A Novel Golden-Chip-Free Clustering Technique Using Backscattering Side Channel for Hardware Trojan Detection

Luong N. Nguyen, *Student Member, IEEE*, Baki Berkay Yilmaz *Student Member, IEEE*, Milos Prvulovic, *Senior Member, IEEE*, and Alenka Zajić, *Senior Member, IEEE*

***Abstract*— Over the past few years, malicious hardware modifications, a.k.a. hardware Trojans (HT), have emerged as a major security threat because integrated circuit (IC) companies have been fabricating chips at offshore foundries due to various factors including time-to-market, cost reduction demands, and the increased complexity of ICs. Among proposed hardware Trojan detection techniques, reverse engineering appears to be the most accurate and reliable one because it works for all circuits and Trojan types without a golden example of the chip. However, because reverse engineering is an extremely expensive, time-consuming, and destructive process, it is difficult to apply this technique for a large population of ICs in a real test environment. This paper proposes a novel golden-chip-free clustering method using backscattering side-channel to divide ICs into groups of Trojan-free and Trojan-infected boards. The technique requires no golden chip or a priori knowledge of the chip circuitry, and divides a large population of ICs into clusters based on how HTs (if existed) affect their backscattered signals. This significantly reduces the size of test vectors for reverse engineering based detection techniques, thus enables deployment of reverse engineering approaches to a large population of ICs in a real testing scenario. The results are collected on 100 different FPGA boards where boards are randomly chosen to be infected or not. The results show that we can cluster the boards with 100% accuracy and demonstrate that our technique can tolerate manufacturing variations among hardware instances to cluster all the boards accurately for 9 different dormant Trojan designs on 3 different benchmark circuits from Trusthub. We have also shown that we can detect dormant Trojan designs whose trigger size has shrunk to as small as 0.19% of the original circuit with 100% accuracy as well.**

***Index Terms*— Hardware Trojan, Hardware security, Clustering, Reverse engineering, Backscattering side-channel, Trojan detection.**

I. Introduction

Over the past few years, a significant shift in the manufacturing model and design flow of IC companies has been observed due to various factors including time-to-market, cost reduction demands, and the increased complexity of ICs. These companies have fully adopted the "horizontal model", in which they use IPs from third-party companies and outsource all hardware fabrication to offshore foundries. While the new design flow model allows for reduction in the cost, time-to-market and fabrication errors, it raises questions on the hardware level trust which provides the base layer of the security and trust that all software layers are depended and built on.

One of the major security concerns is how to detect malicious hardware changes, which are known as hardware Trojans (HT). A typical HT consists of two parts: trigger and payload. The trigger is a circuit that constantly checks for the right conditions to activate the Trojan, and payload is the entire malicious function that the Trojan executes when it is triggered. Typically, HTs are triggered at very rare conditions, which makes them extremely challenging to detect by traditional function verification and testing.

HTs could be injected into an IC by adversaries at any stage of the design and fabrication flow. Figure 1 shows the IC life cycle and a subset of opportunities for inserting HTs into the IC. HT insertion at the foundry is the most common scenario because IC companies fabricate their chips in offshore foundries, which are harder to secure. Hence, numerous HT detection techniques are proposed to detect HT insertion at the foundry stage. These techniques can be classified into two groups: reverse engineering and side-channel approaches.

Reverse-engineering techniques rely on destructive scanning the actual IC layout to re-build the GDSII and netlist level of the chip [1]-[7]. The destructive scanning process consists of decapsulation to remove the die from the package, delayering to strip each layer off the die, and imaging to reconstruct images for every layer. After getting the GDSII and netlist level of the chip, these techniques are capable of detecting any malicious post-RTL-design insertion with very high accuracy by comparing them to the GDSII and netlist of a trusted design. However, reverse-engineering is extremely time-consuming, expensive and destructive because of chip demolishing after reverse engineering. Therefore, applying reverse engineering based HT detection techniques to test a large population of ICs, although accurate and reliable, is not practical.

On the other hand, side-channel analysis based approaches rely on measuring some non-functional properties from outside of the IC while it operates, and comparing the measurements to reference signals produced by either simulation [8]-[10] or by a "golden-sample" device [11]. Potential side-channels include backscattering [11], power consumption [12], [13], leakage current [14], temperature [15], electromagnetic emanations (EM) [8], [16], or a combination of multiple side-channels [17], [18]. In some techniques, additional measurement cir-

This work has been supported, in part, by NSF grants 156399, 1651273 and 1740962, DARPA LADS contract FA8650-16-C-7620, and ONR grants N00014-17-1-2540 and N00014-19-1-2287. The views and findings expressed in this paper are those of the authors and do not necessarily reflect the views of NSF, DARPA, and ONR.

cuitry is added to the design [19], [20], which allows the specific signals to be measured close to the signal source. However, additional circuitry results in circuit size, manufacturing cost, performance, and power overhead. Therefore, the majority of side-channel based detection techniques require no modifications to the chip itself, and rely on measuring side-channel signals outside of the chip. In contrast to reverse engineering techniques, the side-channel based techniques can be applied to a large population of ICs because side-channel measurements do not require damaging the board while conducting testing. However, the disadvantage of side-channel techniques is their dependence on either having a "golden" (HT-free) chip, which is not a practical assumption for foundry-inserted HTs in single-source ICs, or having a detailed simulation model, which is often impractical (complex ICs, 3rd-party IP, etc.).

To overcome these shortcomings of both types of approaches, we propose a novel "golden-chip-free" clustering algorithm using backscattering side-channel. This technique is bridging the gap between destructive reverse-engineering and traditional side-channel detection techniques. The proposed clustering algorithm clusters a large population of ICs based on the effect of a hypothetical HT would have on the backscattering side-channel signal. In practical terms, the technique creates clusters such that the ICs in each cluster can be considered equivalent in terms of presence or absence of an HT. This allows reverse-engineering of one IC in each cluster to be used to assess the status (in terms of HT presence and nature) of that entire cluster.

A number of techniques utilizing clustering algorithms for HT detection have been previously proposed [21]-[24], however, the majority of these methods are pre-silicon approaches, which means that they can not detect HTs inserted in the fabrication stage [21]-[23]. A post-silicon clustering technique using side-channel analysis have been proposed in [24], but authors only test their method on a set of two FPGAs, which does not give enough statistics to evaluate manufacturing variations among different hardware instances. In addition, the technique uses power side-channel, which provides very limited resolution and bandwidth [11]. Unlike these previous approaches, our approach works for HTs inserted at foundries without needing a golden chip or any a priori knowledge of the chip circuitry. We have tested the proposed technique on a set of 100 boards, which provides enough statistics for manufacturing variation, and shows that our technique outperforms other side-channels for HT detections [11].

We evaluate our clustering algorithm for multiple HT and circuit benchmark designs over a set of 100 boards, in which each board will be randomly loaded with either a HT-free or an HT-infected design. In all these experiments the HT (if present) is in a dormant state, i.e. none of the HTs are activated during this evaluation. The results show that our technique is capable of clustering all boards correctly for 9 different Trojan designs on 3 different benchmark circuits from Trusthub [26] with 100 % accuracy. In additional experiments, we make HTs more stealthy by reducing the size of their trigger, resulting in trigger circuits that are as small as 0.19% of the original circuit, and find out that our technique still correctly clusters

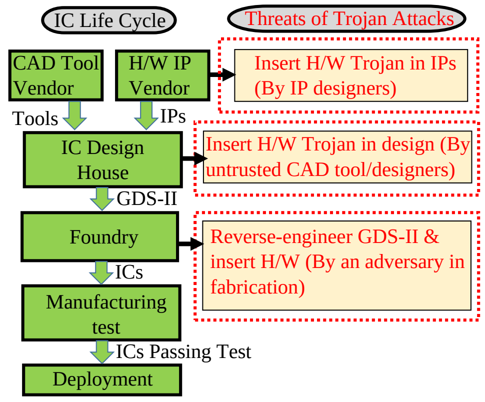

IC Life Cycle
Threats of Trojan Attacks
CAD Tool Vendor
H/W IP Vendor
Insert H/W Trojan in IPs (By IP designers)
Tools
IPs
IC Design House
Insert H/W Trojan in design (By untrusted CAD tool/designers)
GDS-II
Foundry
Reverse-engineer GDS-II & insert H/W (By an adversary in fabrication)
ICs
Manufacturing test
ICs Passing Test
Deployment

Fig. 1. IC life cycle with examples of HT insertion activity [25].

the boards. The following summarizes the contributions of this paper:

- This paper describes a novel clustering algorithm that is capable of classifying a large population of ICs into clusters without having a "golden" (known-to-be-HT-free) chip, and with no a priori knowledge about circuitry of the chip. The algorithm is based on clustering spectral features of backscattering side-channel that tend to be the most impacted by dormant HTs.
- This paper describes a testing environment that includes a set of 100 boards and implemented multiple HT benchmarks. This large set of boards allows a thorough evaluation of the manufacturing variation among different hardware instances with enough statistics, which has not been done before.

The rest of the paper is organized as follows. In Section II, we provide a background for this work. Section III defines the problem and attack scenarios. Section IV explains our clustering technique and algorithm, while Section V describes our experimental setup and testing scheme formulation. Section VI presents the results of our technique, while Section VII discusses related work. Finally, Section VIII concludes the paper.

II. Background

A. Hardware Trojans: Characteristics and Taxonomy

Conventionally, the hardware has been seen as the root of trust, and the only untrusted parts were assumed to be the software or firmware running on top of the hardware. However, several studies on HTs have shown that even the hardware platform cannot be trusted anymore [27]. Over the past several years, numerous papers have been published on the topic of understanding the intent and behavior [28], [25], implementation [29]-[26], and taxonomy of hardware Trojans [26]-[32]. HTs are undesired and unknown malicious modifications to a hardware circuit that have three common characteristics: rarity of activation, malicious purpose, and invasion of detection [25].

Typically, an HT consists of two components: trigger and payload. The trigger circuit gets input from the host circuit to constantly check for the right conditions to activate the payload. In these very rare conditions, the payload is activated

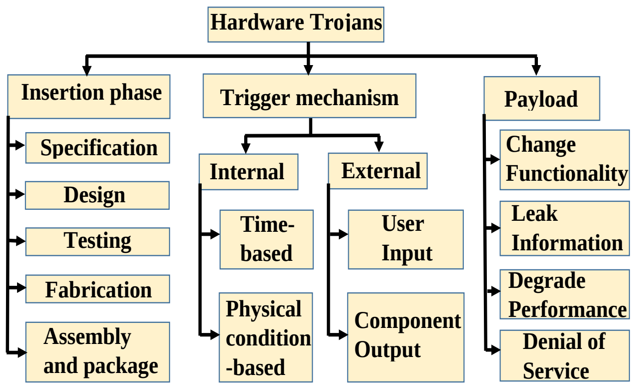


Fig. 2. Hardware Trojans Taxonomy [26].

by the triggering signal from the trigger circuit to perform malicious activities. They could be leaking sensitive information, allowing the attackers to gain access to the hardware, or shortening the operational lifetime of the hardware.

As the number and complexity of HTs increased dramatically, several studies on the topic of characterizing and classifying HTs have been published over the last few years [26]-[33]. The most comprehensive work to date is proposed by [26]. Figure 2 illustrates different ways of classifying HTs. As shown in the figure, HTs can be classified by their activation mechanism, functionality, or the phase in the IC design flow they are inserted into the chip.

B. Backscattering Side-Channels

Backscattering has been used in RFID communication system to enable RFID tags to transmit information to RFID reader for decades [34]. A typical passive RFID tag contains an ASIC chip that can switch between two impedances, where one impedance is selected to maximize the tag's radar cross-section (RCS), while the other one is selected to minimize the RCS [11]. The RFID reader propagates a continuous wave toward the RFID tag and measures signal reflected back that is modulated with information about RCS changes.

Using the analogy with RFID communication systems, the authors in [11], [35] proposed using backscattering signals as a way to collect side-channels that carry information about impedance change in the circuits. Figure 3 shows an example of a CMOS inverter and its equivalent impedance circuits when the output is high and low, respectively. These impedances are different because the geometry and doping levels of PMOS and NMOS are not exactly the same. As a result, similar to the mechanism of RFID tags, this impedance switching changes the circuit's RCS, thus modulates the signal that is backscattered from the circuit with the information about impedance changes in the system. This creates backscattering side-channel.

Note that unlike other analog side-channels such as electromagnetic emanation (EM) and power, which are a consequence of current-flow changes inside the chip, backscattering side-channel is an impedance-based side channel that is the consequence of impedance switching activities inside the chip. These channels can be created by propagating a continuous-wave signal toward the chip. The transistor switching activities

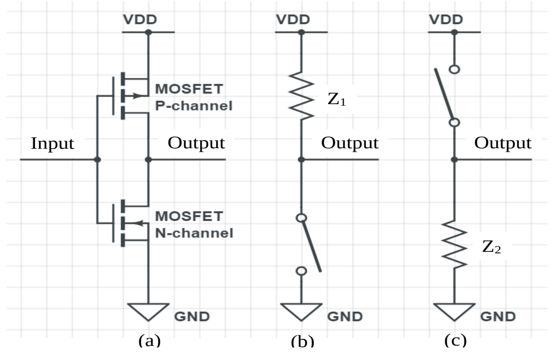


Fig. 3. CMOS NOT gate (a), and the equivalent impedance circuit when its output is high (b), and low (c).

cause changes in the chip impedance, which modifies the radar cross-section (RCS) of the circuit. This RCS change modulates the signal that is backscattered (reflected) from the chip, which creates an impedance-based backscattering side-channel. If hardware Trojan is added to a circuit, it changes the impedance of the circuit even if the Trojan is not activated. The changes will be reflected in the backscattered signal, which is beneficial to the detection of hardware Trojan.

The backscattering side-channel has several advantages compared to other side-channels such as EM and power. These advantages can be listed as follows:

- *High bandwidth:* This provides the capability of detecting small and fast switching Trojan activities.
- *Signal strength not limited by leakage from devices:* One characteristic that sets the backscattering side-channel aside from others is that its signal strength can be improved by increasing the carrier's input power. As a result, the backscattering side-channel can still work when there is very little leakage from devices.
- *Adaptable frequency:* By changing the carrier frequency, we can change the working frequency of the backscattering side-channel. This helps to increase the signal-to-noise ratio by shifting the frequency to avoid interrupts that might distract the changes caused by HT activities.

III. Attack Scenarios and Problem Statement

A. Attack Scenarios

During the fabrication process at foundries, if an adversary has access to the chip layout and adds HTs to the design, a part or the entire population of ICs will be injected HTs, depending on how the ICs are produced. As a result, there are three possible scenarios:

- No adversary: There is no malicious modifications to any chip. Therefore, the entire population of ICs is HT-free.
- Partial insertion: There are malicious modifications to some of the chips. This happens when different batches of ICs are fabricated at different chronological phases of production and the attacker only inserts Trojan at one or some phases. As a result, a part of the population of ICs have Trojans, while the rest are HT-free.
- Full insertion: Malicious modification exists in all of the chips. This happens when all ICs are fabricated at once, and the attacker inserts HTs to the chip layout. As a result, the entire population of ICs will be HT-infected.

B. Problem Statement

As discussed in Section I, there are two methods for the detection of HTs inserted at foundries: reverse engineering and side-channel analysis. Side-channel analysis techniques have advantage of being non-destructive and relatively fast, which is suitable for testing a large number of ICs. However, the problem with side-channel techniques is the dependence on either 1) having a "golden" chip (a chip that is a priori known to be HT-free), which is not a practical assumption if HTs were inserted at foundries, or 2) simulation, which only works for the specific circuits modeled. These difficulties prevent these techniques from being used without other assisting techniques for HT detection in practice. In contrast, reverse engineering techniques are highly accurate and need neither simulation nor a "golden" chip, which allows them to be used for the detection of HTs without any assisting techniques. However, the problem with these techniques is that the reverse engineering process is extremely expensive, time-consuming, and destructive. Hence, these techniques could not be deployed for a large population of ICs.

To circumvent the introduced difficulties faced with the previous methods, we propose a novel clustering method using backscattering side-channel to enable the deployment of reverse engineering techniques to a large population of ICs. The problem statement is as follows: There are *M* fabricated ICs, denoted as IC_1, IC_2,...,IC_{M-1}, IC_M. Utilizing each IC, a trace of features is extracted from its backscattering side-channel signals while it operates. Each IC then can be represented as a point in a high dimensional space. These ICs can be divided into clusters based on how hardware Trojan (if existed) affects their backscattering side-channel signals. The objective of the proposed clustering algorithm is to divide all tested ICs into correct clusters, so that every IC in a cluster should belong to the same type in terms of whether they are affected by HTs or not. This helps to reduce the size of test vectors tremendously for reverse engineering techniques because only one IC is required to test from each cluster.

IV. A Novel Clustering Algorithm For HT Detection

A. The Impact of HTs on Backscattering Side-channel Signal

Nguyen et al. [11] have shown that HTs can be detected by analyzing impedance changes within sub-clock samples, where the changes caused by HTs happen and can be observed on the clock signal. Fig. 4 illustrates a theoretical example of a clock signal modeled as a square wave with added Gaussian noise. Fig. 5 shows a theoretical example of a clock signal affected by HTs. As shown in the figures, if we can capture the backscattered signal of sub-clock samples where the changes caused by HT can be observed, we can detect the presence of HTs. However, the problem with the time-domain signal is that they are often very noisy, therefore, difficult to extract and synchronize measurements to get samples where changes caused by HTs happen.

In contrast, the changes caused by HTs occurring abruptly at some point in the clock cycle can be observed in frequency domain by performing short Time Fourier transformation

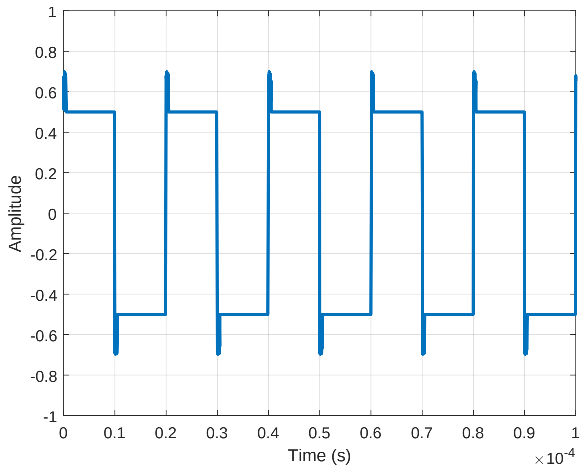


Fig. 4. An example of a clock signal with noise.

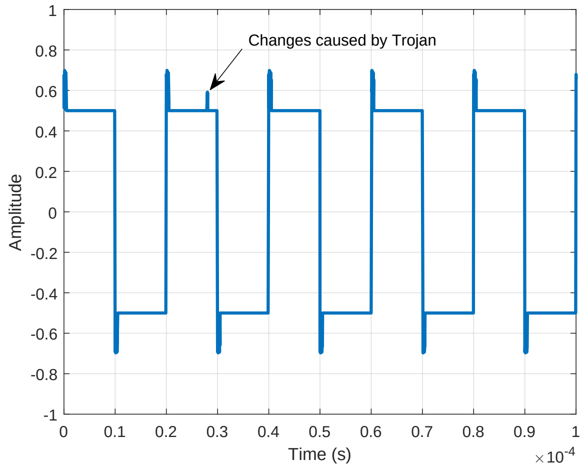


Fig. 5. An example of a clock signal affected by hardware Trojan.

(STFT) on time-domain signal and observe which frequency components of the time domain signal are affected when dormant HT is present. Fig. 6 shows Trojan-free and Trojan-affected clock signals in the frequency domain by taking FFT of signals given in Fig. 4 and Fig. 5, respectively. The signals in the frequency domain are much easier to measure, and the noise power is very small because of focusing a single frequency bin at a time. As a result, instead of measuring the time domain signal, we measure multiple harmonics of the clock in the frequency domain to observe changes in sub-clock samples for HT detection.

The change caused by HTs will be reflected in backscattered signals at the circuit's clock harmonics: $f_{carrier} \pm f_c$, $f_{carrier} \pm 2 * f_c$, etc. The first clock harmonic at $f_{carrier} \pm f_c$ follows the overall RCS change during a cycle, while the remaining harmonics are affected by the rapidity of change (rise/fall times), and timing of the impedance changes within the clock cycle. For each circuit, we measure the amplitude of the first *N* harmonics of the clock from its backscattering side-channel signals to form a vector, which characterizes the circuit's overall amount, timing, and duration of impedance-change activity during a clock cycle. If there is a hardware Trojan in the circuit, this vector will be different from the ones recorded from an HT-free same circuit. As a result, we can represent each circuit by a vector of *N*

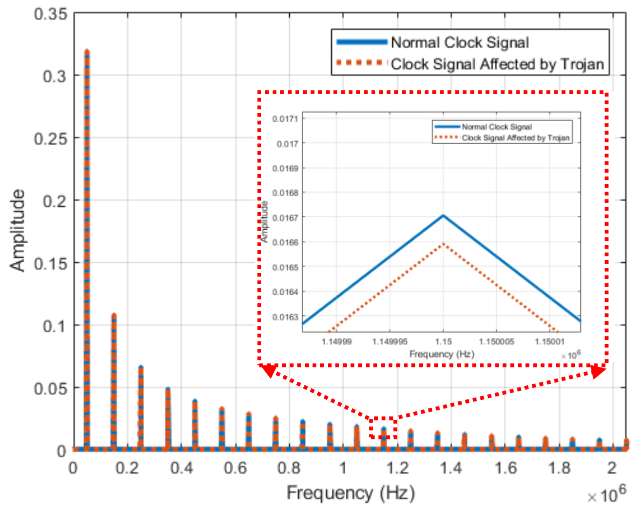


Fig. 6. Trojan-free and Trojan-affected clock signals in frequency domain generated by fast Fourier transforming time domain signals in Fig. 4 and Fig. 5, respectively.

points, which are the amplitudes of the first N harmonics of the clock from its backscattering side-channel signals: $\mathbf{h} = [h_1, h_2, ..., h_{N-1}, h_N]$, where h_j is the amplitude of the j^{th} harmonic of the clock. These vectors will be used as inputs for our clustering algorithm.

If changes caused by HTs in the time-domain signal become briefer in duration, the changes among clock harmonics become smaller in magnitude and shift to higher harmonics which, compared to lower harmonics, tend to be affected more by noise. This is one of the reasons why the backscattering side-channel works better for HT detection than other traditional analog side-channels such as EM and Power side-channels. The backscattering side-channel is a consequence of the impedance changes in switching digital switching circuits, which is caused by the transistors' two-state impedances reflecting a modulated signal. For each gate that switches, the impedance change persists for the rest of the cycle. On the other hand, the EM and power side-channels are consequences of the variation of the current flow in a circuit. As a gate switches, the current will be charged or discharged quickly, which means a current burst occurs for a very short period of time.

B. Graph Model for Clustering Results

This section presents the proposed methodology to categorize ICs into clusters based on how HTs (if present) affect their backscattering side-channel signals. Here, we assume

$$\mathbf{y}_i = \begin{bmatrix} y_{i1} & y_{i2} & \cdots & y_{i(N-1)} \end{bmatrix}$$

to be a vector containing the amplitude ratios of harmonics for the i^{th} board such that

$$y_{ij} = 10 * \log_{10}(\mathbf{h}_i(j+1)/\mathbf{h}_i(j)), \tag{1}$$

where $\mathbf{h}_i \in \Re^N$ is a vector containing the harmonic amplitudes for the i^{th} board. We use the amplitude ratio instead of the amplitude itself to cancel out the attenuation caused by the distance that affects all harmonics. We convert harmonic ratios from linear-domain to dB-domain to prevent the magnitude dominance of the top ratios, and to increase the effect of small harmonic ratios. Matrix $\mathbf{Y}$ is the matrix containing the harmonic ratios of all boards which can be written as

$$\mathbf{Y} = \begin{bmatrix} — & \mathbf{y}_1 & — \\ — & \mathbf{y}_2 & — \\ & \vdots & \\ — & \mathbf{y}_M & — \end{bmatrix}, \tag{2}$$

where M is the number of boards. The objective is to reveal the hidden information that could be crucial to identifying Trojans in the data by removing the redundant information. A popular technique to reduce the dimensionality of the problem is to keep the significant information by applying Principle Components Analysis (PCA). These methods are especially practical for classification when the data exhibits linear characteristics. To utilize these ideas, the first step is to obtain the singular value decomposition (SVD) of $\mathbf{Y}$ which can be written as

$$\mathbf{Y} = \mathbf{U\Sigma V^T}. \tag{3}$$

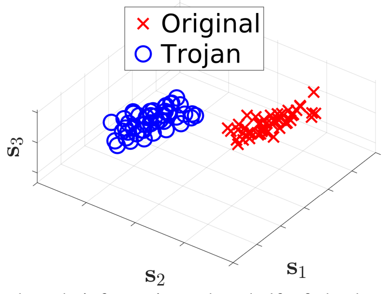


Fig. 7. Ground truth information when half of the boards are randomly injected with a Trojan.

Here, we assume that the first m singular values are the largest m singular values of the matrix $\mathbf{Y}$, and $\mathbf{V}_m$ is a sub-matrix with the first m columns of $\mathbf{V}$ corresponding to these m singular values. Therefore, to reduce the size of the data, we project $\mathbf{Y}$ onto the column space of $\mathbf{V}_m$ as

$$\mathbf{Y}_P = \mathbf{YV}_m. \tag{4}$$

Here, the value of m is selected so that the power of the projected data is very close to the power of $\mathbf{Y}$, i.e.,

$$\|\mathbf{Y}_P\|_F / \|\mathbf{Y}\|_F \approx 1, \tag{5}$$

where $\| \bullet \|_F$ is the Frobenius norm of its argument. For example, in Fig. 7, we plot the projected data when $m = 3$, where $\mathbf{Y}_P$ captures 99% of the power of $\mathbf{Y}$, and when half of the boards are infected with a Trojan. Here, $\mathbf{s}_j$ denotes the singular value direction corresponding to j^{th} largest singular value.

After discarding the redundant information, the next step is to find the clusters in the data. The expectation is that each cluster corresponds to different board groups due to production variability, or existence of a Trojan. To find the clusters and corresponding centroid points, we utilize k-means algorithm. The algorithm requires the number of expected clusters, N_C, and their initial locations, $\mathbf{L}_C \in \Re e^{N_C \times m}$ (each row represents the location of the corresponding cluster), as inputs. A careful selection of the initial cluster locations is important to avoid algorithm to converge to a local optimum. To accomplish that, we apply the following procedure to initiate the k-means algorithm:

1.) Choose a random sample from the projected data as the location of the first cluster.
2.) Find a sample whose total distance is the furthest away from the previously chosen clusters.
3.) Repeat until all centroids are initialized.

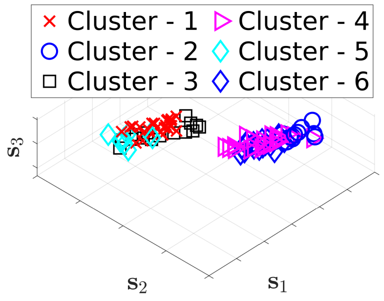


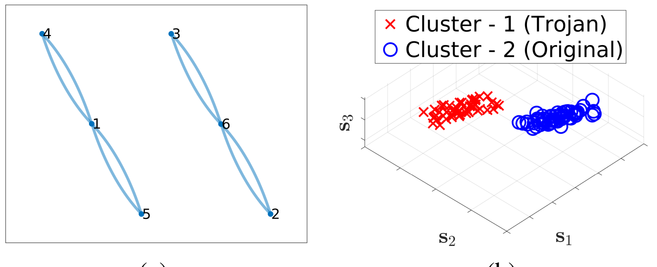
Fig. 8. K-means clustering of the boards when the number of center points is chosen to be six.

The procedure ensures wide separation of the centroids. We need to note here that, N_C is assumed to be larger than actual number of clusters in the data, i.e., larger than the number of Trojan types. The assumption follows the fact that we have no information on how many types of Trojan may exist in the testing devices in a realistic scenario. However, having more than the number of actual clusters can be misleading because it can raise suspicion even when there is no Trojan-affected board in the sample space. For example, in Fig. 8, we plot the results of the algorithm when $N_C = 6$ for the data given in Fig. 7. Comparing the actual labels given in Fig. 7, we observe that there is no cluster that contains both the original and Trojan-affected circuits. Therefore, we require a method that decreases the number of clusters to reveal the existence of Trojan-affected circuits more reliably.

To decrease the number of clusters, we propose to use graph method and the shortest path algorithm. To accomplish that, we create a graph where two centroids belong to the same group if they are at the edges of the same arc. Please note that "group" indicates the Trojan type or whether the board is Trojan-affected. Our proposition is that the group of two closest clusters are the same if the distance of these clusters are below some threshold. In other words, the constraint on arcs is that an arc is valid only if the distance between the cluster centroids at the edges is smaller than a given threshold. In that respect, the first step is to obtain a threshold automatically. We can summarize the process of choosing the threshold as follows:

1.) Calculate the distance among centroids.
2.) Choose the closest two clusters for each cluster, and keep the distances in a list.
3.) Assign threshold as the mean distance of this list.

To illustrate how algorithm works, the graph created by the algorithm is shown in Fig. 9 (a) for the clusters in Fig. 8. The nodes corresponding to the same classes are connected. After generating the graph and identifying the valid arcs, the final step is to check whether a node is reachable from other nodes. If there exists a path between any two nodes, we label these as the same type, otherwise, we decide that the sample space contains at least two clusters, therefore, some boards are Trojan-affected. To obtain the connected nodes automatically, we exploit the shortest path algorithm [36] to check whether a node, i.e., a cluster, is reachable from another node. The algorithm returns null if there is no path between two given nodes, and a path if these two nodes are reachable. Based on the outcome of the shortest path, we relabel the sample space indicating whether the connected nodes belong to the same kind. An example of the process is given in Fig. 9 (b). We observe that although the exact identity of these classes are not known, it is possible to divide data into two groups, and therefore, to determine that the batch contains two circuit designs that are not identical, or some of the boards are Trojan-affected.



(a) (b)

Fig. 9. a) Generation of the graph based on the distances between the centroids of the clusters, b) Clustering the data into two groups as Trojan injected vs. no-Trojan-free boards. Labels inside the parenthesis indicate the ground truth.

V. Experimental Setup and Testing Scheme Formulation

A. Experimental Setup

The experimental setup to evaluate the performance of the proposed algorithm is shown in Fig. 10. The setup includes a transmitter Aaronia E1 electric-field near-field probe [37] connected to an Agilent MXG N5183A signal generator [38], and a receiver Aaronia H2 magnetic-field near-field probe [37] connected to an Agilent MXA N9020A spectrum analyzer [39]. The devices-under-test (DuT) are Altera DE0 Cyclone V FPGA boards [40]. An angle ruler is used as a positioner so that different DE0-CV boards can be tested using approximately the same probe positions. A laptop is used to control the devices and automate the measurements. A 3 GHz continuous sinusoid signal is generated by the signal generator, and backscattered signals are recorded by the spectrum analyzer. The measurements are carried in an open environment setup at the room temperature. The effect of environmental conditions such as temperature and voltage source, if existed, should be the same for all clock harmonics and our technique is based on the ratio between clock harmonics. As a results, environmental conditions do not significantly affect the accuracy of our technique

We choose FPGA instead of taping out ASICs for evaluation because it is much more flexible, time, and cost effective. Although we only evaluate this on FPGAs, there is no reason

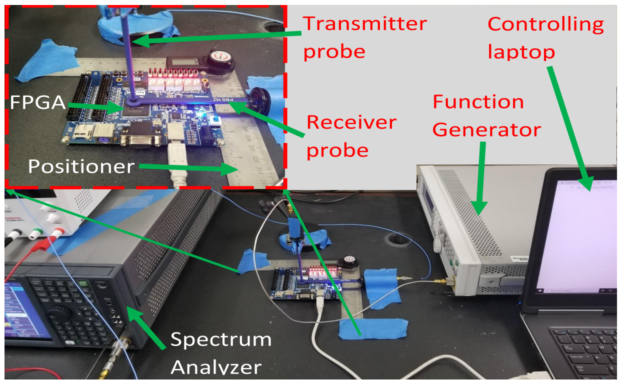


Fig. 10. Measurement setup for IC clustering using backscattering side-channel collection for HT detection.

TABLE I
HARDWARE TROJAN BENCHMARKS AND DETECTION RESULTS

Benchmark	**Size of Trojan (Percentage of HT-free circuit)**		
	Trigger	Payload	Total
AES-T1200	0.32%	1.61%	1.93%
AES-T500	0.28%	1.51%	1.79%
AES-T700	0.27%	1.76%	2.03%
PIC16F84-T100	1.34%	1.81%	3.15%
PIC16F84-T300	1.37%	1.96%	3.33%
PIC16F84-T400	1.35%	1.75%	3.10%
RS232-T300	1.47%	1.58%	3.05%
RS232-T600	1.50%	1.48%	2.98%
RS232-T901	1.53%	1.61%	3.11%

to believe that the results can not be generalized to ASICs. For examples, although the same gate-level design will be smaller in an ASIC, the backscattered signal corresponds to the relative change in impedances, and the relative change of impedances tends to be larger for smaller circuits, i.e. if the overall circuit gets smaller in the FPGA-to-ASIC transition and the HT's trigger gets proportionally smaller, a backscattering-based approach will work as well or possibly even better.

B. Hardware Trojan Benchmark Implementation

To evaluate our technique, we implement three different benchmark circuits AES, RS232, and PIC16F84 from the TrustHUB Trojan repository [41]. There are total of 21 Trojan designs for AES circuit, 4 Trojan designs for PIC16F84 circuit, and 21 Trojan designs for RS232 circuit. Because numerous HTs in the TrustHub repository are similar to each other, we select circuits that exhibit different approaches for their triggers and payloads. Each of these Trojans has a different triggering mechanism such as observing a specific sequence of the input, counting number of encryption rounds, observing the number of execution of a specific instruction, etc., and performs a different payload functionality such as shortening the hardware lifetime, leaking private keys, changing the address to program memory, etc. Table I summarizes the benchmarks we use.

The Trojan-affected and Trojan-free designs are carefully mapped to the FPGA by using ECO (Engineering Change Order) tools so that they have the same layout except for the Trojan part, thus making for a fair comparison. As mentioned in Section II, it is extremely hard to activate an HT without a priori knowledge of its triggering circuit, it is highly desirable for an HT detection technique to be able to detect HT when it is dormant. As a result, our evaluation focuses on evaluating our algorithm for dormant HTs. In other words, all Trojans stay inactive in all experiments.

C. Testing Scheme Formulation

All HT benchmarks are implemented on Altera DE0 Cyclone V FPGA, and we test 100 boards by randomly infecting the boards. To prototype a real testing environment, for each HT benchmark, we randomly program each of the 100 boards with HT-free or HT-infected designs and record its backscattering side-channel signals while the board is running. For each board, we extract the amplitude of the first 40 harmonics of the clock from its backscattering side-channel signal. We only use 40 harmonics because the higher harmonics are very weak and below the noise level. As a result, for each hardware Trojan benchmark, we will have a set of 100 traces, in which each trace contains 40 points, denoted as follow: $\mathbf{h}_i$ = $[h_{i1}, h_{i1}, ..., h_{iN-1}, h_{iN}]$, where $N = 40$, and $1 \leq i \leq 40$. Our clustering algorithm takes these traces as inputs to cluster the ICs.

VI. EVALUATION

A. Evaluation of Existing HT Benchmarks

In this section, we provide the experimental results for Trojan detection. The process can be summarized as follows:

- $\rightarrow$ Collect the data from all boards with the setup given in Fig. 10. The number of boards tested for the experiments is 100.
- $\rightarrow$ Take the ratios of the consecutive harmonics, and convert them into dB-domain.
- $\rightarrow$ Collect the harmonic ratios for all boards in a matrix to generate $\mathbf{Y}$.
- $\rightarrow$ Obtain SVD of $\mathbf{Y}$, and project it into the space defined by the right-singular vectors corresponding to the largest m singular values to generate $\mathbf{Y}_P$. Here, m is chosen such that it is the smallest number of singular values satisfying the following equation:

$$\|\mathbf{Y}_P\|_F / \|\mathbf{Y}\|_F \approx 0.999. \tag{6}$$

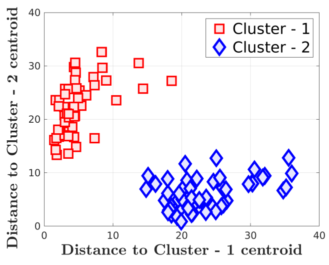
- $\rightarrow$ Apply the k-means algorithm by ensuring N_C is larger than the number of possible Trojan types. The initialization of the centroids are done based on the procedure given in Section 2.
- $\rightarrow$ Generate the graph of similarity with respect to the threshold calculated in Section 2.
- $\rightarrow$ Apply the shortest path algorithm to reveal possible classes in the sample space. If the algorithm returns more than one cluster, the batch of boards contains some Trojan-affected boards.

Since the goal of the paper is to separate the Trojan-free designs from all other Trojan-affected designs, we define the accuracy of the measurements as

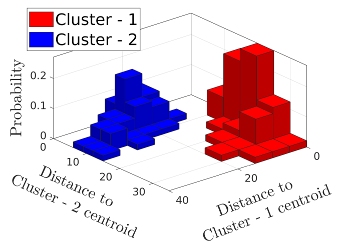
$$\texttt{accuracy}\ (\%) = \frac{\text{\# of correct labeling whether the design is original}}{\text{\# of measurements}} \times 100.$$

Please note that the actual labels of the circuits are only required to calculate the accuracy of the proposed method. Therefore, after having the outcome of the procedure given above, we first identify the group which contains the most of the original designs, and then label this group as the Trojan-free and the rest as the Trojan-affected circuits. Finally, we compare our labels with the actual labels to calculate the accuracy. If the proposed method classifies all the original designs in a cluster, and if this cluster does not contain any samples from Trojan-affected designs, the accuracy of the algorithm will be equivalent to 100%.

The tested designs are given in Table I. We first work on PIC16F84 circuit with 3 different Trojan designs. The results are plotted by considering the singular vectors corresponding largest three singular values. The outcome of the procedure is given in Fig. 11 (1a-1d). The figures in Fig. 11 (1a-1c) correspond to the scenarios when the batch contains only one Trojan type. However, Fig. 11 (1d) includes samples from all Trojan designs. The number of singular values used for these experiments that satisfies the condition given in (6) is 10, and $N_C = 6$. We also plot the sample distances to each cluster centroid in Fig. 12 (a) and their distribution in Fig. 12 (b) for the samples given in Fig. 11 (1a). The mean distances of Cluster - 1 samples to the centroids are 4.96 and 22.27 with standard deviations 3.47 and 5.03, whereas mean distances of Cluster - 2 samples are 23.39 and 6.08 with standard deviations 5.46 and 2.95, respectively. We achieve **100%** accuracy for all of the experiments. We need to note here that the legends of the figures do not give any information whether the group is Trojan-affected or original. They only provide the information that the sample space contains two different groups, hence, one of these groups represents the designs with Trojan. However, we provide the actual labels of the classes in parentheses for a better illustration.



(a)



(b)

Fig. 12. a) Distances of each circuit to the cluster centroids. b) Distribution of distances of each circuit to each cluster centroid.

The other experiments are done with AES and RS232 circuits. Similarly, the results are shown in Fig. 11 (2a-2d) and Fig. 11 (3a-3d) for AES and RS232, respectively. The

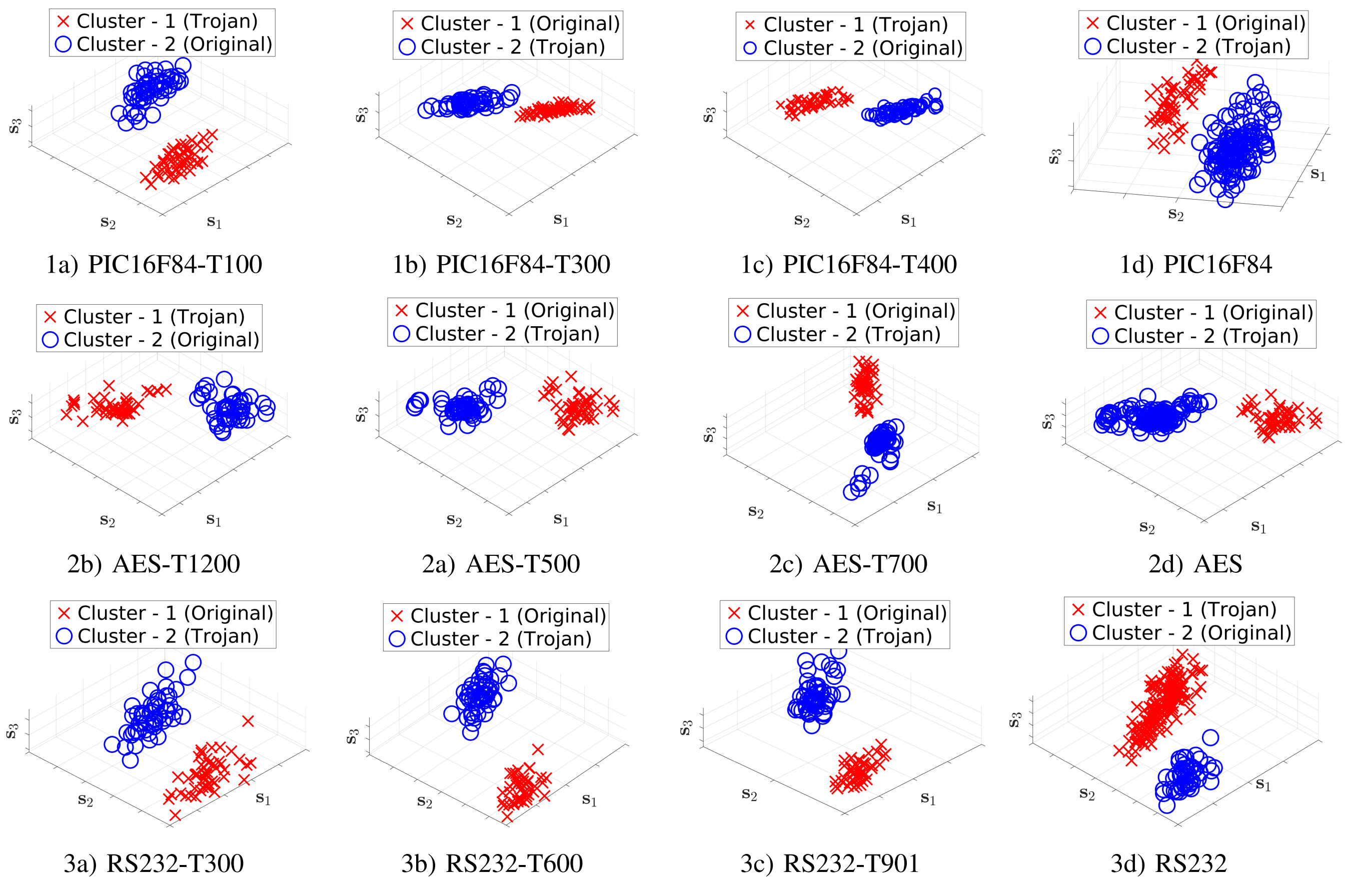


Fig. 11. Separation of the Trojan-free and the Trojan-affected circuits. First three columns contain the plots when only one Trojan exists, and the last column of figures are when all considered Trojans exist in the sample space.

plots in Fig. 11 (2a-2c) and in Fig. 11 (3a-3c) correspond the experiments when the board batch contains only one Trojan design type for AES and RS232, respectively. The experiments with all considered Trojan designs are shown in Fig. 11 (2d) and in Fig. 11 (3d). We keep the number of clusters, N_C, same for PIC16F84 circuit. This time, the number of singular-values satisfying the equation given in (6) corresponds to 12 for each circuit. Similarly, we obtain **100%** accuracy for all these experiments meaning that all the original circuits are separated from the designs that is Trojan-affected, and clustered in a single group.

From the results, we can make the following observations:

I) The backscattering side channel is a powerful mechanism to detect the existence of a Trojan when the ratios of the harmonics are exploited since the separation between the Trojan-free and Trojan-affected circuits are significant.

II) The proposed methodology (backscattered signal plus PCA and k-means algorithm) enables perfect clustering of the Trojan-free and Trojan-affected circuits.

III) When multi-Trojan designs are considered, they still behave like a single group, and the proposed method can successfully distinguish the existence of at least two different classes.

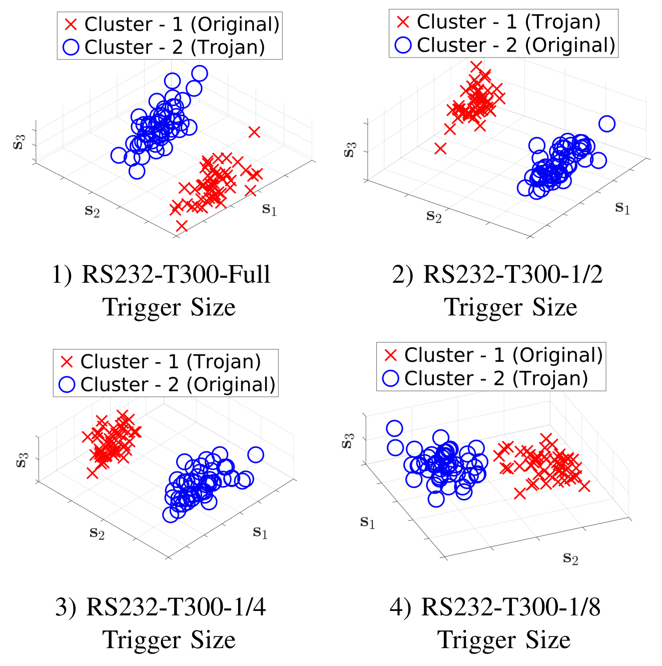


Fig. 13. Separation of the Trojan-free and the Trojan-affected circuits when the size of RS232-T300 varies.

B. *Evaluation of Changing Size of Hardware Trojan Triggers*

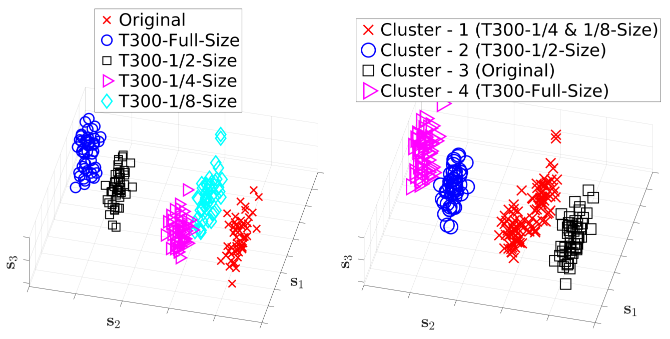
TABLE II
HARDWARE TROJAN BENCHMARKS AND DETECTION RESULTS

Benchmark	**Size of Trojan's Trigger (Percentage of HT-free circuit)**
RS232-T300 w/ 1/2 Trigger Size	0.76%
RS232-T300 w/ 1/4 Trigger Size	0.39%
RS232-T301 w/ 1/8 Trigger Size	0.19%

Because the algorithm performs so well on the existing HT designs in Table I, this section focuses on testing the limit of our algorithm by reducing the size of HTs. The authors in [11] demonstrated that only the trigger is active while the payload stays inert when hardware Trojans are dormant, thus if the trigger is big enough, the Trojans can be detected regardless of its payload size. Therefore, we will focus on changing the size of the trigger to test the limits of the proposed algorithm. The RS232-T300 is chosen for this experiment because the trigger of the Trojan can be meaningfully resized. We change the size of the trigger of RS232-T300 while keeping its payload the same to create test designs that are summarized in Table II.

The first goal is to investigate whether the proposed method still works when only one HT benchmark exists in the board batch. The same parameters with the experiments given in Section VI-A are used for the number of clusters and singular vectors. The clustering results are given in Fig. 13. We again obtain **100%** accuracy in terms of separating the original circuits from the Trojan-affected ones. Here, one important observation is that as the size of the Trojan trigger decreases, the distance between centroids of the two classes decreases, i.e. the Trojan does become more similar to the original circuit when it has only 1/8 trigger size compared to a full-size trigger. To illustrate this, we show clustering results where the measurements from all trigger sizes were included, i.e., five different designs, one HT-free and four variants of an HT-infected design (with diferent trigger sizes), are subjected to our clustering technique. The results are shown in Fig. 14, with actual (ground-truth) labels (left) and with clustering-produced labels (right). In terms of separating HT-free from HT-infected designs, the accuracy of this clustering is still **100%** (all HT-free instances are in one cluster while all HT-infected instances are in other clusters). Furthermore, the technique is able to distinguish (put in separate clusters) different variants of the HT, except for the variants with 1/4 and 1/8 triggers, which are in the same cluster. We note that the technique is able to distinguish the 1/8-trigger variant from an HT-free design, even though it did not distinguish 1/4- from the 1/8-trigger variant (the difference among them is also 1/8 of the full trigger). This is because the additional trigger activity in the 1/4 variant is similar to the trigger activity in the 1/8 variant, i.e., it is only a matter of *how much* trigger activity the design has. In contrast, the HT-free design has no trigger activity at all, so the presence of trigger activity in the 1/8 design allows it to be well-separated from the HT-free design. This implies that HTs whose circuitry and activity mimics that of the original design would be more difficult to detect, but only up to a point – even such activity-mimicking HTs would be detected if they are sufficiently large (in this particular experiment, larger than 0.19% of the original circuit).

Based on the results given in this section and Section VI-A, our main observation is that our technique successfully separates HT-free from HT-infected designs, even for very small HTs (0.19% of the original circuit, in our experiments). Additionally, the technique successfully separates different HT designs from each other, except when the HTs only differ



1) RS232-T300 Ground Truth   2) RS232-T300 Estimates

Fig. 14. Separation of original and Trojan-affected circuits when the size of RS232-T300 varies. The experiments are performed with original, full-Trigger-size, 1/2-Trigger-size, 1/4-Trigger-size, and 1/8-Trigger-size circuits.

in size (but not nature) of their trigger circuitry, and that difference in size is very small (0.19% of the original circuit, in our experiments).

VII. Related Work

Over the past few years, as hardware Trojan has emerged as an increasingly dangerous threat, a number of HT detection techniques using side-channel analysis have been proposed. The authors in [11] propose a novel method to detect hardware Trojans in the fabricated ICs by creating a backscattering side-channel. The results show that their method can detect dormant hardware Trojans with 100% accuracy and 0% false positives. However, similar to the majority of other side-channel techniques, their approach requires having a verified HT-free chip. In [8], the authors present a method using EM to detect HTs without having a golden circuit by modeling the benchmark circuits they used for testing. They have simulated the models to generate EM traces for the circuit and compare them with the measured ones to detect HTs with no HT-free chip. However, in the paper, the authors only test their technique on a single FPGA board, thus the hardware manufacturing variations are not verified. Furthermore, they only evaluate their techniques with activated hardware Trojans, which is also not practical because it is extremely difficult to activate HTs without a priori knowledge of their circuitry and activation mechanisms. In addition, the technique requires a priori knowledge of the chip circuitry, heavily depends on the accuracy of the model and the simulator that generate the reference signals, and might not work for other circuits that are not modeled in the paper.

As machine learning has become prevalent over the last decade, a number of papers exploiting clustering techniques for HT detection have been proposed. In [42], the authors exploit the support vector machine (SVM) and K-means clustering approach to provide automatic layout identification in their reverse engineering-based detection method. The technique does not rely on a golden sample; however, because the nature of reverse engineering is extremely costly and time-consuming, it is not realistic to assume having a large set of data for clustering. The methods in [43], [44] propose a low overhead clustering-based detection technique for runtime Trojan detection. However, the methods need golden samples for training and are only capable of detecting activated HTs. The authors in [45] propose a technique using the AdaBoost Meta-Learner algorithm based on automatic feature selection using Haar-like functions to assist in reverse engineering detection. However, the method also requires to have golden samples.

Only a few clustering techniques can eliminate the need for golden samples [21]-[23]. The authors in [21] present an information-theoretic approach that estimates the statistical correlation between the signals in a design and then use a weight normalization and clustering algorithm to detect HTs. In [22], the authors propose COTD, an HT detection technique based on analyses of the controllability and observability of gate-level netlist and utilizing an unsupervised clustering to detect HTs by exploiting significant inter-cluster distance caused by the controllability and observability characteristics of Trojan gates. [23] proposes a technique based on “outliers”, a procedure to identify suspicious signals in a netlist, and clustering technique to detect HTs. However, all of these methods are pre-silicon approaches, which means that they can not detect HTs inserted in the fabrication stage. A post-silicon clustering technique using side-channel analysis has been proposed in [24], but authors only test their method on a set of two FPGA, which does not give enough statistics to evaluate manufacturing variations among different hardware instances. One of the main challenges of techniques using side channels with external-measurement is that the variation across different hardware instances may cloud the difference caused by hardware Trojans. Therefore, detection accuracy normally decreases dramatically when testing across multiple hardware instances. In addition, the technique uses power side-channels, which provide very limited resolution and bandwidth [11]. As a result, the technique only gives 93.75% accuracy for HT benchmarks from Trusthub, even when testing with only two different FPGA boards.

VIII. Conclusion

This paper proposes a novel golden-chip-free method for clustering fabricated integrated circuits into groups for deployment of reverse engineering based hardware Trojan detection techniques to a large population of ICs. Our technique classifies boards into clusters based on how hardware Trojans (if existed) affect their backscattering side-channel signals. Unlike prior clustering approaches, the paper uses the backscattering side-channel, which has been shown to work better for hardware Trojan detection than other side-channels. We test the proposed algorithm on a set of 100 boards to thoroughly evaluate manufacturing variations among different hardware instances. This approach requires no prior knowledge about the chip or Trojan circuitry to cluster ICs into groups for HT detection. The results show that our technique can tolerate manufacturing variations among hardware instances to cluster all boards correctly for not only 9 different dormant Trojan designs on 3 different benchmark circuits from Trusthub, but also dormant Trojan designs whose trigger size is shrunk to as small as 0.19% of the original circuit.

References

[1] R. Torrance and D. James, "The state-of-the-art in ic reverse engineering," in *Cryptographic Hardware and Embedded Systems-CHES 2009*. Springer, 2009, pp. 363–381.

[2] A. A. Nasr and M. Z. Abdulmageed, "An efficient reverse engineering hardware trojan detector using histogram of oriented gradients," *Journal of Electronic Testing*, vol. 33, no. 1, pp. 93–105, 2017.

[3] M. Fyrbiak, S. Wallat, P. Swierczynski, M. Hoffmann, S. Hoppach, M. Wilhelm, T. Weidlich, R. Tessier, and C. Paar, "Hal—the missing piece of the puzzle for hardware reverse engineering, trojan detection and insertion," *IEEE Transactions on Dependable and Secure Computing*, vol. 16, no. 3, pp. 498–510, 2018.

[4] C. Bao, D. Forte, and A. Srivastava, "On reverse engineering-based hardware trojan detection," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 1, pp. 49–57, Jan 2016.

[5] S. Wallat, M. Fyrbiak, M. Schlögel, and C. Paar, "A look at the dark side of hardware reverse engineering - a case study," in *2017 IEEE 2nd International Verification and Security Workshop (IVSW)*, July 2017, pp. 95–100.

[6] C. Bao, D. Forte, and A. Srivastava, "On application of one-class svm to reverse engineering-based hardware trojan detection," in *Fifteenth International Symposium on Quality Electronic Design*, March 2014, pp. 47–54.

[7] X. Wei, Y. Diao, and Y. Wu, "To detect, locate, and mask hardware trojans in digital circuits by reverse engineering and functional eco," in *2016 21st Asia and South Pacific Design Automation Conference (ASP-DAC)*, Jan 2016, pp. 623–630.

[8] J. He, Y. Zhao, X. Guo, and Y. Jin, "Hardware trojan detection through chip-free electromagnetic side-channel statistical analysis," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 10, pp. 2939–2948, 2017.

[9] R. Vaikuntapu, L. Bhargava, and V. Sahula, "Golden ic free methodology for hardware trojan detection using symmetric path delays," in *2016 20th International Symposium on VLSI Design and Test (VDAT)*, May 2016, pp. 1–2.

[10] Y. Tang, S. Li, L. Fang, X. Hu, and J. Chen, "Golden-chip-free hardware trojan detection through quiescent thermal maps," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, pp. 1–12, 2019.

[11] L. N. Nguyen, C. Cheng, M. Prvulovic, and A. Zajić, "Creating a backscattering side channel to enable detection of dormant hardware trojans," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 27, no. 7, pp. 1561–1574, July 2019.

[12] D. Agrawal, S. Baktir, D. Karakoyunlu, P. Rohatgi, and B. Sunar, "Trojan detection using ic fingerprinting," in *Security and Privacy, 2007. SP'07. IEEE Symposium on*. IEEE, 2007, pp. 296–310.

[13] M. Banga and M. S. Hsiao, "A region based approach for the identification of hardware trojans," in *Hardware-Oriented Security and Trust, 2008. HOST 2008. IEEE International Workshop on*. IEEE, 2008, pp. 40–47.

[14] B. Hou, C. He, L. Wang, Y. En, and S. Xie, "Hardware trojan detection via current measurement: A method immune to process variation effects," in *2014 10th International Conference on Reliability, Maintainability and Safety (ICRMS)*, Aug 2014, pp. 1039–1042.

[15] C. Bao, D. Forte, and A. Srivastava, "Temperature tracking: Toward robust run-time detection of hardware trojans," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 10, pp. 1577–1585, 2015.

[16] X. T. Ngo, Z. Najm, S. Bhasin, S. Guilley, and J.-L. Danger, "Method taking into account process dispersion to detect hardware trojan horse by side-channel analysis," *Journal of Cryptographic Engineering*, vol. 6, no. 3, pp. 239–247, 2016.

[17] K. Hu, A. N. Nowroz, S. Reda, and F. Koushanfar, "High-sensitivity hardware trojan detection using multimodal characterization," in *Proceedings of the Conference on Design, Automation and Test in Europe*. EDA Consortium, 2013, pp. 1271–1276.

[18] A. N. Nowroz, K. Hu, F. Koushanfar, and S. Reda, "Novel techniques for high-sensitivity hardware trojan detection using thermal and power maps," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 33, no. 12, pp. 1792–1805, 2014.

[19] B. Cha and S. K. Gupta, "Trojan detection via delay measurements: A new approach to select paths and vectors to maximize effectiveness and minimize cost," in *Proceedings of the conference on design, automation and test in Europe*. EDA Consortium, 2013, pp. 1265–1270.

[20] M. Lecomte, J. Fournier, and P. Maurine, "An on-chip technique to detect hardware trojans and assist counterfeit identification," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 12, pp. 3317–3330, 2017.

[21] B. Çakir and S. Malik, "Hardware trojan detection for gate-level ics using signal correlation based clustering," in *Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition*. EDA Consortium, 2015, pp. 471–476.

[22] H. Salmani, "Cotd: reference-free hardware trojan detection and recovery based on controllability and observability in gate-level netlist," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 2, pp. 338–350, 2017.

[23] P.-S. Ba, S. Dupuis, M.-L. Flottes, G. Di Natale, and B. Rouzeyre, "Using outliers to detect stealthy hardware trojan triggering?" in *Verification and Security Workshop (IVSW), IEEE International*. IEEE, 2016, pp. 1–6.

[24] M. Xue, R. Bian, W. Liu, and J. Wang, "Defeating untrustworthy testing parties: A novel hybrid clustering ensemble based golden models-free hardware trojan detection method," *IEEE Access*, 2018.

[25] S. Bhunia, M. S. Hsiao, M. Banga, and S. Narasimhan, "Hardware trojan attacks: threat analysis and countermeasures," *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1229–1247, 2014.

[26] B. Shakya, T. He, H. Salmani, D. Forte, S. Bhunia, and M. Tehranipoor, "Benchmarking of hardware trojans and maliciously affected circuits," *Journal of Hardware and Systems Security*, vol. 1, no. 1, pp. 85–102, 2017.

[27] M. Tehranipoor and F. Koushanfar, "A survey of hardware trojan taxonomy and detection," *IEEE design & test of computers*, vol. 27, no. 1, pp. 10–25, 2010.

[28] R. S. Chakraborty, S. Narasimhan, and S. Bhunia, "Hardware trojan: Threats and emerging solutions," in *High Level Design Validation and Test Workshop, 2009. HLDVT 2009. IEEE International*. IEEE, 2009, pp. 166–171.

[29] J. Zhang, F. Yuan, and Q. Xu, "Detrust: Defeating hardware trust verification with stealthy implicitly-triggered hardware trojans," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2014, pp. 153–166.

[30] Z. Chen, X. Guo, R. Nagesh, A. Reddy, M. Gora, and A. Maiti, "Hardware trojan designs on basys fpga board," *Embedded system challenge contest in cyber security awareness week-CSAW*, 2008.

[31] R. S. Chakraborty, I. Saha, A. Palchaudhuri, and G. K. Naik, "Hardware trojan insertion by direct modification of fpga configuration bitstream," *IEEE Design & Test*, vol. 30, no. 2, pp. 45–54, 2013.

[32] R. Karri, J. Rajendran, K. Rosenfeld, and M. Tehranipoor, "Trustworthy hardware: Identifying and classifying hardware trojans," *Computer*, vol. 43, no. 10, pp. 39–46, 2010.

[33] X. Wang, M. Tehranipoor, and J. Plusquellic, "Detecting malicious inclusions in secure hardware: Challenges and solutions," in *Hardware-Oriented Security and Trust, 2008. HOST 2008. IEEE International Workshop on*. IEEE, 2008, pp. 15–19.

[34] C. Cheng, L. N. Nguyen, M. Prvulovic, and A. Zajić, "Exploiting switching of transistors in digital electronics for rfid tag design," *IEEE Journal of Radio Frequency Identification*, vol. 3, no. 2, pp. 67–76, June 2019.

[35] L. N. Nguyen, C. Cheng, M. Prvulovic, and A. Zajić, "Hardware trojan detection using backscattering side channel," in *Hardware-Oriented Security and Trust, 2019. HOST 2019. IEEE International Workshop on*. IEEE, 2019.

[36] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*, 2009.

[37] [Online]. Available: http://www.aaronia.com/products/antennas/Near-Field-Probe-Set-PBS2

[38] [Online]. Available: https://www.keysight.com/en/pdx-x201724-pn-N5183A/mxg-microwave-analog-signal-generator-100-khz-to-40-ghz?pm=spc&nid=-32490.1150253&cc=US&lc=eng

[39] [Online]. Available: https://www.keysight.com/en/pdx-x202266-pn-N9020A/mxa-signal-analyzer-10-hz-to-265-ghz?pm=spc&nid=-32508.1150426&cc=US&lc=eng

[40] [Online]. Available: https://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&CategoryNo=167&No=921&PartNo=2

[41] "Trusthub," http://www.trust-hub.org/benchmarks/trojan.

[42] C. Bao, D. Forte, and A. Srivastava, "On reverse engineering-based hardware trojan detection," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 1, pp. 49–57, 2016.

[43] A. Kulkarni, Y. Pino, and T. Mohsenin, "Adaptive real-time trojan detection framework through machine learning," in *2016 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*. IEEE, 2016, pp. 120–123.

[44] ——, “Svm-based real-time hardware trojan detection for many-core platform,” in *Quality Electronic Design (ISQED), 2016 17th International Symposium on*. IEEE, 2016, pp. 362–367.

[45] A. A. Nasr and M. Z. Abdulmageed, “Automatic feature selection of hardware layout: a step toward robust hardware trojan detection,” *Journal of Electronic Testing*, vol. 32, no. 3, pp. 357–367, 2016.