BUILDING IN WEB APPLICATION SECURITY AT THE REQUIREMENTS STAGE: A TOOL FOR VISUALIZING AND EVALUATING SECURITY TRADE-OFFS

A thesis presented in partial fulfilment of the requirements for the degree of

Master of Information Science
in
Information Systems

at Massey University, Albany,
New Zealand.

Natalia Alekseevna Nehring

2007

Abstract

One dimension of Internet security is web application security. The purpose of this Design-science study was to design, build and evaluate a computer-based tool to support security vulnerability and risk assessment in the early stages of web application design. The tool facilitates risk assessment by managers and helps developers to model security requirements using an interactive tree diagram. The tool calculates residual risk for each component of a web application and for the application overall so developers are provided with better information for making decisions about which countermeasures to implement given limited resources for doing so. The tool supports taking a proactive approach to building in web application security at the requirements stage as opposed to the more common reactive approach of putting countermeasures in place after an attack and loss have been incurred. The primary contribution of the proposed tool is its ability to make known security-related information (e.g. known vulnerabilities, attacks and countermeasures) more accessible to developers who are not security experts and to translate lack of security measures into an understandable measure of relative residual risk. The latter is useful for managers who need to prioritize security spending.

Keywords: web application security, security requirements modelling, attack trees, threat trees, risk assessment.

Acknowledgements

This work has been made possible with the significant help and contribution of a number of special people. I would like to express my sincere thanks to all of them.

First, I would like to thank my supervisor, Dr. Ellen Rose, for her guidance, support, expertise, patience and friendship. She has always been passionate with research and patient with me in explaining different things and available to help me whenever I needed. I was welcomed into her office for help at any time. Dr. Ellen Rose has always shown other angles and made perceptive remarks that have guided my research in the right direction. She has contributed significantly to my research. I really appreciate this help; especially as English is not my first language. Ellen is the person who has set up a valuable example for me to follow, and from whom I have learnt a lot in the last couple of years. She has helped me in developing academic skills which were important for researching at the required level. She has encouraged me in my efforts, prompted me to correct my mistakes and taught me that the best way of doing things is to do your best to make progress as you go along rather than rushing through at the last minute.

I would like to acknowledge the value of two scholarships: the Massey University Masterate scholarship and the NZ Federation of Graduate Women North Shore Branch Top scholarship, which provided me with financial support during the time of my research.

I would like to express my great appreciation for members of *The Code Project* website (www.codeproject.com). This web site is a great source of programming materials, including articles, examples and source code for different programming languages. Some components from this web site made the implementation of the proposed tool easier. Also, I used examples of code from this web site to learn C#; a new programming

language for me.

I must also thank the staff of two companies for giving me their time and helping me with the process of evaluating my tool.

I would like to express very special thanks to my husband Croydon Nehring for being such a supportive person. He has been supportive and encouraging in my doing this research. He has also been a care person for our kids, and has replaced me for them when I was busy. He was with me in my ups and downs that go with a research effort of this scope, and encourages me to be a warrior, to do my best and to look forward.

Other special thanks go to my parents, Alexei and Nelli Gorobets, for their support with providing childcare for my kids. Big thanks go to my children, Maria, Anastasia, Timofei and Nicolai, for their enthusiastic and unconditional love, which has always cheered me on.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction and Background

1.1 Introduction

The security of web applications has become a central issue for online businesses. The e-Crime Watch Survey (2004) found that 40% of businesses feel hackers represent their greatest cyber security threat (CSO magazine, 2004). The 2006 Deloitte Touche Tohmatsu Global Security Survey of top financial institutions recently reported a shift from infrastructure to application layer attacks (p. 35) as well as the following findings. Only 7 percent conduct quarterly security code reviews, 2 percent do semi-annual reviews, 65 percent do ad hoc reviews and 13 percent never do reviews. The number of online attacks reported in this annual survey grew by 25 percent with 78 percent reporting security breaches from external attacks. In the Asia-Pacific region, excluding Japan, the number of online attacks grew from 16 percent in 2005 to 100 percent in 2006; every organisation surveyed in the Asia-Pacific region had been attacked a minimum of once during the 12 month period (Deloitte Touche Tohmatsu, 2006). As current web sites are more likely to be complex online information systems and not just simple HTML pages, web site security has become more complicated. The security of a web site has a number of dimensions; one of them is web application security. John Pescatore, an analyst at Gartner Inc. in Stamford, Connecticut said "web application security is a serious problem for two-thirds of all corporate web sites" (Verton, 2002, p. 9). Unfortunately, the growth in security problems is keeping pace with growth in the number of Internet users and companies using web sites to carry out business online.

In contrast to the predominantly reactive security practise of detecting and correcting web application security problems, this thesis work seeks to design and develop a tool to support web application developers in taking a proactive approach to building in web application security at the requirements stage. The IT community knows about countermeasures, security patterns, attack patterns and existing vulnerabilities but people are still developing web applications which are not secure. To solve this problem, there is a need to make this information more accessible. Since managers must see a reason to invest in security measures, the ability to more effectively assess

risk and the potential loss of not implementing security is needed. This research has designed and prototyped a tool that provides support to both managers and developers in making these tough decisions.

1.2 Research Objectives and Contributions

The Design Science research approach has been taken (Gregg, Kulkarni, & Vinze, 2001; Hevner & March, 2003; Hevner, March, Park, & Ram, 2004; Zelkowitz & Wallace, 1998) in order to achieve the following research objectives:

> Objective 1: To design and prototype a tool for use by managers and developers for visualizing and evaluating security trade-offs and risks in alternative web application designs.
>
> Objective 2: To demonstrate the utility of the tool via evaluation in a real web application development environment. Utility has been measured in terms of user satisfaction with the tool's ability to support risk assessment and to facilitate identification of vulnerabilities during the requirements stage.

The proposed solution is a tool for visualizing and evaluating security trade-offs in alternative web application designs. The tool is designed to help developers visualize attack patterns and build threat trees in order to identify potential vulnerability points in web applications. It also provides the ability to assess risk and to identify trade-offs in order to determine which security requirements should take priority. The tool can generate visual representations of attacks and vulnerabilities for different kinds of web applications to help developers identify and prioritise security requirements rather than reacting to security problems after they happen.

The prototype tool stores information on language independent web application vulnerabilities. Any language-specific problems are delimited to web applications written using PHP and the MySQL database. To get businesses to take proactive security measures more seriously we need to reduce the up front cost for security risk analysis. There is a need to reduce the learning curve and improve access to existing knowledge about potential threats, web application vulnerabilities, countermeasures and potential losses from not implementing countermeasures.

1.3 Background to the Study

The May 2007 Netcraft survey reported the existence of 118,023,363 web sites, an increase of 12.8 million from the 2006 total of 30.9 million (Netcraft Ltd, 2007). The current state of security of such sites was underlined by Auronen (2002, p. 2) who stated that sensitive data is "usually protected by only weak access control mechanisms vulnerable to many types of attack". Database driven web applications are the heart of today's web sites. Given their central role, security requirements should be considered from the initial stage of web application development. Writing code with security in mind could help to make web sites more secure against a wide variety of known attacks. However, a 2006 survey of top financial institutions around the world by Deloitte Touche Tohmatsu shows only 26 percent of respondents named application security as a top priority. 56 percent of respondents stated that poor software development compromises quality and may become a security threat in the future (Deloitte Touche Tohmatsu, 2006).

Potential for exposure must be continually assessed during the iterative process of web application development to ensure changes don't introduce new vulnerabilities and to ensure that protection exists from newly discovered types of attacks. Security breaches can affect the organisation that owns the web site, but can have an even greater impact on customers when private information is revealed or financial losses are incurred (Schneier, 2004).

Security assessment should be thought of as an ongoing process, not a one shot accident according to the Open Web Application Security Project (OWASP, 2005c). This process includes a number of steps. First of all it is necessary to define and know your enemy – vulnerabilities of web applications. Organisations such as the Computer Emergency Response Team (CERT) often publish known vulnerabilities. A list of the ten most dangerous can be found at OWASP (2005b). The second step is taking a proactive approach to ensure security, like building security into the design of web applications. The remaining steps are reactive. They include monitoring web site activity regularly and using this information to maintain running web applications in terms of security enhancements to ensure changes in requirements will not compromise security. Figure 1.1 shows a set of best security practices, meant to be

followed during software development. This research focuses on two early stages of web application development: security requirements and risk analysis, highlighted in Figure 1.1. The study produced a prototype of a software-based security tool to support these two stages. The tool brings together existing security knowledge to reduce the effort required to conduct risk assessments for web applications.

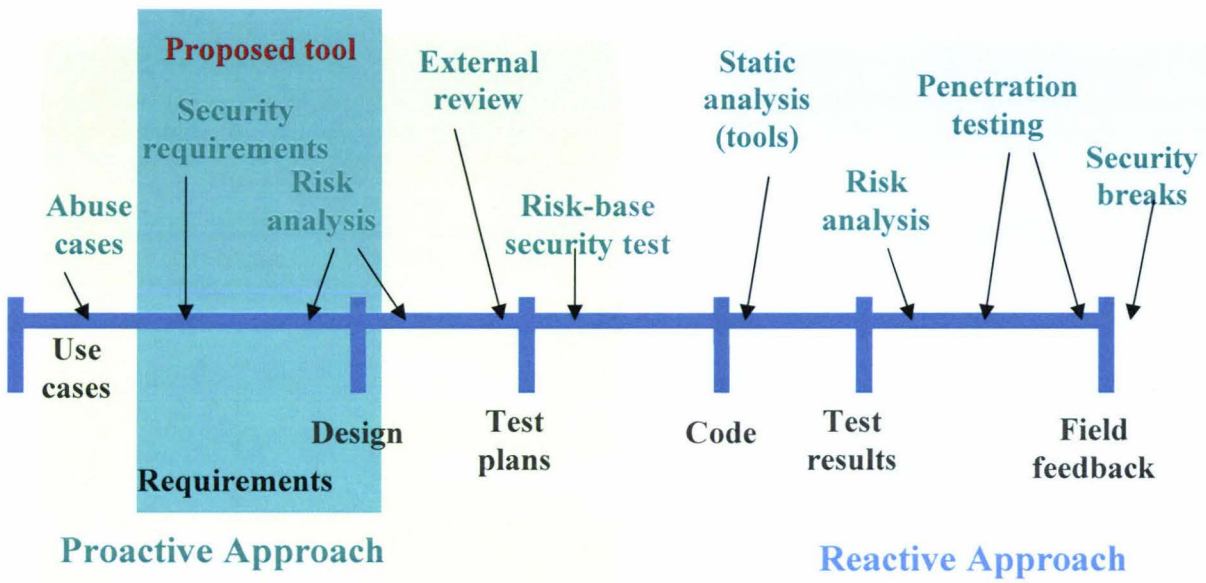

Figure 1.1 Best security practices for software development.

Adapted from: (McGraw, 2004)

An extension of the model shown in Figure 1.1 is illustrated in Figure 1.2 (Barnum and McGraw, 2005). The extension points out specific types of security knowledge (e.g. principles) and identifies the security activities (e.g. risk analysis) in the software development life cycle (e.g. design stage) where the knowledge is likely to be of greatest use. An understanding of these relationships provides a solid base for software security best practice. This becomes extremely important in practical software development given the industry faces a shortage of experienced security experts (Barnum & McGraw, 2005). Barnum and McGraw define three knowledge categories: prescriptive, diagnostic, and historical. The prescriptive knowledge category includes actions or procedures which need to be followed, like data principles, guidelines, and rules. Attack patterns, exploits, and vulnerabilities help in determining the capability of a component to perform its functions and are therefore classified as diagnostic knowledge. Prior diagnostic knowledge helps the practitioner

to understand the real problem based on extensive experience with the same or a similar problem. Common security problems like vulnerabilities and corresponding attacks can be detected and dealt with using prior experience with these problems. This category of knowledge helps in recognising common problems and is invaluable during the development stage. Information that helps to define previously existing risks belongs to the historical knowledge category (Barnum & McGraw, 2005). In relationship to Figure 1.1, this research seeks to design a tool which supports the definition of security requirements in terms of vulnerabilities, known attacks on each type of vulnerability and known countermeasures to reduce the potential damage from an attack. The tool provides a database of vulnerabilities, attacks and countermeasures to support doing a risk assessment in the early stages of web application development. The tool calculates the risk for each component of the web application being assessed and stores this information so it will be available for managers to view at later dates for the purposes of doing what-if analyses and making comparisons between different risk mitigation strategies in terms of residual risk (i.e. unmitigated risk) and the costs associated with implementing countermeasures.

Different types of security knowledge can trigger security activities at different stages of software development. Conducting security assessment activities during the early stages of development is referred to as the proactive approach. Knowledge about attack patterns can be applied at the requirements and design stages to conduct risk assessments. This knowledge is also useful at the test plan creation stage for running risk based security tests. Figure 1.2 claims knowledge about vulnerabilities is only used in the later stages of development. In reality, vulnerability knowledge can also be useful in the early stages of web application development as part of a proactive approach. In contrast, a reactive approach seeks to discover vulnerabilities in the code after it has been released. A proactive approach seeks to prevent or reduce vulnerability (i.e. weaknesses) in the code during development. Developers need to have knowledge of potential vulnerabilities and attacks before they can consider countermeasures to reduce or remove vulnerabilities. Knowing about vulnerabilities before coding helps to save time at later stages where these identified problems can cause significant delays in further development and/or releasing the software. Knowledge about attack patterns can assist developers in writing security requirements and in providing protection against particular identified attacks. This

knowledge can also be used to write risk-based security tests. A principle is defined as a statement of existing security knowledge, which comes from an experienced practitioner and from real-world knowledge of building secure systems (Barnum & McGraw, 2005). Principles are helpful in two ways: in detecting architectural defects in software, and in promoting good security practices. A principle is often documented using a title, description, examples, references, related rules and guidelines. Guidelines can be useful for creating security requirements and evaluating alternative designs (See Figure 1.2.)

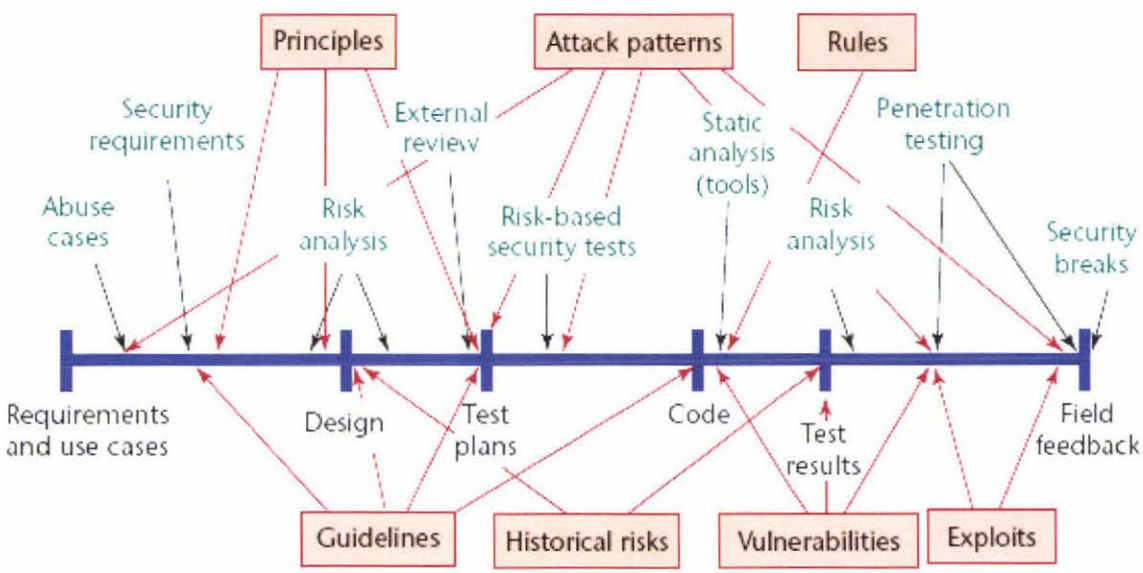

Figure 1.2 Applying security knowledge during the software development life cycle. Source: (Barnum & McGraw, 2005, p.3)

Historical risks are detailed catalogues with descriptions of specific issues which were discovered in real-world software development. For example, a list of discovered software vulnerabilities (i.e. weaknesses) is a type of historical risk. Each risk item of this catalogue has a statement of impact on the business. Historical knowledge can become a valuable security resource, which helps to identify similar issues in new software development. A catalogue of historical knowledge can save developers time and effort in identifying potential security issues. The proposed tool makes a database of historical security information available to web application developers.

Gathering and interpreting available data on vulnerabilities can be an onerous task, taking a considerable amount of a developer's or analyst's time. The old proverb: "A picture is worth a thousand words" implies people may absorb complex information more readily from pictures than from large volumes of text. Visual representations of complex relationships amongst web application components, their vulnerabilities, attacks based on these vulnerabilities, and the magnitude of potential losses can quickly highlight areas of major concern, facilitating security requirements analysis and risk assessment.

1.3.1 Risk

Security assessment is often associated with the concept of risk. Risk can be viewed as a function of the likelihood that a threat will materialise, the level of vulnerability and the potential for loss of resources. Thinking about negative scenarios in these terms is an essential skill for a test engineer (Alexander, 2003). In this sense, a web application designer should also think about requirements in terms of negative scenarios, that is, from a hacker's point of view. An understanding of vulnerability, threats and attacks is relevant to risk measurement (Amoroso, 1994) where:

- A vulnerability is a characteristic (or weakness) of the software that makes it possible for a threat to occur.
- A threat can be defined as an event which can have an undesirable effect on assets and resources (e.g., loss of data, corruption of data, exposure of confidential information).
- An attack is an action by a malicious user that involves exploiting vulnerabilities in order to cause a threat to occur.

Vulnerability and attacks are only of concern if they introduce the potential for threats that would involve significant resource loss (Amoroso, 1994). If you increase any of these three variables, risk also increases. If you reduce them, it decreases. More formally risk has also been defined as "the probability of a vulnerability being exploited in the current environment, leading to a degree of loss of confidentiality, integrity, or availability, of an asset" (Microsoft Corporation, 2006b, p.27). The potential impact of a threat is related to the degree of a resource's vulnerability as well as the resource or asset's value. The higher the value of an asset, the greater the

potential loss will be from realisation of a threat, and consequently the higher the degree of risk (Amoroso, 1994).

Risk needs to be assessed and managed. Microsoft Corporation recently published a document called "The Security Risk Management Guide" (2006). The document outlines an iterative, four phase (i.e. assessing risk, conducting decision support, implementing controls and measuring program effectiveness) proactive approach to risk management based on industry standards. The goal of Microsoft's approach is to balance cost and effectiveness. Qualitative steps for identifying the most important risks are followed using a process which starts with identifying roles and responsibilities. Managers are responsible for assessing asset value and potential impact of a risk, security personnel are responsible for identifying the likelihood of a risk occurring by taking current and proposed countermeasures/controls into consideration and developers are responsible for implementing the countermeasures for risks identified as unacceptable. In this guide, risk management is defined as "the overall effort to manage risk to an acceptable level across the business" and risk assessment is defined as "the process to identify and prioritize risks to the business" (Microsoft Corporation, 2006b, p. 16).

Similarly, Boehm (1991) identified the two main stages of risk management as risk assessment and risk control (Boehm, 1991). Risk assessment includes risk identification, risk analysis, and risk prioritization where (Boehm, 1991):

- Risk identification results in a list of project-specific vulnerabilities which can be dangerous for a project
- Risk analysis assesses the loss probability and loss magnitude for each vulnerability and
- Risk prioritisation ranks the risks in order of those to be dealt with in descending order of urgency.

Today, most organisations understand the importance of risk management and assessment but still experience difficulty with the application of modelling techniques to both risk management and risk assessment. One reason it is difficult to utilise these techniques, is the lack of advice on what to do and how to do it. Security analysts can

define vulnerability, but it can be difficult to see the overall picture with respect to evaluating the impact in terms of costs incurred for either choosing to mitigate or not mitigate specific threats associated with specific vulnerabilities. It is rare for an organization at the project management or portfolio level to use a risk management tool or framework to assess a risk, and identify its impact (Steven, 2006).

Organizations need more accurate information and more accessible information for risk assessment. The right information should be present in a familiar way and be easy for non-security experts to use. A risk calculation in these terms can give a business an improved ability to make better decisions on how much to spend in order to achieve a desired level of security (Steven, 2006). Numbers are not magic, but with the right information from experts they can serve as advisory indicators for a security decision. Serious application level security problems are still present in professionally designed web applications. To address web application security problems, decision-support tools and techniques are needed (Scott & Sharp, 2003).

1.3.2 Why software tools for risk assessment are needed

Proactive approaches to security involve consideration of the risk level which in turn depends on the likelihood of particular threats, the potential for loss, the effort required to execute particular kinds of attacks and the level of vulnerability as well as dependencies between these factors. The following sections summarise key aspects of the need for software tools to support risk assessment, as a part of a proactive security practice. "Security is a process, not a product, but we still need accurate and reliable products to calculate security quantitatively to improve security" (Sahinoglu, 2005, p. 23). Security should not be treated as an add-on feature. Security should be considered from the requirements stage, as a key system requirement, especially for systems that utilise components in both public and private networks. All possible security requirements can not usually be implemented, because available resources are limited. Every software project has limitations in terms of available time, budget and expertise. A change in mindset is required where the following points are considered (Feather, Sigal, Cornford, & Hutchinson, 2001):

- quality and risk estimation can be as important as budget and schedule

- limited project resources should be more optimally allocated.
- trade-off opportunities should be identified and evaluated

To achieve more optimal allocation of resources, managers need better information and more cost-effective ways of analysing that information. It is important for managers to understand the potential costs of not implementing countermeasures in order for them to make more informed decisions about allocating limited resources to security measures. In addition, security issues become more understandable to a business when they are expressed in familiar form. The question of "What data needs to be collected and what needs to be measured?" arises. Security risk assessment tools can provide decision support for managers who aim to balance the cost of a loss with the cost of countermeasures. Business leaders should ask the following questions about security (Geer, Hoo, & Jaquith, 2003):

- How secure am I?
- Am I better off than I was this time last year?
- How do I compare with my peers?
- Am I spending the right amount of money?
- What are my risk-transfer options? (Geer et al., 2003)

Due to the nature of the Internet, web based systems are vulnerable to outsider and insider threats. A number of reasons why web based systems are vulnerable include (Zhou, 2002):

- Web-based information systems can be accessed by any Internet user.
- System applications can be invisible and difficult to review.
- Unauthorised access can be hard to trace.
- The possibility of security breaches in web information systems is higher than in centralised systems. The effects can be costly: systems can be destroyed and sensitive information can be stolen.
- Data records can be accessed indirectly and modified by unauthorised persons.
- Numbers of attacks are rising due to two factors: the Internet is now widely available and many financial systems are now linked to the Internet (e.g. online banking, online trading).

To address security across a software project's lifecycle a number of factors need to be considered. They include security requirements specification from the viewpoints of

various stakeholders, specification of the environment in which the software will operate, specification of the software and hardware modules, and specification of the expected length of time the software will be used. The recent interest shown by companies such as Microsoft in the development of new security tools for the analysis and modelling of security requirements for web applications reveals a change in attitudes towards web application security in the industry. A computer-based security analysis tool can be a valuable aid to the process of risk assessment. Such a tool can provide assistance in the evaluation of which software risks need to be addressed first, helping to mitigate risk, and show the effectiveness of countermeasures (D.P. Gilliam, 2004).

1.4 Overview of the Thesis

This chapter outlined a current problem with web application security, namely the need to think about security early on and to make existing knowledge about vulnerabilities, attacks and countermeasures more accessible to developers and their managers so they can conduct risk assessments. In addition, the research objectives were stated and a brief background provided on why such a study is needed and who might benefit from the outcomes. Chapter 2 outlines the nature of the Design Science research method and explains why it is an appropriate approach for achieving the research objectives. The second chapter begins with a discussion of the generic steps in the Design Science research method then continues with specific details on how this research was done. Iteration through the design process, the changes made to the prototype after each cycle through the process, details on how the tool was evaluated and, how the findings led to prototype changes, as well as the steps involved in the implementation of changes are discussed. The remaining chapters are organised based on the steps and outcomes of the Design Science research method.

Chapter 3 discusses the proposed security tool's functional and non-functional requirements. Requirements are defined as what a system must do. They describe a necessary attribute, capability, characteristic, or quality in order to provide value for the tool's intended users (Sommerville, 2007). Functional requirements are necessary application related capabilities in terms of what the system should be able to do for

the user. Non-functional requirements are quality aspects such as usability, performance, reliability and safety. Chapter 4 discusses the tool's architecture and the detailed design of its major components. Chapter 5 presents the results of three rounds of external evaluations of the tool in terms of its ability to meet the requirements stated in Chapter 3. Chapter 6 relates this study to prior work and compares the proposed tool to two similar security analysis tools. This chapter also discusses how the positive and negative results from the final evaluation led to implications for the use of the tool in practice by different groups of users. Finally, Chapter 7 briefly restates the contributions of this research, discusses the limitations of the tool and draws implications for further research based on the identified limitations.