Location-Based Leakages: New Directions in Modeling and Exploiting

Christos Andrikos
and Giorgos Rassias
National Technical University of Athens
Email: candrikos@cslab.ece.ntua.gr
Email: grassias@cslab.ece.ntua.gr

Liran Lerman
Université libre de Bruxelles
Email: llerman@ulb.ac.be

Kostas Papagiannopoulos
and Lejla Batina
Radboud University Nijmegen
Email: kostaspap88@gmail.com
Email: lejla@cs.ru.nl

***Abstract*—Near-field microprobes have the capability to isolate small regions of a chip surface and enable measurements with high spatial resolution. The capability of distinguishing such small regions gives rise to the location-based side-channel attacks, which exploit the spatial dependencies of cryptographic algorithms in order to recover the secret key. This work discusses our preliminary results and research in the field of location-based leakages and consists of three parts. First, we provide a simple spatial model that partially captures the effect of location-based leakages. Second, we perform the first successful location-based attack on the SRAM of a modern ARM Cortex-M4, using standard techniques such as difference of means and multivariate templates. Third, we expand towards the application of convolutional neural networks as classifiers that can distinguish small regions of SRAM.**

I. Introduction

Side-channel analysis (SCA) allows adversaries to recover sensitive data, by observing and analyzing the physical characteristics and emanations of a cryptographic implementation [1]. Usually, the physical observables allow the adversary to infer key-dependent intermediate values of a cipher, by using the well-established Hamming weight or distance models. A less common form of side-channel leakage that arises in many practical scenarios is location-based leakage. Such leakage stems from the fact that different e.g. registers, memory regions, buses or other chip components exhibit identifiable leakage when accessed or manipulated. If there exists any dependence between the secret key and the activated component, then a side-channel adversary can exploit it to his advantage.

The works of Sugawara et al. [2] demonstrate the presence of such address dependencies in an ASIC scenario. Likewise, the works of Heyszl et al. [3] and Sprecht et al. [4] have exploited similar spatial dependencies on a decapsulated FPGA using near-field microprobes. The location-based leakage can be useful in a plethora of scenarios, enabling us to distinguish either small or large components. For instance, finding which ECC register is used by public-key algorithms such as double-and-always-add can result in direct key recovery [3]. Similarly, recovering the exact address accessed during the AES Sbox lookup can reveal the key, i.e. even photonic emission analysis [5] can be considered as a sub-case of location-based leakage exploitation. Last, side-channel countermeasures such as RSM [6] rely on rotating lookup tables to mask the data. Identifying which lookup table is currently under use can simplify the side-channel analysis. For the sake of clarity, we need to distinguish between location-based leakage and localized leakage. Location-based leakage arises when the location of a component is assisting towards key recovery. On the contrary, we observe localized EM leakage when an intermediate value of the algorithm leaks in a specific chip region while depending on the input and key. For example, exploiting the exact address of an Sbox lookup implies location-based leakage, whereas attacking the Sbox output on a specific chip region implies localized leakage. Localized EM leakage is conceptually similar to power leakage and its potential upside is the fact that localization can improve the signal-to-noise ratio, as seen e.g. in the work of Unterstein et al. [7].

Contribution. This work presents several preliminary results in the field on location-based leakage modeling and exploitation. Analytically, we provide a simple spatial model that partially captures the effect of location-based leakages. In addition, we perform the first successful location-based attack on the SRAM of a modern ARM Cortex-M4, using difference of means, as well as multivariate templates. Last, we apply convolutional neural networks as classifiers in order to distinguish small regions of the SRAM. Section II describes the experimental setup and performs a simple analysis. Section III puts forward the spatial model and identifies future directions in the context of MI-based analysis. Sections IV and V discuss preliminary results with template attacks and convolutional neural networks respectively. We conclude in Section VI.

Notation. Throughout the text, capital letters denote random variables and small case letter denote instances of random variables or constant values. Notation $Uniform(\{a,b\})$, $Bernoulli(p)$, $Binomial(n,p)$ and $\mathcal{N}(\mu,\sigma^2)$ denote random variables with uniform, Bernoulli, binomial and normal probability distributions respectively. Parameter p denotes the probability of Bernoulli/binomial trials and μ,σ^2 denote the mean and variance of the normal distribution. The set $\{a,b\}$ denotes that the uniform distribution can receive value a or b equiprobably.

978-1-5386-3437-0/17/$31.00 ©2017 IEEE

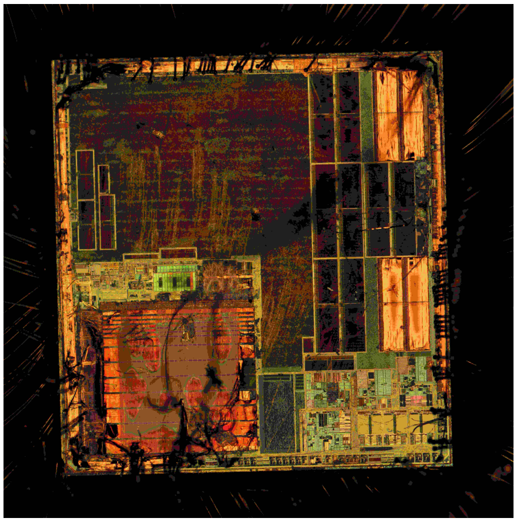

Fig. 1. The decapsulated surface of the device-under-test.

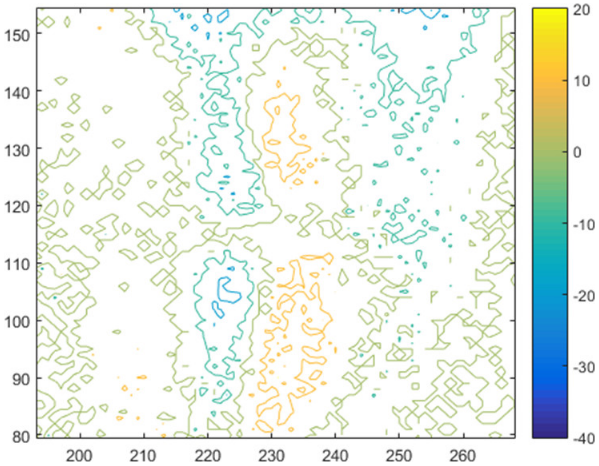

Fig. 2. Distinguishing two 8KByte regions of the SRAM. Yellow region indicates stronger leakage from class 1 and blue region indicates stronger leakage from class 2. The differences below the significance threshold value (4.5) are excluded from the graph.

II. Experimental Setup & T-Test Analysis

The main goal of our experimental evaluation is to examine whether it is possible to detect the access to different SRAM regions in a modern ARM-based device, i.e. examine its susceptibility to location-based attacks. Our measurement setup consists of a decapsulated Pinata device[1], using an ARM Cortex-M4 processor. The decapsulated chip surface (roughly 6 mm^2) is scanned using an ICR HH 100-27 Langer microprobe[2] with diameter of 100 μm. The scan is performed on a rectangular grid of dimension 300, resulting in 300 × 300 measurement spots. The near-field probe is moved over the surface with the assistance of an XYZ-table with positioning accuracy of 50 μm. At every position of the scan grid, a single measurement was performed, using sampling rate of 1 Gsample/sec and resulting in 170k samples. Due to the complex and non-homogeneous nature of a modern chip, several type of EM emissions are present on the surface, most of which are unrelated to the SRAM location. In this particular case study, the signals of interest had amplitude of roughly 70mV, so we have set the oscilloscope voltage range accordingly. In addition, several device peripherals (such as USB communication) have been disabled in order to reduce interference. The decapsulated surface where the scan is performed is visible in Figure 1.

In order to produce location-dependent leakage, we perform sequential accesses to a continuous region of 16KBytes in the SRAM by storing the same value to all positions. The word size of this ARM architecture is 32 bits, i.e. we have accessed to 4096 words in memory. We opted to access the SRAM using ARM assembly instead of a high-level language in order to avoid compiler-induced pitfalls such as code optimizations and removal of redundant code.

Preliminary Results. The initial scan measurements were analyzed using a simple difference-of-means test. To demonstrate the presence of location-based leakage, we partitioned every trace (170k samples) to two classes. The first class contains SRAM accesses from the beginning of the memory until word no. 2047 and the second class contains SRAM accesses from word 2048 until word 4096. For every grid position, we averaged the samples of class 1 and class 2 producing $\bar{t}_1$ and $\bar{t}_2$ respectively and computed the difference $\bar{t}_1 - \bar{t}_2$. Continuing, we performed a Welch t-test with significance level of 0.1% in order to determine if location-based leakage is present. The results are visible in Figure 2, which is focusing in a specific sub-region of the chip surface that exhibits high difference.
In Figure 2 we can observe that location dependencies do exist and can even be distinguished by visual inspection. In addition, we observe that the location dependencies demonstrate strong spatial features. That is we can see two regions at close proximity (yellow region and blue region) where the yellow region shows positive difference between class 1 and 2, while the blue region shows negative difference between class 1 and 2.

III. A Spatial Model for Location Leakages

The prevalence of leakage in intermediate values of cryptographic implementations has led to concentrated efforts towards its modeling. Side-channel research has proposed a multitude of models, starting from the independent noise model [8], which expresses the leakage as the sum of a deterministic part (data) and a random part (noise). In the same direction, modeling has attempted to analyze the noise source, distinguishing between algorithmic and electrical [9], [10], while measuring its effectiveness as a countermeasure. Custom models can also capture certain leakage features that are often ignored. For instance, multivariate models [11], [12]

[1] https://tinyurl.com/lduhaaq
[2] https://tinyurl.com/mcd3ntp

are capable of pinpointing joint leakage effects, in order to enhance the exploitation phase. Similarly, models can encompass electrical & electronic effects directly, managing to bypass protection mechanisms [13].

Unlike the well-established data-dependent models, its location-based counterpart remains less explored. The main reason is the semi-invasive nature of location attacks (often requiring chemical decapsulation) and the lengthy measurement procedures involved. Still, we maintain that such attacks are increasingly relevant due to the fairly small cost, the widespread protection against data-dependent leakages [14], [15], as well as the increasing complexity of electronic circuitry that makes power-oriented attacks harder [16].

Thus, this section puts forward a theoretical model that describes the location leakages observed on a chip surface that are caused by the switching activity of circuit regions. The model can be viewed as the extension of the independent noise model to the spatial domain and is based on the following definitions and assumptions.

Definition. We define a location-oriented side-channel experiment $\mathcal{E}$ as the parameter tuple $\mathcal{E}(s, r, g, \mathbf{c}, \mathbf{v})$. Much like the previous section, the experiment consists of a probe scan over the chip surface in order to distinguish between different regions. The parameter s denotes the area of the chip surface on which we perform measurements (e.g. the whole chip die) and parameter r denotes the area of the measuring probe that we use in our experiments. Typically, we require r to be smaller than s in order to be able to isolate different components (e.g. SRAM regions) on the surface. Continuing, parameter g denotes the measurement grid dimensions, i.e. it specifies the measurement resolution of a uniform rectangular array of antennas [17]. Finally, the vector parameters $\mathbf{c}, \mathbf{v}$ contain the information about the surface components that we try to distinguish. Analytically, if we want to distinguish between n_c components on the chip surface, $\mathbf{c} = [c_1, c_2, \ldots, c_{n_c}]$, where $c_i = 1$ if component i is active at this time and $c_i = 0$ if it is inactive. In addition, we need to specify $\mathbf{v} = [(\mathbf{p}_1, a_1), (\mathbf{p}_2, a_2), \ldots, (\mathbf{p}_{n_c}, a_{n_c})]$. The vector $\mathbf{p}_i$ denotes the position of component i on the chip surface and the parameter a_i denotes its area. For simplicity, we assume the geometry of the surface, probe and components to be square, yet we note that the model can be extended to different geometrical shapes in a straightforward manner. Note also that in our attack scenario, only one out of n_c components is active at a given point in time, thus the system constitutes of n_c possible states. We define state $S = s_i, \forall i = 1, \ldots, n_c$ as the following tuple: $s_i \equiv [c_i = 1, c_j = 0 \; \forall j \neq i]$.

Experiment $\mathcal{E}(25, 2.9, 2, [([0.6, 1.5], 0.8), ([1.6, 4.1], 2.8)])$ is visible in Figure 3, where all parameters are assumed to be in generic units u and square units u^2 accordingly. The surface s, probe r, areas a_1 and a_2 are respectively 25, 2.9, 0.8 and 2.8 u^2. The position of components c_1 and c_2 is $[0.6, 1.5]$ and $[1.6, 4.1]$ respectively. The dimension g of the grid is 2, resulting in a 2×2 scan grid. The system has only two possible states, namely state $s_1 = [c_1 =$'on', $c_2 =$'off'$]$ and state $s_2 = [c_2 =$ 'on', $c_1 =$ 'off'$]$.

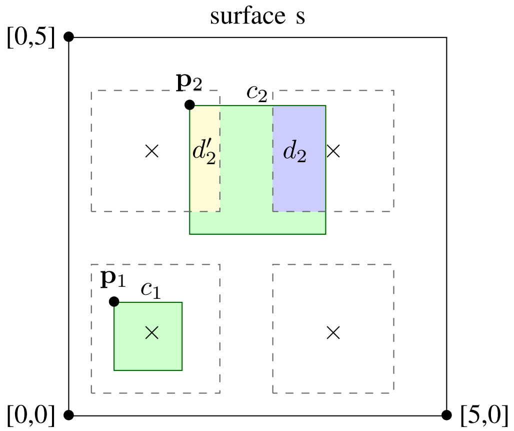

Fig. 3. Sample experiment $\mathcal{E}$. Dashed-lines denote the probe area r. Points $\mathbf{p}_1, \mathbf{p}_2$ show the position of components c_1, c_2, whose area is highlighted by green. The × spots show the measurement points of the scan grid.

Assumption 1: Independent Noise. For a given experiment $\mathcal{E}$, the leakage $L_{x,y}$ at any specified measurement position $[x, y]$ in the grid consists of a deterministic part $l^{det}_{x,y}$, an algorithmic noise part N^{algo} and an electrical noise part N^{el}, thus:

$$L_{x,y} = l^{det}_{x,y} + N^{algo} + N^{el} \tag{1}$$

Assumption 2: Deterministic Leakage. The deterministic part of the leakage $l^{det}_{x,y}$ at position $[x, y]$ is caused by the activation (switching behavior) of any component c_i that is captured by the probe at this grid position. Thus, it depends directly on the current state S. We assume the deterministic leakage to be proportional to the area of the active component located underneath the probe surface. In the following analysis we use the identity leakage function w.r.t. the component area.

$$l^{det}_{x,y}|s_i = \begin{cases} 0, \text{ if } c_i \text{ is not captured at [x,y]} \\ d_i, \; 0 < d_i < a_i \text{ if } c_i \text{ is partially captured at [x,y]} \\ a_i \text{ if } c_i \text{ is fully captured at [x,y]} \end{cases} \tag{2}$$

Figure 3 demonstrates a component c_1 that is fully captured by the probe, i.e. $l^{det}|s_1 = c_1$ on the bottom-left grid spot (×) and zero on all other grid spots. On the contrary, component c_2 is partially captured. Thus, $l^{det}|s_2 = d_2$ on the top-left grid spot (yellow area), $l^{det}|s_2 = d_2'$ on the top-right grid-spot (blue area) and zero elsewhere.

Assumption 2 is motivated by the results obtained in our ARM-based experimental setup (Figure 2) as well as the results of the FPGA-based setups of Heyszl et al. [3] and Sprecht et al. [4]. In several scenarios, the experimental results demonstrate strong spatial features, which tend to be proportional to the component size. Still, we note that the model has several practical limitations. The switching activity in the regions may not be the only source of location leakage, e.g. different regions could also be identified from address-dependent parts of the circuit. Units like the ALU, multiplexers and memory buses often store and manipulate addresses directly, generating data-dependent leakage, where the secret data is the address being accessed. As a result, the exact nature and cause of location-based leakage remains open to investigation.

Assumption 3: Algorithmic & Electrical Noise. We employ the common assumption that the electrical noise N^{el} follows a normal distribution with zero mean and variance σ_{el}^2, i.e. $N^{el} \sim \mathcal{N}(0, \sigma_{el}^2)$. The variance σ_{el}^2 is related to the specific device-under-test and measurement apparatus that we use (probe, oscilloscope, amplifier etc.).

The algorithmic noise in our model is caused by components that, like components c_i, leak underneath the probe in measurement spot on the scan grid. However, unlike components c_i, they exhibit uniformly random switching activity (equiprobable 'on' and 'off') that is independent of the state S. If n_a such components, each with area b_i, $i = 1, \ldots, n_a$, are located under the probe, then we assume their leakage to be again proportional to the captured area b_i . The leakage of these independent, noise-generating component is denoted by $L_i,\ i = 1, \ldots n_a$. Thus, N^{algo} constitutes of the following sum, using again the identity leakage function.

$$N^{algo} = \sum_{i=1}^{n_a} L_i, \text{ where } L_i \sim Uniform(\{0, b_i\}) \tag{3}$$

The algorithmic noise is highly dependent on the device-under-test, i.e. we could potentially encounter cases where there is little or no random switching activity around the critical (targeted) components c_i, or we may face tightly packed implementations that induce such noise. Note that the larger the probe area r, the more likely we are to introduce such components.

Since countermeasure designers opt often for algorithmic noise countermeasures, we investigate N^{algo} for a tightly packed circuit that contains a large number of randomly switching components in order to hide the activated component. We assume every noise-generating component to have area $b \simeq d$, where d is the area of the activated component Note that since we assume large n_a, both the noise-generating components as well as the critical component are small w.r.t. the probe size, i.e. $d \ll r$. In a tightly packed circuit, $n_a \simeq r/d$, i.e. the probe area r contains roughly r/d randomly switching components. The following formula approximates N^{algo} for the particular case we described.

$$\begin{gathered} N^{algo} = \sum_{i=1}^{n_a} L_i = d \cdot \sum_{i=1}^{n_a} B_i = d \cdot A, \text{ where} \\ B_i \sim Bernoulli(0.5) \text{ and } A \sim Binomial(n_a, 0.5) \\ \text{Thus },N^{algo} \xrightarrow[\text{Theorem}]{\text{Central Limit}} \mathcal{N}(d \cdot n_a/2, d \cdot n_a/4) \end{gathered} \tag{4}$$

It holds that $Var[N^{algo}] = d \cdot n_a/4 = r/4$. Thus, for the tightly-packed, small-component scenario we have established a direct link between the probe area r and the level of algorithmic noise, demonstrating how increasing the probe area enhances the algorithmic noise.

Concluding the model discussion, we note that an adversary performing experiment $\mathcal{E}$ for all possible states $s_1, \ldots s_{n_c}$, can generate a multidimensional leakage vector $\mathbf{L}|s_i$ $\forall i = 1, \ldots, n_c$. Subsequently, he can use this g^2-dimensional vectors in order to distinguish between the different states.

Future Directions: Having established a spatial model for location leakage, we aim to proceed towards the theoretical evaluation of the security level. Specifically, we aim to analyze the location-based leakages via the mutual information metric, suggested by Standaert et al. [10] i.e., to evaluate the leakage in the worst-case adversarial scenario. The MI metric can be computed using the following formula.

$$\begin{gathered} MI(S; \mathbf{L}) = H[S] + \sum_{s \in \mathcal{S}} Pr[s] \cdot \int_{\mathbf{l} \in \mathcal{L}^{g^2}} Pr[\mathbf{l}|s] \cdot log_2 Pr[s|\mathbf{l}] \ \ d\mathbf{l} \\ \text{where } Pr[s|\mathbf{l}] = \frac{Pr[\mathbf{l}|s]}{\sum_{s^* \in \mathcal{S}} Pr[\mathbf{l}|s^*]} \end{gathered} \tag{5}$$

We aim to examine the effects of all the parameters involved in the location-oriented side-channel experiment $\mathcal{E}$ under the MI framework in order to pinpoint potential countermeasures that can hinder the adversary.

IV. Template Attacks

Moving towards exploitation, a common analysis option in the field of side-channel analysis is the template attack [18]. The adversary models a priori the leakage (template construction) in a controlled device by estimating the parameters of a multivariate normal distribution. Continuing, the adversary attacks a target device, trying to identify the secret key/state/region via the maximum likelihood approach (template matching).

In the case study on the SRAM of ARM Cortex-M4, the template attacks focus on identifying which region of the SRAM is being accessed. The leakage vector is $\mathbf{L}|s_i$ and its dimensionality is usually large even for modest values of the grid dimension g. Thus, we employ dimensionality reduction techniques (based on the correlation heuristic) so as to detect points of interest (POIs) in the 300 × 300 grid (space) and in the 170k samples (time), aiming to train the statistical model effectively and efficiently. In addition, when performing template matching, we combine several leakage measurements from the test set together, in order to reduce the noise and improve our detection capabilities[3]. Last, we note that template attacks are particularly demanding w.r.t. computational resources, particularly when they involve

[3] Whether this constitutes an option depends on the situation. If any sort of randomization such as masking or re-keying is present in the device then the adversary is limited in the number of test traces that he can combine.

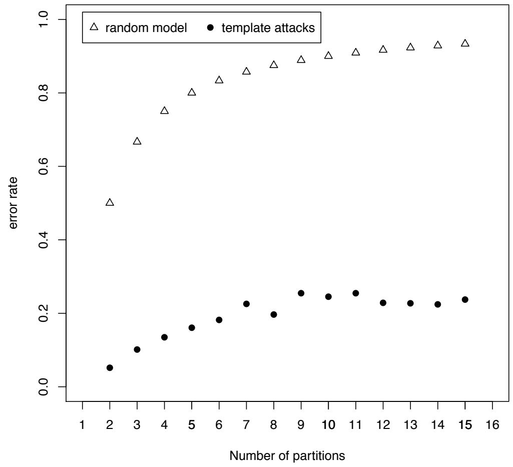

Fig. 4. The error rate of the template-based classifier as we increase the number of components n_c and decrease their area a_i.

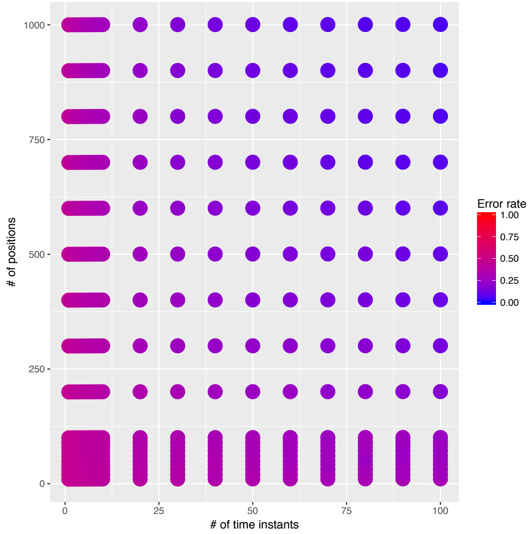

Fig. 5. The error rate of the template-based classifier as a function of the number of POIs selected in the spatial and the temporal domain.

high-dimensional distributions. To tackle this computational problem, we opt for the improved template formulas proposed by Choudary et al. [19] that assume a pooled covariance matrix.

Preliminary Results. In detail, we gradually partition the 16kBytes of SRAM into classes, built the corresponding template for each class and perform matching. Initially the SRAM is partitioned in two regions, i.e. we attempt to distinguish the initial 8KByte region against the following 8 KByte region (as seen in Section II). The analysis continues in the same fashion, partitioning the SRAM into gradually smaller regions (3,4,. . . ,15), i.e. we examine the distinguishing capability of the adversary, as the number of components n_c increases and their respective areas a_i decrease. This preliminary template analysis demonstrates how the error rate of our classification interacts with these two parameters of $\mathcal{E}$. The preliminary results are visible in Figure 4, showing that 15 regions of roughly 1KBytes in size remain sufficiently distinguishable with error rate of 20%. Figure 5 showcases how the number of POIs affects the error rate of a certain class.

Future Directions. What we are particularly interested in is the behavior (error rate) of the template-based classification as the problem becomes increasingly harder. For instance, we aim to apply templating in order to e.g. find how small do the memory regions need to be for the distinguisher to fail. Likewise, we aim to answer similar questions w.r.t. the grid, whose dimension g constitutes a particularly limiting factor during the trace acquisition procedure. Finally, we want to discover if SRAM regions that are positioned closely together are harder to distinguish, i.e. whether we can identify a countermeasure relying on physical proximity.

V. Neural Network Analysis

Despite the fact that the multivariate normal leakage assumption is fairly realistic in the side channel context, applying distribution-agnostic techniques appears to be another rational approach. Over past few years, there has been a resurgence of interest in Deep Learning techniques, powered by rapid hardware evolvement. In this context, we also employ a deep learning approach to reveal the secret state S by invoking a convolutional neural network (CNN). Originally aimed to incarnate computer vision, CNNs perform well on tasks that exploit potential spatial correlation on raw input data, i.e. elements (such as measurement regions) that are closer together are more closely related than distant ones. In this manner we designed and implemented a CNN to tackle the problem of classifying the “images” resulting from the ARM Cortex-M4 experiment.

CNNs are defined as a mathematical workflow composed of some random combination of (1) convolutional, (2) nonlinear, (3) pooling (downsampling), and (4) fully connected layers. The key difference between CNNs and typical deep forward networks is that in CNNs, dimensionality reduction is an immediate result of the training process. Considering two or more inputs of high dimensionality, CNNs can excel in the task of classification as they are able to learn the voting weights of the fully connected layers, as well as the features (filters) on the convolutional ones through Back Propagation.

Figure 6 depicts the 4 basic layers the CNN that we applied in our case study of SRAM on ARM Cortex-M4:

1) Convolutional layers: during the forward phase input data are convoluted with some filters (features) to produce feature maps depicting where the features are actually located. During the backward phase, filter weights are readjusted (learned) in a manner of minimizing a loss function.

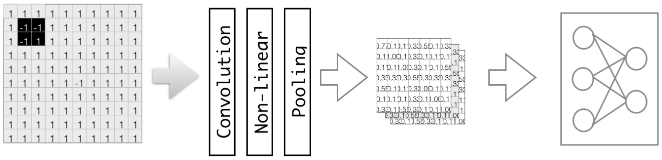

Fig. 6. CNN architecture.

2) Pooling layers: the feature maps are downsized to reduce computational complexity and increase model robustness
3) Nonlinear layers: Nonlinear functions are invoked to provide normalization. Thus scores generated by previous layers are converted to a probability distribution over the classes or are considered zero if they are negative.
4) Fully connected layers: usually the final layers of the entire stack; a classic multi-layer perceptron that implements a voting schema during the forward phase, while during the backward phase the weights are readjusted (learned) by minimizing a loss function.

As mentioned, the location-oriented experiment $\mathcal{E}$ generated 170k measurements of the magnetic field of each of the 300 × 300 points on the chip. We consider this equivalent to generating 170k images of size of 300 × 300 pixels, similar to the ones resulting from Magnetic Resonance Imaging. Thus, our intuition leads us to define a contextual image classification problem focusing on the relationship of the nearby pixels (neighborhood). Given the extensive effort required to train the CNN, we decided to attempt distinguishing small SRAM regions. We partitioned the trace in two consecutive regions (class 1 and class 2) of 1000 images each, i.e. every class roughly corresponds to the activation of 100 bytes of the SRAM.

Preliminary Results. The CNN that was tested follows the architectural principles depicted on Figure 6 and aims to infer the active component on the chip. The first two layers of the proposed architecture are convolutional. In this manner, input is filtered twice to produce some more abstract feature maps. Considering the physical size of the probe relatively to the area of the active SRAM components, we ended up utilizing 5x5 filters for the first and 7x7 for the second convolutional layer. The resulting 64 feature maps are then fed to a non-linear layer consisting of Rectifiers. The latter normalizes the feature maps to prevent numerical computations resulting in inappropriate states, such as overflow or underflow, that can affect the overall computing. The normalized feature maps are then fed to a max pooling layer to be downsized to one fourth of each dimension in favor of computing efficiency. Finally the output of max pooling layer is forwarded to a fully connected multilayer perceptron (MLP) composed of one input, one output and a single hidden layer. Our model was implemented and trained on Tensorflow, an open sourced library with improved computational capabilities introduced by Google. The training dataset was divided in two classes and the training samples were chosen evenly from each class in a random way (uniform distribution).

Using the CNN-based classifier, we conclude that the introduced model performs the classification task by scoring 91.3% in distinguishing between 2 regions of 100 bytes.

Future Directions The CNN model is not completely tested for its capacity or for any overfitting/undefitting effects and increasing the parameter n_c results in higher error rates. Our future direction will focus on fully harnessing the potential of CNNs in the context of location-based leakages and performing a direct comparison between CNNs and template attacks. Last, we also aim to tackle the computationally intensive training that is required by CNNs by employing parallel GPU computations.

VI. Conclusion

The current work takes the first steps towards the in-depth modeling and exploitation of location-based leakages. We have demonstrated preliminary work showing how such effects are exploitable by the adversary via template attacks and convolutional neural networks. This ongoing research will continue by analyzing all relevant experimental parameters of location-oriented side-channel experiments under the MI framework, as well as using real-world case studies. Our long-term goal is to solidify the exploitation techniques, study their interaction with the experimental parameters and finally, move towards the direction of countermeasures against location-based attacks.

Acknowledgment

Removed for submission.

References

[1] P. C. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, 1999, pp. 388–397.

[2] T. Sugawara, D. Suzuki, M. Saeki, M. Shiozaki, and T. Fujino, "On measurable side-channel leaks inside ASIC design primitives," *J. Cryptographic Engineering*, vol. 4, no. 1, pp. 59–73, 2014.

[3] J. Heyszl, S. Mangard, B. Heinz, F. Stumpf, and G. Sigl, "Localized electromagnetic analysis of cryptographic implementations," in *Topics in Cryptology - CT-RSA 2012 - The Cryptographers' Track at the RSA Conference 2012, San Francisco, CA, USA, February 27 - March 2, 2012. Proceedings*, 2012, pp. 231–244.

[4] R. Specht, J. Heyszl, and G. Sigl, "Investigating measurement methods for high-resolution electromagnetic field side-channel analysis," in *2014 International Symposium on Integrated Circuits (ISIC), Singapore, December 10-12, 2014*, 2014, pp. 21–24. [Online]. Available: http://dx.doi.org/10.1109/ISICIR.2014.7029532

[5] A. Schlösser, D. Nedospasov, J. Krämer, S. Orlic, and J.-P. Seifert, *Simple Photonic Emission Analysis of AES*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 41–57.

[6] M. Nassar, Y. Souissi, S. Guilley, and J. L. Danger, "Rsm: A small and fast countermeasure for aes, secure against 1st and 2nd-order zero-offset scas," in *2012 Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2012, pp. 1173–1178.

[7] F. Unterstein, J. Heyszl, F. D. Santis, and R. Specht, "Dissecting leakage resilient prfs with multivariate localized EM attacks - A practical security evaluation on FPGA," *COSADE*, vol. 2017, p. 272, 2017.

[8] J. Doget, E. Prouff, M. Rivain, and F. Standaert, “Univariate side channel attacks and leakage modeling,” *J. Cryptographic Engineering*, vol. 1, no. 2, pp. 123–144, 2011. [Online]. Available: http://dx.doi.org/10.1007/s13389-011-0010-2

[9] G. Bertoni, J. Daemen, N. Debande, T. H. Le, M. Peeters, and G. V. Assche, “Power analysis of hardware implementations protected with secret sharing,” in *2012 45th Annual IEEE/ACM International Symposium on Microarchitecture Workshops*, Dec 2012, pp. 9–16.

[10] F. Standaert, T. Malkin, and M. Yung, “A unified framework for the analysis of side-channel key recovery attacks,” in *Advances in Cryptology - EUROCRYPT 2009, 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cologne, Germany, April 26-30, 2009. Proceedings*, 2009, pp. 443–461.

[11] S. Chari, J. R. Rao, and P. Rohatgi, “Template attacks,” in *Cryptographic Hardware and Embedded Systems - CHES 2002, 4th International Workshop, Redwood Shores, CA, USA, August 13-15, 2002, Revised Papers*, 2002, pp. 13–28.

[12] W. Schindler, K. Lemke, and C. Paar, “A stochastic model for differential side channel cryptanalysis,” in *Cryptographic Hardware and Embedded Systems - CHES 2005, 7th International Workshop, Edinburgh, UK, August 29 - September 1, 2005, Proceedings*, 2005, pp. 30–46.

[13] E. Peeters, F. Standaert, and J. Quisquater, “Power and electromagnetic analysis: Improved model, consequences and comparisons,” *Integration*, vol. 40, no. 1, pp. 52–60, 2007.

[14] S. Nikova, C. Rechberger, and V. Rijmen, “Threshold implementations against side-channel attacks and glitches,” in *Information and Communications Security, 8th International Conference, ICICS 2006, Raleigh, NC, USA, December 4-7, 2006, Proceedings*, 2006, pp. 529–545.

[15] M. Rivain and E. Prouff, “Provably secure higher-order masking of AES,” in *Cryptographic Hardware and Embedded Systems, CHES 2010, 12th International Workshop, Santa Barbara, CA, USA, August 17-20, 2010. Proceedings*, 2010, pp. 413–427.

[16] P. Maurine.

[17] H. L. Van Trees, *Detection, Estimation, and Modulation Theory: Part IV: Optimum Array Processing*. John Wiley and Sons, Inc., 2002.

[18] S. Chari, J. R. Rao, and P. Rohatgi, *Template Attacks*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 13–28.

[19] O. Choudary and M. G. Kuhn, “Efficient template attacks,” in *Smart Card Research and Advanced Applications - 12th International Conference, CARDIS 2013, Berlin, Germany, November 27-29, 2013. Revised Selected Papers*, 2013, pp. 253–270. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-08302-5_17