A MODEL FOR INCORPORATING SECURITY PRACTICES INTO REQUIREMENT PHASE

BY

ASHRAF MOHAMMED MOHAMMED SAEED

A Thesis Presented to the

DEANSHIP OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

In

COMPUTER SCIENCE

MAY 2018

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN- 31261, SAUDI ARABIA

DEANSHIP OF GRADUATE STUDIES

This thesis, written by **Ashraf Mohammed Mohammed Saeed** under the direction of his thesis advisor and approved by his thesis committee, has been presented and accepted by the Dean of Graduate Studies, in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE.

Dr. Mahmood Niazi
(Advisor)

Dr. Khalid Al Jasser
Department Chairman

Dr. Mohammad Alshayeb
(Member)

Dr. Salam A. Zummo
Dean of Graduate Studies

Dr.Sajjad Mahmood
(Member)

16/12/13

Date

Ashraf Mohammed Mohammed Saeed
2018

This thesis is dedicated to my great parents, who never stop giving of themselves in countless ways.

ACKNOWLEDGMENTS

Alhamdulillah, I was allowed by Allah to complete this thesis. This thesis would not exist without the help and support from several people. I am grateful to my advisor, Dr. Mahmood Niazi Khan, who teaches, motivates, and guides me to complete this thesis. Many thanks to Dr. Mohammad Alshayeb and Dr. Sajjad Mahmood, my committee members, who encourage me to improve the quality of this thesis.

I am indebted to King Fahd University of Petroleum and Mineral (KFUPM), College of Computer Science and Engineering (CCSE), and especially Information and Computer Science (ICS) department with the faculties, that provide me the chance to learn so many ICS courses in depth.

Many thanks to my family in Yemen, especially to my parents, my wife, my brothers and my sisters, who always pray, motivate, and support me.

Last but not least, thanks to the support of all my friends, Special thanks to my colleagues, Mr. Abdullah Alsarmi, Mr. Mustafa Alwaridi, and Mr.Rushdi Algodari for their support.

TABLE OF CONTENTS

ACKNOWLEDGMENTS ..... **VI**

TABLE OF CONTENTS ..... **VII**

LIST OF TABLES ..... **X**

LIST OF FIGURES ..... **XII**

LIST OF ABBREVIATIONS ..... **XIII**

ABSTRACT ..... **XIV**

ملخص الرسالة ..... **XVI**

CHAPTER 1 INTRODUCTION **1**

1.1 Introduction **1**

1.2 Problem Statement **3**

1.3 Objectives **4**

1.4 Contributions **6**

1.5 Research Methodology **7**

1.6 Thesis Outline **8**

CHAPTER 2 LITERATURE REVIEW **9**

2.1 Software Security Concepts **9**

2.2 Capability Maturity Model Integration **11**

2.3 Related Works **14**

2.3.1 Initiatives for Secure Software Development Lifecycle 14

2.3.2 Tools to advocate Security for Software Development 19

2.3.3 Existing surveys and systematic mapping studies 20

2.4 Missing Work **22**

CHAPTER 3 RESEARCH METHODOLOGY 23

3.1 Introduction 23

3.2 Systematic Literature Review 24

3.2.1 Research Questions 26

3.2.2 SLR Protocol 26

3.2.3 Determining Research Sources 27

3.2.4 Defining Selection Criteria 27

3.2.5 Developing Search Strategy 28

3.2.6 Collecting Relevant Studies 32

3.2.7 Quality Assessment Criteria 32

3.2.8 Extracting Data and Findings 33

3.3 Sommerville Practices 38

3.4 Building the Questionnaire 38

3.5 Maturity Model Development 40

3.6 Case Study 41

CHAPTER 4 RESULTS 42

4.1 Analysis and Results 42

4.2 Outcomes of Questionnaires: 59

CHAPTER 5 REQUIREMENT ENGINEERING SECURITY MATURITY MODEL 73

5.1 Introduction 73

5.2 Structure of RESMM 74

5.2.1 Specific Goal Component 76

5.2.2 Measurement component 77

5.2.3 Rating of RESMM Process Attributes 77

CHAPTER 6 CASE STUDY 82

6.1 Introduction 82

6.2 Result 83

6.2.1 Organization A 83

6.2.2 Organization B 93

6.3 Evaluation of RESMM 103

6.4 Evaluation Criteria 103

6.5 Feedback Summary 104

6.6 Case Study Lessons Learned 109

6.7 Threats to validity 110

CHAPTER 7 CONCLUSION 112

7.1 Conclusion 112

7.2 Recommendations 114

REFERENCES 115

APPENDIX 126

1. Appendix A (List of Primary Studies) 126

2. Appendix B (Altered Sommerville RE Practices to SRE practices) 137

3. Appendix C (Requirement Engineering Security Maturity Model Practices) 142

4. Appendix D (Explanation of different categories of security practices at requirement phase) … 145

4.1 More details about practices for document security requirements 145

4.2 More details about practices for elicitation of security requirements 148

4.3 More details about practices for analysis and negotiation of security requirements 154

4.4 More details about practices for describing security requirements: 157

4.5 More details about practices for validating of security requirements: 159

VITAE 161

LIST OF TABLES

Table 3.1 Research questions for systematic literature review 26
Table 3.2 List of research sources 27
Table 3.3 Inclusion Criteria 28
Table 3.4 Exclusion Criteria 28
Table 3.5 Tailored search string based on searching rule in the research sources 30
Table 3.6 Quality Assessment Criteria 33
Table 3.7 Distribution of primary studies based on research sources 35
Table 3.8 Distribution of primary studies based on publication channel 36
Table 3.9 The classification of security practices at requirement phase 37
Table 4.1 Security Requirements Documentation Practices 43
Table 4.2 Elicitation Practices of Security Requirements 44
Table 4.3 Security Requirements Analysis and Negotiation Practices 48
Table 4.4 Describing Security Requirements Practices 51
Table 4.5 Modeling Practices of Security Requirements 53
Table 4.6 Validating Practices of Security Requirements 55
Table 4.7 Management Practices of Security Requirements 57
Table 4.8 Security Requirements Engineering Practices for Critical Systems 59
Table 4.9 The details of the organizations and the participants who filled the questionnaire 61
Table 4.10 Security Requirements Document Practices Chosen by Organization 62
Table 4.11 Security Requirements Elicitation Practices Chosen by Organizations 65
Table 4.12 Security Requirements Analysis and Negotiation Practices Chosen by Organizations 67
Table 4.13 Practices of Describing Security Requirements Chosen by Organizations 68
Table 4.14 Practices of Modelling Security Requirements Chosen by Organizations 69
Table 4.15 Practices of Validating Security Requirements Chosen by Organizations 70
Table 4.16 Management Practices of Security Requirements Chosen by Organizations 71
Table 4.17 The outcome of the conducted questionnaire (security practices) 72
Table 5.1 Number of security practices in each category 76
Table 5.2 Structure of SCAMPI Appraisal for RESMM 79
Table 5.3 Appraisal range of value used by IBM Rational Unified Process 81
Table 6.1 Security Requirements documents coverage for the RESMM process area 83

Table 6.2 Security Requirements Elicitation coverage for the RESMM process area 85
Table 6.3 Security Requirements analysis and negotiation coverage for RESMM process area 87
Table 6.4 Describing Security Requirements coverage for the RESMM process area 88
Table 6.5 Security System Modeling coverage for the RESMM process area 89
Table 6.6 Security Requirements Validation coverage for the RESMM process area 90
Table 6.7 Security Requirements Management coverage for the RESMM process area 91
Table 6.8 Summary table of maturity security practices of organization A 92
Table 6.9 Security Requirements documents coverage for the RESMM process area 94
Table 6.10 Security Requirements Elicitation coverage for the RESMM process area 95
Table 6.11 Security Requirements analysis and negotiation coverage for RESMM process area 97
Table 6.12 Describing Security Requirements coverage for the RESMM process area 98
Table 6.13 Security System Modeling coverage for the RESMM process area 99
Table 6.14 Security Requirements Validation coverage for the RESMM process area 100
Table 6.15 Security Requirements Management coverage for the RESMM process area 101
Table 6.16 Summary table of maturity security practices of organization B 102
Table 6.17 Ease of Learning Evaluation of Organization A & B 105
Table 6.18 User Satisfaction Evaluation of Organizations A & B 106
Table 6.19 RESMM Structure Evaluation of Organization A & B 107
Table 6.20 Feedback Results (Essay Answer) of Organizations A & B 108
Table 6.21 More explanation for some of the Practices 109

LIST OF FIGURES

Figure 2.1 Different levels of CMMI 13
Figure 3.1 Research Methodology 24
Figure 3.2 Systematic Literature Review Protocol [23] 25
Figure 3.3 Study Selection Criteria 34
Figure 3.4 Research sources of selected studies 35
Figure 3.5 Selected studies based on publication channel 36
Figure 3.6 Classification of papers based on security practices category 38
Figure 3.7 The structure of the Questionnaire 39
Figure 3.8 CMMI process area structure 40
Figure 4.1 Used Security Requirements Documents Practices during Questionnaire 63
Figure 4.2 Security Requirements Documents Practices Chosen by Organizations 64
Figure 4.3 Security Requirements Elicitation Practices Chosen by Organizations 66
Figure 4.4 Security Requirements Analysis and Negotiation Practices Chosen by Organizations 67
Figure 4.5 Practices of Describing Security Requirements Chosen by Organizations 68
Figure 4.6 Practices of Modelling Security Requirements Chosen by Organizations 69
Figure 4.7 Practices of Validating Security Requirements Chosen by Organizations 70
Figure 4.8 Management Practices of Security Requirements Chosen by Organizations 71
Figure 5.1 RESMM Development 73
Figure 5.2 The full process of the development of RESMM 74
Figure 5.3 CMMI process area structure vs RESMM structure 75
Figure 5.4 RESMM Structure 75

LIST OF ABBREVIATIONS

RE	:	Requirements Engineering
SDLC	**:**	Software Development Lifecycle
SLR	:	Systematic Literature Review
SR	**:**	Security Requirements
SRE	:	Security Requirements Engineering
RESMM	:	Requirement-Engineering Security Maturity Model
CMMI	:	Capability Maturity Model Integration
PP	:	Protection Profile
SoD	:	Separation of Duties
Org	:	Organization

ABSTRACT

Full Name : [Ashraf Mohammed Mohammed Saeed]

Thesis Title : [A Requirement Engineering Security Maturity Model]

Major Field : [Information and Computer Science]

Date of Degree : [May 2018]

Security is considered a critical aspect of software development. Many software programs have different processes or issues with respect to security due to the growth of internet-enabled products [1]. These flaws might exist in different phases of software development lifecycle, such as the requirement phase, design phase, coding phase, etc. [2]. Thus, tackling security at the requirement phase will help to avert the need for rework [3]. Security practices in the requirement engineering stage of software development require knowledge of these practices in order to create secure software. The aim of this research is to develop a Requirement Engineering Security Maturity Model (RESMM) to assist software development organizations in better specifying the requirements for secure software. In addition, the outcomes of this research are expected to provide software development organizations with the ability to measure their maturity of requirement specification for secure software. Eventually, this work will put software development organizations in a better position to deliver software that is more secure. Systematic Literature Review (SLR) has been conducted to identify security practices existing in literature. The identified security requirement practices have been classified into seven categories. Each category contains different security practices that are gathered by the conducted SLR. These practices have been used in addition to the Sommerville [4] practices to develop the questionnaire tool, which was later distributed to ten organizations

to highlight the most common security practices they use. These practices have been used in the development of RESMM. Two case studies were conducted with two software development organizations for the sake of RESMM-based assessment. Furthermore, two post-case studies have been done with two software development organizations to evaluate the applicability of RESMM in identifying the capability maturity levels of security practices in software organizations.

ملخص الرسالة

الاسم الكامل: : أشرف محمد محمد سعيد

عنوان الرسالة: نموذج النضوج لأمن هندسه المتطلبات

التخصص: علوم حاسوب

تاريخ الدرجة العلمية: مايو 2018

تعتبر الحماية من اهم المفاهيم في العديد من البرمجيات، حيث ان العديد من البرمجيات تمتلك بعض المشاكل والثغرات بالنسبة للحماية وذلك نتيجة لارتباط معظم البرمجيات بالأنترنت. هذه الثغرات ممكن ان تتواجد في مختلف مراحل تطوير البرمجيات على سبيل المثال في مرحلة جمع المتطلبات او في مرحلة تصميم البرمجيات او في مرحلة تطبيق البرمجيات عن طريق الكود و ..الخ. فبالتالي التطرق لمشاكل الحماية في مرحلة جمع البيانات سوف يساعد على تجنب إعادة بناء البرمجيات من جديد. فالممارسات الأمنية في مرحلة جمع البيانات تتطلب معرفة ما هي هذه الممارسات التي يجب اتباعها من اجل بناء نظام أمن. فالغرض من هذه الدراسة هي بناء نموذج نضوج لأمنية هندسة المتطلبات يدعى RESMM من اجل مساعدة منظمات تطوير البرمجيات في تحديد المتطلبات لبناء نظام آمن بطريقة جيدة. بالإضافة الى ان العائد من هذا البحث هو مساعدة منظمات تطوير البرمجيات من القدرة على قياس مقدار نضوج تحديد المتطلبات من اجل بناء نظام آمن. وفي الأخير فإن هذا العمل سوف يجعل من منظمات تطوير البرمجيات بأن تكون في محل ثقة من أحل إيصال البرمجيات بأكثر أمنية. فقد تم اعداد دراسة بطريقة منتظمة (SLR) من اجل معرفة ما هي الممارسات المتوفرة في الأبحاث السابقة. فبعد ان تم تحديد ما عي هذه الممارسات المتوفرة في الدراسات السابقة فإنه تم تصنيف هذه الممارسات الى سبع مجموعات. كل مجموعه تحتوي على العديد من الممارسات التي تم الحصول عليها سواء عن طريق الدراسات السابقة او عن طريق الاستبيان الذي تم عمله مع مجموعة من الشركات من اجل معرفة ما هي الممارسات الأمنية المستخدمة في تلك الشركات. فقد تم بناء الاستبيان بناءً على معظم ممارسات الموجودة في كتاب Sommerville، ولقد تم توزيع قائمة الاستبيان على 10 شركات من اجل معرفة ما هي الممارسات الأمنية التي يتم استخدامها في تلك الشركات. فلقد تم استخدام نتائج الاستبيان في بناء نموذج النضوج. فبعد ان تم بناء النموذج (RESMM) فقد تم تطبيق النموذج الذي تم بناءة مع بعض الشركات من اجل عمل تقييم لهذا النموذج في بيئة العمل. فقد ظهرت النتائج عن قابلية استعمال هذا النموذج من اجل معرفة مستوى النضوج للممارسات الأمنية في منظمات البرمجيات.

CHAPTER 1

INTRODUCTION

1.1 Introduction

Now, more than ever, businesses are growing quickly. All companies compete to deliver high-quality products and services quickly and inexpensively. Nearly all organizations in the twenty-first century have started creating increasingly sophisticated products. So, organizations must be able to control and manage more and more sophisticated development and maintenance processes. Moreover, the quality of software is the process which confirms the degree of excellence of the software [5]. Software applications are involved in various aspects of our lives. They are used for many purposes, such as recording and editing personal information, financial transactions, health records, managing critical data for businesses, etc. In this regard, a critical question that can be raised here is: how far can we put our trust in software? Furthermore, who is accountable for the security of the software applications we rely on?

During the last fifteen years, different methodologies have been presented for incorporating security into the development lifecycle of software [6]. Thus, new practices have been implemented at various stages of the software development lifecycle to improve products' security. Security measures such as the code review of security [7] and modeling of threats [8] have been involved in most software security methodologies to date. In the

present day, there are different tools that are used to support and review activities of security code [9]. In addition, there are also some automated tools for security static analysis that analyze programs in order to identify weaknesses of software security beyond requiring the execution of the code [6]. Generally, there are three levels where code is analyzed: the source code level, the binary code level, and the bytecode level [6]. In the current marketplace, there are several methodologies, maturity models, standards, and guidelines that can aid organizations in improving their business. One of these approaches, which focuses on software quality, is the Capability Maturity Model Integration (CMMI) [10]. The main goal of CMMI is to assess the capability of an organization by measuring the degree to which software processes are determined and controlled. There are 22 process areas in CMMI which guide software development organizations in executing each phase of the software development lifecycle (SDLC). However, in CMMI, no process area has been designed to address software security issues in general and, in particular, the requirements of security engineering issues. This does not mean that software security problems were never addressed before, but these problems were often underestimated, misunderstood, and not addressed in the way they should have been [11],[12]. Traditionally, security considerations are incorporated as an afterthought, which leads to a lifecycle of "penetrate-and-patch." In addition, organizations spend vast sums of money on firewalls, intrusion detection systems, antivirus software, antispyware software, and encryption mechanisms [13]. However, this approach does not work perfectly, and organizations remain vulnerable to security risks and cyber-attacks that take advantage of security flaws [12].

Since high-level security goals “tend to make security requirements not specific enough to guide designers and to verify that the requirements are met,” security requirements should therefore be able to “express what is to happen in a given situation, as opposed to what is not ever to happen in any situation” [14]. Furthermore, based on literature and studies [15],[16],[17], it is obvious that it is not easy to provide a simple, effective model to measure software security during the requirement phase. This is because the standards and best practices are too broad and provide too few guidelines for tackling security requirements using a simple method.

Therefore, there is a need to investigate the current “generic and specific” model of goals and practices in relation to security requirements, with an aim to come up with a new process area based on collected evidence from relevant literature and software industry experience. In addition, we need to provide guidelines for measuring security at the requirement engineering phase of SDLC.

Generally, this study is divided into seven chapters. Chapter 1 provides an introduction to the research and puts forward the objectives of the research. Chapter 2 gives a brief background of CMMI, the concept of software security, and related work and relevant literature. Chapter 3 presents the research methodology. Chapter 4 presents the results of both SLR and the questionnaire. Chapter 5 presents the development of RESMM. Chapter 6 presents the evaluation of RESMM via two case studies. Finally, the conclusion and future work is outlined in Chapter 7.

1.2 Problem Statement

Security requirement engineering challenges in the software industry can be handled using various approaches. For instance, the industry can improve the security

awareness of software developers by hiring security experts or by setting up training workshops. In addition, a security consultant can be hired by an organization to investigate the activities of its SRE. But the cost of doing that would be high since these approaches could only be implemented in large-scale organizations.

Several studies have been conducted on the techniques of building secure applications by both researchers and practitioners [18], but few studies have been done on models of security development that are used as guidance in the development process itself [19]. Moreover, incorporating security practices and proce sses into various phases of SDLC, such as the requirement phase, remains a challenge. In addition, prior to this research, there has been no study which provides a technique or tool that software organizations can use to identify the maturity level of their security requirement practices in software development.

The proposed model of incorporating security practices into the requirements phase should be validated based on the usability and reliability of that model in the real world of the software industry. This model should also work in both small and large organizations. In addition, to reach to high influence, it has to cover most security requirement practices.

1.3 Objectives

The aim of this research is to identify an effective way to assist software development organizations in specifying the requirements for secure software. In order to achieve that, we will develop a new process area in CMMI called RESMM. We will draw upon the CMMI structure for the development of RESMM. We will employ practical and evidence-based approaches in order to develop RESMM, e.g., systematic literature review

and empirical studies related to the software industry. This two-step process will give us confidence in the reliability of the collected data. In addition, we will conduct a case study in order to evaluate RESMM in a real-world environment.

It is expected that RESMM will significantly influence the software security issues currently reported in software development projects. This work will provide other researchers with a steady foundation on how to build new approaches to software security. New software security practices will then be developed to address a high number of security issues currently reported in software development projects. In addition, the project outcomes will enable software development organizations to measure their maturity by specifying requirements for secure software. Eventually, this work will put software development organizations in a good place to grant more secure software. RESMM will be available online to Saudi researchers and software practitioners. Managers of local Saudi software development organizations will be able to use RESMM in the evaluation of their strengths and weaknesses in terms of the design, implementation, improvement, and measurement of suitable processes to effectively manage their engineering security requirements.

According to the purpose statement, the main goal of this study is to discover how security can be integrated into CMMI. As such, the following objectives are meant to be achieved:

- Develop a process area, RESMM, for security requirements to assist software development organizations in specifying the requirements for secure software in a better way.

- Discuss the current practices of organizations in regard to software security at the requirements phase in order to identify security practices that are related to security requirements.
- Propose measurement guidelines that can be implemented in the software development lifecycle to measure the level of security at the requirements engineering phase.
- Demonstrate the applicability of the RESMM process area by applying it in a real-life case study.

1.4 Contributions

- Systematic Literature Review (SLR). This study is used to identify and characterize security requirements practices into different categories based on the area cover. The obtained practices of SLR have been used to develop the questionnaire tool, which was later distributed to 10 organizations to highlight the security practices used by them. These practices have been used in the development of RESMM.
- Develop a requirements engineering security maturity model called RESMM that can help organizations to specify their requirements for secure software in a better way. In the development of RESMM, we will draw on CMMI structure, and we will create a process map/model for the requirements process area, RESMM.
- Contribute into the bulk of literature in the field of software security and publish our main contributions in a journal.

1.5 Research Methodology

The research methodology consists of the following four phases:

Phase 1: Conduct a Systematic Literature Review (SLR)

Three database sources (IEEE, ACM, and ScienceDirect) were used to obtain relevant studies for the research questions. A well-known protocol was used for doing SLR to make sure the same results would be obtained whenever the research was replicated by other researchers. Then, security requirement practices that have been addressed in those studies were identified.

Phase 2: The details of Adapted Sommerville practices.

We modified the requirement engineering practices proposed by Sommerville [4] into security requirement engineering practices. We modified 66 RE practices to RE security practices as shown in Appendix B (Altered Sommerville RE Practices to SRE practices).

Phase 3: Administer a questionnaire to organizations

After the identification of security requirement practices from SLR, ten organizations were queried about the security practices they follow during the requirements gathering process. This procedure enhanced the reliability of the collected data and helped in the identification of the practices which would be used in RESMM.

Phase 4: Develop a Requirement Engineering Security Maturity Model

The development of RESMM was based on security practices identified via SLR and through questionnaires with different organizations. The structure of RESMM was based on CMMI. This study utilizes the outcomes of RESMM to develop security requirement practices across different categories. The assessment tool called Standard CMMI Appraisal Method (SCAMPI) [20] was chosen for the measurement of requirements security maturity due to its ability to appraise the events of organizations based on their maturity or capability level [20].

Phase 5: Conduct a case study

Two case studies were conducted with two software organizations to assess their maturity using RESMM. In addition, two post-case studies have been done to evaluate the usability of RESMM in a real-world environment. Feedback from those organizations has been taken into consideration.

1.6 Thesis Outline

This research is organized as follows. Chapter 2 gives a brief background about security requirements as well as an up-to-date literature review. Chapter 3 shows the research methodology in some detail. Chapter 4 presents the results of the application of RESMM, such as the security requirement practices categories. Chapter 5 describes how RESMM was developed. Chapter 6 consists of the case studies and explains their outcomes and feedback. Chapter 7 serves as the conclusion of this work and provides recommendations for future work.

CHAPTER 2

LITERATURE REVIEW

This chapter provides definitions of several software security concepts, standards of security, and security requirements engineering, and it presents some of the previous studies which have been done in the field of security requirements engineering and security practices at a different phases of software development. We also highlight the gaps in the previous research.

2.1 Software Security Concepts

The security software industry has grown rapidly over the last decade and still continues in its growth. Nowadays, software is used to control big financial systems, databases, and communication systems. Each software program has its own roles and functionalities, but each program is also vulnerable to attack due to its nature [21]. As a corollary, software is also growing in various aspects such as complexity, extensibility, size, and connectivity. Recently, attackers have exploited the extensible property of software programs to hack systems remotely [12]. Hence, there is increasing demand for security in software.

Security is often defined as an add-on feature. Many organizations do not give much consideration to security at the pre-development and development phases, but they do consider it as a post-development activity. They incorporate security as a patch [22] after the completion of software development. In addition, organizations spend a lot of

money on the purchase of good firewalls and antivirus programs, thinking that these applications will be enough to make software secure. However, attacks continue to occur.

Given the above discussion, it can be concluded that it is not good enough to secure software in a post-development manner, and there is a demand for discovering means and ways to enhance software security.

System developers, requirement analysts, designers, and architects are often unmindful of the software security concept and specify a few or no security considerations during the development process. Due to such indifference, a number of security problems can occur in the software. According to a systematic mapping study [23], the requirement phase has not received its due interest in academic research. This emphasizes the pressing need to do more work at the requirement phase to address security requirements, since doing that can help with preventive detection and alleviation of security threats in different software systems.

Different people have defined software security in various ways. One of these definitions says that “software security is about building secure software: designing software to be secure, making sure that software is secure, and educating software developers, architects, and users about how to build secure things” [24].

Several factors introduce new problems to software. One of these factors is the lack of software development methodologies [25]. Another factor is the exponential increase in internet-enabled applications [22]. One more factor is the activity of hackers and unconscious internet users [26]. One of the most crucial points common to all of these problems is software vulnerability and weak spots that can be targeted by hackers. Software

vulnerability has been defined as “a weakness in the security system, for example, in procedures, design, or implementation that might be exploited to cause loss or harm” [22].

Some terminologies need to be explained to have a better perception of software vulnerabilities and issues of software security. These terms include *asset*, *defect*, *bug*, *software security error*, and *software security requirements* [23]. The *asset* is whatever is valuable that requires protection. Since the asset is always the target of threats, it needs to be protected. Many studies demonstrate that focusing on security at early stages can help organizations save billions of dollars, yet security concerns are usually addressed as an afterthought to functional requirements [27]. In fact, vulnerabilities will manifest in code if they are not detected during the requirements and design phases. So, building a model can help organizations check the security requirements in SDLC phases.

2.2 Capability Maturity Model Integration

Capability Maturity Model Integration (CMMI) [5] is a group of best practices that can assist organizations in enhancing different processes of their activities. Best practices concentrate on the activities that are used to develop the quality of products and services to meet the needs of the end users and customers. These practices involve the lifecycle of the product from concept through delivery and maintenance. The reason for developing CMMI is to build a framework through the integration of various business maturity models. CMMI has different versions, such CMMI Version 1.1 (released in 2002), CMMI Version 1.2 (released in 2006), CMMI Version 1.3 (released in 2010), and CMMI (released in 2018) [28]. There are 22 process areas in CMMI that guide software development organizations in what to do at each phase of SDLC. But unfortunately, there is no process in CMMI that

tackles requirements engineering security issues. A process area is a group of associated practices in an area which, if it has been totally implemented, will satisfy a set of goals that are considered crucial to making an enhancement in that area. Each process area has its own purpose as well as generic and specific goals in order to satisfy the purpose of that process. A secure software process can be thought of as the group of activities that have been undertaken to develop, maintain, and hand over a secure software solution [29]. So, for our process area, we will define the purpose of the RESMM process and follow other steps of the CMMI process formation to build our process. Both versions, CMMI v1.3 and v2.0, contain five maturity levels in CMMI: initial, managed, defined, quantitatively managed, and optimizing. Figure 2.1 shows the maturity levels of CMMI. Each maturity level tackles specific types of process areas. CMMI v2.0 differs from v1.3 in some aspects. The first difference is that CMMI v2.0 is focused on the performance practices since these practices have been integrated into all maturity levels of the model. This serves to enhance organizational performance in order to maximize return on investment (ROI) [30]. Another difference is that CMMI v2.0 focuses on enhancing the usability feature and integrating guidance since the architecture of CMMI v2.0 is scalable, which facilitates smooth integration of new content with some guidance into certain business needs. Moreover, CMMI v2.0 does not stipulate that technical business language must be written in that format. Thus, CMMI v2.0 is easier to understand and access by even non-native English speakers. In CMMI v2.0., it does not have generic goals and generic practices, which are found in v1.3. Building and sustaining practices have replaced the generic goals and generic practices in v2.0. Furthermore, CMMI v2.0 has a new appraisal method which

involves a statistically-validated random sampling approach. Thus, CMMI v2.0 has an improved value and reliability of appraisals.

Recently, best practices for security have come into existence, but they must be integrated into a model or standard which would provide guidance to the extended developer community. In CMMI-DEV, there is a framework that has security activities, but it lacks guidance of security aspects [31]. According to that, we develop RESMM to assist software development organizations in better specifying requirements for secure software. The developed RESMM is customized with CMMI v1.3 and CMMI v2.0 since these versions of CMMI have the same components of CMMI levels.

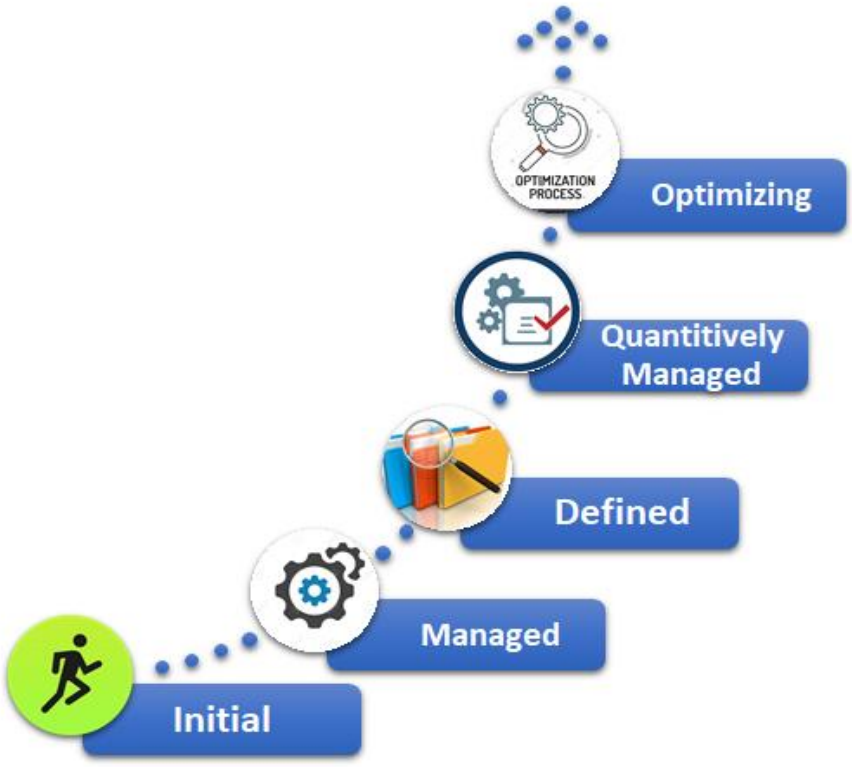

Figure 2.1 Different levels of CMMI

2.3 Related Works

Software security is a hot research topic. In the last decade, valuable contributions have been made in this research area from both academia and the industry. In the following section, we address different methodologies that integrate the security aspect in SDLC, and we show the popular practices that are common in these methodologies. These proposed methodologies assist in underlining the need for building security into the lifecycle of software development.

For decades, almost all software developers seemed to be unconscious of the importance of software security, and they were much more concerned with implementing the functionalities of software to meet deadlines in the delivery of their products, always resolving to repair the inevitable bugs at the next release of the software [32]. This pattern, as adopted by developers, needs to change. Software vendors need to follow a stricter process of software development that shows more concern with security. The proposed process can help to decrease the number of security vulnerabilities at the design phase, coding phase, and documentation phase, and to reveal and eliminate detected vulnerabilities as early as possible in the software development lifecycle [33].

2.3.1 Initiatives for Secure Software Development Lifecycle

There are several studies that address security requirements at different stages of the software development lifecycle. One of these studies has been published by the National Institute of Standards and Technology (NIST) and is titled “Security Considerations in the Information System Development Lifecycle.” This study proposes a framework to integrate the security aspects into the generic development lifecycle of software [34]. Furthermore, many IT managers recognize the need to integrate security

aspects into the software development lifecycle [35]. In 2002, Microsoft started Trustworthy Computing (TwC) [36], which was used by development groups to execute "security pushes" to discover ways to enhance the security of existing code. By 2004, software development at Microsoft followed a mandatory policy called Microsoft Security Development Lifecycle (SDL) [37], which includes amending the software organization's processes by incorporating scales that result in enhancement of software security.

Then, a whitepaper was published by Microsoft in 2013 under the title "Security Development Lifecycle for Agile Development."[38]. It defines a process of using lightweight software security in the case of utilizing Agile software development methods. The aim of their paper is to combine the Agile methodologies with the proven Microsoft SDL by following a path that preserves the basics of both Agile and Microsoft Security Development Lifecycle [38]. The comprehensive Lightweight Application Security Process (CLASP) [39] is a structured and well-organized approach intended to tackle security at the earliest stages of the software development lifecycle. CLASP is an open source application that addresses the Open Web Application Security Project (OWASP) [39]. The objective of CLASP is to give more concern to security when constructing software. CLASP integrates the process of security at each stage of the software development lifecycle. It marks thirty activities of security and is related to one or more project roles [40]. The importance of developing secure software has increased due to the pervasive use of the internet and networked systems. Several ways have been used to prevent attacks, such as anti-virus software, firewalls, and "intrusion detection systems" [41]. However, attacks continue to occur. The reasons behind them are inherent in the software itself, i.e. poor documentation and disregard to security issues during software

design. Thus, a new area of research has come into focus called software security [42],[40],[43].

Software attacks are tackled at the code level by using appropriate coding techniques. Here, however, it is proposed to secure software during the design [24], which means to tackle the security issues during the design phase, i.e. not as an afterthought. Although many organizations and companies have followed that afterthought, this has led to security problems after deployment of the software product. The elimination of such problems is expensive at that phase, compared with doing the same at earlier phases of the software lifecycle. McGraw (2006) [44] observed that 50 percent of security problems are encountered as a result of design flaws, and he demonstrates that the analysis of architectural risk is a core part of any compact security program.

Several methods have been proposed for the analysis of security requirements. One of these methods is threat modeling [8]. The analyzers, who use the threat modeling method, are required to disband the software into tiny components and use the data flow diagram to draw the flow of data between those components. Data flow diagrams elicit the threats. After that, a probability and an amount of potential loss are assigned to be used as an assessment of each threat. If the precise architecture is not defined, it would be very difficult to accomplish this. In fact, threat modeling is more concerned with risk assessment than risk identification. But risk identification is more important at the requirement analysis stage.

Security approaches are procedures or mechanisms that are incorporated during the development of secure applications by using systematic and well-defined methods. For example, risk analysis is a well-defined method that is merged into the Agile development

model. There are different studies on the use of risk analysis and threat modeling as a security technique or approach [45],[46],[47]. Several such studies have shown various degrees of similarity. The purpose of the threat modeling approach is to analyze the system from the security point of view by identifying and preventing any probable security risks to the system. Threat modeling helps in securing the system by building security at the beginning of the system development life cycle.

Another example of a security approach is the use of the agility reduction tolerance approach, which is concerned with the efficiency of integration of some parameters of agility reduction tolerance using an activity integration algorithm. In this approach, it extracts security activities and identifies their degree of agility. However, no empirical studies have been done on comparing the effectiveness of threat modeling with other techniques. Haley et al. [48] presents a method to identify threats using the Aspect-Oriented Software Development (AOSD) approach and problem frames to elicit security requirements. AOSD is focused on eliciting threats but does not focus on design aspects of software.

Another study has been done by Okubo et al. [13] focusing on the design of security aspects of software. They present an approach that would identify the security aspects at an early stage—the requirement analysis stage—with an expansion of misuse cases. The extension of misuse cases has improved the capability of visual assistants to elicit threats and security aspects. They evaluated their approach by applying it to a web application domain. Their approach shows in what place the threat could be encountered. So, it identifies those threats and adds them to a use case diagram since the objective of their work is to identify threats and to measure those threats as much as possible in the later

stage. The benefit of using this approach is to help architects to determine if the specification has met the desired requirements or not. However, there is still a problem with their approach since there are difficulties in clarifying all the prospective assets at the stage of analysis. Use cases have been used a lot in requirements engineering. They are appropriate for most functional requirements, but they have some limitations in eliciting security threats and requirements.

A study by Sindre et al. [46] attempted to demonstrate a systematic approach to for eliciting security requirements depending on use cases. In their approach, they extended the original use cases to involve misuse of different sorts of extra-functional requirements beyond security. Another study showed that the approaches of industrial security can be deduced from the solution world instead of the problem world [45]. With respect to the consideration of security requirements, use cases can be modified to help integrate the work of functional and extra-functional requirements. The extension of use case diagrams to include the negative use case helps clarify the unwanted behavior of the proposed system for the objective of eliciting security requirements. Sindre et al. [46] addressed the guidelines to describe in more detail the method of misuse cases using textual templates. Furthermore, they addressed how method guidelines can elicit the security requirements by using misuse cases. Their approach has been checked in realistic settings and on examples, and it was recently used with security patterns and risk management. Their method guidelines are given to make sure the approach is valuable in the early elicitation of security requirements. However, the given method guidelines are still too general and inaccurate since the number of considered associated threats and potentially critical assets are large and the misuse-case approach is not likely adequate for all types of threats.

Current methodologies present guidelines that are concerned with specific areas, such as secure coding, risk management, and threat modeling [2],[49],[8].

A study by McGraw and Chess (2008) [24] specified a framework for software security with intention to grab the overall high-level comprehension that involves all leading initiatives of software security. This framework has four domains and 12 practices. Each domain has its own practices. The most remarkable software security practices are code review and architecture analysis.

2.3.2 Tools to advocate Security for Software Development

Recently, several tools have emerged to support a secure software development lifecycle. There are tools for the requirement and design phases, such as architecture and modeling tools. In addition, there are tools for the implementation phase, such as code analysis [50]. Furthermore, there are several tools that are used for testing and other tools used for deployment phases (e.g., black box testing tools [51] and other tools for penetration testing [52]). All these tools assist the software development team to integrate security into the development lifecycle. Moreover, there are several tools that have been used for the review practices of security code, such as white box tools [51], which are considered essential for incorporating security in the software development lifecycle. White box tools are classified into static and dynamic analysis tools. Static analysis tools are used to analyze the software without requiring the execution of the software; these include binary code scanners, bytecode scanners, and source code security analyzers. Binary code scanners are used to detect the vulnerabilities of software via disassembly and pattern recognition. In case the source code does not exist, the bytecode scanners are used to detect vulnerabilities of software in the bytecode. Source code security analyzers are

used to check the source code in order to detect and report weaknesses that could result in security vulnerabilities. Static analysis tools are capable of examining the compiled results and can identify any possible vulnerability that could be caused by the compiler. However, the dynamic analysis tools look at the executing application in order to detect potential security vulnerabilities. These tools report scenarios of vulnerability at runtime and analyze the software application from internal and external viewpoints. One of the tools that analyzes the software from outside is the web application vulnerability scanner [53].

2.3.3 Existing surveys and systematic mapping studies

A systematic mapping study called "Reusable knowledge in security requirements engineering" has been done by Amina et al. [54]. They showed the big picture of the existing literature on knowledge of security requirements engineering and reuse in security requirements engineering. They demonstrated the existing methods to be reused in security requirements engineering. In addition, they showed the existing modeling frameworks, techniques, and tools for reuse in security requirements engineering. Their mapping study analyzed more than 30 approaches that have been used in security requirements engineering for almost 20 years of research. The main contribution of their work was to come up with a framework to analyze and compare the various existing proposals as well as the taxonomy of future contributions which are concerned with knowledge reuse and security requirements engineering. They also defined the different forms of knowledge representation, and reuse was identified. Furthermore, they updated the previous surveys and concluded that most methods need to involve more reusable knowledge to control security requirements.

Another systematic mapping study on Security Requirements Engineering has been done by Naurin et al. [55]. In this study, they address the studies that have been done in the period 2010–2015. They analyzed, classified, and discovered the hot spots in the literature. In their classification, they included the different types of studies that have been carried out in security requirements engineering and what the top journals are for those studies. They also observed hot spots of various kinds with regard to security requirements engineering problems and the solutions to those problems that were addressed in the literature.

A study by Shuaibu et al. [42] applied SLR to the security of web applications. They mentioned that there is no preferred development mode or standard to be used in the development of web applications. They found that the Agile development models had been given more attention, perhaps due to the participation of multiple stakeholders when deliberating the security viewpoints, since that helps in the conventional understanding of security requirements instead of imposing it on certain members of the development team. They also found that threat modeling techniques had been used to enhance security during development stages. They mentioned that the reason for using threat modeling techniques may be due to their effectiveness in tackling different kinds of vulnerabilities [42].

A survey [56] administered by the Computer Security Institute and the US Federal Bureau of Investigation in 2006 reports that computer security incidents were the cause of losses sustained by 131 respondents, with their losses estimated to surpass US $50 million in 2006. Excluding the incidents caused by computer security, the remaining damage cost about $30 million. Thus, the average setback an organization experienced was about

$230,000 for the year. This may be due to configuration errors; however, respondents think flaws in software security caused much of this cost.

2.4 Missing Work

Many security practices at the requirement phase [57],[58],[59] have been published in different studies, but these practices need to be tackled in a systematic way if they are to be used in the creation of a model for building secure software from an early stage of software development. In addition, many software developers are not aware of security practices at the requirements engineering phase of the software development lifecycle. Thus, the software industry needs to integrate security into the software development lifecycle, and it has been a crucial requirement for the software industry. However, incorporating security practices and processes into different stages of the software development lifecycle, such as the requirement phase, remains a challenge. In addition, in CMMI, there is no process area that has been designed to address software security issues in general and requirements engineering security issues in particular. So, in our study, we have developed a process area called RESMM. This process assists the software development organizations in better specifying requirements for secure software. Furthermore, no measurements allowing software development organizations to measure their maturity in specifying requirements for secure software are available. Thus, this work enables software development organizations to measure their maturity with respect to the implemented security practices at the requirement phase.

CHAPTER 3

RESEARCH METHODOLOGY

3.1 Introduction

In this chapter, we describe the research methodology for developing a RESMM. We go through different phases to achieve that purpose. First, an SLR is done to identify the existing security practices which are available in the literature. Second, a questionnaire is administered to different organizations to verify the collection practices we found in the SLR. In addition, in the questionnaire, the respondent organizations are asked about the security practices they used during the requirement phase of software development. After that, RESMM is built based on the feedback obtained from those organizations. Finally, a case study has been conducted to evaluate the RESMM with respect to software organizations. Figure 3.1 shows the research methodology followed in this study.

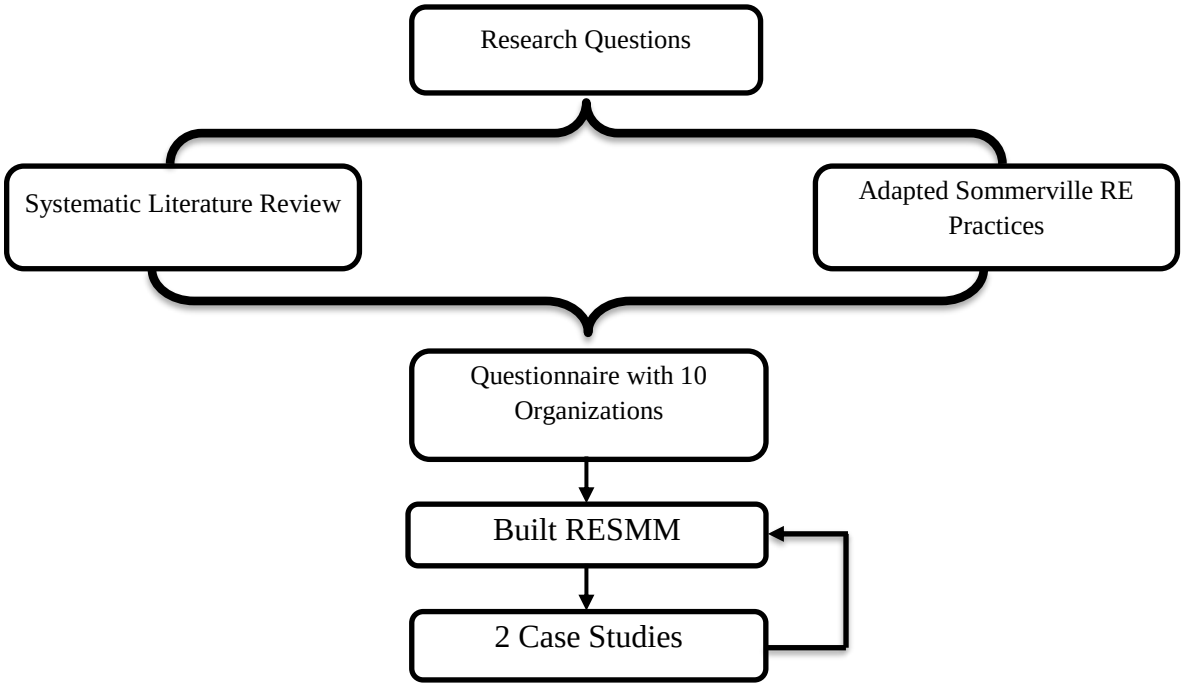

Figure 3.1 Research Methodology

3.2 Systematic Literature Review

SLR is a method for appraisal and composition of primary research papers using a delicate and obviously documented methodology in the search strategy and chosen studies. This will minimize bias in the outcomes. In addition, the apparent documentation of SLR and the decisions taken will enable the reviewer to be updated.

In order to conduct SLR to identify security practices at requirements phase, we need to follow certain steps throughout this thesis. Based on our research questions, we intend to follow a well-defined and accurate method to characterize, appraise, and illustrate all the relevant studies related to our research question. For this purpose, SLR here is established on the review guidelines which are offered by Kitchenham and Charters [60].

SLR protocol has been given to show the details of all procedures that we have considered during SLR (See Figure 3.2). The main procedures are described as follows:

1. Research questions
2. SLR protocol
3. Identification of relevant studies after applying the search string to *Science Direct* database
4. Apply the selection process into chosen studies by the providing criteria
5. Extraction and analysis of collected data.

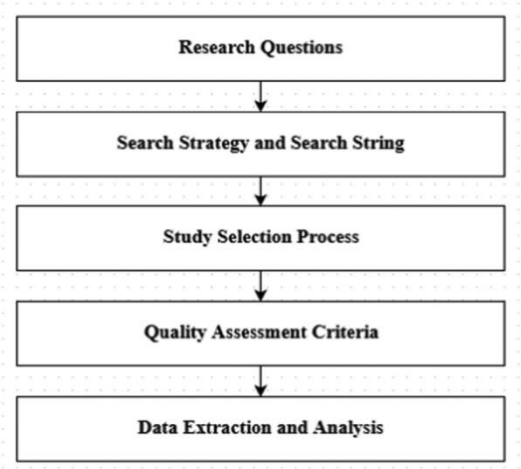

Figure 3.2 Systematic Literature Review Protocol [23]

The result of the RESMM has a big influence on SLR outcomes. Before performing SLR, we need to consider several points that will help in the collection of primary studies, i.e. consideration of search strategy, determination of the digital libraries, identification of the selection criteria, and indication of the quality assessment criteria, if needed.

3.2.1 Research Questions

Before going through the protocol of conducting SLR, we need first to define and construct our research questions. Since, we follow the Sommerville classification of practices in this study, we have identified eight research questions to be considered during SLR. These research questions are listed in Table 3.1.

Table 3.1 Research questions for systematic literature review

No.	Research Question
RQ1	What are the practices for documenting security requirements?
RQ2	What are the practices for eliciting security requirements?
RQ3	What are the practices for analyzing and negotiating of security requirements?
RQ4	What are the practices for describing security requirements?
RQ5	What are the practices for modeling security requirements?
RQ6	What are the practices for validating security requirements?
RQ7	What are the practices for managing security requirements?
RQ8	Security Requirements Engineering for Critical Systems?

3.2.2 SLR Protocol

In this section, we show the procedure of implementing the SLR. There are some points that need to be highlighted, i.e. research sources, selection criteria, search string, identification of the primary studies, and extraction and analysis of the collected data. Each point is thoroughly considered in coming up with convenient results.

3.2.3 Determining Research Sources

We select three databases to serve as the databases source for the SLR. The selected databases have been chosen due to their large capabilities and the fact that the SLR is multi-disciplinary with more focus on science and technology. Indeed, those databases contains a huge bulk of seminal and original research work. Table 3.2 shows the addresses for each research source.

Table 3.2 List of research sources

Research sources	URL of advance search
Science Direct	http://www.sciencedirect.com/science/search
IEEE Xplore	http://ieeexplore.ieee.org/search/advsearch.jsp
ACM	https://dl.acm.org/advsearch.cfm

3.2.4 Defining Selection Criteria

This sub-section is divided into two parts. The first part is the inclusion criteria which determine the studies chosen for investigation in more detail. Selected studies are closely related to our research questions. The second part is the exclusion criteria which rule out studies that are not closely related to our research questions. We select the study only if it satisfies all the inclusion criteria. Table 3.3 shows the inclusion criteria, whereas Table 3.4 shows the exclusion criteria. The main purpose of the selection criteria is to make sure the studies selected are related to the research objectives. The inclusion and exclusion criteria have been taken from a recently published research paper on systematic mapping studies [23].

Table 3.3 Inclusion Criteria

No.	Inclusion Criteria
1	Related to the domain of secure software engineering
2	Studies which focus on the most common security practices at requirement phase
3	Published after 2005.

Table 3.4 Exclusion Criteria

No.	Exclusion Criteria
1	Non-English language
2	Studies that are irrelevant to our research questions
3	White papers, technical reports, master theses, Ph.D. dissertations, and textbooks
4	Studies related to different domains
5	Publications not published in a peer-reviewed format
6	Papers without adequate related work

3.2.5 Developing Search Strategy

In this step, we construct the search string for our study. This can be done by following certain steps, such as population, intervention, outcome of relevance, and experimental design. After that, we are concerned with obtaining the synonyms of the terms that have been obtained from the previous step. In addition, we use the Boolean operator to combine the synonyms of terms. Finally, we verify the collected terms. Thus, we show

how we apply the search string into the ScienceDirect database. We build up our search terms by analyzing the keywords of our research questions from the four perspectives above: population, intervention, outcome, and experimental design. Thus, the strategy of the SLR search is instituted on these four steps as follow:

- Population: security requirement
- Intervention: the existing practices for secure software at an early stage of software development
- The outcome of relevance: “Secure IS development,” “secure software development,” and “secure software.”
- Experimental design: Empirical studies, SLR, expert observation and opinions, theoretical studies, and case studies.

We explore the synonyms of the derived terms and combine the synonyms by means of Boolean operators, such as “AND” and “OR”. In order to test our terms, we validate the selected terms in ScienceDirect databases. Thus, the following synonyms represent the possible relevance to the topic, as follows:

- **Security requirement**: “security requirements engineering” OR “security requirement” OR “SRE” OR “security development” OR “Secure requirement” OR “secure development” OR “insecure requirement” OR “insecure development”.
- **Practice**: “initiative” OR “method” OR “patterns” OR “practice” OR “activity” OR “approach” OR “process” OR “steps” OR “technique” OR “technology” OR “model” OR “framework” OR “guideline”.
- **Secure software:** “secure software development” OR “secure systems development” OR “secure software development lifecycle” OR “systems development lifecycle” OR

"SDLC" OR "software development process" OR "secure IS development" OR "software development lifecycles".

After we identify the synonymous terms, we verify the different terms in database sources. In the end, after many trials, we specify the chosen search string which we follow in this study: ("security requirements engineering" OR "security requirement" OR "SRE" OR "security development" OR "Secure requirement" OR "secure development" OR "insecure requirement" OR "insecure development") AND ("initiative" OR "method" OR "patterns" OR "practice" OR "activity" OR "approach" OR "process" OR "steps" OR "technique" OR "technology" OR "model" OR "framework" OR "guideline") AND ("secure software development" OR "secure systems development" OR "secure software development life cycle" OR "systems development lifecycle" OR "SDLC" OR "software development process" OR "secure IS development" OR "software development lifecycles") [All Sources (Computer Science)].

This search string was tailored to correspond to each research source due to different mechanisms. If the accuracy of the search string was low, then the number of studies collected was too large. Thereafter, it required greater effort to identify the relevant studies. Details of the tailored search strings are listed in Table 3.5.

Table 3.5 Tailored search string based on searching rule in the research sources

Sources	Search string
Science-Direct	("security requirements engineering" OR "security requirement" OR "SRE" OR "security development" OR "Secure requirement" OR "secure development" OR "insecure requirement" OR "insecure development")

	AND (“initiative” OR “method” OR “patterns” OR “practice” OR “activity” OR “approach” OR “process” OR “steps” OR “technique” OR “technology” OR “model” OR “framework” OR “guideline”) AND (“secure software development” OR “secure systems development” OR “secure software development life cycle” OR “systems development lifecycle” OR “SDLC” OR “software development process” OR “secure IS development” OR “software development lifecycles”)
IEEE	("security requirements engineering" OR "security requirement" OR "SRE" OR "security development" OR "Secure requirement" OR "secure development" OR "insecure requirement" OR "insecure development") AND ("initiative" OR "method" OR "patterns" OR "practice" OR "activity" OR "approach" OR "process" OR "steps" OR "technique" OR "technology" OR "model" OR "framework" OR "guideline")
ACM	("security requirements engineering" OR "security requirement" OR "SRE" OR "security development" OR "Secure requirement" OR "secure development" OR "insecure requirement" OR "insecure development") AND ("initiative" OR "method" OR "patterns" OR "practice" OR "activity" OR "approach" OR "process" OR "steps" OR "technique" OR "technology" OR "model" OR "framework" OR "guideline")

3.2.6 Collecting Relevant Studies

To collect potential studies for the research, we applied the search string into the various database sources. The search string was applied several times on the databases to ensure the results were acceptable and accurate. The number of studies generated in each database query was more than a thousand, and most of them were not related to the research. Therefore, it was necessary to improve the search string syntax to ensure more accurate results. After updating the search string syntax and applying it again on the databases sources, the resulting studies were then classified. Classification was made by titles, keywords, and abstracts for relevance to the research. Then those selected were studied thoroughly.

The selection process of relevant studies is divided into two parts. The first part is the initial selection of studies based on the satisfaction of inclusion criteria, and this was done by reading the titles and abstracts of papers. The second part is selection of the final papers from the list of the initially selected papers, provided that they meet the quality assessment criteria. This has been done by reading entire papers to determine whether or not they are strongly relevant to our work.

3.2.7 Quality Assessment Criteria

We adapted the quality assessment criteria proposed by Nabil et al. [23]. A study which obtains a total score of less than 3 is then rejected from the selected studies. The detail of quality assessment criteria is listed in Table 3.6.

Table 3.6 Quality Assessment Criteria

Criteria	Notes
Are the findings and results clearly stated in the paper?	Yes = 1, No = 0
Is there any empirical evidence on the findings?	Yes = 1, No = 0
Are the arguments well presented and justified?	Yes = 1, Partially =0.5, No = 0
Is the paper well referenced (i.e. article references from various journals and peers reviewed conferences)?	Yes = 1, Partially =0.5, No = 0

3.2.8 Extracting Data and Findings

At the beginning, we used Excel to arrange the selected papers after applying the search string in selected databases. We used two sheets in an Excel file—one sheet for the included papers and the other for the excluded papers. On the included papers' sheet, we also created different columns. Each column represents one of the categories of the Sommerville requirement engineering practices. This helps to identify each paper by the practices it covers. Figure 3.3 shows the selection criteria of the papers obtained after applying the search string in each database source.

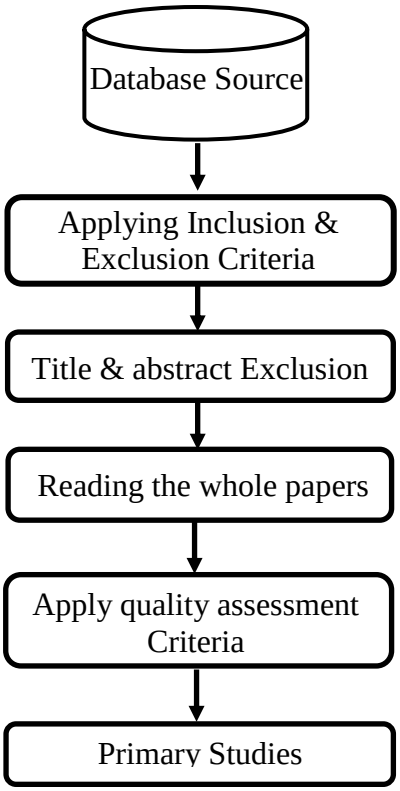

Figure 3.3 Study Selection Criteria

In this section, we present the results of SLR. A number of studies have passed the initial phase of SLR. Table 3.7 shows the distribution process of selecting the primary studies on the chosen research sources. For instance, in ScienceDirect, we found 98 articles from 585 studies based on their relevance as reflected in the abstract and keywords. These selected papers are farther reduced by applying the quality assessment criteria. Thus 98 papers were reduced to 29 which are considered to be the primary studies in the ScienceDirect database. The 29 articles were thoroughly read and analyzed with our research questions in view so as to identify the practices adopted in those papers. Appendix A (List of Primary Studies) is listed the primary studies selected for the review.

Table 3.7 Distribution of primary studies based on research sources

Research Sources	Total result	Initial Selection	Primary Studies
IEEE	1031	139	41
ACM	569	92	26
Science Direct	585	98	29
Total	2185	329	96

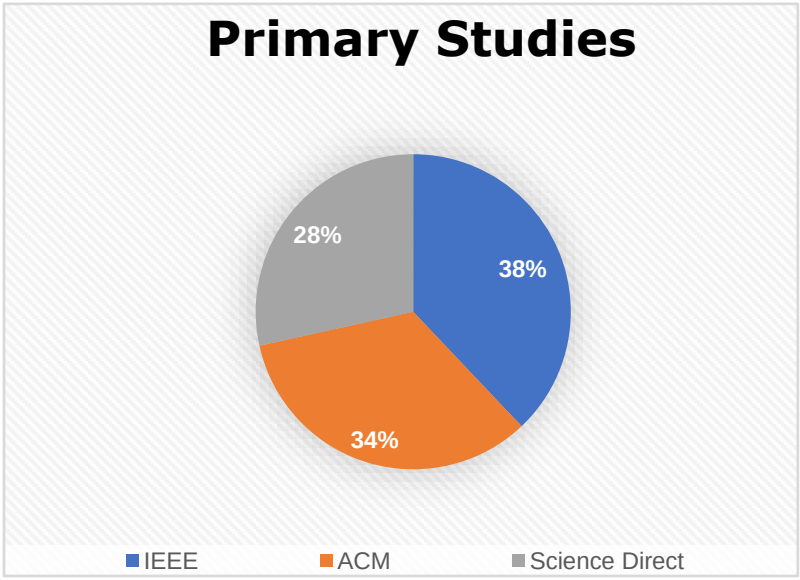

Figure 3.4 Research sources of selected studies

Second, the primary studies have been shown based on the publication channel in Table 3.8 and Figure 3.5. There are 63 primary studies that have been published in conferences, whereas 33 primary studies have been published in journals. This shows that most of the researchers who are interested in security requirement engineering have the chance of publishing their work in the conference or journal.

Table 3.8 Distribution of primary studies based on publication channel

Publication channel	Amount	%
Journal	33	34.02
Conference	63	64.94
Total primary studies	96	100

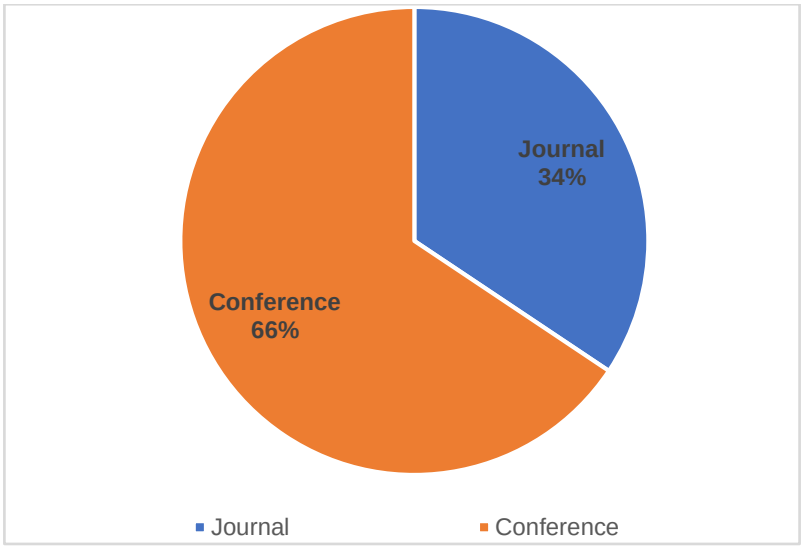

Figure 3.5 Selected studies based on publication channel

We used Sommerville classification of practices in this study to classify the identified papers. Table 3.9 shows the core categories of software security practices which have been identified during SLR. These practices can be classified as follow: security requirements documentation, security requirements elicitation, security requirements analysis and negotiation, describing security requirements, security system modelling, security requirements validation, security requirements management, and security requirements engineering for critical systems. In addition, Table 3.9 and Figure 3.6 show the number of studies related to each category of security practices at requirement phase.

Table 3.9 The classification of security practices at requirement phase

No.	Category	No. of papers in IEEE	No. of papers in ACM	No. of papers in ScienceDirect	Total # of Papers	%
1	Security Requirements Documentation	24	19	9	52	53.61
2	Security Requirements Elicitation	40	37	20	92	94.85
3	Security Requirements Analysis and Negotiation	41	30	19	90	92.78
4	Describing Security Requirements	10	10	7	27	27.84
5	Security System Modelling	24	15	13	52	53.61
6	Security Requirements Validation	18	22	10	50	51.55
7	Security Requirements Management	9	6	6	21	21.65
8	Security Requirements Engineering for Critical Systems	8	7	6	21	21.65

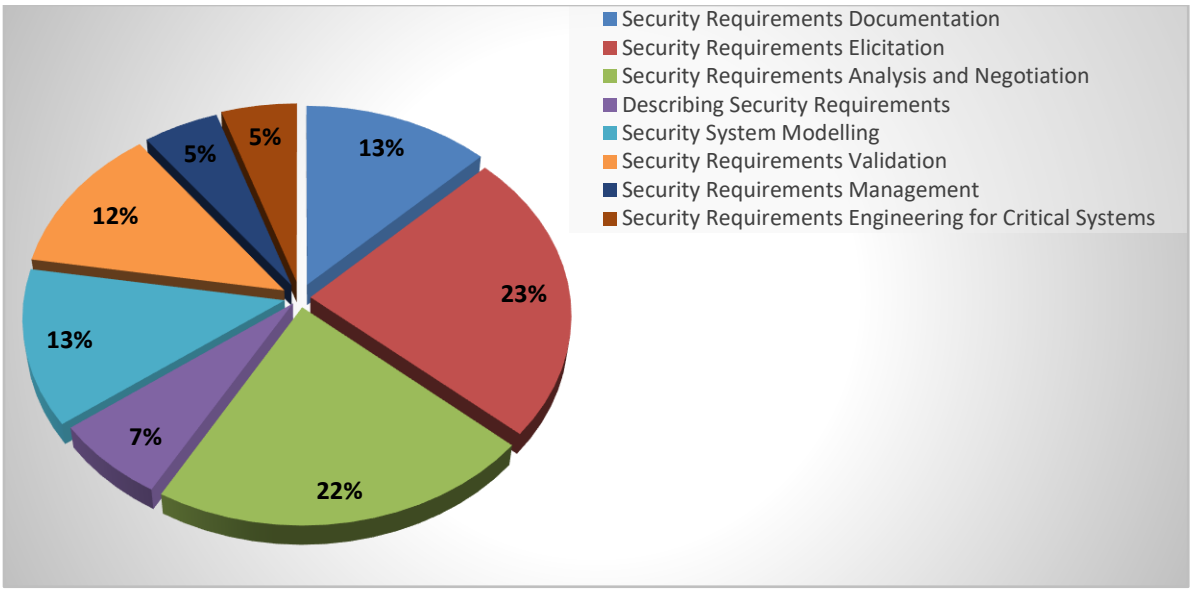

Figure 3.6 Classification of papers based on security practices category

3.3 Sommerville Practices

In this section, we modify the general requirement engineering practices of Sommerville to be security practices instead of general requirement engineering practices [4]. We have modified 66 RE practices to RE security practices as shown in Appendix B (Altered Sommerville RE Practices to SRE practices).

3.4 Building the Questionnaire

After getting the security practices from SLR and after adapting Sommerville requirement engineering practices to develop security practices instead of general requirement engineering practices, we built the questionnaire as shown in Figure 3.7, which was later distributed to 10 organizations to obtain feedback and impressions about these practices and also to inquire about any additional security practices used in their organization. These organizations are developing different software and also provide several services to their customers such as decision support systems, security application

solutions, and market development with reports. Four of these organizations have a certificate on CMMI Maturity Level 3.

We selected 10 organizations from all over the world. They have branches in Saudi Arabia, Egypt, and Canada. These organizations provide different services to their customers such as mobile applications, accounting and administrative systems, ERP systems, intelligent vehicle tracking systems, and IT security services. The results of the questionnaires have been used in the development of RESMM.

In the questionnaire, the respondent organizations were asked about the security practices they use during the requirement phase to obtain feedback, impressions about these practices, and information about any additional security practices used in the organizations. We have removed all those RE security practices which were not used by the organizations. In other words, we have removed the practices that were given the value "zero" by the ten organizations since these practices are never used by those organizations. The results of the questionnaire have been used in the development of RESMM. Figure 3.7 shows the structure of the questionnaire.

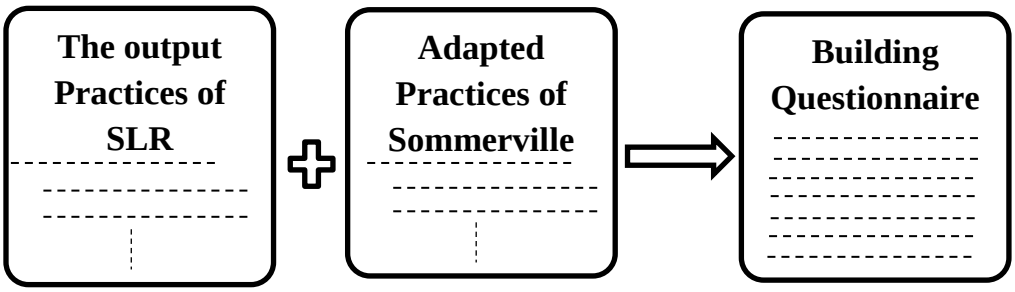

Figure 3.7 The structure of the Questionnaire

3.5 Maturity Model Development

The development of the RESMM is the fundamental task of this thesis. Security maturity model has been adopted based on several security practices published in various studies as obtained from SLR as well as the security practices deployed in different software industries. After the process of collecting security practices, these practices are used in the development of RESMM. Every category of RESMM contains some security practices which are assessed by SCAMPI appraisal [20].

The RESMM followed CMMI structure. CMMI contains different process areas. Each process area has its own goals which can be either generic goals or specific goals. Each goal has its own practices/activities to achieve that goal as shown in Figure 3.8.

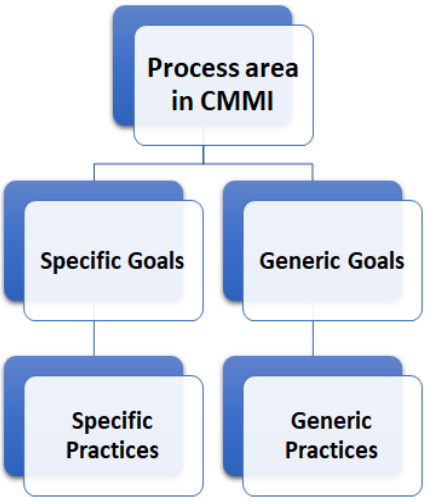

Figure 3.8 CMMI process area structure

3.6 Case Study

A case study can be used to obtain more information about practical examples from the real world [61]. In this way, it is possible to clarify unclear information or measure the impact of a certain phenomenon, taking into account that every step to investigate the role of something in real life will ultimately result in optimization of results.

For our purposes, a case study can be done by either meeting with organization personnel or through completion of an online form. In our case, we opt for an online form since most of the organizations are abroad. At the same time, there are some concerns that need to be considered, such as understanding the rationale of this research project, the awareness of the organization about the topic of this research, organizations' willingness to partake in this study and provide the requested information (security practices used in their organizations), and their knowledge of how to utilize the appraisal tool properly.

CHAPTER 4

RESULTS

This chapter explains the results of the SLR conducted as a method to pinpoint security practices at the requirement phase. In addition, this chapter shows the outcomes of the questionnaire which we administered to 10 organizations. The protocol of SLR has been explained in the previous chapter (Chapter 3.2). At the end of this chapter, a summary is provided.

4.1 Analysis and Results

The last process of the SLR is to analyze and present the results. Different practices are identified and classified into different Sommerville categories. This information is obtained and categorized based on the data extraction fields. However, this process should include analysis, which is required by the research questions. Then, the presented results need to address defined research questions. We use different tables for the purpose of presenting SLR results.

RQ1. What are the practices for documenting security requirements?

Documentation is a way that lets customers, system users, managers, and system developers interact with the system requirements. Some studies have been designed to improve the structure of documents to include security requirement in the documentation of the system. As noticed Table 4.1, a few studies have been done on the practices of security requirements documentation.

Table 4.1 Security Requirements Documentation Practices

Category	#	Ref.	SLR Security Practices
Security requirements document-ation	16	[62]	- Document the Conceptual security artefacts, - Document the security policies of the organization
		[63]	- Initiate templates for documentation of security requirement specification.
		[3]	- Security requirements should be expressed as positive statements and not negative statements. - Develop a security catalog
		[64]	- Defined security policies will improve software security. - Policies of the organization should be set clearly. - Define the impact of policies on stakeholders. - Check and update security policies whenever there is any change in the Org. - Document changes of policy. - Enhance security by implementing overall legislative and regulatory policies.
		[65]	- Security policy documents will help in indicating the laws, rules, and practices to manage sensitive information.
		[40]	- Develop organizational policy document. - Identify resources and trust boundaries - Identify resources capabilities and link them to roles. - Determine profiles of attackers. - Document logistics aspects that are available in the Org.
		[66]	- Document the security-related procedures (Management) - Document security configuration items of the system - Determine the standard procedure
		[67]	- Security document should include a Security Target and Protection Profile.
		[57]	- Use a standard for documenting security requirements
		[68]	- Security document should include a Software Security Authorization Agreement
		[58]	- Use standards for security-related Coding

		[69]	- Use a standard called ISO 15408 that contains Common Criteria for Security Evaluation of Information Technology.
		[70]	- Security requirements specification understandability. - Support for non-security experts.
		[71]	- Security statements Document
		[72]	- Develop a security policy with a separation of duties.
		[73]	- Identifying Document Interdependencies - Develop a security requirements category hierarchy.

RQ2. What are the practices for eliciting security requirements?

This refers to the process of capturing the security requirements of the system by connecting with customers, system users, and others who are interested in the development of the system. Studies have shown various methods and practices for eliciting security requirements of systems. As noticed in Table 4.2, most studies have been done in this area.

Table 4.2 Elicitation Practices of Security Requirements

Category	#	Ref.	SLR Security Practices
Security requirements elicitation	20	[62]	- Identify the security policy of the organization. - Dictate overall security measures. - Perform security assessment on the system.
		[74]	- Specify security needs by using SDA: secure development of applications.
		[75]	- Attack path analysis: to identify internal vulnerabilities. - Perform security assessment. - Identify what should be implemented on safety system security capabilities.
		[76]	- Build threat models.
		[1]	- Initiate a model syntax checker in order to indicate security protocols.

		[63]	- Determine the level of granularity. - Identify the system based on business pattern. - Identify the system based on application pattern. - Determine potential threats at the business level. - Determine potential threats at the application level. - Identify and evaluate threats. - Determine types of attackers and potential attacks. - Assess attack impact. - Evaluate and prioritize security risks. - Identify the system behavior whenever there is an attack. - Specify security requirements.
		[3]	- Elicit adequate security requirements. - Analyze the assets to be protected. - Analyze the threats from which assets should be protected. - Consider security during the elicitation of software system requirements using problem frames. - Identify security requirements with the aid of previous security knowledge.
		[77]	- Analyze the context of organization with respect to the operational environment. - Derive the dependencies of functional requirements with security requirements and trust requirements.
		[65]	- Analyze clearly the possible security risks in order to be reduced. - Identify possible security threats which are relevant to the security requirements by using misuse cases and attack trees. - Identify security violation scenarios.
		[78]	- Use the concept of extended constraint to represent the security concept. - Use the extended Tropos concepts with security in mind which consist of dependency, goal, task, resource, and capability.

		[79]	- Elicit the relevant security and privacy properties of the system. - Analyze of the critical areas to identify new concepts. - Link the new concepts with the threats and issues that connect to critical areas.
		[80]	- Perform a detailed risk assessment and an understanding of regulatory and compliance requirements. - Derive security requirements from business objectives. - Assess high-level risks and use cases related to security threats. - Incorporate prioritization of security vulnerabilities as part of the software release and defect repair planning.
		[40]	- Consider two perspectives: the black-hat and the white-hat. - Elicit the abuse cases for the system by using attack patterns.
		[81]	- Define security requirements by using the i* framework. - Identify threats and attacks on the requirements and the data repository.
		[66]	- Comprise the function of software security. - Cover the security environment. - Consider operation of security environment. - Identify general and specific security requirements.
		[82]	- Do processes of separate risk analysis with considering safety and security respects.
		[83]	- Analyze systems for potential insider threats.
		[57]	- Identify the process of information security and the requirements that are related to information security by using the history information of the product and/or service.
		[68]	- Identifying associated stakeholders - Assess the impact of operation with respect to security.
		[58]	- Identify security-sensitive assets - Formulation of abuser stories.

		[84]	- Identify sensitive system resources. - Identify entities that are possible threats to system. - Identify persons who may be threats to system.
		[85]	- Identify goals. - Derive security requirements from the goals.
		[70]	- Develop misuse scenarios and potential threats to the system. - Usability of security requirements specifications.
		[86]	- Identify the stakeholders of SRE process. - Elicit the goals of requirement.
		[87]	- Identify security goals. - Discover security goals - Organize security goals related to the assets. - Capture relationships among goals.
		[71]	- State the misuse cases. - Derive potential outcomes from the stated misuse cases. - Determine possible threats.
		[72]	- Identify appropriate security goals. - Enumerate security goals based on assets in the system. - Elicit possible harm (threat descriptions).
		[73]	- Use a goal-driven requirements strategy to elicit SR. - Identifying security requirements attributes. - Identifying requirements interdependencies - Perceive related risks in the operational environment.

The preceding practices are the existing security practices in the literature. We have identified those practices under the category of elicitation of security requirements since this helps in the elicitation process of security requirements. These practices are considered as a part of the questionnaire administered to the targeted organizations. After that, and based on the responses of sample organizations, the agreed practices are incorporated in the building process of RESMM, as shown in Chapter 5.

RQ3. What are the practices for analysis and negotiation of security requirements?

After doing the process of security requirements elicitation, we come up with an initial set of security requirements which need to be analyzed for conflicts, inconsistencies, omissions, or overlap. So, in this practice, the developer tries to solve those issues by negotiating with system stakeholders in order to agree on a set of system security requirements. Some studies address how to analyze the agreed security requirements of the system. As noticed in Table 4.3, many studies have been done in this area.

Table 4.3 Security Requirements Analysis and Negotiation Practices

Category	#	Ref.	SLR Security Practices
Security requirements analysis and negotiation	28	[62]	- Use the model-based paradigm to analyze the system needs and requirements. - Analyze use cases to determine a set of possible threats. - Specify minimal set of rights for each access control role. - Identify proper countermeasures for every security threat. introducing a structure mandatory security measure.
		[88]	- Threat modeling: Analyzing the probable attacks or threats to a system in a given context.
		[89]	- Analyze access control requirements by using AuthUML. - Analyze the authorization requirements.
		[75]	- Attack Analysis to identify what are the accessible discipline to the system.
		[76]	- Detect potential vulnerabilities of system by gathering system information from several perspectives.
		[63]	- Identify conflicts due to composition or integration scenarios. - Remove redundancies and refine ambiguous requirements. - Classify elicited security requirements. - Identify inclusion/exclusion relationships among the requirements.

		[3]	- Identify and analyze the system assets, threats, vulnerabilities and requirements. - Make security requirements as adequate as possible.
		[64]	- Hand out revised policies to all related stakeholders. - Make sure about the awareness of security policies. - Conduct comprehensive risk analysis to improve software security.
		[79]	- Analyze the context of the Org. within the environment of the system. - Determine the domain actors. - Determine the dependencies of actors with other actors. - Analyze functional dependencies with security and trust requirements.
		[65]	- Identify all potential security threats using misuse cases and attack trees. - Identify misuse cases (beside the normal use cases). - Prioritize misuse cases. - Create an attack tree which will determine the scenarios of intrusion.
		[78]	- Have an understanding of the security problems of organization by analyzing existing setting of organization. - Describe the operational environment of the system with related functions and security requirements.
		[79]	- Consider the concepts from organization areas. - Define a set of notations and security concepts during the analysis process of software development. - Consider related security and privacy properties, threats, and risks
		[90]	- Risk identification.
		[40]	- The definition of use scenarios.
		[81]	- Goal/Soft-goal analysis: use the security policy document to analyze the goals of the organization.

		[66]	- Conduct threat and hazard analysis. - Determine security configuration items. - Identify the performance of software configuration, security functional.
		[63]	- Apply separate risk analysis for safety of software and software security. - Identify interactions that could exist between security and safety requirements - Identify measures that has to be implemented, changes in the software, and evaluate the effects of the identified measures.
		[83]	- Conditional Reachability Analysis. - Log-trace Reachability Analysis. - Determine the state of the system before, under, and after the attack.
		[91]	- Define information security requirements: focuses on the security issues. - Analyze available environment options - Risk analysis
		[68]	- A risk assessment taxonomy
		[58]	- Abuser story Risk assessment
		[84]	- Analysis of external and internal security threats. - Analysis of Security Risks
		[85]	- Perform risk assessment
		[70]	- Relate requirement artifacts to test case artifacts
		[86]	- Identify security risks based on possible influences of the security threats.
		[71]	- Asses security solutions
		[72]	- Revise possible Harm
		[73]	- Risk assessment taxonomy

The preceding practices are the existing security practices in the literature. We have identified those practices under the category of analysis and negotiation of security requirements since this helps in the analysis of the identified security requirements. These practices are considered as a part of the questionnaire administered to the targeted organizations. After that, and based on the responses of sample organizations, the agreed practices are incorporated in the building process of RESMM as shown in Chapter 5.

RQ4. What are the practices for describing security requirements?

The identified security requirements should be described in a concise, understandable, and unambiguous manner. As noticed in Table 4.4, a few studies have been done in this area.

Table 4.4 Describing Security Requirements Practices

Category	#	Ref.	SLR Security Practices
Describing Security requirements	12	[62]	- Use the simple micropattern textual template. - Security requirements have to be correct, consistent and complete.
		[3]	- Security requirements need to be adequate as possible. This means, they should be explicit, precise, complete and non-conflicting with other requirements.
		[40]	- Describe abuse cases by examples.
		[81]	- Use metadata to show a prototype model for data security - Describe stakeholder concerns and interests by using the Strategic Rationale (SR) model.
		[66]	- Use UML Diagrams to describe security processes. - Misuse Case Description Templates - Security Use Case Description Templates
		[67]	- Protection profiles should be unambiguous

		[92]	- Describe security requirements using metaclasses especially for common rational agent within systems.
		[68]	- Use a standard of DoD process for identifying information security requirements.
		[58]	- User stories using understandable language. - Determine security requirements as standard User stories
		[86]	- Use common terminology to define Requirements which are simple and non-technical jargon.
		[72]	- Describe the used security mechanisms to express security requirements such as ISO 15408
		[73]	- Dealing with Natural-Language Requirements. - Organize the concepts of problem domain and expresses them in natural language regulatory documents.

The preceding practices are the existing security practices in the literature. We have identified those practices under the category of describing security requirements since this helps in the analysis of the identified security requirements. These practices are considered as part of the questionnaire administered to the targeted organizations.

RQ5. What are the practices for modeling security requirements?

The idea of this practice is to build an abstract model for the system that includes the security aspect. This model shows the system environment with respect to security issues and the architecture model for the whole system. This model is considered a high-level model, which can help to reveal hidden requirements. There are different languages used for security modeling, such as Security Risk-Oriented BPMN, Secure TROPOS, KAOS Extension to Security, Misuse Cases, Mal-Activity Diagrams, UMLsec, and SecureUML. Some studies show how to model the system with security concerns. As noticed in Table 4.5, several studies have been done in this area.

Table 4.5 Modeling Practices of Security Requirements

Category	#	Ref.	SLR Security Practices
Security System Modeling	16	[62]	- Model attackers to the system - Model potential attacks (threats) to a system.
		[89]	- Modeling security requirement using UML
		[76]	- Establish a link between security requirements models and security implementation models.
		[3]	- Use problem frames to model security requirements
		[1]	- Use formal process algebra for modelling threats.
		[77]	- Identify the main stakeholders - Identify the objectives of stakeholders by using Actor modeling. - Specify services based on the identification of actors. - Identify actors delegating to other actors by using Permission delegation modeling. - Identify actors who possess the services by using Trust modeling.
		[65]	- Model the System's Environment with security consideration.
		[81]	- Modeling and rationale the environment of organization and its information system. - Use metadata to make a prototype model for system security. - Use Goal modeling diagram or Softgoal modeling diagram.
		[83]	- Model the real-world systems and provide an underlying semantics. - Specify models for system by defining a certain language.
		[67]	- Use a Data Dictionary.

		[82]	- Understand the environment of the system. - Recognize the risks that are related to each domain. - Define the environment and the scope and boundaries of the system. - Show the effects of system failure on the environment due to combination of accidental conditions. - Model partitioned architecture. - Annotate model of the system with security properties which are devoted to safety and security policies.
		[92]	- Manage the own security of each agent by identifying internal concepts required.
		[76]	- Use Graphical modeling approaches: - Semi-formal safety/security cases. - Goal structuring notation which is considered to be a graphical argumentation notation.
		[68]	- Modeling of system's environments and domain knowledge
		[58]	- Formulation of Abuser stories (Threat scenarios)
		[73]	- Using various GenOM modeling constructs to express SR.

The preceding practices are the existing security practices in the literature. We have identified those practices under the category of modeling security requirements since this helps in revealing hidden security requirements. These practices are considered as part of the questionnaire administered to the targeted organizations. After that, and based on the responses of sample organizations, the agreed practices are incorporated into the building process of RESMM as shown in Chapter 5.

RQ6. What are the practices for validating security requirements?

The collected security requirements, which are documented, need to be validated for conflicts, omissions, and ambiguities. As noticed in Table 4.6, few studies have been done in this area.

Table 4.6 Validating Practices of Security Requirements

Category	#	Ref.	SLR Security Practices
Security requirements validation	19	[62]	- Inspect the effects of each security countermeasure manually and link back the security countermeasure to the security requirements.
		[89]	- Validate authorization requirements compliance with separation of duty principle. - Security Requirements Artefacts Inspection.
		[75]	- Publish a report about cyber security for the system. - Penetrate testing to address what are the problems of technical security.
		[63]	- Software Testing Specification Template. - Internal validation by implementing an inspection of the particular security requirements. - External validation by estate review meetings with different actors who are involved in the developing process.
		[65]	- Use contracts items to validate security vulnerabilities.
		[81]	- Internal validation and external verification. - Carry out a verification process that involves checklists, peer reviews, or Fagan's methods. - Use graph-based approach to reveal and solve any conflicts which can be accrued to the specification of Access Control policies.

		[66]	- Perform a program for quality assurance that would support security activities. - Make sure about the regularity of the safety and the features of logical software security in the intended environment. - Perform a review for the quality of security. - Consider the mechanism that is involved in the software development by establishing traceability among these mechanisms. - Conduct periodic reviews.
		[57]	- Security features artefact of the system need to be reviewed and refined by using rigorous scientific evaluation methods with iterative cycle.
		[92]	- Generate a counter-example whenever there is requirements violations. - Use a model checking.
		[67]	- Simulate the platform of the systems by using prototype.
		[68]	- Cross-checked with the operational environment
		[58]	- Validate Security-related User stories directly by using integration testing with other User stories.
		[93]	- Use a Goal Oriented Requirements Engineering techniques for validating the completeness of security requirements and modelling stakeholder rationale, as well as building threat trees and modelling vulnerabilities and their effect.
		[69]	- Traceability and consistency checks between different kinds of UML models.
		[85]	- Inspect and validate requirements.
		[70]	- Misuse Test Cases - Requirement Test Cases - Threat Test Cases.
		[71]	- Security Requirements Test Cases - Misuse Cases Test Cases - Threat Test Cases

		[72]	- Construct a satisfaction argument. - Revise Application Business Goals & Quality Goals - Check the Security Goals against Threats, Assets and Business Goals.
		[73]	- Use testing procedures that can be used for checking to the compliance levels of the target system.

The preceding practices are the existing security practices in the literature. We have identified those practices under the category of validating security requirements since this helps in the validation of conflicts, omissions, and ambiguities of security requirements revealing hidden security requirements. These practices are considered as part of the questionnaire administered to the targeted organizations. After that, and based on the responses of sample organizations, the agreed practices are incorporated in the building process of RESMM as shown in Chapter 5.

RQ7. What are the practices for the management of security requirements?

This process is concerned with the change that could be effected in system security requirements. As noticed in Table 4.7, very few studies have been done in this area.

Table 4.7 Management Practices of Security Requirements

Category	#	Ref.	SLR Security Practices
Security requirements Management	6	[75]	- Select applicable security controls
		[94]	- Use the concepts of case-based management system which involve the knowledge-based management besides an artifacts management.
		[95]	- Use an extend Secure Tropos which involves a risk-driven goal-based process for managing security requirements.
		[96]	- Carry out a role-based access scheme.

			- Implement the principles of separation of duties.
		[73]	- Define the Management control policy for SR.
		[66]	- Define the configuration items of security into software requirements. - Assess the impact of any suggestion of changes. - Evaluate the procedures of operating for compliance with respect to the intentional use. - Analyze the risks of security which may affect the licensee and the system. - Introduce security mechanism to control the environment of software maintenance whenever there is change of data.

The preceding practices are the existing security practices in the literature. We have identified those practices under the category of management of security requirements since this helps in the validation of conflicts, omissions, and ambiguities of security requirements revealing hidden security requirements. These practices are considered as part of the questionnaire administered to the targeted organizations. After that, and based on the responses of sample organizations, the agreed practices are incorporated in the building process of RESMM as shown in Chapter 5.

RQ8. What are the practices of Security Requirements Engineering for Critical Systems?

This process is concerned with the systems that have to contain stringent reliability, availability, maintainability, safety, or security requirements. If these systems fail, the cost here would be very high, so the requirements engineering and system development processes must ensure that stakeholders have confidence in these systems. As noticed in Table 4.8, very few studies have been done in this area.

Table 4.8 Security Requirements Engineering Practices for Critical Systems

Category	#	Ref.	SLR Security Practices
Security Requirements Engineering for Critical Systems	4	[75]	- Select applicable security controls
		[96]	- Consistency of Definitions (Before the security requirements analysis of an organization go farther it have to be agreed about all relevant terms and definitions during the security analysis process.
		[68]	- Use a model for modelling the requirements domain with respect to security requisites and policies; - Perform risk assessment taxonomy. - Provide a process for aspect knowledge - Create Meta-knowledge about information of the system. - Interdependencies between entities of the system.
		[95]	- Use an extend Secure Tropos which involves a risk-driven goal-based process for managing security requirements.

4.2 Outcomes of Questionnaires:

Security Requirements Practices

This study uses the Sommerville classification of practices. This classification consists of core categories of software security practices which have been identified during SLR in addition to the adapting of Sommerville RE practices. These practices can be classified as follows: security requirements documentation, security requirements elicitation, security requirements analysis and negotiation, describing security requirements, security system modelling, security requirements validation, security requirements management, and security requirements engineering for critical systems. A questionnaire was developed and administered to 10 organizations to gauge their responses

toward those practices which are important. In that questionnaire, we asked them to follow the following structure:

For the following section, choose from the following four types of assessments for each SRE practice:

- ***High Perceived Benefits (H)*:** The practice is mandatory.
- ***Medium Perceived Benefits (M):*** The practice occurs often in the organization's software development but is not mandatory.
- ***Low Perceived Benefits (L):*** The practice is only used in certain situation.
- ***Zero Perceived Benefits (Z):*** The practice is never or rarely used in that organization.

In the questionnaire, the respondent organizations were asked about the security practices they use during the requirement phase to obtain feedback, impressions about these practices, and information about any additional security practices used in the organizations. The responses to the questionnaire came from different employees who worked in those organizations, including a technical team leader, an application development manager, and a systems analyst. These employees had worked in those organizations for nearly ten years. There was one submission from each organization, and there were ten total responses. We contacted the ten organizations via official email, and they forward our emails to the corresponding employees, who then answered the questionnaire. Table 4.9 shows the details of these organizations and the participants who filled the questionnaire.

Table 4.9 The details of the organizations and the participants who filled the questionnaire

No.	Org.	Location	Type of Services	Job Title	Experience
1	Organazation1	Saudi Arabia	Mobile Services	Systems Analysis	10
2	Organazation2	Saudi Arabia	Accounting and administrative systems	Senior Software Engineer	8
3	Organazation3	Saudi Arabia	ERP systems	Lead Software Engineer	6
4	Organazation4	Saudi Arabia	Business Solutions	Technical team leader	11
5	Organazation5	Saudi Arabia	Intelligent vehicle tracking systems	Systems Analysis	10
6	Organazation6	Saudi Arabia	Educational Services	Software Developer	7
7	Organazation7	Egypt	Commercial and manufacturing	Senior Systems Analysis	13
8	Organazation8	Canada	Security services	Systems Analysis	5
9	Organazation9	Yemen	Professional software services	Application Development Manager	7
10	Organazation10	Yemen	Commercial Services	Software Engineer	9

After receiving the responses from the ten organizations, we removed all those RE security practices which were not used by the organizations. In other words, we removed the practices that were given a value of “zero” by the ten organizations since these practices

are never used by those organizations. However, if one of the ten organizations gave the practice a score of "low" and the other nine organizations gave the value "zero," we included this practice in the development of RESMM since each organization represented 10% of the total number of organizations. The results of the questionnaire were used in the development of RESMM.

The responses of these ten organizations, including the security practices used by them, were integrated in the requirement phase of software development. We classified these practices into seven categories, following the Sommerville classification.

Table 4.10 Security Requirements Document Practices Chosen by Organization

ID	Practice	Type of Assessment			
		H	M	L	Z
SRD1	Define a standard security document structure	7	2	1	0
SRD2	Explain how to use the security document	7	1	2	0
SRD3	Define Security Definitions, Quality Gates and security policy of the organization.	10	0	0	0
SRD4	Make a separate information security policy such as: (Access Control Policy, Classification Policy, Backup Policy… etc.)	6	1	0	3
SRD5	Define and document a project's security bug bar. (Establish a minimum level of quality)	8	2	0	0
SRD6	Define Security Objectives Document.	0	0	0	10
SRD7	Define Security Requirements Rationale Document.	0	0	0	10

SRD8	Define Protection Profile documents	0	0	0	10
SRD9	Define Security Problem Definition Document which must contain the threats, assumptions, and conformance claims.	0	0	0	10
SRD10	Define Risk Assessment Document	0	0	0	10
SRD11	Include a summary of the security requirements	7	3	0	0
SRD12	Make a business case for the system with respect to .	10	0	0	0
SRD13	Define specialized security terms	8	2	0	0
SRD14	Make document layout readable	8	1	1	0
SRD15	Use languages simply and concisely to explain security requirement. (identification/authentication/authorization/immunity/privacy/ integrity)	7	3	0	0
SRD16	Help readers find information	6	2	2	0
SRD17	Make the document easy to change	7	2	1	0

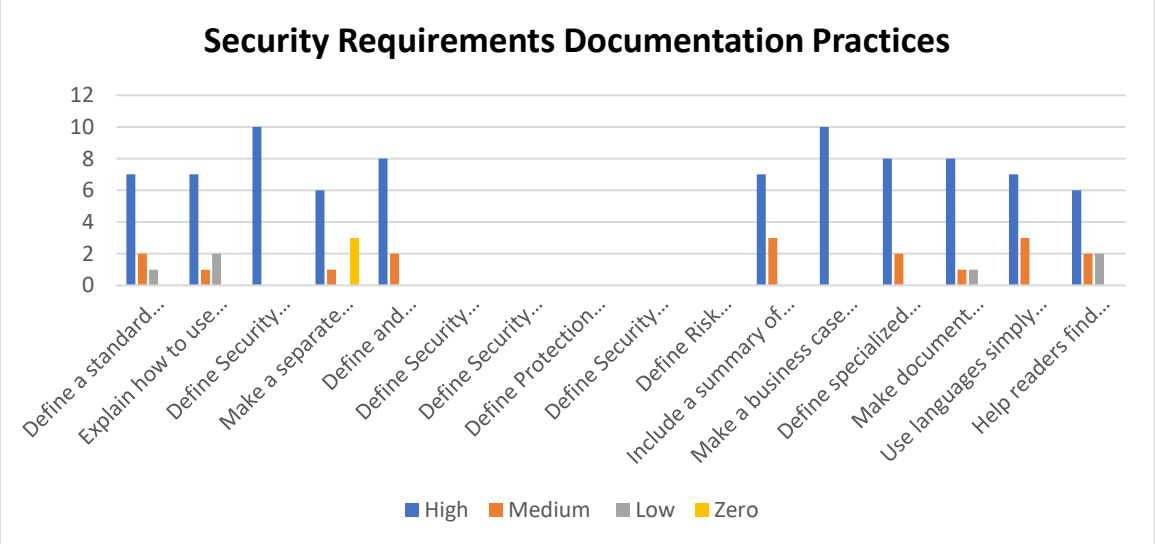

Figure 4.1 Used Security Requirements Documents Practices during Questionnaire

We obtained feedback from the organizations targeted by the questionnaire regarding some of these practices. For example, they referred to some of the security requirements documentation practices, describing them as repetitive and needing to be combined into unified practices. For instance, there is a standard to be followed for a security requirements document which shares the same structure as this document. Thus, there is no need to mention the parts of this standard documentation, such as the security objectives document, security requirements rationale document, protection profile documents, security problem definition document, risk assessment document, etc. Also, we found this part of the feedback helpful, so we combined practices under the definition of "standard for security requirements documents." This led us to reduce the number practices of security requirements documents from 17 to 12 as shown in Figure 4.2.

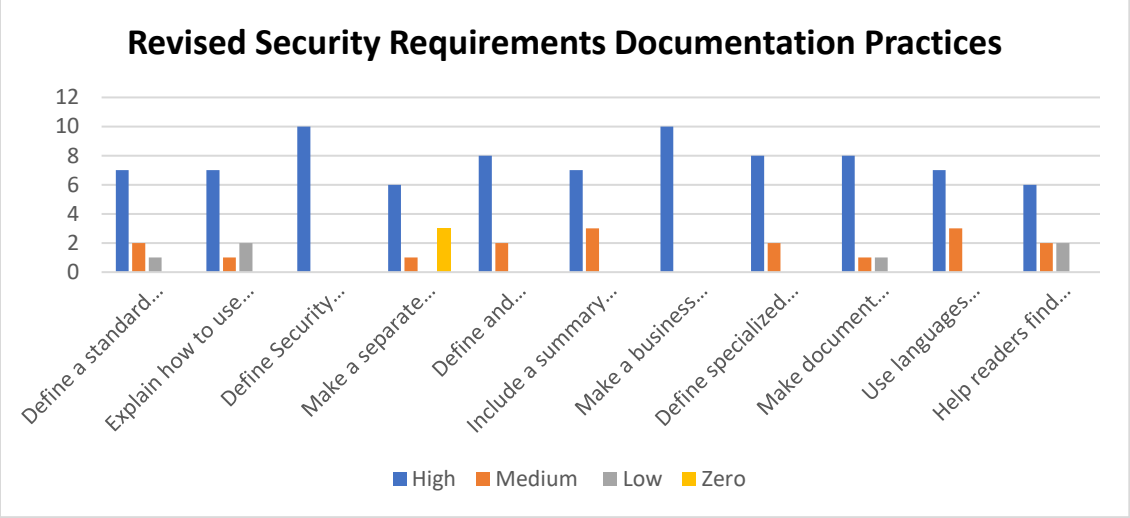

Figure 4.2 Security Requirements Documents Practices Chosen by Organizations

The next series of tables **(**Table 4.10 through Table 4.16**)** contain the practices which have been chosen by the targeted organizations. We have removed the practices that were given the value "zero" and show here only the practices that are used by the organizations.

Appendix C (Requirement Engineering Security Maturity Model Practices) reflects all the practices noted in the questionnaire, Excluding the practices given the value "zero."

Table 4.11 Security Requirements Elicitation Practices Chosen by Organizations

ID	Practice	Type of Assessment			
		H	M	L	Z
SRD1	Assess System Feasibility with respect to security	5	3	2	0
SRD2	Demonstrate of exploitability	6	2	2	0
SRD3	Be sensitive to organizational and political consideration	10	0	0	0
SRD4	Identify and consult system stakeholders (to agree upon a common set of security definitions, definition of the organizational security policies and the security vision of the IS.)	10	0	0	0
SRD5	Identify vulnerable and/or critical assets.	8	2	0	0
SRD6	Identify security objectives and dependencies.	5	3	2	0
SRD7	Identify threats and develop artifacts. (such as misuse cases or attack trees diagrams or UMLSec use cases and classes or sequence/state diagrams)	8	2	0	0
SRD8	Record security requirements sources (Identify Resources and Trust Boundaries)	4	3	3	0
SRD9	Define the system's operating environment (Specify Operational Environment)	5	2	3	0
SRD10	Identifying User Roles and Resource Capabilities	7	2	1	0
SRD11	Use business concerns to drive security requirements elicitation	7	1	2	0

SRD12	Identify and consult security experts	6	2	2	0
SRD13	Select an elicitation method using a systematic tradeoff analysis approach to elicit security	4	4	2	0
SRD14	Look for domain constraints	6	2	2	0
SRD15	Record security requirements rationale	4	3	3	0
SRD16	Collect security requirements from multiple viewpoints	5	2	3	0
SRD17	Prototype poorly understood security requirements	7	2	1	0
SRD18	Use scenarios to elicit security requirements	8	1	1	0
SRD19	Define operational processes	3	3	4	0
SRD20	Reuse security requirements	3	4	3	0

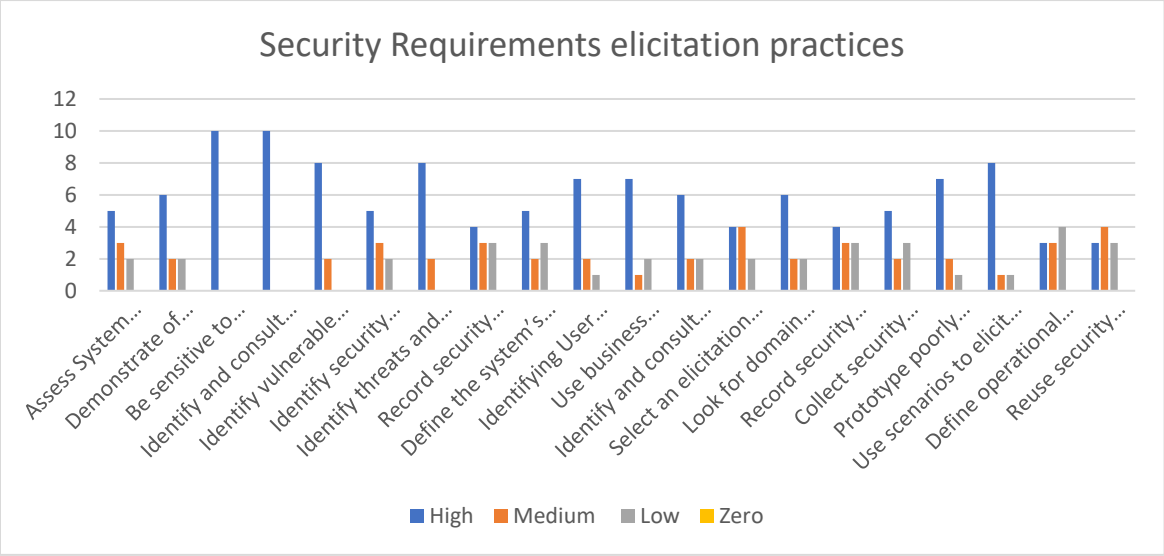

Figure 4.3 Security Requirements Elicitation Practices Chosen by Organizations

Table 4.12 Security Requirements Analysis and Negotiation Practices Chosen by Organizations

ID	Practice	Type of Assessment			
		H	M	L	Z
SRA1	Define security of system boundaries	10	0	0	0
SRA2	Use checklists for security requirements analysis	7	2	1	0
SRA3	Perform Security & Privacy Risk Assessment	9	1	0	0
SRA4	Negotiate quality gates with different stakeholders	8	2	0	0
SRA5	Provide software to support negotiations	9	1	0	0
SRA6	Ensure access requirements are consistent, complete and conflict-free.	7	3	0	0
SRA7	Prioritize security requirements	8	2	0	0
SRA8	Classify security requirements using a multi-dimensional approach	6	2	2	0
SRA9	Use interaction matrices to find conflicts and overlaps	7	3	0	0
SRA10	Review Security Requirements (review security requirements are the confrontation of analysis between Analyst and the Security Team analysis)	8	2	0	0

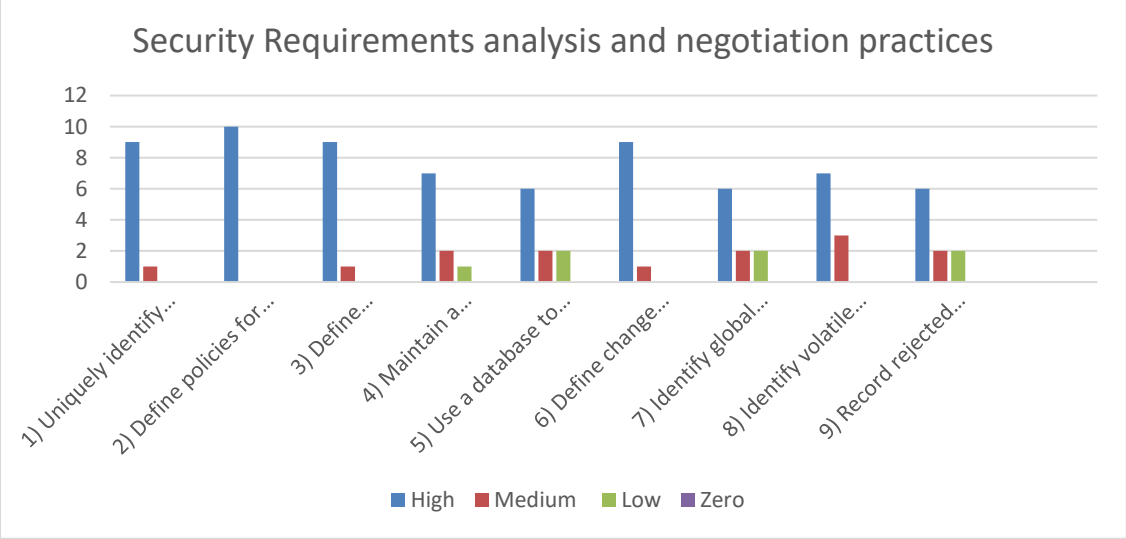

Figure 4.4 Security Requirements Analysis and Negotiation Practices Chosen by Organizations

Table 4.13 Practices of Describing Security Requirements Chosen by Organizations

ID	Practice	Type of Assessment			
		H	M	L	Z
DSR1	Define standard templates for describing security requirements	7	2	1	0
DSR2	Use languages simply and concisely	10	0	0	0
DSR3	Be adequate as possible (explicit, precise, complete and non-conflicting)	10	0	0	0
DSR4	Use diagrams appropriately	10	0	0	0
DSR5	Describe abuse cases by examples	5	3	0	2
DSR6	Describe a prototype model for data security based on metadata	8	2	0	0
DSR7	Supplement natural language with other description of security requirement	7	2	1	0
DSR8	Specify security requirements quantitatively	6	2	2	0

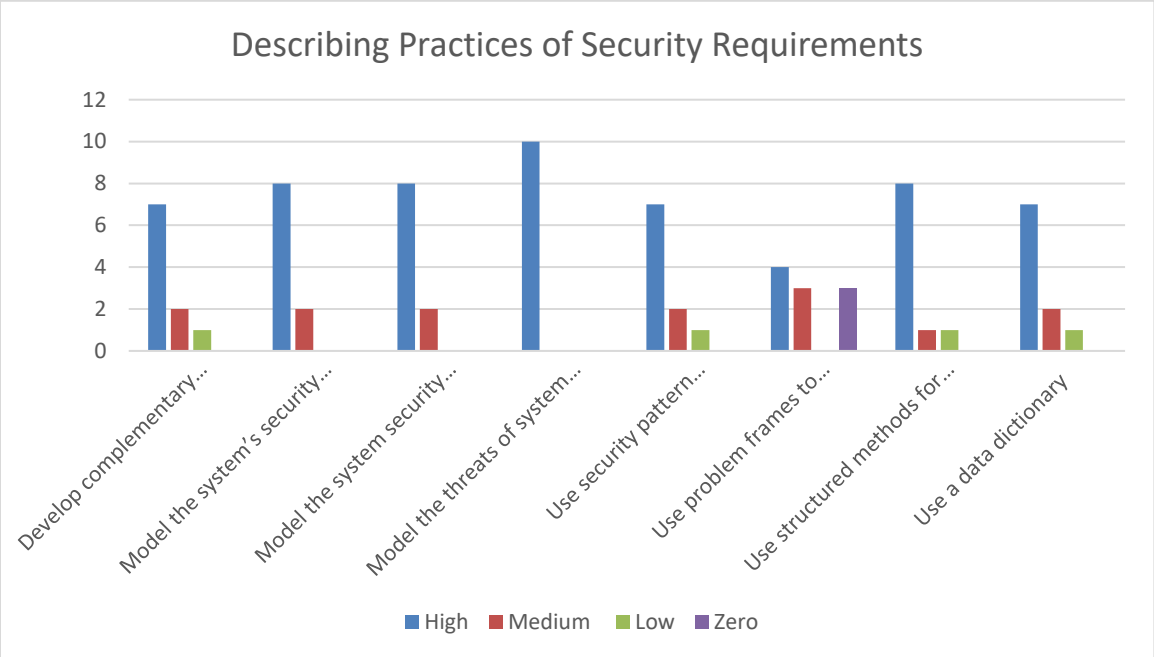

Figure 4.5 Practices of Describing Security Requirements Chosen by Organizations

Table 4.14 Practices of Modelling Security Requirements Chosen by Organizations

ID	Practice	Type of			
		H	M	L	Z
SRM1	Develop complementary system models with respect to security	7	2	1	0
SRM2	Model the system's security environment	8	2	0	0
SRM3	Model the system security architecture	8	2	0	0
SRM4	Model the threats of system	10	0	0	0
SRM5	Use security pattern template to model security requirements	7	2	1	0
SRM6	Use problem frames to model security requirements	4	3	0	3
SRM7	Use structured methods for system security modelling	8	1	1	0
SRM8	Use a data dictionary	7	2	1	0
SRM9	Document the links between stakeholder requirements and system models	6	2	2	0
SRM10	Clearly define the properties that we hope to prevent attackers from violating.	9	1	0	0

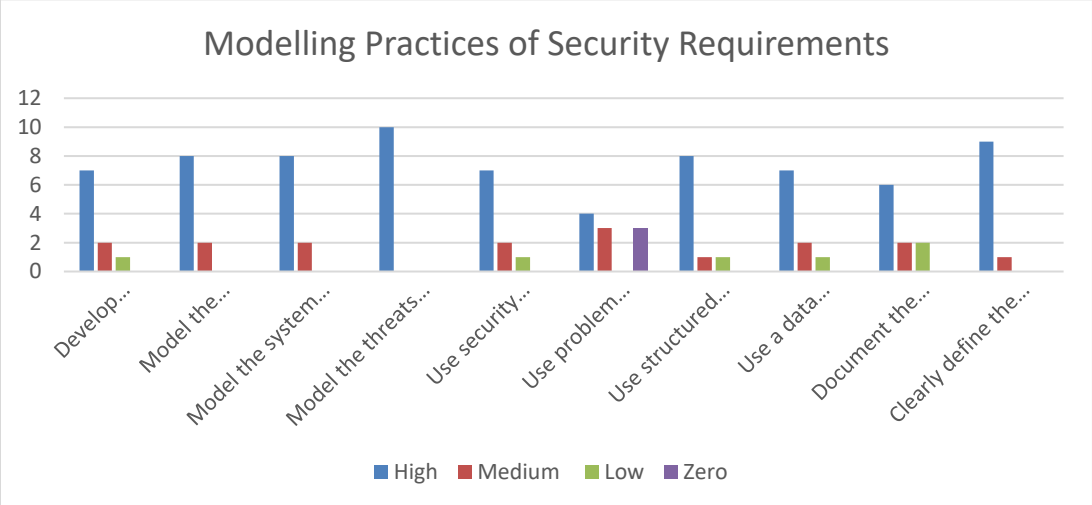

Figure 4.6 Practices of Modelling Security Requirements Chosen by Organizations

Table 4.15 Practices of Validating Security Requirements Chosen by Organizations

ID	Practice	Type of Assessment			
		H	M	L	Z
SRV1	Check that the security requirements document meets your standards	10	0	0	0
SRV2	Organize security requirements inspections	10	0	0	0
SRV3	Use multi-disciplinary teams to review security requirements	8	1	1	0
SRV4	Define validation checklists	9	1	0	0
SRV5	Use prototyping to animate security requirements	8	2	0	0
SRV6	Perform periodic security assessments and review the quality of security activity	10	0	0	0
SRV7	Write a draft user manual	8	2	0	0
SRV8	Propose security requirements test cases	8	2	0	0
SRV9	Paraphrase system security models	7	2	1	0

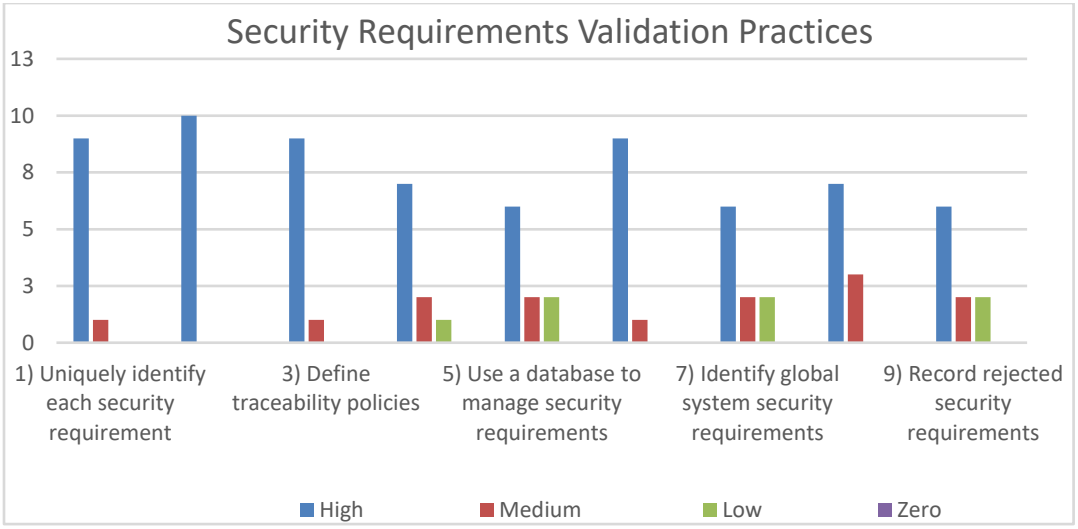

Figure 4.7 Practices of Validating Security Requirements Chosen by Organizations

Table 4.16 Management Practices of Security Requirements Chosen by Organizations

ID	Practice	Type of Assessment			
		H	M	L	Z
MSR1	Uniquely identify each security requirement	9	1	0	0
MSR2	Define policies for security requirements management	10	0	0	0
MSR3	Define traceability policies	9	1	0	0
MSR4	Maintain a traceability manual	7	2	1	0
MSR5	Use a database to manage security requirements	6	2	2	0
MSR6	Define change management policies	9	1	0	0
MSR7	Identify global system security requirements	6	2	2	0
MSR8	Identify volatile security requirements	7	3	0	0
MSR9	Record rejected security requirements	6	2	2	0
MSR10	Manage risks of requirements from laws and regulations	7	2	1	0

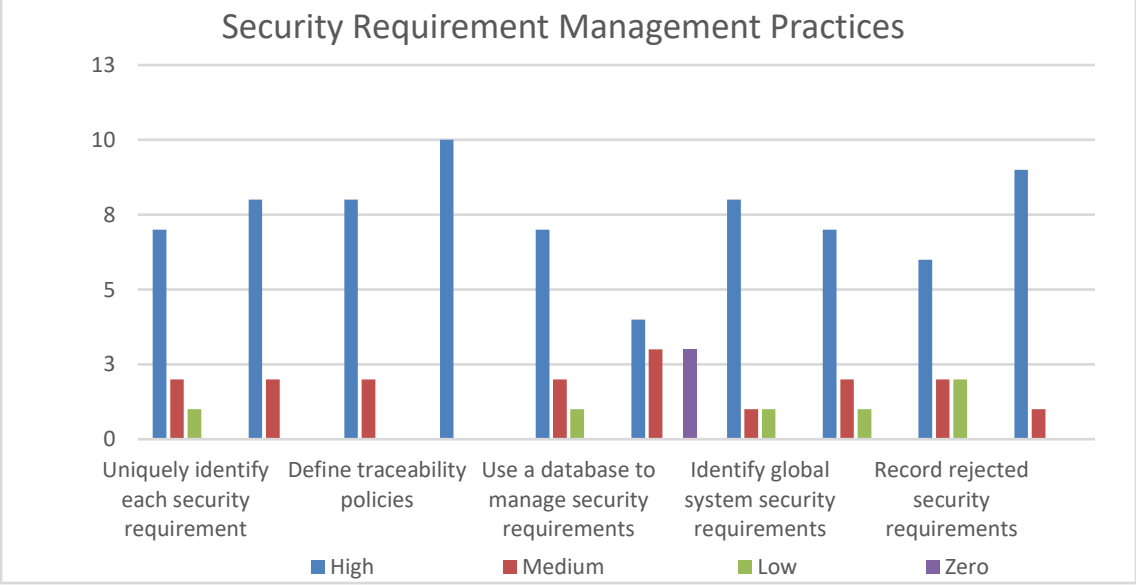

Figure 4.8 Management Practices of Security Requirements Chosen by Organizations

The outcome of the conducted questionnaire is the total number of security practices that have been used in different organization. We have removed all the practices that were given the value "zero" by the ten organizations since these practices are never used by those organizations. Thus, the remaining practices, which were not given the value "zero," have been used to build RESMM. Table 4.17 shows numbers of security practices for each category. There are different goals such as security requirements document goal, security requirements elicitation goal, etc. Each of these goals has its own security practices.

Table 4.17 The outcome of the conducted questionnaire (security practices)

No.	Category	No. of Security Practices
1	Security Requirements Document	12
2	Security Requirements Elicitation	20
3	Security Requirements Analysis and Negotiation	10
4	Describing Security Requirements	8
5	Security System Modelling	10
6	Security Requirements Validation	9
7	Security Requirements Management	10
	The Total number of security practices	79

CHAPTER 5

REQUIREMENT ENGINEERING SECURITY

MATURITY MODEL

5.1 Introduction

This chapter summarizes the development process of Requirements Engineering Security Maturity Model (RESMM). First, we present the structure of proposed RESMM, then a suitable assessment tool is discussed which can be used to measure requirements security maturity of organizations. After that, the RESMM is applied in the real software industry via a case study approach, and feedback obtained from the case study organization is taken into consideration to improve RESMM. Figure 5.1 shows the development flow of the RESMM.

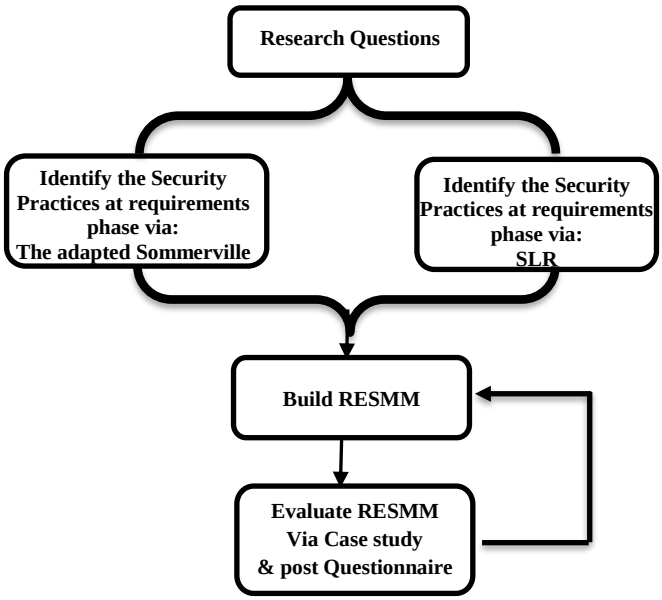

Figure 5.1 RESMM Development

Figure 5.2 shows the full process of development of RESMM with the utilized SCAMPI measurement. RESMM consists of different security practices that have been conducted based on Sommerville practices and organization practices via questionnaire. The outputs of the RESMM, which are security requirements practices classifications, are appraised to assess the organization practices with respect to security.

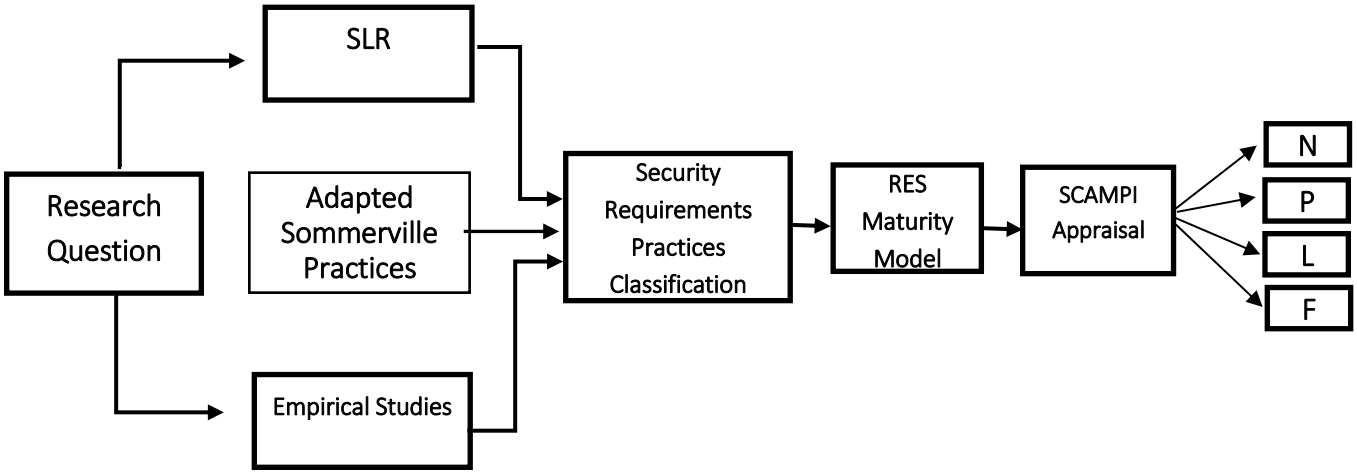

Figure 5.2 The full process of the development of RESMM

5.2 Structure of RESMM

This section explains the RESMM structure. RESMM is designed to assist software development organizations in specifying the requirements for secure software in a better way. RESMM structure is motivated by the structure of CMMI. We employed the concepts of specific goals, specific practices, and a measurement of maturity capability. In RESMM, we have specified generic practices based on the data we obtained from two sources, i.e. previous studies and software industry experience. In addition, we have specified the practices in order to achieve the generic goals. Figure 5.3 shows the structure of our process area (RESMM) vs the CMMI process area.

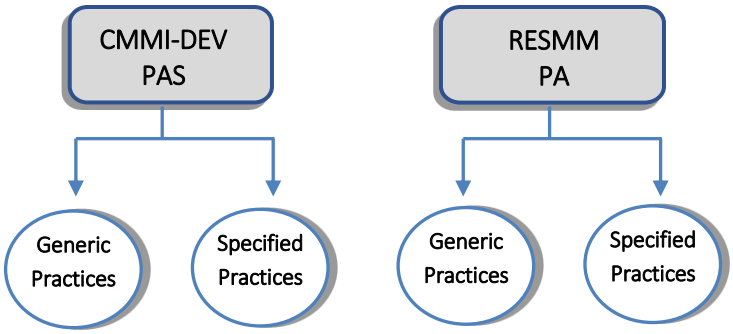

Figure 5.3 CMMI process area structure vs RESMM structure

In our case, general goals of the RESMM process will be security requirement document, security requirements elicitation, security requirement analysis and negotiation, describing security requirement, modeling of security requirement, security requirement validation, and security requirement management. Furthermore, each of these general goals has its own security practices in order to achieve that goal. Figure 5.4 shows the structure of RESMM.

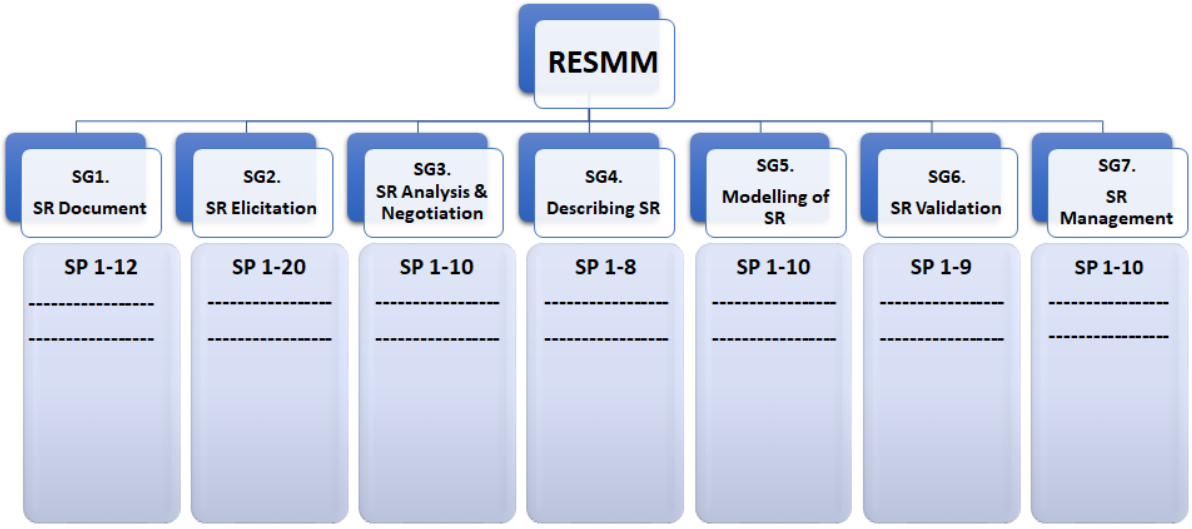

Figure 5.4 RESMM Structure

5.2.1 Specific Goal Component

RESMM consists of seven specific goals. Each goal consists of specific security practices. Details of the collected security practices identified based on our SLR and Sommerville are available in Appendix C (Requirement Engineering Security Maturity Model Practices).

Seven security requirements categories identified via SLR were classified into seven categories. The classification of these security practices was based on the Sommerville. Each category consists of some security requirements practices. Table 5.1 shows the Number of security practices in each category. These practices can be classified as follows: security requirements documentation, security requirements elicitation, security requirements analysis and negotiation, describing security requirements, security system modeling, security requirements validation, security requirements management, and security requirements engineering for critical systems.

Table 5.1 Number of security practices in each category

No.	Category	No. of Security Practices
1	Security Requirements Document	12
2	Security Requirements Elicitation	20
3	Security Requirements Analysis and Negotiation	10
4	Describing Security Requirements	8
5	Security System Modelling	10
6	Security Requirements Validation	9
7	Security Requirements Management	10
	The Total number of security practices	79

The RESMM is a continuous model since we built the RESMM based on a defined set of practices that measure the capability levels within each profile. We employed the concepts of specific goals, specific practices, and measurements of maturity capability. In RESMM, we have specified generic practices. In addition, we have specified the practices in order to achieve the generic goals. RESMM is not a staged model since in a staged model, we need to consider certain security practices for every maturity level over the whole process area.

5.2.2 Measurement component

SCAMPI [20] is used to assess organizational process capability compared to process standards including CMMI and P-CMM. SCAMPI has been used to assign a quality rating of benchmarks, which are relative to CMMI models, containing internal process improvements and external capability determinations. SCAMPI also has different characteristics such as satisfying all appraisal requirements for CMMI and supporting ISO/IEC 15504 assessments. In our case, we have used the structure and concepts of SCAMPI appraisal in order to measure the maturity of each specific goal. The SCAMPI method requires three aspects to be considered: data gathering, analysis, and storage.

5.2.3 Rating of RESMM Process Attributes

To present the levels of achievement of the RESMM process attributes, the ordinal rating scale can be defined as follow: [97]

- ***N Not achieved:*** Here the maturity level can be recognized as having little evidence or no evidence for security considerations through achieving the attribute concerned in the RESMM process. The organization does not seem to care about security in

the system at all and does not afford any preparation for security requirements engineering.

- ***P Partially achieved:*** The defined attribute in the RESMM is somehow partially met. That is, there are signs of partial achievement of the attribute concerned. The organization seems to care just a little about security in the system at different categories of the RESMM process.
- ***L Largely achieved:*** There are signs of a tangible approach to achieve the attribute concerned in the RESMM process. However, some weakness related to this attribute may exist in the RESMM process. The organization seems to care for most security practices but not all the practices at different categories of the RESMM process.
- ***F Fully achieved:*** There are signs of a thorough and well-organized approach to fully achieve the attribute concerned in the RESMM process. No real weaknesses are reported to this attribute in the RESMM process. The organization seems to care greatly for security practices at different categories of the RESMM process.

The corresponding quantitative values shall be:

- N Not achieved: this means the organization has achieved 0 to 15% of the RESMM.
- P Partially achieved: this means the organization has achieved > 15% to 50% of the RESMM.
- L Largely achieved: this means the organization has achieved > 50% to 85% of the RESMM.
- F Fully achieved: this means the organization has achieved > 85% to 100% of the RESMM.

The above range of average values has been identified from a study done by IBM [97]. They built a process area called IBM Rational Unified Process (A CMMI Maturity Level 2 assessment of RUP) [97]. This process was appraised using the measurement described above. The same measurement is adopted in this research to measure each specific goal of RESMM in order to quantify the outcomes as shown in Table 5.2. The structure of the RESMM process area is the same as the CMMI process areas, which contain goals and practices to achieve these goals. In our case, the goals are the categories of Sommerville which we used in our process area. And there are certain practices for achieving these goals.

Table 5.2 Structure of SCAMPI Appraisal for RESMM

ID	Practices	Implementation Level			
		Not	Partially	Largely	Fully
		0	1	2	3
SG1	**Security Requirements documents**				
SP1.1	Define a standard security document structure				
SP1.2	Explain how to use the security document				
SP1.3	Define Security Definitions, Quality Gates and Security Policy of the Organization.				
SP1.4	Make a separate Information security policy such as: (Access Control Policy, Classification Policy, Backup Policy, ... etc.)				
SP1.5	Define and document a project's security bug bar. (establish a minimum level of quality)				
SP1.6	Include a summary of the security requirements				
SP1.7	Make a business case for the system with respect to security				
SP1.8	Define specialized security terms				

SP1.8	Make document layout readable				
SP1.10	Develop a security requirements category				
SP1.11	Help readers find information				
SP1.12	Make the document easy to change				
The Summation for each Column					
The total					
Avg.					

As we mentioned above, we follow the range values that have been used by IBM in their RUP process area. Accordingly, we have four range values in our process area (0, 1, 2, and 3). Thus, in order to convert the percentage values of IBM from to our case values (0, 1, 2, or 3), we multiply 0.15*3 to know the corresponding value for 15%, which will be equal to 0.45. Also, we multiply 0.50*3 to know the corresponding value for 50% which will be equal to 1.5. Moreover, we multiply 0.85*3 to know the corresponding value for 85%, which will be equal to 2.55. Thus, if the average value is between 0 and 0.45, this means the organization does not have an applicable implementation level [97]. If the average value is greater than 0.45 and less than 1.5, this means the organization has a partially applicable implementation level. If the average value is greater than 1.5 and less than 2.55, this means the organization has a largely applicable implementation level. Finally, if the average value is greater than 2.55, this means the organization has a fully applicable implementation level. This has been summarized in Table 5.3 below.

Table 5.3 Appraisal range of value used by IBM Rational Unified Process

No.	Range value in % by IBM	Range of Average Value for RESMM	Maturity Level
1	0 - 15%	If 0 <Avg. <= 0.45	Not applicable
2	15% - 50%	If 0.45 <Avg. <= 1.5	Partially applicable
3	50% - 85%	If 1.5 <Avg. <= 2.55	Largely applicable
4	85% - 100%	If 2.55 <Avg. <= 3	Fully applicable

We have adopted the previous implementation level described in Table 5.3 with the remaining categories of security practices (elicitation security practices, analysis and negotiation of security practices, etc.)

CHAPTER 6

CASE STUDY

6.1 Introduction

A case study is used to obtain more information about practical examples from the real world. In this way, it is possible to clarify information or measure the impact of a certain phenomenon. In this research, the case study helps evaluate the RESMM by different practitioners in the software industry. It also helps to show the effect of using RESMM in real-world software development, identify areas of weakness in the system that need to be improved, and show the results of using RESMM in different organizations.

We communicated with various software development organizations about participating in our questionnaire and provided them with information about certain security practices which are often ambiguous. In addition, we provided them with a reference email so they could contact us regarding any security practice which needed to be explained in more detail. After collecting the organizations' feedback on the questionnaire, the RESMM was developed. Then, the RESMM was conducted on two software development organizations as a case study, using a SCAMPI appraisal to identify the maturity of practices in those organizations. Furthermore, two post-case studies have been done with two software development organizations to evaluate the applicability of RESMM.

6.2 Result

Results of the assessments for surveyed organizations are presented in Table 6.1 through Table 6.7. Each table represents the maturity of the security practices according to their category. Assessment results were also shared with the surveyed organizations to show deficiencies of security practices if there were any.

6.2.1 Organization A

Organization A refers to one of the two organizations in which we conducted our research and is considered as a case study here. It has branches in Egypt and Saudi Arabia. We contacted the Saudi branch, which is located in Riyadh. It has around 80 staff members. The responses to the questions of the case study were made by a technical team leader with ten years of experience in this organization. Recently, he has started developing a system for an international bank in Saudi Arabia. One of his duties is to oversee the collected requirements, analyze these requirements, and try to come up with the desired software ordered by customer.

6.2.1.1 Assessment Outcomes of Organization A

Table 6.1 Security Requirements documents coverage for the RESMM process area

ID	Practices	Implementation Level			
		Not	Partially	Largely	Fully
		0	1	2	3
SG1	**Document Security Requirements**				
SP1.1	Define a standard security document structure		1		
SP1.2	Explain how to use the security document			2	

SP1.3	Define Security Definitions, Quality Gates and security policy of the organization.			2	
SP1.4	Make a separate Information security policy Such as: (Access Control Policy, Classification Policy, Backup Policy, ... etc.)				3
SP1.5	Define and document a project's security bug bar. (establish a minimum level of quality)			2	
SP1.6	Include a summary of the security requirements		1		
SP1.7	Make a business case for the system with respect to security			2	
SP1.8	Define specialized security terms			2	
SP1.9	Make document layout readable			2	
SP1.10	Develop a security requirements category				3
SP1.11	Help readers find information			2	
SP1.12	Make the document easy to change				3
The Summation for each Column		0	2	14	9
The total		25			
Avg.		**Avg.= The total/ # of practices** **Avg.= 25/12 = 2.083**			

Based on Table 6.1, the average of security requirements documents is equal to 2.083, this organization has a largely applicable implementation level of security requirements documents, but it has not reached the highest maturity level. If the organization wants to enhance their security requirements documents practices, they need to implement more of these practices. In that case, the average assessment of the RESMM based on security requirement documents practices will change. If the average reaches to greater than 2.55, this will mean the organization has reached the highest maturity level.

Table 6.2 Security Requirements Elicitation coverage for the RESMM process area

ID	Practices	Implementation Level			
		Not	Partially	Largely	Fully
		0	1	2	3
SG1	**Security Requirements Elicitation**				
SP1.1	Assess System Feasibility with respect to security.		1		
SP1.2	Demonstrate of exploitability.		1		
SP1.3	Be sensitive to organizational and political policy consideration			2	
SP1.4	Identify and consult system stakeholders (to agree upon a common set of security definitions, definition of the organizational security policies and the security vision of the IS.)		1		
SP1.5	Identify vulnerable and/or critical assets.			2	
SP1.6	Identify security objectives and dependencies.			2	
SP1.7	Identify threats and develop artifacts. (such as misuse cases or attack trees diagrams or UMLSec use cases, …)			2	
SP1.8	Record security requirements sources (Identify Resources and Trust Boundaries)		1		
SP1.9	Define the system's operating environment (Specify Operational Environment)			2	
SP1.10	Identifying User Roles and Resource Capabilities			2	
SP1.11	Use business concerns to drive security requirements elicitation		1		
SP1.12	Identify and consult security experts	0			
SP1.13	Select an elicitation method using a systematic tradeoff analysis approach to elicit SR.		1		

SP1.14	Look for domain constraints		1		
SP1.15	Record security requirements rationale			2	
SP1.16	Collect security requirements from multiple viewpoints		1		
SP1.17	Prototype poorly understood security requirements		1		
SP1.18	Use scenarios to elicit security requirements			2	
SP1.19	Define operational processes		1		
SP1.20	Reuse security requirements	0			
The Summation for each Column		0	10	16	0
The total		26			
Avg.		**Avg. = The total/ # of practices Avg. = 26/20 = 1.3**			

Based on Table 6.2, the average of security requirements documents is equal to 1.3, this organization has a partially applicable implementation level of security requirements elicitation practices. The organization needs to enhance its elicitation practices of security requirements to reach to the highest maturity level.

Table 6.3 Security Requirements analysis and negotiation coverage for RESMM process area

ID	Practices	Implementation Level			
		Not	Partially	Largely	Fully
		0	1	2	3
SG1	**Security Requirements analysis and negotiation**				
SP1.1	Define security of system boundaries		1		
SP1.2	Use checklists for security requirements		1		
SP1.3	Perform Security & Privacy Risk Assessment			2	
SP1.4	Negotiate quality gates with different stakeholders		1		
SP1.5	Provide software to support negotiations		1		
SP1.6	Ensures that the access requirements are consistent, complete and conflict-free.		1		
SP1.7	Prioritize security requirements			2	
SP1.8	Classify security requirements using a multi-dimensional approach		1		
SP1.9	Use interaction matrices to find conflicts and overlaps		1		
SP1.10	Review Security Requirements		1		
The Summation for each Column		0	8	4	0
The total		12			
Avg.		**Avg. = The total/ # of practices Avg. = 12/10 = 1.2**			

Based on Table 6.3 the average of security requirements documents is equal to 1.2, this organization has a partially applicable implementation level of security requirements analysis and negotiation practices.

Table 6.4 Describing Security Requirements coverage for the RESMM process area

ID	Practices	Implementation Level			
		Not	Partially	Largely	Fully
		0	1	2	3
SG1	**Describing Security Requirements**				
SP1.1	Define standard templates for describing security requirements			2	
SP1.2	Use languages simply and concisely				3
SP1.3	Be adequate as possible (explicit, precise, complete and non-conflicting)			2	
SP1.4	Use diagrams appropriately			2	
SP1.5	Describe a prototype model for data security based on metadata	0			
SP1.6	Describe abuse cases by examples	0			
SP1.7	Supplement natural language with other description of security requirement			2	
SP1.8	Specify security requirements quantitatively			2	
The Summation for each Column		0	0	10	3
The total		13			
Avg.		**Avg. = The total/ # of practices Avg. = 13/8 = 1.625**			

Based on Table 6.4, the average of security requirements documents is equal to 1.625, this organization has a largely applicable implementation level of describing security requirements practices.

Table 6.5 Security System Modeling coverage for the RESMM process area

ID	Practices	Implementation Level			
		Not	Partially	Largely	Fully
		0	1	2	3
SG1	**Security System Modelling**				
SP1.1	Develop complementary system models with respect to security		1		
SP1.2	Model the system's security environment		1		
SP1.3	Model the system security architecture			2	
SP1.4	Model the Threats of System		1		
SP1.5	Use security pattern template to model SR		1		
SP1.6	Use problem frames to model security	0			
SP1.7	Use structured methods for system security modelling			2	
SP1.8	Use a data dictionary		1		
SP1.9	Document the links between stakeholder requirements and system models		1		
SP1.10	Clearly define the properties that we hope to prevent attackers from violating.			2	
The Summation for each Column		0	6	6	
The total		12			
Avg.		**Avg. = The total/ # of practices Avg. = 12/10 = 1.2**			

Based on Table 6.5, the average of security requirements documents is equal to 1.2, this organization has a partially applicable implementation level of modeling security requirements practices.

Table 6.6 Security Requirements Validation coverage for the RESMM process area

ID	Practices	Implementation Level			
		Not	Partially	Largely	Fully
		0	1	2	3
SG1	**Security Requirements Validation**				
SP1.1	Check that the security requirements document meets your standards		1		
SP1.2	Organize security requirements inspections		1		
SP1.3	Use multi-disciplinary teams to review security requirements	0			
SP1.4	Define validation checklists		1		
SP1.5	Use prototyping to animate security			2	
SP1.6	Perform periodic security assessments and review the quality of security activity		1		
SP1.7	Write a draft user manual	0			
SP1.8	Propose security requirements test cases		1		
SP1.9	Paraphrase system security models		1		
The Summation for each Column		0	6	2	0
The total		8			
Avg.		**Avg. = The total/ # of practices Avg. = 8/9 = 0.89**			

Based on Table 6.6, the average of security requirements documents is equal to 0.89, this organization does not have an applicable implementation level of validating security requirements practices.

Table 6.7 Security Requirements Management coverage for the RESMM process area

ID	Practices	Implementation Level			
		Not	Partially	Largely	Fully
		0	1	2	3
SG1	**Security Requirements Management**				
SP1.1	Uniquely identify each security requirement		1		
SP1.2	Define policies for security requirements management				3
SP1.3	Define traceability policies		1		
SP1.4	Maintain a traceability manual			2	
SP1.5	Use a database to manage security requirements		1		
SP1.6	Define change management policies			2	
SP1.7	Identify global system security requirements			2	
SP1.8	Identify volatile security requirements		1		
SP1.9	Record rejected security requirements			2	
SP1.10	Manage risks of requirements from laws and regulations	0			
The Summation for each Column		0	4	8	3
The total		15			
Avg.		**Avg. = The total/ # of practices Avg. = 15/9 = 1.66**			

Based on Table 6.7 the average of security requirements documents is equal to 1.66, this organization has a largely applicable implementation level of management of security requirements practices.

6.2.1.2 Assessment of Organization A Results

Organization A has maturity limitation in some RESMM areas. It has obtained a score of 1.3 for SR elicitation, 1.2 for SR analysis and negotiation, 1.33 for SR modeling, and 0.89 for SR validation. By contrast, it has secured an acceptable level of maturity in other areas. It has obtained 2.083 for SR documentation, 1.625 for describing of SR, and 1.66 for SR management. This indicates that some areas such as elicitation, analysis and negotiation, modeling, and validation need to be improved in order to reach a higher maturity level. However, their documentation of SR is good since the score is very high (almost mature). This information was helpful for understanding which areas need to be improved.

After we measured the maturity level of security practices in Organization A, we sent a report to the organization to show them their areas of weakness and their areas of strength according to the implemented practices that the organization performed. Table 6.8 summarizes the results of the assessment for Organization A.

Table 6.8 Summary table of maturity security practices of organization A

No.	Security practice category	Avg.	Appraisal of Organization A Using SCAMPI
1	Security requirements documentation	2.083	Largely applicable
2	Security requirements elicitation	1.3	Partially applicable
3	Security requirements analysis and negotiation	1.2	Partially applicable
4	Describing Security requirements	1.625	Largely applicable
5	Security System Modelling	1.2	Partially applicable
6	Security requirements validation	0.89	Partially applicable
7	Security requirements Management	1.66	Largely applicable

6.2.2 Organization B

Another case study was conducted at a different software development organization, which we call “Organization B” here. It is located in Riyadh, Saudi Arabia. This organization is engaged in application development. They advocate for various companies by developing services in different areas, such as application performance management (APM), application/software development, contact centre business units, decision support system (DSS), market development, product development, security application solutions & services, and cloud IP telephony service. They help customers to promote their businesses more effectively through the intelligent and creative use of latest technologies, techniques, and practices. Organization B has more than 650 employees. They have a long experience in Saudi Arabia and other countries. They provide full IT projects by designing, executing, and managing projects with an aim to provide effective and valuable solutions to customers. They have been awarded a certificate in CMMI Maturity Level 3. The responder of our case study was an application development manager with twelve years of experience in software analysis. One of his duties is to furnish end-user requirements and make sure that the development projects match business goals and requirements.

6.2.2.1 Assessment Outcomes of Organization B

Table 6.9 Security Requirements documents coverage for the RESMM process area

ID	Practices	Implementation Level			
		Not	Partially	Largely	Fully
		0	1	2	3
SG1	**Security Requirements documents**				
SP1.1	Define a standard security document structure				3
SP1.2	Explain how to use the security document				3
SP1.3	Define Security Definitions, Quality Gates and security policy of the organization.			2	
SP1.4	Make a separate Information security policy Such as: (Access Control Policy, Classification Policy, Backup Policy, ... etc.)				3
SP1.5	Define and document a project's security bug bar. (establish a minimum level of quality)			2	
SP1.6	Include a summary of the security requirements				3
SP1.7	Make a business case for the system with respect to security			2	
SP1.8	Define specialized security terms				3
SP1.9	Make document layout readable			2	
SP1.10	Develop a security requirements category			2	3
SP1.11	Help readers find information			2	
SP1.12	Make the document easy to change				3
The Summation for each Column		0	0	12	21
The total		33			
Avg.		**Avg.= The total/ # of practices Avg.= 33/12 = 2.75**			

Based on Table 6.9, the average of security requirements documents is equal to 2.75, this organization has a fully applicable implementation level of security requirements documents. The organization B has reached to greater than 2.55 of implementing security requirements documents, this mean that the organization B has reached the highest maturity level.

Table 6.10 Security Requirements Elicitation coverage for the RESMM process area

ID	Practices	Implementation Level			
		Not	Partially	Largely	Fully
		0	1	2	3
SG1	**Security Requirements Elicitation**				
SP1.1	Assess System Feasibility with respect to		1		
SP1.2	Demonstrate of exploitability.			2	
SP1.3	Be sensitive to organizational and political policy consideration			2	
SP1.4	Identify and consult system stakeholders (to agree upon a common set of security definitions, definition of the organizational security policies and the security vision of the IS.)				3
SP1.5	Identify vulnerable and/or critical assets.			2	
SP1.6	Identify security objectives and dependencies.			2	
SP1.7	Identify threats and develop artifacts. (such as misuse cases or attack trees diagrams or UMLSec use cases ...)			2	
SP1.8	Record security requirements sources (Identify Resources and Trust Boundaries)		1		
SP1.9	Define the system's operating environment (Specify Operational Environment)			2	
SP1.10	Identifying User Roles and Resource Capabilities			2	

SP1.11	Use business concerns to drive security requirements elicitation		1		
SP1.12	Identify and consult security experts		1		
SP1.13	Select an elicitation method using a systematic tradeoff analysis approach to elicit SR.			2	
SP1.14	Look for domain constraints			2	
SP1.15	Record security requirements rationale			2	
SP1.16	Collect security requirements from multiple viewpoints		1		
SP1.17	Prototype poorly understood security		1		
SP1.18	Use scenarios to elicit security requirements			2	
SP1.19	Define operational processes			2	
SP1.20	Reuse security requirements		1		
The Summation for each Column		0	7	24	3
The total		34			
Avg.		**Avg. = The total/ # of practices Avg. = 34/20 = 1.7**			

Based on Table 6.10, the average of security requirements documents is equal to 1.7, this organization has a largely applicable implementation level of security requirements elicitation practices. The organization B needs to enhance its elicitation practices of security requirements to reach to the highest maturity level.

Table 6.11 Security Requirements analysis and negotiation coverage for RESMM process area

ID	Practices	Implementation Level			
		Not	Partially	Largely	Fully
		0	1	2	3
SG1	**Security Requirements analysis and negotiation**				
SP1.1	Define security of system boundaries			2	
SP1.2	Use checklists for security requirements analysis		1		
SP1.3	Perform Security & Privacy Risk Assessment			2	
SP1.4	Negotiate quality gates with different stakeholders			2	
SP1.5	Provide software to support negotiations		1		
SP1.6	Ensures that the access requirements are consistent, complete and conflict-free.			2	
SP1.7	Prioritize security requirements				3
SP1.8	Classify security requirements using a multi-dimensional approach		1		
SP1.9	Use interaction matrices to find conflicts and		1		
SP1.10	Review Security Requirements (review security requirements is the confrontation of analysis between Analyst and the Security Team analysis)				3
The Summation for each Column		0	4	8	6
The total		18			
Avg.		**Avg. = The total/ # of practices Avg. = 18/10 = 1.8**			

Based on Table 6.11 the average of security requirements documents is equal to 1.8, this organization has a largely applicable implementation level of SR analysis and negotiation practices.

Table 6.12 Describing Security Requirements coverage for the RESMM process area

ID	Practices	Implementation Level			
		Not	Partially	Largely	Fully
		0	1	2	3
SG1	**Describing Security Requirements**				
SP1.1	Define standard templates for describing security requirements				3
SP1.2	Use languages simply and concisely			2	
SP1.3	Be adequate as possible (explicit, precise, complete and non-conflicting)				3
SP1.4	Use diagrams appropriately			2	
SP1.5	Describe a prototype model for data security based on metadata		1		
SP1.6	Describe abuse cases by examples	0			
SP1.7	Supplement natural language with other description of security requirement			2	
SP1.8	Specify security requirements quantitatively		1		
The Summation for each Column		0	2	6	6
The total		14			
Avg.		**Avg. = The total/ # of practices Avg. = 14/8 = 1.75**			

Based on Table 6.12, the average of security requirements documents is equal to 1.75, this organization has a largely applicable implementation level of describing security requirements practices.

Table 6.13 Security System Modeling coverage for the RESMM process area

ID	Practices	Implementation Level			
		Not	Partially	Largely	Fully
		0	1	2	3
SG1	**Security System Modelling**				
SP1.1	Develop complementary system models with respect to security			2	
SP1.2	Model the system's security environment				3
SP1.3	Model the system security architecture		1		
SP1.4	Model the Threats of System				3
SP1.5	Use security pattern template to model SRs.	0			
SP1.6	Use problem frames to model security requirements	0			
SP1.7	Use structured methods for system security				3
SP1.8	Use a data dictionary	0			
SP1.9	Document the links between stakeholder requirements and system models			2	
SP1.10	Clearly define the properties that we hope to prevent attackers from violating.		1		
The Summation for each Column		0	2	4	9
The total		15			
Avg.		**Avg. = The total/ # of practices Avg. = 15/10 = 1.5**			

Based on Table 6.13 the average of security requirements documents is equal to 1.5, this organization has a partially applicable implementation level of modeling security requirements practices.

Table 6.14 Security Requirements Validation coverage for the RESMM process area

ID	Practices	Implementation Level			
		Not	Partially	Largely	Fully
		0	1	2	3
SG1	**Security Requirements Validation**				
SP1.1	Check that the security requirements document meets your standards			2	
SP1.2	Organize security requirements inspections				3
SP1.3	Use multi-disciplinary teams to review security requirements	0			
SP1.4	Define validation checklists	0			
SP1.5	Use prototyping to animate security requirements	1			
SP1.6	Perform periodic security assessments and review the quality of security activity				3
SP1.7	Write a draft user manual	0			
SP1.8	Propose security requirements test cases			2	
SP1.9	Paraphrase system security models	0			
The Summation for each Column		0	1	4	6
The total		11			
Avg.		**Avg. = The total/ # of practices Avg. = 11/9 = 1.222**			

Based on Table 6.14, the average of security requirements documents is equal to 1.222, this organization have an applicable implementation level of validating security requirements practices.

Table 6.15 Security Requirements Management coverage for the RESMM process area

ID	Practices	Implementation Level			
		Not	Partially	Largely	Fully
		0	1	2	3
SG1	**Security Requirements Management**				
SP1.1	Uniquely identify each security requirement				3
SP1.2	Define policies for security requirements				3
SP1.3	Define traceability policies				3
SP1.4	Maintain a traceability manual			2	
SP1.5	Use a database to manage security requirements			2	
SP1.6	Define change management policies				3
SP1.7	Identify global system security requirements			2	
SP1.8	Identify volatile security requirements				3
SP1.9	Record rejected security requirements				3
SP1.10	Manage risks of requirements from laws and				3
The Summation for each Column		0	0	6	21
The total		27			
Avg.		**Avg. = The total/ # of practices** **Avg. = 27/10 = 2.7**			

Based on Table 6.15 the average of security requirements documents is equal to 2.7, this organization has a fully applicable implementation level of management of security requirements practices.

6.2.2.2 Assessment of Organization B Results

Organization B has good maturity in most areas of RESMM. It has obtained a score of 2.75 for SR documentation, 1.7 for SR elicitation, 1.8 for SR analysis and negotiation, 1.75 for describing of SR, and 1.9 for SR management. By contrast, the organization has maturity limitation in some other areas, such as SR modeling and SR validation. It has obtained 1.2 for SR modeling, and 1.222 for SR validation. This indicates that some areas, such as SR modeling and SR validation, need to be improved in order to reach a higher maturity level. However, their documentation of SR is good since the score is very high (almost mature). This information was helpful for understanding which areas needed to be improved.

After we measured the maturity level of security practices with Organization B, we sent them a report to show them their areas of weakness and strength according to the implemented practices that the organization performed. Table 6.16 summarizes the results of the assessment for Organization A.

Table 6.16 Summary table of maturity security practices of organization B

No.	Security practice category	Avg.	Appraisal of Organization A Using SCAMPI
1	Security requirements documentation	2.75	Fully applicable
2	Security requirements elicitation	1.7	Largely applicable
3	Security requirements analysis and negotiation	1.8	Largely applicable
4	Describing Security requirements	1.75	Largely applicable
5	Security System Modelling	1.5	Partially applicable
6	Security requirements validation	1.222	Partially applicable
7	Security requirements documentation	2.7	Fully applicable

6.3 Evaluation of RESMM

After we built RESMM, we evaluated this model in the real-world environment. Two case studies were conducted with two software organizations. The SPI managers from these organizations were invited to participate in this study. The SPI managers agreed to participate as they were interested in evaluating their RE processes with respect to maturity of RE security practices. These managers were provided with full documentation of the RESMM with complete notes about how to use RESMM.

After completing the case studies, the respondents were asked to fulfill the post-case study questionnaire to evaluate RESMM in the real-world environment. The feedback of the RESMM evaluation was considered to find which part of RESMM needed to be improved.

6.4 Evaluation Criteria

Based on [98], there are two success criteria that need to be achieved to assess the strengths and weaknesses of the RE practices within the model. These success criteria are user satisfaction and ease-of-use. Beside to these two success criteria, we are also concerned with the structure of the created model. We need to see if there are any comments about the structure of our model. Thus, we have used the following criteria to evaluate RESMM:

- Structure of the RESMM: This criterion identifies any flaws on RESMM structure and ways to enhance RESMM structure.

- Usability: This criterion assesses how easy it is to use RESMM. It also evaluates RESMM structure and improves the ease-of-use of RESMM. In order to avoid

building a complex model, we need RESMM to be unambiguous and more flexible to users, because complex models require higher effort and training.

- User satisfaction: This criterion assesses the achievement of specified users' goals according to user needs and expectations of RESMM without confusion or ambiguity.

6.5 Feedback Summary

As has been thoroughly explained in the previous section and in Table 6.1 through Table 6.7. Organization A has some limitation of maturity in some areas, such as security requirements elicitation practices, security requirements analysis and negotiation practices, modeling security requirements practices, and validating security requirements practices. The organization has some acceptable level of maturity in some areas such as security requirements documents practices, describing security requirements practices, and management of security requirements practices. After we measured the maturity level of security practices with Organization A, we sent a report to the organization to show them their areas of weakness and their areas of strength according to the implemented practices that the organization performed. The feedback submitted by respondents of Organization A has been used to evaluate various aspects of the RESMM. As mentioned in the RESMM development section, there are three success criteria (RESMM structure, usability, and user satisfaction). We have used a quantitative measurement to evaluate these success criteria. Furthermore, we have also provided some questions to gather the participants' reviews, any modifications of RESMM, or any suggestions for enhancing RESMM.

We have adopted the tables for each success criteria from a study by Yusuf et al. [99]. First, we adopted a table for the ease of learning of RESMM and asked Organization A to

evaluate the ease of learning of RESMM. Based on Table 6.17, Organization A positively agreed that RESMM is easy to comprehend and learn. However, there is still a need for some training to grasp how to of utilize RESMM accurately. In spite of that, the practitioners involved in the questionnaire were familiar with the process of requirements engineering and security techniques, as they had taken some courses in security requirements engineering.

Table 6.17 Ease of Learning Evaluation of Organization A & B

No.	Ease of Learning	Organizations' viewpoint (n=2)							
		+			-			Neutral	
		Strongly Agree	Agree	%	Strongly Disagree	Disagree	%	Neutral	%
1)	RESMM representation is easy to learn.	0	2	100 %	0	0	0	0	0
2)	Having basic knowledge at least about security requirements engineering is necessary to be able to use RESMM.	0	2	100 %	0	0	0	0	0
3)	It is necessary to learn the practices arranged for each security requirement category.	0	2	100 %	0	0	0	0	0
4)	The assessment method SCAMPI needs to be understood.	0	2	100 %	0	0	0	0	0
5)	It is substantial to use RESMM to measure organization's maturity for security requirement engineering practices.	0	2	100 %	0	0	0	0	0
6)	It is necessary to classify security requirement practices into different categories, e.g. SR documentation practices, SR elicitation practices, etc.	0	2	100 %	0	0	0	0	0
7)	Some kind of training is necessary to facilitate the utilization of RESMM.	0	2	100 %	0	0	0	0	0

The second success criterion is user satisfaction. It assesses users' satisfaction based on the outcome of RESMM. As shown in Table 6.18, Organization A agrees on the

usefulness of RESMM and recommends that other organizations apply RESMM. It has reflected interest in applying RESMM in its own work. Organization A's staff are convinced about RESMM's capability for the discovery of weakness areas that need to be improved.

Table 6.18 User Satisfaction Evaluation of Organizations A & B

No.	Ease of Learning	Organizations' perception (n=2)							
		+			-			Neutral	
		Strongly Agree	Agree	%	Strongly Disagree	Disagree	%	Neutral	%
1)	RESMM can be carried out in most organizations.	0	2	100%	0	0	0	0	0
2)	Each practice is clear and easy to learn.	0	2	100%	0	0	0	0	0
3)	RESMM can identify the areas of weakness and the areas of strength in organizations with respect to security requirements engineering practices which they cover.	0	2	100%	0	0	0	0	0
4)	Using RESMM would enhance the security requirements engineering.	0	2	100%	0	0	0	0	0
5)	If RESMM were accessible in my occupation, I expect to utilize it.	0	2	100%	0	0	0	0	0
6)	I agree with the maturity issues identified by RESMM.	0	2	100%	0	0	0	0	0
7)	Using RESMM as an automated software tool is critical to persuade security requirements engineering in measuring an organization's maturity.	0	2	100%	0	0	0	0	0

The third success criterion is the structural aspect of the RESMM. Table 6.19 shows the evaluation of RESMM structure by Organization A and Organization B. Their positive

responses indicate that that the RESMM structure was very clear, considering that we followed CMMI structure. They mentioned that the classification of security practices into different categories was very helpful and that each practice was put under a suitable category to avoid confusion. Based on their feedback, RESMM can be used effectively to measure the security maturity of software development organizations.

Table 6.19 RESMM Structure Evaluation of Organization A & B

No.	Ease of Learning	Organizations' perception (n=2)							
		+			-			Neutral	
		Strongly Agree	Agree	%	Strongly Disagree	Disagree	%	Neutral	%
1)	Every RESMM category is self-explanatory and requires no further clarification for adequate utilization.	2	0	100%	0	0	0	0	0
2)	Every RESMM category is feasible and suited to the security requirements engineering process.	2	0	100%	0	0	0	0	0
3)	RESMM can be used effectively to identify security requirements engineering weakness areas with an aim to increase organization's maturity for security requirements engineering.	2	0	100%	0	0	0	0	0
4)	The distribution of security practices among various categories (e.g. Documentation, Elicitation, Analysis, etc.) is valuable.	2	0	100%	0	0	0	0	0
5)	The seven categories of RESMM are valuable.	2	0	100%	0	0	0	0	0

As for the suggestions offered in the feedback from the organizations, we received a few from Organization B only. Table 6.20 shows the feedback results of Organizations A and B. One of their suggestions was to provide more clarification for a few practices. Another

suggestion was to enhance the questionnaire of the case study so that the maturity level of the organization could be calculated in the same form that the respondents fill out for the questionnaire. This was resolved by using an Excel sheet to meet this purpose. In fact, there has not been much modification on the developed model since the model was built based on the feedback obtained from the respondents of the first questionnaire (10 organizations)

Table 6.20 Feedback Results (Essay Answer) of Organizations A & B

Question	Response		
	Organization	Organization B	
Do you think there is a missing category that need to be added to RESMM? Please provide the reason for your answer.	No	Since RESMM has followed CMMI categories.	Very Positive
Do you suggest any improvement for RESMM?	No	Need to make a clarification for some practices. For clarity, it would be better to show a note for certain practices to avoid any confusion, especially for those who have no background on security requirement engineering.	Positive
Are there any comments about the assessment method?	No	Made an excel sheet to calculate the average values of each category immediately instead of sending the report after the organization sends their feedback by email.	Positive
Is there any wrong classification of security requirement practices among the various categories?	No	No	Very Positive

Table 6.21 shows the practices that need to be explained more:

Table 6.21 More explanation for some of the Practices

No.	Before	After Adding Explanation for that practice
1	Make a separate Information security policy	Make a separate information security policy, such as (Access Control Policy, Classification Policy, Backup Policy, etc.)
2	Identify threats and develop artifacts.	Identify threats and develop artifacts, such as (misuse cases or attack tree diagrams or UMLSec use cases and classes or sequence/state diagrams)
3	Review Security Requirements	Review Security Requirements (review security requirements is the confrontation of analysis between Analyst and the Security Team analysis)
4	Define Security Definitions, Quality Gates.	Quality gates are basically acceptance criteria reviews that can be used throughout any project.

6.6 Case Study Lessons Learned

There are several lessons learned through the cases study on RESMM. First of all, we have learned how to develop a well-structured and organized questionnaire that could contribute to positive feedback from respondents.

Second, the results obtained from this research might guide researchers to have prior knowledge about the different viewpoints toward various security practices from both researchers' and practitioners' perspectives. Doing so is important to ensure that we collect accurate security practices which will be adopted in the development of RESMM. To gain practitioners' opinions, a well-structured questionnaire could be an excellent medium to obtain the information required.

Third, we have found that there is a need to promote basic knowledge of security practices among software engineers so they can identify and assess areas of weaknesses in organizations for improvement purposes. Organizations will be better informed about the maturity of security in their software if more focus and attention is given to dissemination of security practices information among their staff.

Fourth, from the development of the case study, we have learned that it is important to take into consideration the lack of knowledge about security practices and that the case study should be as simple as possible. Some security practices could be explained in more detail rather than putting them in tables without further explanation. This will ensure positive interaction with potential software organizations' staff. This will also ensure obtaining accurate answers for the questions asked.

Lastly, the feedback we obtained from the respondents in the case study has been important to improve the RESMM which will ensure its applicability in software organizations.

6.7 Threats to validity

This research has some limitation due to conducting a SLR in just three database sources (ScienceDirect, IEEE, ACM). Our reasoning was that if there were different security practices existing in different databases, we might not consider them while conducting SLR. Thus, some studies could have been missed. Nevertheless, we believe our outcomes cover the most relevant published literature, and moreover, doing the questionnaire with ten organizations also has helped in avoiding this problem. The questionnaire has provided us with the most widely used security practices from different organizations. Thus, if there are any practices that may be missed in doing SLR, the

questionnaire will fill that gap. And it will give us confidence in the reliability of the collected data.

Another limitation is that some software organizations were either not willing to provide feedback on the questionnaire or they were not aware of security practices and how to answer the questionnaire. To overcome this limitation, we provided more information about security practices and the objective of our questionnaire and research in an email to those software organizations. To our surprise, some software organizations still did not provide their feedback on the questionnaire. However, there might be a need to conduct more questionnaires with more organizations around the world in order to generalize the outcome of security practices that will be used in the construction of RESMM. That is so because the sizes of organizations are important for collecting more information about the security practices used in those organizations.

Another limitation might be that the case study using SCAMPI appraisals only involves two organizations. This is attributed to the restricted cooperation of organizations as explained above. However, there might be a need to conduct more case studies with different organizations in order to check the applicability of RESMM.

Moreover, since CMMI v1.3 was developed with the waterfall approach, RESMM has worked well with the waterfall approach. In fact, RESMM isn't customized to the Agile approach. The scope and limited space of this study does not allow us to discuss all aspects of applying RESMM with Agile methodology. If we want to check which practices will still be valid with the Agile methodology, we have to make more case studies with different organizations who use this methodology to come up with certain security practices that are consistent with Agile methodology.

CHAPTER 7

CONCLUSION

7.1 Conclusion

This research has aimed to develop the Requirements Engineering Security Maturity Model (RESMM) to assist software development organizations in better specifying requirements for secure software. Software organizations are expected to be able to identify their areas of weaknesses with respect to security practices classification, and this in turn will help these organizations to enhance their software to be secure.

A systematic Literature Review (SLR) was an important part of this research. It aimed to review the most common security practices at the requirement phase. As part of the SLR, 96 primary studies were reviewed in detail to consider existing security practices at the requirements phase. Eventually, the security requirements classifications were utilized in the development of RESMM.

The RESMM has a structure which contains security practices at the requirement phase. This research presented the RESMM with seven categories of security practices; each category contains various security practices which are related to that category in RESMM. Moreover, a questionnaire is administered to different ten organizations to verify the collection practices which we found from the SLR.

The results of the questionnaire from 10 organizations have helped in building RESMM. Some security practices have moved from one category to another. These changes have enhanced the usability and applicability of the RESMM to assess software development organizations in better specifying the requirements for secure software.

To assess the usability of the RESMM, two post-case studies were conducted with two software organizations that specialize in software development and have several branches in Saudi Arabia. The results of each case study were precisely analyzed.

This work will assist software development organizations in better specifying requirements for secure software. In addition, the outcomes of this research will provide software development organizations with the ability to measure their maturity of specifying requirements for secure software. This work will put software development organizations in a better position to deliver software that is more secure.

The contribution of this study is to develop RESMM that assists software development organizations in better specifying requirements for secure software. In addition, we have employed practical and evidence-based approaches to the development of RESMM.

7.2 Recommendations

Taking into consideration the growing need for software development, this research offers some potential suggestions for future research.

- There is a need to consider the interaction between RESMM process areas and CMMI process areas.
- New technologies such as Internet of Things (IoT) and cloud computing have certain attributes of practices that have not been taken into account in this research. There have been very few organizations that were cooperative and agreed to engage in this research.
- There might be a need to conduct SLR with other database sources to ensure full coverage of existing security practices at the requirement phase.
- There is potential in enhancing RESMM by applying it in different organizations and carefully analyzing the feedback to enhance the construction of the Requirement Engineering Security Maturity Model.
- The RESMM has been built and customized with waterfall methodology. There might be a need to consider what are the security practices that can be chosen from RESMM that applicable with Agile methodology.

REFERENCES

[1] A. Patel, “Formal methods, techniques and tools for secure and reliable applications,” *Comput. Stand. Interfaces*, vol. 5, no. 27, pp. 439–443, 2005.

[2] R. L. Jones and A. Rastogi, “Secure coding: Building security into the software development life cycle,” *Inf. Syst. Secur.*, vol. 13, no. 5, pp. 29–39, 2004.

[3] H. El-Hadary and S. El-Kassas, “Capturing security requirements for software systems,” *J. Adv. Res.*, vol. 5, no. 4, pp. 463–472, 2014.

[4] I. Sommerville and P. Sawyer, *Requirements engineering: a good practice guide*. John Wiley & Sons, Inc., 1997.

[5] CMMI Product Team, “CMMI® for Development, Version 1.3,” no. November, 2010.

[6] H. H. AlBreiki and Q. H. Mahmoud, “Evaluation of static analysis tools for software security,” *2014 10th Int. Conf. Innov. Inf. Technol.*, pp. 93–98, 2014.

[7] M. A. Howard, “A process for performing security code reviews,” *IEEE Secur. Priv.*, vol. 4, no. 4, pp. 74–79, 2006.

[8] D. Xu *et al.*, “Security threat modeling and analysis: A goal-oriented approach,” *URL http//dymaxion. org/trike/ …*, vol. 24, no. 12, pp. 94–102, 2006.

[9] G. Goos, J. Hartmanis, and J. van Leeuwen, *Engineering Secure Software and Systems*. 2013.

[10] D. Proença and J. Borbinha, “Maturity Models for Information Systems - A State of the Art,” *Procedia Comput. Sci.*, vol. 100, no. 2, pp. 1042–1049, 2016.

[11] F. Briggs, W. A. Group, and L. Park, “Software Security Challenges in Computing and Communications Environments,” *AIAA InfoTech*, no. March, pp. 1–12, 2011.

[12] S. Ahmed, “Secure software development - Identification of security activities and their integration in software development lifecycle,” no. March, p. 40, 2007.

[13] T. Okubo and H. Tanaka, “Identifying Security Aspects in Early Development Stages,” *2008 Third Int. Conf. Availability, Reliab. Secur.*, pp. 1150–1157, 2008.

[14] C. B. Haley, R. Laney, J. Moffett, and B. Nuseibeh, “Security Requirements Engineering: A Framework for Representation and Analysis,” *IEEE Trans. Softw. Eng.*, vol. 34, no. 1, pp. 133–153, 2008.

[15] S. Islam and P. Falcarin, “Measuring security requirements for software security,” *Cybern. Intell. Syst. (CIS), 2011 IEEE 10th Int. Conf.*, pp. 70–75, 2011.

[16] N. R. Mead, “Measuring The Software Security Requirements Engineering Process,” pp. 583–588, 2012.

[17] K. Balarama, “10 ways to infuse security into your software development life cycle,” 2016. [Online]. Available: https://www.synopsys.com/blogs/software-security/infuse-security-into-your-software-development-life-cycle/. [Accessed: 02-Jan-2018].

[18] G. Mcgraw, “Software security,” *IEEE Secur. Priv. Mag.*, vol. 2, no. 2, pp. 80–83, 2004.

[19] H. Mouratidis, J. Jürjens, and J. Fox, “Towards a Comprehensive Framework for Secure Systems Development,” *Proc. 18th Int. Conf. Adv. Inf. Syst. Eng. (CAiSE 2006)*, pp. 48–62, 2006.

[20] S. U. Team, “Standard CMMI Appraisal Method for Process Improvement (SCAMPI) A, Version 1.3: Method Definition Document,” 2011.

[21] S. Subashini and V. Kavitha, “A survey on security issues in service delivery models of cloud computing,” *J. Netw. Comput. Appl.*, vol. 34, no. 1, pp. 1–11, 2011.

[22] G. McGraw, “Managing software security risks,” *Computer (Long. Beach. Calif).*, vol. 35, no. 4, pp. 99–101, 2002.

[23] N. M. Mohammed, M. Niazi, M. Alshayeb, and S. Mahmood, “Exploring software security approaches in software development lifecycle: A systematic mapping study,” *Comput. Stand. Interfaces*, vol. 50, no. May 2016, pp. 107–115, 2017.

[24] G. McGraw and B. Chess, “Software [In] security: A Software Security Framework: Working Towards a Realistic Maturity Model.” InformIT, 2008.

[25] H. Mohaddes and I. Tabatabaei, “Effects of Software Security on Software Development Life Cycle and Related Security Issues,” vol. 6, no. 8, pp. 4–12, 2015.

[26] E. Nash, “Hackers bigger threat than rogue staff.” VNU Publications, May, 2003.

[27] G. Dhillon, “Principles of information systems security : text and cases,” p. 451, 2007.

[28] “CMMI Development V2.0,” *CMMI Institute*, 2018. [Online]. Available: https://cmmiinstitute.com/cmmi/dev. [Accessed: 05-Oct-2018].

[29] P. Bowen and R. Kissel, “NISTIR 7358: Program Review for Information Security Management Assistance (PRISMA),” pp. 1–60, 2007.

[30] C. Institute, “How Is CMMI V2.0 Different From V1.3,” *2018*. [Online]. Available: https://cmmiinstitute.zendesk.com/hc/en-us/articles/360000175667-How-is-CMMI-V2-0-different-from-V1-3-.

[31] H. Mouratidis and P. Giorgini, “Security Attack Testing (SAT)-testing the security of information systems at design time,” *Inf. Syst.*, vol. 32, no. 8, pp. 1166–1183, 2007.

[32] N. Davis, “Secure software development life cycle processes: A technology scouting report,” *Carnegie M Ielloii Softw. Eng. Inst.*, 2005.

[33] A. G. Siemens, “Security by Design with CMMI for Development, Version 1.3. An Application Guide for Improving Processes for Secure Products,” *C. Inst.*, 2013.

[34] T. Grance, J. Hash, and M. Stevens, *Security considerations in the information system development life cycle*. US Department of Commerce, Technology Administration, National Institute of Standards and Technology, 2004.

[35] C. P. Pfleeger and S. L. Pfleeger, *Security in computing*. Prentice Hall Professional Technical Reference, 2002.

[36] C. Mundie, P. de Vries, P. Haynes, and M. Corwine, “Trustworthy computing,” 2002.

[37] S. Lipner, “The trustworthy computing security development lifecycle,” *20th Annu. Comput. Secur. Appl. Conf.*, pp. 2–13, 2004.

[38] Microsoft, “Security Development Lifecycle for Agile Development,” 2013. [Online]. Available: https://msdn.microsoft.com/en-us/library/windows/desktop/ee790621.aspx.

[39] Owasp, “CLASP Concepts,” 2016. [Online]. Available: https://www.owasp.org/index.php/CLASP_Concepts.

[40] B. De Win, R. Scandariato, K. Buyens, J. Grégoire, and W. Joosen, “On the secure software development process: CLASP, SDL and Touchpoints compared,” *Inf. Softw. Technol.*, vol. 51, no. 7, pp. 1152–1171, 2009.

[41] S. T. Halkidis, A. Chatzigeorgiou, and G. Stephanides, “Brief Review of Software Security History with an Emphasis on Efforts Focused at Early Stages of the Software Lifecycle,” *J. Inf. Priv. Secur.*, vol. 10, no. 1, pp. 3–27, Jan. 2014.

[42] B. Musa Shuaibu, N. Md Norwawi, M. H. Selamat, and A. Al-Alwani, “Systematic review of web application security development model,” *Artif. Intell. Rev.*, vol. 43, no. 2, pp. 259–276, 2013.

[43] R. L. Krutz and A. J. Fry, *The CSSLP Prep Guide: Mastering the Certified Secure Software Lifecycle Professional*. Wiley Publishing, 2009.

[44] G. McGraw, *Software Security Building Security In*. Boston: Pearson Education Ltd., 2006.

[45] M. Howard, "Building more secure software with improved development processes," *IEEE Secur. Priv.*, vol. 2, no. 6, pp. 63–65, 2004.

[46] G. Sindre and A. L. Opdahl, "Eliciting security requirements with misuse cases," *Requir. Eng.*, vol. 10, no. 1, pp. 34–44, 2005.

[47] M. Howard and D. LeBlanc, *Writing secure code*. Pearson Education, 2003.

[48] C. B. Haley, R. C. Laney, and B. Nuseibeh, "Deriving security requirements from crosscutting threat descriptions," *Proc. 3rd Int. Conf. Asp. Softw. Dev. - AOSD '04*, no. March, pp. 112–121, 2004.

[49] M. Islam, C. Sandberg, and T. Olovsson, "A Risk Assessment Framework for Automotive Embedded Systems," pp. 3–14.

[50] B. Chess and G. McGraw, "Static analysis for security," *Secur. Privacy, IEEE*, vol. 2, pp. 76–79, 2004.

[51] B. Potter and G. McGraw, "Software security testing," *Secur. Privacy, IEEE*, vol. 2, no. 5, pp. 81–85, 2004.

[52] B. Arkin, S. Stender, and G. McGraw, "Software penetration testing," *IEEE Secur. Priv.*, vol. 3, no. 1, pp. 84–87, 2005.

[53] P. Black, "Software Assurance Metrics and Tool Evaluation.," *Softw. Eng. Res. Pract.*, no. May, 2005.

[54] A. Souag, R. Mazo, C. Salinesi, and I. Comyn-Wattiau, "Reusable knowledge in security requirements engineering: a systematic mapping study," *Requir. Eng.*, vol. 21, no. 2, pp. 251–283, 2016.

[55] N. F. Khan and N. Ikram, "Security Requirements Engineering: A Systematic Mapping (2010-2015)," *2016 Int. Conf. Softw. Secur. Assur.*, pp. 31–36, 2016.

[56] O. Wh and G. Son, "-Focus* 1," pp. 657–678, 2013.

[57] A. L. Mesquida and A. Mas, "Implementing information security best practices on software lifecycle processes: The ISO/IEC 15504 Security Extension," *Comput. Secur.*, vol. 48, pp. 19–34, 2015.

[58] G. Boström, K. Beznosov, and P. Kruchten, “Extending XP Practices to Support Security Requirements Engineering,” pp. 11–17, 2006.

[59] G. Elahi and E. Yu, “Security Requirements Engineering in the Wild : A Survey of Common Practices,” 2011.

[60] B. Kitchenham, “Procedures for performing systematic reviews,” *Keele, UK, Keele Univ.*, vol. 33, no. TR/SE-0401, p. 28, 2004.

[61] F. C. Seale *et al.*, “Five misunderstandings about case-study research,” *Practice*, pp. 420–434, 2004.

[62] A. V. Uzunov, K. Falkner, and E. B. Fernandez, “A comprehensive pattern-oriented approach to engineering security methodologies,” *Inf. Softw. Technol.*, vol. 57, no. 1, pp. 217–247, 2015.

[63] C. Gutiérrez, D. G. Rosado, and E. Fernández-Medina, “The practical application of a process for eliciting and designing security in web service systems,” *Inf. Softw. Technol.*, vol. 51, no. 12, pp. 1712–1738, 2009.

[64] H. Zafar, “Human resource information systems: Information security concerns for organizations,” *Hum. Resour. Manag. Rev.*, vol. 23, no. 1, pp. 105–113, 2013.

[65] A. M. Hoole, I. Traore, and I. Simplot-Ryl, “Application of contract-based security assertion monitoring framework for telecommunications software engineering,” *Math. Comput. Model.*, vol. 53, no. 3–4, pp. 522–537, 2011.

[66] I. H. Chou and C. F. Fan, “Regulatory-based development processes for software security in nuclear safety systems,” *Prog. Nucl. Energy*, vol. 52, no. 4, pp. 395–402, 2010.

[67] J. Leiwo, L. F. Kwok, D. L. Maskell, and N. Stankovic, “A technique for expressing IT security objectives,” *Inf. Softw. Technol.*, vol. 48, no. 7, pp. 532–539, 2006.

[68] S. W. Lee, R. A. Gandhi, and G. Ahn, “Establishing Trustworthiness in Services of the Critical Infrastructure through Certification and Accreditation,” pp. 1–7, 2005.

[69] K. Beckers, S. Faßbender, D. Hatebur, M. Heisel, and I. Côté, “Common Criteria CompliAnt Software Development (CC-CASD),” pp. 1298–1304, 2013.

[70] J. Romero-mariona, “Secure and Usable Requirements Engineering,” pp. 703–706, 2009.

[71] J. Romero-mariona, I. Donald, I. Donald, D. J. Richardson, I. Donald, and D. Bystritsky, “Towards Usable Cyber Security Requirements.”

[72] J. D. Moffett and B. A. Nuseibeh, “A Framework for Security Requirements Engineering,” *Australas. Inf. Secur. Conf.*, 2006.

[73] S. Lee, R. Gandhi, D. Muthurajan, D. Yavagal, and G. Ahn, “Building Problem Domain Ontology from Security Requirements in Regulatory Documents,” pp. 43–49, 2006.

[74] I. M. Y. Woon and A. Kankanhalli, “Investigation of IS professionals’ intention to practise secure development of applications,” *Int. J. Hum. Comput. Stud.*, vol. 65, no. 1, pp. 29–41, 2007.

[75] J. Park, Y. Suh, and C. Park, “Implementation of cyber security for safety systems of nuclear facilities,” *Prog. Nucl. Energy*, vol. 88, pp. 88–94, 2016.

[76] R. Villarroel, E. Fernández-Medina, and M. Piattini, “Secure information systems development – a survey and comparison,” *Comput. Secur.*, vol. 24, no. 4, pp. 308–321, 2005.

[77] G. Frankova, M. Séguran, F. Gilcher, S. Trabelsi, J. Dörflinger, and M. Aiello, “Deriving business processes with service level agreements from early requirements,” *J. Syst. Softw.*, vol. 84, no. 8, pp. 1351–1363, 2011.

[78] H. Mouratidis, P. Giorgini, and G. Manson, “When security meets software engineering: A case of modelling secure information systems,” *Inf. Syst.*, vol. 30, no. 8, pp. 609–629, 2005.

[79] C. Kalloniatis, H. Mouratidis, M. Vassilis, S. Islam, S. Gritzalis, and E. Kavakli, “Towards the design of secure and privacy-oriented information systems in the cloud: Identifying the major concepts,” *Comput. Stand. Interfaces*, vol. 36, no. 4, pp. 759–775, 2014.

[80] J. Danahy, “The ‘phasing-in’ of security governance in the SDLC,” *Netw. Secur.*, vol. 2008, no. 12, pp. 15–17, 2008.

[81] J. Trujillo, E. Soler, E. Fernández-Medina, and M. Piattini, “An engineering process for developing Secure Data Warehouses,” *Inf. Softw. Technol.*, vol. 51, no. 6, pp. 1033–1051, 2009.

[82] S. Kriaa, L. Pietre-Cambacedes, M. Bouissou, and Y. Halgand, “A survey of approaches combining safety and security for industrial control systems,” *Reliab. Eng. Syst. Saf.*, vol. 139, pp. 156–178, 2015.

[83] C. W. Probst and R. R. Hansen, “An extensible analysable system model,” *Inf. Secur. Tech. Rep.*, vol. 13, no. 4, pp. 235–246, 2008.

[84] H. Belani, Ž. Car, and A. Cari, “RUP-Based Process Model for Security Requirements Engineering in Value-Added Service Development,” pp. 54–60, 2009.

[85] I. Maskani, “Student Research Abstract : A new comprehensive approach to security requirements engineering,” pp. 1136–1137.

[86] R. Laborde, P. Sabatier, A. S. Wazan, P. Sabatier, F. Barrère, and P. Sabatier, “Which Security Requirements Engineering Methodology Should I Choose ? Towards a Requirements Engineering-based Evaluation Approach,” 2017.

[87] M. Riaz, J. Stallings, M. P. Singh, J. Slankas, and L. Williams, “DIGS – A Framework for Discovering Goals for Security Requirements Engineering,” *Emprical Softw. Eng. Meas.*, 2016.

[88] A. V. Uzunov and E. B. Fernandez, “An extensible pattern-based library and taxonomy of security threats for distributed systems,” *Comput. Stand. Interfaces*, vol. 36, no. 4, pp. 734–747, 2014.

[89] K. Alghathbar, “Validating the enforcement of access control policies and separation of duty principle in requirement engineering,” *Inf. Softw. Technol.*, vol. 49, no. 2, pp. 142–157, 2007.

[90] J. Li, M. Li, D. Wu, and H. Song, “An integrated risk measurement and optimization model for trustworthy software process management,” *Inf. Sci. (Ny).*, vol. 191, no. 15, pp. 47–60, 2012.

[91] O. Rebollo, D. Mellado, E. Fernández-Medina, and H. Mouratidis, “Empirical evaluation of a cloud computing information security governance framework,” *Inf. Softw. Technol.*, vol. 58, pp. 44–57, 2015.

[92] G. Beydoun and G. Low, “Generic modelling of security awareness in agent based systems,” *Inf. Sci. (Ny).*, vol. 239, pp. 62–71, 2013.

[93] O. Ox, I. Fléchais, and I. Meta-model, “A Meta-Model for Usable Secure Requirements Engineering,” pp. 29–35, 2010.

[94] M. Saito *et al.*, “A case-based management system for secure software development using software security knowledge,” *Procedia Comput. Sci.*, vol. 60, no. 1, pp. 1092–1100, 2015.

[95] D. Mellado, H. Mouratidis, and E. Fernández-Medina, “Secure Tropos framework for software product lines requirements engineering,” *Comput. Stand. Interfaces*, vol. 36, no. 4, pp. 711–722, 2014.

[96] H. Wang, Z. Jia, and Z. Shen, “Research on Security Requirements Engineering Process,” pp. 1285–1288, 2009.

[97] M. Grundmann, “A CMMI Maturity Level 2 Assessment of RUP,” pp. 1–8, 2005.

[98] M. Niazi, Æ. K. Cox, and Æ. J. Verner, “of requirements engineering process,” pp. 213–235, 2008.

[99] Y. Mufti, M. Niazi, M. Alshayeb, and S. Mahmood, “A Readiness Model for Security Requirements Engineering,” pp. 28611–28631, 2018.

[100] A. Patel, “Formal methods, techniques and tools for secure and reliable applications,” *Comput. Stand. Interfaces*, vol. 27, no. 5, pp. 439–443, 2005.

[101] M. A. Khan, “A survey of security issues for cloud computing,” *J. Netw. Comput. Appl.*, vol. 71, pp. 11–29, 2016.

[102] R. Kazman, H. P. In, and H. M. Chen, “From requirements negotiation to software architecture decisions,” *Inf. Softw. Technol.*, vol. 47, no. 8, pp. 511–520, 2005.

[103] M. Mejri and H. Yahyaoui, “Formal specification and integration of distributed security policies,” *Comput. Lang. Syst. Struct.*, vol. 49, pp. 1–35, 2017.

[104] P. Salini, “A Novel Method : Ontology-based Security Requirements Engineering Framework,” 2016.

[105] P. Salini and S. Kanmani, “A Model Based Security Requirements Engineering Framework Applied for Online Trading System,” in *International Conference on Recent Trends in Information Technology (ICRTIT)*, 2011, pp. 1195–1202.

[106] N. R. Mead, “Measuring the Software Security Requirements Engineering Process,” *2012 IEEE 36th Annu. Comput. Softw. Appl. Conf. Work.*, pp. 583–588, 2012.

[107] A. A. Abdulrazeg, N. Norwawi, and N. Basir, “Extending V-model practices to support SRE to build Secure Web Application,” pp. 213–218, 2014.

[108] D. Mellado, E. Fernández-Medina, and M. Piattini, “Security Requirements Engineering process for Software Product Lines: A Case Study,” in *3rd International Conference on Software Engineering Advances, ICSEA*, 2008, pp. 1–6.

[109] K. Ummah, K. Mutijarsa, and W. Adijarto, “System Security Requirement Identification of Electronic Payment System for Angkot using NIST SP 800-160,” 2016.

[110] “Application of Model Oriented Security Requirements Engineering Framework for Secure E-Voting.”

[111] A. Zuccato, N. Daniels, and C. Jampathom, “Service Security Requirement Profiles for Telecom How Software Engineers may tackle security,” 2011.

[112] M. Riaz, S. Elder, and L. Williams, “Systematically Developing Prevention , Detection , and Response Patterns for Security Requirements,” 2016.

[113] J. Jabeen *et al.*, “Incorporating Artificial Intelligence Technique into DSDM.”

[114] M. Kamal, A. J. Davis, L. R. Pietron, J. Nabukenya, and T. V Schoonover, “Collaboration Engineering For Incident Response Planning : Process Development,” pp. 1–10, 2007.

[115] H. Schmidt, “Threat- and Risk-Analysis During Early Security Requirements Engineering,” 2010.

[116] R. M. Gasca and A. R. M. S. N, “A Model-Driven Engineering approach with Diagnosis of Non-Conformance of Security Objectives in Business Process Models.”

[117] T. Hesse, G. Stefan, T. Roehm, B. Paech, K. Schneider, and B. Bruegge, “Semiautomatic Security Requirements Engineering and Evolution using Decision Documentation , Heuristics , and User Monitoring,” pp. 1–6, 2014.

[118] M. Sadiql, “Software Risk Assessment and Evaluation Process (SRAEP) using Model Based Approach,” pp. 171–177, 2010.

[119] F. Dalpiaz, E. Paja, and P. Giorgini, “Security Requirements Engineering via Commitments,” *Proc. Fifth Int. i* Work.*, pp. 1–8, 2011.

[120] S. Yahya, “A Review on Tool Supports for Security Requirements Engineering,” pp. 190–194, 2013.

[121] K. Beckers, S. Faßbender, and M. Heisel, “Using Security Requirements Engineering Approaches to Support ISO 27001 Information Security Management Systems Development and Documentation,” 2012.

[122] R. Ibrahim, “Software Security Requirement Specification,” pp. 1–4, 2016.

[123] L. Dai, “An Organization-Driven Approach for Enterprise Security Development and Management,” pp. 208–215, 2011.

[124] N. R. Mead, D. Shoemaker, and J. Ingalsbe, “Teaching Security Requirements Engineering Using SQUARE,” 2010.

[125] G. R. Haron, “Extrapolating Security Requirements to an Established Software Process : Version 1 . 0,” no. December, pp. 11–14, 2011.

[126] S. Ouchani and A. Otmane, “A Security Risk Assessment Framework for SysML Activity Diagrams,” 2013.

[127] N. R. Mead and E. D. Hough, “Security Requirements Engineering for Software Systems : Case Studies in Support of Software Engineering Education,” 2006.

[128] J. Du, Y. Yang, and Q. Wang, “An Analysis for Understanding Software Security Requirement Methodologies,” 2009.

[129] M. Menzel, I. Thomas, and C. Meinel, “Security Requirements Specification in Service-oriented Business Process Management,” 2009.

[130] H. Li, X. Li, and J. Hao, “FESR : A Framework for Eliciting Security Requirements based on Integration of Common Criteria and Weakness Detection Formal Model,” 2017.

[131] N. Ikram, S. Siddiqui, and N. F. Khan, “Security Requirement Elicitation Techniques : The Comparison of Misuse Cases and Issue Based Information Systems,” pp. 36–43, 2014.

[132] T. Sven, “The Trouble With Security Requirements,” 2017.

[133] D. Mellado *et al.*, “Automated Support for Security Requirements Engineering in Software Product Line Domain Engineering,” pp. 224–231, 2009.

[134] S. Liu, “Security Requirement Engineering using Structured Applications,” 2017.

[135] M. Umair, A. Khan, and M. Zulkernine, “On Selecting Appropriate Development Processes and Requirements Engineering Methods for Secure Software,” 2009.

[136] N. R. Mead, “Incorporating Security Requirements Engineering into the Dynamic Systems Development Method,” pp. 949–954, 2008.

[137] L. Yin and F.-L. Qiu, “A novel method of security requirements development integrated common criteria,” *2010 Int. Conf. Comput. Des. Appl.*, vol. 5, no. Iccda, pp. V5-531-V5-535, 2010.

[138] K. Khanmohammadi and S. H. Houmb, “Business Process-based Information Security Risk Assessment,” 2010.

[139] S. Markose, X. F. Liu, and B. Mcmillin, “A Systematic Framework for Structured Object-Oriented Security Requirements Analysis in Embedded Systems,” pp. 75–81, 2008.

[140] D. Hatebur, M. Heisel, and H. Schmidt, “A Pattern System for Security Requirements Engineering,” 2007.

[141] P. Salini and S. Kanmani, “Elicitation of Security Requirements for E-Health System by Applying Model Oriented Security Requirements Engineering (MOSRE) Framework,” *ACM Second Int. Conf. Comput. Sci. Eng. Inf. Technol. CCSEIT*, pp. 126–131, 2012.

[142] T. Ahmed and A. R. Tripathi, “Specification and Verification of Security Requirements in a Programming Model for Decentralized CSCW Systems,” vol. 10, no. 2, pp. 1–34, 2007.

[143] M. Riaz, J. Slankas, J. King, L. Williams, and N. Carolina, “Using Templates to Elicit Implied Security Requirements from Functional Requirements A Controlled Experiment,” pp. 18–19, 2014.

[144] M. Giacalone, F. Massacci, F. Paci, and R. Perugino, “Security Triage : An Industrial Case Study on the Effectiveness of a Lean Methodology to Identify Security Requirements.”

[145] R. Bloem, “Towards a Secure SCRUM Process for Agile Web Application Development.”

[146] S. G. Yoo, H. P. Vaca, and J. Kim, “Enhanced Misuse Cases for Prioritization of Security Requirements,” pp. 1–10, 2017.

[147] J. Viega, “Building security requirements with CLASP,” *ACM SIGSOFT Softw. Eng. Notes*, vol. 30, p. 1, 2005.

[148] R. De Landtsheer and A. Van Lamsweerde, “Reasoning About Confidentiality at Requirements Engineering Time,” pp. 41–49, 2005.

[149] M. Kainerstorfer, J. Sametinger, and A. Wiesauer, “Software Security for Small Development Teams – A Case Study,” pp. 5–7, 2011.

[150] K. Beckers, I. Côté, and L. Goeke, “A Catalog of Security Requirements Patterns for The Domain of Cloud Computing Systems,” *29th Symp. Appl. Comput.*, pp. 337–342, 2014.

[151] A. Rashid, S. Asad, A. Naqvi, M. Edwards, and M. A. Babar, “Discovering ‘ Unknown Known ’ Security Requirements,” 2016.

[152] K. Beckers, M. Heisel, F. Moyano, and C. Fernandez-gago, “Engineering Trust- and Reputation-based Security Controls for Future Internet Systems,” pp. 1344–1349, 2015.

[153] R. Crook, D. Ince, L. Lin, and B. Nuseibeh, “Security requirements engineering: When anti-requirements hit the fan,” *Proc. IEEE Int. Conf. Requir. Eng.*, vol. 2002–Janua, pp. 203–205, 2002.

APPENDIX

IN THE NEXT TABLE, THE COULMN PRACTICE TYPE REPRESENTED BY:

A : Security Requirements documents practices

B : Security Requirements elicitation practices

C : Security Requirements analysis and negotiation practices

D : Describing Security Requirements Practices

E : Security System Modelling Practices

F : Security Requirements Validation Practices

G : Security Requirements Management Practices

1. Appendix A (List of Primary Studies)

No.	Primary studies	Year	Practice type	Empirical type	Publication channel
1	"A comprehensive pattern-oriented approach to engineering security methodologies" [62]	2015	A, B, C, D, E, F	Case study	journal
2	"An extensible pattern-based library and taxonomy of security threats for distributed systems" [88]	2014	C, G	Case study	journal
3	"Investigation of IS professionals' intention to practise secure development of applications" [74]	2007	B, C	No	journal
4	"Validating the enforcement of access control policies and separation of duty principle in requirement engineering" [89]	2007	C, E, F	No	journal

5	"Implementation of cyber security for safety systems of nuclear facilities" [75]	2016	B, C, F, G	No	journal
6	"Secure information systems development – a survey and comparison" [76]	2005	B, C, E	No	journal
7	"The practical application of a process for eliciting and designing security in web service systems" [63]	2009	A, B, C, D, E, F, G	Case study	journal
8	"Capturing security requirements for software systems" [3]	2014	A, B, C, E	Case study	journal
9	"Formal methods, techniques and tools for secure and reliable applications" [100]	2005	A, E	No	journal
10	"Human resource information systems- Information security concerns for organizations" [64]	2013	A, C	No	journal
11	"Deriving business processes with service level agreements from early requirements" [77]	2011	B, C, E	Case study	journal
12	"Application of contract-based security assertion monitoring framework for telecommunications software engineering" [65]	2011	A, B, C, E, F	Case study	journal
13	"When security meets software engineering- a case of modelling secure information systems" [78]	2005	B, C, E	Case study	journal
14	"Towards the design of secure and privacy-oriented information systems in the cloud- Identifying the major concepts" [79]	2014	B, C	No	journal

15	"An integrated risk measurement and optimization model for trustworthy software process management" [90]	2012	C, F	Case study	journal
16	"The 'phasing-in' of security governance in the SDLC" [80]	2008	B	No	Conference
17	"On the secure software development process- CLASP, SDL and Touchpoints compared" [40]	2009	A, B, C, E	Case study	journal
18	"An engineering process for developing Secure Data Warehouses" [81]	2009	B, C, D, E, F	Case study	journal
19	"Regulatory-based development processes for software security in nuclear safety systems" [66]	2010	A, B, C, F, G	No	journal
20	"A survey of security issues for cloud computing"[101]	2016	A, E, F	Case study	journal
21	"An extensible analysable system model" [83]	2008	C, E	Case study	journal
22	"Empirical evaluation of a cloud computing information security governance framework" [91]	2015	A, C, G	Case study	journal
23	"Validating the enforcement of access control policies and separation of duty principle in requirement engineering"[89]	2016	B, F	Case study	journal
24	"A technique for expressing IT security objectives" [67]	2006	A, C, D, E	Case study	journal
25	"Implementing information security best practices on software lifecycle processes- The ISO-IEC 15504 Security Extension" [57]	2015	A, B, F, G	Case study	journal

26	“From requirements negotiation to software architecture decisions”[102]	2005	A, C, D	Case study	journal
27	“A survey of approaches combining safety and security for industrial control systems” [82]	2015	B, C, D, F, G	Case study	journal
28	“Generic modelling of security awareness in agent based systems” [92]	2013	B, C, D, E, F	Case study	Conference
29	“A Case-based Management System for Secure Software Development Using Software Security Knowledge” [94]	2015	G	Case study	journal
30	“Secure Tropos framework for software product lines requirements engineering” [95]	2014	G	Case study	journal
31	“Formal specification and integration of distributed security policies”[103]	2017	A, C	No	journal
32	“The practical application of a process for eliciting and designing security in web service systems”[63]	2009	C, E, F	No	journal
33	“A technique for expressing IT security objectives”[67]	2007	B, D, E	Case study	journal
34	“A novel method- Ontology-based security requirements engineering framework”[104]	2016	A, B, C, D, E, F, G	No	Conference
35	“A model based security requirements engineering framework applied for online trading system”[105]	2011	A, B, C, E, F	Case study	Conference
36	“Measuring The Software Security Requirements Engineering Process” [106]	2012	B, C, F	Case study	Conference

37	“Extending V-model practices to support SRE to build Secure Web Application”[107]	2014	B, C, E, F	No	Conference
38	“Research on Security Requirements Engineering Process”[96]	2009	B, C	No	Conference
39	“Security Requirements Engineering Process for Software Product Lines- A Case Study”[108]	2008	B, C	Case study	Conference
40	“System Security Requirement Identification of Electronic Payment System for Angkot using NIST SP 800-160”[109]	2016	A, B, C, F, G	Case study	Conference
41	“Application of Model Oriented Security Requirements Engineering Framework for secure E-Voting”[110]	2012	A, B, C, E, F	Case study	Conference
42	“Service Security Requirement Profiles for Telecom: How Software Engineers May Tackle Security”[111]	2011	A, B, C	Case study	Conference
43	“Systematically Developing Prevention, Detection, and Response Patterns for Security Requirements”[112]	2016	A, B, C	Case study	Conference
44	“Incorporating artificial intelligence technique into DSDM”[113]	2014	A, B, C, E, F	Case study	Conference
45	“Collaboration Engineering For Incident Response Planning: Process Development and Validation”[114]	2007	A, B, C, F	Case study	Conference
46	“Threat- and Risk-Analysis During Early Security Requirements Engineering”[115]	2010	B, C, E	Case study	Conference

47	"A Model-Driven engineering approach with diagnosis of non-conformance of security objectives in business process models"[116]	2011	B, C, F	Case study	Conference
48	"Semiautomatic security requirements engineering and evolution using decision documentation, heuristics, and user monitoring"[117]	2014	A, B, C	Case study	Conference
49	"Software risk assessment and evaluation process (SRAEP) using model based approach"[118]	2010	A, B, C, E, G	Case study	Conference
50	"Security requirements engineering via commitments"[119]	2011	A, B, E	Case study	Conference
51	"Security Requirements Engineering- A Framework for Representation and Analysis"[14]	2008	A, B, C, E, F, G	Case study	Journal
52	"A review on tool supports for security requirements engineering"[120]	2013	A, B, C, E, F, G	Case study	Conference
53	"Using Security Requirements Engineering Approaches to Support ISO 27001 Information Security Management Systems Development and Documentation" [121]	2012	A, B, C	Case study	Conference
54	"A Hybrid Threat Model for Software Security Requirement Specification" [122]	2016	A, B, C, E	Case study	Conference
55	"An Organization-Driven Approach for Enterprise Security Development and Management" [123]	2011	B, E	Case study	Conference

56	"Teaching Security Requirements Engineering Using SQUARE" [124]	2009	A, B, C, F	Case study	Conference
57	"Extrapolating security requirements to an established software process- Version 1.0" [125]	2011	A, B, C, F	Case study	Conference
58	"A Security Risk Assessment Framework for SysML Activity Diagrams" [126]	2013	A, E, F	Case study	Conference
59	"Security Requirements Engineering for Software Systems- Case Studies in Support of Software Engineering Education" [127]	2006	B, C	Case study	Conference
60	"An Analysis for Understanding Software Security Requirement Methodologies" [128]	2009	A, B, C, E, F	Case study	Conference
61	"Security Requirements Specification in Service-Oriented Business Process Management" [129]	2009	A, B, D, E	Case study	Conference
62	"FESR- A Framework for Eliciting Security Requirements Based on Integration of Common Criteria and Weakness Detection Formal Model" [130]	2017	A, B	Case study	Conference
63	"Security Requirements Engineering in the Wild- A Survey of Common Practices" [59]	2011	A, B, E	No	Conference
64	"Security requirement elicitation techniques- The comparison of misuse cases and issue based information systems" [131]	2014	B	Case study	Conference

65	“The Trouble with Security Requirements” [132]	2017	B, C, D	Case study	Conference
66	“Automated Support for Security Requirements Engineering in Software Product Line Domain Engineering” [133]	2009	A, B, C, D, F, G	Case study	Conference
67	“Security Requirement Engineering Using Structured Object-Oriented Formal Language for M-Banking Applications” [134]	2017	A, B, D, E, F	Case study	Conference
68	“On Selecting Appropriate Development Processes and Requirements Engineering Methods for Secure Software” [135]	2009	A, B, C, E, F	No	Conference
69	“Incorporating Security Requirements Engineering into the Dynamic Systems Development Method” [136]	2008	A, B, C	Case study	Conference
70	“A novel method of security requirements development integrated common criteria” [137]	2010	A, B, C, D	No	Conference
71	“Business Process-Based Information Security Risk Assessment” [138]	2010	C, D	Case study	Conference
72	“A Systematic Framework for Structured Object-Oriented Security Requirements Analysis in Embedded Systems” [139]	2008	B, C, E	Case study	Conference
73	“A Pattern System for Security Requirements Engineering” [140]	2007	A, B, C, D	Case study	Conference
74	“Establishing Trustworthiness in Services of the Critical	2005	A, B, C, D, E, F, G	Case study	Conference

	Infrastructure through Certification and Accreditation" [68]				
75	"Extending XP practices to support security requirements engineering" [58]	2006	A, B, C, D, E, F	Case study	Conference
76	"A Meta-Model for Usable Secure Requirements Engineering" [93]	2010	B, C, E, F	Case study	Conference
77	"Common criteria compliant software development (CC-CASD)" [69]	2013	A, F	Case study	Conference
78	"Secure and Usable Requirements Engineering" [70]	2009	A, B, C, F	Case study	Conference
79	"Which Security Requirements Engineering Methodology Should I Choose Towards a Requirements Engineering-based Evaluation Approach" [86]	2017	B, D, F	No	Conference
80	"DIGS – A Framework for Discovering Goals for Security Requirements Engineering" [87]	2016	B	Case study	Conference
81	"Towards Usable Cyber Security Requirements" [71]	2009	A, B, C, F	Case study	Conference
82	A Framework for Security Requirements Engineering [72]	2006	A, B, C, D, F	No	Conference
83	"Building Problem Domain Ontology from Security Requirements in Regulatory Documents" [73]	2006	A, B, C, D, E, F, G	Case study	Conference
84	"Elicitation of Security Requirements for E-Health System by applying Model Oriented Security Requirements	2012	B, C, D, E	Case study	Conference

	Engineering (MOSRE) Framework" [141]				
85	"Specification and Verification of Security equirements in a Programming Model for Decentralized CSCW Systems" [142]	2007	A, F	No	Journal
86	"Using Templates to Elicit Implied Security Requirements from Functional Requirements í A Controlled Experiment" [143]	2014	B	Case study	Conference
87	"Security Triage- An Industrial Case Study on the Effectiveness of a Lean Methodology to Identify Security Requirements" [144]	2014	B, C	Case study	Conference
88	"Towards a Secure SCRUM Process for Agile Web Application Development" [145]	2017	A, B, C, F	Case study	Conference
89	"Enhanced Misuse Cases for Prioritization of Security Requirements" [146]	2017	B, C	Case study	Conference
90	"A Risk Assessment Framework for Automotive Embedded Systems" [49]	2016	B, C	Case study	Conference
91	"Building Security Requirements with CLASP" [147]	2005	B, C	Case study	Conference
92	"Reasoning About Confidentiality at Requirements Engineering Time" [148]	2005	B, C, F	Case study	Conference
93	"Software Security for Small Development Teams – A Case Study" [149]	2011	A, B, C	Case study	Conference

94	“A Catalog of Security Requirements Patterns for the Domain of Cloud Computing Systems” [150]	2014	A, B, C	Case study	Conference
95	“Discovering “Unknown Known” Security Requirements” [151]	2016	B, C	Case study	Conference
96	“Engineering Trust- and Reputation-based Security Controls for Future Internet Systems” [152]	2015	B, C, E, F	Case study	Conference

2. Appendix B (Altered Sommerville RE Practices to SRE practices)

2.1. Security Requirement Documentation Practices

No	Practices
1	Define a standard security document structure
2	Explain how to use the security document
3	Include a summary of the security requirements
4	Make a business case for the system with respect to security
5	Define specialized security terms
6	Make document layout readable
7	Use languages simply and concisely to explain security requirement. (identification/authentication/authorization/ immunity/privacy/integrity)
8	Help readers find information

2.2. Security Requirement Elicitation Practices

No	Practices
1	Assess System Security Feasibility
2	Take into consideration organizational and political issues
3	Determine and consult stakeholders of the system
4	Record security requirements sources
5	Identify the operating environment of the system
6	Use concerns related to business to motivate security requirements elicitation
7	Search for domain constraints
8	Record rationale for security requirements

9	Gather security requirements from different and various views
10	Prototype poorly understood security requirements
11	Use hypothetical cases to elicit security requirements
12	Identify operational processes
13	Reuse security requirements

2.3. Security Requirements analysis and negotiation Practices

No	Practices
1	Define security of system boundaries
2	Make use of checklists to analyze security requirements
3	Provide software to support negotiations
4	Consider conflicts and how to resolve them
5	Identify priorities in security requirements
6	Sort out security requirements through a multi-dimensional approach
7	Employ interaction matrices to identify conflicts and overlaps
8	Assess risks of security requirements

2.4. Describing Security Requirement Practices

No	Practices
1	Identify standard templates to describe security requirements
2	Use simple and concise languages
3	Use diagrams appropriately
4	Supplement natural language with other description of security requirement
5	Identify quantitative/ qualitative requirements of security

2.5. System Security Modelling Practices

No	Practices
1	Develop system models that are complementary
2	Model the system's security environment
3	Model the system security architecture
4	Use structured methods for system security modelling
5	Use a data dictionary
6	Document the links between stakeholder requirements and system models

2.6. Security Requirement Validation Practices

No	Practices
1	Make sure the security requirements document satisfies your standards
2	Organize inspections for security requirements
3	Use multi-disciplinary teams to assess security requirements
4	Identify validation checklists
5	Use prototyping to animate security requirements
6	Write a draft user manual
7	Suggest test cases for security requirements
8	Paraphrase system security models

2.7. Security Requirement Management Practices

No	Practices
1	Specifically define each security requirement
2	Identify policies for management security requirements
3	Define traceability policies
4	Maintain a traceability manual
5	Make use of a database to handle security requirements
6	Define policies for change management
7	Identify global system security requirements
8	Identify volatile security requirements
9	Record rejected security requirements

2.8. Security Requirement Engineering for Critical Systems Practices

No	Practices
1	Develop checklists for safety requirement
2	Engage external reviewers in the validation process
3	Identify and analyze hazards
4	Obtain safety requirements from hazard analysis
5	Cross-check operational and functional requirements against safety requirement
6	Specify systems using a formal specification
7	Collect incident experience
8	Learn from incident experience
9	Develop a culture of organizational safety.

3. Appendix C (Requirement Engineering Security Maturity Model Practices)

Security Requirement Documentation Practices		**Security Requirement Elicitation Practices**	
1	Define a standard security document structure	5	Identify security objectives and dependencies.
2	Explain how to use the security document	6	Identify and consult system stakeholders
3	Define Security Definitions, Quality Gates and security policy of the Org.	7	Identify threats and develop artifacts.
4	Make a separate Information security policy	8	Record security requirements sources (Identify Resources and Trust Boundaries)
5	Define and document a project's security bug bar. (establish a minimum level of quality)	9	Define the system's operating environment (Specify Operational Environment)
6	Include a summary of the security requirements	10	Identifying User Roles and Resource Capabilities
7	Make a business case for the system with respect to security	11	Use business concerns to drive security requirements elicitation
8	Define specialized security terms	12	Identify and consult security experts
9	Make document layout readable	13	Select an elicitation method using a systematic tradeoff analysis approach to elicit SRs
10	Use languages	14	Look for domain constraints
11	Help readers find information	15	Record security requirements rationale
12	Make the document easy to change	16	Collect security requirements from multiple viewpoints
Security Requirement Elicitation Practices		17	Prototype poorly understood security requirements
1	Assess System Feasibility with respect to security.	18	Use scenarios to elicit security requirements
2	Demonstrate of exploitability.	19	Define operational processes
3	Be sensitive to organizational and political policy consideration	20	Reuse security requirements
4	Identify and consult system stakeholders		

Security Requirements analysis and negotiation Practices		**Security Requirement Modelling Practices**	
1	Define security of system boundaries	1	Develop complementary system models with respect to security
2	Use checklists for SRs analysis	2	Model the system's security environment
3	Perform Security & Privacy Risk Assessment	3	Model the system security architecture
4	Ensures that the access requirements are consistent, complete and conflict-free.	4	Use security pattern template to model SRs
5	Use interaction matrices to find conflicts and overlaps	5	Model the Threats of System
6	Negotiate quality gates with different stakeholders	6	Use structured methods for system security modelling
7	Prioritize security requirements	7	Use problem frames to model SRs
8	Classify security requirements using a multi-dimensional approach	8	Use a data dictionary
9	Provide software to support negotiations	9	Document the links between stakeholder requirements and system models
10	Review Security Requirements	10	Clearly define the properties that we hope to prevent attackers from violating.
Describing Security Requirement Practices			
1	Define standard templates for describing SRs.		
2	Use languages simply and concisely		
3	Be adequate as possible (explicit, precise, complete and non-conflicting)		
4	Use diagrams appropriately		
5	Describe a prototype model for data security based on metadata		
6	Describe abuse cases by examples		
7	Supplement natural language with other description of SRs.		
8	Specify security requirements quantitatively		

Security Requirements Validation Practices		Security Requirements Management Practices	
1	Check that the security requirements document meets your standards	1	Uniquely identify each security requirement
2	Organize security requirements inspections	2	Define policies for security requirements management
3	Use multi-disciplinary teams to review SRs.	3	Define traceability policies
4	Define validation checklists	4	Maintain a traceability manual
5	Use prototyping to animate security requirements	5	Use a database to manage SRs
6	Perform periodic security assessments and review the quality of security activity	6	Define change management policies
7	Write a draft user manual	7	Identify global system security requirements
8	Propose security requirements test cases	8	Identify volatile security requirements
9	Paraphrase system security models	9	Record rejected security requirements
		10	Manage risks of requirements from laws and regulations

4. Appendix D (Explanation of different categories of security practices at requirement phase)

4.1 More details about practices for document security requirements:

There are several security requirements documentation practices that mentioned in previous studies. Some of these practices are:

- **Document the Conceptual security artifacts**.

In requirements engineering, we concentrate on what needs to be done not how to do it. So, the term “artifacts” refers to the “what” objects of security requirements engineering need to be documented. Security artifacts take the concept of assets and harm threats to those assets. There is a collection of security artifact which can be determined or documented explicitly. These security artifacts can have a different format such as structured (e.g. a catalog) or unstructured [62].

- **An information security policy**

Security policies and procedures are crucial for any organization. It is considered to be the cornerstone of any information security program. It reflects the objectives of organization for security and the accepted level about the strategy of management for securing information [62]. Security policy or security protocol is a document that contains how does the organization protect its assets, including physical or information technology assets. It is utilized to determine the rules, laws, practices, and regulations that decide how to manage, safeguard and move sensitive information to the organization [65]. This document is often updated in each time the technology and employee requirements change. The

Security policy of an organization may contain an acceptable use policy, an explanation on how the organization intend to educate its employees on the way of preserving the organization assets, a description about the way of carrying out or enforcing security measurements, and a process of how to evaluate the effectiveness of the security policy to ensure mandatory correction [64]. Another study [40] made a comparison between the processes of secure software development such as Touchpoints, SDL, and CLASP. In this study, some activities of these processes are mentioned. For example, the Clasp process indicates the importance of the organizational policy document. It also points out that security policy has to be considered as a baseline for all software projects [40]. In addition, some development processes of secure software such as SDL has clarified the importance of addressing the logistics aspects and to document logistics aspects available in the Organization since this will ensure the organization has the necessary tools for securing the system and specify the type of security bugs that may be addressed in the organization [40]. Some recommendations provided by previous research about the security policies include that they should determine the impact of those polices on the stakeholders; continuous reviews and updates to those polices should be made whenever there are any changes in the structure of organization or that the organization merges with another organization; the revised security policies should be distributed to all relevant stakeholders to ensure all stakeholders are conscious about those polices.

- **Initiate templates for documentation of security requirement specification**

This is achieved by using the repository of "Software Requirement Specification Template". The security artifact can be used to derive the documentation which follows

the standard of software requirements specification IEEE 1998-830. Thus, in this practice, it also concentrates on using a standard in the documentation of security requirements [63].

- **Express security requirements as positive statements** instead of negative statements: the benefit of expressing security requirements as positive statements can help in the satisfaction of those security requirements [3].
- **Security catalog:** The security requirement document should include a catalog of security which involves the security models for the threats and the corresponding security requirements. This catalog contains the security threats that could be exploited by the malicious users or attackers. Each security threat will be described in this catalog; the ways the attackers can use to breach security of the system. This does not mean to cover all the possible threats that can affect the system but, it means this catalog will help in cover a broad range of threats that could be harmful to the system. Such as that catalog is STRIDE threats modeling.

- **Develop documented access control procedures:** This will help implement security measures that handle the process of software engineering in a defined and managed way. Thus, the development process assurance will be improved [75].
- **Document the security-related procedures**: There are standards for these procedures such as standard development procedures that contain practical checklist to ensure the design of the system without undocumented functions [66].
- **Protection Profile (PP)** is a document that is considered by ISO/IEC 15408 and the Common Criteria (CC) to be a portion of the certification process. It specifically demands some external party like government or standard body. It has to protect each security objective. In addition, it has to carry out security functionality in order to present

every security functional requirement. Furthermore, it has to supply assurance evidence to show that the developed product has met the “evaluation assurance level” which is defined in the protection profile [67]. “**Security Target** includes an overview of the product and product's security features, an evaluation of potential security threats and the vendor's self-assessment detailing how the product conforms to the relevant Protection Profile at the Evaluation Assurance Level the vendor chooses to test against”.

- **Use standard for documenting security requirements:** There is some standard that deals with security requirements such as ISO/IEC 15504-5 and ISO/IEC 27002. For example, ISO/IEC 27002 talks about the information security policies. Moreover, ISO/IEC 15504-5 has a Documentation process that can be utilized to preserve the recorded information that is achieved by the information security activities [57].

4.2 More details about practices for elicitation of security requirements:

- **Identify the prescriptive of security requirements** which are equivalent to security policies with high level perspective. At the beginning, these security policies are obtained from an organization and set up immediately to a given project. Then, measures for the overall security are dictated. The implemented security policies with high-level will work as the security requirements for the developed system.
- **Resultant security requirements** are the result of assessment for the system. In this activity, the developer will consider any attacks to the system. After that, a trade-off will be done between the output requirements with other non-functional requirements. In addition, it is also mandatory to consider what protective measures are necessary in the developed system.

- **Secure Development Application** will enhance the capturing of security requirements by incorporating security aspects throughout the development lifecycle of the application. SDA can be seen as an improved set of guidelines and practices that could be integrated to existing methodology of software development of the organization [74].
- **Assessing security impact on the system integrity:** First, we need to know what does system integrity mean? To answer this question, system integrity is the state of the system "where its intended functions are being performed without degradation or being impaired by other changes or disruptions to its environments." Thus, in this practice, we need to know the impact of security on the system under different circumstances.
- **Establish the path of attack analysis** in order to identify what are the internal vulnerabilities of the system. The reason behind this is to avoid the amount of time that it takes when using technical assistance of the platform manufacturer. The attack path can be revealed by reviewing the documentation such as the interfaces description of the system, the configuration diagrams of the system, and hardware configuration of the system. By doing these reviews, we can identify the path for those attacks [75].
- **Identify what should be implemented on safety system security capabilities.** The "security team identified the security controls for the system according to the result of security assessment. The security team assigned a priority to a security control due to the relationship of attack scenarios [75]."
- **Identify security controls** that can be used to rule out vulnerabilities in the system's pathways and exclude those vulnerabilities from the system. This activity is used to identify accessible pathways to the cabinet of the system whether physically or logically [75].

- **Perform security assessment:** this practice will help in identifying potential security vulnerabilities at requirements phase of system lifecycle. The outcome of security assessment is to determine the security controls for the system. The risks of intrusions to the system have to be assessed and managed during the software development lifecycle. In addition, the focus should be done on the system functionalities besides the involved people who used the system since the attacker tries to exploit every weakness that might exist in the system [75][66].
- The extended UML is used to present security notions. There are two basic tools which are the use cases and the corresponding scenarios to construct threat models and elicit security requirements [76].
- **Web Services Security Requirements (WSSecReq) approach** has certain practices to be done in order to elicit security practices such as the identification of system security threats,; construction of a group of security artifacts which are inter-related; determining misuse cases that are collected in security profiles; applying risk analysis methodology (ISO-compliant 15408); and using reusable approach to specify security requirements [63].
- **Elicit adequate security requirements** during the requirements engineering process with the aid of previous security knowledge. A security catalog, based on problem frames, is constructed for this purpose. This elicitation of security requirements can be done by analyzing the assets that need to be protected, analyzing the threats from which these assets should be protected, considering security while eliciting the requirements of software systems using problem frames, and identifying security requirements with the aid of previous security knowledge through constructing **a security catalog** for this purpose. The

security catalog consists of problem frame models for threats and the corresponding security requirements. Threats are modeled using abuse frames while security requirements are modeled using security problem frames [3].

- **The Activities of security requirement elicitation** can be done by model the security requirements which are planned to alleviate the threats leading to vulnerabilities. Thus, to model these security requirement, we use security problem frames which can be found in the security catalog. If the domain of the security problem frame is not available in the catalog, we follow another technique that is proposed by Heley to elicit security requirements. Another way to mitigate the discovered threats is by using the trust assumptions. The idea behind the trust assumptions is to make the properties of system domains trusted at acceptable level that turn out the system to be safe from vulnerabilities. The trust assumption is only used when there is a dilemma with the analysts that make them incapable to go further with the problem since there is a belief this problem can be solved in another context.
- **"introduces a model syntax checker for specifying security protocols** and presents the communication behavior of the communication principals under the Dolev–Yao threat model. The author invites further studies to be carried out that consist of applying the calculus of communicating systems methodologies more rigorously and developing more formal tools for the analysis of security and cryptographic protocols."
- At the beginning of requirement engineering, we first try to analyze the context of the organization within which a system will eventually operate. We identify the goals that have to be fulfilled by analyzing the domain of the existing actors and the dependencies between

different actors. And for security purpose, we require to rationale about delegation of authority and trust relationships [77].

- **Come up with functional dependencies and the requirements of security and trust.** This can be done by employing the Secure Tropos modeling framework that comes up with the functional dependencies. In addition, it also derives the security and trust requirements [77].
- **Fully understanding of possible security risks.** This can be done by using misuse cases that can be used in security requirement elicitation. Furthermore, the identified misuse cases require to be prioritized in order to make a balancing between risk and cost due to the large number of misuse cases that can be produced. After identifying the misuse cases, we go with the identification of security violation scenarios. There are different techniques for identifying the security violation scenarios. One of these techniques, is by using an attack tree [65].
- **Use the Tropos concepts that deal with dependency, task, goal, capability and resource which are also expanded with security concerns.** In order to achieve security constraint during software development, we can use secure goals for that help to achieve that purpose. In addition, there is also a secure task for satisfying a secure goal in a specific way. Furthermore, a secure resource is a resource which is relevant to security constraint or secure entity.
- **There is an existing literature to elicit related privacy and security properties** such as the "European Commission Draft Report on Security Issues in Cloud Computing" [14], "CSA report" [13], "NIST guideline" [29], and Other relevant literature"**.** Thus, we can use these literatures to elicit security requirements [79].

- **Use business objectives to derive security requirements.** This can be done by doing some practices such as identifying requirements for user access identity or user access authentication which is considered to be a role-based access controls; defining what are the level of data privacy that are associated with the project; and identifying the abuse cases beside the use cases [80].
- "Vulnerabilities are discovered by analyzing threats to and attacks on both the requirements and the DW repository. One of the best-known techniques through which to model threats/attacks are attack trees, which contain threats, and their possible attacks [153]."
- **Determine what are the general security requirements?** And what are the specific security requirements? Specific security requirements are requirements type that deal with some aspects such as Access control requirement, Data communication requirement, Maintenance requirement, and Retirement requirement. Access control requirement has the structure of a collection of, property, knowledge, or personal features. Using access control is preferred than using just a password.
- There are some tasks to identify security requirements such as by establishing the structure of information security governance; by determining the roles of participant; and by identifying profiles that include the structure of security governance. In addition to the previous tasks, there are also some practices such as creating a committee for security governance and defining top-level security policies. These top-level polices of security include the organization's goals and strategy of business.

4.3 More details about practices for analysis and negotiation of security requirements:

- **Use the model-based paradigm** to analyze the system needs and requirements.

  In order to make a discussion with different stakeholders, we can use various conceptual or architectural models. These models can be generated to reflect the requirements. Thus, using these models will help in the analysis of security requirements with different stakeholders.

- **knowledge of security** is a basic necessity prior to practicing security requirement:

  The analyst should have background on how to identify and analyze the system assets, threats, vulnerabilities and requirements. One of the characteristics of security requirements is to be adequate as possible. Of the other characteristics are explicitly, precision, and completion and that they are conflict-free with other requirements.

- **AuthUML** is used to analyze access control requirements at requirement engineering in order to make sure the requirements are consistent, complete, and conflict-free for the application being developed. AuthUML will analyze the access due to use cases and operations. AuthUML helps in detecting easily the inconsistencies and conflicts of access control requirements in small systems due to have just few number of entities and engineers who wrote those requirements. On the other hand, in extensive systems, inconsistencies and conflicts in access control requirements can be detected using AuthUML by specifying rules for that purpose. Since, detecting inconsistencies and conflict as early as possible will help to prevent them from spreading to next phases of the lifecycle. AuthUML has four phases to be followed which are tackling access control requirements, make sure about

completeness, consistency, and conflict-free for both accesses to use cases and access for operations, and ensuring compliance of authorization requirements with the Principle of Separation of Duties (SoD).

- **Authorization requirements** which are specified in the Unified Modeling Language (UML) can be analyzed by using logical-based framework. This will ensure consistency, completion of the access requirement, as well as that they are conflict-free [89].
- There are different ways to minimize the level of security expertise that is required during the development of the system such as by using the threats library or attack taxonomies. Threats libraries can have two forms which are structured or unstructured. These libraries have been realized to be effective in different industry scenarios. On the other hand, attack taxonomies represent a classification scheme that will assist the developer to find out the relevant attacks to the system easily. On the other hand, penetration test is used to check if the attacker can endanger the system. It helps to identify known and unknown vulnerabilities to the system.
- There is a need to analyze the requirements in order to detect any potential vulnerabilities to the system. This process consists of three models that gather information of the system from various perspectives which are multilevel object model that expresses static features; multilevel dynamic model that expresses dynamic features; and multilevel functional model that expresses transformation features of the system [76].
- **Comprehensive risk analysis** will enhance security. This can be done by following certain steps. First, by identifying functions of the system, boundaries for the system, and criticalities which exist in the system. The benefits of doing that are to minify the risks to an organization's data and information systems. Secondly, by identifying vulnerabilities

and security threats. This requires the methodical process of checking and documenting the security posture of an organization's information systems. Thirdly, by calculating risk factors. This can be achieved by the analysis of possible dangers that the firm faced. Fourthly, by making sure all risks that have a critical negative impact are investigated. Lastly, by documenting the outputs of the risk analysis [64].

- **Risk identification** can be done using different approaches such as Checklist approach. The software process can use checklist to identify risk and assessing risk exposure in a quick and low-cost way [90].
- In this study [83], they showed two analysis techniques for security requirements. The first technique is Conditional Reachability Analysis which is created to specify the locations that can be reached by an actor. The second technique is Log-trace Reachability Analysis which has an input as log file format and depending on this it will specify the place of actors, what actions they perform, and what data can they access.
- The system can be analyzed by identifying the status of the system before, under, and after the attack. First, *Before the attack,* this can be achieved by identifying the system parts that can be accessed by which users. In addition, by identifying the location that can be reached by the user. Furthermore, by identify potential flows in an access-control system and determine who has the authority to access specific locations or restore data. All of these can be done by doing conditional reachability analysis (CRA). Second, ***After the attack*** by be ready for any potential attack. This planning can derive in various forms, it can has the form of actions logging to be performed by users. In addition, by Identify what might have occurred (unrecognized) in between two log entries. In this situation, the log-trace

reachability analysis might be helpful. The log-trace reachability analysis affords the investigators with this knowledge.

In this study [91], there are some tasks to be consider during the analysis of security requirements. Task A: define Information of security requirements. The security requirements specification has storage service's which are intimately relevant to its technical specification. Task B: analysis of available software environment options. In this task, existing alternatives of security mechanism were analyzed to assess their security. The final choice must take into account these assessments and satisfactorily negotiate the security weight. Task 3: security risk analysis. The organization determined to use Risk Assurance Framework called ENISA's [150] to advocacy this analysis.

- At the beginning, there is a need to identifying service-related assets information and identify potential threats affecting those assets. Thus, personal information beside other information related to physical and unphysical assets were also identified. After the identification of information assets and possible threats, assessment of the risk would take place to evaluate the disclosure to risks. The quantification of risk was needed in order to come up with risk management guidelines that would reduce the top threats to the system.

4.4 More details about practices for describing security requirements:

- In order to describe security requirements, a simple micropattern textual template can be used for phase modifiers. This textual template is enhanced by adding two fields which are the Role and Artefacts. The Role describes roles of technical team who are sharing in the phase of development process. Whereas, the artefacts are work products that are in the state of development at certain phase of software development.

- **Describe abuse cases by examples:** By using examples, we can help readers to have a good vision of what might happen in different situations. Thus, we use examples to describe the abuse cases for the system.
- **Use metadata** to describe a prototype model for data security: The main goal here is to minimize the queries of user to only the users who have an access for those data. In addition, the Strategic Rationale model is made use of to describe the interests and concerns of stakeholder, and the way they could be represented by different configurations of systems and environments. Furthermore, to describe the process of concrete software development or describe a family of relevant software development process this can be done by using SPEM process metamodel. The specification of SPEM follows the structure of UML profile, and it is equipped with a complete MOF-based metamodel [81].
- **Misuse Case Description Templates** is used for representing a misuse case by text-based notation. The benefit of doing that is to describe the misuse cases in a detailed manner and more complete which provides the analysis with extra information about the security threats that is difficult to be represented in a diagram form.
- The unambiguity of protection profiles would facilitate Security Targets that elicit the letter and the intent of the protection profiles.
- **UML notations** are used to describe the processes of security. UML notation is helpful for software licensees and developers to enhance the understanding of security requirements. It also improves the linkage between licensees, regulators, and developers [66].
- **Describe security requirements using metaclasses** especially for common rational agent within systems. These metaclasses will help software engineers to describe security requirements for prevalent rational agent that exists in the systems. This will identify

security requirements that are associated with their engendered features (mobility, co-operation, and autonomy). And it will also identify the primitive concepts of modelling that needed to express them [92].

4.5 More details about practices for validating of security requirements:

- **Manual review** can be utilized to inspect impacts of security countermeasures and track them back to security requirements. This will help in making sure that these security requirements are consistent, correct, and complete. Inspections and reviews should have a structure format. This require checking requirements in a systematic manner or by using a checklist which will be used as a guidance for the process. All development stages can use that as part of verification activities in all phase (security) process patterns [62].
- **The compliance of the authorization requirement** can be validated by using AuthUML which introduce new phase using the Separation of Duty principle. It validates the obligation of the Separation of Duty over the requirement engineering. The validation of AuthUML consists of four isolated steps. The first step validates the compliance according to separate duties at requirement phase only. The second step validates the compliance whenever there are any designate users to roles. . As for the third step, it also verifies the compliance whenever the user endeavor to suppose a role to carry out a certain action. The fourth and final step also verifies the compliance whenever the user tries to carry out an operation [89].
- **Define security requirements as portion of the overall system requirements**. By doing that, we will induce the potential vulnerabilities by the functional requirements and this will help in the validation of security requirements. In addition, to make sure that the safety

system does not involve not recommended and unverified functions and review the traceability of requirements from the security perspective [75].

- **The Template of security artifact** is made use of to formulate the validation case test document of security requirements that are determined from the outset. In addition, internal validation can be carried out by performing an inspection of the specified security requirements in order to examine these security requirements that are not ambiguous, conflict-free, and the traceability of the inclusion/exclusion relationships were valid. Moreover, External validation can also be carried out by performing review meetings with the actors that have contributed to the subprocess of verifying that the collected security requirements are satisfied their interests [63].
- **One task of contracts is to validate security vulnerabilities.** In addition, contract can be used for identifying and tracking security vulnerabilities. Contracts can also be used to increase requirements-based on the assertion of security through SDLC. For instance, CB_SAMF is a kind of contract that can be incorporated into a development life-cycle to verify suspicious vulnerabilities that exist in Linux kernel and relevant device drivers [65].
- In verification step, there are two validations that needed to be done which are internal validation and external verification as presented in [81]. They verify that all requirements have been probably implemented.

Vitae

Personal Information	Name :Ashraf Mohammed Mohammed Saeed Nationality :Yemeni Date of Birth :5/2/1986 Email :Ashmhd@gmail.com	
Research Interest	Software Security, Security issues, Machine Learning (Deep learning). Early Detection of Cyber-attacks, Security and Privacy	
Education	**King Fahd University of Petroleum and Minerals**, Saudi Arabia. M.Sc. in Information and Computer Science. Major: Software Engineering. **GPA: 3.41/4** (graduated in May 2018) **Taiz University**, Yemen B.Sc. in Computer Science, Faculty of Applied Science GPA: 87.00/100	2014 – 2017 2005 - 2009
Training Programs	**Exchange Arab Universities Students training program**, Egypt Al-Zakazeek University **Web Design training program**, Egypt Al-Zakazeek University **Image Processing training Program**, Egypt Al-Zakazeek University **C - Language training Program**, Egypt Al-Zakazeek University **Oracel and Developer Course**, Yemen (SQL- PLSQL-DEVELOPER-REPORT) Digital Elite Institute. **Cisco CCNA 1 Course**, Yemen General Telecom Institute.	July 2008 July 2008 July 2008 July 2008 May 2008 July 2013

Languages	**TOEFL PBT Test**, Saudi Arabia King Fahd University of Petroleum and Minerals **Courses in English Language (Level 2, Skills, Level 3)**, Yemen Mali Institute **New-Interchange (1A & 2A & 1B & 2B)**, Yemen Yemeni Canadian Institute	2014 June 2010 2009 - 2010
Work Experience	**ALturba University** Computer Science Principle (Lecturer). **Taiz University** Computer Programming (Lecturer). **IT Consultant: at Global Community Organization,** Yemen Maintenance of Networks and Software.	Demonstrator 2012 – 2014 Demonstrator 2010 – 2012
Publications **Under final steps**	**[1] A Requirement Engineering Security Maturity Model submitted** to GSS Journal in November 20, 2018. **[2] Comparative Literature Survey of Requirements Engineering Frameworks** Ongoing work With Dr. Mahmood Niazi **[3] Multi-Scale Retenix for night-time images with low Illumination.** Ongoing work With Dr. Sabri **[4] Formal Specification and Verification of of String Transformation Combinators of DReX using Maude** Ongoing work With Dr. Musab Ahmad Alturki	
Thesis	**[1] M.Sc. Thesis**: A Requirement Engineering Security Maturity Model. Graduated in: May 2018 Advisor: Dr. Mahmood Niazi	

Developed tools and Term projects	• Control Computer Via Voice using (C#, XML). • Voice Print to Security System Using MATLAB. • Develop system for Institute Education Center. • Systematic Mapping Study for Big Data Architecture. • Comparative Literature Survey of Requirements Engineering Frameworks. • Semantic of String Manipulation in DReX language using (Maude Language) • Image Enhancement for nighttime Images (MATLAB) • Network Traffic Classification using Machine Learning (Python Language)
Honors and Awards	**[2009]** Certificate of Appreciation for being of the outstanding and second students in my patch, Programming department. [**2014**] Graduate Scholarship for 4 years from King Fahd University of Petroleum and Minerals to study M.Sc. in Computer Science.
Technical Skills	• **Programming Languages**: very comfortable in python, Assembly, C, C++, C#, Visual Basic.Net, Basics of Object-Oriented Programming (OOP). • **Databases**: MySQL, Oracle, Database Management Systems. • **System Analysis:** Unified Process &UML. • **Document Processing**: Word, LATEX • **Machine Learning tools:** Weka • **Platforms:** Windows, Linux. • **Virtualization:** VMware, VirtualBox
Graduate Courses	• Digital Image Processing 2016 • Advanced Machine Learning 2016 • Database Design & Implementation 2015 • Advanced Computer Algorithms 2015 • Software Requirements Engg 2015 • Research Methodology & Exp. Dsgn In Comp 2015 • Des & Impl Of Programming Languages 2015 • Software Design. 2015
Selected Undergraduate Courses (Bachelor Courses)	C & C++ Language Micro. and Assembly Lang Visual Basic.NET Data Structure Algorithm Design Computer Networks Operating System Data Security System Analysis Computation Theory

References	
	Dr. Mahmood Niazi Associate Professor Department of Information and Computer science E-mail: mkniazi@kfupm.edu.sa College of Computer Science and Engineering King Fahd University of Petroleum and Minerals. Saudi Arabia **Dr. Sajjad Mahmood** Associate Professor, Department of Information and Computer science E-mail: smahmood@kfupm.edu.sa College of Computer Science and Engineering King Fahd University of Petroleum and Minerals. Saudi Arabia **Dr. Musab Ahmad Alturki** Associate Professor Department of Computer Engineering E-mail: musab@kfupm.edu.sa College of Computer Science and Engineering King Fahd University of Petroleum and Minerals. Saudi Arabia