This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2946044, IEEE Transactions on Information Forensics and Security

Trigger Identification Using Difference-amplified Controllability and Dynamic Transition Probability for Hardware Trojan Detection

Yun He, Kai Huang

***Abstract*—To remain dormant in the validation and manufacturing test, Trojans tend to have at least one trigger signal at the gate-level netlist with a very low transition probability. Our paper exploits this stealthy nature of trigger signals to detect Trojans using static and dynamic transition probabilities. The proposed trigger identification is a reference-free scheme, and no prior knowledge of a Trojan-free design is required. First, we reveal the relation between combinational 0/1-controllability and 0/1-probability and propose a static transition probability analysis based on our proposed difference-amplified controllability, which can be easily obtained by the Sandia Controllability/Observability Analysis Program. The k-means clustering method is adopted for potential trigger classification to extend the scalability and adaptability to different circuit sizes. Second, we propose to utilize the transition probability of a dynamic simulation for correction of the results. Experiments show that the proposed detection scheme can obtain a 0% false negative rate and a maximum 11.7% false positive rate on Trust-HUB benchmarks.**

***Index Terms*—Hardware Trojan, static probability analysis, dynamic probability analysis, difference-amplified controllability, k-means clustering**

I. Introduction

WITH the rapidly increasing integration and complexity of integrated circuits, to reduce research and development costs and time to market, the system-on-chip (SoC) design methodology based on intellectual property (IP) reuse has become the mainstream in application-specific integrated circuit (ASIC) design. Alongside the convenience of using IPs provided by third parties, the high risk of malicious code hidden in the IP [1], [2], the so-called hardware Trojans, challenges the whole design.

Globalization of manufacturing is another source of Trojan insertion, in addition to the third-party Trojan attack model. Theoretically, hardware Trojans can be inserted in any phase of the IC design flow, from specification to fabrication and the assembly phase. Fig. 1 illustrates the possible attacks in a typical IC life cycle [3]. The traditional verification and validation flow, such as the functional test and formal verification, can detect some simple Trojans. However, those that are carefully designed and inserted in the design phase or fabrication phase can evade the traditional verification flow. Since third-party IP and fabrication procedures are both outsourced to untrusted business groups, Trojans inserted in these phases will cause a lack of availability of a reference design model or a golden chip for validation. The detection difficulties will further facilitate hardware Trojan attacks. We classify hardware Trojans into three attack models: the 3PIP attack, foundry attack, and interior attack. The third-party IP attack model contains Trojans inserted in third-party IPs or the design house, where the original design has been tampered with and no reference design is available. The foundry attack model refers to Trojans inserted during the fabrication procedure, which means that no reference chip is available. Other Trojans are inserted between design and manufacture, such as in untrusted CAD tool or internal designer insertion, and a reference model is available in some format.

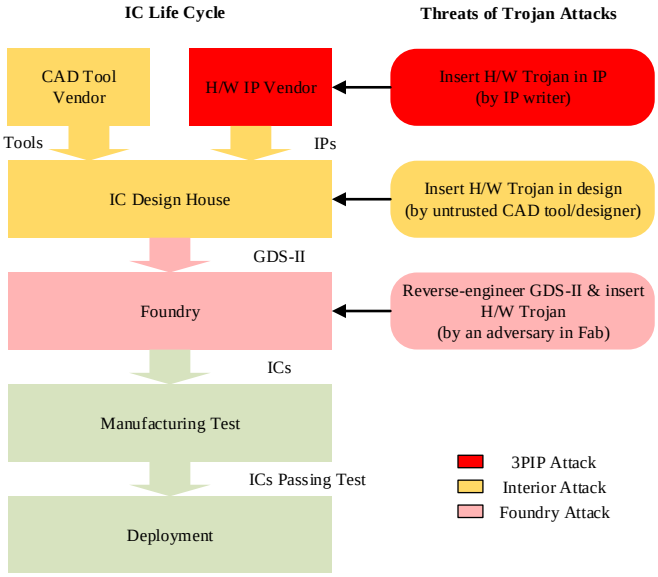

Fig. 1: Hardware Trojan attacks in a typical IC life cycle.

There has been considerable emphasis placed on countermeasures for Trojan detection, and the state-of-the-art detection techniques are illustrated in Fig. 2. According to the design procedure, these techniques can be classified into design-time, test-time and run-time countermeasures. Design-time techniques include logical obfuscation [4]–[7], proof-carrying hardware [8], [9] and auxiliary test logic [10]–[14], which aims at increasing testability and monitoring certain signals. Test-time approaches contain a logic test and side channel analysis. Formal verification [8], [9], [15]–[17], a functional test [16]–[20] and logical or structural feature analysis [21]–[26] are the main techniques used in logic test detection.

Y. He and K. Huang are with the Institute of VLSI Design, Zhejiang University, Hangzhou, 310027, China. E-mail: yun.ee@zju.edu.cn, huangk@vlsi.zju.edu.cn

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2946044, IEEE Transactions on Information Forensics and Security

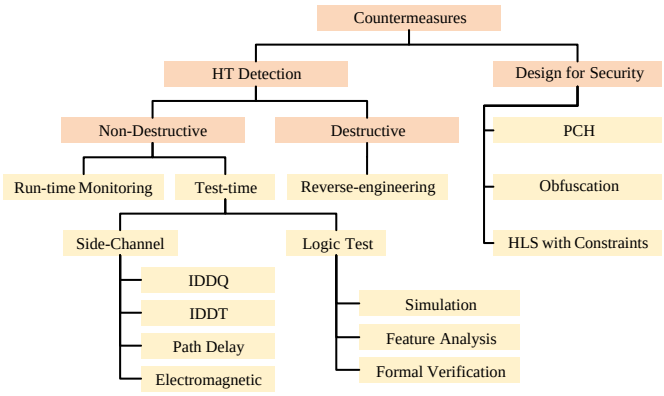

Fig. 2: Countermeasures for hardware Trojan threats.

Side-channel information, such as the current, power, delay and electromagnetic characteristics, can all be used as the circuit signature for Trojan classification [27]–[33]. Run-time methods [20], [21], [33]–[37] often employ auxiliary circuitry in the design time that monitors the behavior or state of a design. Reverse engineering [38] is a destructive technique applied after fabrication, which is often used to construct Trojan-free ICs.

Among these techniques, logic obfuscation is mainly used to prevent attackers from totally comprehending the logic in order to protect the original design, which can increase the difficulty of the interior and foundry attacks. PCH works well with data flow or state transition tracking by encoding certain security properties into the design, which is suitable for detecting data leakage and malicious modifications to registers. The typical scenario of the run-time technique is that the inserted auxiliary logic monitors the critical signals of a design to prevent Trojan infection or the suspicious signals extracted during test time to witness a trigger occurrence. Therefore, the critical work of run-time methods is to label suspicious signal lists. Design-time and run-time techniques can protect the design by increasing the difficulty of Trojan insertion but are not sufficient for Trojan detection. Among test-time detection techniques, side-channel analysis can detect Trojans inserted during fabrication with a golden chip. In addition, some self-calibration structures have been proposed to eliminate the process and environmental variation of side-channel information measurement. Side-channel analysis aims at Trojans inserted in fabrication, so it is significant to have extra detection steps for Trojans inserted before fabrication, which include the threatening 3PIP and interior insertion models. Formal methods, such as equivalence checking and model checking, can be utilized to detect Trojans when a reference model is available, such as interior inserted Trojans. However, for the 3PIP attack model, no Trojan-free design can be obtained. The large design space, low design complexity and high detection difficulty of 3PIP Trojans make reference-free detection techniques indispensable in Trojan detection.

While the interior attacks can be overcame by traditional verification techniques and the self-calibration side-channel analysis approach can relieve the foundry attack to some extent, the reference-free logic test detection for the 3PIP attack model is always challenged by new Trojans. Therefore, our research will focus on logic test techniques that detect 3PIP Trojans, and no reference model or prior knowledge is required. The paper is organized as follows. Section II introduces previous work on reference-free detection. Section III discusses the motivation and contribution of this paper. Section IV presents a trigger model that can prove that high controllability and observability values are not necessarily true for Trojan circuits; thus, simple controllability and observability-based analysis, such as COTD, is insufficient in Trojan detection. Then, we propose to identify the trigger signal by its switching probability, and the static and dynamic transition probability analysis method is proposed in section V. Section VI shows the experimental results on Trust-HUB benchmarks [39]. After that, discussions and conclusions are presented in sections VII and VIII.

II. Previous Work

There have been many reference-free test-time detection schemes proposed to detect 3PIP Trojan insertions, from unused logic identification to structural feature analysis and others. Most of their Trojan models for detection are from their own construction and Trust-HUB website benchmarks [39]. The following discussion describes the main logic test detection techniques presented in the recent literature.

1) UCI: Hicks *et al.* proposed a MUX-based Trojan model [21] that can evade code coverage checks. In addition, the unused circuit identification (UCI) algorithm was proposed as a complement to code coverage. The proposed Trojan model employing MUX-based circuits is acceptable since the basic AND gate and OR gate can be replaced by MUXs. The UCI method considers all paths between every two signals, checks whether the data path was activated in the simulation, and marks any other path as a non-stuck-at one if one path between the same two signals was activated in the simulation. The two signals with any path left will be connected in a straight manner, and other paths between these two signals are regarded as unused Trojan paths. The method adapts only to the situation where only two paths exist between all signal pairs; otherwise, a dormant path will be ruled out since no pairs will be left as long as two or more activated normal paths exist. A path can only be regarded as an activated one after both 0-transition and 1-transition have been observed. Therefore, the detection method was soon defeated by other Trojan models because of these two limitations, such as in [40]. To completely eliminate false negatives, all path transition activations should be checked. On the other hand, false positives will be unavoidable for an inadequate test case.

2) FANCI: Waksman *et al.* proposed functional analysis for nearly-unused circuit identification (FANCI) [22], which uses a control value to measure the degree of control that an input has on the operation and outputs of a digital circuit. It is a static analysis approach to identify signals with low control value fan-ins as suspicious ones to be carefully checked. FANCI considers only combinational logic, so an unfolding solution named FIGHT-Metric for sequential elements was proposed in [41] that replaces the original edge in the directed graph

with a new one. Both methods consider only the structural information of the design, so malicious circuits can adopt more stages by inserting flip-flops to increase the control value of each stage. Furthermore, the complexity of FANCI and FIGHT-Metric will increase fast as the circuits undergoing validation grow. In addition, the selection of the threshold for the signal control value will affect the false positives and false negatives, and a manual code review is needed to confirm whether they are real Trojans. A simple Trojan model [23] shows the trade-off between the false positives and false negatives caused by the choice of threshold.

3) VeriTrust: To eliminate the dependency on the Trojan circuit structure, a new method named VeriTrust that tries to detect nonredundant inputs instead of unused logic was proposed in [23]. VeriTrust is another simulation-based detection method developed after UCI to detect Trojans. The tracer records the triggered Boolean products and sums, and a 3-level checker checks whether an input is redundant by comparing each activated product and sum. VeriTrust checks redundant inputs of combinational logic, so the Trojan model in [42] that combines Trojan logic with normal logic or with inserted registers will evade the detection. Rigorous segmentation is required to identify unused inputs, which is impractical in large designs.

4) FASTrust: A feature based analysis method named FASTrust was proposed in [43] for third party IP verification, which is independent of simulation results. Four features for the trigger circuit have been proposed: The first one is for time-triggered Trojans, which detects the large loop groups. The second is for data-triggered Trojans with large in-degree signals. The third one identifies sequential-triggered Trojans with a large total in-degree loop group. The last one is for implicitly triggered Trojans with small out-degrees. This feature-based method was developed further in [25], where quantifiable in-degree and stages were proposed, and a scoreboard method was adopted to reduce the false positive rate. A toggle rate combined method was proposed in [26] to reduce false positives caused by the four features in FASTrust. Feature-based detection must extract enough properties from known Trojans, which may cause its failure on new Trojans. In addition, false positives are an unavoidable problem since there is not much difference between Trojans and normal circuits in structure.

5) COTD: The Sandia Controllability/Observability Analysis Program (SCOAP) was introduced into Trojan detection in [24], where combinational controllability reflects the difficulty of setting a signal line to a required logic value from primary inputs and observability reflects the difficulty of propagating the logic value of the signal line to primary outputs [44]. An assumption is made in COTD that signals should have considerably low controllability and observability to remain dormant during circuit authentication. First, the controllability and observability values are obtained through the Synopsys TetraMAX tool, then the signals in the gate-level netlist are classified into three clusters by the k-means unsupervised clustering method using combinational controllability and observability values, and the inter-cluster distances are used for judging whether Trojan signals exist. The superiority of SCOAP makes it an outstanding static analysis method, and the use of the unsupervised clustering method eliminates manual intervention for code review.

6) A total state transition probability analysis method was proposed in [45] to compute the transition probability of signals in state registers, which was an accurate model for the function mode state probability calculation. However, the model aims at the foundry attack model, as the author claimed that prior knowledge of the finite state machine in the design was needed for size-reduced analysis; otherwise, the calculation will become infeasible for large scale circuits. The authors in [12], [46], [47] proposed a detection scheme using the signal probability signature of the reference design, which can also be applied only to the foundry attack model.

III. Motivation and Contribution

Unused logic analysis techniques, such as UCI and VeriTrust, are limited to small designs; otherwise, it is infeasible to extract the Trojan circuit exactly from the large design for analysis. The unused inputs or logic in Trojans may be part of normal circuits. Exploitation of original signals as a trigger circuit will cause a false negative in these two methods and structural feature analysis-based detection, such as FASTrust. For structural feature analysis, a false positive is an inevitable problem since Trojan circuits have nothing structurally different from normal ones. Different from the structural feature analysis mentioned above, the transition probability is the most direct reflection of the stealthy nature of Trojan triggers. To some extent, the control value and Sandia controllability/observability can also reflect the stealthiness of Trojans. All of these three logic feature-based measurements can be calculated in a topology-based way. Truth table analysis of the control value can only process combinational logic. The calculation complexity and inaccuracy, which depend on the circuit scale and number of reconvergent points, make traditional probability analysis of large designs impractical. The Sandia controllability/observability can be easily obtained in modern IC design flow. However, there is no accurate numerical relation between controllability and probability, and we will present a Trojan model proving that the COTD method is limited to some special Trojans with a high Sandia controllability or observability value, which is not necessarily true for Trojan circuits. We perform an in-depth review of the relation between probability and controllability. Based on these observations, we propose a static and dynamic combinational trigger identification scheme based on the proposed difference-amplified controllability and dynamic simulation-based transition probability. The contributions of this paper are as follows:

- Present a Trojan trigger model proving that a low transition probability is not necessarily related to a high controllability value and that simple controllability analysis is insufficient for such Trojan detection.
- Illustrate the relation between signal controllability and probability, and propose a novel static probability analysis scheme based on the proposed difference-amplified controllability.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2946044, IEEE Transactions on Information Forensics and Security

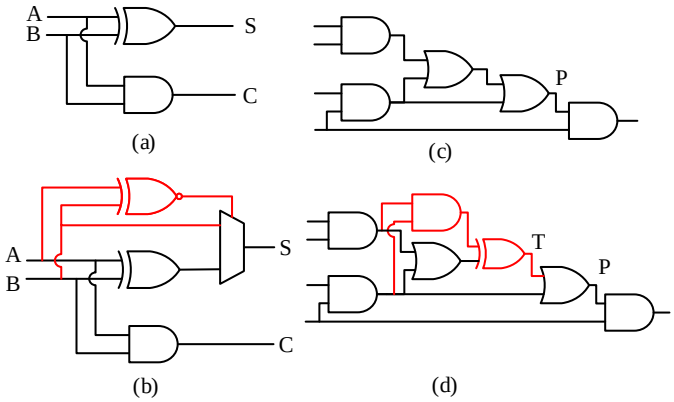

Fig. 3: Trojan examples in [18] and [48]

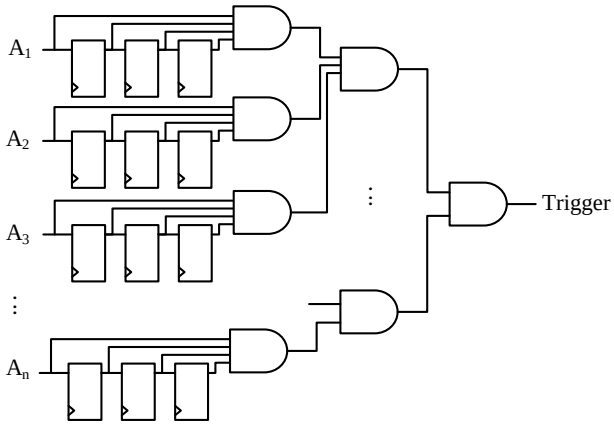

Fig. 4: A practical trigger circuit for the experiment.

- Demonstrate the limitation of the static transition probability, and propose a combined scheme by introducing dynamic transition analysis.
- Propose a reference-free trigger identification scheme for Trojan detection based on our static and dynamic transition probability analysis, which needs no golden circuits or prior knowledge of the original design.

IV. Inspection of the Trojan Model

Because of the huge space for hardware Trojan design, there is no specific description for hardware Trojans. It is generally recognized that hardware Trojans should be stealthy and malicious, i.e., Trojan signals should remain dormant during verification and test time to avoid being detected by a comparison of the logic response or side channel response. For the general Trojan to implement its malicious attacks, the victim or payload signal should be open for selection, as the target victim signals may vary with application. Therefore, we focus our efforts on the stealthiness of trigger conditions, and the trigger condition is defined as the occurrence of the Trojan circuit making an explicit difference in the original design. Fig. 3 presents two Trojan-inserted circuits (b) and (d), mentioned in [18] and [48], and their corresponding Trojan-free designs (a) and (c). For Trojan (b), the real trigger condition this paper is concerned with is $A = B = 1$, as $A = B = 0$ makes no difference from the original one. Following this principle, circuit (d) can be left without processing. Though the payload may affect some internal states or memory values causing high observability values, it must cause some difference that may be easily detected by functional verification.

Once the trigger is activated, the payload circuit passes the malicious behavior to the design. To remain dormant in presilicon simulation and postsilicon test, the Trojan circuit should have a rare trigger condition. A low trigger probability is the basic approach to implement a rare trigger condition, and reflected at the gate level, there is a low transition signal coupled with victim signals to act as the trigger. Since the stealthiness of Trojan circuits is guaranteed by the low switching probability of the trigger signal, the payload is open to selection for different malicious intentions. Therefore, the observability value can be very low since the payload may leak information through primary outputs. In addition, we found that a high controllability value is not necessary for a low transition probability. We present a trigger circuit model in Fig. 4, and the trigger signal has a very low probability and a relatively low controllability simultaneously.

The inputs $A_1 \sim A_n$ can be primary inputs or any internal signals in the original design, and the output signal $Trigger$ can be utilized as the trigger of Trojan circuits. If the inputs $A_1 \sim A_n$ are the primary inputs, then the combinational controllability and probability can be initialized as $(CC0, CC1) = (1, 1), (P_0, P_1) = (0.5, 0.5)$. If $n = 8$, the controllability of signal $Trigger$ is $(1, 4n) = (1, 32)$, and the 1-probability is $P_1 = \frac{1}{2^{4n}} = \frac{1}{2^{32}}$; the final controllability computed according to COTD is $CC = \sqrt{CC0^2 + CC1^2} = \sqrt{1^2 + 32^2} \approx 32$. The trigger circuit can be scaled for the required trigger probability. The circuit has several variations; first, the outputs of registers connected to the first level AND gate can be inverted or noninverted randomly. Second, the registers can be removed, and the inputs of the first level AND gates can be replaced by normal inputs. Third, all of the 4-input AND gates can be replaced by 2-input AND gates and an OR gate or an XOR gate can be inserted to evade feature analysis detection. The trigger circuit can be easily controlled from primary inputs by malicious users but with a very low transition probability and a reasonable controllability value. We replace the trigger circuits in Trust-HUB benchmarks with the above trigger circuit, and all inputs $A_1 \sim A_8$ are connected to primary inputs for simplicity and assurance that the Trojan is effective and controllable. The COTD clustering flow is applied to these benchmarks, and Fig. 5 shows the k-means clustering on 4 example benchmarks.

Table I shows the clustering results, where the trigger signal was classified into cluster 2, which was regarded as the Trojan cluster in the first three benchmarks, but was classified into cluster 1 in the last benchmark, which was a normal cluster. In addition, the intercluster distance was close to the Trojan-free benchmarks, which may cause the whole benchmark to be regarded as Trojan-free netlists. The false positive rates of the first three benchmarks are 15.6%, 30.3% and 26.1%. The false positives and false negatives will keep increasing with the scale of the design. Since the trigger probability is $\frac{1}{2^{32}}$, it

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2946044, IEEE Transactions on Information Forensics and Security

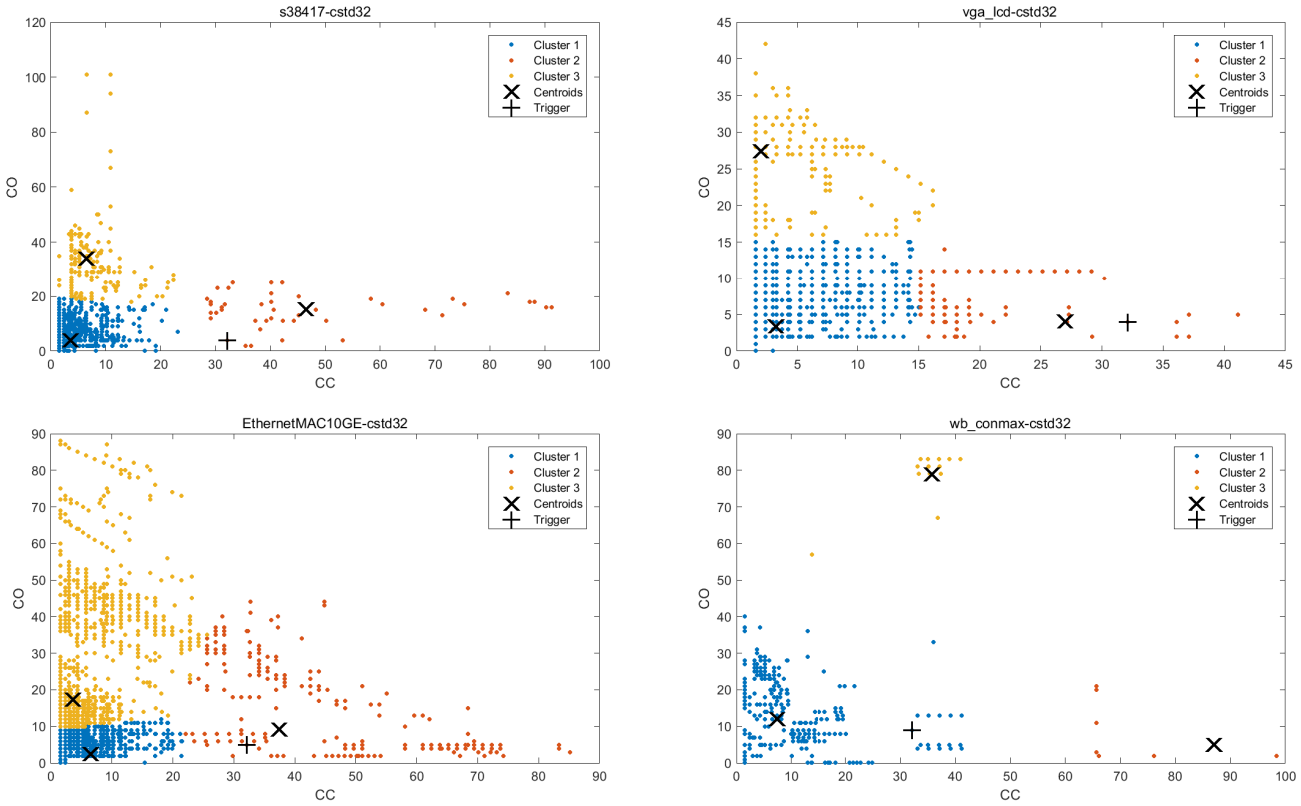

Fig. 5: COTD clustering for the proposed trigger model

TABLE I: COTD Clustering Results on the Proposed Trigger

		s38417	vga	Ethernet	wb
C1	Centroid NoC	(3.5, 3.8) 4909	(3.3, 3.4) 48892	(6.4, 2.5) 76249	(7.4, 11.9) 22043
C2	Centroid NoC	(46.5, 15.2) 161	(27.0, 4.1) 10445	(37.4, 9.1) 461	(87.1, 5.0) 25
C3	Centroid NoC	(6.3, 33.5) 763	(2.1, 27.4) 10844	(3.6, 17.3) 26412	(35.6, 78.9) 161
Trigger	Value Cluster	(32.1, 4.0) C2	(32.1, 4.0) C2	(32.1, 5.0) C2	(32.1, 9.0) C1
dis_c1c2		44.5	23.7	31.7	80.0
dis_c1c3		29.9	24.0	15.1	72.7
dis_c2c3		44.2	34.1	34.8	90.0

is almost infeasible to activate the Trojan, and the inserted registers make the ATPG programs fail to generate proper patterns in the functional mode. To avoid this embarrassment, we propose a detection method directly aimed at the low transition probability nature of the trigger signal.

V. Proposed Detection Method

Since feature analysis and static testability analysis, such as COTD, can be avoided by deliberately designed Trojans, we propose to detect trigger signals of Trojan circuits at the gate level based on its stealthy nature, the low transition probability. We first illustrate a static probability analysis method.

A. Static Probability Analysis

The basic idea to compute the signal probability is to perform the calculation from primary inputs toward primary outputs in a breadth-first manner. The output probability of each gate is calculated according to certain rules for that gate after the probability measures of all its inputs have been calculated. The probability calculation rules for combinational gates are summarized in Table II, where P_0 and P_1 stand for the probability of the signal being 0 and 1, respectively.

TABLE II: Probability Calculation Rules

	P_0	P_1
AND	$1-P_1$	$\prod(input\ P_1)$
OR	$\prod(input\ P_0)$	$1-P_0$
NOT	$input\ P_1$	$input\ P_0$
XOR2	$1-P_1$	$P_0(a)*P_1(b)+P_1(a)*P_0(b)$
MUX21	$1-P_1$	$P_0(sel)*P_1(a)+P_1(sel)*P_1(b)$

To start the computation, it is reasonable to initialize the (P_0, P_1) of all primary inputs to $(0.5, 0.5)$, although the functional mode probability may differ from this. The reason behind this approach is that Trojan logic is not limited to the function mode, and all inputs can be manipulated freely. With the calculation rules and initial values of primary inputs, the probability of all combinational circuits can be calculated. For sequential components, such as a positive edge flip-flop with synchronous high-active set and reset, the rules can be represented as

$$\begin{aligned} P_0(Q) &= [P_1(r) + P_0(r)\cdot P_0(s)\cdot P_0(D)]f_r \\ P_1(Q) &= [P_1(s) + P_0(s)\cdot P_0(r)\cdot P_1(D)]f_r \end{aligned} \tag{1}$$

where f_r is the frequency ratio to the reference clock, and the reference clock can be either a real system clock or a

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2946044, IEEE Transactions on Information Forensics and Security

virtual clock. Similar rules for flip-flops with asynchronous high-active set and reset can be deduced as

$$\begin{aligned} P_0(Q) &= P_1(r) + P_0(r) \cdot P_0(s) \cdot P_0(D) \cdot f_r \\ P_1(Q) &= P_1(s) + P_0(r) \cdot P_0(s) \cdot P_0(D) \cdot f_r \end{aligned} \tag{2}$$

These rules can help construct the probability of sequential elements, and an approximate probability measure of sequential circuits can be obtained by solving equations of loop structures. To calculate the probability of state registers, a total-state transition model was proposed in [45], where all states of an FSM and all inputs influence the states constructed in the total-state transition matrix; then, the probability of states were deduced from the state transition equations. The requirement of the knowledge of a Trojan-free design to simplify the state matrix cannot be satisfied in the 3PIP attack model, and the complexity $O(N_{state} * N_{branch} * 2^{N_{rec}})$ can grow rapidly as the circuit scales, where N_{state} and N_{branch} represent the numbers of total states and branches, respectively, their maximum values are $2^{b_{state}}$ and $2^{b_{branch}}$, respectively, b_{state} and b_{branch} are the numbers of register bits, and N_{rec} stands for the number of reconvergent wires. We hence propose a novel method for static probability analysis.

B. Controllability Instead Analysis

As we know, SCOAP has been widely used in test generation as another topology-based testability analysis technique, and it has been supported by the Synopsys ATPG tool TetraMAX. The calculation rules for combinational controllability are presented in Table III. In the original SCOAP, there should be an additional increment (+1) in each equation listed in Table III as in [44], which is not considered in the actual values calculated by TetraMAX. The increment is not the major part, and for simplicity, we adopted these calculation rules throughout this paper. Suppose the initial signal probability of all primary inputs is $P_0 = P_1 = 0.5$ and the controllability is initialized as $CC0 = CC1 = 1$. The output probability and controllability of the three example circuits is shown in Fig. 6.

TABLE III: Combinational Controllability Calculation Rules

	$CC0$	$CC1$
AND	$min\{input\ CC0\}$	$\sum(input\ CC1)$
OR	$\sum(input\ CC0)$	$min\{input\ CC1\}$
NOT	$input\ CC1$	$input\ CC0$
XOR2	$min\{CC0(a) + CC0(b), CC1(a) + CC1(b)\}$	$min\{CC0(a) + CC1(b), CC1(a) + CC0(b)\}$

The pure AND gate or pure OR gate circuit obtains the lowest probability with the same number of gates, so we employed the pure AND gate in the trigger model proposed above, which can be replaced by a mixed one. When the circuit changes from the pure one to a mixed one, both the difference between 0-controllability and 1-controllability and the difference between 0-probability and 1-probability become smaller. The simplest case of Fig. 6(a) shows the simplest case of the pure 2-input AND gate circuit, for which the probability and controllability can be represented as $(CC0, CC1) = (1, 2^n), (P_0, P_1) = (1 - (\frac{1}{2})^{2^n}, (\frac{1}{2})^{2^n})$. The other two circuits in Fig. 6(b) and Fig. 6(c) take intermediate values. The three circuits show that a strong correlation exists

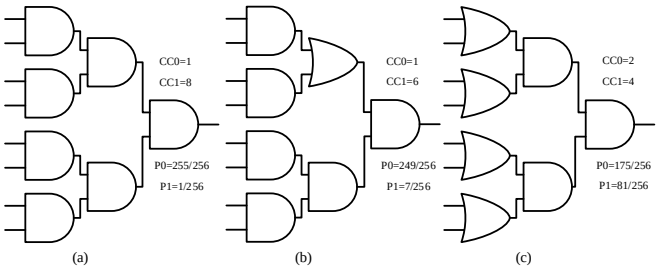

Fig. 6: Circuit example of controllability and probability

between the probability and controllability. We further observe that the rules for probability calculation and controllability have great similarity. For discussion purposes, we present the following definition.

Definition 1. For AND logic output, if the output value of a gate is the AND condition of its inputs, we call it an AND logic output; for example, output 1 of an AND gate and output 0 of an OR gate are both AND logic outputs since they are asserted when all of the inputs are set to 1 or 0, respectively, while output 0 of the AND gate is a non-AND logic output.

Both controllability and probability are topology-based analysis, and the calculation rules for them are similar. They both accumulate on the AND logic output and propagate on the opposite output value, although in different directions and ways. The probability decreases by multiplication of the input probability on the AND logic output and increases on the non-AND logic output, while the controllability increases by addition of the input controllability on the AND logic output and decreases on the non-AND logic output. The different rules on the non-AND logic output mean that no deterministic relationship between the probability and controllability exists, which has been proven by the example circuit in Fig. 6(a) and Fig. 6(b) since the signals with the same controllability have different probabilities. The calculation rules and example circuits show that both the controllability and probability will enlarge the difference between the AND logic output and non-AND logic output, especially at AND gates and OR gates. Moreover, these differences between value 0 and 1 of the probability and controllability vary synchronously. Therefore, we propose the following hypothesis.

Hypothesis 1. There exists a strong correlation between the signal probability and combinational controllability, that is, imbalanced signal 0/1-probability is always accompanied by imbalanced 0/1-controllability; i.e., if we would like to find imbalanced 0/1-probability, we need only to search for imbalanced 0/1-controllability signals.

Theoretically, the combinational controllability reflects the effort required to set a signal line to a certain value from primary inputs; imbalanced 0-controllability and 1-controllability mean that it is more difficult to control one value than another, which will cause one value to occur with lower probability than the other, i.e., an imbalanced probability. Therefore, it is reasonable to make this hypothesis. To further confirm this hypothesis, we conducted more experiments on ISCAS benchmarks. Fig. 7 shows the correlation between controllability and probability. The controllability was calculated by the Synopsys

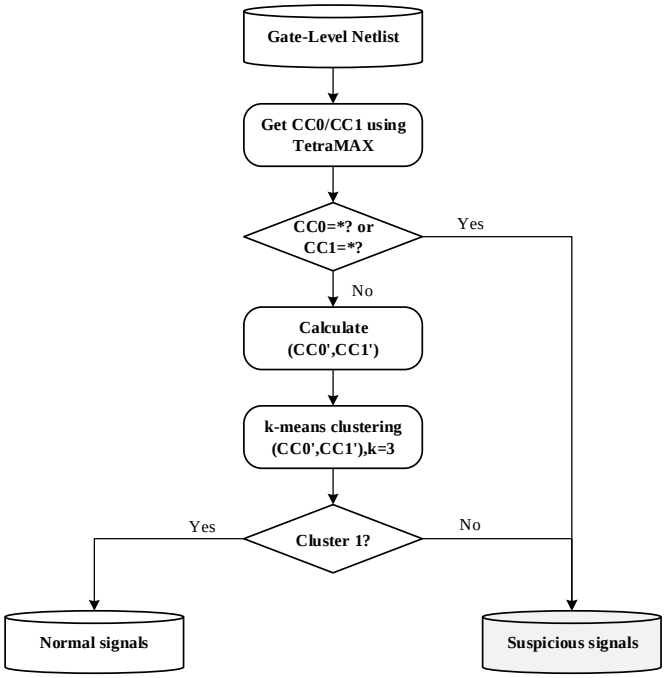

Fig. 8: Static probability analysis flow

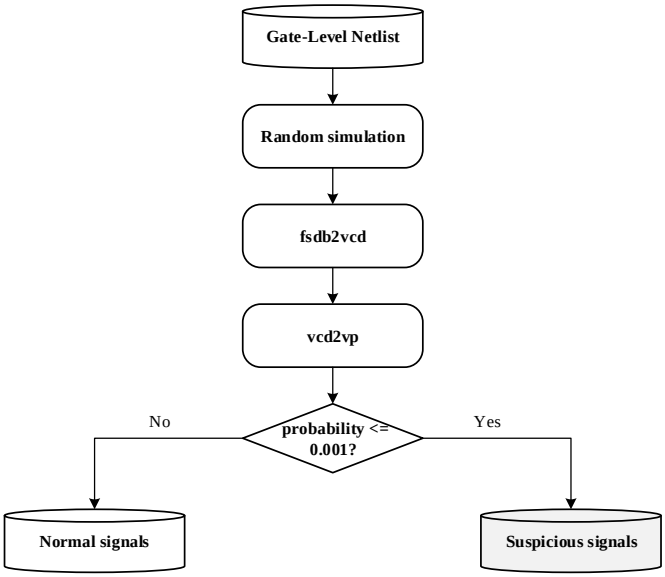

Fig. 9: Dynamic probability analysis flow

EDA tool TetraMAX, and the probabilities were computed according to the static rules mentioned above. Since we only care about the lower probability, the probability illustrated is the lower one of P_0 and P_1. The figure shows that the signal probability exhibits an obvious decreasing tendency when it is away from line $P_0 = P_1$ toward $P_0 = 0$ or $P_1 = 0$.

Based on this hypothesis, imbalanced 0-controllability and 1-controllability indicate imbalanced 0-probability and 1-probability, which is a low transition probability. We can take difference analysis between 0-controllability and 1-controllability in Trojan detection instead of probability analysis. Considering the inaccuracy caused by reconvergent signals, we choose to handle the signals by the k-means clustering method, which is not very sensitive to the absolute values or circuit scale.

The proposed static analysis flow is shown in Fig. 8. We analyze the imbalance of 0-controllability and 1-controllability instead of probability since controllability values can be obtained easily during ATPG design flow by TetraMAX. To emphasize the imbalance between 0-controllability and 1-controllability, we propose a difference-amplified controllability transformation, marked by a single quote. The equation for computing the difference-amplified controllability is

$$(CC0', CC1') = (\frac{CC0^2}{\sqrt{CC0 \cdot CC1}}, \frac{CC1^2}{\sqrt{CC0 \cdot CC1}}) = (CC0\sqrt{\frac{CC0}{CC1}}, CC1\sqrt{\frac{CC1}{CC0}}) \quad (3)$$

After transformation, all the observations $(CC0', CC1')$ fall into the hyperbola cluster of inverse proportional functions. This model can compress the chaotic points into the hyperbolic limited range and highlight the imbalance metric. The model can still partly keep the common mode component since CC0 or CC1 is multiplied, which is useful since the controllability value itself stands for the difficulty of controlling signal values. To make the difference-amplified values more compact, we limit all transformed values that exceed 254 to 254. In addition, all points whose original 0-controllability or 1-controllability exceeds 254, represented by "*", are directly regarded as suspicious signals since it is meaningless to calculate on an uncertain value, and it is reasonable to take signals with such low testability as suspicious ones even in design for testability analysis.

After the difference-amplified controllability values have been obtained, k-means clustering was applied to the difference-amplified controllability pairs. Since we know that all observations fall on the hyperbola cluster and there should be two kinds of imbalanced cases and a normal one, we classify these signals into three clusters: an origin cluster including the normal signals close to the coordinate origin and two asymptotic clusters that include high CC0, low CC1 and low CC0, high CC1 signals. Based on this, we set k=3 clusters. Additionally, we choose the point closest to the origin, the point farthest from the X-axis and the point farthest from the Y-axis as the initial centroids of the three clusters. The clustering procedure was completed in the MATLAB environment. The two clusters away from the coordinate origin will be classified into the suspicious list if their final centroids exhibits an obvious imbalance in controllability.

C. Dynamic Probability Analysis

The static probability-based controllability clustering will generate three clusters, and we classify the clusters with maximum CC0 or maximum CC1 as suspicious clusters. Since continuity exists in controllability and probability values, it is inevitable that some normal signals will be classified into suspicious clusters. To reduce the false positive rate as much as possible, we propose a simulation-based probability analysis to further identify whether a signal in suspicious clusters is more likely to be a Trojan or normal one. Fig. 9 shows the dynamic probability analysis flow.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2946044, IEEE Transactions on Information Forensics and Security

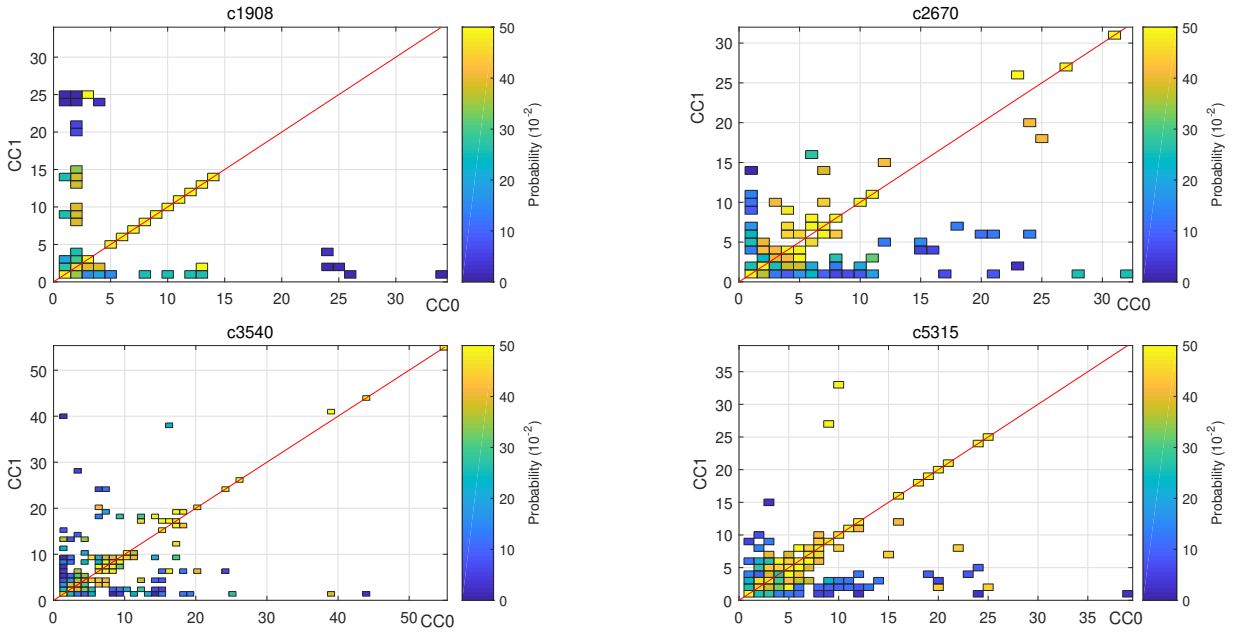

Fig. 7: Controllability and probability on ISCAS benchmarks

The simulation in the first step can be a random simulation or a mixed one with a functional simulation, which may promote the accuracy of the functional mode probability, but the functional test pattern is not necessary since the Trojan is not limited to the normal work mode. For the Trust-HUB benchmarks, we employ only the random simulation, and the simulation time is set to 320000 cycles considering both the simulation time and accuracy. If the low transition rate signal is activated in the simulation, the corresponding test patterns should be kept and applied in the functional test and validation flow. We generate a fast signal database (FSDB) file instead of a value change dump (VCD) file directly, as this will reduce the simulation time and decrease the wave file size greatly. Then, we use the Synopsys tool fsdb2vcd to convert the generated FSDB file to VCD for probability analysis since FSDB is in an unreadable format. The converted VCD file is still much smaller than the directly generated one in the simulation. Then, we write a Perl script to extract the value-probability of all signals in the VCD file, which is the vcd2vp step. After all of the signal probabilities have been obtained, we can simply set a threshold, such as 0.001, or choose the lowest 1% transition probability wires as the suspicious signal list. The ideal threshold should be smaller than all normal signals and greater than all Trojan signals. Experiments show that 0.001 will obtain a relatively low false positive rate and cause no false negatives in most cases since Trojan signals always have a transition probability less than 0.001 to remain dormant in the functional validation. When the circuit is relatively small, the threshold can correspondingly increase or the lowest 1% of signals can be taken as the suspicious list in the dynamic analysis. It is a trade-off to balance the burden of processing false Trojan signals and the risk of taking Trojan signals as normal ones.

D. Static and Dynamic Combined Detection

The static analysis provides a simple and fast evaluation of the signal probability, and the SCOAP testability and TetraMAX tools have been widely integrated into modern IC design flow. In addition, the static analysis can better reflect the structural information of the design, and the cluster method can separate the outliers from normal wires with no threshold, which can be better applied to circuits of different scales. The exploitation of the unsupervised clustering method makes the analysis less dependent on an accurate controllability value and independent of the circuit scale. If the clustering result shows a clear distinction among the three clusters, the static analysis can obtain an acceptable and reliable detection result. Otherwise, dynamic analysis should be applied to reduce the false positive rate. Since the sequential logic loops and reconvergent points will increase the computational burden and cause inaccuracy in a topology-based algorithm, including both controllability-based or direct probability-based analysis and dynamic simulation can fix these dependency and reconvergence problems. However, the dynamic simulation quality is affected by test vectors, and no structural information is reserved. The test generation methods proposed in [18], [19], [49]–[51] in test mode can be employed to reduce false positives, but they are not necessary since the dynamic probability has only to reflect the open mode probability. After the static and dynamic analyses have separately obtained the suspicious signal lists, we take the intersection of these two lists as the final malicious triggers.

VI. Experimental Results

The proposed detection method was applied to the original Trojan version and the Trojan replaced by the trigger circuit proposed above on Trust-HUB and the benchmarks proposed in [52]. The static and dynamic analysis flows

can be conducted simultaneously; for static analysis, Sandia combinational controllability CC0 and CC1 are obtained by Synopsys TetraMAX, and values that exceed 254, represented by "*", are categorized into the suspicious list directly since they deserve to be inspected for such high controllability values even in DFT design. Then, the difference-amplified controllability pairs are computed according to (3), and three clusters are obtained by MATLAB k-means clustering. Then, we mark the cluster close to the origin as Cluster 1 and the other two clusters as Cluster 2 and Cluster 3. Cluster 1 is regarded as a normal cluster, and the centroids of the other two clusters are used as judging criteria for whether the Trojan exists. Moreover, the signals in Cluster 2 and 3 are extracted as the static suspicious list if their centroids have imbalanced controllability. For the dynamic simulation, we utilize a random test, since we have only to obtain the fair signal probability, so no complex Trojan excitation algorithm is needed.

We first dump the FSDB format wave file and then translate it to the VCD file using the fsdb2vcd tool; then, we extract the signal transition information in the VCD file. After we acquire the dynamic signal transition information, we set the probability threshold to 0.001 and take all signals whose minimum probability is lower than the threshold as dynamic suspicious signals. Finally, we regard signals falling into both the static and dynamic suspicious lists as probable triggers. Fig. 10 shows the static k-means clustering of some typical benchmarks. All signals were classified into three clusters; Cluster 1 includes the normal signals, and the other two clusters contain two kinds of imbalanced signals. There is no trigger signal in the fourth subgraph, as the trigger signal has been directly classified into the suspicious list because of the large controllability that exceeds 254. The first two subfigures show the trigger circuit proposed above based on the s38417-T100 and vga_lcd-T100 benchmarks on Trust-HUB. In addition, the following two are the basic benchmarks on Trust-HUB. The last four come from the TRIT-TC benchmarks. The first part of the benchmark name represents the basic design, such as s38417 and wb_conmax, and the second part indicates the Trojan type; e.g., cstd32 represents the trigger circuit in Fig. 4. These eight example benchmarks can represent the different types of Trojans on Trust-HUB and the different situations that our proposed clustering method will encounter.

Table IV depicts the centroid and number of points in each cluster of the k-means clustering results. As Fig. 10 shows, all trigger signals are classified into Cluster 2 or 3 except for the EthernetMAC testbench since its trigger signal has been excluded from clustering because it has been regarded as a static suspicious signal in preprocessing. The results show that all trigger signals fell into Cluster 3, and the difference between 0-controllability and 1-controllability of the centroid of Cluster 3 was much greater than that of Cluster 1. In addition, the number of signals in Clusters 2 and 3 is much less than that in Cluster 1, which means that the false positive rate is acceptable in static clustering.

Fig. 11 shows the dynamic probability distribution of the eight benchmarks mentioned above. There are three probability peaks near $P = 0, 0.5, 1$, and the numbers of signals with probability close to $P = 0$ and $P = 1$ are considerable in the first four benchmarks. The low probability signals increase as the circuit size and complexity increase, such as wb_conmax, vga_lcd and EthernetMAC10GE, and these low transition signals are mostly those not activated with $P = 0$ or $P = 1$, which is partly caused by the inefficient random test patterns. As the circuit scales, this inefficiency may become worse, and the functional test pattern and ATPG test pattern can be adopted to reduce the inefficiency caused by the limitation of the random simulation. On the other hand, the false positives can be overcome to some extent by our static analysis, so only the random simulation was adopted in our experiments. Table V shows the accurate numbers of signals with a probability of less than 0.01 and 0.001 and the total number of signals in the dynamic probability analysis.

TABLE IV: Clustering Results on Example Benchmarks

	Cluster 1		Cluster 2		Cluster 3		Trigger	
	Centroid	NoC	Centroid	NoC	Centroid	NoC	Value	Cluster
s38417	(3.8, 3.7)	5696	(195.1, 0.6)	87	(0.9, 214.9)	50	(0.2, 173.4)	3
vga	(2.6, 2.7)	59895	(134.4, 0.2)	5522	(0.2, 134.5)	4764	(0.2, 173.4)	3
wb	(11.3, 3.2)	21869	(118.0, 1.1)	256	(0.8, 225.0)	72	(1.1, 254.0)	3
Ethernet	(7.3, 7.9)	103021	(234.0, 0.3)	47	(0.6, 224.8)	21	-	-
c5315	(3.5, 5.1)	2480	(150.0, 0.5)	1	(0.4, 167.7)	10	(0.2, 198.3)	3
c6288	(8.9, 15.4)	1988	(188.6, 0.5)	2	(14.9, 84.8)	447	(0.3, 254.0)	3
s1423	(3.3, 2.3)	563	(101.8, 0.4)	5	(0.2, 254.0)	3	(0.2, 254.0)	3
s35932	(3.2, 1.9)	8348	(45.3, 0.3)	6	(0.7, 197.2)	3	(0.2, 233.1)	3

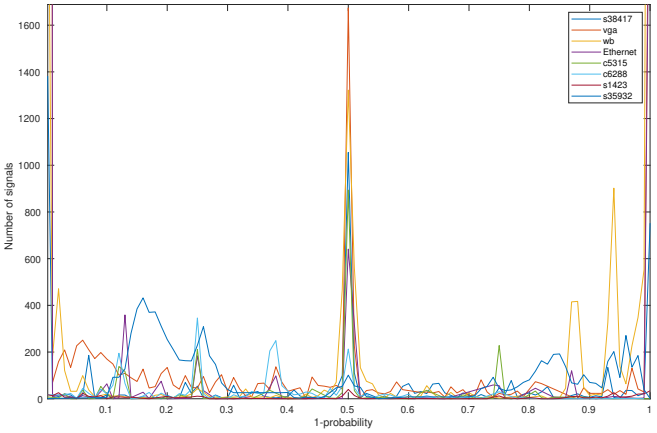

Fig. 11: Dynamic probability distribution

TABLE V: Results of Dynamic Probability Analysis

	0.001(per)	0.01(per)	Total
s38417	2342 (40.2%)	2489 (42.7%)	5833
vga	38771(55.2%)	38901(55.4%)	70181
wb	8884 (40.0%)	10191(45.9%)	22197
Ethernet	54863(53.2%)	54946(53.3%)	103092
c5315	2 (0.08%)	11 (0.44%)	2491
c6288	18 (0.74%)	18 (0.74%)	2437
s1423	32 (5.60%)	64 (11.2%)	571
s35932	4 (0.05%)	16 (0.19%)	8357

Table VI shows the final results of the static and dynamic combined probability analysis. Column 2 lists the total number of final suspicious signals, and column 3 shows the false positive rate of each testbench. The false positive (FP) and

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2946044, IEEE Transactions on Information Forensics and Security

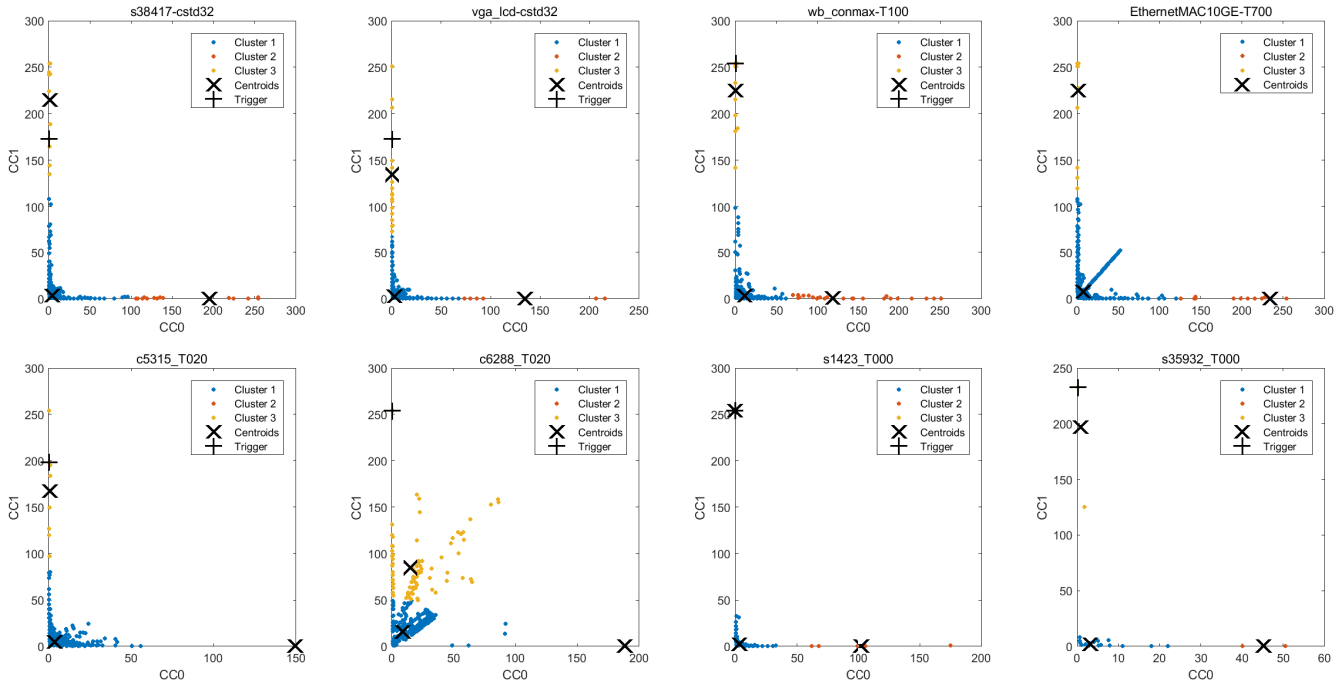

Fig. 10: Clustering on example benchmarks

false negative (FN) rate are defined as

$$FPR = \frac{FP}{N} = \frac{FP}{FP + TN}$$
$$FNR = \frac{FN}{P} = \frac{FN}{FN + TP} \quad (4)$$

where FP includes the signals labeled as Trojan signals that are actually normal signals, and FN refers to those labeled as normal signals that are actually Trojan signals. TP refers to the signals labeled as Trojan signals that are true Trojan signals, and TN refers to the signals labeled as normal signals that are true normal signals.

All of the signals except for the trigger signal in the suspicious list are considered as false positive signals, including signals in the trigger and payload circuits, when computing the false positive rate. After the final probable Trojan signals were obtained, we extracted their fan-in and fan-out circuits from the design for a further check. Since we have extracted the trigger signals, the whole Trojan circuit and payload signals can be easily reconstructed. According to our application, we can further reduce the probable Trojan signals or treat them as Trojans directly. All feature analysis, functional test and run-time monitoring techniques can be adopted for Trojan confirmation. Moreover, suspicious signals can be removed or used by run-time monitoring schemes as input signals.

TABLE VI: Final Results of Probability Analysis

Benchmark	Suspicious	False Positive
s38417	87	1.47%
vga	2591	3.69%
wb	160	0.72%
Ethernet	63	0.06%
c5315	2	0.04%
c6288	18	0.70%
s1423	2	0.18%
s35932	3	0.02%

We applied the proposed combined probability analysis to all of the 914 benchmarks of TRIT-TC and TRIT-TS generated by [52] on Trust-HUB. Fig. 12 illustrates the false positive and false negative rates of our proposed detection and COTD detection on these benchmarks, where the x-axis shows the serial number of benchmarks sorted alphabetically, with TRIT-TC first, the left y-axis shows the false positive and negative rates in COTD and our proposed detection, and the right y-axis draws the mean silhouette in k-means clustering. The closer to 1 the mean silhouette is, the better the clustering result. In our detection schemes, there were only 16 benchmarks whose false positive rates were greater than 4%, and the largest false positive rate was 11.7%. In addition, there were 159 benchmarks whose false positive rates were less than 0.01%. In the dynamic simulation procedure, there were 29 benchmarks whose triggers were activated among all of the 914 benchmarks, whose trigger signals and bench names are listed in Table VII. Since all the trigger signals can be detected or activated in the dynamic simulation, the false negative rates are all 0, and there is only false positives for our proposed method in Fig. 12. The false positives and false negatives of COTD detection on the same benchmarks are shown in the same figure. There were 327 benchmarks whose false negative rates were not zero, which means there were undetected Trojan signals. The false positive rates were also much greater than our scheme, especially in dealing with combinational Trojans and the full scan or partial scan sequential Trojans.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2946044, IEEE Transactions on Information Forensics and Security

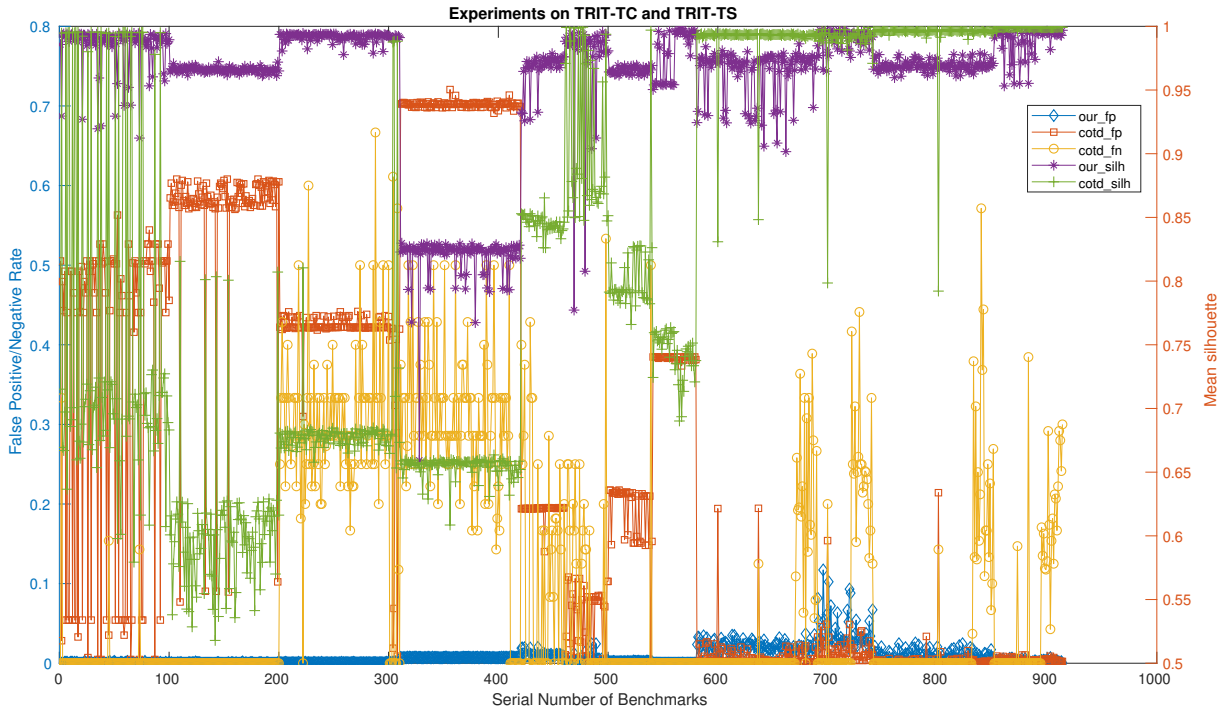

Fig. 12: FPR and FNR on TRIT-TC and TRIT-TS benchmarks

TABLE VII: Benchmarks triggered and trigger signals

Benchmark	Triggered trigger signal
c2670_T052	Trigger_en52_0
c2670_T081	Trigger_en81_0
c3540_T003	Trigger_en3_0
c3540_T005	Trigger_en5_0
c3540_T072	Trigger_en72_0
c3540_T080	Trigger_en80_0
c3540_T090	Trigger_en90_0
c3540_T096	Trigger_en96_0
c5315_T048	Trigger_en48_0
s13207_T2019	Trigger_en19_0
s1423_T2000	Trigger_en0_0
s1423_T2002	Trigger_en2_0
s1423_T2015	Trigger_en15_0
s35932_T008	Trigger_en8_0
s35932_T013	Trigger_en13_0
s35932_T2006	Trigger_en6_0
s35932_T2019	Trigger_en19_0
s13207_T481	Trojan_out0
s35932_T403	Trojan_out0
s35932_T405	Trojan_out0
s35932_T409	Trojan_out0
s35932_T422	Trojan_out0
s35932_T423	Trojan_out0
s35932_T433	Trojan_out0
s35932_T442	Trojan_out0
s35932_T610	Trojan_out0
s35932_T611	Trojan_out0
s35932_T613	Trigger_en1_13
s35932_T619	Trojan_out0

VII. Discussion

The results show that all of the trigger signals in the benchmarks generated by [52], s38417-T100, vga_lcd-T100, wb_conmax-T100, and EthernetMAC10GE-T700 on Trust-HUB, and the Trojans proposed above have been detected by our static and dynamic probability analysis scheme. To our knowledge, we are some of the few researchers to present trigger detection schemes for Trojan detection in addition to [45].

A. Trigger Detection or Trojan Detection

Different from the whole Trojan circuit detection schemes, our method aims only at possible trigger signals. We believe that stealthiness is the basic feature of Trojan circuits, especially for triggered Trojans, and the stealthiness is mainly guaranteed by trigger signals. Consequently, there will always be a signal that acts as a trigger signal that has a lower probability or transition activity at the gate level that couples with payload signals, and our probability method just detects these trigger signals. Considering that an attacker can exploit the low transition signals in the original design, our probability analysis and trigger detection method will have better detection ability than side-channel and feature-based techniques or other Trojan circuit detection approaches because the maliciously added circuits can be very small compared to the original design. The already known Trojan features can be avoided by deliberate design. However, the low transition and probability feature must be kept to evade the functional test and verification. Reflected in the gate level, it is always an explicit or implicit trigger with a low transition probability signal connected to a certain input port of a gate. Some Trojan designs may introduce a trigger circuit with more than one

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2946044, IEEE Transactions on Information Forensics and Security

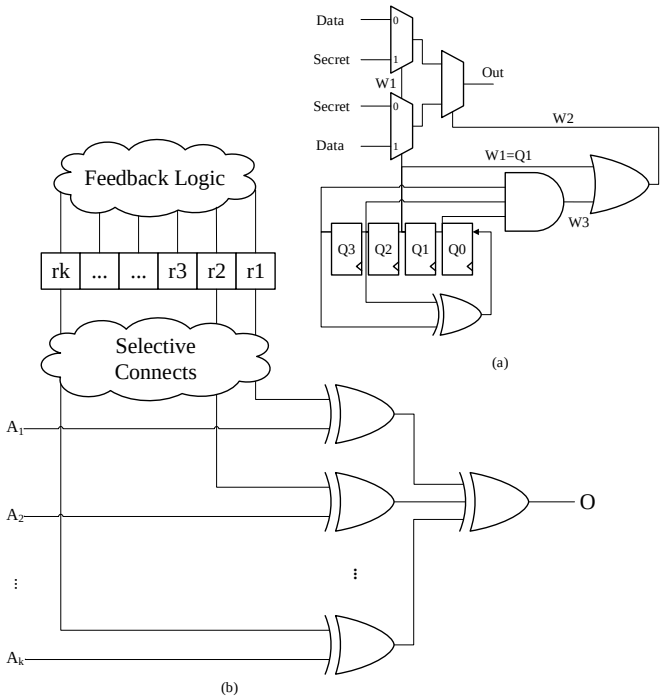

Fig. 13: The XOR-LFSR Trojan

signal to couple with the victim signal, such as the newly proposed XOR-LFSR Trojan in [48] shown in Figure 13. The author claims $(W1, W2)$ as trigger signals, but in fact, signal $W3$ is the source of stealthiness in Fig.13(a). There is no essential difference between the XOR-LFSR model and other Trojans, as $W3$ and O act as trigger signals. Therefore, XOR-LFSR Trojans can be detected by our methods, and if the LFSR circuit is not connected to other inputs, they will be uncontrollable and thus detected by static analysis, which has been proven in [24].

B. Always-on Trojans

Since we aim at the trigger signal in Trojan circuits, an always-on Trojan is an unavoidable topic to be discussed here. The always-on Trojan is another Trojan type, for which no trigger signal exists or triggers have a higher transition probability. An example of an always-on Trojan is the inverter-based ring-oscillator that can accelerate aging of ICs. As mentioned in [24], the ring-oscillator circuits will have a high controllability value that is easily detected by static analysis. Always-on Trojans may also be used for side-channel leakage of secret information. Combined with a side-channel-based detection method, we can easily detect those always-on Trojans in the dynamic simulation step since they always have considerable power or electromagnetic influence. In addition, we can mask these side-channel information leakages to mute the effects of Trojan circuits by introducing true random noise into the critical chip device. That is, we can handle these Trojans through other existing effective approaches. Therefore, we can adopt some existing sophisticated techniques for these always-on Trojans that may evade our trigger detection schemes.

C. Static and Dynamic Mixed Analysis

Since we exploit the relationship between Sandia combinational controllability and probability, our static analysis becomes simpler and is easily integrated into current IC design flow, which still maintains enough accuracy for trigger analysis. The dependency and reconvergence of signals are sources of inaccuracy in all static probability analysis methods, including [45]. Another problem of static transition analysis is the inaccurate representation of the transition probability,

$$P_\tau = P_0 \cdot P_1 \tag{5}$$

According to this equation, all bits of a sequential counter are the same since the probability of every bit in the counter is $P_0 = P_1 = 0.5$, but the actual transition probabilities are different from each other. The highest bit obviously has a lower transition frequency than the lowest bit of the counter. The dynamic probability can reflect this transition difference, and the dependency and reconvergence can be solved naturally in dynamic simulation. However, the quality of the dynamic simulation results seriously relies on test vectors, which may cause a huge false positive rate. This has been proven by the dynamic probability histogram and probability distribution table in Fig. 11 and Table V. As the primary inputs and sequential elements increase, the increasing state space causes a decrease of the simulation quality. Therefore, the dynamic simulation transition is mainly used as a static probability correction for reconvergent points and inner dependency, which can eliminate the limitation of the test pattern efficiency and finite simulation time. In addition, the static analysis can reflect the structural and topological information of circuits and can be obtained in linear time, so we take advantage of both static and dynamic analysis.

A state transition probability calculation was proposed in [45], but the requirement of prior knowledge of state machines and the computing complexity caused by reconvergent signals make it unsuitable for 3PIP Trojan detection. The SCOAP is easily obtained since it has been integrated into the modern IC design tool, and TetraMAX already has good processing ability for reconvergent signals. Fig. 14 shows the reconvergent signals detected by the controllability measure in c6288 benchmarks, as signals $N1684, N1399, N6245$ have $CC0 = 0, CC1 = *$. However, the state probability calculation in [45] only processes one-level reconvergent signals, and at most 10 reconvergent signals can be processed. We have shown that the simple controllability cannot reflect the signal probability, so we propose the difference-amplified controllability to replace the direct signal probability computation as our static analysis. In our proposed model, we consider the difference between controllability, which can partly eliminate the influence of common mode inaccuracy caused by controllability computation. The unsupervised clustering method k-means is adopted to handle the controllability values, which is not very sensitive to the accurate value of the controllability and the circuit scale. It can separate the outliers from normal signals under any circuit size without requiring a threshold.

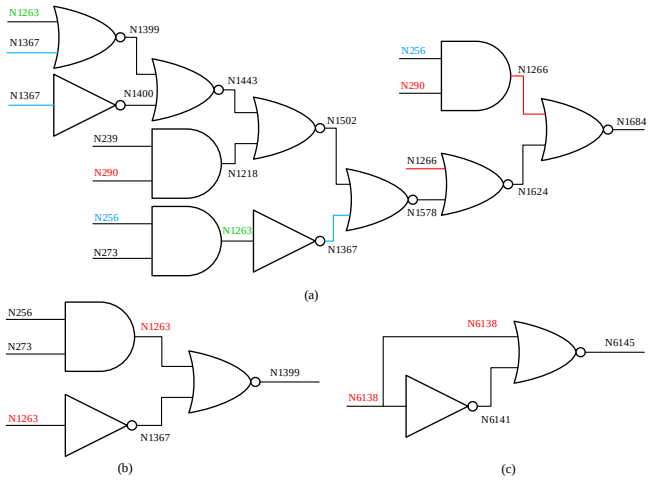

Fig. 14: Reconvergent signals detected in c6288_T000

VIII. Conclusion

In this paper, we aimed at low switching probability trigger signals at the gate level directly and proposed a static and dynamic combined probability analysis approach for hardware Trojan detection. The static probability analysis flow is based on the observation of the correlation between the controllability and probability that we have found and exploits the high accuracy and efficiency of the sophisticated Sandia Controllability/Observability Analysis Program, which can be easily integrated into the current IC design flow. We proposed the difference-amplified controllability and utilized k-means clustering to categorize the signals to further improve the efficiency. The dynamic probability analysis takes advantage of the accessibility of the standard VCD format and the high efficiency of the Synopsys FSDB format. The dynamic analysis flow is also easily integrated into the modern IC design flow. As discussed above, the static and dynamic combined method can take advantage of both techniques and provide a much lower false positive rate. The experiments on benchmarks of Trust-HUB show that all triggers can be identified either by dynamic simulation or by the combined probability analysis flow.

The proposed trigger identification method is aimed at the 3PIP Trojan attack model and can also be applied to the interior attack. The detection mainly focuses on triggered Trojans, and a portion of always-on Trojans can also be detected as discussed above. furthermore, the technique is compatible with many other approaches; e.g., the suspicious signals can be used in run-time monitoring schemes as their blacklist to be monitored. Feature analysis can be applied to the suspicious list for further Trojan confirmation. This analysis is a complement of side-channel analysis, which aims at the foundry attack model.

Acknowledgment

This work was supported by the National Key R&D Program of China (2018YFB0904900, 2018YFB0904902).

References

[1] S. Adee, "The hunt for the kill switch," *IEEE Spectrum*, vol. 45, no. 5, pp. 34–39, May 2008.

[2] S. Skorobogatov and C. Woods, "Breakthrough silicon scanning discovers backdoor in military chip," in *Proceedings of the 14th International Conference on Cryptographic Hardware and Embedded Systems*, ser. CHES'12. Berlin, Heidelberg: Springer-Verlag, 2012, pp. 23–40. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-33027-8_2

[3] 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, Aug 2014.

[4] R. S. Chakraborty and S. Bhunia, "Security against hardware trojan through a novel application of design obfuscation," in *2009 IEEE/ACM International Conference on Computer-Aided Design - Digest of Technical Papers*, Nov 2009, pp. 113–116.

[5] V. S. Rathor, B. Garg, and G. K. Sharma, "A novel low complexity logic encryption technique for design-for-trust," *IEEE Transactions on Emerging Topics in Computing*, pp. 1–1, 2018.

[6] A. Nejat, D. Hely, and V. Beroulle, "Escalation: Leveraging logic masking to facilitate path-delay-based hardware trojan detection methods," *Journal of Hardware and Systems Security*, vol. 2, no. 1, pp. 83–96, Mar 2018. [Online]. Available: https://doi.org/10.1007/s41635-018-0033-6

[7] J. Dofe and Q. Yu, "Novel dynamic state-deflection method for gate-level design obfuscation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, no. 2, pp. 273–285, Feb 2018.

[8] X. Guo, R. G. Dutta, and Y. Jin, *IP Trust Validation Using Proof-Carrying Hardware*. Cham: Springer International Publishing, 2017, pp. 207–225. [Online]. Available: https://doi.org/10.1007/978-3-319-49025-0_10

[9] E. Love, Y. Jin, and Y. Makris, "Proof-carrying hardware intellectual property: A pathway to trusted module acquisition," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 1, pp. 25–40, Feb 2012.

[10] H. Salmani, M. Tehranipoor, and J. Plusquellic, "A novel technique for improving hardware trojan detection and reducing trojan activation time," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 20, no. 1, pp. 112–125, Jan 2012.

[11] B. Zhou, W. Zhang, S. Thambipillai, J. T. K. Jin, V. Chaturvedi, and T. Luo, "Cost-efficient acceleration of hardware trojan detection through fan-out cone analysis and weighted random pattern technique," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 5, pp. 792–805, May 2016.

[12] X. Ye, J. Feng, H. Gong, C. He, and W. Feng, "An anti-trojans design approach based on activation probability analysis," in *2015 IEEE International Conference on Electron Devices and Solid-State Circuits (EDSSC)*, June 2015, pp. 443–446.

[13] M. Banga and M. S. Hsiao, "Odette: A non-scan design-for-test methodology for trojan detection in ics," in *2011 IEEE International Symposium on Hardware-Oriented Security and Trust*, June 2011, pp. 18–23.

[14] A. Shabani and B. Alizadeh, "Pmtp: A max-sat based approach to detect hardware trojan using propagation of maximum transition probability," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1–1, 2018.

[15] J. Rajendran, V. Vedula, and R. Karri, "Detecting malicious modifications of data in third-party intellectual property cores," in *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, June 2015, pp. 1–6.

[16] T. Reece and W. H. Robinson, "Detection of hardware trojans in third-party intellectual property using untrusted modules," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 3, pp. 357–366, March 2016.

[17] H. A. Amin, Y. Alkabani, and G. M. Selim, "System-level protection and hardware trojan detection using weighted voting," *Journal of Advanced Research*, vol. 5, no. 4, pp. 499 – 505, 2014, cyber Security. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2090123213001446

[18] S. Saha, R. S. Chakraborty, S. S. Nuthakki, Anshul, and D. Mukhopadhyay, "Improved test pattern generation for hardware trojan detection using genetic algorithm and boolean satisfiability," in *Cryptographic Hardware and Embedded Systems – CHES 2015*, T. Güneysu and H. Handschuh, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 577–596.

[19] J. Cruz, F. Farahmandi, A. Ahmed, and P. Mishra, "Hardware trojan detection using atpg and model checking," in *2018 31st International*

Conference on VLSI Design and 2018 17th International Conference on Embedded Systems (VLSID), Jan 2018, pp. 91–96.

[20] S. K. Haider, C. Jin, M. Ahmad, D. Shila, O. Khan, and M. van Dijk, "Advancing the state-of-the-art in hardware trojans detection," *IEEE Transactions on Dependable and Secure Computing*, pp. 1–1, 2018.

[21] M. Hicks, M. Finnicum, S. T. King, M. M. K. Martin, and J. M. Smith, "Overcoming an untrusted computing base: Detecting and removing malicious hardware automatically," in *2010 IEEE Symposium on Security and Privacy*, May 2010, pp. 159–172.

[22] A. Waksman, M. Suozzo, and S. Sethumadhavan, "FANCI: Identification of stealthy malicious logic using boolean functional analysis," in *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, ser. CCS '13. New York, NY, USA: ACM, 2013, pp. 697–708. [Online]. Available: http://doi.acm.org/10.1145/2508859.2516654

[23] J. Zhang, F. Yuan, L. Wei, Y. Liu, and Q. Xu, "VeriTrust: Verification for hardware trust," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 7, pp. 1148–1161, July 2015.

[24] 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, Feb 2017.

[25] M. Oya, N. Yamashita, T. Okamura, Y. Tsunoo, M. Yanagisawa, and N. Togawa, "Hardware-Trojans Rank: Quantitative Evaluation of Security Threats at Gate-Level Netlists by Pattern Matching," *IEICE Transactions on Fundamentals of Electronics Communications and Computer Sciences*, vol. 99, pp. 2335–2347, 2016.

[26] X. Chen, Q. Liu, S. Yao, J. Wang, Q. Xu, Y. Wang, Y. Liu, and H. Yang, "Hardware trojan detection in third-party digital intellectual property cores by multi-level feature analysis," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1–1, 2017.

[27] S. Narasimhan, D. Du, R. S. Chakraborty, S. Paul, F. G. Wolff, C. A. Papachristou, K. Roy, and S. Bhunia, "Hardware trojan detection by multiple-parameter side-channel analysis," *IEEE Transactions on Computers*, vol. 62, no. 11, pp. 2183–2195, Nov 2013.

[28] Y. Yang, L. Wu, X. Zhang, and J. He, "A novel hardware trojan detection with chip id based on relative time delays," in *2017 11th IEEE International Conference on Anti-counterfeiting, Security, and Identification (ASID)*, Oct 2017, pp. 163–167.

[29] 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, Dec 2017.

[30] Y. Cao, C. Chang, and S. Chen, "A cluster-based distributed active current sensing circuit for hardware trojan detection," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 12, pp. 2220–2231, Dec 2014.

[31] Y. Huang, S. Bhunia, and P. Mishra, "Scalable test generation for trojan detection using side channel analysis," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 11, pp. 2746–2760, Nov 2018.

[32] Y. Zheng, S. Yang, and S. Bhunia, "Semia: Self-similarity-based ic integrity analysis," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 1, pp. 37–48, Jan 2016.

[33] 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, Oct 2015.

[34] N. Cornell and K. Nepal, "Combinational hardware trojan detection using logic implications," in *2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)*, Aug 2017, pp. 571–574.

[35] T. F. Wu, K. Ganesan, Y. A. Hu, H. S. P. Wong, S. Wong, and S. Mitra, "TPAD: Hardware trojan prevention and detection for trusted integrated circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 4, pp. 521–534, April 2016.

[36] G. Bloom, B. Narahari, and R. Simha, "OS support for detecting trojan circuit attacks," in *2009 IEEE International Workshop on Hardware-Oriented Security and Trust*, July 2009, pp. 100–103.

[37] M. Abramovici and P. Bradley, "Integrated circuit security: New threats and solutions," in *Proceedings of the 5th Annual Workshop on Cyber Security and Information Intelligence Research: Cyber Security and Information Intelligence Challenges and Strategies*, ser. CSIIRW '09. New York, NY, USA: ACM, 2009, pp. 55:1–55:3. [Online]. Available: http://doi.acm.org/10.1145/1558607.1558671

[38] 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.

[39] "Trust-HUB," http://www.trust-hub.org.

[40] C. Sturton, M. Hicks, D. Wagner, and S. T. King, "Defeating UCI: Building stealthy and malicious hardware," in *2011 IEEE Symposium on Security and Privacy*, May 2011, pp. 64–77.

[41] D. Sullivan, J. Biggers, G. Zhu, S. Zhang, and Y. Jin, "FIGHT-metric: Functional identification of gate-level hardware trustworthiness," in *2014 51st ACM/EDAC/IEEE Design Automation Conference (DAC)*, June 2014, pp. 1–4.

[42] 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*, ser. CCS '14. New York, NY, USA: ACM, 2014, pp. 153–166. [Online]. Available: http://doi.acm.org/10.1145/2660267.2660289

[43] S. Yao, X. Chen, J. Zhang, Q. Liu, J. Wang, Q. Xu, Y. Wang, and H. Yang, "FASTrust: Feature analysis for third-party ip trust verification," in *2015 IEEE International Test Conference (ITC)*, Oct 2015, pp. 1–10.

[44] L.-T. Wang, C.-W. Wu, and X. Wen, *VLSI Test Principles and Architectures: Design for Testability (Systems on Silicon)*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2006.

[45] M. Zou, X. Cui, L. Shi, and K. Wu, "Potential trigger detection for hardware trojans," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, no. 7, pp. 1384–1395, July 2018.

[46] S. Jha and S. K. Jha, "Randomization based probabilistic approach to detect trojan circuits," in *2008 11th IEEE High Assurance Systems Engineering Symposium*, Dec 2008, pp. 117–124.

[47] J. Popat and U. Mehta, "Transition probabilistic approach for detection and diagnosis of hardware trojan in combinational circuits," in *2016 IEEE Annual India Conference (INDICON)*, Dec 2016, pp. 1–6.

[48] S. K. Haider, C. Jin, and M. van Dijk, "Advancing the state-of-the-art in hardware trojans design," in *2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)*, Aug 2017, pp. 823–826.

[49] R. S. Chakraborty, F. Wolff, S. Paul, C. Papachristou, and S. Bhunia, "MERO: A statistical approach for hardware trojan detection," in *Cryptographic Hardware and Embedded Systems - CHES 2009*, C. Clavier and K. Gaj, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 396–410.

[50] Z. Zhou, U. Guin, and V. D. Agrawal, "Modeling and test generation for combinational hardware trojans," in *2018 IEEE 36th VLSI Test Symposium (VTS)*, April 2018, pp. 1–6.

[51] Y. Huang, S. Bhunia, and P. Mishra, "Mers: Statistical test generation for side-channel analysis based trojan detection," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '16. New York, NY, USA: ACM, 2016, pp. 130–141. [Online]. Available: http://doi.acm.org/10.1145/2976749.2978396

[52] J. Cruz, Y. Huang, P. Mishra, and S. Bhunia, "An automated configurable trojan insertion framework for dynamic trust benchmarks," in *2018 Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2018, pp. 1598–1603.