COTD: Reference-free Hardware Trojan Detection in Gate-level Netlist

Hassan Salmani
Howard University
hassan.salmani@howard.edu

Abstract: *This paper presents a novel reference-free hardware Trojan detection technique in gate-level netlist based on controllability and observability analyses. Using an unsupervised clustering analysis, the paper shows that the controllability and observability characteristics of Trojan signals present significant inter-cluster distance from those of genuine signals in a Trojan-inserted circuit such that Trojan signals are easily distinguishable.*

Keywords: Hardware Trojan Detection; Gate-level netlist; Reference-free; Controllability; Observability.

Introduction

The high complexity of modern designs, the constraint of time-to-market window, and the cost restriction of final product highly drive the horizontal design process. The third-party intellectual properties (3PIPs) are widely used while they expose a design to hardware Trojans (HTs) that may tamper with the design and causes its malfunction under very rare circumstances [1].

Hardware Trojans have negligible effects on circuit parameters and rarely become fully activated. Some work have investigated hardware Trojans in early design stages and several techniques have been proposed to study the switching activity of circuit signals and their correlation to identify a hardware Trojan in gate-level netlist [2][3][4][5][6]. Some of the main issues with these techniques are none-zero false positive and false negative rates, significant processing time and their scalability limitations. Furthermore, it has been shown they can be defeated, and a carefully designed Trojan can remain undetected [6][7]. In addition, neither of the techniques are able to detect always-on Trojans such as a ring-oscillator Trojan that is not connected to the main circuit, and presents normal switching activities, but aims to reduce circuit reliability.

While majority of existing techniques mainly generate list of suspicious signals, it remains the responsibility of a design house to still scrutinize the netlist. To be used in practice, a HT detection technique should be easily integrable into commercial circuit design flows. Furthermore, they should present 0 false positive and false negative HT detection rates. In addition, authentication time should be small to meet economic constraints.

This paper introduces the Controllability and Observability for hardware Trojan Detection (COTD) technique for hardware Trojan detection in gate-level netlist. Controllability reflects the difficulty of setting a signal line to a required logic value, and observability reflects the difficulty of propagating the logic value of the signal line to observation points. The COTD technique combines the controllability and observability analyses and an unsupervised machine learning to determine a circuit is Trojan inserted or Trojan free and to evaluate false positive and false negative rates. ***The contributions of this work are***: (1) COTD does not need any golden circuit as a reference, (2) COTD does not need any test pattern application for Trojan partial/full activation, (3) COTD presents 0 false positive and false negative Trojan detection rates, (4) COTD can be readily integrated into current commercial integrated design flows, and (5) COTD presents time complexity of linear function of the number of signals in the circuit. COTD has been applied to all gate-level Trojans available on Trust-HUB [8] and introduced by DeTrust [6], always-on Trojans, and HaTCh [7], and COTD has successfully detected all of them in a fraction of a minute with 0 false positive and false negative rates.

The COTD Flow

Figure 1 presents the COTD flow. It takes gate-level netlist as the input, and the *Controllability and Observability Analyses* step performs the controllability and observability analyses to determine controllability and observability values, i.e. *CC*0: combinational 0-controllability, *CC*1: combinational 1-controllability, and *CO:* combinational observability [9]. Afterwards, the *Unsupervised Clustering Analysis* step is performed to cluster signals based on their controllability and observability values. Two signal lists are produced: *Trojan Signals List* and *Genuine Signals List*. The Genuine Signals List only contains all genuine/original signals in the netlist. The Trojan Signal List only contains all Trojan signals if any Trojan exists. The circuit is Trojan free if the list is empty; otherwise, it is Trojan inserted.

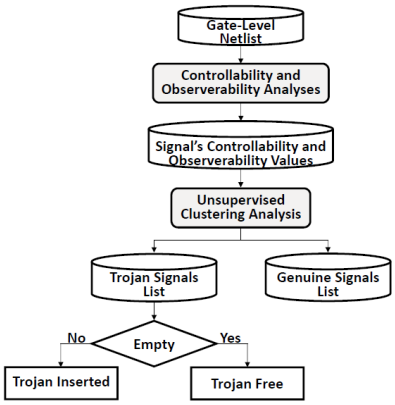

Figure 1. The COTD Flow

One major issue associated with hardware Trojan detection is the lack of golden circuit as a reference. Unsupervised learning is a type of machine learning algorithm used to explore datasets consisting of input data without labeled responses. Cluster analysis or clustering is the most

common unsupervised learning method used for grouping data and two factors determine the quality of clustering: intra-cluster distance and inter-cluster distance. While clustering, it is desired to maximize inter-cluster distances and to minimize intra-cluster distances. One of the most common clustering algorithms is k-means clustering that strives to meet both goals at the same time [10].

The COTD technique obtains the pairs <CC, CO> for every signal in a circuit (CC=(CC0^2+CC1^2)^1/2). Then it passes them in the form of a 2-dimensional dataset to the k-means algorithm where k=3. Signals can be divided into three clusters: (i) genuine signals whose both CC and CO values are small, and Trojan signals (ii) with large CC values or (iii) with large CO values. Therefore, the k parameter is set to 3 to distinguish Trojan signals from genuine signals. It is possible a Trojan signal to have both large CC and CO values. Such a Trojan signal is still distinguished and reported in Trojan Signal List.

As Trojan signals should have large CC or CO values to avoid their detection as opposed to genuine signals that should have small CC or CO values to be testable, the 3-means algorithm effectively separates all genuine signals in one cluster and isolates all Trojan signals in different clusters with considerable inter-cluster distances from the cluster of genuine signals. The COTD technique can effectively determine whether a circuit is Trojan inserted without need for any golden circuit.

Usability of any HT detection technique strongly depends on its time complexity. Figure 2 shows a comparison of time complexity of different countermeasures. COTD outperforms and presents a linear relationship with the number of circuit inputs. The inconsiderable time complexity of COTD makes it a perfect choice for hardware Trojan detection in gate-level netlist.

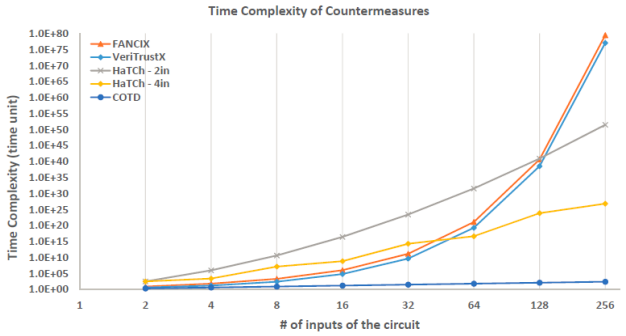

Figure 2. Comparison of computational complexities of different.

Results

The COTD technique is applied to gate-level Trojan-inserted netlist provided on Trust-HUB [8] and some other circuits infected by Trojans proposed by DeTrust [6], always-on Trojans and HaTCh [7].

Figure 3 depicts the results of unsupervised k-mean clustering with k=3 for s38417-T100 [8]. The figure shows genuine signals are localized in the bottom left corner of graph while Trojan signals are mainly located in three other corners of graph where CC, CO, or both are high. Furthermore, Figure 3 shows the significant inter-cluster distance between Trojan signals and genuine signals.

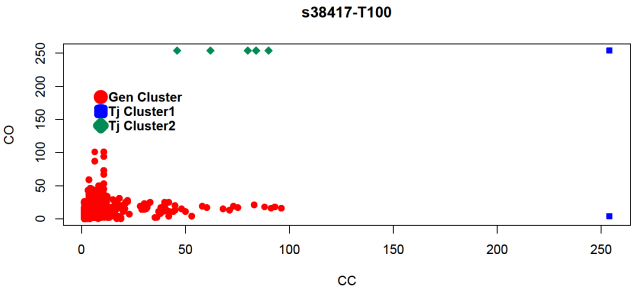

Figure 3. Unsupervised k-mean clustering with k=3 for s38417-T100 [8].

Table 1 presents the maximum and minimum of |<CC, CO>| for signals in the genuine circuit and Trojan circuit for Trust-HUB gate-level netlist, respectively. The results clearly indicate even the minimum magnitude of |<CC, CO>| for Trojan signals is much higher than the maximum magnitude of |<CC, CO>| for genuine signals. The larger **this difference is, the stealthier a Trojan behaves. All** Trojans have a minimum |<CC, CO>| value above 254 while the maximum |<CC, CO>| value for genuine signals ranges between 12 and 101.

Table 1. Analysis of controllability and observability values for genuine and Trojan

Trojan	Genuine Circuit	Trojan Circuit
	Max $\|<CC, CO>\|$	Min $\|<CC, CO>\|$
s38417—T100	101.57	254.03
s38417—T200		254.05
s38417—T300		255.13
Ethernet–T700	88.01	254.01
Ethernet–T710		254.01
Ethernet–T720		254.03
Ethernet–T730		254.00
RS232—T1000	13.19	254.00
RS232—T1100		254.03
RS232—T1200		254.00
RS232—T1300		254.00
RS232—T1400		254.00
RS232—T1500		254.00
RS232—T1600		254.00
s15850—T100	86.15	254.00
s35932—T100	12.08	254.03
s35932—T300		254.00
s38584—T200	35.41	256.26
s38584—T300		254.13
vga_lcd—T100	42.05	256.00

Table 2 presents the inter-cluster distances between signals after executing unsupervised k-mean clustering with k=3 on Trojan-free gate-level Trust-HUB benchmarks in Table 1. Detailed analyses show that the inter-cluster distance of Trojan signal clusters and genuine signal clusters in Table 1 is much larger that the inter-cluster distance of genuine signal clusters. The average inter-cluster distance for Trojan-inserted circuits is about 286.85 and the average inter-cluster distance for Trojan-free circuits in Table 2 is about 18.45 that is about 15 times smaller. This results support zero false positive and zero negative of the COTD technique.

Figure 4 depicts the three clusters for Trojan-free gate-level s38417 benchmark. The figure shows there are overlaps between clusters and it is not possible to group signals into well-separated clusters. While Trojan signals present high controllability and observability values to realize stealthy Trojans, clusters of genuine signals present small inter-cluster distances even if some are possibly overestimated. Therefore, the characteristics of genuine signals effectively makes the false positive rate zero.

Table 2. Inter-cluster distances in Trojan-free gate-level Trust-HUB benchmarks [8]

Circuit	Inter-cluster Distance		
	Clst1-Clst2	Clst1-Clst3	Clst2-Clst2
s38417	45.12	31.88	45.87
Ethernet	13.79	41.38	28.32
RS232	4.71	3.56	5.52
s15850	16.78	14.85	16.38
s35932	6.14	6.68	3.42
s38584	4.58	9.41	13.60
vga_lcd	34.16	23.73	23.36

The results signify realizing a stealthy hardware Trojan getting rarely activated would result in Trojans whose signals have significantly high controllability and observability values. Otherwise, they are easily detectable. For example, Table 3 shows s35932-T200 [8] presenting very small CC and CO close to those for the genuine circuit. The results show the Trojan experiences high switching activity (# of Trojan Activation) in only 4261 test clock cycles.

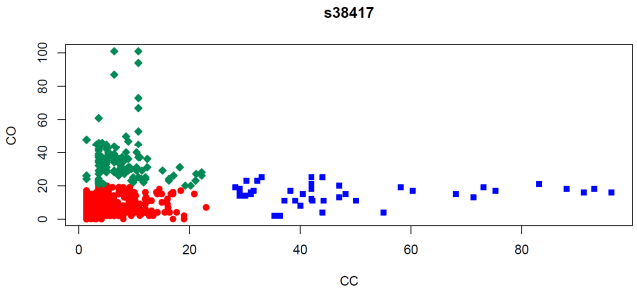

Figure 4. Unsupervised k-mean clustering with k=3 for hardware Trojan-free s38417.

Any signal situated in the zone of controllable and observable signals including Trojan signals would experience switching activity. Hence, it is not possible to implement a Trojan whose signals have controllability and observability in the range of genuine signals in a hosting circuit but remain inactive during circuit testing. Such a behavior flags them as untestable signals, and it would be followed with their thorough analyses.

Table 3. A Gate-level Trojan with low controllability and observability values [8].

Trojan	Genuine Circuit		Trojan Circuit		# of Trojan Activation	# Test clock cycles
	MAX CC	MAX CO	MAX CC	MAX CO		
s35932-T200	8.48	12	17.02	20	42	4261

To evaluate switching activity of signals, Figure 5 shows the frequency of switching activity in each original circuit in Table 1 after applying random test vectors for 500 test cycles. While the number of transitions are sorted ascending in each circuit, the horizontal axis indicates the signal index and the vertical axis presents the number of transitions from '0' to '1' or '1' to '0'. The axes are presented in logarithmic scales to provide a finer look at signals with low switching activities. Using the signal index, it is possible to observe how many signals exist with switching activities below a certain number. For example, Ethernet benchmark has four signals whose switching activity is two. The circuits have different sizes with different functionality, and larger circuits like Ethernet benchmark have larger number of signals with low switching activities in comparison with smaller circuits like RS232 benchmark.

While the circuits contain signals with wide range of switching activities, there is zero signal with zero switching activity even after applying a small number of random test patterns.

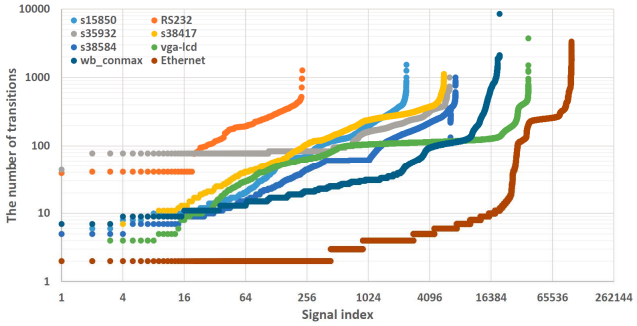

Figure 5. Switch activity of signals in Trojan-free circuits after applying random test vectors for 500 test cycles.

The results emphasize any signal whose the controllability and observability values even if overestimated is in the range of controllability and observability values for the original signals experiences switching activity. This also holds true for a Trojan signal has controllability and observability values less than or close to the maximum |<CC, CO>| of genuine signals. This fact is already discussed in Table 3 where a hardware Trojan consisting of signals whose maximum CC and CO are slightly larger than the maximum CC and CO of genuine signals in hosting original circuits. Table 2 shows considerable switching activities of Trojan signals after applying few thousands test patterns.

It should be noted that there might be some signals in the original circuit that are undetectable meaning either not controllable or not observable. A signal can be undetectable if it is unused, blocked, or redundant. Undetectable signals are distinguishable using ATPGA tools; therefore, a Trojan signal cannot be implemented as an undetectable signal.

It is so valuable to fully recover an inserted Trojan in a circuit to understand the purpose of adversary. Identifying Trojan trigger and Trojan payload circuitry present detailed

information about Trojan implementation. Further, it would also make it possible to determine (i) which signals are being used as inputs for the Trojan trigger, and (ii) which signals are targeted by the Trojan payload.

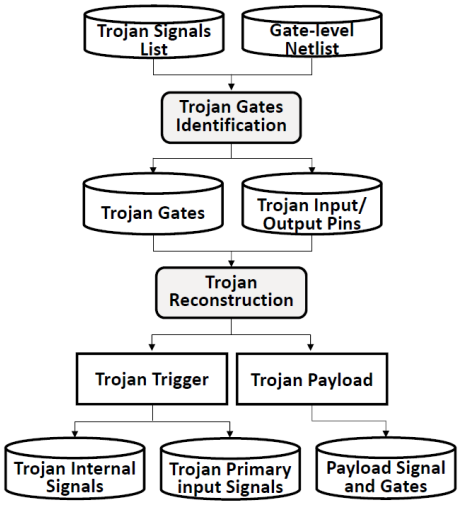

Figure 6. The hardware Trojan recovery flow for gate-level netlist.

After isolating Trojan signals using COTD, it is possible to fully recover a hardware Trojan inserted in a gate-level netlist. Figure 6 presents the proposed hardware Trojan recovery flow in a gate-level netlist. Trojan signals identified by COTD and Trojan-inserted circuit are used as inputs. The Trojan Gates Identification step extracts Trojan gates, and their input and output pins. With this information, it is possible to execute the Trojan Reconstruction step. In this step, Trojan trigger and Trojan payload circuitry are restored. Signals connected to Trojan gates' pins are obtained, and the interconnection between Trojan gates is reconstructed. Further, it is determined which signals from the main circuit are being used as Trojan triggering signals, and which signals in the main circuit are attacked by the Trojan payload. Any signal that drives a Trojan gate and is not driven by any Trojan gate is identified as a Trojan triggering signal. Any signal that is not a Trojan signal but passing through a Trojan gate is identified as a Trojan payload signal. Any gate whose one of inputs is a payload signal composes the Trojan payload circuitry. The remaining Trojan gates compose the Trojan trigger circuitry. The hardware Trojan recovery flow is implemented in Synopsys' design compiler and only consists of about 100 lines, and its complexity is an order of the number of Trojan signals. The flow is being applied to all gate-level netlist on Trust-HUB, and all Trojans are successfully recovered.

As a sample, Figure 7 presents the output of flow for the s38417-T100 Trojan on Trust-HUB. While Figure 7 shows a part of report, the report includes (1) Trojan gates for each of which input and output pins with their connected nets are reported, (2) Trojan internal signals are distinguished, (3) Trojan triggering signals are detected, and (4) Trojan payload gates, along with targeted nets are identified.

Comparing the report in Figure 7 with the Trojan inserted in s38417-T100 circuit shows the proposed hardware Trojan recovery flow can perfectly extract the inserted Trojan. While a Trojan circuit can be completely isolated, it is also possible to clean up the Trojan-inserted circuit.

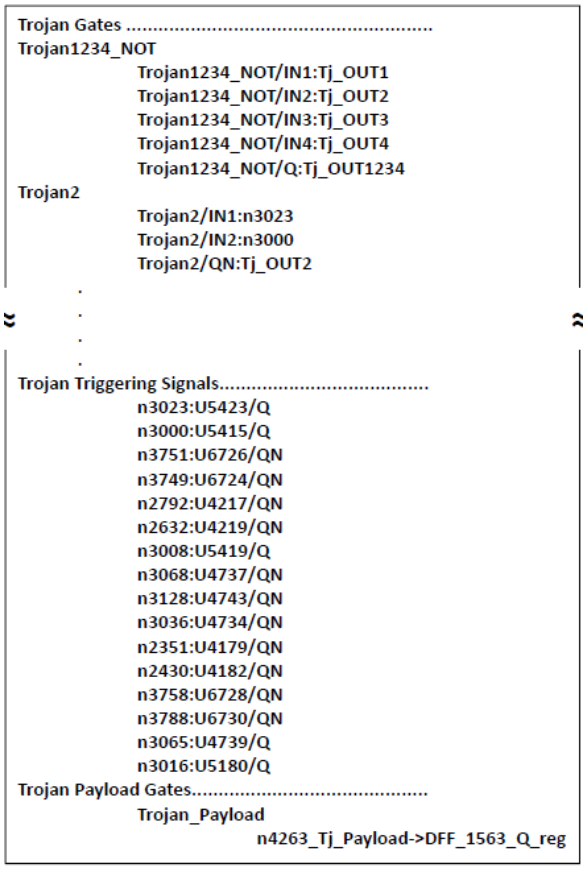

Figure 7. The restored s38417-T100 Trojan.

References

[1] M. Tehranipoor and C.Wang, "Introduction to Hardware Security and Trust," Springer New York Dordrecht Heidelberg London, August 2011

[2] B. Çakir, and S. Malik, "Hardware Trojan Detection for Gate-level ICs Using Signal Correlation Based Clustering," DATE 2015.

[3] M. Oya, and *et. al*, "A Scorebased Classification Method for Identifying Hardware-Trojans at Gate-level Netlists," DATE 2015.

[4] J. Zhang and *et. al*, "VeriTrust: Verification for Hardware Trust," DAC 2013

[5] A. Waksman and *et. al*, "FANCI: identification of stealthy malicious logic using Boolean functional analysis," CCS 2013.

[6] J. Zhang and *et. al*, "DeTrust: Defeating Hardware Trust Verification with Stealthy Implicitly-Triggered Hardware Trojans," CCS 2014

[7] S. K. Haider and *et. al*, "HaTCh: A Formal Framework of Hardware Trojan Design and Detection," Cryptology ePrint Archive, Report 2014/943, 2014.

[8] H. Salmani and *et. al*, "On Design vulnerability analysis and trust benchmark development" ICCD 2013.

[9] C. Wu, L. Wang, and X. Wen "VLSI Test Principles and Architectures: Design for Testability," The Morgan Kaufmann Series in Systems on Silicon, 2006.

[10] G. A. F. Seber, "Multivariate Observations," John Wiley & Sons, Inc., 1984.