EDA Tools Trust Evaluation through Security Property Proofs

Yier Jin
Department of Electrical Engineering and Computer Science, University of Central Florida
{yier.jin@eecs.ucf.edu}

***Abstract*—The security concerns of EDA tools have long been ignored because IC designers and integrators only focus on their functionality and performance. This lack of trusted EDA tools hampers hardware security researchers' efforts to design trusted integrated circuits. To address this concern, a novel EDA tools trust evaluation framework has been proposed to ensure the trustworthiness of EDA tools through its functional operation, rather than scrutinizing the software code. As a result, the newly proposed framework lowers the evaluation cost and is a better fit for hardware security researchers. To support the EDA tools evaluation framework, a new gate-level information assurance scheme is developed for security property checking on any gate-level netlist. Helped by the gate-level scheme, we expand the territory of proof-carrying based IP protection from RT-level designs to gate-level netlist, so that most of the commercially trading third-party IP cores are under the protection of proof-carrying based security properties. Using a sample AES encryption core, we successfully prove the trustworthiness of Synopsys Design Compiler in generating a synthesized netlist.**

I. Introduction

The increasingly globalized Integrated Circuit (IC) supply chain has recently given rise to questions regarding chip trustworthiness. The fundamental concern is the potential inclusion of malicious functionality, known as hardware Trojans, which may cause erroneous behavior, steal sensitive information, incapacitate, or even destroy a chip. The potential implications of this problem have resulted in a large body of current research on this topic, with the majority of the efforts focusing on prevention and detection at the post-silicon stage [1], [2], [3], [4], [5], assuming that the culprit will act at the manufacturing site. However, because of the time-to-market pressure and the request to lower design costs, circuit designers and system integrators rely more on third-party IP cores and commercial EDA tools than ever before. Even under the circumstance that more researchers started to worry about the trustworthiness of RTL designs, and have developed various solutions to ensure the security of RTL design in the form of HDL code [6], [7], [8], [9], little work has been done to ensure the trustworthiness of EDA tools. Even worse, for fear of losing ownership of the IP cores and to prevent chip counterfeiting, many design houses will only deliver a synthesized netlist instead of RTL code, which, in turn, complicates researcher's efforts to assess the trustworthiness of third-party IP cores. Due to a lack of both gate-level netlist trust evaluation and trusted EDA tools, all previously proposed RT-level trust evaluation methods are invalidated. This is because the trusted RTL design may be contaminated by untrusted EDA tools, where malicious logic has been inserted into a synthesized or routed netlist. Given the complexity of modern EDA tools, it is fairly challenging, if not impossible, to thoroughly check all the software code that generates the executable EDA programs. Therefore, in this paper, we propose to evaluate the trustworthiness of EDA tools through their functional behavior. A trust evaluation framework is proposed in the scope of hardware proof-carrying code (HPCC) and information flow checking [10] to decide whether malicious logic is inserted in the synthesis process by checking the consistency of security properties on RTL designs and their synthesized netlist. The proposed framework performs the trust evaluation procedure in three steps: 1) Proof-carrying based security properties verification on RT-level designs [10]; 2) Verification of the same set of security properties on the synthesized gate-level netlist; 3) Trust evaluation of EDA tools based on the security properties check results from the previous two steps. Through this approach, we can ensure that no malicious modifications have been added by the EDA tools to the synthesized netlist and finally make a decision about the underlying EDA tools trustworthiness The contribution of the paper includes:

- Distinct from previous efforts in RT-level security properties checking [8], [10], our gate-level information assurance scheme is developed to perform security properties checking on any synthesized netlist. As a result, almost all commercial level IP cores are covered by the proof-carrying hardware IP (PCHIP) protocol.
- An EDA tools trustworthiness evaluation framework is proposed which, for the first time, monitors the EDA tool behavior in order to ensure that no malicious modifications have been made to the RTL circuit.
- Using an AES module as the experimental vehicle along with the widely used Synopsys Design Compiler as the experimental platform, we successfully demonstrate the effectiveness of the proposed framework in evaluating the trustworthiness of EDA tools.

II. EDA Tools Trust Evaluation

EDA tools, especially commercial EDA tools, are used in almost all steps throughout the whole IC supply chain to help designers overcome new challenges in micro- and nano-meter designs. From the very early stages of the product specification to the end stage of PCB design, EDA tools affect every step of the IC supply chain. For example, the Xilinx ISE Core Generator can help designers to automatically generate IP cores for FPGA devices [11]; The Synopsys Design Compiler is the industrial standard for RTL circuit synthesis and netlist generation [12]; Cadence Encounter is one of the most popular layout tools both in the industry and in academia [13].

978-3-9815370-2-4/DATE14/©2014 EDAA

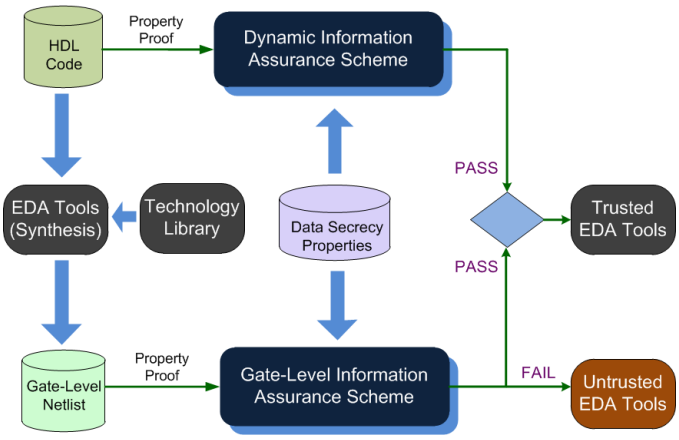

Fig. 1. Working procedure of the trustworthiness evaluation on EDA tools

However, ever since the emergence of EDA tools decades ago, improvements have been made only to their functionality and performance. Security consideration and evaluation has long been ignored. In fact, delayed security considerations have resulted in EDA tools becoming so complicated that many of the software security analysis methods are economically unfeasible to implement as EDA tool trustworthiness checks. Even worse, the assumption that EDA companies will not insert backdoors in their products weakens the motivation for EDA companies to run trust evaluation on their own products. Therefore, new methods should be proposed to ensure the trustworthiness of EDA tools to facilitate the ultimate goal of trusted integrated circuit designs by hardware security researchers.

To address this concern, we propose a new EDA tools trust evaluation method which is constructed on top of the proof-carrying hardware IP (PCHIP) framework [8], or more specifically, the dynamic information assurance scheme [10] and a newly proposed gate-level netlist information assurance scheme. Figure 1 shows the diagram of the proposed method as well as its working procedure. EDA tools users, especially design houses, will construct RT-level circuits in the form of HDL code and pass them through the proof-carrying based dynamic information assurance scheme to check whether the data secrecy properties hold for the RTL designs. If security theorems can be proven for HDL code, the RTL design will then be synthesized using EDA tools along with a technology library to generate a process specific gate-level netlist. The synthesized netlist will be validated by a gate-level information assurance scheme where the same set of data secrecy properties are applied. If the netlist still fulfills the security properties, which means the synthesis process relying on EDA tools does not alter the security properties of the underlying design, we can then claim that the used EDA tools are trusted given the data secrecy properties. Note that we assume that untrusted EDA tools will insert malicious modifications to all circuits under process because it will be difficult for attackers to turn on/off the malicious functionalities of untrusted EDA tools after the shipment of the EDA tools.

III. Gate-Level Information Assurance

The previously proposed dynamic information assurance in [10] provides a high-confidence IP protection option, but it falls short of trust evaluation on the gate-level netlist. To prevent unauthorized duplications and to protect the ownership of the IP cores, design houses prefer selling a synthesized netlist, or even hard cores, to providing RT-level designs. Thus, the IP consumers are unlikely to read through the gate-level netlist to manually check the design functionality because even a small-scale ASIC design with hundreds of lines of HDL code can end up with thousands of lines of gate descriptions. As a result, the PCHIP framework will be more attractive if it can also apply to gate-level netlist trust evaluation beyond its role in HDL code trust evaluation. Upon this request, we developed a new gate-level information assurance scheme which is similar to the IP transaction and proof preparation procedure as outlined by the dynamic information assurance scheme, but works for a synthesized netlist.

Figure 2 illustrates the preparation process of the trusted bundle defined by the gate-level information assurance scheme. According to the functional specification, HDL code will first be prepared by IP vendors and then be synthesized to netlist based on a specific technology library. A gate-level Coq formal logic is developed for code conversions from the synthesized netlist to a Coq netlist (note that the sensitivity transition model used in the gate-level scheme is the same as that in the dynamic information assurance scheme [10]). Relying on the gate-level Coq formal logic and the signal sensitivity transition model, the IP vendor will convert the netlist into a structural Coq netlist. The applied data secrecy properties, if presented in English, meaning “no internal sensitive information will be leaked through primary outputs of the target design”, will be formalized from English text into three kinds of theorems relating to the fix point sensitivity list: 1) Existence: There exists a fix point sensitivity list; 2) Accessibility: The fix point sensitivity list is achievable from a legitimate initial sensitivity list; 3) Trustworthiness: The circuit secrecy status defined by a fix point sensitivity list is trusted. All three theorems will be proven for the converted Coq netlist to validate the data secrecy properties in the delivered netlist.

Figure 3 shows the data secrecy property verification procedure performed by the IP consumer in the proposed scheme. Upon receiving the trust bundle (the trust bundle includes a synthesized netlist, security theorems and proof, and initial/fix point sensitivity lists) an IP consumer will first check the contents of the initial signal sensitivity list and the fix point signal sensitivity list, which represent the circuits initial secrecy status and its stabilized status, respectively. The validity of the initial list is checked to ensure that sensitivity levels are appropriately assigned to all input/output/internal signals. The circuit’s stable sensitivity status contains complete information of the distribution of sensitive information across the whole circuit, so the fix point list can be carefully evaluated to detect any Trojan channels that may leak sensitive information to primary outputs. Only if both sensitivity lists pass the

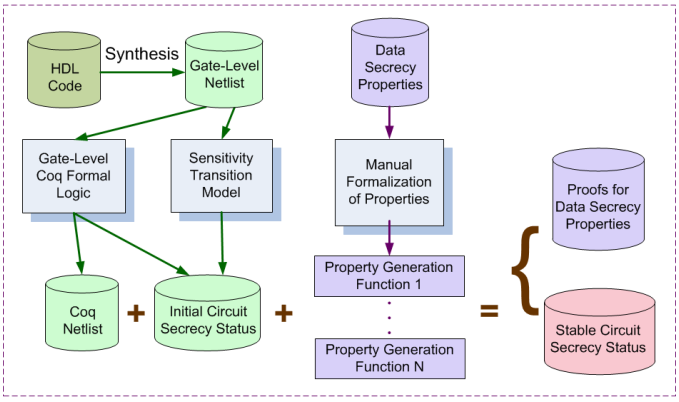

Fig. 2. Trusted bundle preparation in the gate-level information assurance scheme

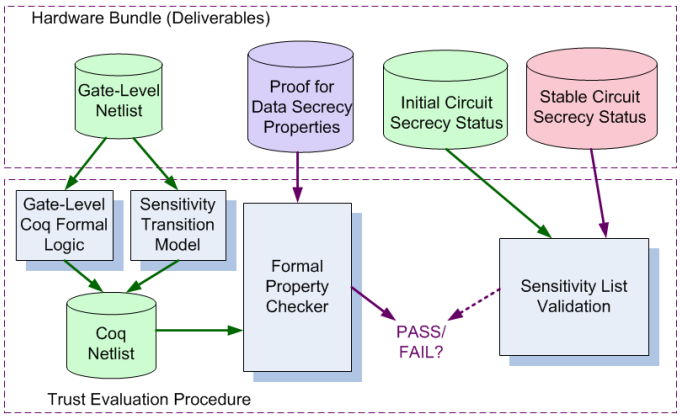

Fig. 3. Data secrecy property verification in the gate-level information assurance scheme

initial checking can IP consumers proceed to the next step of properties verification by an automatic property checker. A "PASS" signal provides evidence that the netlist does not contain any malicious leaking channels prohibited by the data secrecy properties. However, a "FAIL" result is a warning signal that some of the data secrecy properties are breached in the delivered IP netlist.

A. Gate-Level Coq Semantic Model

Although the newly proposed gate-level information assurance shares many steps in the dynamic information assurance scheme [10], the gate-level Coq formal logic needs to be redeveloped to support any gate-level netlist[1]. The new gate-level formal logic includes four main sections: 1) signals, 2) expressions, 3) expression evaluation, and 4) module definition of the converted Coq netlist.

1) Signals: All signals in the Coq netlist are assigned values which indicate their position in a centralized sensitivity list where all sensitivity information throughout the whole design is stored. The sensitivity levels, which are also assigned natural numbers, indicate the sensitivity levels of the underlying signals. A '0' means the underlying signal does not include any sensitive information, but a positive integer indicates the underlying signal carries sensitive information. The larger the number, the more sensitive the information is on the signal

2) Expressions: Since the gate-level netlist only contains combinatonal/sequential gates available in the technology library, all Coq netlist expressions are derived from the technology library with the exception of some special operations (e.g., `ECONS` converts a constant into a signal; `ESIG` treats a single signal as an expression; `ECOND_B` and `ECOND_EX` deal with scan flip-flops). Note that the same sensitivity downgrading expressions are derived from the dynamic information assurance scheme to make sure the same sensitivity transition model is applied to both the dynamic scheme and the gate-level scheme (See Figure 1). In the case of the AES encryption core, which we will use for our demonstration, the `EXOR2_key` and the `EXNOR2_key` are the only two expressions that can downgrade the signal sensitivities. These two sensitivity downgrading expressions can be invalidated if the XOR/XNOR logic is represented by complex gates of AND/OR/INV logic. In this case, gate pattern matching will be used but it is out of the scope of this paper.

3) Expression Evaluation: For expressions in the form of gate-level logic, we also developed the mechanism to decide the sensitivity of any expressions based on their inputs. This process is called expression evaluation and is defined formally for each of the expressions.

4) Module Definition: The module definition lists the input/output/internal signal types as well as the start/end indication of a module. Moreover, two signal assignments are available to explicitly express the different behavior of sequential gates (e.g., registers, latches) and combinational gates (e.g., AND2, XOR2).

B. Data Secrecy Property

For the gate-level information assurance scheme, the security properties we would like to prove for any netlist are the data secrecy properties, which if proven, can prevent any kind of sensitive data leakage through the primary outputs. Relying on the theorem generation functions proposed in [10], three theorems are generated addressing the three properties constituting the data secrecy properties. A sample of three generated theorems for the **existence**, the **accessibility** and the **trustworthiness** of the fix point list in an AES encryption core will be shown in the Demonstration Section.

IV. Demonstration

In order to demonstrate the effectiveness of the proposed EDA tools trust evaluation framework and the gate-level information assurance scheme, the Synopsys Design Compiler is selected as the target EDA tool and an AES encryption core is used as the sample circuit design. As showed in Figure 1, the EDA tools trust verification would be performed in three steps:

[1]We share the same sensitivity transition model as that in [10] so the rules to adjust the signal sensitivities in the gate-level framework are exactly the same as those in RTL model. More specifically, the same set of downgrading operations are used, e.g., the XORing with the round keys is still the only case for an Advanced Encryption Standard (AES) circuit as we will demonstrate shortly.

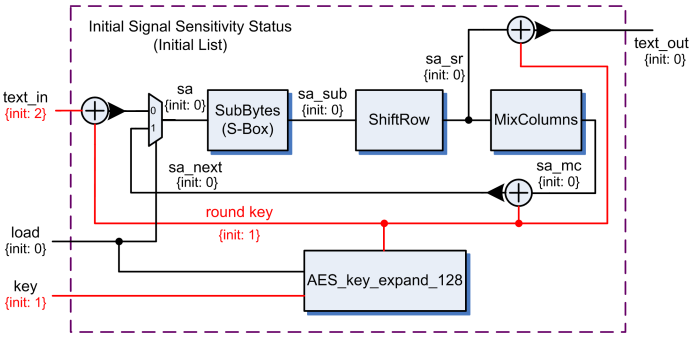

Fig. 4. AES Circuit Architecture and Initial Sensitivity Status [10]

1) the proof of data secrecy properties on the RT-level AES core; 2) the proof of data secrecy properties on the synthesized netlist; 3) trust evaluation of the targeted EDA tools.

A. RT-Level AES Core Security Analysis

The diagram of the AES encryption core is shown in Figure 4, where the top module only instantiates *AES_key_expand_128* to generate round keys. `SubBytes` (non-linear byte substitution), `ShiftRow` (row shifting), and `MixColumns` (column mixing), though shown in an abstract way, represent top-level logic rather than module instantiations. Details of the code conversion and initial/fix point sensitivity list generation can be found in [10] where 95 signals are defined in the RT-level AES circuit in total.

B. AES Netlist Security Analysis

Using the Synopsys Design Compiler, the AES RT-level code has been synthesized to a post-synthesis netlist. Although performing the same functionality, the synthesized netlist is of much larger size than the RT-level description. For example, there are 1964 signals defined in the synthesized netlist compared to the 95 signals defined in the HDL code.

The similarity between the gate instantiation in the synthesized netlist and expressions in the Coq netlist makes it possible to develop automation tools for code conversion. In fact, code auto-conversion and theorem auto-generation tools are developed using Perl scripting language. The introduction to the automation tools will be introduced in our later publications.

Finally, the formal theorems to prove data secrecy properties are of the same format as those in the dynamic scheme and have been successfully proven on the Coq netlist.

```
(* Stability *)
Lemma aes_sen_stable : update_sensitivity
        aes aes_stable_list = aes_stable_list.
(* Accessibility *)
Theorem fp_list_accessability : forall t : nat,
        t > 5 ->  (check_sensitivity t aes
            aes_initial_list) = aes_stable_list.
(* Trustworthiness *)
Theorem no_leaking_1 :
        nth done aes_stable_list 0 = 0.
......
Theorem no_leaking_N :
        nth text_out_127 aes_stable_list 0 = 0.
```

C. Design Compiler Security Analysis

The above demonstrates that the data secrecy properties have been proven for both the RT-level description and synthesized netlist and provide strong evidence that the synthesis process relying on Synopsys Design Compiler does not alter the trustworthiness of the AES circuit. Therefore, we can claim the trustworthiness of the used Synopsys Design Compiler under the data secrecy properties.

V. Conclusions

The security concerns of EDA tools have long been omitted because IC designers and integrators pay more attention to their functionality and performance than their trustworthiness. In this paper, a novel EDA tools trust evaluation framework has been proposed to ensure the trustworthiness of EDA tools by verifying its functional operations. The newly proposed framework can lower the evaluation cost and is a better fit for hardware security researchers. Meanwhile, a new gate-level information assurance scheme is developed for security property checking on any gate-level netlist in order to support the EDA tools evaluation framework. The proposed framework has been successfully applied on Synopsys Design Compiler using an AES encryption core as a sample design.

Acknowledgements

This work was partially supported by the National Science Foundation (NSF-1319105).

References

[1] D. Agrawal, S. Baktir, D. Karakoyunlu, P. Rohatgi, and B. Sunar, "Trojan detection using IC fingerprinting," in *IEEE Symposium on Security and Privacy*, 2007, pp. 296–310.
[2] Y. Jin and Y. Makris, "Hardware Trojan detection using path delay fingerprint," in *IEEE International Workshop on Hardware-Oriented Security and Trust*, 2008, pp. 51–57.
[3] R. M. Rad, X. Wang, M. Tehranipoor, and J. Plusquellic, "Power supply signal calibration techniques for improving detection resolution to hardware Trojans," in *IEEE/ACM International Conference on Computer-Aided Design*, 2008, pp. 632–639.
[4] Y. Jin and Y. Makris, "Hardware Trojans in wireless cryptographic ICs," *IEEE Design and Test of Computers*, vol. 27, pp. 26–35, 2010.
[5] M. Tehranipoor and F. Koushanfar, "A survey of hardware Trojan taxonomy and detection," *Design Test of Computers, IEEE*, vol. 27, pp. 10–25, 2010.
[6] M. Banga and M. Hsiao, "Trusted RTL: Trojan detection methodology in pre-silicon designs," in *IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, 2010, pp. 56–59.
[7] S. Drzevitzky and M. Platzner, "Achieving hardware security for reconfigurable systems on chip by a proof-carrying code approach," in *6th International Workshop on Reconfigurable Communication-centric Systems-on-Chip*, 2011, pp. 1–8.
[8] 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, 2012.
[9] Y. Jin and Y. Makris, "Proof carrying-based information flow tracking for data secrecy protection and hardware trust," in *IEEE 30th VLSI Test Symposium (VTS)*, 2012, pp. 252–257.
[10] Y. Jin, B. Yang, and Y. Makris, "Cycle-accurate information assurance by proof-carrying based signal sensitivity tracing," in *Hardware-Oriented Security and Trust (HOST), IEEE International Symposium on*, 2013, pp. 99–106.
[11] *http://www.xilinx.com/ise/products/coregen_overview.pdf.*
[12] Synopsys Inc., *Design Compiler User Guide*, Version X-2005.09.
[13] Cadence Design Systems, Inc., *Encounter User Guide*, Version 4.1.5.