Towards a comprehensive and systematic classification of hardware Trojans

J. Rajendran, E. Gavas, J. Jimenez, V. Padman and R. Karri
Polytechnic Institute of NYU, New York - USA

***Abstract*— Recently, there have been reports of Trojans being inserted at the hardware level. It is necessary to understand the characteristics of these hardware Trojans to be able to develop systematic methods to detect their presence. Absent this, only ad-hoc methods can be developed. Also, developed detection methods are not being adequately tested on a comprehensive set of benchmarks. We propose a comprehensive taxonomy of hardware Trojans based on five intuitive attributes. We organized the *embedded systems challenge (ESC)* to compile Trojans and analysed them to validate the taxonomy.**

I. Introduction

A hardware trojan is any malicious and deliberate change to an integrated circuit (IC) design that may cause unwanted effects. Consider the following example: This trojan consists of a single multiplexer. During normal operation, the encrypted data is sent as the output. When the trojan is active, the plain text bypasses the crypto-hardware and is sent as the output.

Fig. 1. Simple trojan

With the steady increase in outsourcing of semiconductor IC manufacturing, the concerns for this form of attacks are increasing within the military and the commercial sectors. Recently, Trojans in ICs used by military equipment have been reported [3]. Trojans can change the functionality of an IC and affect mission critical equipment. Trojans can also disable a system at will. These concerns caused the Defense Advanced Research Projects Agency (DARPA) to initiate the *Trust in ICs* program. This program focuses on developing trojan detection methods [1], [2].

To facilitate the development of trojan mitigation, detection and protection techniques, it is necessary to systematically categorize hardware Trojans and the benefits are threefold:

- classification of Trojans will enable a systematic study of their characteristics.
- trojan detection and mitigation techniques can be developed for each of the trojan classes.
- benchmarks targeting each of the classes can be developed.These benchmark techniques can serve as a basis to compare competing trojan detection methods.

This paper has two main sections. In Section 2, we describe the proposed taxonomy of Trojans. In Section 3 we validate the taxonomy based on Trojans from the embedded systems challenge.

II. Taxonomy of hardware Trojans

Hardware Trojans can be classified based on five attributes: 1) Phase in the IC design cycle effected, 2) Hardware abstraction level effected, 3) How the trojan is activated, 4) General effects, and 5) Physical location.

Fig. 2 shows the proposed taxonomy. In [16], the Trojans are classified based on their physical characteristics, their triggering mechanism and their functionality. The classes of Trojans considered in the previous taxonomy [16] are shown in red color. It can be seen that certain classes were not considered in that taxonomy.

Below we will explain our taxonomy using the simple trojan in Fig. 1 to explain the attributes considered.

A. At which phase of hardware design is a trojan inserted?

Consider the development cycle for an IC as shown below in Fig. 3.

Throughout the development cycle, individuals with access can modify the design at various phases.

1) In the *specification* phase, many characteristics are defined (e.g., target environment, expected function, size, power, delay). Within this phase, the specification can be altered to modify the functional or the design constraints [7] of a system. In the simple trojan, the original specification was altered to include into a bypass of the crypto-hardware.
2) *Designs* at the functional, logical and gate level are carried out for different components in the system. Designers may use third party intellectual property (IP) blocks and standard cells. Trojans can be inserted at any of these abstraction levels of the system. In the simple trojan, the could have been designed into the system.

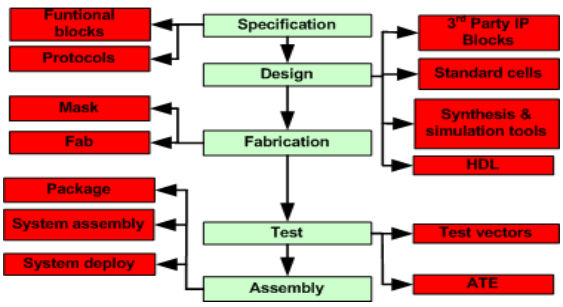

Fig. 3. Development cycle of an IC. Center boxes show the different phases. Outer boxes show possible vulnerabilities.

978-1-4244-5309-2/10/$26.00 ©2010 IEEE

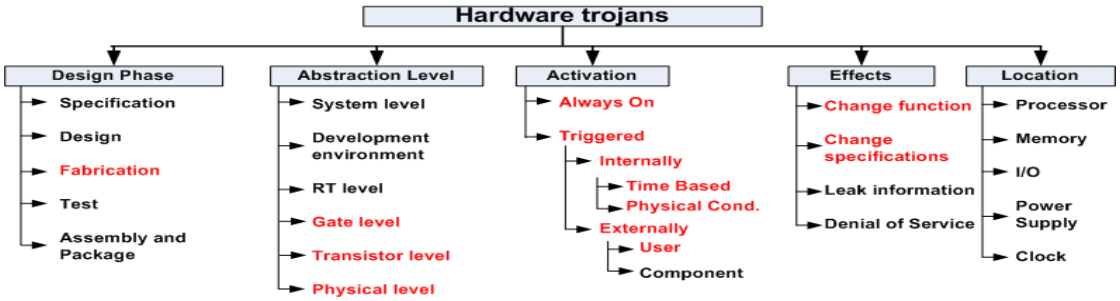

Fig. 2. Five attribute hardware Trojan taxonomy. The classes of Trojans considered in the previous taxonomy [16] are shown in red color (lightly shaded in case viewed in black and white).

3) *Fabrication* involves preparing masks and wafers preparation, oxidation, diffusion, ion implantation, chemical vapour deposition, metallization and photo lithography. The masks and wafer used are beyond the control of the designers, and can be a means of attacking by changing process parameters, geometries of the masks, etc. For example, chemical compositions may be altered to increase the electro-migration in critical circuitry like power supply and clock trees which would accelerate failures [3].
4) In the *testing* phase, test vectors are applied to the fabricated ICs by using automatic test equipment (ATE). Test vectors and the ATE can be constructed to mask the effect of Trojans. In our simple example, test vectors could be compromised to not detect the bypass logic.
5) The tested chip and other hardware components are *assembled* on a printed circuit board (PCB). Trojans may be inserted into the interfaces during packaging. The system may be assembled to allow the Trojans to be triggered[10]. In the example, the select line of the multiplexer may be triggered by an external component.

B. At what abstraction level is a trojan inserted?

Trojan circuits can be inserted at various hardware abstraction levels as shown in 4.

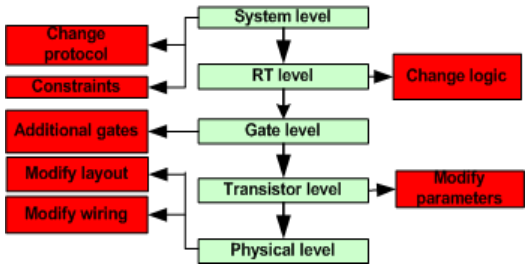

Fig. 4. Abstractions in a top down VLSI design flow. The forward arrow represents the synthesis. The red boxes on either side show the example Trojans in that level (darkly shaded in case viewed in black and white).

1) At the *system level* different hardware modules, interconnections and communication protocols used are defined. At this level, the Trojans may be triggered by the modules in the target hardware.
2) A typical *development environment* includes synthesis, simulation, verification and validation tools. The CAD tools and the scripts have been used to insert Trojans [15]. Software Trojans inserted in these CAD tools may mask the effects of the hardware Trojans.
3) At the *RT level* each functional module is described in terms of registers and signals. A trojan can be easily designed at the RT level as confirmed by the results to be discussed later.
4) At the *gate level* the design is represented as an interconnection of logic gates. This level allows an attacker to carefully control all aspects of the inserted trojan including size and location.
5) *Transistors* are used to build logic gates. This level gives the trojan designer control over circuit characteristics like power and timing. Individual transistors can be inserted or removed, altering the circuit *functionality* [16]. Transistor sizes can be modified to alter circuit *parameters* [16].
6) At the *layout level*, the dimensions and locations of all circuit components are described. This is the concrete level of the design where a trojan can be inserted. Trojans may be inserted by modifying the wire sizes, distances between circuit elements and re-assigning metal layers.

C. How is a trojan activated?

Some Trojans are designed to be *always on*, others may remain dormant until *triggered*. A triggered trojan needs an event - *internal* or *external*- to be activated. Once the trigger activates a trojan, it may remain active forever or return to a dormant state after some time.

1) *An internally triggered* trojan is activated by an event that occurs within the target device. The event may be either *time based* or *physical condition based*. Hardware counters in the design can trigger the trojan at a pre-determined time. They are also called time-bombs. Triggering circuitry may monitor physical parameters such as temperature and power consumption of the target device. When these parameters reach a pre-determined value, they trigger the trojan.
2) *An externally triggered* trojan requires external input into the target module to activate the trojan. The external triggered can be *a user input* or *a component output*. *User input* triggers may include push-buttons, switches, keyboards or keywords/phrases in the input data stream. *External component* triggers may be from any of the components that interact with the target device.

D. What are the effects of a trojan?

Trojans can also be characterized by their undesirable actions. The severity of their effects on the target hardware and/or system can range from the subtle disturbances to catastrophic system failures.

1) A trojan can *change function* of the target device and can cause subtle errors that may be difficult to detect.
2) A trojan can *change specifications* by intentionally changing device parameters. They may change the functional, interface or parametric specifications like power and delay.
3) A trojan can *leak sensitive information*. This can occur through both covert and overt channels. Information can be leaked by radio frequency, optical, thermal, power and timing side channels and also via interfaces like RS 232 and JTAG.
4) *Denial of service (DoS)* Trojans can prevent operation of a function or resource. A trojan may cause the target module to exhaust scarce resources like bandwidth, computation and battery power. Some Trojans may physically destroy, disable or alter the configuration of the device. In the example, the trojan denies the system of the crypto services. *DoS* may be either *temporary* or *permanent*.

E. Where are the Trojans located?

A trojan can be inserted in a single component or spread across multiple components. The Trojans can be located in the *processing units*, *memory*, *I/O*, *power grid* or *clock tree*. Trojans distributed across multiple components may act independently of each other or act as a group.

III. Validating the proposed taxonomy

We organized the embedded systems challenge (ESC) to compile a diverse set of Trojans and use them to validate the proposed taxonomy. In this section we will describe the challenge, analyse the Trojans, study the winning Trojans and discuss the limitations of this work.

1) The ESC, started with forty participants from universities worldwide. We short listed ten finalists based on a short report of their understanding of and approaches to designing Trojans. We provided each of the finalists a crytpo hardware platform, Alpha shown in Fig. 5, that implemented the advanced encryption standard (AES). Alpha uses a Digilent BASYS Spartan-3 FPGA board.

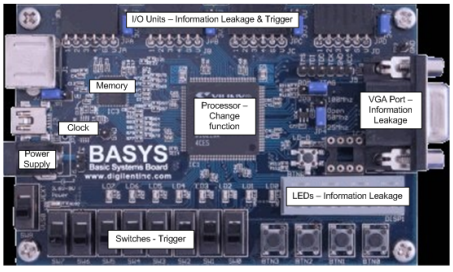

Fig. 5. The Alpha platform

The main processor interacts with Alpha through an RS 232 serial port. 256 shared secret keys are hard coded into Alpha. Leaking these secret keys or obtaining the unencrypted messages is the objective of an attacker. The user of the device selects a private key using the switches on Alpha. The encrypted data is sent through the RS-232 port. Alpha emulated a real world crypto accelerator, typically used to secure communications in a hostile environment. Alpha may be infected with Trojans by an attacker as described in this taxonomy. Each of the finalists played the role of an attacker and embedded Trojans into Alpha.

2) *Analysis:* The ESC Trojans are categorized based on the taxonomy and tabulated in Table I. Trojans were not reported for some of the classes like Trojans inserted in specification, fabrication, test and assembly phases. Hence these columns have been omitted from the table.
   - Most of the Trojans in the ESC were inserted in the design phase and at the abstraction level where the attacker had complete control of the target. Nearly 90% of the Trojans were inserted in design phase and at the RT level. Also, since Alpha was physically accessible, more than 50% of the Trojans were activated by user input and 75% of them located in the I/O units.
   - Trojans that are easier to trigger by user inputs, are inserted, when the attacker can physically interact with the target device. On the other hand, if the target devices are in a remote location, the inserted Trojans are most likely always on. Devices that are far away from the attacker are hard to trigger. So, he may make the trojan always on.
   - It is not always necessary to insert the trojan into the functional modules. Trojans can leak data through I/O interfaces like RS 232 as well. Sniffing using optical, thermal, and other such side channels may be much more difficult than through physical interfaces. In the ESC, many participants leaked the information using side channels that exist in the I/O interfaces.
   - The amount of power that Trojans consume and their area overhead are important metrics. Certain types of Trojans can consume more power when activated. Fig.6 summarizes the increase in power consumption of the hardware when the Trojans are active. Trojans that consumer more power are likely to be detected by power analysis based Trojan detection methods. On average, the number of flip flops and look-up-tables increased by 3% and 0.5%.
   - A trojan that changes the parameters of the circuit can be easily inserted in fabrication phase but it is hard to detect. Modifications to the specifications are typically easy to make but intuitively may also be easy to detect.
3) *Case studies:*
   - *The winning team* [5] inserted a trojan into the RS 232 interface module in the FPGA that can send the secret key at 15600 baud to the receiver. Since, the user is monitoring the channel for data at 9600 baud

TABLE I

CLASSIFICATION OF TROJANS SUBMITTED TO THE 2008 ESC. THERE ARE FIVE MAJOR COLUMNS CORRESPONDING TO THE FIVE ATTRIBUTES. THE MINOR COLUMNS REPRESENT SUB CATEGORIES.

Trojan	At what phase?		At what level		How is it activated?					What does it do?							Where it is located?			
	Spec.	Design	Dev. env.	RTL	Triggered				Always on	Leak Info.				Change of func.	DOS		Processor	I/O units	Power	Clock
					Internally		External input			RF	Optical	Thermal	Peripheral		Temp.	Perm.				
					Time	Phys. parameter	User	Extl. Comp.												
A1[5]		✓		✓					✓				✓				✓			
A2[5]		✓		✓					✓				✓				✓			
A3[5]		✓		✓	✓										✓			✓		
A4[5]		✓		✓					✓			✓						✓		
A5[5]		✓		✓					✓	✓							✓			
A6[5]		✓		✓					✓	✓							✓			
A7[5]		✓		✓					✓		✓							✓		
B1[7]		✓		✓			✓						✓					✓		
B2[7]		✓		✓		✓	✓						✓					✓		
B3[7]	✓	✓		✓				✓					✓					✓		
C1[14]		✓		✓			✓						✓					✓		
C2[14]		✓		✓			✓						✓					✓		
C3[14]		✓		✓			✓						✓					✓		
D1[10], [11]		✓		✓			✓						✓					✓		
D2[10], [11]		✓		✓			✓									✓			✓	
D3[10], [11]		✓		✓			✓							✓				✓		
D4[10], [11]		✓		✓			✓						✓				✓			
D5[10], [11]		✓		✓					✓				✓					✓		
D6[10], [11]		✓		✓		✓										✓	✓			
D7[10], [11]		✓		✓			✓						✓					✓		
D8[10], [11]		✓		✓			✓				✓							✓		
E1[9]		✓		✓			✓								✓		✓			
E2[9]		✓		✓					✓				✓					✓		
F1[8]		✓		✓			✓									✓				✓
F2[8]		✓		✓			✓									✓		✓		
F3[8]		✓		✓			✓							✓			✓			
F4[8]		✓		✓					✓				✓					✓		
G1[12]		✓		✓					✓		✓							✓		
G2[12]		✓		✓			✓									✓		✓		
G3[12]		✓		✓			✓						✓					✓		
H1[4]		✓	✓	✓					✓	✓								✓		
I1[6]		✓		✓					✓				✓					✓		
I2[6]		✓		✓			✓						✓					✓		
I3[6]		✓		✓			✓									✓		✓		
I4[6]		✓		✓					✓		✓							✓		
J1[13]		✓		✓			✓						✓			✓		✓		

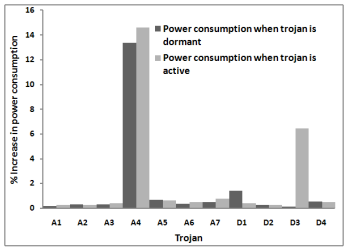

Fig. 6. The plot shows the % increase in power consumption when the inserted Trojans are activated

he cannot detect this covert transmission. This trojan is inserted during the specification phase described at the RT level and inserted into the I/O unit to leak sensitive information.

- *The runner up team* [10], [11] inserted a trojan, triggered by pressing F12 key, which can lock and ignore any plain text, unless the FPGA is reprogrammed. This trojan is inserted during the design phase, at RT level, is user input triggered, and denies the function located on the crypto-processing unit.

IV. CONCLUSIONS

The proposed taxonomy is quite comprehensive. However, the analysis in this paper is based on the Trojans submitted to the ESC. Because of constraints used in the challenge, the Trojans did not span the gamut of the taxonomy. The taxonomy proposed here can serve as a useful tool to study hardware Trojans and develop systematic trojan detection methods for the missing classes of Trojans.

V. ACKNOWLEDGEMENTS

This research was supported in part by NSF grants CNS 0831349 and CCF 0829824.

REFERENCES

[1] http://www.acq.osd.mil/dsb/reports/2005-02-HPMS_Report_Final.pdf.
[2] http://www.darpa.mil/mto/solicitations/baa07-24/index.html.
[3] The hunt for the kill switch. http://www.spectrum.ieee.org/semiconductors/design/the-hunt-for-the-kill-switch.
[4] D. Kouttron A. Tamoney and A. Radocea. Rpi team: number crunchers. http://isis.poly.edu/~vikram/rpi.pdf.
[5] A. Baumgarten, M. Clausman, B. Lindemann, M. Steffen, B. Trotter, and J. Zambreno. Embedded systems challenge. http://isis.poly.edu/~vikram/iowa_state.pdf.
[6] S. Cassidy, R. Ghilduta, D. Liu, and J. Szymaniak. Csaw 08 embedded systems challenge team tropicana. http://isis.poly.edu/~vikram/rit.pdf.
[7] Z. Chen, X. Guo, R. Nagesh, A. Reddy, M. Gora, and A. Maiti. Hardware trojan designs on basys fpga board. http://isis.poly.edu/~vikram/vt.pdf.
[8] C. Mitchell D. Stefan and C.G. Almenar. Trojan attacks for compromising cryptographic security in fpga encryption systems. http://isis.poly.edu/~vikram/cooper.pdf.
[9] M. Hicks. Breaking the code: hacking the alpha secure communications hardware platform. http://isis.poly.edu/~vikram/uofi.pdf.
[10] Y. Jin and N. Kupp. Csaw 2008 team report. http://isis.poly.edu/~vikram/yale.pdf.
[11] Y. Jin, N. Kupp, and Y. Makris. Experiences in hardware trojan design and implementation. In *Proc. IEEE Workshop on Hardware-Oriented Security and Trust*, pages 50–57, June 2009.
[12] J. Jimenez K. Gulurshivaram, R. Hailemichael and A. Raju. Implementation of hardware trojans. http://isis.poly.edu/~vikram/poly_team1.pdf.
[13] A. Kozak, A. Ivannikov, E. Poon, and I. Brutman. Csaw 2008 hardware challenge. http://isis.poly.edu/~vikram/poly_team2.pdf.
[14] A. Philip. Csaw 2008 hardware challenge: report of trojans. http://isis.poly.edu/~vikram/uofa.pdf.
[15] J. A. Roy, F. Koushanfar, and I. L. Markov. Extended abstract: Circuit cad tools as a security threat. In *Proc. IEEE Workshop on Hardware-Oriented Security and Trust*, pages 65–66, June 2008.
[16] Xiaoxiao Wang, M. Tehranipoor, and J. Plusquellic. Detecting malicious inclusions in secure hardware: Challenges and solutions. In *Hardware-Oriented Security and Trust, 2008. HOST 2008. IEEE International Workshop on*, pages 15–19, June 2008.