Pluggable Model-Based Security Policy Enforcement Mechanism for Software Development

by
Javier Navarro-Machuca

Submitted in partial fulfillment
of the requirements for the degree of
Doctor of Professional Studies
in Computing

at

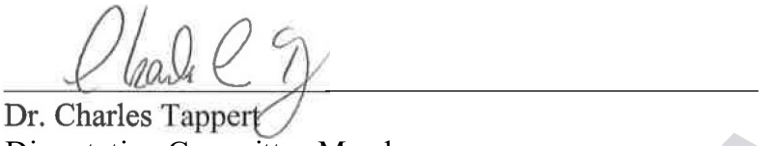
School of Computer Science and Information Systems

Pace University

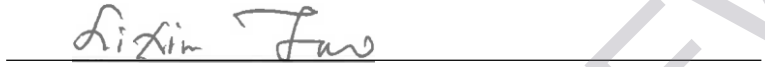
July 2016

We hereby certify that this dissertation, submitted by **Javier Navarro-Machuca**, satisfies the dissertation requirements for the degree of *Doctor of Professional Studies in Computing* and has been approved.

July 1, 2016
Dr. Li-Chiou Chen — Date
Chairperson of Dissertation Committee



July 1, 2016
Dr. Charles Tappert — Date
Dissertation Committee Member



July 1, 2016
Dr. Lixin Tao — Date
Dissertation Committee Member

School of Computer Science and Information Systems
Pace University

Abstract

A Pluggable Model-Based Security Policy Enforcement Mechanism for Software Development

by
Javier Navarro-Machuca

Submitted in partial fulfillment
of the requirements for the degree of
Doctor of Professional Studies
in Computing

July 2016

Security in software applications is frequently an afterthought. Even if developers are aware of software vulnerabilities, they possess little knowledge of how to secure the applications while writing codes. In addition, the lack of tools for security automation makes it more challenging to protect systems and applications. This dissertation introduces a framework to incorporate security policies for data fields in the transactions of software application during its development phase. The objective is to facilitate developers to apply security policies on the data required by the regulations. The extensibility of the presented model gives the flexibility to accommodate different security requirements and to implement them as security functions. With the simplicity of mapping data fields of business structures with security policies and their associated security functions, this approach provides the programmers, business domain experts and security experts a collaborative process to define and incorporate security requirements in software. The proposed model-based security policy mechanism addresses the complexity of securing confidential information at the process level by enforcing pre-defined security policies on the data before the data is transmitted outside the application boundary, regardless of the destination or repository that the data will be stored. The separation of security policies and the application provides a granular control to protect the data field via different security techniques such as access control or encryption. This mechanism is flexible so that it can be used in either legacy applications or new applications. The application of this approach on the payment card industry payment application data security standard has been evaluated to validate the flexibility and extensibility of the proposed model.

Acknowledgements

I would like to express my sincere gratitude to Pace University for offering me the opportunity to complete my degree here. I also thank Dr. Li-Chiou Chen, my dissertation advisor, whom for without her guidance, encouragement and understanding, I wouldn't have made it this far. In addition, I remain grateful and thankful to all my professors whom have managed and directed me throughout my time of study here.

In memoriam of my Software Engineering professor Dr. Robert B. France, Department of Computers Science, Colorado State University. Your teachings inspired this dissertation.

Finally, and most importantly, huge thank you to my wife Lorena for her full supports and for giving me two beautiful twin daughters who were born in the middle of this journey. To the Almighty God, for His grace in me.

"Education is the key, not tools" – Dr. Robert B. France

Table of Contents

List of Tables

List of Figures

List of Charts

Chapter 1

Introduction

1.1 Motivation

Application developers are commonly aware of security vulnerabilities, but they generally have little or no knowledge about the causes and measures that be incorporated in applications to avoid those vulnerabilities [1] during the software development life cycle. The lack of security consideration in software applications are caused by different reasons. In some circumstances the development team does not possess the expertise of developing applications with security in mind [1] [2], it usually leaves the developers to make security decisions instead of providing them a holistic view of security policies and requirements needed by the applications. Besides, many cybersecurity practitioners focus on the promise of network security as the silver bullet solution [3] which does not solve the fundamental problems caused by software vulnerabilities.

Without security design upfront, developers will need to fit the security design late in the development phase which often involves high level of redesign and code refactoring, causing delays in delivery dates and increasing the cost to fix security problems. Companies from different sectors have spent considerable amounts of money to secure their systems, but current efforts are not enough to secure their systems since data breaches are reported constantly, which had resulted in millions of losses in various incidents [4]. For example, an attack on direct marketer Epsilon had put an estimated 60 million records at risk. In another incident, one million passwords were stolen from Sony Pictures, 77

million accounts were compromised at the company's PlayStation network, and 25 million records were breached at Sony Online Entertainment [5].

Table 1. Top 6 incidents all time (exposed records count) [6]

Breach Reported Date	Summary	Records Exposed	Organization's Name	Industry-Sector	Breach Location
Highest All Time 8/22/2014	Hack of websites exposes names, registration numbers, usernames and passwords	220 Million	Organization's Name has not been reported	Unknown	South Korea
Number 2 12/28/2015	Misconfigured database exposes voter names, dates of birth, addresses, phone numbers, political party affiliations, genders, and other assorted personal details	191 Million	Organization's Name has not been reported	Unknown	United States
Number 3 6/21/2014	Hack exposes trip details of customers after de-anonymizing MD5 hashes	173 Million	NYC Taxi & Limousine Commission	Government -City	United States
Number 4 10/3/2013	Hack exposed customer names, IDs, encrypted passwords and debit/ credit card numbers with expiration dates, source code and other customer order information.	152 Million	Adobe Systems, Inc.	Business - Technology	United States

Number 5 3/17/2012	Firm may have illegally bought and sold customers' information	150 Million	Shanghai Roadway D&B Marketing Services Co. Ltd	Business - Data	China
Number 6 5/21/2014	Hack exposes names, encrypted passwords, email addresses, registered addresses, phone numbers and dates of birth	145 Million	eBay, Inc.	Business - Retail	United States

The later a security problem is detected in software application, the more expensive it is to resolve the problem. Design-level problems accounted for about 50% of the security flaws [1]. There is an urgent need to help the average programmers/developers to implement security mechanisms at the transactions and data levels while they are developing and maintaining the applications to process these transactions and data. Modern applications require to be secured wholly, and securing them completely is not a trivial task since attackers have the luxury of choice [3]. As many developers are not security experts, they will need assistance in determining security policies and security configurations [1] [7].

With the introduction of distributed systems, especially the ones that are Internet enabled, securing data is one of the major priorities for organizations. Many database management systems provide enhanced mechanisms to protect the data they store [7]. Some of these mechanisms are: user access control and data encryption. But data

confidentiality is in risk when an authorized database administrator gets access to pull the data from the database. Encrypting sensitive information at the database level is considered to be a good practice, but it is not enough to secure the data in multi-tier applications. Database systems decrypt the data when they are requested from applications, and consequently these applications perform several actions on the sensitive information. These kind of activities may put the data at risk since, regardless of the effort of securing the data at the database, these data are vulnerable once are out of the database boundaries. Besides, computer systems do not store data in databases only, they also use files or cloud providers to persist confidential information.

1.2 Problem Description

The challenge in security of complex distributed systems does not lie anymore in encryption or access control of a single middleware platform, but in the protection of the system as a whole [8]. An application can send confidential information safely via an encrypted channel to a database that saves the data encrypted as well, but the same application may create temporary files that contains sensitive data in plain text. When an intruder breaks into a system or when a malicious system administrator logs into a server, data get exposed. The user with bad intention may pull data from databases or files. Consequently, data confidentially should become a priority for corporations.

Data have become one of the most important assets for corporations, hence, they need to be protected and handled responsibly. Categorization and classification of data are practices that are commonly required in information security standards and regulations,

they help with identifying what level of security severity should be applied to what data fields, as a consequence, all stakeholders should be aware of the risks of not protecting the data properly. Data are usually created and manipulated in software applications, therefore, these applications need to know how to keep sensitive data secured before, during, and after software transactions. Programmers need to implement security controls in software applications to guarantee that data will be secured as per specified in the business requirements, standards and regulations.

Once we have become aware of the problems on securing data in software applications we ask: ***how can programmers apply security controls on data fields during application development as per the business requirements and regulations?***

Here, we propose and approach that addresses the previous question. We want to enable the programmers/developers to leverage the knowledge and decisions made by both the security experts and the domain experts to incorporate security in software design. Such design will be driven by business requirements determined by domain experts and the security policies mapping to the transactions/data decided by the security experts. The goal of this research is to provide a new method for the development team to embed security policies at the design phase and when writing application code.

Software applications are built to perform transactions and manage data for multiple business domains. For some domains, there are organizations that outline the standards and specifications of how the users and computer systems interact with sensitive information. Information security standards are mainly defined to increase controls on

sensitive data that are usually manipulated by computer systems and applications. Therefore, it is important that programmers can implement such security controls in software application to fulfill such requirements. One main objective of the proposed mechanism is to be domain agnostic. Thus, its class model needs to be extensible enough to implement any data security requirement via the model-based mapping technique, this technique allows us to classify sensitive data in application code, then we can automate the execution of security policies on them. This implementations can be packaged and reused in other new or existing applications that need to comply with the same or similar requirements.

With the proposed mechanism we want to make very difficult for attackers to understand the protected information, by ensuring that the sensitive data get safely handled before they leave the boundaries of the applications regardless of the destination and how they are transmitted. Analogically to "teach your child to swim instead of locking the fence doors or placing lifeguards in all the swimming pools," this approach "teaches" the software application to protect the sensitive information instead of securing all the systems that exchanges these data with it; specially if those systems are external and we do not have control on them. The research focuses on the model-based technique for embedding security policies during software design and implementation. The enforcement of the security policy requires a highly reliable mechanism [9] and yet without being intrusive to the application development. This document provides the definitions, specification and object interactions for the presented model-based mechanism. In addition, we will discuss

an application that uses this approach in securing credit cardholder information based on the PCI PA-DSS specifications [10].

The security gateway practice [11] is the least intrusive one among the three approaches but applications behind the security gateway are still not secure when information is exchanged between internal services. Therefore, it is needed to have a security policy enforcement technique that can be pluggable to the application and available immediately after starting the application. Aspect orientated programming [12] can provide such functionality, but some legacy systems may not support it. A model-based approach is ideal for reliability since it allows programmers to call upon the security controls during coding. Implementing model-based security enforcement is more complex than the other two approaches but the one presented in this document will simplify the process.

Also, the proposed approach aims to promote agile practices [13] [14] in software development. The three main roles, domain expert, programmer, and security expert, should interact directly or indirectly with the security policy enforcement mechanism. In addition, each role can be performed by different people and one person can do more than one roles depending the organizational structure and the scope and size of the application. Depending on the size of the application under development, more than one person may be needed for each one of the roles.

Domain experts such as business owners/managers can add security related user stories to the backlog using techniques such as in [15]. Then, security user stories should

be assigned to programmers along with security experts setting the security policies implementation details. With this practice, domain experts would consider security as a business priority and bring security requirements to an early stage of the software development life cycle. Similarly, programmers become aware of the security requirements and write code with security mechanisms in mind. As stated before, programmers do not need to be security experts and they only need to know when to apply the security policy enforcement.

Implementing security controls in the software applications alone will not guarantee the total protection of application and the data handled by it. Validation and verification steps will be needed to ensure the proper execution of the security controls. Testers and security savvy programmers can add security validation mechanisms based on the requirements, and further automate the testing execution to successfully validate the correct handling of the sensitive information by the application across the software development cycle. Techniques like test-driven development, acceptance test driven development and penetration testing will be necessary to be carried out from the early stages of the application development.

1.3 Summary

In this chapter we talked about the motivation of the research problem, what causes it, how it impacts to the industry as well as why it is worth to pursue a solution to address it. We also briefly described the model-based software approach that would be used to

solve the problem and why business domain and the requirements are important reference points to this technique.

Chapter 2

Background

2.1 Proposed Features

Many different approaches were reviewed to identify how they have addressed the problem of helping developers to apply security measures to their applications. Based on the literature review findings we have put together a list of the proposed feature that we consider that a robust solution should have. The feature list is the following:

- *Clear separation of roles*: the new approach shall clearly define the roles that team members will play when using it as well as their responsibilities and their interactions with the other roles.
- *Agility in the SDLC support*: the mechanism shall allow the easy implementation of requirements specified in user stories and support of a comprehensive testing technique for a simple understanding of the results by all the stake holders.
- *Application-level security*: security controls shall be implemented and executed within the boundary of the application under development.
- *Business domain independency*: the security mechanism shall be used to fulfill the implementation of any security requirement from any business domain in a software application.
- *Application domain mapping*: the approach shall allow the mapping of data fields described in the domain model of the application with security policy implementations.

- *Data in motion protection*: the mechanism shall facilitate the easy integration of security mechanisms in the software application to protect sensitive information when being transmitted to other computer systems.
- *Data at rest protection*: the mechanism shall provide the easy integration of security mechanisms in the software application to protect confidential data when being stored in the file system or databases.
- *Polyglot persistence protection*: sensitive data shall be safely handled by the application regardless of the system destination that will store the data.
- *Policy in XML format*: security policies shall be defined in a computer and human readable format like in XML.
- *Policy ontology*: security policies shall be defined in a structural way that they can be easily categorized to identify which one should be executed to perform the application business transaction in context.
- *In-process policy enforcement*: the security policies shall be executed within the boundary of the application execution to guarantee that data is properly handled before they leave the limit of the process control.
- *Pluggable to legacy code*: the approach shall be used in new application code development as well as be plugged easily into existing application code to implement security requirements additions.
- *GUI support*: the mechanism shall provide a graphical interface to create security policy definitions and map the data field members of the domain model classes to existing security policies.

2.2 Literature Review

Various security maturity models have been proposed to provide a template for integrating security practices into the business functions and goals of software systems. While the proposed approach to embed security mechanisms in software is different from these security maturity models, the goal to enforce security policies in applications is in-sync with these models in which security requirements and policies are identified in the early stage of the software development life cycle. Examples of security maturity models include OWASP's Software Assurance Maturity Model (OpenSAMM) [16], Build Security In Maturity Model (BSIMM) [17] and Microsoft's Security Development Lifecycle (MSDL) [18]. Both OpenSAMM and BSIMM map security practices into stages of software development. The goal is to incorporate security in software during its developmental stages instead of just testing for security vulnerabilities after the software being completed. MSDL provides a similar reference model and incorporates security practices into Microsoft's operating systems and programs. However, it is not clear if a model-based approach like the introduced in this document is developed to facilitate the process.

Approaches to embed security mechanisms in software development have been proposed in previous studies. Depending on how the policies are enforced and if they are integrated into the software, these works fall into one of the three categories: model-driven approach, aspect oriented programming, and security gateway. In model-driven approach, such as in [19], codes are generated automatically based on a model definition and then plugged into the software and served as an integral part of the application. One main