Hardware Trojan Detection through Golden Chip-Free Statistical Side-Channel Fingerprinting

Yu Liu
EE Department - UT Dallas
Richardson, TX 75080, USA
yxl119120@utdallas.edu

Ke Huang
EE Department - UT Dallas
Richardson, TX 75080, USA
ke.huang@utdallas.edu

Yiorgos Makris
EE Department - UT Dallas
Richardson, TX 75080, USA
yiorgos.makris@utdallas.edu

ABSTRACT

Statistical side channel fingerprinting is a popular hardware Trojan detection method, wherein a parametric signature of a chip is collected and compared to a trusted region in a multi-dimensional space. This trusted region is statistically established so that, despite the uncertainty incurred by process variations, the fingerprint of Trojan-free chips is expected to fall within this region while the fingerprint of Trojan-infested chips is expected to fall outside. Learning this trusted region, however, assumes availability of a small set of trusted (i.e. "golden") chips. Herein, we rescind this assumption and we demonstrate that an almost equally effective trusted region can be learned through a combination of a trusted simulation model, measurements from process control monitors (PCMs) which are typically present either on die or on wafer kerf, and advanced statistical tail modeling techniques. Effectiveness of this method is evaluated using silicon measurements from two hardware Trojan-infested versions of a wireless cryptographic integrated circuit.

Categories and Subject Descriptors

B.8.1 [**Integrated Circuits**]: Performance and Reliability—*Reliability, Testing, and Fault-Tolerance*

General Terms

Algorithms, Design, Security

Keywords

Hardware Trojan, Golden Chip, Side-Channel Fingerprinting, Wireless Cryptographic IC, Process Control Monitor

1. INTRODUCTION

Over the last decade, the problem of hardware Trojans in manufactured integrated circuits (ICs) has been a topic of intense investigation by academic researchers and governmental entities alike [1, 9]. Hardware Trojans are malicious modifications introduced in a manufactured IC, which can be exploited by a knowledgeable adversary to cause incorrect results, steal sensitive data, or even incapacitate a chip. Given the sensitive nature of applications wherein hardware Trojan-infested ICs may be deployed, developing detection methodologies has become paramount. Indeed, traditional test methods fall short in revealing hardware Trojans, as they are geared towards identifying modeled defects and, therefore, cannot reveal unmodeled malicious inclusions.

While various hardware Trojan detection approaches have been explored in the literature, *statistical side-channel fingerprinting* has been among the most heavily investigated ones. Starting with the global power consumption-based method presented in [2] and the path delay-based method introduced in [7], constructing *fingerprints* of ICs based on side-channel parameters and using these fingerprints to statistically assess whether an IC is contaminated by a hardware Trojan or not became a popular direction. Indeed, numerous researchers in the hardware trust area developed this idea further by using various side-channel measurements, including power supply transient signals [14, 15], leakage currents [11], regional supply currents [3], temperature [6], wireless transmission power [8], as well as multi-parameter combinations thereof [10, 13].

The underlying premise of these methods is that hardware Trojans will distort the side-channel parametric profile of an IC, even if they do not alter its functionality. While for a well-designed hardware Trojan this distortion is minute and carefully hidden within the design margins allowed for process variation, it is systematic; therefore, statistical analysis should be able to identify the presence of additional structure in the side-channel fingerprint of an IC and, thereby, reveal the Trojan. Accordingly, assuming availability of a representative set of trusted (i.e. "golden") ICs, a classifier (e.g. neural network, support vector machine, etc.) can be trained to learn the boundary separating Trojan-free and Trojan-infested chips in the side-channel parametric space, as shown in Fig. 1.

Reliance on a set of "golden" chips, however, has always been the Achilles' heel of statistical side-channel fingerprinting methods in the eyes of their critics. Indeed, access to a trusted foundry facility, even if simply for producing a low volume of "golden" chips, is typically of prohibitive cost, if at all available. Moreover, the parametric profile of devices produced by a trusted foundry will most likely differ from the parametric profile of devices produced in another facility for the same technology node, even within the same company. An alternative method for obtaining access to trusted chips

DAC'14 June 01 - 05 2014, San Francisco, CA, USA

is through advanced imaging and painstaking reconstruction of its netlist through image processing methods [4]. Besides requiring expensive equipment and tedious effort, however, such methods often need to perform destructive de-layering of an IC to reach high accuracy in contemporary nodes.

Herein, we investigate an alternative approach to gaining access to a "golden" IC population for the purpose of learning the classification boundary. Similar to most side-channel fingerprinting methods, our attack scenario assumes that the culprit is at the foundry. Therefore, a trusted Spice-level simulation model of the circuit is available. Nevertheless, as we will show experimentally, the straightforward solution of learning the classification boundary from synthetic parametric fingerprints obtained via post-layout Monte Carlo circuit simulations is doomed to failure, even when enhanced through advanced tail modeling methods. Indeed, since Spice models are updated infrequently, there is bound to be a discrepancy between the statistics of the simulation model and the actual statistics produced by the foundry process. In other words, an anchor point in actual fabricated silicon is necessary in order to learn an accurate and effective classification boundary.

To this end, we propose to utilize the Process Control Monitors (PCMs) which typically exist either on the wafer kerf (i.e. the area in-between die) or on the die for process characterization and monitoring purposes. PCMs (a.k.a e-tests) are simple circuit structures for measuring fundamental parameters of the fabricated silicon, which reflect the operating point of the fabrication process. As such, they can provide the silicon anchor point needed to establish an accurate classification boundary from simulation data. Specifically, we propose to first learn the relation between PCMs and side-channel fingerprints through *non-linear regression models* built using Monte Carlo Spice-level simulation data. At the same time, we propose to measure the PCMs of the devices under Trojan test (DUTTs) and to use them along with a *kernel mean matching method* in order to calibrate the simulation PCM data to the foundry operating point that produced these devices. The learned regression models can then be used to obtain a synthetic trusted fingerprint population from the calibrated simulation PCM data. Finally, we propose to enhance the synthetic trusted fingerprint population data through advanced *tail modeling methods*, and to use this enhanced population to learn the classification boundary, which will be used to decide whether the measured fingerprints of the DUTTs reflect Trojan-free or Trojan-infested ICs.

At first glance, this proposition may appear to simply shift the need for a core root of trust from "golden" ICs to "golden" PCMs. In other words, one might argue that a resourceful and determined attacker can fiddle with the PCMs, just like he/she would with the IC, to throw the proposed method off. This, however, is not as simple as it may sound; there are a couple of fundamental differences which we would like to highlight. First, PCMs are very simple structures which are shared across most designs implemented on a given technology node and which are thoroughly scrutinized for yield learning and process monitoring purposes. Any systematic modification of PCMs will result in deviation from expected parametric measurement statistics and is bound to trigger action by process engineers. Second, even if an attacker was able to hide the PCM modification within the process variation margin, there exists no systematic method for ensuring that such a modification would bring the fingerprints of Trojan-infested devices within the trusted region, since PCMs and side-channel fingerprints are functionally independent. In short, modifying PCMs is a much more difficult and riskier endeavor than inserting hardware Trojans, making the "golden" PCM requirement of the proposed method far more plausible than "golden" ICs.

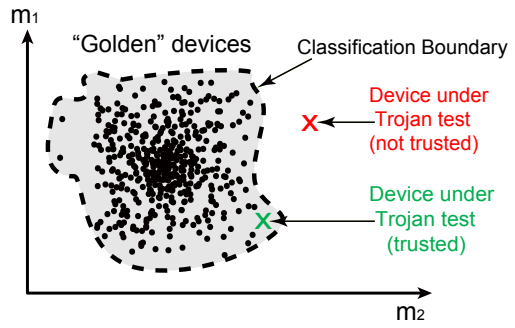

Figure 1: Statistical side-channel fingerprinting

The remainder of this paper is organized as follows: in Section 2, we introduce the details of the proposed method. In Section 3, we demonstrate its effectiveness using simulation data and silicon measurements from a Trojan-free and two Trojan-infested versions of a wireless cryptographic IC which we designed and fabricated for this study. Conclusions are drawn in Section 4.

2. OVERVIEW OF PROPOSED APPROACH

Our approach consists of three stages: pre-manufacturing, silicon measurement, and Trojan test. These three stages are presented in detail in this section.

2.1 Pre-manufacturing stage

Pre-manufacturing stage involves Spice-level Monte Carlo simulation of n "golden" devices, subject to process variation. Let $\vec{m}_p = \{m_{p,1}, \ldots, m_{p,n_p}\}$ denote the n_p-dimensional PCM measurement vector and $\vec{m} = \{m_1, \ldots, m_{n_m}\}$ denote the n_m-dimensional side-channel fingerprint vector, where n_p and let n_m are the considered number of PCMs and side-channel fingerprints, respectively. Based on the n samples obtained by the Monte Carlo simulation, we can learn non-linear regression functions to map the PCM measurement pattern $\vec{m}_p$ to the values of each side-channel fingerprint of interest m_j, $j = 1, \ldots, n_m$. In particular, we train n_m regression functions $g_j : \vec{m}_p \mapsto m_j$, as shown in Fig. 2. These non-linear regression functions implicitly learn the unknown underlying dependency between $\vec{m}_p$ and all m_j using statistical data analysis.

In parallel, we also use the n_m-dimensional side-channel fingerprint vectors produced by the Spice-level Monte Carlo simulation of n golden devices to train a one-class classifier. Specifically, we establish a classification boundary $B1$ to enclose the trusted n devices, as shown in Fig. 2. DUTTs are then considered Trojan-free if their side-channel fingerprint is inside the boundary and Trojan-infested otherwise.

Such a simplistic and straightforward boundary, however, suffers from two major weaknesses: (i) Monte Carlo simulation produces few devices at the tails of the distribution, which is the area that matters the most when trying to establish a classification boundary, and (ii) it has no anchor point in silicon and cannot reflect the process shifts that

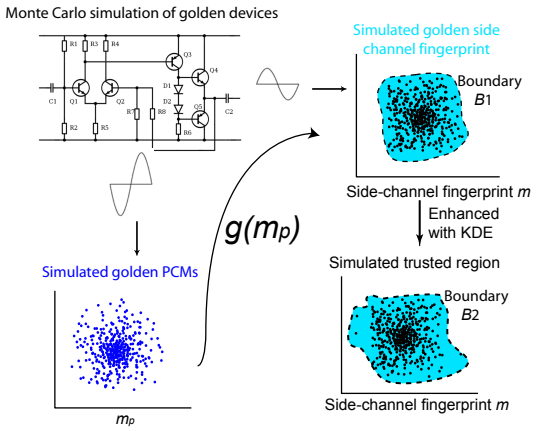

Figure 2: Pre-manufacturing stage

have taken place between the creation of the Spice simulation model and the current operating point of the foundry.

In order to address the first weakness, we can employ advanced tail modeling methods. For example, we can estimate the kernel density function (KDE) $\hat{f}(\vec{m})$ and, subsequently, use it to generate an arbitrarily large volume of enhanced synthetic data, in order to accurately reflect the tails of the distribution $f(\vec{m})$. Tail modeling and enhanced synthetic data generation is discussed in detail in Section 2.5. Based on the enhanced synthetic dataset of side-channel fingerprints, we can again use a one-class classifier to learn an improved classification boundary $B2$, as shown in Fig. 2.

2.2 Silicon measurement stage

The enhanced trusted boundary $B2$ may better model the tails of the Spice-generated distribution of devices, yet it still suffers from the discrepancy between the Spice parameter values and the actual silicon values, as produced by the operating point of the foundry when the DUTTs are fabricated. One way to address this limitation is to predict the golden side-channel fingerprints $\vec{m}'$ from a set of PCM measurements $\vec{m}'_p$ taken from the actual fabricated devices, through the non-linear regression functions learned from the simulation data, i.e. $m' = g(\vec{m}'_p)$. As discussed in the introduction, PCM measurements $\vec{m}'_p$ are unlikely to be targeted by an attacker, hence considering PCMs as the core root of trust is a more realistic assumption than requiring "golden" ICs. Thus, the PCMs of Trojan-free and Trojan-infested chips under the same process variation will be generated from the same distribution, whereas their side-channel fingerprints may exhibit different behavior. Based on the predicted m', we can then construct a more realistic classification boundary $B3$, as shown in Fig. 3.

However, the PCM samples of the DUTT silicon data $\vec{m}'_p$ used to learn $B3$ may be very limited and their values may be centered at the mean values or reflect only a narrow portion of the distribution, especially when chips are from the same fabrication lot. In order to obtain PCM samples that better reflect the actual distribution, we propose to utilize the PCM samples obtained from the Monte Carlo simulation described in the previous section, along with a mean shifting method which will calibrate them to the silicon PCM data measured on the DUTTs.

To do this, we first compute $\vec{m}''_p$ which denotes mean shifted value of PCM measurements $\vec{m}'_p$ from Monte Carlo simulation using a kernel mean matching (KMM) method. The details of KMM are discussed in Section 2.4. Note that

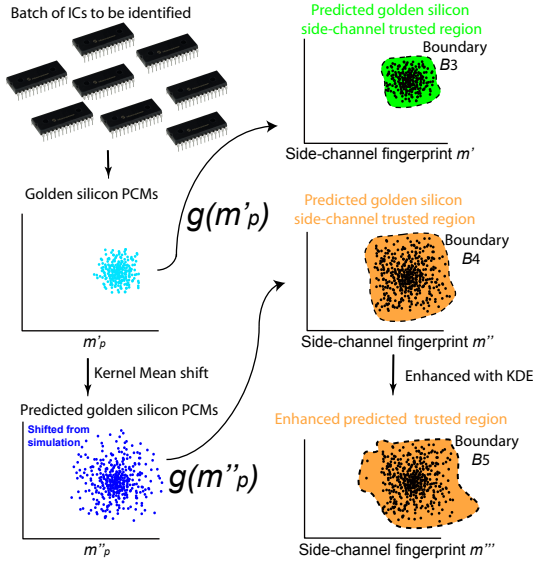

Figure 3: Silicon measurement stage

$\vec{m}''_p$ will have a wider-spread distribution as compared to $\vec{m}'_p$, since the number of Monte Carlo simulated devices can be much larger than the DUTT population. Once $\vec{m}''_p$ is computed, we can again use learned regression functions to generate side channel fingerprint samples, i.e. $m''_j = g_j(\vec{m}''_p)$, on which a one-class classifier can be trained to learn a further improved classification boundary $B4$, shown in Fig. 3.

As a last improvement, we can again address the aforementioned limitation of Monte Carlo simulation through tail modeling. Specifically, we can estimate the probability density function $\hat{f}(\vec{m}'')$ and generate a large volume of new synthetic side-channel fingerprints from m'', which will enhance the tails of the distribution $\hat{f}(\vec{m}'')$, as described previously. Using the tail-enhanced synthetic side-channel fingerprint samples, we can train a one-class classifier to outline the trusted region for the DUTTs, as shown by the boundary $B5$ in Fig. 3.

2.3 Trojan test stage

Once the classification boundary outlining the trusted region in the side-channel fingerprint space is established, testing for Trojans is a simple application of the traditional side-channel fingerprinting method shown in Fig. 1. The only difference is that the classification boundary has not been learned through "golden" ICs but rather through the sequence of steps outlined above.

For any of the above classification boundaries $B1$ through $B5$, we can evaluate their effectiveness in detecting Trojan-infested devices by computing False Positive (FP) and False Negative (FN) rates. For a particular device, let the indicator functions $I_1^{(i)}/I_2^{(i)}$ be equal to '1' if the predicted outcome of the i-th device is Trojan-free/Trojan-infested while the actual device is Trojan-infested/Trojan-free, and let $I_1^{(i)}/I_2^{(i)}$ be equal to '0' otherwise. Then the overall FP and FN rates are defined as:

$$FP = \sum_{i=1}^{N} I_1^{(i)} \tag{1}$$

$$FN = \sum_{i=1}^{N} I_2^{(i)} \tag{2}$$

where N is the number of DUTTs.

2.4 Kernel mean matching

When the regression functions $g_j : \vec{m}_p \mapsto m_j$ are learned, the input observation $\vec{m}_p$ is obtained from the input distribution $p(\vec{m}_p)$, while the corresponding m_j is drawn from the conditional distribution $p(m_j|\vec{m}_p)$. For these regression functions to be effective, the underlying assumption is that $p(\vec{m}_p)$ remains the same between training samples and the actual DUTTs. However, when $p(\vec{m}_p)$ changes between training and testing, training data will not accurately reflect the testing data, due to what is known as *covariate shift*.

To overcome this problem, we use kernel mean matching (KMM) [5] to re-weight the training set. KMM infers the importance weight directly by non-parametric distribution matching between the training set and the testing set, without estimating the probability density function of these sets. KMM works by minimizing the discrepancy between the means of the importance-weighted training and testing sets in a reproducing kernel space.

The objective function is given by the difference of the two empirical means:

$$\| \frac{1}{n_{tr}} \sum_{i=1}^{n_{tr}} \beta_i \Phi(\vec{m}_{p,i}^{tr}) - \frac{1}{n_{te}} \sum_{i=1}^{n_{te}} \beta_i \Phi(\vec{m}_{p,i}^{te}) \|^2 \tag{3}$$

where n_{tr} is the number of training samples, and n_{te} is the number of testing samples, subject to

$$\beta_i \in [0, B] \quad and \quad | \frac{1}{n_{tr}} \sum_{i=1}^{n_{tr}} \beta_i - 1 | \leq \epsilon$$

where $B > 0$ and $\epsilon > 0$ are tuning parameters, and Φ are the mapping functions. The quadratic problem to find a suitable β becomes:

$$min \quad \frac{1}{2} \beta^T K \beta - \kappa^T \beta \tag{4}$$

where $K_{ij} = k(\vec{m}_{p,i}^{tr}, \vec{m}_{p,j}^{tr})$, $\kappa_i = (n_{tr}/n_{te}) \sum_{j=1}^{n_{tr}} k(x_i^{tr}, x_j^{te})$ and k is the kernel function. Interested readers are referred to [5] for more details on the KMM method.

2.5 Tail modeling & synthetic data generation

In order to generate a large volume of synthetic data which accurately reflect the distribution of trusted side channel fingerprints, we use non-parametric kernel density estimation (KDE). This method relies on the estimation of the densities $f(\vec{m})$, using the available observations $\vec{m}_i$, $i = 1, \cdots, M$, where M is the number of available samples used to build the density. We do not make any assumption regarding its parametric form (e.g. normal). Instead, we use non-parametric KDE, which allows the observations to speak for themselves. The kernel density estimate is defined as [16]

$$\hat{f}(\vec{m}) = \frac{1}{M \times h^d} \sum_{i=1}^{M} K_e(\frac{1}{h}(\vec{m} - \vec{m}_i)) \tag{5}$$

where h is a parameter called bandwidth, $d = n_m$ is the dimension of $\vec{m}$, and $K_e(t)$ is the Epanechnikov kernel

$$K_e(t) = \begin{cases} \frac{1}{2} c_d^{-1} (d+2)(1 - t^T t) & \text{if} \quad t^T t < 1 \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

and $c_d = 2\pi^{d/2} / (d \cdot \Gamma(d/2))$ is the volume of the unit d-dimensional sphere. The kernel density estimate can be interpreted as the normalized sum of a set of identical kernels centered on the available observations for the 1-dimensional case. The bandwidth h corresponds to the distance between the center of the kernel and the kernel's edge, where the kernel's density becomes zero.

In this work, we use an adaptive version of (5). In particular, we allow the bandwidth h to vary from one observation $\vec{m}_i$ to another, allowing larger bandwidths for the observations that lie at the tails of the distribution. The adaptive kernel density estimate is defined as [16]

$$\hat{f}_\alpha(\vec{m}) = \frac{1}{M} \sum_{i=1}^{M} \frac{1}{(h \cdot \lambda_i)^d} K_e(\frac{1}{h \cdot \lambda_i}(\vec{m} - \vec{m}_i)) \tag{7}$$

where the local bandwidth factors λ_i are defined as

$$\lambda_i = \{ \hat{f}(\vec{m}_i)/g \}^{-\alpha}, \tag{8}$$

$\hat{f}(\vec{m}_i)$ is the pilot density estimate given in (5), g is the geometric mean

$$\log g = M^{-1} \sum_{i=1}^{M} \log \hat{f}(\vec{m}_i) \tag{9}$$

and α is a parameter which controls the local bandwidths. The larger the α, the larger the diagnostic measurement space where the density $\hat{f}(\vec{m})$ is nonzero.

Once the probability density $\hat{f}(\vec{m})$ is estimated, we can easily sample $\hat{f}(\vec{m})$ to generate a large volume of new synthetic data $s_n = \{\vec{m}_1, \ldots, \vec{m}_{M'}\}, M' \gg M$. The generated synthetic data set S_n will better reflect the distribution tails.

3. EXPERIMENTAL RESULTS

3.1 Experimentation platform

The proposed method is evaluated using a wireless cryptographic IC as our experimentation platform. The digital part of the circuit consists of an Advanced Encryption Standard (AES) core and a serialization buffer, while the analog part is an Ultra-Wide-Band (UWB) transmitter. This circuit takes a plaintext as an input, encrypts it using AES with a key that is stored on the chip, serializes the ciphertext and passes it on to the UWB, which transmits it in blocks of 128 bits over a public channel. More details regarding this platform can be found in [12].

In total, 40 chips were fabricated using TSMC's 350nm technology. Each of these chips hosts a Trojan-free and two different Trojan-infested versions of the design. In total, we have $40 \times 3 = 120$ devices, of which 40 are Trojan-free and 80 are Trojan-infested. The Trojan-infested devices leak the AES encryption key by hiding it in the wireless transmission power amplitude/frequency margins allowed for process variations, while ensuring that the device continues to meet all of its functional specifications. Specifically, along with each 128-bit ciphertext, the 128 bits of the AES are leaked by modulating the amplitude or the frequency of each ciphertext bit transmission based on the value of the leaked AES key bit: when the leaked key bit is '1', the amplitude/frequency is left unaltered, while when the leaked key bit is '0', they are slightly increased. These Trojans have been shown to be extremely powerful and capable of leaking the key to an attacker who knows what to listen for on the public channel, while evading all traditional manufacturing test methods. Nevertheless, statistical side-channel fingerprinting, where the transmission power of the "golden" ICs is used to establish a classification boundary, is able to *perfectly* separate the Trojan-infested from the Trojan-free

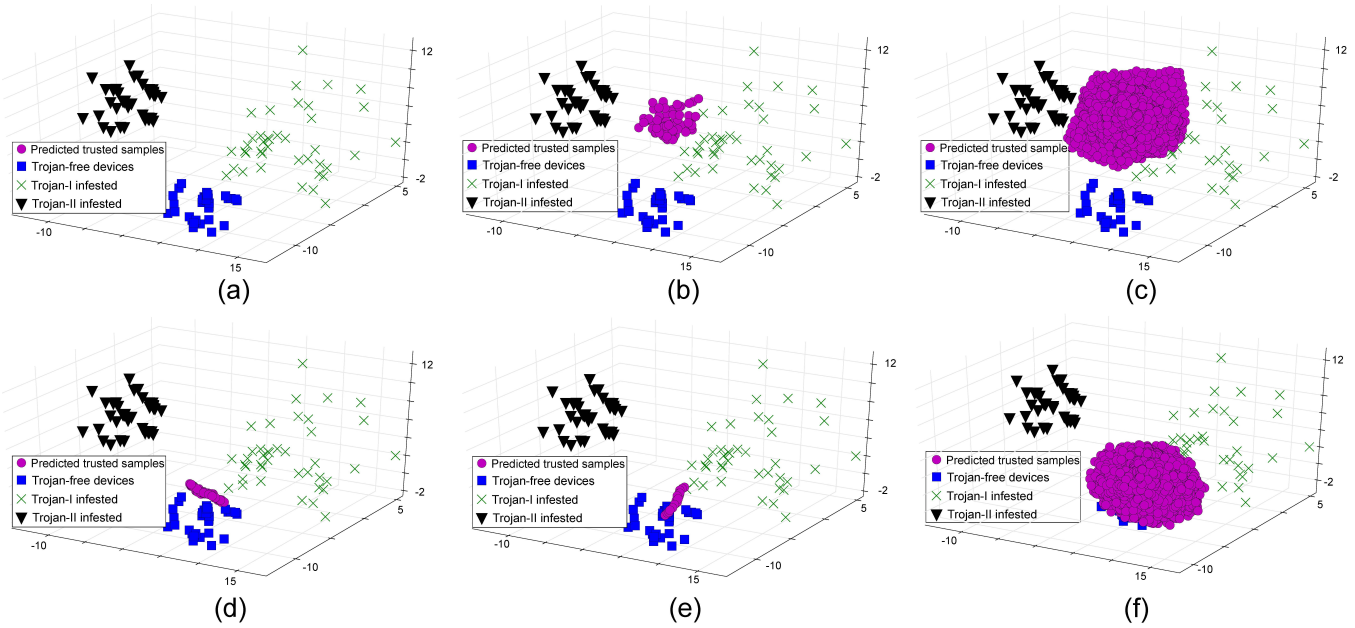

Figure 4: Visualization of side-channel fingerprints of fabricated Trojan-free and Trojan-infested devices, along with generated datasets S_1 through S_6 from which classification boundaries $B1$ through $B5$ are learned.

populations [12]. This result sets the stage for assessing the effectiveness of the proposed method, which does not rely on "golden ICs" to learn the boundary.

In our experiment, we use $n_m = 6$ side-channel fingerprints, namely the measured output power when transmitting 6 randomly chosen 128-bit ciphertext blocks, encrypted with a randomly chosen key, over the public wireless channel. Also, as a proxy for on-die PCMs, we use $n_p = 1$ delay measurement on a simple digital path, which we included on our chip for silicon characterization purposes.

3.2 Dataset generation & visualization

In order to learn the classification boundaries $B1$ through $B5$ which were introduced in Section 2, we use the post-layout extracted circuit netlist and the actual fabricated devices to generate the following data sets S_1 through S_5:

(1) S_1 is a 100×6 matrix containing 100 Trojan-free golden samples of the 6 side-channel fingerprints, obtained by Monte Carlo simulation: $S_1 = [\vec{m}_1, \ldots, \vec{m}_{100}]^{\mathrm{T}}$.

(2) S_2 is a $10^5 \times 6$ matrix containing 10^5 tail-enhanced synthetic data generated from S_1 using KDE, as described in Section 2.5: $S_2 = [\vec{m}_1, \ldots, \vec{m}_{10^5}]^{\mathrm{T}}$.

(3) S_3 is a 120×6 matrix containing the 6 predicted side-channel fingerprints from the 120 fabricated devices: $S_3 = [\vec{m}''_1, \ldots, \vec{m}''_{120}]^{\mathrm{T}}$. S_3 is predicted from actual PCM measurements on the 120 fabricated devices, using non-linear regression functions $g(\vec{m}'_p)$ as described in Section 2. In this work, Multivariate Adaptive Regression Splines (MARS) were used to train the regression models.

(4) S_4 is a 100×6 matrix containing 6 predicted side-channel fingerprints from shifted 100 PCM measurements obtained by Monte Carlo simulation: $S_4 = [\vec{m}'_1, \ldots, \vec{m}'_{100}]^{\mathrm{T}}$. KMM was used to obtain the 100 shifted PCM measurements $\vec{m}'_p$, followed by an application of the learned non-linear regression functions $g(\vec{m}'_p)$, in order to generate S_4.

(5) S_5 is a $10^5 \times 6$ matrix containing 10^5 tail-enhanced synthetic data, generated from S_4 using KDE, as described in Section 2.5: $S_5 = [\vec{m'}_1, \ldots, \vec{m'}_{10^5}]^{\mathrm{T}}$.

To gain insight regarding the structure of the generated 6-dimensional datasets S_1 through S_6, we perform a Principal Component Analysis (PCA) on each of them and we project the generated population on the top three principal components, as depicted in Fig. 4(a)-(f). In the first of these plots, blue squares indicate the measured side-channel fingerprints of the 40 fabricated Trojan-free devices, while green x's and black triangles represent the measured fingerprints of the 80 fabricated Trojan-infected devices (with modulated amplitude and modulated frequency, respectively). In each of the subsequent five plots, purple circles represent the side-channel fingerprint population of devices in sets S_1 through S_5, respectively. These purple dot datasets are subsequently used to learn the classification boundaries $B1$ through $B5$, which were introduced in Section 2. In our case, a 1-class Support Vector Machine (SVM) is used for this purpose.

3.3 Results & observations

Table 1 shows the effectiveness of the 5 boundaries $B1$ through $B5$ in correctly classifying the 40 Trojan-free and 80 Trojan-infested devices based on their side-channel fingerprint. For each of the five boundaries, we report the false positive (FP) and false negative (FN) metrics, which were defined in Equations 1 and 2, respectively.

As can be observed, learning the trusted boundary directly from Monte Carlo simulation results in a very poor identification accuracy, as shown in the first line of Table 1. While all Trojan-infested devices are indeed classified as such, so are all the Trojan-free devices, signifying that the learned classification boundary is significantly off with respect to the process. The situation remains unchanged even after enhancing the synthetic population with the KDE-based tail modeling method, as can be observed in the second line of Table 1. This observation can be further explained by Fig. 4(b) and (c), where the purple dots representing the "golden" samples from which the classification boundary is learned are cleanly separable from the Trojan-infested devices, shown by green x's and black triangles, but

Table 1: Trojan detection metrics for each data set

Data set used to train the trusted region	FP	FN
S_1	0/80	40/40
S_2	0/80	40/40
S_3	0/80	24/40
S_4	0/80	18/40
S_5	0/80	3/40

do not encompass any of the Trojan-free devices, shown by blue squares. This inefficiency is mainly due to the process shift and the discrepancy between the Spice simulation model and the actual silicon.

The situation improves significantly when using the classification boundary $B3$, which is learned through the actual PCM measurements of the fabricated devices, from which the "golden" fingerprints of dataset S_3 are derived using the regression functions learned on the simulation data. Indeed, while all 80 Trojan-infested devices continue to be correctly classified, so are 16 out of the 40 Trojan-free devices, as can be observed in third line of Table 1. The partial overlap between the purple dots and the blue circles in Figure 4(d) intuitively supports this result. The results improve even further when the classification boundary $B4$ is employed, learned on the population S_4 which is the result of using KMM to calibrate the PCM measurements obtained via Monte Carlo simulation to the actual silicon operation points, as reflected in the silicon PCM measurements of the fabricated devices. In this case, while correctly classifying all 80 Trojan-infested devices, boundary $B4$ also classifies correctly 22 out of the 40 Trojan-free devices, as can be observed in fourth line of Table 1. The improved partial overlap between the purple dots and the blue circles in Figure 4(e), as compared to Figure 4(d), supports this result.

Finally, when the classification boundary $B5$ is learned using the "golden" fingerprint dataset S_5, which is enhanced through the proposed KDE-based tail modeling method, its effectiveness becomes comparable to the boundary learned from the actual "golden" ICs. Indeed, as can observed in the last line of Table 1, all Trojan-infested devices are correctly classified and all except three Trojan-free devices are also correctly classified. The almost complete overlap of the purple dots with the blue squares and the clean separation from the green x's and black triangles, as shown in Fig. 4(f), corroborates the effectiveness of the proposed method in detecting hardware Trojans through side-channel fingerprinting without relying on "golden" ICs.

4. CONCLUSIONS

While statistical side-channel fingerprinting has been among the favorite hardware Trojan detection methods, it has long been criticized for its reliance on a set of trusted "golden" ICs from which the classification boundary is learned. Herein, we demonstrated that this requirement can be revoked with minimal impact on the quality of the learned classification boundary. Indeed, as theoretically discussed and experimentally demonstrated on silicon measurements from a Trojan-free and two Trojan-infested versions of a wireless cryptographic IC, an almost equally effective trusted region can be learned through a combination of a trusted simulation model, silicon measurements from PCMs, and advanced statistical tail modeling techniques.

5. ACKNOWLEDGMENTS

This research was partially supported by the National Science Foundation (NSF 1149465).

6. REFERENCES

[1] S. Adee. The hunt for the kill switch. In *IEEE Spectrum*, pp. 45(5):34–39, 2008.

[2] D. Agrawal, S. Baktir, D. Karakoyunlu, P. Rohatgi, and B. Sunar. Trojan detection using IC fingerprinting. In *IEEE Symposium on Security and Privacy*, pp. 296–310, 2007.

[3] D. Du, S. Narasimhan, R. Chakraborty, and S. Bhunia. Self-referencing: A scalable side-channel approach for hardware Trojan detection. In *Cryptographic Hardware and Embedded Systems*, pp. 173–187, 2010.

[4] P. Song, F. Stellari, D. Pfeiffer, J. Culp, A. J. Weger, A. Bonnoit, B. Wisnieff, and M. Taubenblatt. Marvel: Malicious alteration recognition and verification by emission of light. In *IEEE International Symposium on Hardware-Oriented Security and Trust*, pp. 117–121, 2011.

[5] A. Gretton, A. Smola, J. Huang, M. Schittfull, K. Borgwardt, and B. Schölkopf. Covariate shift by kernel mean matching. In *Dataset Shift in Machine Learning*, pp. 131–160, 2009.

[6] K. Hu, A. Nowroz, S. Reda, and F. Koushanfar. High-sensitivity hardware Trojan detection using multimodal characterization power mapping of integrated circuits using AC-based thermography. In *IEEE/ACM Design, Automation and Test in Europe*, pp. 1271-1276, 2013.

[7] Y. Jin and Y. Makris. Hardware Trojan detection using path delay fingerprint. In *IEEE International Workshop on Hardware-Oriented Security and Trust*, pp. 51–57, 2008.

[8] Y. Jin and Y. Makris. Hardware Trojans in wireless cryptographic ICs. In *IEEE Design and Test of Computers*, pp. 27(1):26–35, 2010.

[9] R. Karri, J. Rajendran, K. Rosenfeld, and M. Tehranipoor. Toward trusted hardware: Identifying and Classifying Hardware Trojans. In *IEEE Computer*, pp. 43(10):39–46, 2010.

[10] F. Koushanfar and A. Mirhoseini. A unified framework for multimodal submodular integrated circuits Trojan detection. In *IEEE Transactions on Information Forensics and Security*, pp. 6(1):162–174, 2011.

[11] C. Lamech, J. Aarestad, J. Plusquellic, R. Rad, and K. Agarwal. REBEL and TDC: two embedded test structures for on-chip measurements of within-die path delay variations. In *IEEE/ACM International Conference on Computer-Aided Design*, pp. 170–177, 2011.

[12] Y. Liu, Y. Jin, and Y. Makris. Hardware Trojans in wireless cryptographic ICs: Silicon demonstration & detection method evaluation. In *IEEE/ACM International Conference on Computer-Aided Design*, pp. 399–404, 2013.

[13] S. Narasimhan, D. Du, R. Chakraborty, S. Paul, F. Wolff, C. Papachristou, K. Roy, and S. Bhunia. Multiple-parameter side-channel analysis: A non-invasive hardware Trojan detection approach. In *IEEE International Symposium on Hardware-Oriented Security and Trust*, pp. 13–18, 2010.

[14] R. Rad, J. Plusquellic, and M. Tehranipoor. Sensitivity analysis to hardware Trojans using power supply transient signals. In *IEEE International Workshop on Hardware-Oriented Security and Trust*, pp. 3–7, 2008.

[15] R. M. Rad, X. Wang, M. Tehranipoor, and J. Plusquellic. Power supply signal calibration techniques for improving detection resolution to hardware Trojans. In *IEEE/ACM International Conference on Computer-Aided Design*, pp. 632–639, 2008.

[16] B. W. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman & Hall/CRC, 1986.