UNIVERSITI PUTRA MALAYSIA

A TOOL FOR MODELING SOFTWARE SECURITY REQUIREMENTS USING SECURITY PATTERNS

ZULFIKAR AHMED MAHER

FSKTM 2016 49

A TOOL FOR MODELING SOFTWARE SECURITY REQUIREMENTS USING SECURITY PATTERNS

By

ZULFIKAR AHMED MAHER

© COPYRIGHT UPM

Thesis Submitted to the School of Graduate Studies, Universiti Putra Malaysia, in Fulfillment of the Requirements for the Degree of Master of Science

October 2016

COPYRIGHT

All material contained within the thesis, including without limitation to text, logos, icons, photographs and all other artwork, is copyright material of Universiti Putra Malaysia unless otherwise stated. Use may be made of any material contained within the thesis for non-commercial purposes from the copyright holder. Commercial use of material may only be made with the express, prior, written permission of Universiti Putra Malaysia.

Copyright © Universiti Putra Malaysia

© COPYRIGHT UPM

DEDICATION

I would like to dedicate my thesis to my beloved parents for their love and unconditional support to me. Thank you for giving me a chance to prove and improve myself through all my walks of life. I love you.

© COPYRIGHT UPM

Abstract of thesis presented to the Senate of Universiti Putra Malaysia in fulfillment of the requirement for the Degree of Master of Science

A TOOL FOR MODELING SOFTWARE SECURITY REQUIREMENTS USING SECURITY PATTERNS

By

ZULFIKAR AHMED MAHER

October 2016

Chairman : Associate Professor Nor FazlidaMohdSani, PhD
Faculty : Computer Science and Information Technology

Security requirements of today's software systems are increasing and becoming complex. Software industry has well recognized that security should be incorporated at earlier stages of the software development. It is not easy for the programmers and developers to incorporate security in the software without proper expertise in it. For that reason different security patterns were proposed by the security experts for implementation of security by non-security experts. A security pattern provides well proven solution for the existing security problem in a specific context provided by the security experts. Security patterns usually are in textual format due to which they are often neglected at the design level. Security patterns do not constitute an intuitive solution that can be used by software designers because they are not useful without a systematic way to apply them. Security patterns lack comprehensive structure that conveys essential information inherent to security engineering (SE). This research presentsmethodology for presenting secure software requirements using Security Patterns that is tailored to meetthe needs of secure system development. In order to maximize comprehensibility, well-known notations of Unified Modeling Language (UML) is used to represent structuraland behavioral aspects of design. Only 13% of the papers published till 2015 involve tooling support for security patterns. To encounter this limitation, a methodology which focuses on the providing solution provided by the security pattern in the form of standard UML notations. As the proposed method results in an extension of Deployment diagram, it is named as Security Patterns Deployment Diagram (SPDD). It represents the solution provided by security patterns in standard UML graphical notation, which includes the compulsory elements of security patterns that are context, problem, actors, relations and solution including where attacks will be fended off in the early design stage of the software system in a single view. SPDD is proposed along with security modeling tool called SPDD Editor for modeling security pattern solution using proposed methodology. Security patterns research uses UML for modeling regardless of security patterns to be dealt with. It could be because UML is the most widely accepted formalism for the analysis and design of software. Therefore, itis considered as security pattern modeling method. This extension of deployment diagram provides a

© COPYRIGHT UPM

suitable way to define semantics for each solution provided by security pattern and allowing developers to easily understand software security requirements and their implementations in detail. A Plug-in for SeaMonster security designing tool has been developed to support the designing of the proposed diagram using Eclipse Graphical Modeling Framework (GMF) and Eclipse Graphical Editor Framework (GEF). The validation of SPDD has been done with the Hospital Information System (HIS) and E-Commerce System case studies.

An expert review was performed to verify the proposed methodology and proposed tool support. SPDD editor tool and both methods SPDD and Component based application (CBA) were also evaluated by three experts in the field. The expert review results showed positive results towards acceptance of SPDD method and tool. Experimental comparison with twenty participants was also performed to validate the effectiveness and to find out the better method in terms of designing solution provided by security patterns from the participant's point of view. The CBA method was selected to compare with proposed SPDD method because of the fact that most of the programmers and developers usually known to component diagram and there is no need to teach them its application and they can easily perform the tasks related to CBA method and also security pattern modeling application using CBA is previously proposed in literature. The experimental results from participants showed that there is a significant difference in designing threats and mitigation using SPDD editor in two methods. The SPDD method is used to design more threats and mitigation as compared to CBA method. By using proposed methodology and SPDD editor tool it is easier for the non-security expert to incorporate security at earlier stages of software development. It provides the facility of designing the security requirements in the architecture at design stage with incorporating expert knowledge of the security experts provided by the security patterns.

© COPYRIGHT UPM

Abstrak tesis yang dikemukakan kepada Senat Universiti Putra Malaysia sebagai memenuhi keperluan untuk Ijazah Master Sains

PERALATAN UNTUK KEPERLUAN KESELAMATAN PERMODELAN PERISIAN MENGGUNAKAN CORAK KESELAMATAN

Oleh

ZULFIKAR AHMED MAHER

Oktober 2016

Pengerusi : Profesor Madya Nor FazlidaMohdSani, PhD
Fakulti : Sains Komputer dan Teknologi Maklumat

Industri perisian telah mengenal pasti bahawa faktor keselamatan perlu diterapkan pada peringkat awal pembangunan perisian. Secara praktikal bukan mudah bagi pengaturcara untuk menggabungkan keselamatan dalam perisian tanpa memiliki kepakaran yang tepat mengenainya. Oleh sebab itu, corak keselamatan yang berbeza telah dicadangkan oleh pakar-pakar keselamatan bagi pelaksanaan sistem keselamatan kepada bukan pakar-pakar keselamatan. Corak keselamatan mampu memberi penyelesaian yang berkesan untuk masalah keselamatan yang sedia ada dalam konteks tertentu yang dicadangkan oleh pakar-pakar keselamatan. Kebiasaannya corak keselamatan dalam format teks menyebabkan ia sering diabaikan pada peringkat reka bentuk. Corak keselamatan bukan merupakan penyelesaian intuitif yang boleh digunakan oleh pereka perisian kerana ia tidak berkesan tanpa adanya cara yang sistematik untuk diaplikasikan.

Kekurangan pada corak keselamatan dapat mendedahkan maklumat penting kepada sesiapa yang tidak berkaitan. Oleh itu ,pihak keselamatan mencadangkan arahan untuk menghentikan penyalahgunaan maklumat yang khusus perlu digunakan dalam meningkatkan faktor keselamatan. Bagi menangani isu ini, kaedah metodologi lebih menjurus kepada penyelesaian merupakan corak keselamatan dalam bentuk Unified Modeling Language (UML) notasi standard. Kami mencadangkan lanjutan daripada rajah penggunaan UML dipanggil Rajah Penggunaan Corak Keselamatan (SPDD). Ini merupakan penyelesaian yang diberikan oleh corak keselamatan standard notasi UML grafik, yang merangkumi elemen wajib corak keselamatan iaitu konteks, masalah, pelakon, hubungan dan penyelesaian termasuk dapat menghalang serangan pada peringkat reka bentuk awal sistem perisian dalam paparan tunggal. Lanjutan rajah penggunaan menyediakan cara yang sesuai untuk menentukan semantik bagi setiap penyelesaian yang disediakan oleh corak keselamatan dan membolehkan pengguna dengan mudah memahami keperluan sistem keselamatan dan pelaksanaannya secara terperinci. *Plug-in* untuk alat keselamatan SeaMonster telah diwujudkan untuk menyokong aktiviti gambar rajah yang dicadangkan menggunakan Rangka Kerja

© COPYRIGHT UPM

Eclipse Grafik Model (GMF) dan Rangka Kerja Editor Grafik Eclipse (GEF). Perbandingan SPDD telah dilakukan dalam kajian Sistem Maklumat Hospital (HIS) dan Sistem E-Perdagangan.

Semekan pakar yang dilakukan untuk mengesahkan kaedah yang dicadangkan dan sokongan peralatan yang dicadangkan. Peralatan editorSPDD dan kedua-dua kaedah SPDD dan Aplikasi BerdasarkanKomponen (CBA) turut dinilai oleh tiga pakar dalam bidang ini. Keputusan semakanpakar menunjukkan hasil yang positif terhadap penerimaan kaedah SPDD dan peralatan. perbandingan eksperimen dengan dua puluh peserta juga telah dilakukan untuk mengesahkan keberkesanan dan untuk mengetahui kaedah yang lebih baik dari segi mereka bentuk penyelesaian yang disediakan oleh corak keselamatan dari sudut pandanganpeserta. Kaedah CBA telah dipilih untuk membuat perbandingan dengan kaedah SPDD yang dicadangkan kerana hakikat bahawa kebanyakan pengaturcara dan pembangun biasanya diketahuisebagai rajah komponen dan tidak ada keperluan untuk mengajar mereka mengaplikasinia dan mereka dengan mudah boleh melaksanakan tugas-tugas yang berkaitan dengan kaedah CBA dan juga aplikasi pemodelancorak keselamatan menggunakan CBA adalah sebelum ini dicadangkan dalam kesusasteraan. Keputusan eksperimen daripada peserta menunjukkan bahawa terdapat perbezaan yang signifikan dalam mereka bentuk ancaman dan penguranganya menggunakan editorSPDD dalam dua kaedah. Kaedah SPDD digunakan untuk mereka bentuk lebih ancaman dan pengurangan berbanding kaedah CBA. Dengan menggunakan kaedah yang dicadangkan dan peralatan editor SPDD ia adalah lebih mudah untuk bukanpakar keselamatan untuk menggabungkan keselamatan di peringkat awal pembangunan perisian. Ia menyediakan kemudahan mereka bentuk keperluan keselamatan dalam seni bina pada peringkat reka bentuk dengan menggabungkan pengetahuan pakar daripada pakar-pakar keselamatan yang disediakan oleh corak keselamatan.

© COPYRIGHT UPM

ACKNOWLEDGEMENTS

First and foremost praises and thanks to the Almighty Allah, for his showers of blessings and help throughout my life and especially in accomplishing this research successfully.

This work would have been impossible without the continuous support and supervision of my supervisor, Associate Professor Dr. Nor Fazlida Mohd Sani. All steps taken on the way to finish this thesis were under her direct guidance and also the other members of my supervisory committee, Dr. Jamilah Din and Associate Professor Dr. Marzanah A. Jabar. I am also thankful to Dr. Novia Indriaty Admodisastro for helping me as expert reviewer of my work and provide valuable guidelines to improve my research.

I would like to express my full gratitude to my beloved wife who helped me all the time and have to live away without me from last two years. Similarly many thanks to my parents, sisters, brother and my kids Muhammad Amin and Yashfeen Amna, who supported me with their prayers and encouragements and they endured the pain of being away for more than 4 years. I am also thankful to all my friends for their constant support.

I am also greatly indebted to the financial support of Sindh Agriculture University, Tandojam, Pakistan for my MS studies at Universiti Putra Malaysia.

© COPYRIGHT UPM

I certify that a Thesis Examination Committee has met on 28 October 2016 to conduct the final examination of Zulfikar Ahmed Maher on his thesis entitled "A Tool for Modeling Software Security Requirements using Security Patterns" in accordance with the Universities and University Colleges Act 1971 and the Constitution of the Universiti Putra Malaysia [P.U.(A) 106] 15 March 1998. The Committee recommends that the student be awarded the Master of Science.

Members of the Thesis Examination Committee were as follows:

Nur Izura binti Udzir, PhD
Associate Professor
Faculty of Computer Science and Information Technology
Universiti Putra Malaysia
(Chairman)

Rusli bin Hj. Abdullah, PhD
Professor
Faculty of Computer Science and Information Technology
Universiti Putra Malaysia
(Internal Examiner)

Shamsul Sahibuddin, PhD
Professor
University Technology Malaysia
Malaysia
(External Examiner)

NOR AINI AB. SHUKOR, PhD
Professor and Deputy Dean
School of Graduate Studies
Universiti Putra Malaysia

Date: 22 March 2017

This thesis was submitted to the Senate of the Universiti Putra Malaysia and has been accepted as fulfillment of the requirement for the degree of Master of Science. The members of the Supervisory Committee were as follows:

Nor Fazlida Mohd Sani, PhD
Associate Professor
Faculty of Computer Science and Information Technology
Universiti Putra Malaysia
(Chairman)

Marzanah A. Jabar, PhD
Associate Professor
Faculty of Computer Science and Information Technology
Universiti Putra Malaysia
(Member)

Jamilah Din, PhD
Senior Lecturer
Faculty of Computer Science and Information Technology
Universiti Putra Malaysia
(Member)

ROBIAH BINTI YUNUS, PhD
Professor and Dean
School of Graduate Studies
Universiti Putra Malaysia

Date:

© COPYRIGHT UPM

Declaration by graduate student

I hereby confirm that:

- this thesis is my original work;
- quotations, illustrations and citations have been duly referenced;
- this thesis has not been submitted previously or concurrently for any other degree at any institutions;
- intellectual property from the thesis and copyright of thesis are fully-owned by Universiti Putra Malaysia, as according to the Universiti Putra Malaysia (Research) Rules 2012;
- written permission must be obtained from supervisor and the office of Deputy Vice-Chancellor (Research and innovation) before thesis is published (in the form of written, printed or in electronic form) including books, journals, modules, proceedings, popular writings, seminar papers, manuscripts, posters, reports, lecture notes, learning modules or any other materials as stated in the Universiti Putra Malaysia (Research) Rules 2012;
- there is no plagiarism or data falsification/fabrication in the thesis, and scholarly integrity is upheld as according to the Universiti Putra Malaysia (Graduate Studies) Rules 2003 (Revision 2012-2013) and the Universiti Putra Malaysia (Research) Rules 2012. The thesis has undergone plagiarism detection software

Signature: _____ Date: _____

Name and Matric No: Zulfikar Ahmed Maher, GS31877

© COPYRIGHT UPM

Declaration by Members of Supervisory Committee

This is to confirm that:

- the research conducted and the writing of this thesis was under our supervision;
- supervision responsibilities as stated in the Universiti Putra Malaysia (Graduate Studies) Rules 2003 (Revision 2012-2013) were adhered to.

Signature: _____
Name of Chairman of Supervisory Committee: Associate Professor Dr. Nor FazlidaMohdSani

Signature: _____
Name of Member of Supervisory Committee: Associate Professor Dr. Marzanah A. Jabar

Signature: _____
Name of Member of Supervisory Committee: Dr. Jamilah Din

© COPYRIGHT UPM

TABLE OF CONTENTS

© COPYRIGHT UPM

© COPYRIGHT UPM

© COPYRIGHT UPM

LIST OF TABLES

© COPYRIGHT UPM

LIST OF FIGURES

© COPYRIGHT UPM

© COPYRIGHT UPM

LIST OF ABBREVIATIONS

CBA	Component Based Application
ESRMG	Enterprise Security and Risk Management Grammer
EMF	Eclipse Modeling Framework
GEF	Graphical Editor Framework
GMF	Graphical Modeling Framework
GOOCA	Generic Object-Oriented Cryptographic Architecture
HIS	Hospital Information System
ISDF	Institute Sûreté de Fonctionnemen
IU	Intension to Use
KAOS	Knowledgeable Agent-Oriented System
MDA	Model Drive Architecture
MDE	Model Drive Engineering
NIST	National Institute of Standards and Technology
OMG	Object Management Group
PEU	Perceived Ease of Use
PloP	Pattern Languages of Programs
PMR	Primary Medical Record
PU	Perceived Usefulness
SAS	Statistical Analysis System
SCRIP	Security Pattern Integration Process
SD	Software Development
SDLC	Software Development Life Cycle
SE	Software Engineering
SP	Security Patterns

© COPYRIGHT UPM

SPDD	Security Pattern Deployment Diagram
SPSS	Statistical Package for the Social Sciences
SQL	Structured Query Language
SSD	Secure Software Development
SSS	Secure Software System
TESEM	Test Driven Secure Modeling Tool
RBAC	Role Based Access Control
RPC	Remote Procedure Call
UML	Unified Modeling Language

© COPYRIGHT UPM

CHAPTER 1

INTRODUCTION

1.1 Overview

Traditionally, security in software development life cycle (SDLC) is not considered at early stages, usually it is incorporated at later stages. Realizing security at later stages of software development (SD) results in increased risks of occurring security flaws. Fixing system risksand vulnerabilities after SDcost high for developers and users.There are many best practices are available to address these issues, but often are difficult to reuse due to their implementation-specific nature. Furthermore, understanding of the root causes of security flaws in detail has led to a greater importance of security into design phase. For this reason security patterns were proposed by security experts for implementing security in the software system by the non-security experts and for those with least expertise in security implementation.

Security pattern provides well proven documented description of a solution for the recurring security problem (Bouaziz&Kammoun, 2016) in a specific context in the textual format by the security experts. It is often neglected by the software developer because of the lack of the guidance provided for their concrete application(Bouaziz and Coulette, 2012). Security patterns alone are not sufficient because of thelack of providing systematic guidelines in order to allow their easy application. That is why at the design level these patterns are often neglected and unable to provide clear solution that can be used by software developers(Bouaziz and Coulette, 2012). Considering this situation, using standard modeling languages such as UML (Unified Modeling Language) for providing solution gathered by security experts in a security patterns is helpful for both software architects for designing a secure architecture as well for the software developers to understand the solution provided by security pattern.

1.1.1 Software security andsecurity patterns

Security is a non-functional property that software developers have to implement during the SD. The security requirements of today's software systems are increasing day by day and it is not easy for software developers to incorporate security in the software without proper expertise in security. During development of software, faults and flaws are introduced either from the implementation or from the design of the software. During runtime, these faults and flaws can propagate into failures that can result in vulnerabilities. Security flaws and presence of vulnerabilities needs developers spend more time on maintenance instead of new features and also increases the total cost. Typically software developers are experts in the functional requirements with a minimal security knowledge, which causes weak security decisions (Mourad*et al.,*2010). Security patterns were proposed for this reason by security experts so that non security experts can implement security in software system with least amount of expertise in security.

© COPYRIGHT UPM

A security pattern provides well proven documented description of a solution for the recurring security problem in a specific context provided by the security experts (Kim*et al*., 2006; Lincke 2012). Generally,it can be said that a security pattern is a pattern with focus on the security implementation of a system. The main characteristic of a security pattern is a solution of a problem which occurs in a specific context (Lincke, 2012). Therefore, a security pattern does not only provide a solution but it also includes a context and a problem for which it should be used.

1.2 Research background and motivation

A report from Software Engineering Institute's CERT Coordination Centre showed that number of application vulnerabilities increased from 171 to5990 during the period of 1995 to 2005(Kim *et al.,*2006). Another reports presented by National Institute of Standards and Technology (NIST) described that 59.5 billion U.S dollars were cost on repairs of faulty software and breakdowns on security and reliability each year(Haley *et al*., 2008). Above facts showed that security and reliability needs immediate attention and needs major improvements. Therefore, security need to be considered at early stages of development otherwise it will be very expensive and difficult to considerably improve it on deployed software(Lincke*et al.,*2012). Secure software engineering (Secure SE) aims to avoid these flaws in SD by considering security aspects from the very beginning of SDLC. Requirements engineering is the area which provide foundation for developing quality software. Security requirements elicitation plays a central role in requirement engineering process; it allows secure SD by providing security requirements at early design stage.

Different techniques for security requirement elicitation has been proposed in literature, among these techniques Misuse case diagrams (MCD)(Sindre and Opdahl, 2005)are widely accepted for eliciting negative scenario based security requirements by modeling the possible future attacks to the system. MCD have been proven useful in providing the image of vulnerable attacks to new software, during the requirements stage. The major problem in using MCD is that their outputs can be very lengthy, which makes themdifficult to understand and hard to analyze(Rizzi, 2003; Mourad*et al.,*2010). It has a wide range of application possibilities (Brandozzi and Perry, 2001; Ren and Taylor, 2005; Karpati*et al.,*2010) as general rather than being a specific technique. It is an open-ended method, so the results are very much dependent on the modeler's creativity(Ren and Taylor, 2005). Misuse case diagrams provide image of future attacks to the system but where these attacks are handled is missing in the diagram(Lincke*et al.,*2012). MCDs are good at defining security threats or attacks, but unable to define where security should be implemented(Lincke*et al.,*2012).

Along with defining expected security threats to software with the help of MCD, every security concerned enterprise selects its own security measures in order to avoid unexpected events and accidents. The objective of all the security measures is to protect the enterprise's own resources and assets from damage. Most of the time, the accidents or disasters take place in enterprise are similar in nature, and are caused by similar kind of vulnerabilities. However, many security analysts find it difficult to select the right security measure for a particular problem because the previous proven

© COPYRIGHT UPM

solutions are not properly documented. In this context Security Patterns could be helpful since they present the proven solutions that potentially could be reused in the similar situations(Bouaziz and Coulette, 2012).

Schumacher *et al.*(2013) reported that security patternsare well identified solution for the information security problem. They present a solution to a security problem or threat by the knowledge accumulated by security experts for security of a software system. A large number of security patterns have been proposed, but they are generally without guiding developers for their application details(Bouaziz and Coulette, 2012). Security patterns alone are not sufficient(Fernandez, 2009) because they do not provide systematic guidelines in order to allow their easy application. This is the main reason for neglecting of security patterns at design level and does not provide a solution that can easily be used by software designers(Bouaziz and Coulette, 2012). Security patterns have some limitations such as security patterns do not provide guidance to indicate how specific misuses of information can be stopped also do not indicate when they should be applied along the SDLC. Unnecessary security patterns may increase overhead and complexity.

Mostly requirements engineers are untrained and not good at security at all, and those few who are trained have only been given an overview of security mechanisms such as encryption and passwords rather than training them about actual security requirements (Salini and Kanmani, 2010). As a result, methodology which guides developers with the capabilities to apply the security patterns with minimal security expertise is becoming a very challenging issue in this domain of research(Mourad*et al.,*2010).

In order to enhance the usefulness ofMCD and to make security patterns techniques more applicablefor clear elicitationof security requirements at the earliest stage of SDLC, this study present an efficient security requirement elicitation and security threat modeling method. The suggested method incorporates security patterns with MCD and deployment diagrams to address their limitations. This study is beneficial to clearly identify the security requirements and also the mitigating strategy at early stage of SD. The proposed method is generic in nature which allows developing more secure software systems more efficiently.

1.3 Problem statement

Software developers are generally not security experts (Bouaziz&Kammoun, 2016) and face difficult time to deploy security constraints(Vieira and Antunes, 2013).Security mechanisms are complex and it is difficult for the average developer to understand how to fulfill the security requirements of these mechanisms and how to achieve the goal of secure implementations. Software developers are not necessarilysecurity experts, identifying potential threats and vulnerabilitiesin the early stage of the development process is difficult for them Kobashi*et al*. (2015). No clear solution has been provided for these challenges(Fernandez, 2009). Major difficulty in integrating security at SDphase is the selection of security mechanisms to be used,

© COPYRIGHT UPM

secondly where these mechanisms are applied in the system and lastly at what level of abstraction is needed for application of these mechanisms (Bouazi*et al.,*2011). Developers need concrete guidelines for constructing secure applications(Lodderstedt*et al.,*2002). Concrete in a sense that it should include how different security attacks can be mitigated across system is also important(Lincke*et al.,* 2012).Vysoky(2012) discussed that there is a lack of tools to exactly model and analyze the system and through which it is possible to detect potential threats and impacts of attacks from users.

Security has been integrated into UML diagrams using various techniques such asmis-sequence diagrams, security patterns and packages state diagrams.However thesetechniques are unable to show the concrete security deployment technique, which should include how these attacks can be handled in the systems. For example, to mitigate SQL attack, the input validation must occur. If this validation only occurs at client side, the security mechanism is inadequate. Therefore, the location of security code in the system is as important as its existence(Lincke*et al.,*2012). Different methods are presented for dealing security requirements at early SD phases, but no solution addresses security requirements in relation to design of secure architectures (Howard and Lipner, 2009). Most of the security patterns research uses UML for modeling regardless of security patterns to be dealt with. It could be because UML is the most widely accepted formalism for the analysis and design of software. Therefore, UML is considered as security pattern modeling method (Ito*et al*., 2015).

Many studies on security patterns are presented in literature and recently it has received much needed attention as a solution for capturing security solutions. A lot of security patterns are proposed without guiding developers for their concrete application(Bouaziz and Coulette, 2012). Security patterns alone are not sufficient because they do not provide systematic guidelines for their easy application. That is why at the design level these patterns are often neglected and unable to provide clear solution that can be used by software designers(Bouaziz and Coulette, 2012). Security patterns possess some limitations such as they lack directions to stop the specific misuses of information for the developer and also do not specify when they should be applied along the SDLC, whichresults in increased complexity in the systemdue to an unnecessary use of a security patterns. Modeling methods of security are uncertain and demand for efficient and reliable techniques to applying SPs is high(Ito*et al*., 2015).

Security patterns have two major deficiencieswhile using for design of secure software systems (SSS)(Horvath and Dörges, 2008). Firstly,they are informal descriptions like design patterns and explain what to do and secondly, they are not suitable for describing complex architectures. One more major problem with security patterns is that they are expressed in a textual format and does not include sufficient descriptions for their method and to extend their use(Hamid *et al.,*2010).

UML diagrams are commonly used to plan and build software systems based on the Object-Oriented approach. These diagrams allow to understand the system architecture and implementation details, as well as system functioning. When a system

© COPYRIGHT UPM

securityanalysis is performed, many aspects of the system are considered such as, operation, functioning, data flow, data types, architecture and implementation details must be well known and modeled in order to determine possible weak points for the system security. The various UML diagrams supply all the information needed for a security system analysis and many aspects of the UML methodology can be applied for the same purpose. UML diagrams can be used efficiently for security system environment analysis (Rachel*et al*.,2006). There are many previous studies present in the literature which focus on security using UML diagrams. UML is the most widely accepted formalism for the analysis and design of software. Therefore, UML is considered as security pattern modeling method (Ito *et al*., 2015). Using the UML diagrams to model security specification and mitigation will have advantages such as it will not require new set of semantics and notations. Using other modeling languageshave some disadvantages such as (i) new notations and semantics will not be compatible with available commercial tools, (ii)developers need to learn new semantics and notations to understand. These disadvantages justify the choice of using standardized UML diagrams for encountering the security requirement problem in SD industry.

1.4 Research Questions

In this research work, there are three research questions which help to achieve the research objectives. These questions will also help to understand the overall purpose and contribution of this research.

RQ 1: How to model security requirements of a software system?
This questions stands as the main research problem of this thesis. In literature review part different methods of modeling security requirements has been discussed. As an answer to this question, how business security analyst and developers could use Security Patterns to model security requirements will be described.

RQ 2: How to model security requirements of a software system using Security Patterns?
As an answer to this question, this research focuses on developing a new approach for modeling software security requirements using security patterns.

RQ 3: How to evaluate the applicability of the proposed methodology and tool in the industry ?
This question seeks the answer regarding the applicabilityof proposed SPDD methodology and tool. The answer will be presentedin the results of expert review and experimental comparison between proposed and existing method.

1.5 Research objectives

1. To propose a method for modeling secure requirements for designing software.
2. To evaluate the effectiveness of proposed method and tool for designing secure software requirements

© COPYRIGHT UPM

1.6 Proposed solution

To encounter the imitations of security patterns such as lack of suggesting directions to stop the specific misuses of information and fail to describe when these patterns should be applied along the development lifecycle, Security Patterns Deployment Diagram (SPDD) is proposed.SPDD combines the power of security patterns for providing proven solutions gathered by security experts to secure a software system. A SPDD can show threat, actors, relations and where attacks will be fended off in the early design stage of the system in a single view. A SPDD is used at requirements stage to show where these attacks should be addressed in the system.

As patterns are often neglected at the design level and do not constitute an intuitive solution that can be used by software designers because they are not very useful without a systematic way to apply(Fernandez*et al.*2009). In order to provide designers with guidelines, a secure software design using SPDD is proposed which shows in a single picture how software is deployed including the security implementation provided by security patterns. The proposed solution will improve the expression power of security patterns by providing their visually illustration (design) in the form of SPDD.

1.7 Significance andscope of the study

The result of the study will provides a suitable way to define semantics for each solution proposed by security pattern. It allows developers to easily understand security requirements of the systems and their implementations detail with the help of Security Patterns Deployment Diagram. According to the literaturenot much work has been done in the past to visualize the deployment or implementation of security patterns to help the developers for better understanding of their security implementation.

In contrast to the previous studies on security elicitation, proposed methodology addresses security requirements separately from the functional requirements by combining the security knowledge from experts, in the form of security patterns with the UML Diagrams. Most of the security patterns research uses UML for modeling regardless of security patterns to be dealt with. It could be because UML is the most widely accepted formalism for the analysis and design of software. Therefore, UML is considered as security pattern modeling method.Our contribution includes adapting proposed security patterns that capture security knowledge and integrate these patterns to UML models for better understanding of security implementation by the developers. It is a systematic approach which deals with the security from the beginning.Realizing the elicited security requirements by security patterns on design artifacts contributes to reduce architectural flaws. A systematic approach is provided which (i) deals with security from the beginning (ii) realize the elicited security requirements by security patterns on design artifacts and (iii) Contributes to reduced architectural flaws in security implementation.

© COPYRIGHT UPM

1.8 Thesis organization

The rest of the thesis is mainly isconsisting four chapters. Each chapter has its sections relevant to the topics discussed in that chapter.Chapter 2 surveys existing research in the area of security patterns. The review explores the state-of-art and practices in their representation and identifies gaps and need for a diagram which visualize the solution provided by a security pattern. It explores how can potentially leverage on the state-of-art.Chapter 3 discussed the methodology used to resolve the problem identified in Security Pattern representation.

Chapter 4 consists of the data collection and analysis method used for comparison of the proposed method with the existing method for representation of security patterns. Tool development process was also discussed. A case study is performed using the developed tool to implement proposed method using identified security patterns for its usage validation. Chapter 5 discussed the finding of the experimental comparison and results of the study however Chapter 6 consists of conclusion drawn from this thesis and future recommendation.

© COPYRIGHT UPM

REFERENCES

Alexander C. 1977. *A pattern language: towns, buildings, construction*. Vol. 2: Oxford University Press.

Alkussayer, A., & Allen, W. H. 2009. The ISDF Framework: Integrating security patterns and best practices. SDF framework: Integrating security patterns and best practices. In *International Conference on Information Security and Assurance*, 17-28.

Araujo, I., & Weiss, M. 2002. Linking patterns and non-functional requirements. Proceedings of the *Ninth conference on pattern language of programs (PLOP 2002)*.

Bandara, A., Shinpei, H., Jurjens, J., Kaiya, H., Kubo, A., Laney, R., Mouratidis, H., Nhlabatsi, A., Nuseibeh, B., Tahara, Y. & Tun. 2010. "Security patterns: Comparing modeling approaches"*Technical Report*,75-111.

Bass, L. 2007. *Software architecture in practic*, Pearson Education India.

Benameur, A., Fenet, S., Saidane, A. & Sinha, S.K. 2009. A Pattern-Based General Security Framework: An eBusiness Case Study. In *High Performance Computing and Communications, 2009. HPCC'09*, 339-346.

Blakley, B, & C Heath.2004.*Technical Guide: Security Design Patterns*. The Open Group.

Blakley, B, & C Heath.2004. "Members of the Open Group Security Forum." Security Design Patterns, *Open Group Technical Guide*.

Bouaziz, R., & Coulette, B. 2012. Secure component based applications through security patterns. *IEEE International Conference on Green Computing and Communications*.

Bouaziz, R., Hamid, B., & Desnos, N. 2011. Towards a better integration of patterns in secure component-based systems design. In *Computational Science and Its Applications (ICCSA 2011)*, 607-621. Springer.

Bouaziz, R., Kallel, S., & Coulette, B. 2014. An approach for security patterns application in component based models. In *Computational Science and Its Applications–ICCSA 2014*, 283-296.

Bouaziz, R., &Kammoun, S. 2016. SCRIStUDIO: A security pattern integration tool. *International Conference on Information Technology for Organizations Development (IT4OD)*, 1-6.

Braga, A., Rubira, C., & Dahab, R. 1999. Tropyc: A pattern language for cryptographic software.In: Proc. *5th Conference on Pattern Languages of Programs (PLoP)* Monticello, IL, USA.

© COPYRIGHT UPM

Brandozzi, M. & Perry, D.E. 2001. Transforming Goal-Oriented Requirement Specifications into Architecture Prescriptions. In Workshop *from Soft. Req. to Arch*, 54-61.

Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F., & Mylopoulos, J. 2004. Tropos: An agent-oriented software development methodology. *Autonomous Agents and Multi-Agent Systems*, 8(3):203-236.

Brown, F.L., DiVietri, J., de Villegas, G.D. & Fernandez, E.B. 1999, August. The authenticator pattern. In Proceedings of *Pattern Language of Programs*, 15-18.

Chris, V., T. Nikos, & F. Mark, 2002.Case research in operations management.*International Journal of Operations & Production Management*. 22(2): 195.

Cotroneo, D. ed. 2013. *Innovative technologies for dependable ots-based critical systems*, Springer.

Da Silva Junior, L.S., Guéhéneuc, Y.G. & Mullins, J. 2013. An approach to formalise security patterns. *Technocal report*, École Polytechnique de Montréal, Montréal Quebéc.

Darimont, Robert, Emmanuelle Delor, Philippe Massonet, & Axel van Lamsweerde. 1997. "GRAIL/KAOS: an environment for goal-driven requirements engineering." In Proceedings of the *19th international conference on Software engineering*, 612-613.

Davis, F.D., 1989. Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS quarterly*,319-340.

De Win, B., Scandariato, R., Buyens, K., Grégoire, J., & Joosen, W. 2009. On the secure software development process: CLASP, SDL and Touchpoints compared. *Information and software technology*,51(7):1152-1171.

Dougherty, C. R., Sayre, K., Seacord, R., Svoboda, D., & Togashi, K. 2009. Secure design patterns, Carnegie Mellon University, Software Engineering Institute, *TECHNICAL REPORT CMU/SEI 2009-TR-010*, Available: www.cert.org/archive/pdf/09tr010.pd

Essmayr, W., Pernul, G. & Tjoa, A.M. 1998. Access controls by object-oriented concepts. *Database Security XI*, 325-340.

Fernandez, E.B. 2000. Metadata and authorization patterns. Departament of Computer Science and Eng., *Technical report*, Florida Atlantic University TR-CSE-00-16.

Fernandez, E.B. 2009.Security patterns and a methodology to apply them. *In Security and Dependability for Ambient Intelligence*, 37-46. Springer.

© COPYRIGHT UPM

Fernandez, E. B., Ballesteros, J., Desouza-Doucet, A. C., & Larrondo-Petrie, M. M. 2007. Security patterns for physical access control systems. In *IFIP Annual Conference on Data and Applications Security and Privacy*, 259-274.

Fernandez, E. B., Larrondo-Petrie, M. M., & Gudes, E. 1994. A method-based authorization model for object-oriented databases. *Security for Object-Oriented Systems*, 135-150. Springer.

Fernandez, E. B & R. Pan. 2001. A pattern language for se-curity models. In *8th Conference on Pattern Languages of Programs (PLoP)*.

Firesmith, D. 2007. Common Requirements Problems, their negative consequences, and the industry best practices to help solve them. *Journal of Object Technology*, 6(1): 17-33.

Gorrieri, A. A. R., & Martinelli, F. 2005. *Foundations of Security Analysis and Design III*,Springer.

Graham, I. 2007. *Business rules management and service oriented architecture: a pattern language*: John Wiley & Sons.

Hafiz, M. 2013. Security pattern catalog. Available www.munawarhafiz.com/securitypatterncatalog/index.php

Hafiz, M., Adamczyk, P., & Johnson, R. E. 2007. "Organizing security patterns." *IEEE software*, (4):52-60.

Haley, C., Laney, R., Moffett, J. & Nuseibeh, B., 2008. Security requirements engineering: A framework for representation and analysis. *IEEE Transactions on Software Engineering*, 34(1): 133-153.

Hall, A., & Chapman, R. 2002. Correctness by construction: Developing a commercial secure system. *Software*, IEEE, 19(1):18-25.

Hallowell, M. R., & Gambatese, J. A. 2010. Qualitative research: application of the Delphi method to CEM research. *Journal of construction engineering and management*, 136(1): 99-107.

Hamid, B., Desnos, N., Grepet, C. & Jouvray, C. 2010. September. Model-based security and dependability patterns in RCES: the TERESA approach. In Proceedings of the *International Workshop on Security and Dependability for Resource Constrained Embedded Systems*, 8. ACM.

Hays, V., Loutrel, M., & Fernandez, E. B. 2000. The object filter and access control framework. *Conference on Pattern Languages of Programs (PLoP)*, 12-17.

Heyman, T., Yskout, K., Scandariato, R., Schmidt, H., & Yu, Y. 2011. The security twin peaks. In *International Symposium on Engineering Secure Software and Systems*, 167-180.

© COPYRIGHT UPM

Hogg, J. 2006. Web service security: *Scenarios, patterns, and implementation guidance for Web Services Enhancements (WSE) 3.0*, O'Reilly Media, Inc..

Horvath, V., & Dörges, T. 2008. From security patterns to implementation using petri nets. Proceedings of the *fourth international workshop on Software engineering for secure systems*.

Howard, M., & Lipner, S. 2009. *The security development lifecycle*, O'Reilly Media, Incorporated.

Humphrey, Watts S. 2000. *Introduction to the team software process (sm)*, Addison-Wesley Professional.

Infrastructure, P. K., & Profile, T. P. 2002. Common criteria for information technology security evaluation. National Security Agency.

Ito, Y., Washizaki, H., Yoshizawa, M., Fukazawa, Y., Okubo, T., Kaiya, H., Hazeyama, A., Yoshioka, N. & Fernandez, E.B. 2015. Systematic Mapping of Security Patterns Research. *In Proceedings of the 22nd Conference on Pattern Languages of Programs Conference(PLoP)*.

Jackson, M. 2001. *Problem frames: analysing and structuring software development problems,* Addison-Wesley.

Jacobson, I. 1993. *Object-oriented software engineering: a use case driven approac,*. Pearson Education India.

Jürjens, J. 2002. September. UMLsec: Extending UML for secure systems development. In *International Conference on The Unified Modeling Language*, 412-425. Springer Berlin Heidelberg.

Karpati, P., Redda, Y., Opdahl, A. L., & Sindre, G.2014. Comparing attack trees and misuse cases in an industrial setting.*Information and software technology*, 56(3):294-308.

Karpati, P., Sindre, G., & Opdahl, A. L. 2010. Visualizing cyber attacks with misuse case maps.*Requirements Engineering: Foundation for Software Quality*, 262-275. Springer.

Keblawi, F., & Sullivan, D. 2006. Applying the common criteria in systems engineering. *Security & Privacy*, IEEE 4(2):50-55.

Kienzle, D. M., Elder, M. C., Tyree, D., & Edwards-Hewitt, J. 2002. "Security patterns repository version 1.0." DARPA, Washington DC.

Kim, J., Kim, M., & Park, S. 2006. Goal and scenario based domain requirements analysis environment. *Journal of Systems and Software*, 79(7):926-938.

© COPYRIGHT UPM

Kobashi, T., Yoshizawa, M., Washizaki, H., Fukazawa, Y., Yoshioka, N., Okubo, T. & Kaiya, H. 2015. TESEM: A Tool for Verifying Security Design Pattern Applications by Model Testing. In *8th International Conference on Software Testing, Verification and Validation (ICST1)*,8. IEEE.

Konrad, S., Cheng, B. H., Campbell, L. A., & Wassermann, R. 2003. Using security patterns to model and analyze security requirements. *International Conference on Requirements Engineering (RE)*.IEEE.

Laverdiere, M.A., Mourad, A., Hanna, A. & Debbabi, M. 2006. Security design patterns: Survey and evaluation. *Canadian Conference on Electrical and Computer Engineering*, 1605-1608. IEEE.

Li, T., Horkoff, J., & Mylopoulos, J. 2014. Integrating security patterns with security requirements analysis using contextual goal models. *The Practice of Enterprise Modeling*, 208-223. Springer.

Li, T., & Mylopoulos, J. 2014. Modeling and applying security patterns using contextual goal models. *The 7th International i* Workshop, iStar14*.

Lin, L. C., Nuseibeh, B., Ince, D., Jackson, M., & Moffett, J. 2003a. Analysing security threats and vulnerabilities using abuse frames. *European Joint Conferences on Theory and Practice of Software (ETAPS-04)*.

Lin, L., Nuseibeh, B., Ince, D., Jackson, M., & Moffett, J. 2003b. Introducing abuse frames for analysing security requirements. In *Requirements Engineering Conference,* 371-372. IEEE.

Lincke, Susan J. 2012. Designing software security with UML extensions: post-conference workshop. *Journal of Computing Sciences in Colleges*, 28(1):149-152.

Lincke, S.J., Knautz, T.H. & Lowery, M.D. 2012. Designing system security with UML misuse deployment diagrams. In *Sixth International Conference on Software Security and Reliability Companion (SERE-C)*, 57-61. IEEE.

Lodderstedt, T., Basin, D. & Doser, J. 2002. SecureUML: A UML-based modeling language for model-driven security. In *International Conference on the Unified Modeling Language*, 426-441. Springer Berlin Heidelberg.

Luján-Mora, S. & Trujillo, J. 2004. Physical modeling of data warehouses using UML. In Proceedings of the 7th ACM *Iinternational workshop on Data warehousing and OLAP*, 48-57.

Massacci, F., Mylopoulos, J., & Zannone, N. 2007. An ontology for secure socio-technical systems. *Handbook of ontologies for business interaction*, 1:469.

McGraw, G. 2006. *Software security: building security. Vol. 1*, Addison-Wesley Professional.

© COPYRIGHT UPM

Meland, P. H., Spampinato, D. G., Hagen, E., Baadshaug, E. T., Krister, K. M., & Velle, K. S. 2008. "SeaMonster: Providing tool support for security modeling." *Norsk informasjonssikkerhetskonferanse, NISK*.

Mellado, D., Fernández-Medina, E., & Piattini, M. 2007. A common criteria based security requirements engineering process for the development of secure information systems *Computer standards & interfaces*, 29(2):244-253.

Mellado, D., Blanco, C., Sánchez, L.E. &Fernández-Medina, E. 2010.A systematic review of security requirements engineering.*Computer Standards and Interfaces*, 32(4): 153-165.

Memon, M., Menghwar, G. D., Jalbani, A. A., Depar, M. H., & Rahu, G. A. 2012. Modeling Authentication and Authorization Security Service for WS-Security Architecture *Science International*, 24(3):243-249.

Meredith, J.,1998. Building operations management theory through case and field research. *Journal of Operations Management*, 16(4): 441.

Mourad, A., Otrok, H. and Baajour, L. 2010. A novel approach for the development & deployment of security patterns. *In Second International Conference on Social Computing (SocialCom)*, 914-919. IEEE.

Mouratidis, H., & Giorgini, P. 2007. Secure tropos: a security-oriented extension of the tropos methodology. *International Journal of Software Engineering and Knowledge Engineering*, 17(02):285-309.

Mouratidis, H., Weiss, M., & Giorgini, P. 2006. Modeling secure systems using an agent-oriented approach and security patterns.*International Journal of Software Engineering and Knowledge Engineering*, 16(03):471-498.

Nhlabatsi, A., Bandara, A., Hayashi, S., Haley, C.B., Jurjens, J., Kaiya, H., Kubo, A., Laney, R., Mouratidis, H., Nuseibeh, B. & Tun, T.T. 2010. Security Patterns: Comparing Modeling Approaches. *Software Engineering for Secure Systems: Industrial and Research Perspectives*, 75.

Ortiz, R., Moral-García, S., Moral-Rubio, S., Vela, B., Garzás, J., & Fernández-Medina, E. 2010. Applicability of security patterns. *On the Move to Meaningful Internet Systems (OTM 2010)*, 672-684. Springer.

Ren, J., & Taylor, R. 2005. A secure software architecture description language. In *Workshop on Software Security Assurance Tools, Techniques, and Metrics,* 82-89.

Richards, M. 2015. *Software architecture patterns*. O'Reilly Media.

Rizzi. 2003. Open problems in data warehousing: eight years later. In Proceedings of the *5th International Workshop on Design and Management of Data Warehouses(DMDW'03)* , Berlin, Germany.

© COPYRIGHT UPM

Rowe, G., & Wright, G. 1999. The Delphi technique as a forecasting tool: issues and analysis. *International Journal of Forecasting*, 15(4): 353-375.

Rozanski, N., & Woods, E. 2011. *Software systems architecture: working with stakeholders using viewpoints and perspectives*: Addison-Wesley.

Salini, P, & S Kanmani. 2010. A model based security requirements engineering framework.*International Journal of Computer Engineering and Technology*, 1 (1):180-195.

Schmidt, H. 2010. A Pattern and Component-Based Method to Develop Secure Software" (PhD Thesis); Baden-Baden, Germany: Deutscher Wissenschafts-Verlag.

Schumacher, M. 2006. *Security patterns: integrating security & system engg*, John Wiley & Sons.

Schumacher, M., Fernandez-Buglioni, E., Hybertson, D., Buschmann, F., & Sommerlad, P. 2013. *Security Patterns: Integrating security and systems engineering*, John Wiley & Sons.

Sindre, G., & Opdahl, A. L. 2005. Eliciting security requirements with misuse cases.*Requirements engineering*,10(1):34-44.

Spears, J.L. & Parrish, J.L., Jr. 2013. Teaching Case: IS Security Requirements Identification from Conceptual Models in Systems Analysis and Design: The Fun & Fitness, Inc. Case*, Journal of Information Systems Education*, 24(1): 17.

Steel, C., & Nagappan, R. 2006. *Core Security Patterns: Best Practices and Strategies for J2EE", Web Services, and Identity Management*, Pearson Education India.

Steven, J. 2006. Adopting an enterprise software security framework. *Security & Privacy*, IEEE, 4(2):84-87.

Supaporn, K., Prompoon, N. & Rojkangsadan, T., 2007, December. Enterprise assets security requirements construction from esrmg grammar based on security patterns. In *14th Asia-Pacific Software Engineering Conference (APSEC-2007)* , 112-119. IEEE.

Uzunov, A. V., Fernandez, E. B., & Falkner, K. 2012.Securing distributed systems using patterns: A survey. *Computers & security*, 31(5):681-703.

Van Lamsweerde, A. 2004. Elaborating security requirements by construction of intentional anti-models. In *Proceedings of the 26th International Conference on Software Engineering*, 148-157.

Van Lamsweerde, A., Darimont, R., & Letier, E., 1998. May. GRAIL/KAOS: an environment for goal-driven requirements engineering. In *Proceedings of the 19th international conference on Software engineering*, 612-613. ACM.

© COPYRIGHT UPM

Van Lamsweerde, A. 2009. *Requirements engineering: from system goals to UML models to software specifications*. Wiley Publishing.

Vieira, M. & Antunes, N. 2013. *Introduction to Software Security Concepts. In Innovative Technologies for Dependable OTS-Based Critical Systems*,29-38. Springer Milan.

Vysok, M., 2012. Diagram of Security. Information Sciences & Technologies: *Bulletin of the ACM Slovakia*, 4(1).

Wang, J., Song, W.T. & Chung, L. 2005. August. Analysis of secure design patterns: A case study in e-commerce system. In *Third ACIS International Conference on Software Engineering Research, Management and Applications (SERA'05)*, 174-181.

Winer, Benjamin J. 1962. *Latin squares and related designs*. McGraw-Hill.

Yin, R.K., 2003. *Case Study Research: Design and Methods,* Sage Publications. Thousand Oaks, California.

Yoder, J., & Barcalow, J. 1998. Architectural patterns for enabling application security. *Urbana*, 51:61801.

Yoshioka, N., Washizaki, H., & Maruyama, K. 2008. A survey on security patterns. *Progress in informatics*, 5(5):35-47.

Yu, E. S. 1997. Towards modelling and reasoning support for early-phase requirements engineering. Requirements Engineering. 1997.Proceedings of the *Third IEEE International Symposium*, 226-235.

Yu, Y., Kaiya, H., Washizaki, H., Xiong, Y., Hu, Z., & Yoshioka, N. 2008. Enforcing a security pattern in stakeholder goal models. Proceedings of the 4th *ACM Workshop on Quality of Protection*.

© COPYRIGHT UPM