 10 Fibonacci numbers, sort 10 specified integers in ascending order, and find the square root of a specified floating-point number. The compiled codes were first disassembled using KIEL's 8051 emulator, providing a reference assembly code to judge the accuracy of the proposed disassembler.

Third, the potency of fine-grained EM SCA approach is evaluated by implementing the proposed feature-selection and classification methodology using a coarse-grained EM SCA setup (with a relatively large probe [47]) and comparing the success rates of the two approaches. Here, the measurement configurations are optimized only over the time dimension as there is a single fixed probe location and orientation.

4.6.1 Feature-Selection Results

To construct the database with the proposed profiling codes, a total of $N_{\mathrm{samp}} = N\bar{N}_{\mathrm{inst}}N_{\mathrm{pc}}N_{\mathrm{t}} = 816 \times 5202 \times 1000 \sim 4.2 \times 10^{9}$ samples (after they were averaged 50 times by the oscilloscope) were acquired. For comparison, consider applying the methods presented in [44]-[47] directly to the presented fine-grained EM SCA setup: Assuming N_{pc} and N are the same as in this work, but using the same $\bar{N}_{\mathrm{inst}}$ and N_{t} values as in the previous works, the methods would require ~222× [44], ~17× [46], ~2.2× [47], and ~650× [48] more samples than the proposed method.

Results for feature selection phase are exemplified in Fig. 4.11, which shows that the envelope-to-envelope distances reduce across space and time at the lower stages of the hierarchy. This behavior is expected for well-designed hierarchies that progressively refine the granularity of recovered instruction. It was also observed that the spatio-temporal distributions of distances for each stage were different, i.e., each stage of the hierarchy impacted the probed fields differently. Further, it was observed that features for all classifiers were limited to the region marked with white in Fig. 4.11. Consequently, measurements for the classification phase were limited to this region (25×25 locations).

4.6.2 Classification Results

First, the test codes with 100 randomized instantiations of each instruction were disassembled and the recovered results were compared to the reference assembly code line by line. The accuracy is then simply computed as a ratio of correctly recovered instructions to the total number of instructions. The success rate of the disassembly was 10130 out of 10200 instructions (~99.3%). Evaluating accuracy stage-wise showed that the disassembled instructions had 100% accuracy for all instructions in Stages I-III, i.e., all misclassifications were in Stage IV. Therefore, the incorrectly recovered instructions still

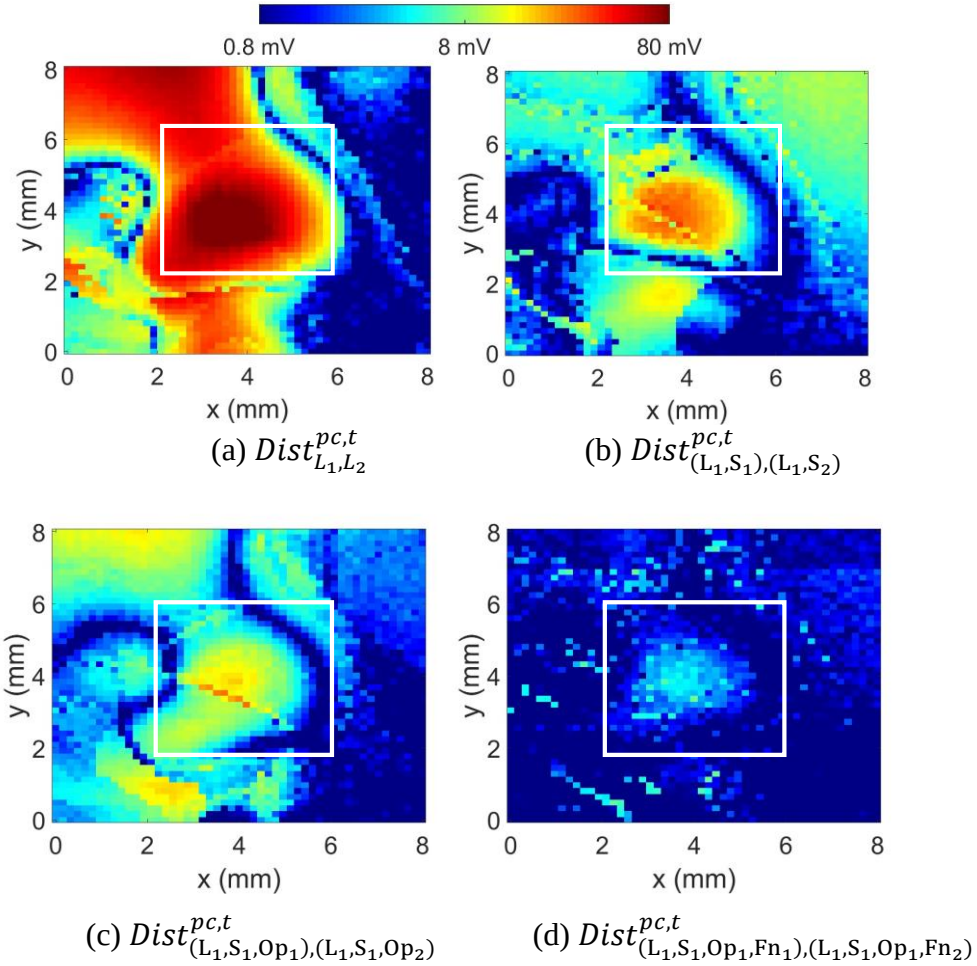


Figure 4.11: Example spatial maps of the envelope-to-envelope distances computed during feature selection phase in stages (a) I (t~30 ns), (b) II (t~270 ns), (c) III (t~360 ns), and (d) IV (t~70 ns), observed at the most optimal time instants. The distances between instruction classes are smaller at lower stages of the hierarchy.

contained some relevant information. It was also observed that all conditional branches were correctly identified, including if the branch was taken or not. Such high success rates are to be expected because these codes follow a similar template to the profiling codes.

Results for the disassembly of application benchmarks are shown in Table 4.3. The total accuracy for the fine-grained setup was found to be ~97%, with less than $\pm$2% variation among the 4 benchmarks. Similar to the evaluation of the test codes, no

misclassifications were observed in the first three stages, and a 100% accuracy was observed in identifying conditional branch instructions. While a slight decrease in the disassembly accuracy was observed for the benchmarks, the difference is minimal compared to the disassemblers demonstrated in [44] and [48]. Finally, the most misidentified instruction for both test codes and benchmarks was the ADDC Acc, Reg, commonly misclassified as instruction ADD Acc, Reg (misclassified in 22 out of 123 instances). Potential reasons for the misclassification have to do with the close functional relation between the ADD and ADDC (i.e., add with carry) instructions, since in the absence of a carry bit, identical operations are performed by the microarchitecture. The opcodes of these instructions in the ISA are also very similar, including how they are decoded. Similar misclassifications were also observed for rotate and rotate with carry instructions that only differ minimally in functionality and operation. However, these instructions are not frequently used by the compiler thereby limiting inaccuracies and misclassification rates in large benchmarks.

The disassembler implemented using the coarse-grained EM SCA only showed a success rate of ~70% disassembling test codes and ~65% accuracy disassembling the benchmarks (Table 4.3). Contrary to the fine-grained measurement setup, misclassifications were observed in Stages II, III, and IV. Clearly, the fine-grained EM SCA setup resulted in a more potent disassembler. An example demonstrating the differences between database envelopes for the fine-grained and coarse-grained EM setups are shown in Fig. 4.12. It was observed that envelopes from the fine-grained setup were narrower and had sharper signal variations compared to the envelopes from the coarse-grained setup. Consequently the min-max envelopes predicted by the coarse-grained setup overlap for multiple classes at selected configurations leading to misclassifications, even when distance predicted between instruction classes is high (Fig. 4.12). Further, the overlap

Benchmark	Code Size (bytes)	# of Instructions	Fine-Grained EM	Coarse-Grained EM
			# of Correct Instructions	Accuracy (%)
GCD	55	111	108	~97.3
FIB	303	804	794	~98.7
sort	572	2665	2556	~95.9
SQRT	1167	2006	1972	~98.3

Table 4.3: Results of Benchmark Evaluations

is also observed to increase in the coarse-grained case, as the classification moves to the lower hierarchical levels.

While the choice of MCU significantly simplified the feature selection and classification phases, parts of the proposed work may be extended to more complex systems. For instance, the feature selection phase in heavily pipelined processors can be split into two sub-phases: The first sub-phase can implement the feature-selection methodology described in Section 4.3, using few select instructions padded with NOPs (Section 4.3.1). Once a sufficiently small set of potent probe configurations are identified, the NOP instructions can be replaced with randomized instructions and operands, similar to the methodology proposed for the power SCA disassembler in [64]. Various feature selection and classification strategies used by power SCA disassembler can therefore be extended to fine-grained EM SCA. However, this is only feasible once an initial characterization, such as the low-cost feature selection proposed in this work, reduces the large search space of potent probe configurations across the chip.

4.7 SUMMARY

A fine-grained EM SCA based disassembler was proposed to recover instructions executed on a general-purpose micro-controller. The proposed method uses a hierarchical

framework to improve feature selection and classification. It identifies optimal measurement configurations that distinguish instruction classes in the first phase by (i) executing model-based profiling codes to efficiently collect probed fields in a database, (ii) finding envelopes that bound the probed fields and, a novel quantity, differential signals derived from them. In the second phase, measured signals with these optimal measurement configurations are classified by comparing them to the signal envelopes of instruction classes one pair at a time. The comparisons were performed by quantifying the deviation of the measured signals from the signal envelopes. The proposed disassembler was shown to successfully and feasibly recover ~97% to ~99% instructions from application benchmarks and test codes executed on an AT89S51 microcontroller. Further, all conditional branch executions were correctly identified, enabling control-flow leakage prediction. It was also observed that the fine-grained EM SCA was significantly more potent compared to a coarse-grained EM SCA analysis.

The proposed disassembler can potentially detect malware within basic blocks [56], as well as those impacting control flow integrity [57]-[59]. Combined with appropriate tools quantifying vulnerabilities in side channels [54], [60], [62], the disassembler can further enable programmers to optimize programs to minimize leakage. Finally, the instruction level granularity of the disassembler enables detection of small-scale hardware trojans that are more challenging to address compared to malicious code [63].

The DUT used in this article simplifies the disassembly significantly because of its low-complex multi-cycle architecture; additional work is required to extend the proposed work to more complex embedded processors. For instance, in [64], randomized instructions were introduced based on the number of pipeline stages, while profiling individual instruction classes. A similar extension can be proposed for the fine-grained

disassembler in this work; e.g., the feature-selection phase in heavily-pipelined processors can be split into two sub-phases: The first sub-phase can implement the feature-selection methodology, using a few select instructions padded with NOPs (Section 4.4.2). Once a sufficiently small set of potent probe configurations are identified, the NOP instructions can be replaced with randomized instructions and operands for reduction, depending on the number of pipeline stages. Additional datasets can also be created for groups with a large number of instructions, to improve their disassembly, similar to [64].

The disassembly can be improved further by recovering data values of operands [4], in addition to instructions. There is also potential to improve disassembly with higher-resolution probes. A more optimal method of combining features from multiple configurations can also reduce misclassifications, with the potential to re-examine predicted results and observe anomalies. Further, differential signals are a novel quantity that requires further exploration, potentially being used to observe changes across multiple pipeline stages as the instruction is executed, adding a new dimension to the analysis. Finally, imposing more restrictions on evaluators in the classification phase, similar to generic black-box testing threat models, may necessitate the use of more potent post-processing techniques in combination with some of the aforementioned potential improvements to the setup. Code monitoring through instruction disassembly presents a non-invasive pathway to detect intrusions, and therefore evaluate embedded hardware security.

5. Modelling Information Leakage in EM SCA[5]

This chapter presents methods to model information leakage via EM side-channels and uses them to evaluate complex embedded systems. The first part of the chapter extends the ANOVA methodology presented for AES in Chapter 2 to generic embedded systems. This method is used to evaluate vulnerabilities in a server implementation of the Bluetooth low energy protocol. The final section of the chapter introduces data-dependent EM basis functions to model side-channel leakage.

5.1 ANOVA For a Generic Computing Chip

The analysis presented in Section 2.1.4 can be extended for any generic embedded computing chip. By assuming independence of target and background signal quantities, ANOVA can potentially be used to evaluate complex embedded systems. Let the array $\mathbf{V}^{pc,t}$ list all the measured signals corresponding to all possible combinations of processes in a chip. Each observed signal in the array, $V^{pc,t}_{\mathrm{Tpr}_i,\mathrm{Bpr}_j^k,r}$, can be decomposed into three independent, abstract signals $T^{pc,t}_{\mathrm{Tpr}_i}$, $N_r^{pc,t}$, and $B^{k,pc,t}_{\mathrm{Bpr}_j^k}$. Here, $T^{pc,t}_{\mathrm{Tpr}_i}$ and $B^{k,pc,t}_{\mathrm{Bpr}_j^k}$ represent the contribution of the target and background processes Tpr_i and Bpr_j^k to the observed signal, whereas $N_r^{pc,t}$ represents the effect of measurement-to-measurement variations. In information-revealing measurement configurations, the observed signal will depend strongly on $T^{pc,t}_{\mathrm{Tpr}_i}$ and will be insensitive to $N_r^{pc,t}$ and $B^{pc,t}_{\mathrm{Bpr}_j^k}$. If the quantities$T^{pc,t}_{\mathrm{Tpr}_i}$, $N_r^{pc,t}$, and $B^{pc,t}_{\mathrm{Bpr}_j^k}$ are listed in the arrays $\mathbf{T}^{pc,t}$, $\mathbf{N}^{pc,t}$, and $\mathbf{B}^{k,pc,t}$, for N_b background processes, their variances are related as

[5] This chapter is partly based on two previous publications:
(i) V.V. Iyer and A.E. Yilmaz, "Estimating near-field signals emanated by embedded systems using data-dependent EM profiles as basis functions," in *Proc. USNC-URSI Rad. Sci. Meet*, July 2023.
(ii) V.V. Iyer and A. Yilmaz, "EM side-channel analysis of data leakage near embedded bluetooth low energy modules," in *Proc. WAMICON*, April 2023.
The author contributed to the formulation, implementation, and measurements presented in this article, as well as the writing of these manuscripts.

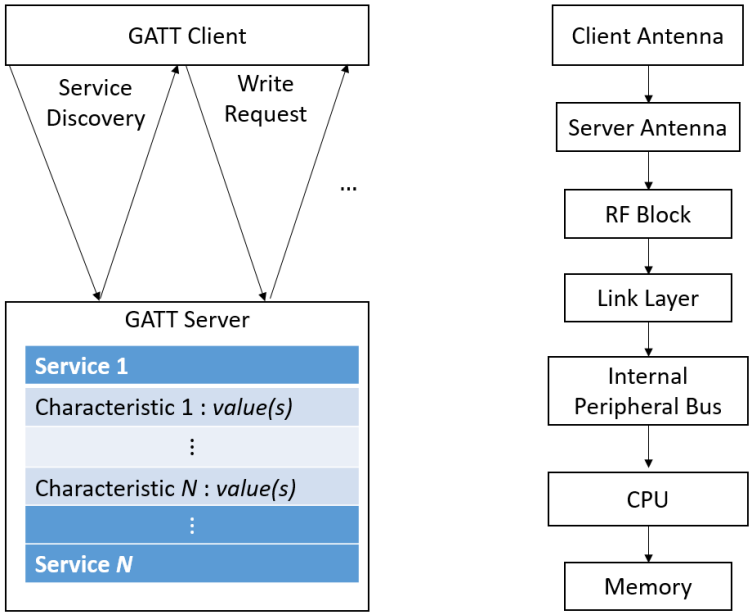


Figure 5.1: (a) Information access flow in the GATT protocol. (b) Flow of data during a write operation [72].

$$\frac{\mathrm{Var}(\mathbf{V}^{pc,t})}{\mathrm{Var}(\mathbf{T}^{pc,t})} = 1 + \underbrace{\frac{\mathrm{Var}(\mathbf{N}^{pc,t})}{\mathrm{Var}(\mathbf{T}^{pc,t})}}_{1/F_N^{pc,t}} + \sum_{k=1}^{N_\mathrm{b}} \underbrace{\frac{\mathrm{Var}(\mathbf{B}^{k,pc,t})}{\mathrm{Var}(\mathbf{T}^{pc,t})}}_{1/F_B^{k,pc,t}} \tag{5.1}$$

As shown previously in Section 2.1, the ratios can be estimated using ANOVA F-statistics. The choice of target and background vary on the computations and data of interest; e.g., in the AES case, a single byte is the data of interest, and remaining 15 byte computations contribute to algorithmic noise. In this chapter, an example demonstrating the usage of the ANOVA F-statistic for data-recovery from a BLE module is presented.

5.2 EM SCA Analysis of a BLE Server Using ANOVA

This section presents an overview of BLE modules, potential vulnerabilities, the method used to evaluate the implementation and the measurement results of the evaluations.

5.2.1 Background

Bluetooth low energy (BLE) is a commonly used wireless communication standard in low-power applications [71]. Once paired, BLE devices can communicate via the GATT protocol (Fig. 5.1(a)): A peripheral GATT server offers various services and characteristics, whose attributes are locally indexed on the embedded device. These attributes are accessed by a paired GATT client, which can read, write, or send requested data using appropriate mechanisms as per the BLE 5.0 standard [71]. In this work, the computations performed by the GATT server as it receives and processes data written to a specific characteristic in a chosen service are the *computations of interest* and the values actually written to the characteristic are the *data of interest*. Thus, data corresponding to the service and characteristic handles, as well as those introduced by various layers in the protocol stack, are extraneous.

As shown in Fig. 5.1(b), once a GATT client antenna sends signals encapsulating the data of interest to the server, the analog circuits and the link layer decode the information and send it to the CPU using a system bus. These decoded signals trigger the "event" corresponding to the chosen service and characteristic, and the data of interest are sent to the appropriate memory location from where they are handled as per functions defined in the software application. The movement of data between registers, via buses, creates vulnerabilities that can be exploited by fine-grained EM SCA attacks [4], [41].

Fine-grained EM SCA attacks are generally implemented in two phases. Phase I has two goals: (i) Identify optimal measurement configurations, where on-chip processes related to the computations of interest contribute most to the signals—or equivalently, where measurement and algorithmic noise have minimal impact [41]. This requires repeating the computations of interest using a few carefully chosen data while collecting signals with numerous (ineffective) measurement configurations; e.g., instructions

executed by a microcontroller were extracted by an EM SCA approach in [70] by identifying 10 optimal probe configurations per instruction pair out of 5.2×10^3 possible configurations. (ii) Construct a database of signals to be referenced in Phase II for data recovery. These signals are collected with *only the optimal probe configurations*, potentially by executing the computations of interest with additional data. In Phase II, the optimal probe configurations are monitored and the data of interest are recovered by referencing the database. The specific methods used in this work are detailed next.

5.2.2 Measurement Protocol

Information leakage via the EM side channel is generally abstracted using leakage models such as HW and Hamming Distance (HD) [4], [41], [70]. Recently, computing the ANOVA F-statistic on EM signals corresponding to data with extreme HW/ HD was found to be a rapid, low-cost method for isolating near-field vulnerabilities [41]. Here, to identify optimal measurement configurations, a similar approach is used in Phase I, which requires repeatedly measuring signals as the same exact computations are executed. Therefore, in the following, the probed signals are denoted as $V_{pr,r}^{pc,t}$; they are functions of not just the probe configuration pc, which represents the probe's location, orientation, and height, the measurement time t, and the processes performed by the chip pr, but also the repetition r as each signal contains varying levels of measurement noise [4].

Phase I starts with the evaluator pairing a client to the peripheral GATT server. The evaluator may then select a specific service and characteristic offered by the server, and write a value to the characteristic. Assuming this value has a size of 1 byte, the data with the minimum and maximum HW of 0 (0x00) and 8 (0xFF) are chosen. Both data values are written N_{r} times while the signals near the chip operating the GATT server are measured. Because the chosen service and characteristic are fixed for each write operation,

the only variation in measured signals comes from variation in the written value and measurement noise. To quantify the impact of this noise at each measurement configuration, the sample mean $\bar{x}_{\mathrm{HW}_{0/8}}^{pc,t}$ and sample variance $s_{\mathrm{HW}_{0/8}}^{pc,t}$ of the N_{r} measured signals ($V_{pr,1}^{pc,t}$, $\cdots$, $V_{pr,N_{\mathrm{r}}}^{pc,t}$) are computed. Then, the F-statistic is computed as [41],

$$F_N^{pc,t} = \frac{2N_{\mathrm{r}} \times \mathrm{Var}(\bar{x}_{\mathrm{HW}_0}^{pc,t}, \bar{x}_{\mathrm{HW}_8}^{pc,t})}{\mathrm{Mean}(s_{\mathrm{HW}_0}^{pc,t}, s_{\mathrm{HW}_8}^{pc,t})} \tag{5.2}$$

Notice that only signals corresponding to the data with extreme HW values (HW 0 and 8) are collected and analyzed here. Next, the computed F-statistic is compared to a critical value $F_{N,\mathrm{c}}$—effective configurations have F-statistic $F_N^{pc,t} > F_{N,\mathrm{c}}$—derived from F-distributions for a confidence interval of 99.99% [41] to judge the potency of the measurement configuration using null-hypothesis testing.

Phase I ends with the evaluator identifying the optimal probe configuration $pc^{\mathrm{opt}} = \underset{pc}{\mathrm{argmax}}\ F_N^{pc,t}$ and constructing the database: The evaluator uses the GATT client to write data spanning all HWs to the GATT server N_{r} times while probing the fields with pc^{opt}. The database stores a single reference signal—the average of the N_{r} collected signals—for each HW.

Next, the EM fields near the chip operating the GATT server are probed with pc^{opt} while arbitrary data values are written to it either by the evaluator's GATT client or an independent one, without any repetition. Measured signals for the test data are compared to the reference signals to identify the best fit as the reference signal with the minimum average distance from the test signal when $F_N^{pc^{\mathrm{opt}},t} > F_{N,\mathrm{c}}$.

5.2.3 Measurement Setup

The setup shown in Fig. 5.2 was used for the measurements. A Cortex M4 processor in the RA4W1 BLE development board [71], operating at 48 MHz, was used as the device under test. A Nokia C01 smartphone was used as the evaluator's client in Phase I. The

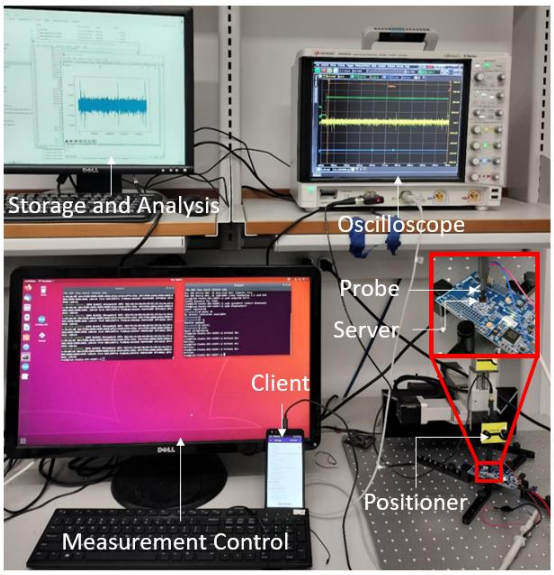


Figure 5.2: The near-field measurement setup used for EM SCA attacks. Experiments are performed on an RA4W1 test board. Near-fields were sensed using an H-field probe, scanning the chip at a height of 0.5 mm.

GATT browser installed in the smartphone paired with the server and accessed the services and characteristics provided by the demo application installed in the board [72]. The service chosen was the Renesas LED Switch Service, the characteristic selected was the LED Blink Rate, and the allowed values to be written ranged from 0x00 to 0xFF. To simplify measurements, markers were included within the software implementing the BLE operation to trigger a Keysight MSOS054A oscilloscope, sampling at 10 GS/s. The oscilloscope was also used to store and analyze data. A Riscure probe station was used to position a 1-mm diameter H-field probe from Langer in 41×41 locations over the 8×8 mm^2 chip surface. Measurements were performed in x and y orientations of the probe at a height of 0.5 mm above the chip. Bluetooth transfers from the smartphone application were automated using the Android debug bridge. Phase II used both the Nokia smartphone and a OnePlus Nord N20 smartphone as a client, both using the same Bluetooth service and characteristic.

5.2.4 Measurement Results

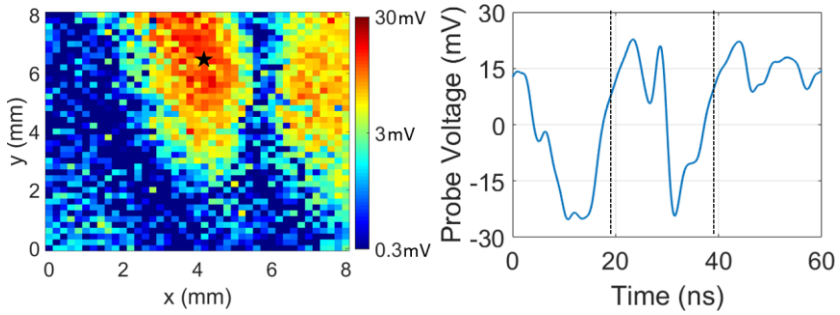


Figure 5.3: Spatio-temporal distribution of measured signals using an *x*-oriented probe. The spatial map is plotted at t~30 ns and time plot is shown for an optimal configuration (star) for three clock cycles, each cycle being ~20 ns.

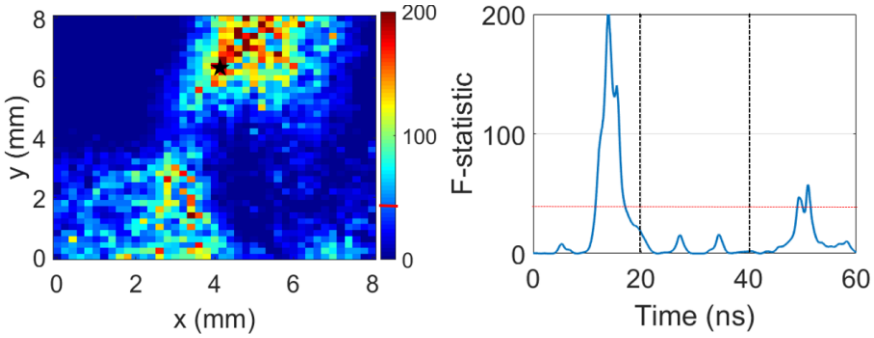


Figure 5.4: Spatial map of $\max_t F_N^{pc,t}$ (left) and time plot of the F-statistic (right) at an optimal configuration (starred). The measurement configuration is suitable for data recovery if the F-statistic is greater than the threshold $F_{N,c}$ (red).

A total of 3362 probe configurations were used to observe signals for 1290 clock cycles, used by the BLE protocol to set the LED blink rate. The write operations were repeated $N_\mathrm{r} = 10$ times at each probe configuration. By replicating these operations at each location, at the end of a scan, the evaluator can generate composite signal maps (Fig. 5.3). Phase I took ~40 hours to complete, primarily due to bottlenecks in the automation of Bluetooth data transfer. Storage requirements for these measurements were ~65 GB. Phase I results are shown in Fig. 5.4 for an *x*-oriented probe. The threshold $F_{N,\mathrm{c}}$ was set to be ~40. Although signals were measured for ~1300 clock cycles, only 2 clock cycles were found

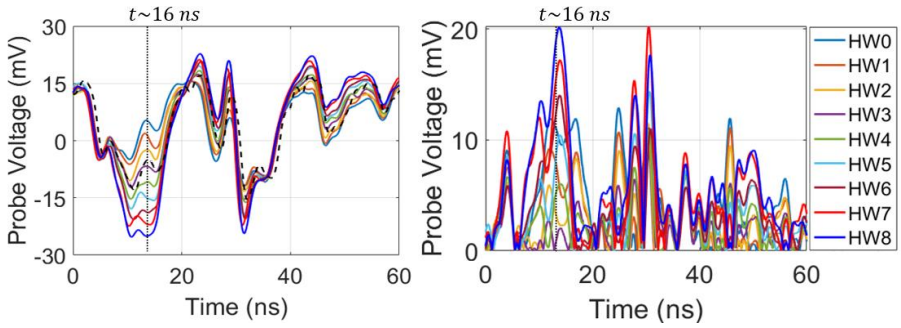


Figure 5.5: Spatial map of $\max_t F_N^{pc,t}$ (left) and time plot of the F-statistic (right) at an optimal configuration (starred). The measurement configuration is suitable for data recovery if the F-statistic is greater than the threshold $F_{N,c}$ (red).

to be optimal time-intervals for data leakage. Further, about 750 effective probe configurations were identified across both orientations, close to the top-right and bottom-left regions of the chip. The optimal probe configuration pc^{opt} was an *x*-oriented probe at location (4, 6, 0.5) mm. Using this optimal configuration, the database of reference signals was constructed for the data values 0x00 (HW 0), 0x01 (HW 1), 0x03 (HW 2), 0x07 (HW 3), 0x0F (HW 4), 0x1F (HW 5), 0x3F (HW 6), 0x7F (HW 7), and 0xFF (HW 8).

A total of 256 test values were sent to the GATT server, covering all 8-bit binary numbers. These values were written by the evaluator's client used in Phase I, as well as the other client. The experiments were repeated 5 times for both clients. Comparing the measured signals to the reference signals, the HW of the test values were identified (Fig. 5.5) with success rates ranging from ~98.1% to ~99.2% for both the evaluator's client and the other client, across the 5 repeated experiments. The high success rate for both clients indicates that the data leak is due to chip processes and computations in the Bluetooth GATT server related to the service and characteristic, which are independent of the client connected to it. This implies that once attackers profile certain characteristics in the GATT

server, they no longer require access to the communication channel, and may monitor fields near the server to recover information from any other connected device.

The proposed method could be extended to isolate configurations related to other BLE services, such as read or notify. Because such attacks do not rely on eavesdropping of the communication channel, they are harder to mitigate using robust authentication methods.

5.3 Data-Dependent EM Profiles as Basis Functions

Previously, EM SCA methods utilized leakage models that abstract sources of leakage such as data-bus transfer, switching in registers, etc. using simplified quantities [4], [21], [54]. These models essentially reduce observed signals for multiple data values into a small number of "signal profiles", where the profiling is based on the properties of data being computed during observation. For example, EM SCA on simple microcontrollers have assumed that emanated fields are linearly dependent on the number of bits with value 1 in data, i.e., the observed fields are dependent on the Hamming Weight (HW) of the data value in data bus or registers [4], [21]. Here, the data with same HWs, such as 0x01 and 0x02 (both have HW 1), are assumed to have the same signal profile. The profiles can be constructed by using a single value conforming to the leakage model, or by averaging signals for a few values. This simplification is sufficient for analysis if variance of observed signals assigned to the same profile is much smaller than the variance among signal profiles. Such profiles can have limited accuracy, however, when representing large number of data values; e.g., in an 8-bit processor, a single profile constructed for data with HW 4 represents 70 different data values. Further, assuming the same leakage models are used for concurrently operating buses and registers, there may be a large difference between aggregated profiled signals and the actual signal [75].

This section introduces a novel approach that improves upon existing leakage models to synthesize near-field signals for arbitrary data values using the fewest possible measurements. These measurements are performed to derive basis functions, which are then linearly combined to generate fields at multiple near-field probe configurations. The method, evaluation and results are summarized next.

5.3.1 Representing Data with Binary Basis Vectors

As Any *n*-bit binary data of interest can be considered an ordered sequence of *n* elements represented as a vector $\mathbf{a} = [a_n, \ldots, a_1]^{\mathrm{T}}$, from most- to least-significant bit, where a_i is an element of the binary field $\mathbb{F}_2$. The vector **a** can also be represented using the canonical basis vectors $\mathbf{e}_i = [e_i^1, e_i^2, \ldots, e_i^n]^{\mathrm{T}}$, where $e_i^j = 1$ if $i = j$ and 0 otherwise, as

$$\mathbf{a} = a_1\mathbf{e}_1 \oplus a_2\mathbf{e}_2 \oplus \ldots \oplus a_n\mathbf{e}_n, \tag{5.3}$$

where $\oplus$ represents the XOR operation. Typically, profiling EM signals for SCA involves using data with different HWs. To link the data representation in (2) to the HW leakage model, a different (non-orthogonal) set of basis vectors $\mathbf{b}_i = [b_i^1, b_i^2, \ldots, b_i^n]^{\mathrm{T}}$ are used, where $b_i^j = 1$ if $i \leq j$ and 0 otherwise; here, each $\mathbf{b}_i$ has a different HW. These basis vectors can be used to express the vector **a** as,

$$\begin{aligned}\mathbf{a} = \mathbf{b}_0 \oplus a_1(\mathbf{b}_1 \oplus \mathbf{b}_0) \oplus a_2(\mathbf{b}_2 \oplus \mathbf{b}_1) \\ \oplus a_3(\mathbf{b}_3 \oplus \mathbf{b}_2) \ \ldots \oplus a_n(\mathbf{b}_n \oplus \mathbf{b}_{n-1})\end{aligned} \tag{5.4}$$

where $\mathbf{b}_0$ is the null vector with all entries valued 0.

Assuming the vector **a** represents the *n*-bit binary data of interest used for computations by the processor, let the signals measured during these computations at each probe configuration pc—combination of location (x, y), orientation o, and height z—and time instance t/frequency f, be denoted by the signal $V_{\mathbf{a}}(x, y, z, o, t/f)$. If linear superposition were applicable, the measured signals could be expressed exactly by linearly

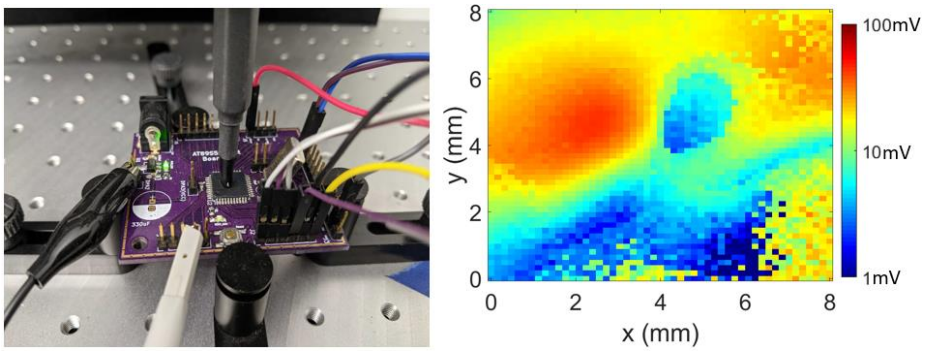


Figure 5.6: Fine-grained EM SCA attack setup [4] (left) probes the chip at multiple locations during chip operations. Measurements can be repeated at multiple probe configurations to generate field maps at a given time instance (right).

combining the signals $V_{\mathbf{b}_0}, V_{\mathbf{b}_1}, \ldots, V_{\mathbf{b}_n}$; these signals are measured at each measurement configuration $(pc, t/f)$ by setting the data of interest to the corresponding basis vector.

5.2.2 Data-Dependent EM Basis Functions

The signal of interest $V_{\mathbf{a}}$ can thus be estimated as,

$$\begin{aligned} V_{\mathbf{a}} &\approx V_{\mathbf{b}_0} + \sum_{i=1}^{n} a_i \left(V_{\mathbf{b}_i} - V_{\mathbf{b}_{i-1}}\right) \\ &= V_{\mathbf{b}_0}(pc, t) + \sum_{i=1}^{n} a_i g_i(pc, t) \end{aligned} \quad , \tag{5.5}$$

where $g_1, g_2, \ldots, g_n$ are the proposed data-dependent basis functions at each measurement configuration. The choice of basis functions in (5.5) reflects the physical significance of the XOR operation and the null vector. While the XOR operation represents both addition and subtraction in the $\mathbb{F}_2$ binary field, this is not extendable to the basis functions that are used to represent the observed fields. Further, both binary values 0 and 1 contribute to the observed fields, implying both must be accounted for while estimating the fields for a given data value. This necessitates the introduction of the null vector in (5.4) and the corresponding EM signal. Next, it is hypothesized that, despite the underlying non-linear nature of the digital blocks, the EM signals measured for binary data can be

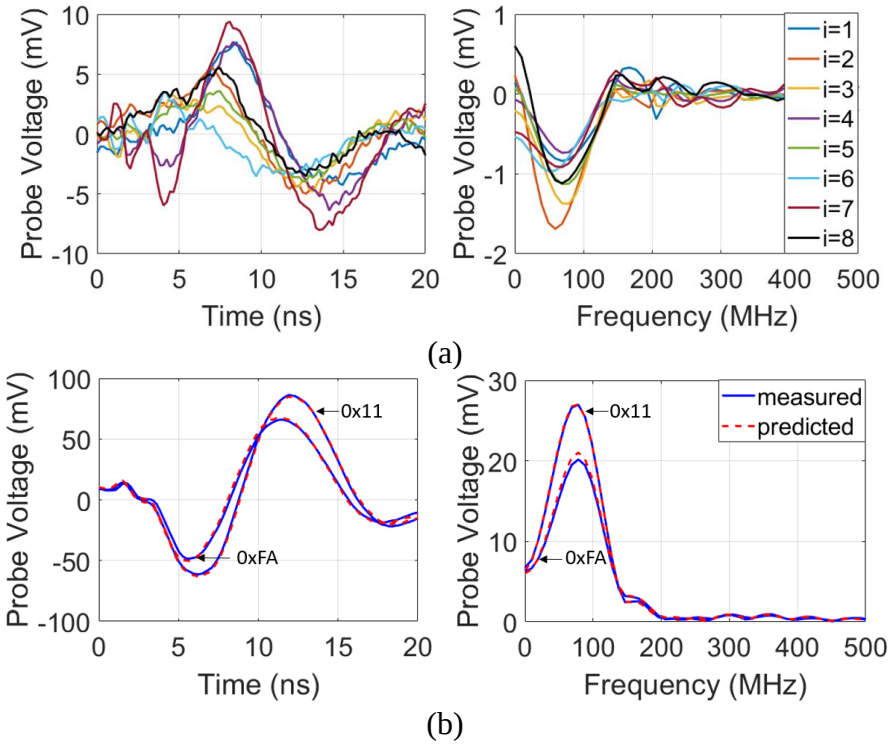


Figure 5.7: (a) Basis functions plotted at the center of the chip in time (left) and frequency (right) for an *x*-oriented probe. The derived functions represent the contribution of each individual bit i. (b) Performance of the model for two arbitrarily chosen data values at the center of the chip.

expressed as a linear combination of signal contributions for each individual bit; as a result, signals for any arbitrary *n*-bit data can be estimated using only *n*+1 measurements.

Although such assumptions can be extended to signals $V_{\mathbf{e}_i}$ for the canonical basis vectors in (5.1), there are differences between a processor's handling of data $\mathbf{e}_i$ and $\mathbf{b}_i$, even for the same instructions. This is because processors perform background computations on such data to set certain flag bits depending on their values; e.g., the parity flag is set as 1 if odd number of bit values are 1. Since vectors $\mathbf{e}_i$ produce similar flag bits, they prove less potent in replicating background processes, and consequently their corresponding signals, for arbitrary data, compared to those derived from HW models.

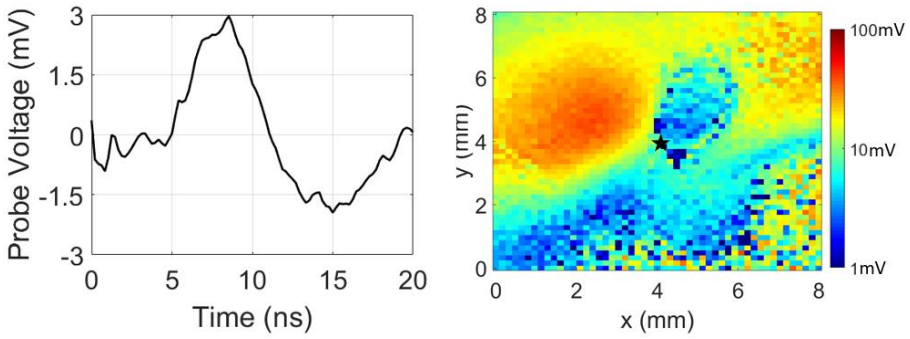


Figure 5.8: Error observed at the center of the chip (starred) for data value 0xFA in time (left), and predicted fields for this data value at the same time instance as the observed field plotted in Fig. 5.6 (t~8 ns).

5.2.3 Measurement Setup and Results

The setup used for measurements in this article is shown in Fig. 5.6 and detailed in [4],[41]. The proposed method was evaluated on an 8-bit Intel i8051 micro-controller; thus, $n = 8$. Near-field scans were performed at $N_l = 51 \times 51$ locations for 2 orientations (*x*- and *y*-orientations) and 1 height (~0.5 mm above the chip surface). Measurements were limited to a time-interval where fields are dependent on output data values [4],[54]. Fields were observed at each probe configuration as the chip performed the MOV instruction, which moves data from program memory to the accumulator register. Results from the experiment are shown in Figs. 5.7-5.8. The predicted and measured fields showed good agreement at several configurations, particularly those identified as optimal to leak output-related information, close to the center of the chip. Error between the predicted and measured signals were primarily observed when fields varied sharply over space (comparing field maps in Fig. 5.6 and Fig. 5.8) and time (comparing signals in Fig. 5.7(b) and the error plot in Fig. 5.8). This is due to small sub-sample time delays between signals $V_{\mathbf{b}_i}$, which introduces small error in (5.5) that aggregates to create the discrepancies. cost.

5.4 SUMMARY

Two approaches were presented to model information leakage and evaluate computing chips. The first approach used signal variances to isolate information leakage on-chip, which is further processed to recover information. This approach is suited to common side-channel inversion problems involving data recovery. The approach was demonstrated to evaluate the vulnerabilities of a BLE server implementation.

The second approach was chosen to improve upon leakage models that are commonly used in SCA attacks. This modelling approach is used to accurately synthesize spatio-temporal near-fields dependent on data computed on chip. The proposed approach "separates" sources of information leaking signals as basis functions, linearly combining them appropriately to predict observed signals. The method can be combined with those existing in literature, such as [3], to improve SCA-based evaluations. The low-cost method requires only $n+1$ measurements to estimate 2^n signals, making it extendable to processors with larger registers. Further, basis functions for multiple processes can potentially be superposed to generate fields for heavily pipelined processors. While presented for a forward problem in this chapter (i.e., predicting fields from few select functions), there is potential to use them as a foundation for side-channel data recovery, which requires further work.

6. Conclusion

This dissertation presented several methods to effectively evaluate the electromagnetic security of embedded computing chips, primarily targeted at addressing their vulnerability to fine-grained EM SCA attacks. The original contributions of the dissertation can be briefly summarized as: (i) comprehensively comparing the effectiveness of different SCA attack modalities, (ii) implementing a multi-stage ANOVA-based measurement protocol to rapidly analyze AES chip vulnerabilities, (iii) developing a hierarchical disassembler to recover execution trace of programs, and (iv) introducing leakage modeling methods and using them to evaluate complex systems.

Chapter 2 presented a systematic comparison between different SCA attack modalities. The effectiveness of coarse- and fine-grained electromagnetic EM SCA attacks, as well as power SCA attacks, were empirically evaluated on implementations of the AES algorithm. While the effectiveness of the attacks were compared based on the marginal costs required to identify the AES key, the acquisition costs of the fine-grained EM SCA attacks were also analyzed thoroughly, by evaluating the AES implementations under various constraints (i.e., threat models). Various search methods were developed, tailored to different threat models, to effectively evaluate the chip's security and compare potency of different SCA attack modalities.

Chapter 3 presented a multi-stage measurement protocol based on the ANOVA F-statistic to rapidly evaluate an AES module's security. The F-statistic and correlation-based attacks were linked by assuming the independence of target, background, and measurement noise signals in side-channel measurements. The work detailed the associated costs of implementing the protocol and compared it with existing alternatives. Additionally the protocol's effectiveness was demonstrated on several baseline and hardened AES modules.

The performance of the protocol could be improved further by adding an additional pre-characterization stage as proposed in [41].

Chapter 4 presented a hierarchical instruction-level disassembler that analyzes leakages via the EM side-channel created during program execution. The disassembler used a database constructed bottom-up at each hierarchical level, using specially designed execution codes. The classification is performed top-down across the hierarchy using binary classification with majority voting. The disassembler was contrasted with several previously proposed works based on the acquisition costs and accuracy of disassembly. Differential signals were introduced as a novel metric to improve side-channel analysis. The disassembler could be improved and extended to more complex processors by introducing additional stages with randomized operands as proposed in [64].

Finally, Chapter 5 presented methods to link information leakage and observed signals, and examples of how they can be used to implement complex systems. The first method assumed independence of target signals and algorithmic and measurement noise, and utilized this to evaluate a BLE server implementation. The second method linked data and fields through spatio-temporal basis functions that are linearly superposed to predict signals from a general-purpose micro-controller. The second approach requires further exploration to utilize it more effectively for forward and inverse problems in EM side-channel analysis.

The work presented in this dissertation opens future research avenues that include security evaluations in complex system-on-chip modules, malware detection, root-cause analysis of side-channel leakage, and improving side-channel signal modelling for simulation-based tools. For instance, systems with multiple integrated components can be analyzed by treating each component as an independent source of EM emanations [54]. Using methods described in this dissertation, potentially in combination with more

powerful statistical tools, sources contributing to information leakage can be isolated within each block/sub-block. Further, basis functions can be built for individual leaking sources that may be superposed to isolate the data-dependent signal components. Deviations in these components can potentially be used for monitoring programs. Using binary basis functions in such cases can also result in a feasible number of measurements\simulations to be performed during analysis, which may otherwise increase as an exponential with the addition of each sub-block if not optimized [73], [74]. The granularity of data recovery from SCA attacks may also benefit from using finer probe resolutions; e.g., there is a possibility of recovering exact data values instead of limiting the attack to recovering data HWs. The extension of these evaluations for different threat models also requires further work. For instance, in restricted white/gray box scenarios, where internal variables of a processor such as counters and flag registers cannot be accessed, evaluators may need to consider these contributions separately as extraneous algorithmic noise, unless they perform sufficient number of measurements to characterize them and correlate them to the executed instructions. For even more restrictive black/red models, it is possible that evaluators may only be limited to basic blocks of code, without the ability to perform instruction-by-instruction analysis. While this dissertation studied measurement protocols for various threat models, it was limited to SCA attacks on AES modules. A similar study of trade-offs between protocols for various threat models, in terms of acquisition cost, and accuracy and granularity of data recovery, may also be performed for general embedded systems.

Bibliography

[1] M. Vuagnoux and S. Pasini, "An improved technique to discover compromising electromagnetic emanations," in *Proc. IEEE Int. Symp. Electromagn. Compat.*, pp. 121-126, July 2010.

[2] A. Zajic and M. Prvulovic, "Experimental demonstration of electro-magnetic information leakage from modern processor-memory systems," *IEEE Trans. Electromagn. Compat.*, vol. 56, no. 4, pp. 885-893, Aug. 2014.

[3] F. Werner *et al.*, "A method for efficient localization of magnetic field sources excited by execution of instructions in a processor," *IEEE Trans. Electromagn. Compat.*, vol. 60, no. 3, pp. 613-622, June 2018.

[4] V. V. Iyer and A. E. Yılmaz, "Using the ANOVA F-statistic to isolate information-revealing near-field measurement configurations for Embedded Systems," in *Proc. IEEE Int. Symp. Electromagn. Compat.,* Aug. 2021.

[5] Y.-I. Hayashi *et al.*, "Efficient evaluation of EM radiation associated with information leakage from cryptographic devices," *IEEE Trans. Electromagn. Compat.*, vol. 55, no. 3, pp. 555–563, Jun. 2013.

[6] V. V. Iyer and A. E. Yilmaz, "Using the ANOVA F-statistic to rapidly identify near-field vulnerabilities of cryptographic modules," in *Proc. IEEE Int. Microw. Symp.,* June 2021.

[7] L. Sauvage, S. Guilley, and Y. Mathieu, "Electromagnetic radiations of fpgas: High spatial resolution cartography and attack on a cryptographic module," *ACM Trans. Reconfigurable Technol. Syst.*, 2009.

[8] M. Alam *et al*. "One&Done: a single-decryption EM-based attack on OpenSSL's constant-time blinded RSA," in *Proc. USENIX*, pp.585-602, Aug 2018.

[9] V. Iyer, M. Wang, J. Kulkarni, and A. Yilmaz, "A systematic evaluation of EM and power side-channel analysis attacks on AES implementations," in *Proc. IEEE ISI*, Nov. 2021.

[10] V. V. Iyer and A. E. Yilmaz, "An adaptive acquisition approach to localize electromagnetic information leakage from cryptographic modules," in *Proc. IEEE Texas Wireless Symp.,* Mar. 2019.

[11] J. Danial, D. Das, S. Ghosh, A. Raychowdhury, and S. Sen, "SCNIFFER: low-cost, automated, efficient electromagnetic side-channel sniffing," *IEEE Access*, vol. 8, pp. 173414-173427, Sep. 2020.

[12] M. Wang *et al.*, "Galvanically isolated, power and electromagnetic side-channel attack resilient secure AES core with integrated charge pump based power management," in *Proc. IEEE CICC*, Apr. 2021.

[13] D. Das, M. Nath, B. Chatterjee, S. Ghosh, and S. Sen, "STELLAR: a generic EM side-channel attack protection through ground-up root-cause analysis," in *Proc. IEEE HOST,* May 2019.

[14] J.-S. Coron and I. Kizhvatov, *An Efficient Method for Random Delay Generation in Embedded Software*, CHES, Berlin: Springer, 2009, vol. 5747.

[15] C. Sui, J. Wu, Y. Shi, Y. Kim, and M. Choi, "Random dynamic voltage scaling design to enhance security of NCL S-box," in *Proc. IEEE MWSCAS,* Aug. 2011.

[16] A. Singh *et al*., "Improved power/EM side-channel attack resistance of 128-bit AES engines with random fast voltage dithering," *IEEE J. Solid-State Circ*., vol. 54, pp. 569–583, Feb. 2019.

[17] M. Yamaguchi *et al.*, "Development of an on-chip micro shielded-loop probe to evaluate performance of magnetic film to protect a crypto-graphic LSI from electromagnetic analysis," in *Proc. IEEE Int. Symp. Electromagn. Compat.*, pp. 103–108, Jul. 2010.

[18] M. Wang *et al.*, "Physical design strategies for mitigating fine-grained electromagnetic side channel attacks," in *Proc. IEEE CICC,* Apr. 2021.

[19] G. Li, V. Iyer, and M. Orshansky, "Securing AES against localized EM attacks through spatial randomization of dataflow," in *Proc. IEEE HOST,* May 2019.

[20] V. Iyer, A. Thimmaiah and A. Yilmaz, "Testing the resilience of cryptographic modules against fine-grained time- and frequency-domain EM side-channel analysis attacks," in *Proc. IEEE ICEAA,* Aug 2021.

[21] V. V. Iyer and A.E. Yilmaz, "An ANOVA method to rapidly assess information leakage near cryptographic modules," *IEEE Trans. Electromagn. Compat.*, vol. 64, no. 4, Aug. 2022.

[22] A. Kumar, C. Scarborough, A. E. Yilmaz, and M. Orshansky, "Efficient simulation of EM side-channel attack resilience," *in Proc. ICCAD,* pp. 123 – 130, Nov. 2017.

[23] D. Fujimoto et al., "On-chip power noise measurements of cryptographic VLSI circuits and interpretation for side-channel analysis," in *Proc. IEEE Int. Symp. on Electromagn. Compat.*, pp. 405–410, Sep. 2013.

[24] G. Becker, ''Test vector leakage assessment (TVLA) methodology in practice,'' in *Proc. Int. Cryptograph. Module Conf.*, Sep 2013.

[25] C. Whitnall and E. Oswald, "A cautionary note regarding the usage of leakage detection tests in security evaluation", Cryptology ePrint Archive, Rep. 2019/703, 2019.

[26] F. Unterstein *et al.*, "Dissecting leakage resilient prfs with multivariate localized em attacks," in *Proc. COSADE,* Jul. 2017.

[27] NIST FIPS Pub. “197: Advanced encryption standard (aes)”. *Federal information processing standards publication*, 197(441):0311, 2001.

[28] I. Buhan, L. Batina, Y. Yarom, and P. Schaumont, “SoK: design tools for side-channel-aware implementations.” Jun. 2021. Available: *ArXiv* abs/2104.08593.

[29] V. V. Iyer, “An adaptive measurement protocol for fine-grained electromagnetic side-channel analysis of cryptographic modules,” M.S. thesis, Univ. of Texas, Austin, Aug. 2019.

[30] K. A. Remley *et al.*, “Millimeter-wave modulated-signal and error-cector-magnitude measurement with uncertainty,” *IEEE Trans. Microw. Theory Tech.*, vol. 63, no. 5, pp. 1710-1720, May 2015.

[31] K. Freiberger, H. Enzinger, and C. Vogel, "A noise power ratio measurement method for accurate estimation of the error vector magnitude," *IEEE Trans. Microw. Theory Tech.*, vol. 65, no. 5, pp. 1632-1645, May 2017.

[32] B. F. Jamroz *et al.*, “Accurate monte carlo uncertainty analysis for multiple measurements of microwave systems,” in *Proc. IEEE MTT-S Int. Microw. Symp.*, Jun. 2016.

[33] ChipWhisperer, Github Repository [online], available: https://github.com/newaetech/chipwhisperer

[34] Information Technology Standard, “Security techniques – testing methods for the mitigation of non-invasive attack classes against cryptographic modules,” *International Organization for Standardization*, Geneva, CH, 2016.

[35] C. Whitnall and E. Oswald, “A critical analysis of ISO 17825 (’Testing Methods for the mitigation of non-invasive attack classes against cryptographic modules’),” in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.*, Dec. 2019, pp. 256–284.

[36] M. Nassar, Y. Souissi, S. Guilley, J.-L.Danger, “RSM: a small and fast countermeasure for AES, secure against 1st and 2nd-order zero-offset SCAs,” in *Proc. DATE*, Mar. 2012, pp.1173-1178.

[37] M.-L. Akkar and C. Giraud, An Implementation of DES and AES Secure against Some Attacks, CHES, ser. LNCS, LNCS, Ed.,. Springer, May 2001, pp. 309–318.

[38] Aoki laboratory, Tohoku University, Japan [online], available: http://www.aoki.ecei.tohoku.ac.jp/crypto/web/cores.html

[39] R. Gilmore, N. Hanley, and M. O'Neill, “Neural network based attack on a masked implementation of AES,” *in Proc. ICCAD*, pp. 106 – 111, Nov. 2015.

[40] T. Kubota *et al.*, “Deep learning side-channel attack against hardware implementations of AES,” *Microprocessors and Microsystems*, vol. 87, Nov. 2021.

[41] V. V. Iyer and A. E. Yilmaz, “Rapid pre-Characterization of fine-grained EM side-channel (in)vulnerability of AES modules,” in *Proc. USNC/URSI Rad. Sci. Meet., July 2022*.

[42] Y. Liu *et al.*, “On Code Execution Tracking via Power Side-Channel,” in *Proc. ACM SIGSAC*, Oct. 2016.

[43] R. Callan, F. Behrang, A. Zajic, M. Prvulovic, and A. Orso, “Zero-overhead profiling via EM emanations,” in *Proc. ACM ISSTA*, July 2016.

[44] T. Eisenbarth, C. Paar, and B. Weghenkel, “Building a side channel based disassembler,” *Trans. comput. sci.*, vol. 10, pp. 78-99, Jan. 2010.

[45] Msgna M., Markantonakis K., Mayes K., *Precise Instruction-Level Side Channel Profiling of Embedded Processors*, Lecture Notes in Computer Science, Cham, Switzerland:Springer 2014, vol. 8434.

[46] J. Park, X. Xu, Y. Jin, D. Forte and M. Tehranipoor, “Power-based Side-Channel Instruction-level Disassembler,” in *Proc. ACM/IEEE DAC*, June 2018.

[47] V. M. Vaidyan and A. Tyagi, “Instruction level disassembly through electromagnetic side-chanel: machine learning classification approach with reduced combinatorial complexity,” in *Proc. ACM SPML*, Oct. 2020.

[48] D. Strobel, F. Bache, D. Oswald, F. Schellenberg, and C. Paar,“ Scandalee: a side-channel-based disassembler using local electromagnetic emanations,” in *Proc. DATE*, Mar. 2015.

[49] V. Cristiani, , M. Lecomte,, T. Hiscock, “A Bit-Level Approach to Side Channel Based Disassembling” in *Proc. CARDIS*, Nov. 2019.

[50] E. Peeters, F.-X. Standaert, and J.-J. Quisquater, “Power and electromagnetic analysis: improved model, consequences and comparisons”, *Integr. VLSI J.*, vol. 40, pp. 52-60, Jan. 2007.

[51] J. Wharton, “An Introduction to the Intel MCS-51 Single-Chip Microcomputer Family,” Intel Corporation, Application Note AP-69, May 1980.

[52] ATMEL, “8-bit microcontroller with 4K bytes in-system programmable flash,” AT89S51 datasheet, June 2008.

[53] Dalton Project, “Benchmark applications for synthesizeable VHDL model”, i8051 Benchmarks. [Online] Available:

http://www.ann.ece.ufl.edu/i8051/i8051benchmarks/index.html

[54] A. Thimmaiah, V. V. Iyer, A. Gerstlauer, M. Orshansky, “High-level simulation of embedded software vulnerabilities to EM side-channel attacks,” in *Proc. SAMOS*, July 2022.

[55] J. Maillard, T. Hiscock, M. Lecomte and C. Clavier, “Towards fine-grained side-channel instruction disassembly on a system-on-chip,” in *proceedings of the 25th Euromicro Conference on Digital System Design*, pp. 472-479, 2022.

[56] A. Nazari, N. Sehatbakhsh, M. Alam, A. Zajic, and M. Prvulovic, “EDDIE: EM-Based Detection of Deviations in Program Execution,” in *Proc. ISCA*, pp. 333–346, 2017. https://doi.org/10.1145/3079856.3080223

[57] N. Carlini and D. Wagner, “ROP is still dangerous: breaking modern defenses,” in Proc. USENIX, Aug. 2014.

[58] L. Davi, A.-R. Sadeghi, and M. Winandy. “ROP defender: a detection tool to defend against return-oriented programming attacks,” in *Proc. ACM ICSS*, pp. 40–51, 2011.

[59] T. Bletsch, X. Jiang, V. W. Freeh, and Z. Liang, “Jump-oriented programming: a new class of code-reuse attack,” in *Proc. ASIACCS*, 30–40, 2011. https://doi.org/10.1145/1966913.1966919

[60] N. Carlini, A. Barresi, M. Payer, D. Wagner, and T. R. Gross, “Control-flow bending: on the effectiveness of control-flow integrity,” in *Proc. USENIX* 161–176, 2015.

[61] J. Demme, R. Martin, A. Waksman, and S. Sethumadhavan, “Side-channel vulnerability factor: a metric for measuring information leakage,” in *Proc. ISCA*. IEEE Computer Society, USA, 106–117, 2012.

[62] R.Callan, A. Zajić, and M. Prvulovic. “A Practical Methodology for Measuring the Side-Channel Signal Available to the Attacker for Instruction-Level Events,” in *Proc. IEEE/ACM International Symposium on Microarchitecture*, IEEE Computer Society, 242–254, 2014. https://doi.org/10.1109/MICRO.2014.39

[63] H. Gao, Q. Li, Y. Zhu, Y. Liu, “Code-controlled hardware trojan horse”, in *Proc. ICICA*, Communications in Computer and Information Science, vol 308. Springer,2012. doi: 10.1007/978-3-642-34041-3_26.

[64] J. Van Geest, and I. Buhan, “A side-channel-based disassembler for the ARM-Cortex M0,” in *Proc. Applied Cryptography and Network Security Workshops: ACNS*, June 2022.

[65] D. Agrawal, B. Archambeult, J.R. Rao, and Rohatgi, P., “The EM side-channel (s): attacks and assessment methodologies,” *in Proc. CHES*, Aug. 2002.

[66] R. Wang, H. Wang, and E. Dubrova, “Far field EM side-channel attack on AES using deep learning,” in *Proc. ACM ASHES*, Nov. 2020.

[67] G. Camurati *et al.*, “Screaming channels: when electromagnetic side channels meet radio transceivers,” in *Proc. ACM SIGSAC CCS*, Oct. 2018.

[68] A. S. Khader and D. Lai, “Preventing man-in-the-middle attack in Diffie-Hellman key exchange protocol,” in *Proc. IEEE ICT*, Apr. 2015.

[69] V.V. Iyer, A. Thimmiah, M. Orshansky, A. Gerstlauer, A. Yilmaz, “A hierarchical classification method for high-accuracy instruction disassembly with near-field EM measurements,” in *ACM TECS*.

[70] M. Woolley, “Bluetooth core specification version 5.0,” Bluetooth SIG.

[71] Renesas Electronics, “Renesas RA4W1 Group Datasheet”, R01DS0359EJ0100 Rev.1.00, Mar. 2021.

[72] Renesas Electronics, “Renesas RA4W1 Group BLE Sample Application”, R01AN5402EJ0107 Rev.1.07, Oct. 2022.

[73] C. Thuillet, P. Andouard and O. Ly, “A smart card power analysis simulator," in *Proc. IEEE ICCSE*, Aug. 2009.

[74] N. Veshchikov, “SILK: high level of abstraction leakage simulator for side channel analysis,” in *Proc. ACM PPREW-4* Dec. 2014.

[75] V.V. Iyer and A.E. Yilmaz, “Estimating near-field signals emanated by embedded systems using data-dependent EM profiles as basis functions,” in *Proc. USNC-URSI Rad. Sci. Meet*, July 2023.

[76] V.V. Iyer and A. Yilmaz, “EM side-channel analysis of data leakage near embedded bluetooth low energy modules,” in *Proc. WAMICON*, April 2023.