An Automated Web Tool for Hardware Trojan Classification

Nicholas Houghton, Samer Moein, Fayez Gebali, T. Aaron Gulliver
Department of Electrical and Computer Engineering
University of Victoria
P.O. Box 1700 STN CSC
Victoria, B.C. V8W 2Y2
Email: {nhoughto, samerm, fayez, agullive}@uvic.ca

***Abstract*—The complexity of modern integrated circuits has made them vulnerable to malicious insertions and modifications known as hardware trojans. A comprehensive trojan taxonomy was recently developed and incorporated into a technique for quantitatively classifying trojans. This can provide valuable insights into the origins and risk of a trojan based on the corresponding attributes, but the method can be difficult to use and prone to error. Thus, a tool has been developed to automate this classification. Widespread use of this tool will lead to a comprehensive database with detailed descriptions of all known trojans which can be used as a centralized reference for attackers and defenders. A discussion of the automated tool design is given including a case study to demonstrate its usefulness.**

I. Introduction

Several hardware trojan taxonomies have been proposed in the literature [1]–[4]. In [1], trojans were organized based solely on their activation mechanisms. A scheme based on the location, activation and action of a trojan was presented in [2], [3]. However, these approaches do not consider the manufacturing process. A taxonomy was proposed in [4] which employs five categories: insertion, abstraction, activation, effect and location. While this is more extensive than previous methods, it fails to account for the physical characteristics of the trojan. A comprehensive taxonomy was proposed in [5] which considers the manufacturing process in characterizing a hardware trojan.

The creation of an Integrated Circuit (IC) involves numerous phases known as the IC life-cycle. As shown in Fig. 1, these phases are **specification**, **design**, **fabrication and production**, and **testing and deployment** [5]–[9]. A malicious circuit can be inserted during any of these phases, or the IC circuitry can be modified. The phase where this is done determines some of the trojan attributes. The taxonomy in [5] describes the relationship between these attributes and the IC life-cycle. It was shown that the attributes of a trojan can be used to determine in which phases a trojan can be inserted, or conversely what trojan attributes are possible if it is inserted in a given phase. However, this requires an analysis of the trojan attributes which can be tedious and thus prone to error. To alleviate this problem, an on-line tool called the Hardware Trojan System (HTS) has been developed to automate this process. With this tool, users can quickly and easily select attributes for trojan analysis. The HTS automates the necessary computations, provides documentation, and compiles a database of known trojans.

The contributions of this paper are as follows.

1) A method of analyzing trojans based on a taxonomy is proposed.
2) A web-based tool which automates the analysis of hardware trojans is described.
3) A database to store hardware trojans input to the system is developed.

The remainder of this paper is organized as follows. Section II describes the hardware trojan taxonomy. Section III introduces the classification method and its software implementation, and the HTS is described in Section IV. Section V provides a case study to demonstrate use of the HTS. Finally, Section VI presents some concluding remarks.

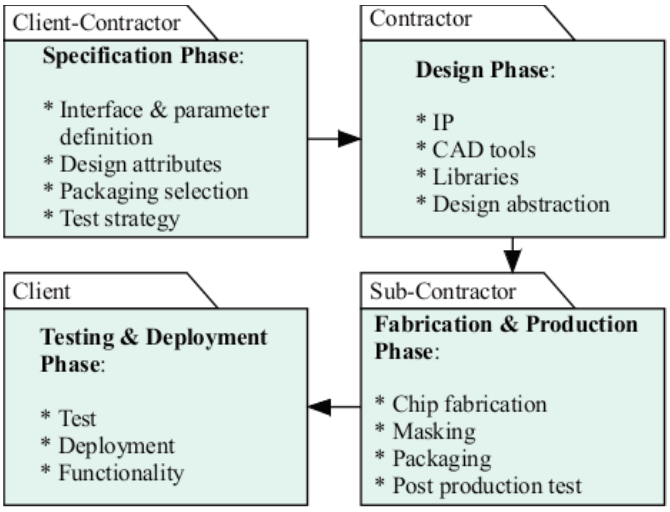

Fig. 1. The integrated circuit life-cycle [5].

II. Hardware Trojan Taxonomy

The taxonomy proposed in [5] is comprised of thirty-three attributes. This taxonomy is comprised of thirty-three attributes organized into eight categories as shown in Fig. 2. These categories can be arranged into four levels as in Fig. 3 and described below.

ISBN: 1-60132-433-2, CSREA Press ©

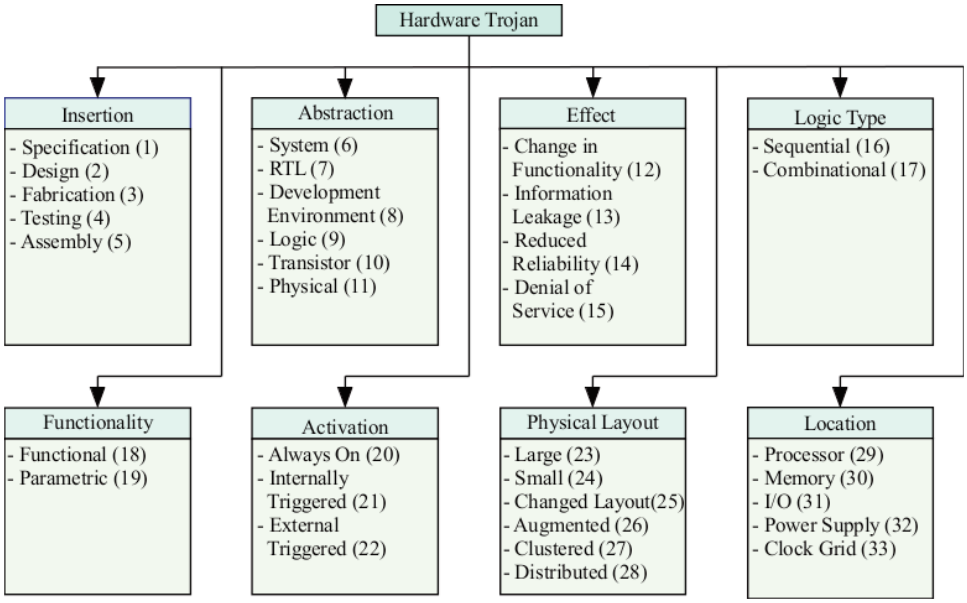

Fig. 2. The hardware trojan attribute taxonomy [5].

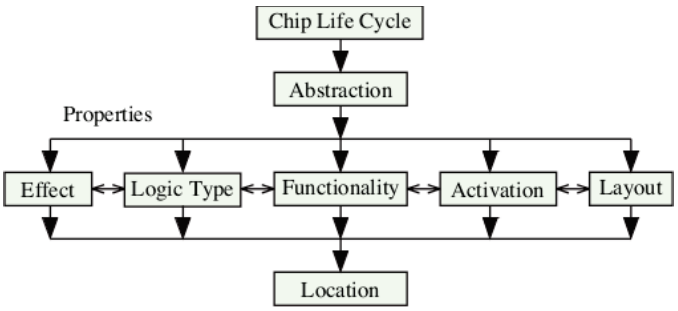

Fig. 3. The hardware trojan taxonomy levels [5].

1) The **insertion** (chip life-cycle) level comprises the attributes pertaining to the IC production stages.
2) The **abstraction** category/level corresponds to where in the IC abstraction the trojan is introduced.
3) The **properties** level comprises the behavior and physical characteristics of the trojan.
4) The **location** category/level corresponds to the location of the trojan in the IC.

The properties level consists of the following categories.

- The **effect** describes the disruption or effect a trojan has on the system.
- The **logic type** is the circuit logic that triggers the trojan, either combinational logic or sequential.
- The **functionality** differentiates between trojans which are functional or parametric.
- The **activation** differentiates between trojans which are always on or triggered.
- The **layout** is based on the physical characteristics of the trojan.

The relationships between the trojan attributes shown in Fig. 2 can described using a matrix $\mathbf{R}$ [5]. Entry $r(i,j)$ in $\mathbf{R}$ indicates whether or not attribute i can lead to attribute j. For example, $r(2,3) = 1$ indicates that design (attribute 2) can lead to fabrication (attribute 3). This implies that if an IC can be compromised during the design phase (attribute 2), it may influence the fabrication phase (attribute 3).

The matrix $\mathbf{R}$ is divided into sub matrices as follows

$$\mathbf{R} = \begin{bmatrix} \mathbf{R_1} & \mathbf{R_{12}} & 0 & 0 \\ 0 & \mathbf{R_2} & \mathbf{R_{23}} & 0 \\ 0 & 0 & \mathbf{R_3} & \mathbf{R_{34}} \\ 0 & 0 & 0 & \mathbf{R_4} \end{bmatrix}$$

where $\mathbf{R_1}$, $\mathbf{R_2}$, $\mathbf{R_3}$ and $\mathbf{R_4}$ indicate the attribute relationships within a category. For example, $\mathbf{R_1}$ is given by

$$\mathbf{R_1} = \left[\begin{array}{c|ccccc} A & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 0 & 1 & 0 & 0 & 0 \\ 2 & 0 & 0 & 1 & 0 & 0 \\ 3 & 0 & 0 & 0 & 1 & 0 \\ 4 & 0 & 0 & 0 & 0 & 1 \\ 5 & 0 & 0 & 0 & 0 & 0 \end{array}\right]$$

Submatrix $\mathbf{R_{12}}$ relates the attributes of the insertion category to the attributes of the abstraction category. An example of this submatrix is

$$\mathbf{R_{12}} = \left[\begin{array}{c|cccccc} A & 6 & 7 & 8 & 9 & 10 & 11 \\ \hline 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 1 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0 & 0 & 1 \\ 4 & 1 & 0 & 0 & 1 & 0 & 0 \\ 5 & 1 & 0 & 0 & 0 & 0 & 0 \end{array}\right]$$

When a trojan is discovered by a defender it can be assessed to obtain the properties according to Figs. 2 and 3. These

$\mathbf{R} =$

A	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33
1	0	1	0	0	0	1	0	0	0	0	0																						
2	0	0	1	0	0	0	1	0	0	0	0																						
3	0	0	0	1	0	0	0	0	0	0	1																						
4	0	0	0	0	1	1	0	0	1	0	0																						
5	0	0	0	0	0	1	0	0	0	0	0																						
6						0	1	0	0	0	0	1	1	0	1	0	0	1	1	1	0	0	0	0	0	0	0	0					
7						0	0	1	0	0	0	1	0	0	1	1	1	1	0	1	1	1	1	1	0	0	0	0					
8						0	0	0	1	0	0	1	0	0	1	1	1	1	0	1	1	1	1	1	1	1	1	1					
9						0	0	0	0	1	0	1	0	0	1	1	1	1	0	1	1	1	0	0	0	0	0	0					
10						0	0	0	0	0	1	1	0	1	0	0	1	1	1	1	0	0	0	1	0	1	1	0					
11						0	0	0	0	0	0	1	1	1	0	0	0	1	1	1	0	0	1	1	1	1	1	1					
12												0	0	0	0	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
13												0	0	0	0	0	1	0	1	1	0	1	0	1	0	1	1	1	1	1	1	1	1
14												0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	1	1	1	1	1	1
15												0	0	0	0	1	1	1	0	1	1	1	1	1	1	1	1	1	1	0	1	1	1
16												1	0	0	1	0	0	1	0	0	1	1	1	0	1	1	1	1	1	1	1	1	1
17												1	1	0	1	0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
18												1	0	0	1	1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
19												0	1	1	0	0	0	0	0	1	0	0	0	1	0	1	0	1	0	0	1	1	1
20												1	1	1	1	0	1	1	1	0	0	0	1	1	1	1	1	1	1	1	1	1	1
21												1	0	0	1	1	1	1	0	0	0	0	1	1	0	1	1	1	1	1	1	1	1
22												1	1	0	1	1	1	1	0	0	0	0	0	1	0	1	1	0	1	1	1	1	1
23												1	0	0	1	1	1	1	0	1	1	0	0	0	1	1	1	1	1	1	0	0	0
24												1	1	1	1	0	1	1	1	1	1	1	0	0	0	1	1	0	1	1	1	1	1
25												1	0	0	1	1	1	1	0	1	0	0	1	0	0	1	1	0	1	1	1	1	1
26												1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1
27												1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	0	0	1	1	1	1	1
28												1	1	1	1	1	1	1	1	1	1	0	1	0	0	1	0	0	1	1	1	1	1
29																																	
30																																	
31																																	
32																																	
33																																	

properties determine the physical location, abstraction level and insertion point based on $\mathbf{R}$. An attacker can perform a similar analysis. A detailed description of the procedures for an attacker or defender is provided in [10].

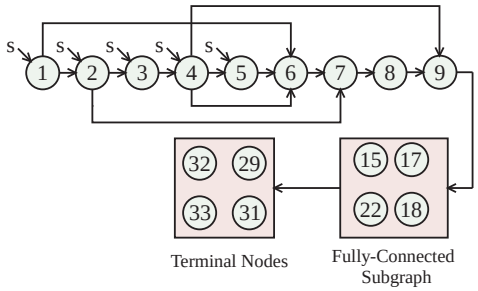

Fig. 4. The directed graph corresponding to a trojan.

III. Hardware Trojan Classification

The classification of a hardware trojan based on the attributes in the previous section can be used to visualize the trojan using a directed graph. The attributes are denoted by nodes and the relationships between them are represented by edges. For example, consider a trojan which has attributes 15, 17, 18 and 22 and is represented by the directed graph in Fig. 4. Submatrices $\mathbf{R_1}$ and $\mathbf{R_{12}}$ indicate that attribute 1 is connected to attributes 2 and 6. This relationship can be observed in the figure. Fig. 1 indicates that the effects of a trojan in a life-cycle phase can affect subsequent phases, and this is reflected by the possible insertion points denoted by S in Fig. 4. This representation of a trojan provides insight into the relationships between attributes. For example, consider an attacker who uses this method to generate the graph in Fig. 4. If they determine that it is not possible to insert the trojan in the design phase (attribute 2), then attribute 2 is removed from Fig. 4. Without access to the design phase (attribute 2), the trojan can only be inserted in the specification phase (attribute 1). Without the connection from attribute 1, attributes 3, 4 and 5 must also be removed. The trojan must access the abstraction level attributes directly via the system attribute (attribute 6).

A. The Classification Tool

The HTS is an easy to use tool which automates the hardware trojan classification process. To investigate a trojan, users first select attributes from the categories in Fig. 2 via an easy to use selection User-Interface (UI). Once the attributes are chosen, the tool performs the necessary analysis using $\mathbf{R}$ and displays the resulting directed graph.

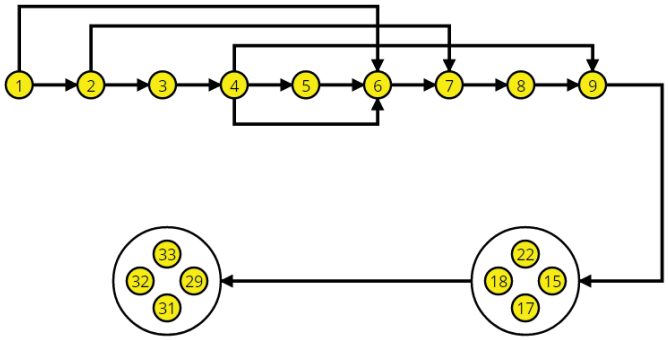

Fig. 5. The directed graph obtained by analyzing a hardware trojan.

Suppose an attacker decides to insert a trojan which has attributes 15, 17, 18 and 22. They can use the tool to obtain the directed graph shown in Fig. 5. If the design (attribute

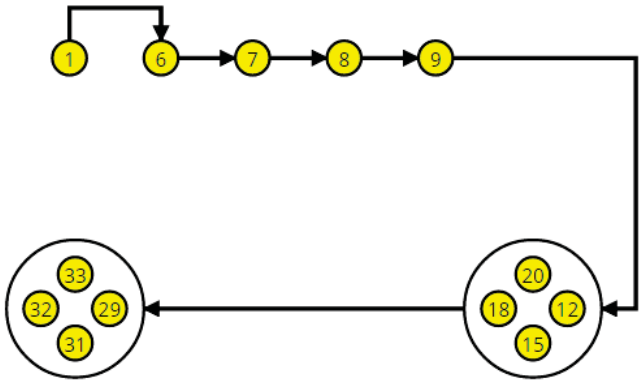

Fig. 6. The directed graph after attribute 2 is removed.

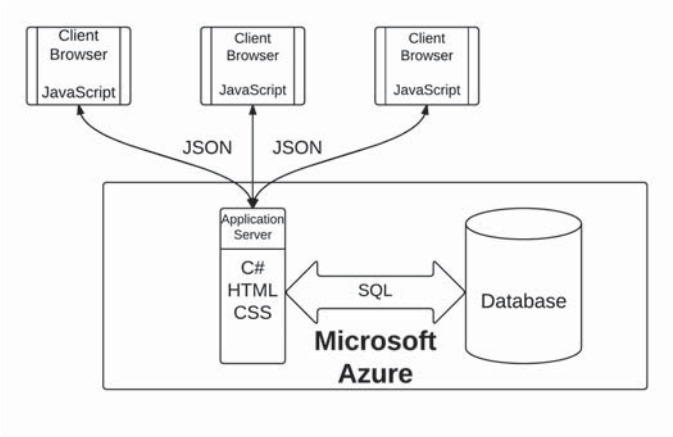

Fig. 7. Block diagram of the hardware trojan automated tool.

2) phase takes place in a secure location, the attacker will conclude that it is not possible to insert the trojan in this phase. To determine the possible trojans that can be inserted without access to the design phase, attribute 2 should be removed from Fig. 5. The classification tool provides an attribute removal feature. When an attribute is removed, the directed graph is recreated according to $\mathbf{R}$ and the new result displayed to the user. The result of removing attribute 2 from Fig. 5 is shown in Fig. 6. From the new graph, it is clear that the attacker can still compromise the system down to logic (attribute 9) in the abstraction category. The possible trojans locations remain the same, but the potential effects of the trojan have changed. Without access to the design phase (attribute 2) the trojan cannot be composed of combinational logic (attribute 17) or be externally triggered (attribute 22). Note that the attacker did not intend for the trojan to possess the attributes change in functionality (attribute 12) and always on (attribute 20), but the tool has determined that these attributes are possible.

IV. The Web Environment

The hardware trojan classification tool was designed as a web utility for portability and easy distribution. The application server performs all of the computations and generates page markup to minimize the burden on client-side browsers. It communicates directly with a remote database used to store user account information and application data (attributes, categories and matrices). Both the application server and the database are hosted on the Microsoft Azure Cloud platform [14]. The cloud improves reliability, portability, and flexibility, provides 'on-demand' resources that are automatically managed for scalability requirements, and provides the ability for maintenance to take place anywhere. Fig. 7 gives a block diagram of the HTS. The technologies employed are as follows.

- **ASP.NET Web Form:** A user interface focused, event-driven model of the .NET framework. It allows powerful data-binding, separation of server-client side activities, a native security structure, and increased client performance [11].
- **Entity Framework:** An object-relational database mapper designed for the .NET framework. It provides a library of high speed SQL statements wrapped in C# commands to simplify development and ensure performance [12].
- **D3.js:** A Java-script library for visualizing data with HTML, SVG and CSS [13].
- **Azure:** The Microsoft cloud system [14].

Fig. 8 provides an overview of the structure of the trojan system website. The *home, contact*, *about*, and *application information* pages are accessible to all traffic. The application information page contains three sub-pages providing information on each of the primary applications. Users are required to create an account and be logged in to access the remainder of the site. Email confirmation is used to verify user accounts.

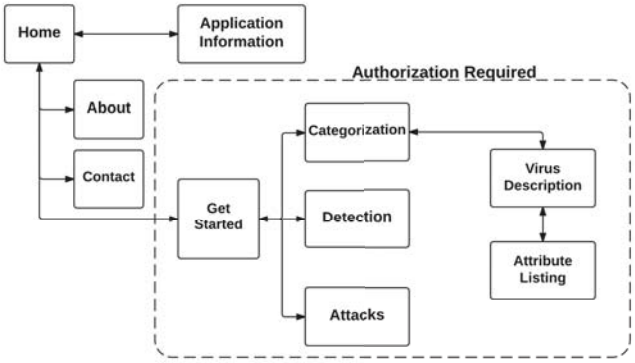

Fig. 8. An overview of the website architecture.

A. Database

The attributes, categories, trojan descriptions and user information are all stored in a database on the *Microsoft Azure Cloud System*. The database was designed using a model first approach with the Entity Framework. The matrix $\mathbf{R}$ discussed in Section III is stored in a relational table. Each element of the matrix is stored as an entry in the table. Each table entry

contains the row number, column number and value from the matrix. The application server uses this table to perform the trojan analysis. This approach allows for the use of the highly streamlined and efficient querying power of SQL. In addition, any modifications to the matrix values or attribute definitions can be done centrally with simple Create, Read, Update and Delete (CRUD) operations. All user results are stored in the database for easy data mining capabilities. This will allow for search and statistical operations in the future.

V. Case Study

Consider an attacker working as a design engineer in a IC manufacturing facility who inserts the circuitry shown in Fig. 9. The original output of the circuit was along wire C, but the addition of the trigger and payload results in $C_{modified}$. The output $C_{modified}$ is the same as C except when the inputs A and B are both zero ($A = B = 0$). This is an example of a combinational logic triggered trojan. If $A = B = 0$ is a relatively rare event, it is unlikely to be discovered during testing. A defender who discovers this trojan can extract the attributes: change in functionality, combinational, functional, internally triggered, small, clustered, and augmented. Thus, attributes 12, 17, 18, 21, 24, 26, and 27 are the input to the classification method.

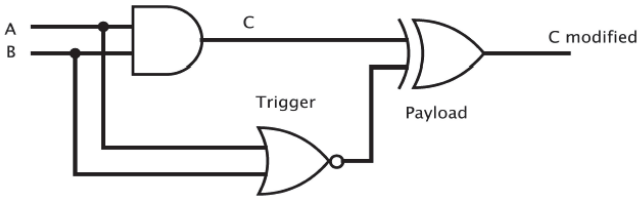

Fig. 9. A combinational trojan [10].

A. Hand Analysis

The observed trojan attributes are all from the properties category given in submatrix $\mathbf{R_3}$. Examining the corresponding columns in $\mathbf{R_{23}}$, these attributes are directly connected to the development environment in the abstraction category (attribute 8). From $\mathbf{R_{12}}$, there is no direct connection between this attribute and an attribute in the insertion category (attributes 1 to 5). However, $\mathbf{R_2}$ provides a connection between the development environment (attribute 8) and RTL (attribute 7). Then attribute 7 is connected to design (attribute 2) and system (attribute 6). Attribute 6 is connected to specification, testing, and assembly (attributes 1, 4 and 5).

The observed properties of the trojan have now been used to analytically determine that the insertion of the trojan must have taken place during either specification (attribute 1), testing (attribute 4) or assembly (attribute 5). Further, the likely physical location of the trojan in the system can be determined. The observed property attributes (12, 17, 18, 21, 24, 26, and 27) are scanned along the rows of submatrix $\mathbf{R_3}$ and into the rows of submatrix $\mathbf{R_{34}}$. There it can be clearly seen that this trojan leads to all location attributes (columns 29, 30, 31, 32, and 33). The final directed graph is shown in Fig. 10.

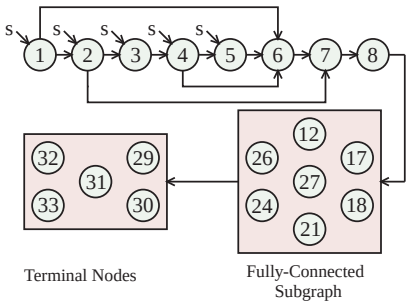

Fig. 10. The directed graph corresponding to the trojan in Fig. 9.

Fig. 11. The selection of property attributes.

B. Classification Application

The HTS classification application avoids analysis errors by automatically producing the directed graph for the hardware trojan. Fig. 11 shows the simple UI that users employ to select attributes. Once the attributes have been selected, users are directed to the description page. The description page lists the attributes along with any relevant information as shown in Fig. 12. Users can remove attributes as required to adjust the trojan characteristics. The user can then obtain the corresponding directed graph. The analysis of the combinational trojan in Fig. 9 provides the directed graph shown in Fig. 13.

If a defender knows that a particular insertion point is secure, the tool provides a simple means of removing the corresponding attribute and recreating the matrix and directed graph. For example, suppose the attacker is aware that the testing phase (attribute 4) is not accessible. The removal of this attribute results in the revised directed graph shown in Fig. 14, which limits the insertion of the combinational trojan in Fig. 9 to the specification or design.

VI. Conclusion

The relatively new field of hardware security requires a systematic means of investigating hardware trojans. In this paper, a comprehensive taxonomy is used in the development of an on-line suite of tools called the Hardware Trojan System (HTS). An HTS application was presented which automates

ID	Name	Category	F_in	F_out	Select Attribute	On/Off	Remove Item
21	Internally Triggered	Properties	13	15	☐	Off	☐
12	Change In Functionality	Properties	18	17	☐	Off	☐
27	Clustered	Properties	17	19	☐	Off	☐
24	Small	Properties	16	17	☐	Off	☐
18	Functional	Properties	19	18	☐	Off	☐
17	Combinational	Properties	17	18	☐	Off	☐
26	Augmented	Properties	19	21	☐	Off	☐

Fig. 12. The description page.

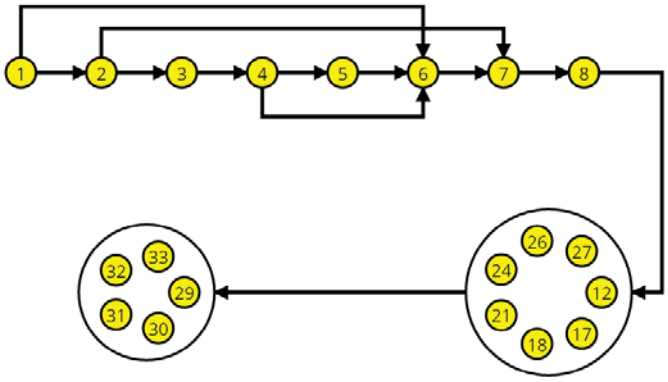

Fig. 13. The trojan directed graph obtained via the classification application.

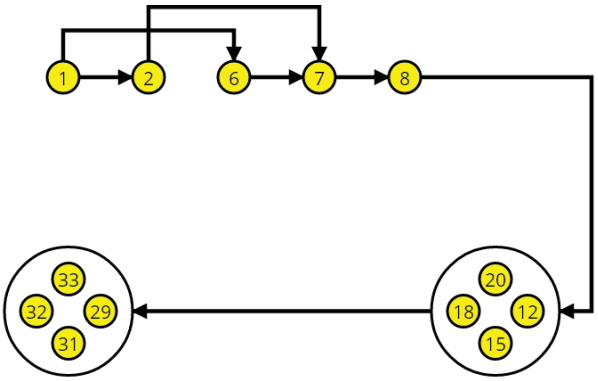

Fig. 14. Trojan directed graph after the removal of attribute 4 from Fig. 13.

the classification of trojans. This tool uses the observed attributes to automatically provide a quantitative assessment of hardware trojans. The application also compiles a trojan database.

It is left for future work to implement data mining techniques and a User Interface (UI) which allows users to browse existing trojans in the database. The addition of this functionality will provide a centralized means of evaluating the current state of hardware security.

REFERENCES

[1] F. Wolff, C. Papachristou, S. Bhunia, and R. S. Chakraborty, "Towards trojan-free trusted ICs: Problem analysis and detection scheme," in *Proc. Conf. on Design, Automation and Test in Europe*, Munich, Germany, Mar. 2008, pp. 1362–1365.

[2] R. M. Rad, X. Wang, M. Tehranipoor, and J. Plusquellic, "Power supply signal calibration techniques for improving detection resolution to hardware trojans," in *Proc. IEEE/ACM Int. Conf. on Computer-Aided Design*, San Jose, CA, Nov. 2008, pp. 632–639.

[3] R. Karri, J. Rajendran, K. Rosenfeld, and M. Tehranipoor, "Trustworthy hardware: Identifying and classifying hardware trojans," *IEEE Computer*, vol. 43, no. 10, pp. 39–46, Oct. 2010.

[4] X. Wang, M. Tehranipoor, and J. Plusquellic, "Detecting malicious inclusions in secure hardware: Challenges and solutions," in *Proc. IEEE Int. Workshop on Hardware-Oriented Security and Trust*, Anaheim, CA, Jun. 2008, pp. 15–19.

[5] S. Moein, S. Khan, T. A. Gulliver, F. Gebali, and M. W. El-Kharashi, "An attribute based classification of hardware trojans," in *Proc. Int. Conf. on Computer Eng. and Sys.*, Cairo, Egypt, Dec. 2015, pp. 351–356.

[6] T. Zhou and T. B. Tarim, "An efficient and well-controlled IC system development flow: Design approved specification and design guided test plan," in *Proc. IEEE Int. Symp. on Circuits and Systems*, Kobe, Japan, May 2005, pp. 2775–2778.

[7] B. Sharkey. (2007) TRUST in integrated circuit program - briefing to industry. [online]. Available: http://cryptocomb.org/DARPA - Trust in Integrated Circuits Program.pdf Accessed: Nov. 11, 2015.

[8] S. Padmanabhan. (2013) Discover a better way to go from C-level to synthesis for SoC designs. [online]. Available: http://electronicdesign.com/technologies/discover-better-way-go-c-level-synthesis-soc-designs Accessed: Nov. 11, 2015.

[9] M. Tehranipoor and F. Koushanfar, "A survey of hardware trojan taxonomy and detection," *IEEE Design and Test of Computers*, vol. 27, no. 1, pp. 10–25, Jan.-Feb. 2010.

[10] S. Moein, "Systematic Analysis and Methodologies for Hardware Security," *PhD Dissertation*, University of Victoria, Victoria, BC, Canada, 2015.

[11] Microsoft (2002) ASP.NET 4.5 [Computer Software] Available: http://www.asp.net/ Accessed: May 7, 2015.

[12] Microsoft (2008) Entity Framework v6.0 [Computer Software] Available: https://msdn.microsoft.com/en-ca/data/ef.aspx Accessed: May 7, 2015.

[13] Microsoft (2011) D3.js v3.5.6 [Computer Software] Available: https://d3js.org/ Accessed: May 7, 2015.

[14] Microsoft (2010) Azure Cloud System [Computer Software] Available: azure.microsoft.com Accessed: May 7, 2015.