Possible Organization for Writing a Thesis including a LaTeXFramework and Examples

by

A Graduate Advisor
B.Sc., University of WhoKnowsWhere, 2053
M.Sc., University of AnotherOne, 2054

A Dissertation Submitted in Partial Fulfillment of the
Requirements for the Degree of

DOCTOR OF PHILOSOPHY

in the Department of Whichever

Possible Organization for Writing a Thesis including a LaTeXFramework and Examples

by

A Graduate Advisor
B.Sc., University of WhoKnowsWhere, 2053
M.Sc., University of AnotherOne, 2054

Supervisory Committee

Dr. R. Supervisor Main, Supervisor
(Department of Same As Candidate)

Dr. M. Member One, Departmental Member
(Department of Same As Candidate)

Dr. Member Two, Departmental Member
(Department of Same As Candidate)

Dr. Outside Member, Outside Member
(Department of Not Same As Candidate)

Supervisory Committee

Dr. R. Supervisor Main, Supervisor
(Department of Same As Candidate)

Dr. M. Member One, Departmental Member
(Department of Same As Candidate)

Dr. Member Two, Departmental Member
(Department of Same As Candidate)

Dr. Outside Member, Outside Member
(Department of Not Same As Candidate)

ABSTRACT

This document is a possible Latex framework for a thesis or dissertation at UVic. It should work in the Windows, Mac and Unix environments. The content is based on the experience of one supervisor and graduate advisor. It explains the organization that can help write a thesis, especially in a scientific environment where the research contains experimental results as well. There is no claim that this is the *best* or *only* way to structure such a document. Yet in the majority of cases it serves extremely well as a sound basis which can be customized according to the requirements of the members of the supervisory committee and the topic of research. Additionally some examples on using LaTeXare included as a bonus for beginners.

Contents

List of Tables

List of Figures

ACKNOWLEDGEMENTS

I would like to thank:

my cat, Star Trek, and the weather, for supporting me in the low moments.

Supervisor Main, for mentoring, support, encouragement, and patience.

Grant Organization Name, for funding me with a Scholarship.

I believe I know the only cure, which is to make one's centre of life inside of one's self, not selfishly or excludingly, but with a kind of unassailable serenity-to decorate one's inner house so richly that one is content there, glad to welcome any one who wants to come and stay, but happy all the same in the hours when one is inevitably alone.

Edith Wharton

DEDICATION

Just hoping this is useful!

Chapter 1

Introduction

1.1 Overview

1.2 Motivation

1.3 Contributions

1.4 Thesis Organization

Chapter 2

Hardware Security

2.1 Overview

2.2 Viruses

2.3 Detection Methods

Chapter 3

More Research

Chapter 4

Quantitative Analysis

4.1 Introduction

Samer Moein originally proposed a series of techniques for the express purpose of quantifying the hardware security science in his Ph.D dissertation *Systematic Analysis and Methodologies for Hardware Security* [5]. This chapter will provide an overview of these techniques which are the basis for the *Hardware Trojan System*.

4.2 Classification

To properly understand the wide-ranging forms and origins of trojans a classification system is used that employs their key aspects [5], [9]. A list of thirty-three attributes is proposed that quantify the characteristics of hardware trojans. Each of the attributes are taxonomically grouped into one of eight classes as outlined in Figure 4.1. These taxonomic classes are then further organized into the four main categories as shown in Figure 4.2 that are used by the classification tool discussed later in section 5.3.1. These four primary categories are:

1. The **Insertion** category (also know as the *Chip Life Cycle* category) comprises attributes pertaining to the stages of production of an IC that are vulnerable to malicious attack.

2. The **Abstraction** category looks at the abstraction level of the IC in which a trojan is introduced.

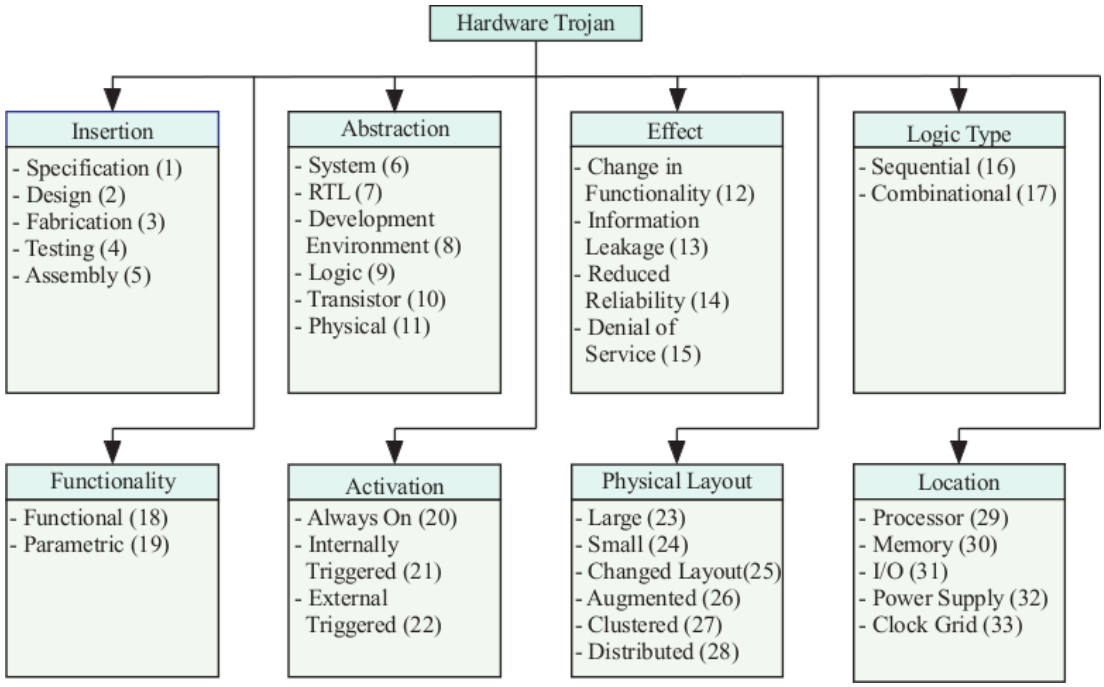

Figure 4.1: The Taxonomy of Attributes and Subcategories

3. The **Properties** section is comprised of qualities all pertaining to the observed behaviors and physical construction of trojans. This main category is the only one of the four that contains more than one of the taxonomic classes: *effect*, *logic type*, *functionality*, *abstraction* and *layout*.

4. The **Location** category looks at the physical place in the integrated circuit where the trojan lies.

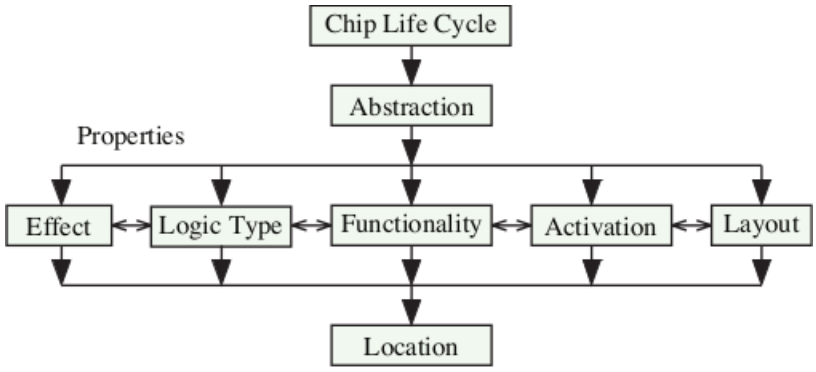

Figure 4.2: Hardware Trojan Levels of the Four Main Categories

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

Figure 4.3: Matrix R. Sub matrices Composition

When a trojan is discovered it is analyzed for the possession of the attributes from the *Properties* category. An nxn relation matrix, referred to as R, is used to relate each of the thirty-three attributes to each and every other attribute. Due to its size the matrix R is not included in this paper however it is summarized in Figure 4.3; refer to [9] for the whole matrix R. Each entry in the matrix, denoted $r(i, j)$, represents whether attribute i can lead to attribute j. For example, referring to Figure 4.4, an entry $r(2, 3) = 1$ indicates that attribute 2 *(Design)* can lead to attribute 3 *(Fabrication)*. Colloquially this means that if a malicious member of the design team is capable of compromising the integrated circuit during the design phase (insertion point 2) the damage can propagate to the fabrication phase (insertion point 3).

This relationship between attribute two and three can be described as *"inter-category"* as both the source and target attributes belong to the same category, *insertion*. Non *"inter-category"* relationships relate attributes of different categories. For example, from the *Abstraction* category, a virus that is introduced in the *Register Transfer Level* (attribute 7) can not directly cause a trojan that exhibits the *Reduced Reliability* (attribute 14) property because the relation $r(7, 14) = 0$. The same virus is however capable of causing a *Denial of Service* because the relation $r(7, 15) = 1$.

The matrix R is divided into sub matrices as shown in Figure 4.3. Sub matrices R_1, R_2, R_3 and R_4 contain the *"inter-category"* relations; sub matrix R_1 is show in detail in Figure 4.4. Notice how the same attributes (1-5) are represented in both the rows and columns.

Non *"inter-category"* sub matrices provide the relations of attributes from one category to another. Figure 4.5 shows an example of a non *"inter-category"* sub matrix. R_{12} relates the insertion point attributes of matrix R_1 to the abstraction level attributes of R_2. Suppose it is found that a trojan pertains only to the abstraction *"Register Transfer Level"* (attribute number 7). Inspection of column 7 in Figure 4.5 determines that the only possible insertion point is attribute number 2 (Design). This

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

Figure 4.4: Sub Matrix R1, *"inter-category"* Example

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

Figure 4.5: Sub Matrix R12, non *"inter-category"* Example

tells us that the only part of the production life cycle where a malicious person could have inserted this virus was during the design phase. This method has drastically reduced the search time for the guilty party.

Matrix R allows us to visualize the nature of the trojan via a directed graph. Figure 4.6 shows an example visualization of a virus that was discovered to have properties 12, 17, 18, 21, 24, 26 and 27. For a more detailed explanation of how each of the attributes relate and how the matrix R is used to analyze a trojan please refer to [5] and [9].

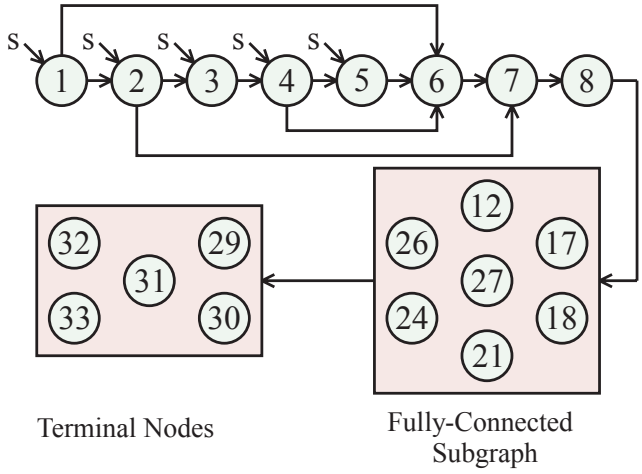

Figure 4.6: Directed Graph of Example Virus

4.3 Detection

The lack of consistent structure across the spectrum of processor design results in a massive diversity in regards to the structure, behavior and insertion points of trojans and as of yet there is no single method of detection capable of handling all possibilities. Specific trojan characteristics requires equally specific means of detection. The natural segregation of each individual trojan detection pair makes it extremely difficult to compare and contrast competing techniques. To combat such ambiguity a ranking system has been developed that will not only aid in the selection and definition of detection methods but will aid in the selection of techniques based on observed characteristics of trojans.

To construct a means of comparison between a trojan and a detection scheme each are given a numerical value representing the level of applicability in each of the eight classes outlined in Figure 4.1. When a trojan virus is analyzed using the classification method described in section 4.2 the selected attributes are used to create a series of values to build a vector known as the trojan's *Severity* ranking. Each class contains multiple attributes, the combination of the attributes present in a trojan are the determining factor for the assigned value. Figure 4.7 shows the ranking table for the *effect* class. Every class is assigned two values, one to represent the unique combination (I_E in this class) and a second to represent the severity level (C_E).

k	Attribute	I_E	C_E
12	Change in Functionality	1	2
13	Information Leakage	2	4
14	Reduce Reliability	3	1
15	Denial of Service	4	2
12 & 13		5	6
12 & 14		6	3
12 & 15		7	4
13 & 14		8	5
13 & 15		9	6
14 & 15		A	3
12 & 13 & 14		B	7
12 & 13 & 15		C	8
12 & 14 & 15		D	5
13 & 14 & 15		E	7
12 &13 & 14 & 15		F	9

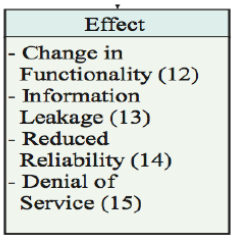

Figure 4.7: The Ranking Values of The Effect Class

Table 4.1 provides an example of the use of the severity rating of two hardware trojans. With use of this ranking system two completely different viruses can be quantitatively compared and contrasted for efficacy in each class. When a new detection method is produced the same ranking tables are used to produce a similar vector where the only difference is that the severity value is referred to as the method's *Coverage Level*. Coverage is used to determine the methods ability to detect a trojan of a common class. Table 4.2 shows the coverage vectors of two detection methods. By comparing these methods with the trojans in Table 4.1 a user can quickly and accurately determine the applicability of a method for a particular trojan. By comparing *Trojan A* with detection method 6 it can be seen that the insertion point ranking value (I_R) is higher in the detection method than it is in the trojan. This implies that method 6 would be an appropriate choice for detecting *Trojan A* in regards to where it is inserted. For a complete description of the ranking tables and more thorough discussion of their application refer to [8].

Table 4.1: Severity Rating Vectors of two Trojans

Techniques	Parameters (I_P)								Severity (C_P)							
	I_R	I_A	I_E	I_L	I_F	I_C	I_P	I_O	C_R	C_A	C_E	C_L	C_F	C_C	C_P	C_O
Trojan A	2	6	2	1	2	1	7	7	2	6	4	1	2	1	5	2
Trojan B	3	3	1	2	1	2	8	1	3	3	2	2	1	3	6	1

Table 4.2: Hardware Trojan Detection Vector

Techniques	Parameters (I_P)									Coverage (C_P)								
	I_R	I_A	I_E	I_L	I_F	I_C	I_P	I_O	I_G	C_R	C_A	C_E	C_L	C_F	C_C	C_P	C_O	C_G
[2]	3	3	B	1	2	4	7	V	1	3	3	7	1	2	3	5	5	2
[7]	3	3	1	2	1	4	7	V	4	3	3	2	3	1	3	5	5	3

Table 4.1 gives a side by side comparison of the severity rating vectors for two trojan viruses. Trojan A has a lower coverage rating value than Trojan B in the *Insertion* subcategory denoted by C_R. This signifies that Trojan B could be inserted in more stages of the manufacture process than Trojan A making it a more dangerous virus. Table 4.2 gives a similar comparison between two competing detection methods. Method 6 has a higher coverage rating in the *effect* subcategory than method 16 signifying there is a larger number of virus effects this method is able to detect. By

referring to Table 4.2 and Figure 4.7 it can be devised that method 6 is capable of detecting attributes 12, 13 and 14 where as method 16 is only capable of detecting attribute 12. Further more it is possible to compare a virus and a detection method in a similar manner. In this scenario the special case of the virus and method having an equal coverage/severity rating the detection method's coverage rating value takes precedence over the virus's severity. Colloquially this implies that a coverage rating is capable of detecting a severity rating of an equal value.

4.4 Attacks

Chapter 5

Discussion, Platform and Results

5.1 Introduction

In this chapter we present a full description of the automated tool. The provision of a machine which executes the methodologies previously described concretes their definition and propels them towards acceptance as standard practice. Interested parties can use this tool as a source of reference when describing their own investigations. The HTS further allows the collection and centralization of data. The avid use of this tool in the hardware security science community will naturally create a catalog of information from which statistical analysis on the field can be performed.

5.2 Web Environment

The HTS is comprised of three applications which perform the analysis of the quantitative methods described in the previous chapter in sections 4.2, 4.3 and 4.4. The applications are nested in a web site that is easily accessible worldwide, provides documentation and instructions for users as well as an easy to navigate user interface. The system operates primarily on an application server which performs all of the computation and generates page mark-up to minimize overburdening client-side browsers with processing responsibility. The server communicates directly with a remote database used to store user login and account information, application data (attributes, categories and the computation matrices) as well as any work performed by the user both complete or incomplete. Both the application server and the database are hosted on *Microsoft's Azure Cloud* platform. The cloud platform improves reliability, portabil-

ity and flexibility, provides 'on-demand' resources that are automatically managed for scalability requirements and provides the ability for maintenance to take place anywhere.

5.2.1 Technologies Used

In this section we provide a review of the tools and technologies used to implement the HTS.

ASP.NET Web Form Framework

The web application is built using *Microsoft's* ASP.NET web application framework [3] which provides multiple programming models for advanced web application development. The trojan classification tool employs .NET's *Web Forms* model. The *Web Forms* model provides pages that are requested by a client-side browser. The server uses what is referred to as *"behind the page"* code to dynamically compile application logic and generate HTML markup which is sent to the client. The use of this technology provides automated compatibility with any browser, powerful C# application logic programming capability, intuitive inter-end state management and easy scalability.

Entity Framework

Entity Framework [4] is an object-relational mapper designed for the .NET framework. It allows for easy database schema and query design. Use of Entity allows server developers to model database entities and table relationships using a high-level programming layer which is easy to integrate into application logic. Entity also provides model visualizers for powerful database design. To demonstrate, Entity's visualizer was used to create Figure 5.1. Entity was used to design the database and application specific queries used regularly in the application.

Data Driven Documents (D3)

Data-Driven Documents (D3) [1] is a JavaScript library for manipulating documents based on data. The results of the server side virus description are parsed into a JSON string which is received by the client side browser. The client side JavaScript employs

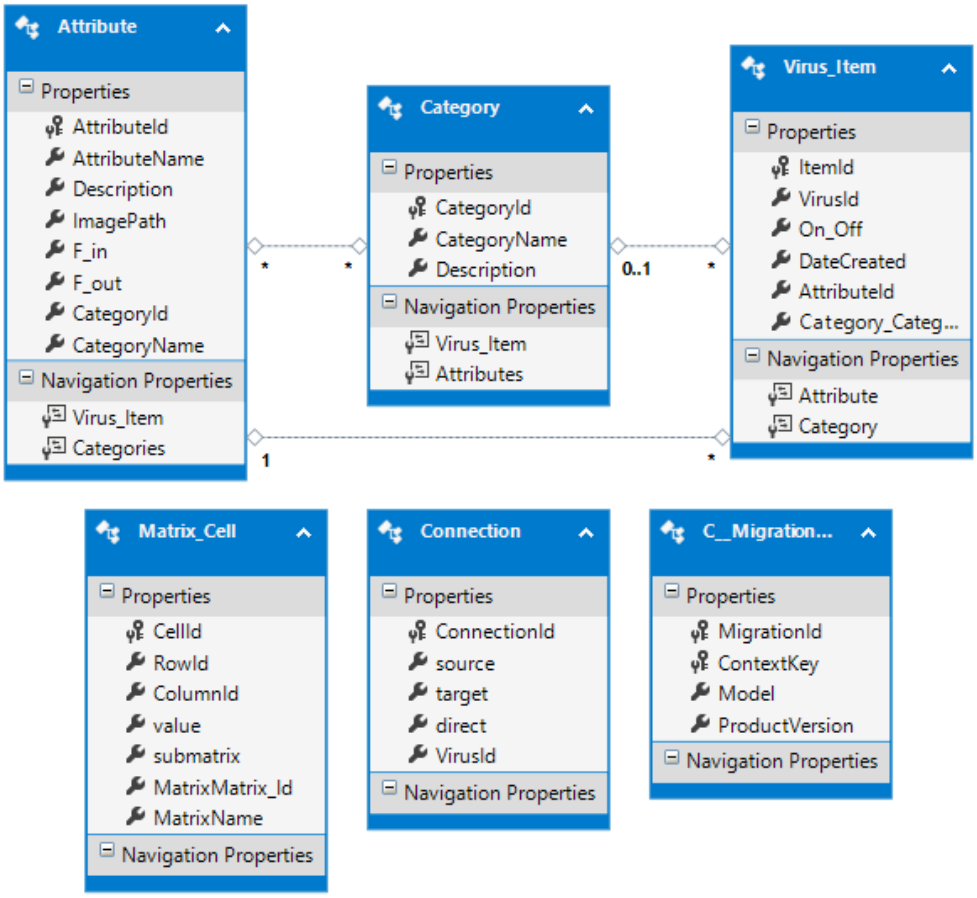

Figure 5.1: Visual Representation of the Database Design

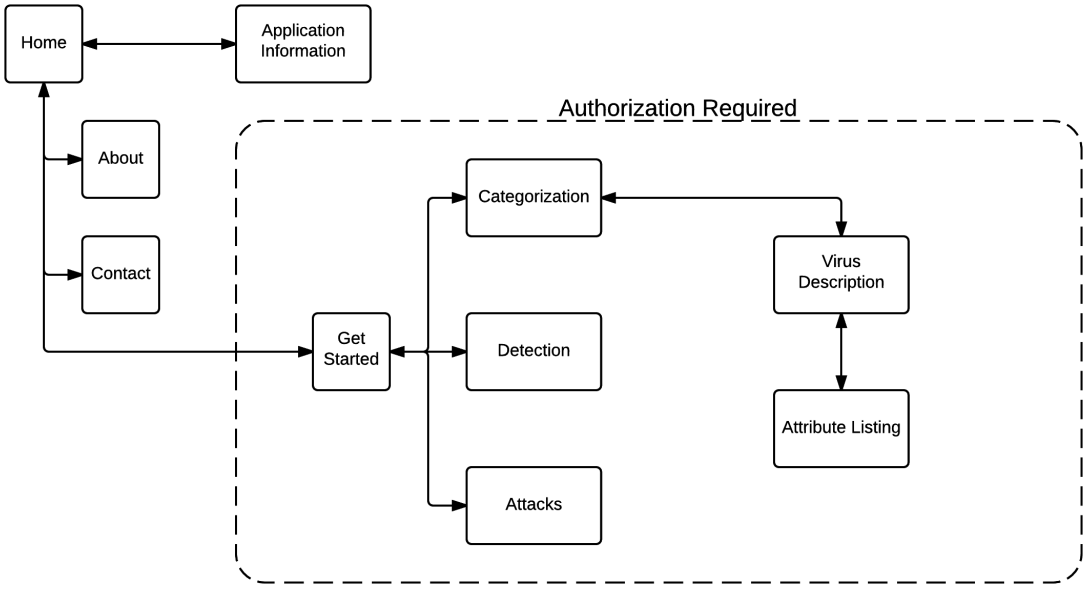

Figure 5.2: Overview of Web Site Architecture

the D3 library to create a visualization representing the received attributes as nodes and their interconnections as directed arrows.

Microsoft Azure

Microsoft's Azure cloud system is a high performance distributed hosting platform used to host both the database and application server for the HTS. The platform was built to allow developers the ability to design, host and manage applications using a variety of technologies. The powerful yet flexible infrastructure provides a reliable and responsive host for the system.

5.2.2 Website Architecture

Figure 5.2 provides a simple diagrammatic overview of the structure of the trojan system website. The *home, contact* and *about* pages are accessible to all traffic as well as the application information page which consists of three separate pages all providing background information for each of the primary applications. Users are required to create an account and be logged in to access the remainder of the site as displayed by the dashed line of Figure 5.2. Email confirmation is used to verify user account applications.

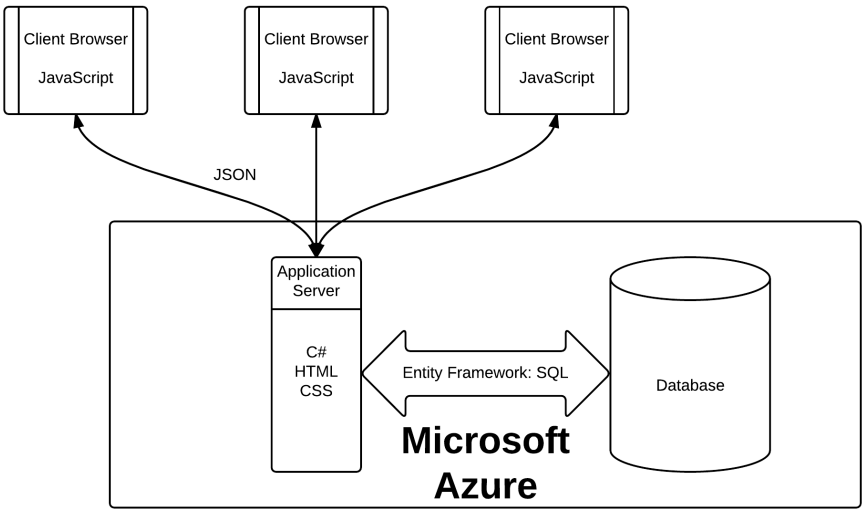

Figure 5.3: Distribution of Trojan System

Once users are authorized they are granted access to the *Get Started* page which provides basic information and access to each of the applications. To perform the classification method outlined in section **??** users are required to browse through a listing of the available 33 attributes shown in Figure 4.1. The selected attributes are then displayed on the *Virus Description* page where the user is able to visualize the result. For a more complete description of this application refer to section **??**.

5.2.3 Distributed System Architecture

As previously mentioned the hardware trojan system is hosted and operated in the 'cloud'. Microsoft provides a cheap and powerful cloud service named *Azure* which is discussed in section **??**. The *Azure* system is comprised of multiple server nodes and the HTS is hosted primarily in the western US node in California. The application server and the database, both in the 'cloud', are assigned network resources as needed. This dynamic allocation reduces service fees and provides a well defined structure for scalability. In addition Azure provides by the minute statistics for site management including traffic and data routing, data usage, page view statistics and site access locations as show in Figure 5.4.

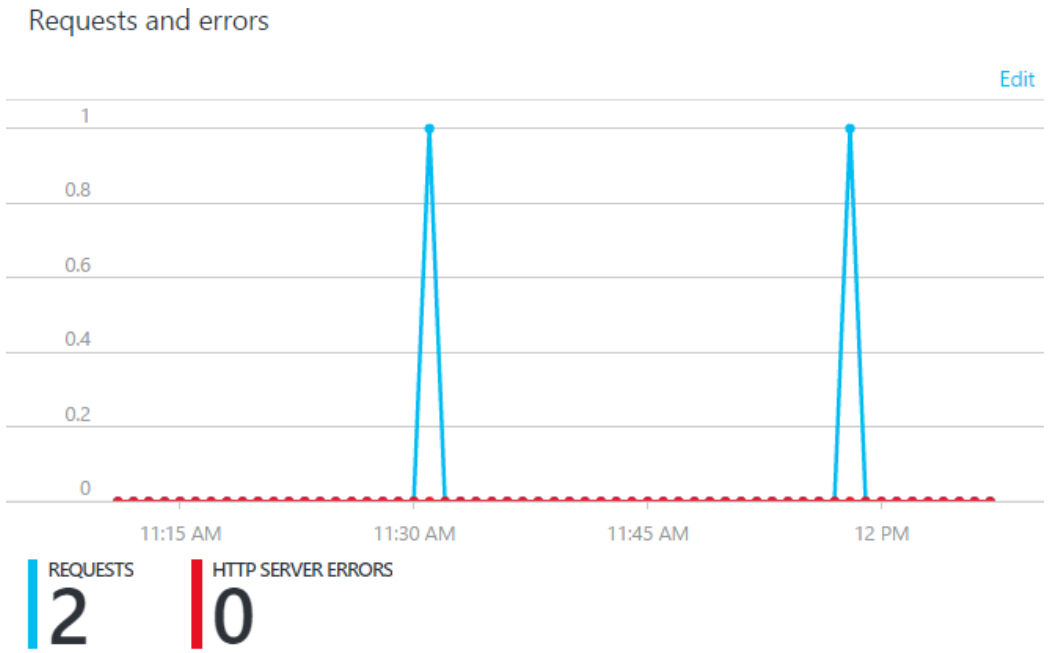

Figure 5.4: Example of Website Access and Error Statistics

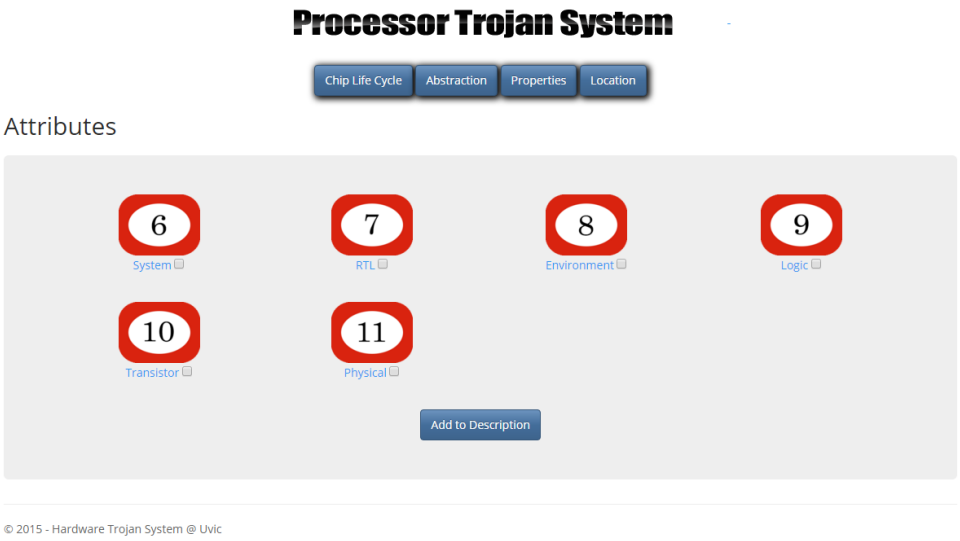

Figure 5.5: The Attribute Selection Page Filtered for Abstraction

5.3 Applications

5.3.1 Classification

The trojan classification tool is built on an on-line store model. A list of available attributes is provided for users to select from in a check box style. Figure 5.5 is a screen shot of the attribute selection page. By selecting one of the four buttons lining the header of the screen the attribute list page can be filtered to a particular category; in the case of Figure 5.5 the page has been filtered to only display attributes of the *Abstraction* category. Figure 5.1 is a visual representation of the schema used in the database. Note how the *Attribute* and *Category* entities hold a one to many relationship. When one of the filter buttons in Figure 5.5 is selected the corresponding *Category* entity is selected from the database and its *Attributes* navigation key is used to find all related entries in the *Attributes* table. The result is then displayed on the page.

When the user makes a selection of attributes and selects the 'Add to Description' button the selected attributes are added to a 'Shopping Cart' which is assigned a unique id based on the Http session state. This 'Cart' is then stored in the database as a list of objects in the *VirusItems* table which includes the user's unique id string and the id string for the cart. When the user is finished making selections they are

transferred to the 'Virus Description' page which can be seen in Figure 5.6. The 'Virus Description' page queries the database for the current 'Shopping Cart' by searcing the *VirusItems* table for all entries that match the current users id key and the current session state key.

Each entry in the cart contains an 'On/Off' value that allows the user to add the item to 'Virus Description' page then easily toggle each on and off. The provided grid displays the pertinent information for each entry in the cart as well as the 'On/Off' toggle and the ability to remove a chosen item from the cart.

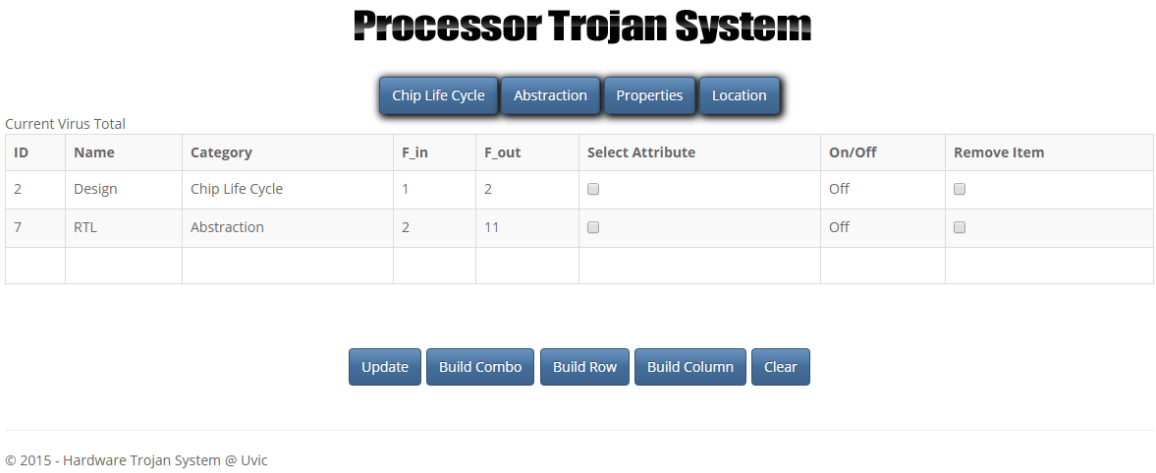

Figure 5.6: The Virus Description Page

When the user has finished their attribute selections and toggled all of the needed attributes to 'On' via the 'Update' button they are able to use the 'Build Combo' button. The HTS performs the matrix analysis described in section 4.2 and uses the D3 JavaScript API [1] to draw the directed graph shown in Figure 5.7.

As can be seen in Figure 5.7 the classification tool provides a drop down list that allows the user to make modifications to the produced graph. To investigate scenarios of trusted insertion points or undesired attributes the user is able to remove attibutes from the visualization. Figure 5.8 demonstrates the resulting visualization after attribute 4 has been removed from the graph of Figure 5.7. When an attribute is removed the system performs a modified version of the *Bellman-Ford Shortest Path* algorithm developed specifically for this application.

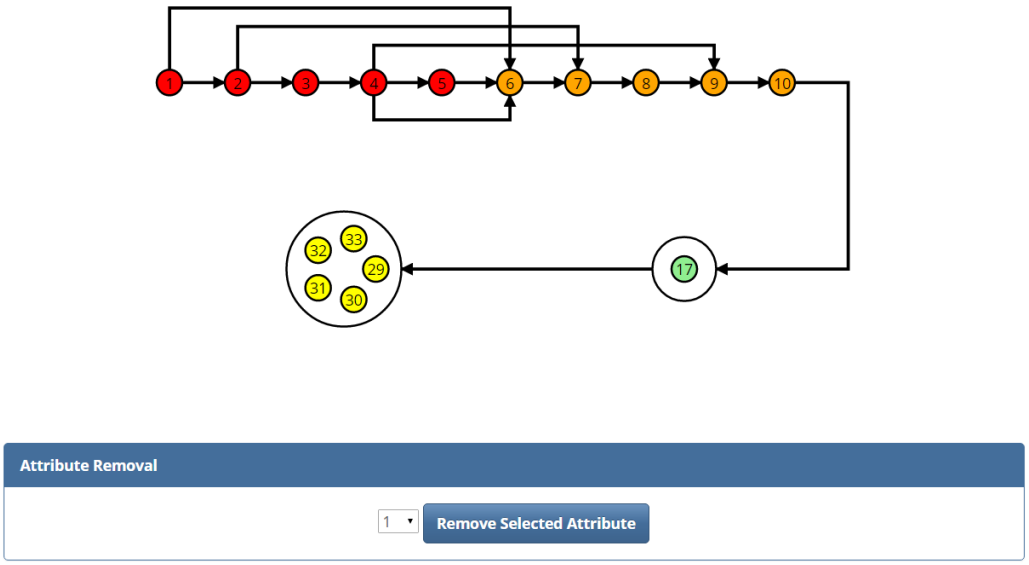

Figure 5.7: A Visual Representation of a Trojan Virus

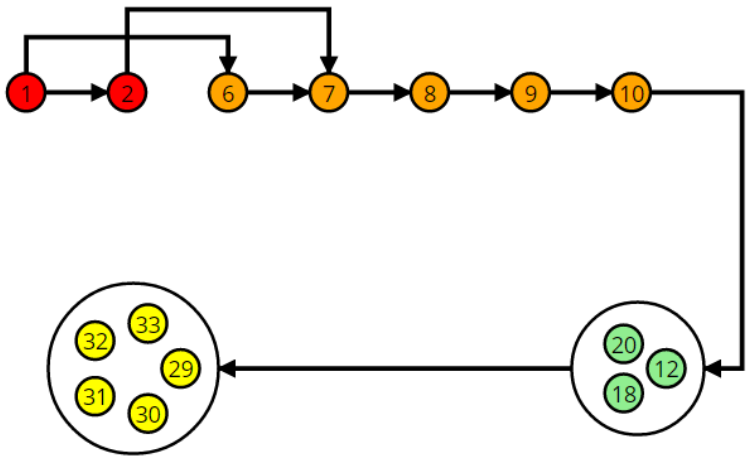

Figure 5.8: The Visual Representation after Attribute 4 is Removed

Finally, the classification tool computes and provides the severity rating for the computed trojan according to the method described in section 4.3, as can be seen in Figure 5.9. Once the user is satisfied with the resulting trojan they have built they can save their work via the 'Save' button at the bottom of the page which is not shown and include a 'nickName'. The virus and corresponding severity rating are saved in the database for access by the detection application described in section 5.3.2.

iR	iA	iE	iL	iF	iC	iP	iO	cR	cA	cE	cL	cF	cC	cP	cO
5	0	N/A	1	N/A	N/A	N/A	V	5	0	N/A	1	N/A	N/A	N/A	5

Figure 5.9: The Calculated Severity

5.3.2 Detection

When developers produce a new method of detecting hardware trojans they require some means of determining how it compares to currently known viruses and other detection methods. The ranking system described in section **??** provides a means for these methods to be compared and contrasted. In the HTS *Detection* application developers are able to investigate their methods systematically and derive a quantitative status value. Figure 5.10 provides a screen shot of the user interface used to create a coverage vector for a new detection method. The user is able to select values for each of the eight classes as well as assign the method a name. When saved this vector is stored in the database, related to the user's account. The HTS provides a number of automated tools that allow for the analysis of trojan viruses and detection methods. One such tool is the Trojan classification system described in [6] which employs a method of visualizing the categorization scheme described in [**?**]. The classification tool described in [6] provides and additional feature which constructs a severity rating for any trojan analyzed. These ratings can be saved to the database to be used in the detection tool. Please refer to [6] for more information. In the detection application users are able to search their saved viruses and retrieve the severity ratings to use for comparison. In Figure 5.11 it can be seen how the comparison tool provides drop-down boxes where users can search and select from either their saved detection

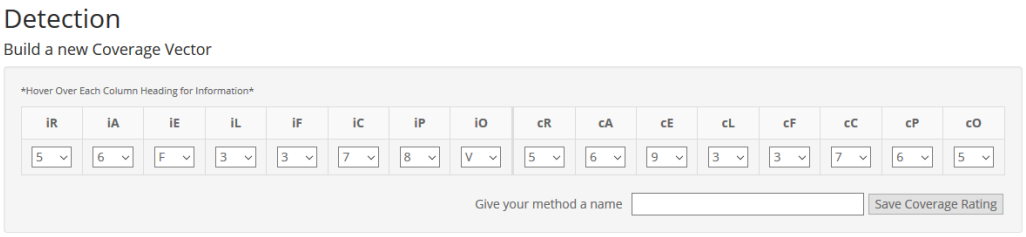

Figure 5.10: The User Interface for Creation of a new Coverage Vector for a Detection Method

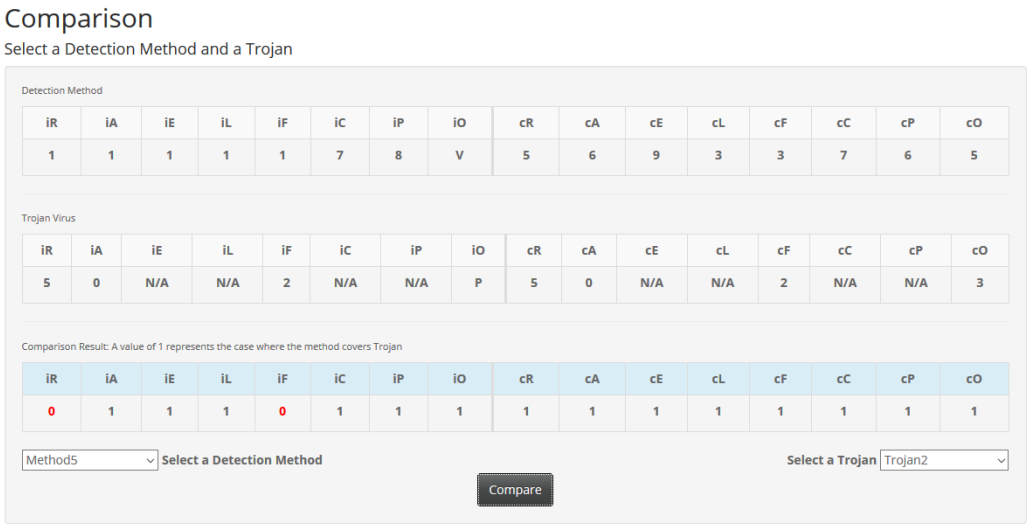

Figure 5.11: Comparison of a Detection's Coverage Vector and a Trojan's Severity Vector

methods on the left and their saved trojan severity vectors on the right. Once a vector is chosen for both a detection method as well as a virus the user is able to use the compare button which performs a comparison between the two vectors. The results row of Figure 5.11 displays a 1 in the case where the detection method has a value greater than or equal to its trojan counter part and displays a red 0 otherwise. This automated comparison allows the quick and effective investigation of competing trojans and detection methods. By use of this tool researchers have an easily accessible, centralized standard for development which will also provide a growing record of entries which can be used for statistical analysis.

5.3.3 Attacks

Chapter 6

Contributions

Appendix A

Additional Information

This is a good place to put tables, lots of results, perhaps all the data compiled in the experiments. By avoiding putting all the results inside the chapters themselves, the whole thing may become much more readable and the various tables can be linked to appropriately.

The main purpose of an Appendix however should be to take care of the future readers and researchers. This implies listing all the housekeeping facts needed to continue the research. For example: where is the raw data stored? where is the software used? which version of which operating system or library or experimental equipment was used and where can it be accessed again?

Ask yourself: if you were given this thesis to read with the goal that you will be expanding the research presented here, what would you like to have as housekeeping information and what do you need? Be kind to the future graduate students and to your supervisor who will be the one stuck in the middle trying to find where all the stuff was left!

Bibliography

[1] M. Bostock. D3.js v 3.5.6 [computer software], 2011.

[2] M. Nelson M. Potkonjak, A. Nahapetian and T. Massey. Hardware trojan horse detection using gate-level characterization. *Proc. IEEE/ACM Design Automation Conf.*, pages 688–693, 2009.

[3] Microsoft. Asp.net framework [computer software], 2002.

[4] Microsoft. Entity framework [computer software], 2008.

[5] Samer Moein. *Systematic Analysis and Methodologies for Hardware Security.* PhD thesis, University of Victoria, 3800 Finnerty Rd, Victoria, BC V8P 5C2, 2015.

[6] F. Gebali N. Houghton, S. Moein. A web tool for the automation of hardware trojan classification. *IEEE International Symposium on Circuits and Systems, submitted*, 2015.

[7] R. Chakaraborty S. Paul F. Wolff C.Papachristou K. Roy S. Narasimhan, D. Du and S. Bhunia. Hardware trojan detection by multiple-parameter sidechannel analysis. *IEEE Trans. Comput.*, 62(11):2183–2195, 2013. An optional note.

[8] T. Aaron Gulliver Fayez Gebali Samer Moein, Jayaram Subramnian and M. Watheq El-Kharashi. Classification of hardware trojan detection techniques. *IEEE International Conference on Computer Engineering and Systems*, 2015.

[9] T. Aaron Gulliver Fayez Gebali Samer Moein, Salman Khan and M. Watheq El-Kharashi. An attribute based classification of hardware trojans. *IEEE International Conference on Computer Engineering and Systems*, 2015.