FIGHT-Metric: Functional Identification of Gate-Level Hardware Trustworthiness

Dean Sullivan*, Jeff Biggers*, Guidong Zhu+, Shaojie Zhang*, and Yier Jin*

*Department of Electrical Engineering and Computer Science, University of Central Florida
+ Guangzhou Haige Communications Group Co., Ltd
yier.jin@eecs.ucf.edu

ABSTRACT

To address the concern that a complete detection scheme for effective hardware Trojan identification is lacking, we have designed an RTL security metric in order to evaluate the quality of IP cores (with the same or similar functionality) and counter Trojan attacks at the pre-fabrication stages of the IP design flow. The proposed security metric is constructed on top of two criteria, from which a quantitative security value can be assigned to the target circuit: 1) Distribution of controllability; 2) Existence of rare events. The proposed metric, called FIGHT, is an automated tool whereby malicious modifications to ICs and/or the vulnerability of the IP core can be identified, by monitoring both internal node controllability and the corresponding control value distribution plotted as a histogram. Experimentation on an RS232 module was performed to demonstrate our dual security criteria and proved security degradation to the IP module upon hardware Trojan insertion.

Keywords

Security Metric, Trustworthy Hardware

1. INTRODUCTION

An increasingly large third-party Intellectual Property (IP) market provides consumers with more options when designing electronic systems, and reduces the development time and expertise needed to compete in a marker where profit-windows are very narrow [4]. However, one key issue that has been neglected is the security of electronic systems with integrated third-party IP cores. Historically, IP consumers put more weight in IP functionality and performance than IP security. The "prejudice" against the development of robust security policies is reflected in the IP design flow, where IP core specification often only covers functionality and performance measurements. This lack of security covering third-party IPs is a real-world threat; recently, a large body of side-channel based attacks have been reported to have leaked sensitive information from systems that were purportedly mathematically unbreakable [7].

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

http://dx.doi.org/10.1145/2593069.2596681

The emergence of hardware Trojans embedded in third-party IP cores has largely re-shaped the IP transaction market, and there are currently no complete detection schemes for identifying hardware Trojans [6, 8]. Some Trojan detection methods such as side-channel fingerprinting combined with statistical analysis have shown mostly positive results [3, 5], but these approaches are only useful after the chip has been fabricated. High-cost reverse engineering techniques are also potentially effective in determining whether manufactured chips are genuine. However, both of these methods suffer because they can only be used on a sample of chips with no guarantee that the remaining untested chips are Trojan-free [3]. What's more, the question remains: *how can we comparatively assess the security of levels of IP cores, from different vendors, with the same/similar functionality.* In other words, rather than being limited to applying high cost Trojan detection methods at the post-fabrication stage, IP consumers should have the ability to estimate the vulnerability of the delivered IP cores before they are integrated into the hardware infrastructure.

To address this requirement, we propose a novel security metric that can quantitatively measure the likelihood that a given IP core contains malicious logic and/or how vulnerable the IP core is to hardware Trojans attacks. The crux upon which the metric is based lies in searching for low controllability nodes in IP cores, an idea originally explored in [9]. However, as will be addressed in section 3.1, the algorithm has several shortcomings. In order to address these shortcomings, and address our new security metric, we developed an algorithm called Functional Identification of Gate-level Hardware Trustworthiness (FIGHT) to accurately evaluate the controllability of internal nodes for synthesized netlists (control values). Our tool deals with both large sequential logic and feedback elements accurately, as well as providing a means to evaluate the security vulnerability of a given IP prior to system integration. Part of the work herein has been submitted to the CSAW Embedded System Challenge hosted by NYU-Poly in 2013 where our work was awarded Second Place [1].

2. RT-LEVEL SECURITY METRIC

Hardware Trojan detection methods targeting RT-level designs rely on security enhanced functional testing in order to activate any malicious logic embedded in the design. Using this knowledge, attackers intentionally obfuscate their Trojans so that they can circumvent these testing procedures by primarily relying on low controllability nodes [9], which

prevents accidental activation through ATPG generated or randomly selected input testing patterns. Two kinds of low controllability nodes can be utilized by attackers in the context of hardware Trojans: first, attackers can use existing low controllability nodes among the target design to trigger the Trojan; second, attackers can insert additional logic to create low controllability nodes and thereby use them to trigger a Trojan. In either case, low controllability nodes represent weak points in a design that are easily leveraged by attackers for stealthy Trojan design[1].

In addition to low controllability nodes, there exists a large amount of medium controllability nodes, and even though Trojans triggered by such events would be more easily detected by random input patterns, the large number of such events within any give SoC makes it impossible to exhaustively test. Furthermore, the analysis of medium probability events is unproductive simply because many of the events exist legally in an IP, which makes it difficult to differentiate between legitimate and illegitimate events. Correspondingly, the analysis of RTL IP cores reveals that a well-designed circuit will have the majority of its internal nodes located in the medium controllability domain. If we plot all control values in a histogram with the X-axis indicating all control values and the Y-axis indicating the count of all unique control value, we will generate a peak close to the medium controllability domain (See Figure 1(a)). That is, relatively few nodes will be associated with low controllability[2]. However, for a poorly designed circuit, and/or a Trojan infected circuit, according to the low controllability propagation rule (See Observation 1), there will exist more peaks at the relatively low controllability domain, so that the overall control value distribution will have more than one dominant peak (See Figure 1(b) - (h) and Figure 1(g)). The distribution will then serve as the critical component in deciding the security levels of any given IP module.

According to the model offered above, we have designed a security metric in order to evaluate the quality of IP cores (with the same or similar functionality) to counter Trojan attacks at the pre-fabrication stages of the IP design flow. The proposed security metric is constructed on top of two criteria, from which a quantitative security value will be assigned to the target circuit: 1) Distribution of controllability; 2) Existence of extremely low controllability nodes. Based on control value calculations of every internal node in the target circuit, a zero or positive value will be assigned in order to delineate its level of security; the larger the number, the stronger the security. For circuits with extremely low controllability nodes, a value of 0 will be assigned to indicate that some internal nodes are unlikely to be activated during traditional functional testing, and to highlight to the IP user to check those nodes for the presence of malicious logic. Additionally, for target circuits which do not contain extremely low controllability nodes, a histogram of all internal node control values will be matched with its closest normal distribution using the Kruskal-Wallis test. The derived matching parameter p will then be used to represent its security level. Using the matching paramater p to represent the security level helps IP users compare IP cores which are of similar, but not unique, functionality.

Observation 1: Rules of controllability propagation[3]. In most of the non-cryptographic circuits where XOR logic is not the dominant logic, low controllability nodes are likely to be propagated to their decedents causing the low controllability nodes accumulation effect.

The metric can be broken into three steps:

Step I: Normal Distribution Generation. After we collect all control values within the target circuit, we will calculate their logarithmics to re-scale the internal node values from [0,1] to [-∞, 0]. A histogram of all control values will then be plotted, and based on this histogram, a best-fit normal distribution will be generated (See Figure 1).

Step II: Rare Events Identification. The generation of the best-fit normal distribution also provides us the mean value μ and the standard deviation σ. A boundary is then outlined at the point $\mu - 3 \times \sigma$ so that all nodes with control values located to the left of $\mu - 3 \times \sigma$ will be treated as extremely low controllability nodes. For any target circuit with extremely low controllability nodes, its security metric value will be assigned 0 to reflect the fact that those nodes can be easily targeted for the insertion of hardware Trojans, or that those nodes already serve as Trojan triggers.

Step III: Kruskal-Wallis (K-W) Test. For all other target circuits that do not contain extremely low controllability nodes, the control value distribution will then be evaluated to measure its deviation from the derived normal distribution. To quantitatively calculate the deviation between the derived normal distribution and the actual histogram, the K-W test is applied to verify whether these samples originate from the same distribution by comparing the medians of the samples and returning the p-value as the result. After applying the test, the matching parameter p will be assigned to the security level of the circuit where a value larger than 0.05 means a good fit.

3. FIGHT

3.1 FANCI: Functional Analysis for Nearly-unused Circuit Identification [9]

The FANCI tool was designed to evaluate an IP core encoded as a gate-level netlist for maliciously inserted logic. The tool evaluates wires and gates based on value dependence, which means the input functionally controls the output of a given gate. A new concept in measuring internal node dependency and controllability with respect to primary inputs is proposed, called the control value [9].

However, the FANCI algorithm suffers from several security limitations, one of which is based on sequential logic. These types of hardware Trojans are not triggered by a single combinational input, but by a series of inputs. This type of Trojan will not be easily detected by FANCI because sequential logic is evaluated as its combinational equivalent. That is, the tool does not have a built in time-domain specification. What's more, the detection scheme evaluates sequential blocks as non-clocked combinational elements and builds an equivalent truth-table to be used for control value computations, which physically misrepresents the circuit. Feed-

[1]Please be aware that the measurements of "low", "medium", "extremely low" controllability are heuristic concept and will vary between different circuits throughout the paper.

[2]This rule is valid for many functional IP cores, excluding the cryptographic IP modules. Within cryptographic circuits, most of internal nodes are of high controllability and are very sensitive to input changes. More results on judging cryptographic IPs will be presented in our later work.

[3]The detailed explanation of this observation is omitted due to page limit.

back loops are also misrepresented, and difficult to reduce without disrupting the primitive Boolean of the circuit.

As part of the 2013 ESC competition for CSAW, we designed three hardware trojans on an LFSR core that leveraged both sequential logic and feedback loop shortcomings in FANCI for Trojan activation. All these Trojans evaded the FANCI detection method.

3.2 FIGHT: Functional Identification on Gate-level Hardware Trustworthiness

In order to accurately represent hardware Trojans inserted along a D-flip-flop chain, a model for time-domain functional analysis is needed. A time-domain specification is also necessary to ensure the physical (circuit level) representation of sequential logic elements match their software equivalents. This would also solve the problem of *pathological pipelined backdoors* where chains of logic inserted across multiple modules will not have to be severed. Feedback loop functionality can similarly be addressed, whereby the feedback is simply a wire connecting the output to the input across a new time domain; this is true for both sequential and combinatorial feedback logic.

Our algorithm makes improvements by first creating a directed graph where all gates are nodes, and the wires are edges in the graph connecting the inputs to outputs of each gate. We use the strongly connected components concept from graph theory to find key nodes that form cycles within the netlist representation. This is done to account for sequential feedback loops whereas the feedback edge is removed, and a new node is created with an edge connected to the input of the node from which the feedback edge was originally connected. Then a topological sort is performed on the directed graph and expressions built based on the returned result. For every sequential element, we attach a time-domain to all inputs of the expression of that gates Boolean. When the next sequential element is encountered the time-domain is incremented and the procedure repeated. For the FIGHT tool, we retained the same control value and heuristics calculations as FANCI [9]. The algorithm for FIGHT is shown below:

```
Algorithm: Netlist Parser and Expression Builder
1: for all modules m do
2:  create directed graph dg of m
3:   for all strongly connected components t of dg do
4:    if t is a sequential logic gate then
5:     for outgoing edges [e] from t to [x]
6:      if path exists from [x] to t
7:       remove edge e in dg
8:       create node t' in dg
9:       create edge e' from t' to x
10:     end if
11:    end for
12:   end if
13:  end for
14:  for all nodes n in dg do
15:   for all outgoing edges e of n do
16:    if n is sequential logic gate then
17:     T <- TruthTable(FanInTree(TimeDomain(e)))
18:    else
19:     T <- TruthTable(FanInTree(e))
20:    end if
21:   ...continue from line 6 of FANCI algorithm [5]
```

In order to evaluate the FIGHT detection scheme, we reverse engineered the detection mechanism proposed by FANCI. We first check if FANCI evaluates a Trojan infected DFF chain as its physical equivalent, or as a buffer by printing out each expression evaluated from the netlist. The same procedure is repeated for FIGHT, where the generated expressions should reflect the time-domain specification. We then compared the control values of FANCI against FIGHT. Shown below for reference is the netlist:

```
                Trojan DFF Netlist
**************************************************
module top_4DFF (data_in, clock, trigger, data_out);
  input data_in, clock;
  output trigger, data_out;
  wire   Q1, Q2, Q3, n3, n4;

  DFFPOSX1 Q1_reg (.D(data_in), .CLK(clock), .Q(Q1));
  DFFPOSX1 Q2_reg (.D(Q1), .CLK(clock), .Q(Q2));
  DFFPOSX1 Q3_reg (.D(Q2), .CLK(clock), .Q(Q3));
  DFFPOSX1 data_out_reg (.D(Q3), .CLK(clock),
  .Q(data_out));
  BUFX2 U5 (.A(n4), .Y(n3));
  INVX1 U6 (.A(n3), .Y(trigger));
  NAND3X1 U7 (.A(Q2), .B(Q1), .C(Q3), .Y(n4));
endmodule
```

Our assumption was that the wires along the D-FF chain would all be reduced to the global input via line 4 of the FANCI algorithm, whereby a truth table is built as a function of the fan-in tree of all the input wires in a module. This means that all of the expressions built along a D-FF chain will reduce to a single input. This is shown in the netlist above for gate U5. Correspondingly, any gate that uses taps along that chain will also be reduced to a single input. Functionally this means that a 3-input Trojan gate designed using taps along the DFF chain will be reduced to data-in and the associated control value will be 1.

The expressions and control value calculations were performed using FIGHT and the results, along with computed control values, are given below.

```
Output: trigger
  ('data_in_2:0.25','data_in_3:0.25','data_in_4:0.25')
Output: n4
  ('data_in_2:0.25','data_in_3:0.25','data_in_4:0.25')
Output: n3
  (data_in_2:0.25','data_in_3:0.25','data_in_4:0.25')
Output: Q3  ('data_in_4:1')
Output: Q2  ('data_in_3:1')
Output: Q1  ('data_in_2:1')
```

Note that because gate U5 was not reduced to a single buffer expression, but is instead split across its corresponding time-domain, it gives a more accurate control value computation. In addition, the input wires to the Trojan trigger are each unique so that the trigger control value, with respect to its fan-in tree, will not be misrepresented.

4. DEMONSTRATION

We designed our experiment to evaluate the security of a UART module before and after Trojan insertion. Using FIGHT, we evaluated one golden version and nine Trojan infected variants [2] by calculating the control values for all triggering nodes including primary inputs as well as internal nodes. As described in Section 2, for each IP module under test, we plotted a histogram of all control values on a semi-log scale. Figure 1 (a) shows the genuine UART circuit and figures 1(b) - (j) show the Trojan-infected UART circuit where all nine Trojan types have been used.

We applied our dual security criteria to develop a quantitative measure of security for each circuit. At the first level, control values are computed in all internal nodes in the circuit. This is performed in order to assign a security level value to indicate either the potential existence of an

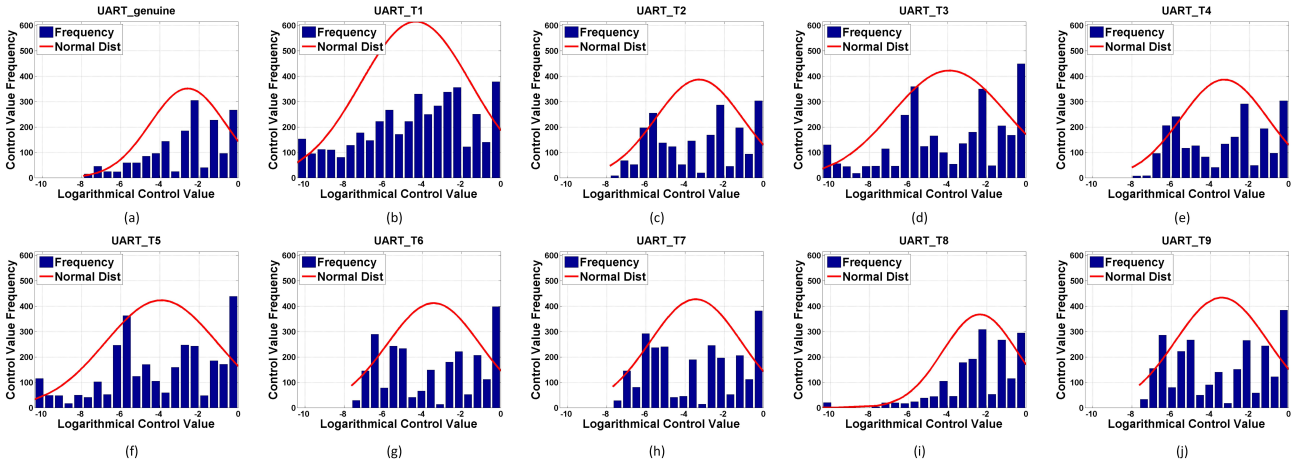

Figure 1: Control Values Histogram and the Best-Fit Normal Distribution for UART Module (a) Genuine (b) Trojan Type 1 (c) Trojan Type 2 (d) Trojan Type 3 (e) Trojan Type 4 (f) Trojan Type 5 (g) Trojan Type 6 (h) Trojan Type 7 (i) Trojan Type 8 (j) Trojan Type 9

Table 1: Security Metric for genuine and Trojan-Infected RS232 Circuits

	Gen	T1	T2	T3	T4	T5	T6	T7	T8	T9
Rare Nodes (Y/N)	N	N	N	N	N	N	N	N	Y	N
K-W Test:	5.45e-02	1.59e-03	4.17e-03	3.97e-03	6.65e-03	4.29e-03	2.90e-03	4.17e-03	1.86e-01	1.99e-03
Security Metric:	5.45e-02	1.59e-03	4.17e-03	3.97e-03	6.65e-03	4.29e-03	2.90e-03	4.17e-03	0	1.99e-03

inserted Trojan, or that the target circuit is likely vulnerable to hardware Trojan attacks. At the second level, for all of the control values corresponding to the entire circuit, we fit the measured distribution with a normal distribution to set a boundary of security for K-W Test parameter p. As shown in the Table 1, the genuine circuit is assigned the highest security level.

From figure 1 (a), it is apparent that there are several low controllability nodes so that the golden model is potentially vulnerable to attacks. For example, it shows that the majority of control values are distributed within a range [-3,-1], but that there exist several nodes with low probabilities. However, there do not exist any nodes below the boundary $\mu - 3 \times \sigma$ outlined in Step II of Section 2.

Figures 1 (b) - (j) indicate that the inclusion of hardware Trojans deteriorate the security level of the target circuit by largely affecting the control value distribution. The inserted Trojans have shifted the peak control values distribution to the left such that the accumulation of node control values are found to be at -6 whereas the original control values were found to be distributed around -2. The large number of nodes with low controllability indicates that the IP core is an easy target for attacks, or that the IP core may already contain malicious logic. In either event, the cost to exhaustively include all testing patterns in an effort to trigger any malicious logic would be prohibitively exorbitant.

As we mentioned earlier, the real power of the proposed metric is to provide a quantitative metric for IP users to compare security levels of IP cores with similar functionality, but supplied by different vendors. It also provides IP vendors with another means of supporting their claim that their IP cores are more secure than others.

5. CONCLUSION

A security metric is developed to quantitatively measure the security level of any IP core. The developed metric provides IP users with a valuable reference attempting to compare the quality of IP cores (with the same or similar functionality). The metric has been demonstrated on an RS232 module, along with nine unique Trojan designs, that the insertion malicious logic will degrade the module's security.

Acknowledgements

This work was funded in part by the NSF grant CNS 0958510, 1319105, and Intel.

6. REFERENCES

[1] *https://esc.isis.poly.edu/*.

[2] *https://www.trust-hub.org/*.

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

[4] E. Greenbaum. Open source semiconductor core licensing. *Harvard Journal of Law & Technology*, 25(1):131–157, 2011.

[5] Y. Jin and Y. Makris. Hardware Trojans in wireless cryptographic ICs. *IEEE Design and Test of Computers*, 27:26–35, 2010.

[6] R. Karri, J. Rajendran, K. Rosenfeld, and M. Tehranipoor. Trustworthy hardware: Identifying and classifying hardware Trojans. *IEEE Computer*, 43(10):39–46, 2010.

[7] P. Kocher, J. Jaffe, and B. Jun. Differential power analysis. In *Advances in Cryptology – CRYPTO'99*, pages 789–789. 1999.

[8] M. Tehranipoor, H. Salmani, X. Zhang, X. Wang, R. Karri, J. Rajendran, and K. Rosenfeld. Trustworthy hardware: Trojan detection and design-for-trust challenges. *Computer*, 44(7):66–74, 2011.

[9] A. Waksman, M. Suozzo, and S. Sethumadhavan. FANCI: Identification of stealthy malicious logic using boolean functional analysis. In *Proceedings of the ACM SIGSAC Conference on Computer & Communications Security*, CCS '13, pages 697–708, 2013.