DESIGN AND DEVELOPMENT OF A NOVEL METHODOLOGY FOR SECURITY THREAT ORIENTED REQUIREMENTS ENGINEERING

Thesis submitted in fulfillment of the requirements for the Degree of

DOCTOR OF PHILOSOPHY

in

INFORMATION TECHNOLOGY

by

MD TARIQUE JAMAL ANSARI

Supervised by

Dr. DHIRENDRA PANDEY
Department of Information Technology
Babasaheb Bhimrao Ambedkar University, Lucknow

Submitted to

BABASAHEB BHIMRAO AMBEDKAR UNIVERSITY LUCKNOW

JANUARY-2020

Dedicated to my beloved parents, brother and sister...

Declaration

I, **Md Tarique Jamal Ansari**, solemnly declare that this thesis of research on "**Design and Development of a Novel Methodology for Security Threat Oriented Requirements Engineering**" is my original work. The study has been conducted under the guidance of Dr. Dhirendra Pandey, at Department of Information Technology, Babasaheb Bhimrao Ambedkar University (A Central University), Lucknow. It is further declared that to the best of my knowledge and belief it has not been submitted earlier for the award of any degree. I also undertake that the thesis is essentially free from all kinds of plagiarism.

Dated:

Md Tarique Jamal Ansari
Researcher
Department of Information Technology
Babasaheb Bhimrao Ambedkar University
(A Central University)
Lucknow, Uttar Pradesh, India

Certificate

This is to certify that the thesis entitled ‘‘**Design and Development of a Novel Methodology for Security Threat Oriented Requirements Engineering**’’ submitted by **Md Tarique Jamal Ansari** is an original research work and has not been previously submitted in part or full for the award of any other degree or diploma to this or any other University.

This thesis submitted to Babasaheb Bhimrao Ambedkar University, Lucknow satisfies all the requirements as stipulated in the Doctor of Philosophy (Ph.D.) regulations-1999 as amended in 2013 and it is fit for submission and evaluation for the award of the degree of Doctor of Philosophy of the University.

Supervisor **Head of the Department**

Dated:

ACKNOWLEDGEMENT

Undertaking this Ph.D. in Information Technology has been a truly life-changing experience for me and it would not have been possible to do without the support, guidance and blessings that I received from The Supreme Power and many people.

First, I would like to thank God for the gift of life and for blessings all through my life that allowed me to get here.

I am extremely indebted to my supervisor **Dr. Dhirendra Pandey**, in addition to being the best supervisor one could hope for he made the whole experience so exciting and enjoyable throughout. I am extremely grateful for providing me with the valuable insights that allowed me to consider my work in different contexts. He not only showed belief in me and supported me when I started the long journey, but also provided me with constructive criticism that helped me to refine my work. His enthusiasm and involvement in my research have been instrumental in helping me stay motivated and excited about my dissertation all along. I will always be highly obliged to him for his wholehearted support and kindness extended to me during my entire course.

I would like to extend my sincere gratitude to the Dean and Head of Department **Prof. Raees Ahmad Khan** for his guidance, invaluable support and consultations during the course of the study. I will remain indebted to his valuable suggestions during the entire research and thesis work and providing thoughtful feedback to improve its content. I will always be gratified to him for his unconditional support and kind-heartedness extended to me during my entire course.

I would also like to thank to **Dr. Mamdouh Alenezi**, Dean Deanship for Educational Services & Associate Professor of College of Computer & Information Sciences, Prince Sultan University, Riyadh, Saudi Arabia for his cooperation and continuous support extended during my research work.

Special thanks to my mother **Mrs. Sairah Begum**, my father **Dr. Munir Ahmad Ansari**, my sister **Dr. Nusrat Jahan,** my brother **Er. Manauwar Ali Ansari,** my cousin **Dr. Nagma Ansari** and **Sahil Ansari** for motivation, unconditional love, attention, affection, understanding and teachings always given to me. I would also like

to say thanks to my best friend **Dr. Samiya Ansari**, for affection and happiness, for making my life lighter and more joyous. Non-academic support also has been apart in succeeding this work. Mentioning which this I can never skip to pen down my humble gratitude toward my uncle **Mr. Arshad Ali Ansari** without him I would not have been able to show enthusiasm and zeal to execute the task.

My colleagues have been a source of inspiration to me and I would like to convey my sincere thanks to **Dr. Shilpi Singh, Dr. Kanika, Dr. Prabhishek, Mr. Neeraj, Mr. Amal, Mr. Ajay** and **Mr. Asim** for their feedback, cooperation and of course friendship and support during the entire Doctorate. I thank you all from the core of my heart.

Finally, there are my friends **Mr. Naseem Khan, Mr. Anas, Mr. Adnan Ansari, Mr. Ranjeet Singh, Mr. Niwas Singh**. We were not only able to support each other by deliberating over our problems and findings, but were also happy to talk about things other than just our papers.

To all, despite not mentioned, who directly or indirectly have contributed to the accomplishment of this work. Thank you for all your effort extended to my personal and professional training.

Md Tarique Jamal Ansari

ABSTRACT

Nowadays, in the age of information technology, a number of organizations frequently process, manage and exchange their most crucial information using technology-intensive systems which are directly connected to the internet. Such systems are being progressively exposed through the use of information technology and its tools, such as web services that allow other web services, which are itself software systems. This process may be with or without human involvement, to access, manipulate and exploit this sensitive and confidential information. This enhanced transparency has made more accessible sensible business knowledge and the software systems that manage it.

With growing dependency on information technology, it is important to be aware of new and emerging security threats that deliberately target software systems. Such software projects fail at the cost of millions of dollars, with several jobs lost and, unfortunately, with the loss of life. One of the most frequent and severe cyber security attacks ever reported against enterprises in a wide range of industries have been seen in the last few years. While security experts anticipate for another record-breaking year of network infringements and information security risks, it is essential that organizations become knowledgeable of the new cyber threats in development and guarantee that their security measures are strong enough to compete.

Software security is daunting and vital, because so many essential activities are entirely software-dependent. This makes software a very valuable target for attackers whose intentions can be malicious, illegal, adversarial, and profitable. The practically guaranteed presence of vulnerabilities makes it so easy for adversaries to target software which can be manipulated to infringe one or more of the software's security measures or to force the system into extremely dangerous situation. Secure software can't be compelled to malfunction deliberately. When security requirements are not clearly specified, the resulting program may not be tested until implementation for success or failure. In software development process most of the time security requirements consideration often developed independently of other requirements engineering activities. As a result, specific security requirements are often ignored

and functional requirements are specified in blissful ignorance of safety aspects. Requirements engineers are generally well qualified in functional requirements but not in software security. Very few qualified engineers have learned only basic security architectural skills, such as password security, encryption, and decryption. They don't have thorough understanding of accurate security requirements engineering. It comes as no surprise that security requirements engineering is critical to the success of any major software development project.

Several researchers have proposed security requirements engineering techniques, tools, framework, methodology, and norms for eliciting security requirements during the early stages of the software development cycle. Engineering security requirement into the early stages of the development process is profitable, secure and also provides quality software product. Security requirement engineering should be systematic, repeatable and capable of eliciting complete, reliable, clear, simple and easy to analyzable by the other members of the software development process. Typically, security requirements engineering performs different engineering tasks independently from other functional requirements. Some critical security requirements are therefore consistently ignored, and the requirements engineer concentrated only on functional requirements while ignoring important software safety aspects. Numerous security requirement engineering frameworks, techniques, processes and methodologies have been proposed by different authors, but there is still a need to improve them.

The absence of effective and efficient security requirements engineering approach in today's software development process often produces software with exploitable weaknesses. These software weaknesses are known as vulnerability that makes a threat to enter into the software system to perform unauthorized actions within a software system. To exploit vulnerability, an attacker must have at least one applicable tool or technique that can connect to a system weakness. It may be due to inadequate design, config errors or improper and insecure coding strategies.

The elicitation of effective and efficient security requirements is an important and challenging task. It is an important task because there are many problems associated with the consideration of security issues during the software development phases that must be overcome. Generally, software security is not considered by the developers during the early phases of the software development life cycle. Many

security Requirements engineering approach normally does not comprise all significant stakeholders and does not use the well-organized techniques for stakeholder identification and prioritization. Most of the time the security requirements specification is incomplete, unclear, contradictory, not cohesive, disorganized, infeasible, outdated, unable to be validated, and not usable by their expected persons. All elicited security requirements should be well organized in a systematic manner otherwise the software system cannot be evaluated for accomplishment. Therefore, there is a need to develop an approach which is capable of eliciting more effective and efficient security requirements by considering all the issues which are neglected by the previous approaches.

The software system which is to be developed may have several stakeholders. Only some stakeholders have engaged in the production of software applications, but all stakeholders are aligned with the software system. Some stakeholders can assist in the asset identification process. They suggest assets of the system from their point of view. These assets may have vulnerabilities. The threat exploits these vulnerabilities to get access to the system. The risk associated with each identified threat is not the same. The attacker has always targeted the vulnerability with high risk. The requirement engineer categorizes and prioritizes the identified threats.

In order to design and develop an efficient security requirements engineering approach the researcher first determines the selection criteria which influence the software developers in selecting the effective security requirements engineering and priority ranking of these criteria by using the Analytic Hierarchy Process (AHP) model. The study was planned and conducted to identify the different criteria which are considered by the software developer during the selection of effective security requirements engineering approach. Further the priorities of selection criteria were determined by using AHP model. Effective security requirements engineering selection criteria were compared in pairs by the security experts.

Further based on the ranking given by different security experts, we design and develop a novel security requirements engineering approach. The main objective of this research is to design and develop a security requirements engineering methodology by considering available literature analysis and security expert's necessities and that should also be powerful in eliciting security requirements. The proposed methodology is capable of eliciting security requirement which is effective,

efficient, complete, clear, consistent, organized, feasible, up to date and easy to be validated. This methodology is especially suitable for any type of software project, web based applications that requires security from the beginning of the development process.

This research presents a novel Security Threat Oriented Requirement Engineering (STORE) methodology which is a ten step systematic process, which provides an effective, efficient and systematic way of eliciting and documenting security requirements for the software as well as web-based applications from the early phases of software development. In this methodology, security requirements are often discussed in the context of threats. Threat helps the security requirements engineer to calculate the risk associated with it and also represents the adversary's abilities. Stakeholder plays an important role in the STORE methodology. A software system which is to be developed can have several stakeholders, only a few stakeholders are associated with the security of software products. Those stakeholders, who have security concern of the software system, have knowledge about the related assets of the systems which are to be protected from the threats. In STORE methodology we identify and prioritize all such stakeholders based on their importance. It is important to consider every significant stakeholder from the beginning of software development. This proposed methodology considers security threats for identifying security requirements with the help of potential stakeholders. These stakeholders help the requirement engineer in asset identification of the software product.

We have also applied the proposed STORE methodology to the Enterprise Resource Planning (ERP) web-based application software. A college ERP system is capable of managing student records, department, faculties, library, and other information. It has all the information about the students, faculties, staff, library, departments and other confidential information which requires security. The proposed STORE methodology is also compared with the two existing security requirements engineering approaches i.e. MOSRE framework and SQUARE methodology and found that the proposed STORE methodology is better in terms of eliciting complete and well-organized security requirements.

Furthermore the researcher has also validated the proposed STORE methodology. The validation of STORE methodology was done by the security

expert evaluation. This form of validation requires one to construct a systematic evaluation process where security expert's plays a role of assessing the methodology based on some pre-defined criteria. It is necessary for each security expert to understand and experience the operational mechanism of the proposed methodology before participating in the systematic evaluation process. They have to experience the functionality of proposed methodology through the different example problem to show how the STORE methodology presented in this thesis help elicits effective and efficient security requirements.

ABBREVIATION

- CC: Common Criteria
- CEM: Common Evaluation Method
- CMM: Capability Maturity Model
- ERP: Enterprise Resource Planning
- ICT: Information and Communication Technology
- PoA: Point of Attack
- PoB: Point of Belief
- PoC: Point of Conjecture
- PoD: Point of Dependency
- SIREN: SImple REuse of software requirements
- SNS: Social Networking Sites
- SO: Security Objective
- SQA: Software Quality Assurance
- SQUARE: Security Quality Requirements Engineering
- SR: Security Requirement
- SRE: Security Requirements Engineering
- SREP: Security Requirements Engineering Process
- SSE: Systems Security Engineering
- STORE: Security Threat Oriented Requirements Engineering
- UML: Unified Modeling Language

TABLE OF CONTENTS

DECLARATION…….. (i)
CERTIFICATE…… (ii)
ACKNOWLEDGEMENT……... (iii)
ABSTRACT…….. (v)
ABBREVIATION……. (x)
LIST OF TABLES…… (xvi)
LIST OF FIGURES……. (xix)

CHAPTER 1 INTRODUCTION…….…….. (1-13)

1.1 Background…… **1**
1.2 Security Requirements……... 4
1.3 Software Requirements Engineering……. 6
1.4 Research Issues……..... 7
1.5 Research Problems…….... 8
1.6 Objectives of the Research……. 8
1.7 Methodology Followed……. 9
1.8 Expected Deliverables…….. 10
1.9 Limitations……..... 10
1.10 Thesis Outline……. 10

CHAPTER 2 LITERATURE REVIEW (14-36)

2.1 Background……. 14
2.2 Review Method…….. 15
2.3 Selected Literature for Review……... 15

2.4	Comparative Analysis…….....	27
2.5	Relevant Findings.....	35
2.6	Conclusion……..…….…….	36
CHAPTER 3	**CRITERIA DECISION MAKING FOR EFFECTIVE SECURITY REQUIREMENTS ENGINEERING**	**(37-54)**
3.1	Background…….	37
3.2	Analytic Hierarchy Process (AHP)..……..	38
	3.2.1. Determine the relative weights of the decision criteria……..	39
3.3	Implementation and Ranking…….....	44
3.4	Results……	45
3.5	Discussion……..	53
3.6	Conclusion……..…….……....	54
CHAPTER 4	**PROPOSED SECURITY REQUIREMENTS ENGINEERING METHODOLOGY…….**	**(55-69)**
4.1	Background…….	55
4.2	STORE Methodology…….	57
4.3	Details of STORE Methodology Steps…	60
	4.3.1 Identify System Goals…….....	61

4.3.2 Identify & Prioritize Stakeholders……. 62

4.3.3 Agreed upon Goals…… 63

4.3.4 Asset Identification…….. 63

4.3.5 Security Attack Analysis…….. 64

(A) Point of Attack (PoA)…….. 64

(B) Point of Belief (PoB)…… 64

(C) Point of Conjecture (PoC)…… 64

(D) Point of Dependency (PoD)……. 65

4.3.6 Threat Identification and Categorization… 65

4.3.7 Risk Evaluation and Prioritization…… 67

4.3.8 Security Requirements Elicitation…… 68

4.3.9 Security Requirements Validation…… 68

4.3.10 Security Requirements Specification Document…… 68

4.4 Conclusion…… 69

CHAPTER 5 CASE STUDY EVALUATION (70-87)

5.1 Background…… 70

5.2 Case Study of ERP System……. 71

5.3 Execution of the Case Study…….. 72

5.3.1 Identify System Goals……..... 72

5.3.2 Identify & Prioritize Stakeholders……. 73

5.3.3 Agreed upon Goals…… 73

5.3.4 Asset Identification…… 74

5.3.5 Security Attack Analysis……. 75

(A) Point of Attack (PoA)……... 75

(B) Point of Belief (PoB)…… 77

(C) Point of Conjecture (PoC)……. 77

(D) Point of Dependency (PoD)…….. 78

5.3.6 Threat Identification and Categorization….. 79

5.3.7 Risk Evaluation and Prioritization…… 81

5.3.8 Security Requirements Elicitation……. 82

5.3.9 Security Requirements Validation……. 83

5.3.10 Security Requirements Specification Document…… 83

5.4 Results and Discussion……... 83

5.4.1 Comparison with the SQUARE Methodology… 84

5.4.2 Comparison with the MOSRE Framework…… 85

5.5 Conclusion…….. 86

CHAPTER 6 VALIDATION OF PROPOSED METHODOLOGY (88-107)

6.1 Background…….. 88

6.2 Evaluation and Validation of STORE Methodology……. 89

6.2.1. Review during Design Time…… 89

6.2.2. Systematic Evaluation Process……. 89

6.3 Systematic Evaluation Process..... 90

6.3.1. Design of Experiment..... 91

6.3.2. Pre-Tryout..... 92

6.3.3. Review and Revision..... 98

6.3.4. Tryout..... 99

6.3.5. Finalization..... 105

6.4 Results and Discussion……... 106

6.5 Conclusion……..……..... 107

CHAPTER 7 CONCLUSION AND FUTURE WORK (108-116)

7.1 Background……..... 108

7.2 Major Research Contributions…… 110

7.3 Significance of the Work……..……. 113

7.4 Future Directions……..…….. 114

7.5 Research Findings……..……. 115

7.6 Conclusion…… 116

REFERENCES …….. (117-125)

APPENDIX A QUESTIONNAIRE SET A……. (126)

APPENDIX B QUESTIONNAIRE SET B..... (128)

APPENDIX C QUESTIONNAIRE SET C..... (130)

APPENDIX D PLAGIARISM REPORT …….. (131)

LIST OF THE TABLES

Table Number	Name of the Table	Page No
Table 2.1	Selected security requirements engineering literature for comparative review……..	16
Table 2.2	Comparative Study of security requirements engineering approach	27
Table 3.1	Saaty's 1–9 scale……..	41
Table 3.2	Random index for pairwise comparison matrices……	44
Table 3.3	Criteria weight matrix by Participant 1……	46
Table 3.4	Criteria weight matrix by Participant 2……	46
Table 3.5	Criteria weight matrix by Participant 3……	46
Table 3.6	Criteria weight matrix by Participant 4…….	47
Table 3.7	Criteria weight matrix by Participant 5…….....	47
Table 3.8	Criteria weight matrix by Participant 6…….....	48
Table 3.9	Criteria weight matrix by Participant 7……..	48
Table 3.10	Criteria weight matrix by Participant 8……..	48
Table 3.11	Criteria weight matrix by Participant 9…….....	49
Table 3.12	Criteria weight matrix by Participant 10…….	49
Table 3.13	Combined weighted geometric mean of all participants…….	50
Table 3.14	Comparison Matrix for Effective Security Requirements Engineering Criteria……	51
Table 3.15	Rank wise criteria for effective Security Requirements Engineering approach……...	51

Table 4.1	Steps in the proposed STORE Methodology……..	60
Table 4.2	Microsoft's STRIDE Threat classification system…….....	66
Table 5.1	Identified system goals for college ERP system……..	72
Table 5.2	List of possible stakeholders with their significance for college ERP system……...	73
Table 5.3	Identified assets of the college ERP system…….	74
Table 5.4	List of identified Point of Attack (PoA)……...	75
Table 5.5	List of Identified Point of Belief (PoB)……	77
Table 5.6	List of identified Point of Conjecture (PoC)……	78
Table 5.7	List of identified Point of Dependency (PoD)……..	78
Table 5.8	Identified threats for the college ERP system……..	79
Table 5.9	Prioritize threats to their DREAD risk value……	81
Table 5.10	Elicited security requirements for each threat……..	82
Table 5.11	STORE methodology comparison results with the SQUARE methodology……..	84
Table 5.12	STORE methodology comparison results with the MOSRE framework……..	86
Table 6.1	Ease of Use evaluation result of STORE Methodology in Pre-Tryout phase……..	93
Table 6.2	Ease of Learning evaluation result of STORE Methodology in Pre-Tryout phase……...	94
Table 6.3	User satisfaction evaluation result of STORE Methodology in Pre-Tryout phase……..	96
Table 6.4	Descriptive feedback result from the security experts……..	97

Table 6.5	Ease of Use evaluation result of STORE Methodology in Tryout phase……	100
Table 6.6	Ease of Learning evaluation result of STORE Methodology in Tryout phase……	102
Table 6.7	User Satisfaction evaluation result of STORE Methodology in Tryout phase……	104

LIST OF THE FIGURES

Figure Number	Name of the Figure	Page No
Figure 1.1	Different type of Security Requirements ……	5
Figure 1.2	Summary of the thesis chapters ……..	11
Figure 3.1	Flowchart of the Analytic Hierarchy Process ……...	42
Figure 3.2	The AHP model combining the purpose and criteria ..	45
Figure 3.3	Comparative result for each criterion of security requirements engineering……...	52
Figure 3.4	Pie chart representation of the result obtained ……..	53
Figure 4.1	Security requirement engineering concept model for STORE methodology……..	58
Figure 4.2	Activity diagram of STORE Methodology …….	59
Figure 4.3	Four Points of Security Attack Analysis……..	65
Figure 6.1	The systematic evaluation process diagram for STORE methodology…….	91
Figure 6.2	Bar graph of Ease of Use feedback in Pre-tryout phase	94
Figure 6.3	Bar graph of Ease of Learning feedback in Pre-tryout phase……..	95
Figure 6.4	Bar graph of User satisfaction feedback in Pre-tryout phase……..	97
Figure 6.5	Bar graph of Ease of Use feedback in Tryout phase…….	102
Figure 6.6	Bar graph of Ease of learning Feedback in Tryout phase...	103
Figure 6.7	Bar graph of Satisfaction Feedback in Tryout phase……..	105

Chapter 1 Introduction

"Securing a computer system has traditionally been a battle of wits: the penetrator tries to find the holes, and the designer tries to close them."

- ***Gosser***

1.1 Background

Information Technology (IT) based services and applications provide the use of computers and network resources to store, access, transfer, and update the data in several sectors. Many digital services are opening up to the great world of Internet, which now connects not only computers but also a number of smart and miniaturized objects of our daily life such as embedded electronic chips, sensors, actuators, and smart software, for providing persistent information access [1]. In recent years, several organizations are becoming heavily dependent on information processing systems to get more benefit quickly. This dependency has produced a need for protecting software systems from threats. With increasing use and dependency on information technology for business and many other sectors, the volume of cyber-attack also rises exponentially. Therefore, software security has become an essential issue [2].

Software security vulnerabilities and flaws are outcomes of poorly built software that can lead to easy exploitation by the hackers. Inappropriate security requirements engineering is one of the reasons for developing bad quality software products. This problem can be reduced by taking security concern into account from the early phases of the software development process. The development process needs to shape their security properties by adding security methodology and implementing them correctly,

by observing security principles, and by avoiding defects. Security requirements guide this design, implementation, and verification work [3]. Gartner, Inc. forecasts worldwide enterprise security spending to total $96.3 billion in 2018, an increase of 8% from 2017. This report shows that the digital industries are spending huge money on maintaining security consequences of regulations, changing and attracting customer mindset, awareness of emerging threats and the advancement to a digital industry planning [4].
The White House Cybersecurity coordinator Rob Joyce has outlined that receiving and maintaining the essential cyber capabilities to protect the nation creates a tension between the government's need to maintain the resources to follow attackers in the digital world through the use of digital exploits, and its obligation to share its knowledge of flaws in software and hardware with responsible parties who can ensure digital infrastructure is upgraded and made stronger in the face of growing cyber threats [5]. Amoroso argue that the cybersecurity community, in particular, has been vocal about finding ways to improve the software, because most vulnerability involve exploitable weaknesses introduced through badly written code [6].

Several researchers have proposed security requirements engineering techniques, tools, framework, methodology, and norms for eliciting security requirements during the early stages of the software development cycle. Engineering security requirement into the early stages of the development process is profitable, secure and also provides quality software product. Security requirement engineering should be systematic, repeatable and capable of eliciting complete, reliable, clear, simple and easy to analyzable by the other members of the software development process [7]. Security requirements are often developed independently of other requirements engineering activities. Generally, security requirement engineering carried out separately of other functional requirements engineering activities. Therefore, many vital security requirements are repeatedly ignored, and the requirement engineer only focused on functional requirements by ignoring important security aspects [8]. Several security requirement engineering frameworks [9-19], techniques [20-26], processes [27-35] and methodologies [36-40] have been proposed by different authors.

Normally, Requirements engineers are well skilled in functional requirements, but not in software security. Very few requirement engineers that have been skilled have only been known the basic security architectural knowledge like password security, encryption, and decryption. They don't have deep knowledge of authentic security

requirements engineering. The software system which is to be developed may have several stakeholders. Only some stakeholders participated in the development of software products, but all stakeholders are associated with the software system. Some stakeholders are capable to help in the identification process of the asset. They suggest assets of the system from their point of view. These assets may have vulnerabilities. The threat exploits these vulnerabilities to get access to the system. The risk associated with each identified threat is not the same. The attacker has always targeted the vulnerability with high risk. The requirement engineer categorizes and prioritizes the identified threats. After that the identified threats are mitigated by using some security mechanism which satisfies the security requirements of the system. The elicitation of effective and efficient security requirements is an important and challenging task. It is an important task because there are many problems associated with the consideration of security issues during the software development phases that must be overcome. Security requirement engineering is challenging because there are requirements that such a security-oriented approach must satisfy. Generally, software security is not considered by the developers during the early phases of the software development life cycle [18].

Most security Requirements engineering approach normally does not comprise all significant stakeholders and does not use the well-organized techniques for stakeholder identification and prioritization [41]. Normally the security requirements specification is incomplete, confusing, conflicting, not cohesive, disorganized, infeasible, obsolete, unable to be validated, and not usable by their anticipated persons [8]. Several authors have been proposed methodology for analyzing security requirements engineering. All effective and efficient security requirements should be well organized in a systematic manner otherwise the software system cannot be evaluated for accomplishment. Therefore, there is a need to develop a framework which is capable of eliciting more effective and efficient security requirements by considering all the issues which are neglected by the previous approaches.

The main objective of this research is to design and develop a novel security threat oriented security requirements engineering methodology that is capable of eliciting security requirement which is effective, efficient, complete, clear, consistent, organized, feasible, up to date and easy to be validated. The proposed methodology is especially suitable for any type of security-critical software product that requires security from the beginning of the software development process. This thesis presents a novel security

threat oriented requirement engineering (STORE) methodology which is a ten step systematic process. The proposed STORE methodology is capable of eliciting accurate, complete and well-organized security requirements for the secure and quality software product development.

1.2 Security Requirements

According to McGraw, the principal and serious aspect of the computer security issue is a software problem [42]. Security requirements are a type of non-functional requirements [45]. The software security engineering as a discipline which considers capturing and modelling for security, design for security, adopting best practices, testing for security, managing, and educating software security to all stakeholders [43]. Security requirements have usually been considered to be “non-functional” or “quality” requirements [44-48] and many others. Security requirement engineering is a vital phase in the software development process which ensures secure software development from the beginning. In the last few years, the security has become a crucial issue in the development and deployment of secure software products. This crucial issue has been considered by several security professionals and researchers for publishing research papers on integrating security considerations into the software development process. Yet, only a small amount of papers illustrates complex case studies [49]. To solve the security problem with software and web-based applications, an increasing number of literature publication, symposium tracks, and workshops in recent years call attention to the increasing attention of researchers and authors in designing and developing security requirements engineering frameworks, techniques, process, and methodologies.

Security requirements are mostly concerned with checking for problems and taking actions if these problems occur. By contrast, many types of security requirements cover the different threats faced by a system. Firesmith in [55] identified 10 types of security requirements that may be included in a system specification:

1. Identification requirements specify whether or not a system should identify its users before interacting with them.
2. Authentication requirements specify how users are identified.
3. Authorization requirements specify the privileges and access permissions of identified users.

4. Immunity requirements specify how a system should protect itself against viruses, worms, and similar threats.
5. Integrity requirements specify how data corruption can be avoided.
6. Intrusion detection requirements specify what mechanisms should be used to detect attacks on the system.
7. Nonrepudiation requirements specify that a party in a transaction cannot deny its involvement in that transaction.
8. Privacy requirements specify how data privacy is to be maintained.
9. Security auditing requirements specify how system use can be audited and checked.
10. System maintenance security requirements specify how an application can prevent authorized changes from accidentally defeating its security mechanisms.

Figure 1.1 Different type of Security Requirements

Rushby [56] suggests that security requirements are hard to write because certain security properties do not match behavioural properties that can be expressed using formal methods, and also because security requirements are counterfactual.

1.3 Security Requirements Engineering

Security requirement engineering relates to non-functional requirements of the software system which is a very important task to achieve for a quality software product. Improving the quality of security requirements is also important. But it is a difficult objective to achieve. To understand the reason, one should first define what requirements engineering is really about [50]. Security requirements of computer-based systems mostly concern confidentiality, integrity, and availability features [51]. Security is considered as a balance between confidentiality, integrity, and availability [52].

According to Firesmith in [53] security requirement enhanced the quality of the software product by adding a necessary quantity of security regarding conditions related to a system and a low level of an associated quality measure that is essential to convene one or extra security policies. Engineering software security is an analysis of identifying security issues for a software product in a systematic approach. This is a very important process taken into consideration at an early stage of the software development process for the achievement of the secure software product. Security requirements are necessary and it takes place when the stakeholders found that some objects in the context of the software system are precious, they might be tangible or intangible, and they have some value that the stakeholders desire to protect [54].

There are many security requirements engineering methods, frameworks, methodologies, techniques, best practice guidelines, and tools [43]. It is important to get the security requirements accurate, concise, and unambiguous. Security requirements engineering is a collaborative effort that includes many stakeholders such as business analysts, software requirements engineer, software architect, and test managers. Security requirements engineering process should deliver a clear and complete set of security specific needs and expected performance of a system. The key objective is to safeguard systems assets (data and files) and unauthorised access to the system from intentional attacks to the application software systems and other forms of internet based security attacks such as spam, denial of service, identity theft, viruses, and many other

forms of intentional attacks that emerges every day. Security needs to be measured as highly specific set of requirements for every functional requirements that has been identified, and has to be applied throughout the life cycle so that we can achieve build in security (BSI). In addition current RE methods have considered mostly what the system must do, but what the system must not do. This is the key issue that will be considered when selecting RE methods for software security. Moreover, in Software security RE methods, there are more stakeholders than traditional RE methods have considered such as social engineers, security specialist, business process modeling experts, service computing specialists, and users. Often attackers look for defects in the system, not the system features and functionalities.

The researcher define the Security Requirements Engineering as

> ***"Security requirements engineering is a process of generating prerequisite non-functional requirements that stipulate a compulsory amount of system quality to prevent the adversary's attack. It should be considered early during the software development process to deliver a trustworthy software product."***

1.4 Research Issues

In this case, the questions include the investigation of the main challenges and problems that can be found with respect to the topic of security requirements engineering, along with another question whose objective is to discover the main security dimensions on which researchers are focusing their efforts. Finally, the researcher wished to discover what different techniques, methodologies or models have already been developed in order to deal with these problems. Thus, reviewing the literature about security requirements engineering and associated security, the following research questions arise which needs to be addressed:

- What are the major challenges with respect to secure and trustworthy software product?
- What is the current status of security requirements engineering research?

- Does the stakeholder's involvement in security requirements engineering process have been addressed previously?
- Is there any security requirements engineering approach available which provide effective security attack analysis?

1.5 Research Problems

From the foregoing discussion, it is pertinent that software security is becoming a challenging issue in the current information technology era where insecurity is everywhere. There is an extremely need of the trustworthy and quality software development which should be capable to prevent many types of modern attacks. A major problem with user's personal data is that his sensitive data is shared with unauthorised party and he is not even aware about it. For adequate secure and quality software product, there must be an effective and efficient approach that can prevent software from unauthorised access and different cyber-attacks. Keeping this in mind, the researcher has formulated a problem as under in order to improve the quality of software products early during the software development process by eliciting complete and well-organized security requirements.

"Design and Development of a Novel Methodology for Security Threat Oriented Requirements Engineering"

1.6 Objectives of the Research

In order to achieve the most general goal to improve the process of secure, quality and trustworthy software product development, the following objectives have been set forth:

- To review and critically examine the existing security requirements engineering approaches and to identify key issues that needs to be addressed in the real-world

implementation of security requirements engineering approach in software organizations.

- To study the different criteria which are necessary for an effective security requirements engineering approach, and rank those criteria based on the interests of different security experts.
- To examine the impact of threats and stakeholders involvement on security requirements engineering process.
- To appreciate the involvement, importance and significance of different stakeholders in the security requirements engineering process.
- To design and develop a novel security requirements engineering methodology.
- To design an algorithm to elicit effective and well-organized security requirements.
- To conduct a case study implementation of the proposed methodology.
- To evaluate the performance of the proposed security requirements engineering methodology.
- To compare the performance of the proposed security requirements engineering approach with the other existing approaches in the area.
- To validate the proposed system.

1.7 Methodology Followed

The basic methodology is tantamount to a list of things that we might try in order to reach out the ultimate goal.

- Review of the available literature
- Conceptualization of the approaches
- Expert-Review and Revision of the proposed approaches
- Implementation of the proposed approach
- Experimentation
- Assessment of Effectiveness
- Documentation and Finalization

1.8 Expected Deliverables

The following are the expected outcomes of the research:

- A prioritized ranking list of different criteria which are needed for an effective security requirements engineering approach.
- Design and development of a novel methodology for effective and efficient elicitation of security requirements.
- An experimental study showing the practicality and benefits of the proposed work.
- A comparative study to prove that the proposed methodology is better one.
- A study carrying out the validation of the proposed security requirements engineering methodology.

1.9 Limitations

In order to keep the research precise and within the time boundary, the thesis has few limitations. These are as follows:

- The proposed work has been tested on the small software project only.
- Implementation of the proposed work has not been done in real scenario.
- The researcher has not evaluated this approach through global security experts.

1.10 Thesis Outline

A thesis of the research has been prepared to present a detailed study on the research problem and to answer the research questions. The thesis has six chapters apart from annexure, references and other components. The graphical representation of thesis chapter summary presented in Figure 1.2

Figure 1.2 Summary of the thesis chapters

Chapter- 2: Literature Review

This chapter discusses the various available approaches, techniques, frameworks, methodologies, standards for security requirements engineering. It presents the literature review, basic terminology and several existing security requirements engineering techniques. It also presents the general overview of several SRE approaches, and consequently introduces the several issues with security requirements engineering.

Chapter- 3: Criteria Decision Making for Effective Security Requirements Engineering

In this chapter the objective is to determine the selection criteria for the software developers in selecting the effective security requirements engineering and priority ranking of these criteria by using the AHP model. The study was planned and conducted to identify the different criteria which are considered by the software developer during the selection of effective security requirements engineering approach. Further the priorities of selection criteria were determined by using AHP model. SRE selection

criteria were compared in pairs. 10 security experts were asked to fill the evaluation form.

Chapter- 4: Proposed Security Requirements Engineering Methodology

In this chapter, the researcher has proposed a novel security requirements engineering approach that is known as Security Threat Oriented Requirements Engineering (STORE) methodology. The STORE methodology is a ten-step sequential process which provides an effective, efficient and systematic way of eliciting and documenting security requirements for the software as well as web-based applications from the early phases of software development. In this methodology, security requirements are often discussed in the context of threats. This approach is suitable for eliciting complete and well-organized security requirements for security-critical software products.

Chapter- 5: Case Study Evaluation

In this chapter 4, the researcher proposed STORE methodology applied to the Enterprise resource planning (ERP) web-based application software. This college ERP system is capable of managing student records, department, faculties, library, and other information. The college ERP system has all the information about the students, faculties, staff, library, departments and other confidential information which requires security. The proposed STORE methodology is compared with the two existing security requirements engineering approaches i.e. MOSRE framework and SQUARE methodology and found that the proposed STORE methodology is better in terms of eliciting complete and well-organized security requirements.

Chapter- 6: Validation of Proposed Methodology

In this chapter, concept of validation has been discussed and methodology for validation has been designed. During validation, feedback suggestions have been formulated and have been tested on the basis of statistical analysis. Experimental results are presented graphically. The systematic evaluation process of the STORE methodology was conducted in a systematic manner. In this chapter, concept of validation has been discussed and methodology for validation has been designed.

Chapter- 7: Conclusion and Future Work

This chapter describes the findings of this research. It presents an overview of the research along with its major findings. In addition, it demonstrates the significant contribution of this research in reference to secure and trustworthy software development. It also discusses probable limitations of the research and proposes directions for future research.

Chapter 2 Literature Review

"Risk of security cannot be removed completely when spreading the data over the Internet"

- ***Mavridis***

2.1 Background

In a software development process, customers, users, and vendors of a software product must agree on the requirements before the software can be built. In recent days, recognition of security at requirements engineering time is the new attention of the present world. Security requirements engineering is an emerging field of software engineering study, with the understanding that security needs to be evaluated early. Safety is the main issue for ensuring full quality applications so it must be done consistently through the different stages of the specifications engineering process. Since security is non-functional requirement many times it is ignored in the requirements phase of Software Development Life Cycle. However, at the very first stage of software development process, it is easy to reduce the cost and time of software development to identify user security requirements. The main objective is to present the user security requirements in combination with user functional requirements collected in the Software Development Life Cycle (SDLC) from the requirement phase. Attempting to develop secure software will be assured from the very beginning, if we can define user security requirements and address those security requirements in the software development requirements process.

This chapter discusses the several existing security requirements engineerinh

2.2 Review Method

In this chapter, the literature review method for security requirements engineering is based on the structural strategy. This section discusses the strategy for search, the sources, the study selection and the selection execution. The systematic review was used of the security requirements engineering literature, using a structure strategy in English language from 1996 to 2018. Search strategies were performed the following databases: Scopus, Web of Science, DOAJ, EBSCO, OCLC, Google Scholar, Yahoo, Eric, Google books, Proquest, JSTOR. ACM digital library, IEEE digital library, Science Direct, SREIS symposium, ESORICS symposium, REFSQ conference, DEXA conference, WOSIS workshop, ICCSA conference, Requirements Engineering Journal, IEEE International Requirements Engineering Conference, ICSE conference, COMPSAC conference. Major search keywords consist of: Non-functional requirements, Security requirements, Security requirements engineering, requirements engineering, security engineering, secure development, secure IS development, secure software development. Research publications selected based on the inclusion /exclusion criteria. Inclusion criteria include criteria pertaining to publication characteristics, such as full text published, peer reviewed publication, English language publication. Exclusion criteria include duplicate publications, asymmetric information and research with a focus on information and communication technology removed.

2.3 Selected Literature for Review

After applying inclusion/exclusion criteria on available literature on security requirement engineering approaches, 28 research papers are accepted for comparative analysis. Information extracted from these research papers must contain the proposed approach, procedures, methods, steps, strategies or any kind of creativity to elicit security requirements in an effective and efficient way during the early phases of software development. The information forms defined for this systematic review will contain the study identification, the study methodology, the study results, the study problems and our general impressions and abstractions. Regarding the study methodology, we shall focus on the modeling of the security requirements, on the modelling / development standard and on the security standards, along with the

technical criteria defined within the analytical framework explained in the following section. The following sub-section provides a brief outline of each of the selected studies/initiatives shown in the previous section, according to the extracted information obtained through the information forms.

The database search for the literature of security requirements engineering produced 5266 papers. However, after analyzing the literature search result on the basis of paper title, 3035 duplicated studies excluded. After removing duplicate papers, 1685 papers are screened by title/abstract and 546 papers are screened on the basis of full text. Again, after reviewing abstract and full text literatures, 1670 papers are excluded by title/abstract and 533 papers are excluded on the basis of full text. At last, in the literature inclusion/exclusion process, total 29 research papers met the inclusion criteria. The accepted literature data for comparative analysis is presented in the following Table 2.1.

Table 2.1 Selected security requirements engineering literature for comparative review

S. No.	Author(s)	Title	Type	Year	Citation
1	AI Anton	Goal-based requirements analysis	Conference	1996	754
2	McDermott & Fox	Using abuse case models for security requirements analysis	Conference	1999	527
3	Yu & Liu	Modelling trust for system design using the i* strategic actors framework	Conference	2001	157
4	Lodderstedt, Basin, & Doser	SecureUML: A UML-based modeling language for model-driven security	Conference	2002	912
5	Jürjens	UMLsec: Extending UML for secure systems development	Conference	2002	809

6	Toval, Nicolás, Moros & García	Requirements reuse for improving information systems security: a practitioner's approach	Journal	2002	144
7	den Braber, Dimitrakos, Gran, Lund, Stolen & Aagedal	The CORAS methodology: model-based risk assessment using UML and UP	Book Chapter	2003	55
8	Firesmith	Security use cases	Journal	2003	263
9	Lin, Nuseibeh, Ince, & Jackson	Using abuse frames to bound the scope of security problems	Conference	2004	88
10	Zuccato	Holistic security requirement engineering for electronic commerce	Journal	2004	50
11	Mead and Stehney	Security quality requirements engineering (SQUARE) methodology	Technical Report	2005	354
12	Sindre and Opdahl	Eliciting security requirements with misuse cases	Journal	2005	1172
13	Mayer, Rifaut & Dubois	Towards a risk-based security requirements engineering framework	Workshop	2005	70
14	Myagmar, Lee & Yurcik	Threat modeling as a basis for security requirements	Symposium	2005	272

15	Peeters	Agile security requirements engineering	Symposium	2005	54
16	Viega	Building security requirements with CLASP	Notes	2005	59
17	Asnar & Giorgini	Modelling risk and identifying countermeasure in organizations	Workshop	2006	89
18	Mellado, Fernández-Medina & Piattini	Applying a security requirements engineering process	Symposium	2006	64
19	Tsoumas and Gritzalis	Towards an ontology-based security management	Conference	2006	142
20	Gürses & Santen	Contextualizing Security Goals: A Method for Multilateral Security Requirements Elicitation	Journal	2006	19
21	Mouratidis & Giorgini	Secure tropos: a security-oriented extension of the tropos methodology	Journal	2007	348
22	Lamsweerde	Engineering requirements for system reliability and security	Book Chapter	2007	44
23	Hatebur, Heisel & Schmidt	A security engineering process based on patterns	Workshop	2007	41
24	Hussein and Zulkernine	Intrusion detection aware component-based systems: A specification-based	Journal	2007	34

		framework			
25	Haley, Laney, Moffett & Nuseibeh	Security requirements engineering: A framework for representation and analysis	Journal	2008	421
26	Salini & Kanmani	Model oriented security requirements engineering (MOSRE) framework for web applications	Conference	2013	14
27	Paja, Dalpiaz & Giorgini	Modelling and reasoning about security requirements in socio-technical systems	Journal	2015	30
28	Riaz, Stallings, Singh, Slankas & Williams	DIGS – A Framework for Discovering Goals for Security Requirements Engineering	Symposium	2016	14
29	Ansari et. al.	STORE: Security Threat Oriented Requirements Engineering Methodology	Journal	2018	6

2.3.1 GBRAM

Annie I. Anton [57] proposed goal based requirements analysis. She has discussed goal from the viewpoint of goal analysis and goal evolution. She developed and reviewed his experiences in applying our method to a relatively large example. She also validated some of the problems that experts face when using a goal-based approach to identify the requirements for a system.

2.3.2 Abuse Cases

McDermott and Fox [58] proposed a new technique of capturing and analyzing of security requirements in an easy way with the help of object oriented modeling

technique. They used use cases to model abuse cases. An abuse case model is easily understood by the users, customers and developers who understand either use case models or UML.

2.3.3 Secure i*

Yu and Liu [59] have developed the i*framework to support requirement analysis and high-level design in an agent-oriented system development paradigm. This framework models intentional dependency relationships between different strategic actors and their rationales. These actors depend on each other for goals to be accomplished, tasks to be achieved, and resources to be well-appointed.

2.3.4 SecureUML

Lodderstedt et. al. [60] presented a modeling language for the model-driven development of secure and distributed software system founded on the Unified Modeling Language (UML). This method is constructed on role-based access control with additional support for specifying authorization constraints. They presented how UML can be used to identify information related to access control in the overall design of an application and how this information can be used to automatically generate complete access control infrastructures. This approach can be used to improve efficiency during the development of secure distributed systems and the quality of the resulting systems.

2.3.5 UMLsec

Jürjens [20] intended to aid the problematic task of developing security-critical systems in an approach based on the notation of the Unified Modeling Language (UML). They presented the extension of UML known as UMLsec defined in form of a UML profile using the standard UML extension mechanisms. The UMLsec permits fast security relevant information inside the diagrams in a system specification. The related constraints give standards to assess the security aspects of a system design, by mentioning to a formal semantics of a basic fragment of UML.

2.3.6 SIREN

Toval et. al. [27] presented an applied technique to elicit and specify the system and software requirements with a source comprising reusable requirements, a spiral process model and a set of requirements templates. This technique is absorbed on the security of information systems and, thus, the reusable requirements repository contains all the requirements taken from MAGERIT, the Spanish public administration risk analysis and management method, which conforms to ISO 15408, Common Criteria Framework. Any information system together with these security requirements must consequently pass a risk analysis and management study performed with MAGERIT.

2.3.7 CORAS Methodology

Den Braber el. al. [61] introduced the CORAS methodology. This methodology combined Unified Modeling Language (UML) and Unified Process (UP) together to assist the model-based risk assessment on security-critical systems. In the CORAS methodology, an outdated risk management process is combined with UP, which is a well-accepted system development process. The CORAS methodology attempts to express how UML can contribute to better understanding, documentation, and communicating during the different phases of the risk management process. The CORAS methodology addresses both systems under development and systems already in use.

2.3.8 Security Use Cases

Use cases are mostly used modeling method for engineering functional requirements but they are often misrepresented when engineering non-functional requirements like security requirements because requirements engineers unreasonably specify security architectural mechanisms instead of security requirements. Misuse cases are extremely effective technique of investigating security threats but are unsuitable for the analysis and specification of security requirements. Firesmith [62] presented Security use cases which used to specify requirements that the application shall successfully protect itself from its relevant security threats. He offers some steps which allow security requirements to be defined from reusable templates.

2.3.9 Abuse Frames

Lin et. al. [63] have revised a proven object-oriented modeling technique, use cases, to identify and analyze security requirements in an easy way. They call this revision an abuse case model. From a user viewpoint the relationship of abuse cases to other security engineering work products is relatively easy.

2.3.10 Holistic SRE

Zuccato [29] has proposed an approach named ‘‘holistic security requirement engineering’’. This approach is intended to identify security requirements according to system- theoretic considerations. This shows that security requirements can be defined with the help of investigations in the business environment, workshops with stakeholders and risk analysis. This multidimensional approach leads to a holistic understanding of the requirements that fit into the system development life cycles.

2.3.11 SQUARE Methodology

Mead and Stehney [38] have presented the Security Quality Requirements Engineering (SQUARE) Methodology for identifying and prioritizing security requirements for software development projects. The SQUARE methodology developed under NSS program, is a nine steps process, The NSS Program continues to develop SQUARE, which has proven effective in helping organizations understand their security posture and produce products with supportable security requirements.

2.3.12 Misuse Cases

Sindre and Opdahl [24] presented a systematic method to identifying security requirements based on use cases, with importance on description and method procedures. This method extends traditional use cases to also cover misuse, and is potentially beneficial for numerous other types of additional-functional requirements beyond security.

2.3.13 ISSRM

Mayer et. al. [64] presented, that using and adapting an appropriate set of existing tools and techniques of risk analysis methods, improves the effectiveness of an iterative security engineering method starting at the earliest stage of IS development.

2.3.14 Threat based SRE

Myagmar et. al. [30] investigated how threat modeling can be used as foundations for the specification of security requirements. They observed the variances between modeling software products and complex systems, and outline their approach for identifying threats of networked systems. They also presented three case studies of threat modeling: Software-Defined Radio, a network traffic monitoring tool, and a cluster security monitoring tool.

2.3.15 Abuser Stories

Agile procedures have been believed inappropriate for security sensitive software development as the inflexibilities of assurance are realized to conflict with the lightweight and informal nature of agile processes. Though, such deceptively conflicting demands may be acquiescent by presenting the new conception of abuser stories in the requirements domain. Peeters [22] has extended the agile practices to deal with security in an informal, communicative and assurance driven spirit. These extend the well-established idea of user stories to accomplish security requirements traceability and thus open the door to effective security assurance, exactly because of their informal and lightweight nature.

2.3.16 CLASP

Viega [11] presented how to develop security requirements in a structured manner that is encouraging to iterative modification and, if followed properly, metrics for evaluation. He has provided a framework that is an understandable development over traditional methods that do not consider security at all. He also delivered an example using a simple three-tiered architecture. The methodology he documented is a subset of CLASP, a set of process pieces for application security.

2.3.17 Tropos Goal-Risk

The Tropos framework has been proved effective in modeling strategic interests of the stakeholders at organizational level. Asnar & Giorgini [65] have introduced the extended Tropos goal model to analyses risk at organization level and they also

demonstrated a number of different techniques to help the analyst in identifying and enumerating relevant countermeasures for risk mitigation.

2.3.18 SREP

Mellado et. al. [66] presented SREP (Security Requirements Engineering Process), which is a standard-centred process and a reuse-based approach which deals with the security requirements at the earlier stages of software development in a systematic and intuitive way by providing a security resources repository and by integrating the Common Criteria into the software development lifecycle.

2.3.19 Security Ontology

Tsoumas and Gritzalis [12] have presented a security management framework of an arbitrary information system (IS) which builds upon knowledge-based resources, such as security ontology (SO) providing reusable security knowledge interoperability, aggregation and reasoning exploiting security knowledge from diverse sources; in addition, the separation of security requirements from their technical implementations facilitates the security management. They also provided a feasible framework, which links the high-level policy statements and deployable security controls and facilitates the security expert's work.

2.3.20 MSRA

Gürses and Santen [39] have introduced a method that integrates the process of identifying security requirements of the end-users into the requirements elicitation process of a multilaterally secure system. Throughout the method emphasis is put on contextualizing security goals by analyzing the different viewpoints like whose security goal is it? against whom? for which functionality? which other users have a mutual interest in or conflict with the given security goal?

2.3.21 Secure Tropos

Mouratidis & Giorgini [67] have introduced extensions to the Tropos methodology to enable it to model security concerns throughout the whole development process. A description of the new concepts and modelling activities is given along with a discussion on how these concepts and modelling activities are integrated to the current

stages of Tropos. They used a real life case study from the health and social care sector is used to demonstrate the approach.

2.3.22 KAOS

Van Lamsweerde [32] overviews a systematic, goal-oriented approach to requirements engineering for high-assurance systems. The target of this approach is a complete, consistent, adequate, and structured set of software requirements and environment assumptions. The approach is model-based and partly relies on the use of formal methods when and where needed for RE-specific tasks, notably, goal refinement and operationalization, analysis of hazards and threats, conflict management, and synthesis of behavior models. The method, known as Keep All Objectives Satisfied (KAOS), has been developed and refined for more than fifteen years of research, tool development, and experience in multiple industrial projects.

2.3.23 SEPP

Hatebur et. al. [68] have presented a security engineering process based on security problem frames and concretized security problem frames. Both kinds of frames constitute patterns for analyzing security problems and associated solution approaches. They are arranged in a pattern system that makes dependencies between them explicit. They also described step-by-step how the pattern system can be used to analyze a given security problem and how solution approaches can be found.

2.3.24 UMLintr

Hussein & Zulkernine [14] present a framework for developing components with intrusion detection capabilities. This framework uses UMLintr, a UML profile for intrusion specifications. The profile allows developers to specify intrusion scenarios using UML diagrams. Specifying intrusion scenarios using the same language that is used for specifying software behavior eliminates the need for separate languages for describing intrusions. Other software specification languages can be easily adopted into this framework. The outcome of this framework is components equipped with intrusion detectors.

2.3.25 SREF

Haley et. al. [15] presents a framework for security requirements elicitation and analysis. This framework is based on constructing a context for the system, representing security requirements as constraints, and developing satisfaction arguments for the security requirements. The system context is described using a problem-oriented notation, then is validated against the security requirements through construction of a satisfaction argument. The satisfaction argument consists of two parts: a formal argument that the system can meet its security requirements and a structured informal argument supporting the assumptions expressed in the formal argument. The construction of the satisfaction argument may fail, revealing either that the security requirement cannot be satisfied in the context or that the context does not contain sufficient information to develop the argument.

2.3.26 MOSRE Framework

Salini & Kanmani [16] have proposed a Model oriented framework to Security Requirement Engineering (MOSRE) for Web Applications and applied MOSRE framework for E-Voting system. By applying Modeling technologies to Requirement phases, the Security requirements and domain knowledge can be captured in a well-defined model and it is better than traditional process.

2.3.27 STS

Paja et. al. [26] propose the STS approach for modelling and reasoning about security requirements. In STS, security requirements are specified, via the STS-ml requirements modelling language, as contracts that constrain the interactions among the actors in the socio-technical system. The requirements models of STS-ml have a formal semantics which enables automated reasoning for detecting possible conflicts among security requirements as well as conflicts between security requirements and actors' business policies. They also applied STS to a case study about e-Government, and report on promising scalability results of our implementation.

2.3.28 DIGS Framework

Riaz et. al. [17] developed Discovering Goals for Security (DIGS) framework, which models the key entities in information security, including assets and security goals. They systematically developed a set of security goal patterns that capture multiple dimensions of security for assets. DIGS explicitly captures the relations and assumptions that underlie security goals to elicit implied goals. They map the goal patterns to NIST controls to help in operationalizing the goals. They also evaluated DIGS via a controlled experiment where 28 participants analyzed systems from mobile banking and human resource management domains.

2.3.29 STORE: Security Threat Oriented Requirements Engineering Methodology

Ansari et. al. presented the STORE Methodology which is a ten-step sequential security requirements engineering methodology. This methodology is based on security threats analysis, which includes the identification of four points: PoA, PoB, PoC and PoD for effective security attack analysis. The STORE methodology identifies and priorities all such stakeholders based on their importance. The STORE methodology considers security threats for identifying security requirements with the help of potential stakeholders [74].

2.4 Comparative Analysis

The following Table 2.2 shows the summary of different security requirements approaches. After the analysis the researcher has reached on the result that each of the selected initiatives provides us with highly important aspects that have to do with security requirements engineering. This is the summary that can be used as the basis for new methodologies / approaches / frameworks / techniques or as extensions to those approaches that already exist.

Table 2.2 Comparative Study of security requirements engineering approach

SRE	Initiative	Year	Standard	Based on	Contribution

Approaches					
GBRAM	AI Anton	1996	-	Goal based	Formulate privacy and security policies using heuristic activities
Abuse Cases	McDermott	1999	-	Abuse Cases	Develop abuse cases with the help of use cases to capture and analyze security requirements in a simple way.
Secure i*	Yu	2001	-	Business process modeling, Software process modelling	i* framework for modelling and reasoning about organizational environments and their information systems
SecureUML	Lodderstedt et. al.	2002	-	Model Driven Architecture	SecureUML (security modelling language for formalizing access control requirements based on UML)
UMLSec	Jürjens	2002	ISO/IEC 15408, ISO/IEC 27001, ISO/IEC 17799, ISO/IEC 13335, IEEE 830-1998	Unified Process	UMLSec (UML extension for secure systems development)
SIREN	Toval et. al.	2002	IEEE 830-1998, IEEE-1233,	Spiral process	Present a practical method to elicit and specify the system and software requirements

			IEEE-1207.1 & partially ISO/IEC 15408		
CORAS	F den Braber et. Al.	2003	ISO 31000	Model based Sequential Process	Combined UML and Unified Process (UP) together to assist the model-based risk assessment.
Security use cases	Firesmith	2003	ISO/IEC 9126-1 and 9126-2	Conducted by assets and risk	Security use cases (UML extension for modelling security requirements in use case diagrams)
Abuse frames	Lin et. al.	2004	ISO 13335		Object-oriented modeling technique, use cases, to identify and analyze security requirements.
Holistic SRE	Zuccato	2004	ISO/IEC 15408, ISO/IEC 17799, ISO 9000:2000, ISO/IEC 13335	Unified Process	Elicit security requirements according to system-theoretic considerations
SQUARE	Mead and Stehney	2005	-	Sequential Process	SQUARE: 9-step process for eliciting,categorizing, and prioritizing securityrequirements
Misuse cases	Sindre and Opdahl	2005	-	Threats and risks	Extends traditional use cases to cover misuse actions

ISSRM	Mayer et. al.	2005	ISO 27001	Sequential Process	Improves the effectiveness of an iterative security engineering method
Threat Based SRE	Myagmar et al.	2005	-	Threat modeling based	Threat modeling based approach for SRE
Abuser stories	Peeters	2005	-	Agile requirements engineering	Extended the agile practices to deal with security in an informal
CLASP	Viega	2005	-	Resource-centric	CLASP: handles security requirements through a structured walkthrough of resources
Tropos Goal-Risk	Asnar & Giorgini	2006	-		Extended Tropos goal model to analyses risk.
SREP	Mellado el al.	2006	CC	Asset based and Risk driven	Standard-centred process and a reuse-based approach which deals with the security requirements.
Security ontology	Tsoumas and Gritzalis	2006	CRAMM, COBIT	ontologies	A security ontology which extends the DMTF Common Information Model (CIM)
MSRA	Gürses	2006	-	Multilateral security requirements	Integrates the process of identifying security requirements of the end-users into the requirements elicitation process

Secure Tropos	Mouratidis & Giorgini	2007	ISO/IEC 17799	Agent oriented software development	Methodology Tropos Framework for modelling and analyzing security and trust requirements
KAOS	Lamsweerde	2007			Use of antimodels to elaborate security requirements
SEPP	Hatebur el. al.	2007	CC		SRE based on security problem frames and concretized security problem frames.
UMLintr	Hussein and Zulkernine	2007	-	Component-Based Software Engineering	UMLintr (UML profile for intrusion identifications)
SREF	Haley et. al.	2008	-		Framework for security requirements elicitation and analysis
MOSRE	Salini & Kanmani	2013	-	Model based	Model oriented framework for SRE
STS	Paja et. al.	2015	-		Approach for modelling and reasoning about security requirements
DIGS	Riaz et. al.	2016	-	Security Goal based	Framework, which models the key entities in information security
STORE	Ansari et. al.	2018		Threat based	Methodology provides threat oriented security requirements

Several requirements engineering approaches have been developed in order to develop a secure software product. Each approach represents unique features that make

it syntactically and semantically different from the other one. Toval et al. [27] presented a practical approach for managing the security of the information systems from the requirement engineering process. This approach is a particularization of a general-purpose process for requirements reuse called SIREN, which shortens the development process since the analyst starts from a reusable set of requirements. Jürjens [20] and Popp et al. [21] presented the extension of UMLsec of UML to add security-related information within the UML diagram in a system specification. This security requirements model has multilevel security and compulsory access control. Basin et al. [69] have proposed a model-driven security approach. For that, they specialized model-driven architecture paradigm into model-driven security. After that, they designed an application for developing software systems through process models, in which they integrate the process design language UML with a security modeling language SecureUML for assigning access control requirements. The process models in the integration of UML and SecureUML are used to repeatedly produce security architectures for distributed software applications.

System-theoretic considerations based approach [29] was proposed for the elicitation of security requirements. This approach shows that security requirements can be described with the aid of analysis in the business environment, meeting with all the potential stakeholders and risk investigation. The Tropos methodology [36] is anticipated to make the analysis and design tasks easier in the software development process. Tropos approach is based on ranking and revised components of the i* framework. Tropos approach based on the concept of using requirements modeling idea of making a model of the system-to-be within its operational environment, which is incrementally sophisticated and extensive to provided that both a common interface for a variety of software development tasks and as a foundation for the documentation and development of the software product. The development phases of Tropos approach are mainly initial requirement, late requirements, architectural design, complete design, and implementation. Several authors Giorgini et al. [70], Mouratidis & Giorgini [67], Ali et al [71], Dalpiaz et al. [72] have proposed the extensions of Tropos. Giorgini et. al. [70] is one of the most significant extensions which present a formal framework for analyzing and modeling security and trust requirements.

Further Jennex [37] proposed a methodology which is based on meta-notation to insert security information to access system development diagrams. This approach

considered the technique of integrating security design into the software development lifecycle. Another author Firesmith [55, 53, 23] proposed a technique with some steps which define security requirements from reusable templates. He has performed a security analysis which is based on two fundamental concepts acquired from OCTAVE (Operationally Critical Threat, Asset, and Vulnerability Evaluation). He proposed security use cases as a method that should be used to identify the security requirements that the applications will effectively accomplish to secure themselves from the significant security threats in the software development process. SQUARE [38] is a methodology which consists of nine step systematic security requirements engineering methodology. This methodology delivers a technique for eliciting, categorizing, and prioritizing security requirements for IT application and software product. The SQUARE methodology is mainly used for building security conceptions in the early phases of the software development lifecycle. The SQUARE methodology may also be used for documenting and evaluating the security requirements of software systems. Another author Peeters [22] extends the agile practices to solve the security issues in an informal, open and assurance determined way. With the intention of rising the agility of requirement engineering, He sets, advancing the concept of using “abuser stories”. These stories identify how the attackers may abuse the system and threaten stakeholder’s assets. The abuser stories thus make the establishment of security requirements easier.

Sindre and Opdahl [24] extend the traditional use case approach to also consider misuse cases, which signify the functionality, not wanted in the system to be developed. Viega [11] proposed a subset of CLASP, which is a set of process pieces for helping development organizations improve the security of their software. The basic idea behind the way that CLASP handles security requirements is the performance of a structured walk-through of resources, determining how they address each core security service throughout the lifetime of that resource. Another approach which is based on threat modeling for security requirement elicitation done by Myagmar et al. [30] investigate how threat modeling can be used as foundations for the specification of security requirements and they also present three case studies of threat modeling.

Tsoumas & Gritzalis [12] designated a security framework of an arbitrary information system. This is a security ontology-based framework which extends the DMTF Common Information Model (CIM) with ontological semantics with the intention of

practicing it as a container for information system security associated info. This framework delivers security attainment and knowledge management in an effective way. Gürses [39] introduced MSRA (Multilateral security requirements analysis) method that integrates the process of eliciting security requirements of the end-users into the requirements elicitation process of a multilaterally secure system. The aim of this approach is to identify and analyze security requirements from the multiple views of stakeholders. Basin et al. [25] have proposed a model-driven approach for building secure software system and implemented this approach in a UML-based CASE tool.

Hussein & Zulkernine [14] have proposed a framework for developing components with intrusion detection capabilities. In this framework, the first step elicits the requirements, in which the developers detect services and intrusions. Specifically, they capture user requirements concerning the services and functionalities provided by the components and identify the unwanted or illegal usage of components by intruders. Intrusion scenarios are elicited through the use of additional approach misuse cases of a UML profile called UMLintr. Van Lamsweerde [32] extends KAOS to comprise the embellishment of security requirements. His proposed approach is a model based and to a certain extent relies on the use of formal methods when and where needed for RE-specific tasks, notably, goal refinement and operationalization, analysis of hazards and threats, conflict management, and synthesis of behavioral models. Mellado et al. [31] proposed a standards-based process, named SREP (Security Requirements Engineering Process) that deals with the security requirements during the early stages of software development in a systematic and intuitive way.

Haley et al. [15] proposed a framework for security requirements elicitation and analysis. This framework is based on creating a background for the software system, demonstrating security requirements as constraints, and elaborating satisfaction arguments for the security requirements. Hassan et al. [73] proposed a formal analysis and design for engineering security (FADES) as the first goal-oriented software security engineering approach. This approach provides an automated bridge between the goal-oriented semi-formal Knowledge Acquisition for automated Specifications (KAOS) framework and the B formal method. Salini & Kanmani [16] proposed a Model oriented framework to Security Requirement Engineering (MOSRE) framework that uses a use case diagram to elicit security requirements. MOSRE has been applied to E-Health web applications. To determine the security requirements, it has the ability to identify,

quantify and rank the risks of security threats and vulnerabilities. El-Hadary & El-Kassas [40] proposed a methodology based on problem frames and abuse frames for security requirement elicitation. They used problem frames to build a security catalog and to represent security requirements, while the abuse frames are used for threat modeling. Paja et al. [26] proposed STS approach for modeling and reasoning about security requirements. In this approach, security requirements are identified by the STS-ml requirements modeling language. The requirements models of STS-ml have a formal semantics which allows automated reasoning for detecting possible conflicts among security requirements along with conflicts between security requirements and actors' business policies. Riaz et al. [17] developed Discovering Goals for Security (DIGS) framework, that models the key entities in information security, including assets and security goals. Ansari & Pandey [74] proposed a framework by integrating three effective security requirements elicitation techniques, Threat modeling, Misuse case and Attack pattern for the elicitation of security requirements.

Recently, authors have discussed and proposed new approaches to security requirements engineering. Mufti et. al. [35] developed the Requirements Engineering Readiness Model (SRERM) to allow organizations to measure their security requirements engineering (SRE) readiness levels. They conducted two case studies to measure the usability of the SRERM. Rehman & Gruhn [19] proposed security requirements framework for CPSs that overcomes the issue of security requirements elicitation for heterogeneous CPS components.

2.5 Relevant Findings

Most security requirements engineering methods consider the complete CIA triad. Some of them also address other requirements than security requirements. Although considering stakeholders view in security requirement engineering is an important concern, but only some SRE approaches like MSRA, KAOS, Secure Tropos, SQUARE, STS, DIGS, and SRERM address this concern. This doesn’t mean that it is impossible to consider the views of different stakeholders using other methods. However, most of the security requirements do not capture this issue in their various activities. MSRA by Gürses & Santen [39] is the only security requirements engineering approach that proposes steps to establish a compromise between different security concerns accepted

by different stakeholders. All stakeholders incorporate during this process [75]. The concept of a counter-stakeholder in MSRA cannot be considered a threat agent, because it does not imply that the counter-stakeholder will threaten the system. SecureUML is concerned with access control only. Therefore, general threat analysis is out of the scope of this method.

2.6 Conclusion

This chapter reviewed several security requirements engineering approaches proposed by different authors. Several key concepts of different security requirements engineering approaches have been discussed. The researcher also discussed several security challenges that are raised by existing security requirements engineering approaches. Many security requirements engineering approaches consider the complete CIA triad. Some of them also address other requirements than security requirements. Although considering stakeholders view in security requirement engineering is an important concern, but only some security requirements engineering approaches address this concern. This doesn't mean that it is impossible to consider the views of different stakeholders using other methods. However, most of the security requirements do not capture this issue in their various activities. Security requirements engineering approach should propose steps to establish a compromise between different security concerns accepted by different stakeholders. All potential stakeholders must be incorporated during the security requirements engineering process. There are comprehensive security threat models but security attack analysis is not emphasized. There is a need to provide steps for considering security interests of all the stakeholders of the software system and also involves the effective mechanism for threat agent identification and risk analysis for easy, complete, effective and efficient security requirements engineering.

Chapter 3 Criteria Decision Making for Effective Security Requirements Engineering

"In a room full of top software designers, if two agree on the same thing, that's a majority."

- ***Bill Curtis***

3.1 Background

In recent years, literature has offered a number of security requirements engineering methods that assist the software system designers and developers to implement security concern presented in traditional development model. Software security requirements have become an important part of the overall requirements analysis process during software development process. Security requirements are one of the most important parts among all non-functional requirements. Negotiation with software security requirements during software development may result in disastrous failure of the software product affecting enormous damage of valuable assets. A special attention should be given to the security requirements of the software product during the software development as software system does not exist physically. The researcher noticed that different security requirements engineering techniques are available for the elicitation of security requirements in order to develop a quality software product.

The objective of this chapter is to identify the different criteria for the analysis of different security requirements engineering approach. Further the researcher decides what criteria are important while designing an effective security requirements

engineering approach. Which criteria in a security requirements engineering approach requirement has a higher priority than the other criteria. The researcher evaluate each criterion by itself or in comparison with other requirements, while ignoring the relationships existing between them, and also without regard to the effect of other on this priority value. In this chapter, the researchers formally review the different security requirements engineering approaches and identify the different criterions which are crucial for the development of an effective security requirements engineering approach.

The main purpose of prioritizing security requirements engineering criterion is to help in selection and development of effective and efficient security requirements engineering approach. In order to improve decision making and to achieve this, it is necessary to establish a process adapted to requirements which takes into account these relationships, to help provide consistency to the prioritization done. For that, before assigning a final value of priority to each criterion it is important to consider the operational significance of those criteria in context of effective security requirements engineering with which it is in interdependency. In this chapter the researcher use the principle of pair-wise comparisons of AHP [76], the method that is deliberated as the most helpful method, to help to accomplish the best decisions conceivable and to clearly present the rationality of the decision made about prioritization.

3.2 Analytic Hierarchy Process (AHP)

Thomas L. Saaty proposed a multi-criteria decision-making approach known as Analytic Hierarchy Process (AHP) [76]. The AHP method consists of three main methods, including construction of hierarchy, priority analysis, and verification of continuity [77]. The most distinctive feature of the AHP is the clear consideration and application of personal judgment relative to other approaches. The AHP algorithm consists essentially of two steps:

- Determine the relative weights of the decision criteria.
- Determine the relative rankings (priorities) of alternatives.

Only a first step is used in some studies to rank criteria [78]. First, we need to break down complex multi-criteria decision issues into their component parts of which all possible attributes are arranged into multiple hierarchical levels.

3.2.1 Determine the relative weights of the decision criteria

The construction of the hierarchical structure is a foundation stone of AHP. It is considered as an important step of AHP, and there is no specialized approach for making a hierarchy. The construction of hierarchy is a top-down process and comprises of several levels. The elements of hierarchical levels are managed in such a way that they are on the same scale and magnitude. The elements of the same hierarchy level must be correlated with the other corresponding factors of the structure. The formation of AHP hierarchy normally starts from the higher-level.

Software security engineering procedure should include the use of repeatable and organized processes in an effort to guarantee that the set of requirements found is complete, reliable, easy to recognize and analyzable by the different participants involved in the software development process [7]. Security needs to be considered as a quality constraint in all the phases of software development process [79]. To develop a security-critical software system [80] many security requirement frameworks have been developed by different authors. Some of the famous security requirements engineering approaches are STORE [74], MOSRE [16], Secure Tropos [67], SREP [66], CLASP [11], SQUARE [38], CORAS [61], and UMLsec [20]. Security requirements engineering is an important activity since bad security requirements can lead to ineffective security or worth security holes [81].

However many methodologies and frameworks have been proposed for software, but there is still a requirement to improve them [82]. Many other authors identified different issues with the existing security requirements engineering approaches [75, 83, 85]. There is a need to understand the concern of different security experts before the development of any security requirements engineering approach. A security requirements engineering approach can have different features to provide the effective elicitation of security requirements. The assessment features of different security requirements engineering approaches have been classified as eight main criteria groups after analyzing the several security requirements engineering approaches: Usability, Learnability, Stakeholder Involvement, Accuracy, Risk Analysis, Threat Identification, Adaptability and Scalability. The characteristics classified according to these main criteria groups are discussed below.

- **Usability**

This criterion ensures that the security requirements engineering method isn't excessively difficult and can be suitably performed effortlessly. The requirements engineers and stakeholders can completely perform the elicitation method in a sensible length of time. The security requirements engineering approach should be easy to use by the end-users, deprived of the need of undertaking any specific training.

- **Learnability**

  This criterion ensures that the software developers and stakeholders can easily learn the implementation and execution of the security requirements engineering method in a sensible length of time.

- **Stakeholder Involvement**

  This criterion assess that the stakeholders are expected to involve in the process of security requirements engineering method in analysing their security-critical requirements.

- **Accuracy**

  This criterion ensures that the outcome of security requirements engineering method in form of specification documents conforms to the correct, complete and standard.

- **Risk Analysis**

  This criteria ensures that the security requirements engineering method is capable of identifying, analysing the security risks in a software-intensive system.

- **Threat Identification**

  Threat identification criteria guarantee that the security requirements engineering approach capable in recognizing complex modern threats, verification, and validation task.

- **Adaptability**

  The criteria assess the performance to generate effective security requirements in multiple environments. For example, the security requirements engineering approach works equally as well with a software system that is near completion as with a software project in the preparation phases.

- **Scalability**

  The criteria evaluate the performance the security requirement engineering approach can be used to elicit the security requirements of software projects of different sizes, from readiness-level systems to minor applications.

After that, we have to compare each criterion in a pair on the basis of their own experience and knowledge at the same point. For example, in the second level, each of the two criteria is compared with the target at each time, whereas in the third level, each of the two attributes of the same criteria is compared with the corresponding criterion at a time. Comparing pairs of elements is generated by assigning the pairs a priority value to each criterion, using a scale of 1-9 (1: equal importance, 3: moderate importance, 5: strong importance, 7: very strong importance, 9: extreme importance and (2, 4, 6, 8): intermediate between the two adjacent judgments). The following Table 3.1 shows the Saaty's 1-9 scale.

Table 3.1 Saaty's 1–9 scale

Numerical rating	Description	Explanations
1	Equal importance	Two criteria contribute to the objective with equal relevance.
3	Moderate importance	A criteria is slightly more important than one other
5	Strong importance	Judgment strongly favors one element over another
7	Very strong importance	An element is strongly important than another
9	Extreme importance	The compared element is favored over another
2,4,6,8	Intermediate values	Used to represent a compromise between the above-mentioned preferences

AHP is a simple technique to prioritize criteria that takes into account the relationships that exist between the different types of criteria, emphasizing that this is in a general context. The overall procedure of the AHP method is shown in the Figure 3.1.

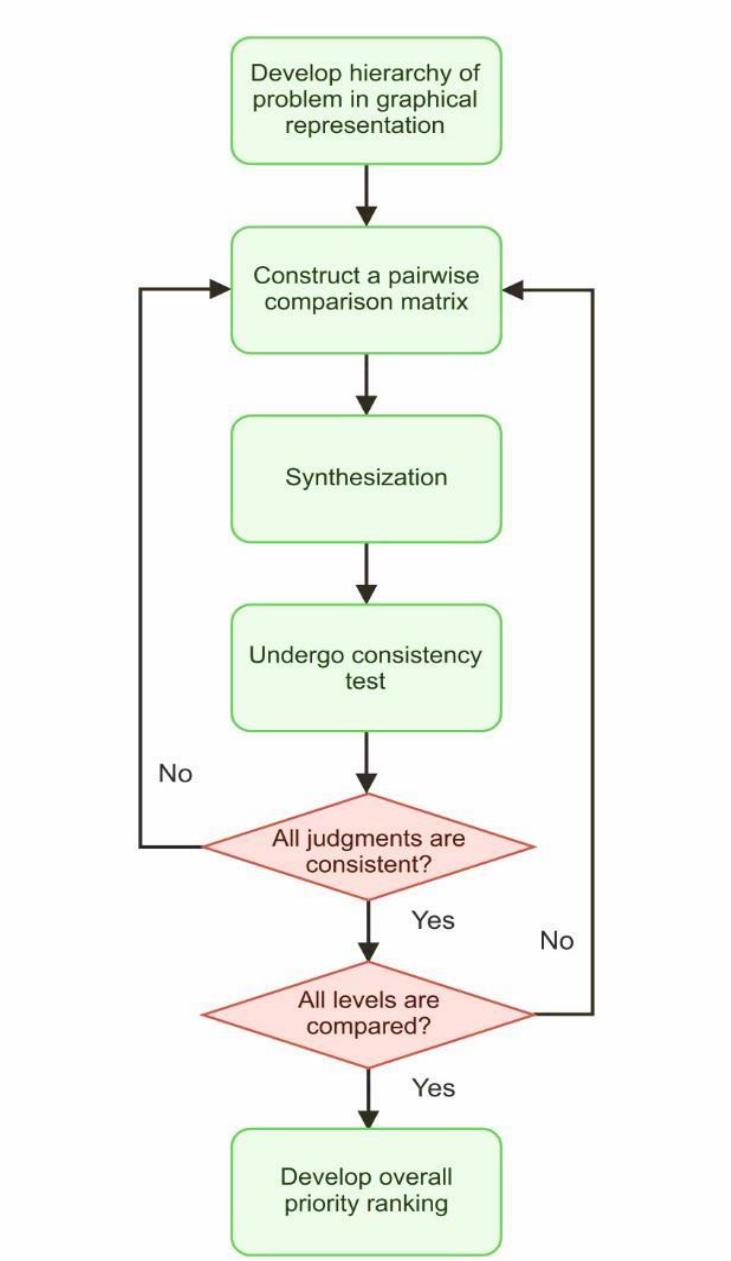

Figure 3.1 Flowchart of the Analytic Hierarchy Process

In AHP process the researcher first structure the decision problem elements into a hierarchy of goal, criteria, sub-criteria, and alternatives then construct a set of pair-wise comparison matrices, the criteria are compared using Saaty's pair-wise comparison scale (1980), as shown in Table 3.1. The pair-wise comparison matrix (A) is illustrated in Equation (1).

$$A = \begin{bmatrix} 1 & a_{12} & \dots & a_{1n} \\ \frac{1}{a_{12}} & 1 & \dots & a_{2n} \\ \dots & \dots & 1 & \dots \\ \frac{1}{a_{n1}} & \frac{1}{a_{n2}} & \dots & 1 \end{bmatrix} \quad (1)$$

Further calculate the eigenvector with the intention of determine the priority weights for the different criteria. In this stage, the column entries are normalized by dividing each entry by the sum of the column. Then the overall row averages are measured. Additionally determine the priority weights of alternatives with respect to criteria.

After determination of priority weights, it is very important to calculate the Consistency Index (CI), as in Equation (2). Where n represents the matrix size and λ_{max} is the biggest eigenvalue of matrix A.

$$CI = \frac{(\lambda_{max} - n)}{(n-1)} \quad (2)$$

Furthermore, the researcher verifies the Consistency Ratio (CR) in order to validate and determine the acceptance of the weights. Since the comparisons are made by means of personal or objective decisions, a degree of ambiguity may have emerged. To ensure consistent decisions, the final operation called consistency checking, which is considered to be one of the AHP's most advantages, is implemented to calculate the degree of consistency between the pair comparisons by measuring the consistency ratio. If the consistency ratio is found to exceed the limit, we should review and review the comparisons between pairs. Once all comparisons are made at all levels and are accurate, the decisions can then be synthesized to decide the priority rating of each criterion. If and only if the CR ≤ 10%, then the obtained weights are acceptable. If CR is more than 0.1, then the results are inconsistent and the judgments must be repeated (Saaty, 1980). Consistency ratio is computed as in Equation (3).

$$CR = \frac{CI}{RI} \quad (3)$$

where Random Indexes (RI) is the average value of CI for random matrices using the Saaty scale obtained by Forman (Forman, 1990) and Saaty only takes a matrix as a consistent one if and only if CR < 0.1. Table 3.2 illustrates the RI with dimensions from 1 to 10.

Table 3.2 Random index for pairwise comparison matrices

Size of matrix (n)	1	2	3	4	5	6	7	8	9	10
RI	0	0	0.52	0.89	1.11	1.25	1.35	1.40	1.45	1.49

3.3 Implementation and Ranking

The first step in this process is the establishment of a hierarchy model. The hierarchy model is composed of the eight criterion group (Usability, Learnability, Stakeholder Involvement, Accuracy, Risk Analysis, Threat Identification, Adaptability, and Scalability). After the hierarchy has been established, the criteria must be evaluated in pairs so as to determine the relative importance between them and their relative weight to the global goal. The study was planned and conducted to identify the different criteria which are considered by the software developer during the selection of effective security requirements engineering approach. A survey form was prepared to determine the prioritization of the characteristics incorporated when choosing effective security requirements engineering approach. This form was given to 10 security experts to collect their estimation on the pair-wise criteria comparisons and an AHP model was created. The hierarchy model incorporating the objectives and criteria when choosing an effective security requirements engineering approach can be seen in Figure 3.2.

The researcher mailed pairwise questionnaire form based on saaty's scale to each of the security experts. The security experts were given a questionnaire that contained a pairwise comparison sheet. The members consisted of 10 security experts who were serving in the different software organizations and who had experience exceeding 10 years having deep knowledge about security requirements engineering process during software development. Please see Appendix A for the pairwise questionnaire form. All security experts were security domain experts who practice different security requirements in their working places to build quality software products and have valuable knowledge about the different modern threats and software attack mechanism. They responded about their satisfaction with their choices. The consistency ratios were less than 0.10 for all the 10 security experts in collected form responses. Further the

collected data analyzed separately for each security expert. A median was calculated according to the importance ratings and a new combined weighted geometric mean of all participants calculated.

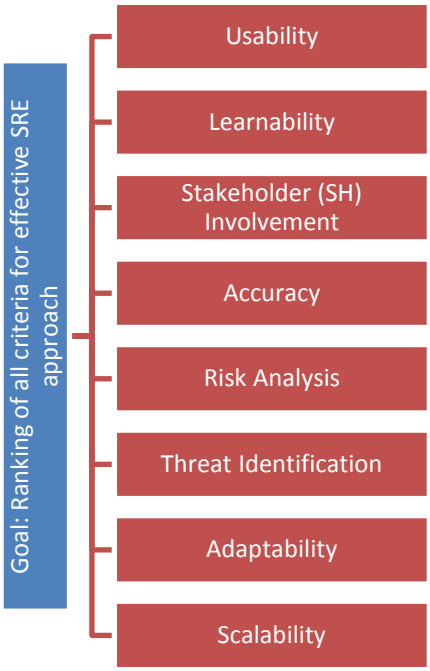

Figure 3.2 The AHP model combining the purpose and criteria

3.4 Results

The priorities of selection criteria were determined by using AHP model. Effective security requirements engineering approach selection criteria were compared in pairs. Ten security experts were asked to fill the pair wise criteria comparison form. The assessment begins by determining the relative weight of the criteria groups by the experienced ten security experts. When the measures for each criterion had been calculated independently, the AHP can aggregate these scores into the higher-level complexity construct. The researcher then calculated the geometric prioritization row geometric mean (RGM) method for each participant. RGM method is a prioritization method that computes the final priority order for alternatives. The researcher computed the weighted scores using the pairwise comparisons of all respondents. The Table 3.3 to Table 3.12 shows the resulting weights for the criteria by the security experts based on their pairwise comparisons.

Table 3.3 Criteria weight matrix by Participant 1

A	1	2	3	4	5	6	7	8
1	**1**	2	1/4	1/3	1/9	1/9	3	3
2	1/2	**1**	1/8	1/3	1/8	1/8	1	1
3	4	8	**1**	2	1/7	1/7	2	2
4	3	3	1/2	**1**	1/7	1/7	3	3
5	9	8	7	7	**1**	1/2	8	9
6	9	8	7	7	2	**1**	9	9
7	1/3	1	1/2	1/3	1/8	1/9	**1**	1
8	1/3	1	1/2	1/3	1/9	1/9	1	**1**

Consistency Ratio (CR) = 7% ≤ 10%

Table 3.4 Criteria weight matrix by Participant 2

B	1	2	3	4	5	6	7	8
1	**1**	2	1/4	1/2	1/9	1/9	4	4
2	1/2	**1**	1/8	1/2	1/9	1/9	1	2
3	4	8	**1**	1	1/7	1/8	5	3
4	2	2	1	**1**	¼	1/8	2	3
5	9	9	7	4	**1**	1	8	8
6	9	9	8	8	1	**1**	9	9
7	1/4	1	1/5	1/2	1/8	1/9	**1**	1
8	1/4	1/2	1/3	1/3	1/8	1/9	1	**1**

Consistency Ratio (CR) = 7% ≤ 10%

Table 3.5 Criteria weight matrix by Participant 3

C	1	2	3	4	5	6	7	8
1	**1**	2	1/4	1/2	1/9	1/9	3	3
2	1/2	**1**	1/3	3	1/9	1/9	4	3
3	4	3	**1**	1	1/8	1/8	2	2
4	2	1/3	1	**1**	1/7	1/7	3	5

5	9	9	8	7	**1**	2	9	9
6	9	9	8	7	½	**1**	8	8
7	1/3	¼	1/2	1/3	1/9	1/8	**1**	1/2
8	1/3	1/3	1/2	1/5	1/9	1/8	2	**1**

Consistency Ratio (CR) = 8% ≤ 10%

Table 3.6 Criteria weight matrix by Participant 4

D	1	2	3	4	5	6	7	8
1	**1**	4	1/4	1/2	1/8	1/8	3	3
2	1/4	**1**	1/7	1/3	1/9	1/9	3	3
3	4	7	**1**	2	1/7	1/7	4	4
4	2	3	1/2	**1**	1/8	1/8	2	2
5	8	9	7	8	**1**	1	8	8
6	8	9	7	8	1	**1**	9	9
7	1/3	1/3	1/4	1/2	1/8	1/9	**1**	2
8	1/3	1/3	1/4	1/2	1/8	1/9	1/2	**1**

Consistency Ratio (CR) = 8% ≤ 10%

Table 3.7 Criteria weight matrix by Participant 5

E	1	2	3	4	5	6	7	8
1	**1**	4	1/2	1	1/9	1/9	2	2
2	1/4	**1**	1/7	1/2	1/9	1/9	2	2
3	2	7	**1**	1	1/5	1/5	4	2
4	1	2	1	**1**	1/8	1/8	3	3
5	9	9	5	8	**1**	1/2	9	9
6	9	9	5	8	2	**1**	8	9
7	1/2	½	1/4	1/3	1/9	1/8	**1**	3
8	1/2	½	1/2	1/3	1/9	1/9	1/3	**1**

Consistency Ratio (CR) = 6% ≤ 10%

Table 3.8 Criteria weight matrix by Participant 6

F	1	2	3	4	5	6	7	8
1	**1**	4	1/6	1/2	1/9	1/9	2	2
2	1/4	**1**	1/8	1/3	1/8	1/8	4	4
3	6	8	**1**	8	1/3	1/3	7	7
4	2	3	1/8	**1**	1/8	1/8	2	3
5	9	8	3	8	**1**	1/2	9	8
6	9	8	3	8	2	**1**	9	9
7	1/2	1/4	1/7	1/2	1/9	1/9	**1**	2
8	1/2	1/4	1/7	1/3	1/8	1/9	1/2	**1**

Consistency Ratio (CR) = 8% ≤ 10%

Table 3.9 Criteria weight matrix by Participant 7

G	1	2	3	4	5	6	7	8
1	**1**	1	1/5	1/4	1/9	1/9	2	2
2	1	**1**	1/7	1/3	1/8	1/9	2	3
3	5	7	**1**	2	½	1/5	4	5
4	4	3	1/2	**1**	1/7	1/7	4	5
5	9	8	2	7	**1**	1	8	8
6	9	9	5	7	1	**1**	9	9
7	1/2	1/2	1/4	1/4	1/8	1/9	**1**	3
8	1/2	1/3	1/5	1/5	1/8	1/9	1/3	**1**

Consistency Ratio (CR) = 6% ≤ 10%

Table 3.10 Criteria weight matrix by Participant 8

H	1	2	3	4	5	6	7	8
1	**1**	4	1/2	1	1/9	1/9	2	2
2	1/4	**1**	1/6	1/4	1/8	1/9	2	1
3	2	6	**1**	4	1/3	1/3	8	8
4	1	4	1/4	**1**	1/8	1/9	8	3

5	9	8	3	8	**1**	1	8	8
6	9	9	3	9	1	**1**	9	9
7	1/2	½	1/8	1/8	1/8	1/9	**1**	1/2
8	1/2	1	1/8	1/3	1/8	1/9	2	**1**

Consistency Ratio (CR) = 6% ≤ 10%

Table 3.11 Criteria weight matrix by Participant 9

I	1	2	3	4	5	6	7	8
1	**1**	3	1/3	1	1/8	1/8	3	4
2	1/3	**1**	1/5	1/3	1/8	1/8	1	2
3	3	5	**1**	5	½	1/3	5	6
4	1	3	1/5	**1**	¼	1/6	1	3
5	8	8	2	4	**1**	1/3	6	5
6	8	8	3	6	3	**1**	8	8
7	1/3	1	1/5	1	1/6	1/8	**1**	3
8	1/4	½	1/6	1/3	1/5	1/8	1/3	**1**

Consistency Ratio (CR) = 6% ≤ 10%

Table 3.12 Criteria weight matrix by Participant 10

J	1	2	3	4	5	6	7	8
1	1	2	1/5	1	1/7	1/8	3	1/3
2	1/2	1	1/8	1/3	1/7	1/9	1/2	1/2
3	5	8	1	7	½	1/3	6	7
4	1	3	1/7	1	¼	1/7	6	1
5	7	7	2	4	1	1	7	8
6	8	9	3	7	1	1	9	8
7	1/3	2	1/6	1/6	1/7	1/9	1	2
8	3	2	1/7	1	1/8	1/8	1/2	1

Consistency Ratio (CR) = 6% ≤ 10%

All individual security experts decisions Consistency Ratio (CR) is less than 10%. Further, the researcher aggregates all individual responses. This aggregation combines

all 10 security experts' inputs to get the aggregated group result. The researcher uses the weighted geometric mean of the decision matrices criteria using the individual security expert's weight as given in the security expert's survey form. The consolidated calculation is based on the geometric mean of the participants as shown below.

Table 3.13 Combined weighted geometric mean of all participants

	1	2	3	4	5	6	7	8
1		2.56	0.27	0.59	0.12	0.12	2.63	2.17
2	0.39		0.15	0.44	0.12	0.12	1.69	1.83
3	3.68	6.46		2.48	0.25	0.21	4.3	4.03
4	1.69	2.29	0.4		0.16	0.13	2.9	2.86
5	8.57	8.27	3.98	6.24		0.78	7.95	7.91
6	8.69	8.69	4.79	7.46	1.28		8.69	8.69
7	0.38	0.59	0.23	0.35	0.13	0.12		1.49
8	0.46	0.55	0.25	0.35	0.13	0.12	0.67	

Number of participants (k) = 10

Number of criteria (n) = 08

Table 3.13 shows the combined weighted geometric mean of participants relative weights to each criterion that have been determined by researchers after analyzing the operational mechanism of different security requirements engineering approach. The resulting weights are based on the principal eigenvector of the decision matrix. The influence of each criterion to the effective security requirements engineering goal is determined by calculations made using the priority vector or Eigenvector. The Eigenvector shows the relative weights between each criterion it is obtained in an approximate manner by calculating the arithmetic average of all criteria. It is observed that the sum of all values from the vector is always equal to one (1). The exact calculation of the Eigenvector is determined only on specific cases. This approximation is applied most of the times in order to simplify the calculation process, since the difference between the exact value and the approximate value is less than 10% [108]. These are the resulting weights for the criteria based on your pairwise comparisons:

The researcher finally arrived at the weights by calculating the geometric prioritization (RGGM) of the different criteria for each dimension according to all security experts,

including normalization according to [109]. The following Table 3.14 shows the final weights for each criterion.

Table 3.14 Comparison Matrix for Effective Security Requirements Engineering Criteria

Comparison Matrix		Usability	Learnability	SH Involvement	Accuracy	Risk Analysis	Threat Identification	Adaptability	Scalability	**NPE**
		1	2	3	4	5	6	7	8	
Usability	1	1	2	1/5	1	1/7	1/8	3	1/3	4.39%
Learnability	2	1/2	1	1/8	1/3	1/7	1/9	1/2	1/2	2.31%
SH Involvement	3	5	8	1	7	1/2	1/3	6	7	20.45 %
Accuracy	4	1	3	1/7	1	1/4	1/7	6	1	6.39%
Risk Analysis	5	7	7	2	4	1	1	7	8	26.11 %
Threat Identification	6	8	9	3	7	1	1	9	8	32.35 %
Adaptability	7	1/3	2	1/6	1/6	1/7	1/9	1	2	3.42%
Scalability	8	3	2	1/7	1	1/8	1/8	1/2	1	4.57%

NPE = Normalized Principal Eigenvector

The following Table 3.15 shows the final rank wise list of each criterion significant during the selection of effective security requirements engineering approach.

Table 3.15 Rank wise criteria for effective Security Requirements Engineering approach

Criterion Comment		**Weights**	**+/-**	**Rank**
1	Usability	4.4%	2.3%	6
2	Learnability	2.3%	0.6%	8
3	SH Involvement	20.5%	9.8%	3
4	Accuracy	6.4%	5.0%	4

5	Risk Analysis	26.1%	7.0%	2
6	Threat Identification	32.4%	11.2%	1
7	Adaptability	3.4%	2.2%	7
8	Scalability	4.6%	3.2%	5

Consistency Ratio (CR) = 9.1 % < 10%

The following Figure 3.3 shows the combined comparative weight result for each criterion given by the security experts for the selection of effective and efficient security requirements engineering approach. Figure 3.4 shows the pie chart representation of the outcome.

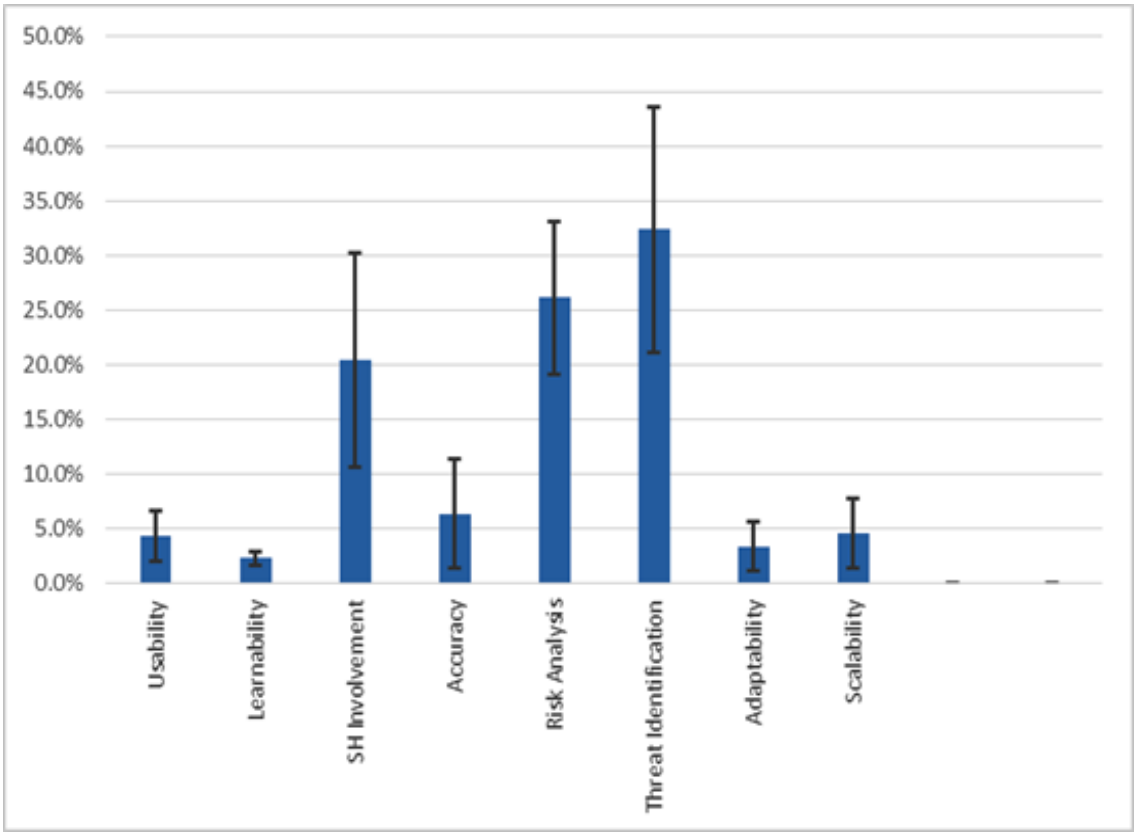

Figure 3.3 Comparative result for each criterion of security requirements engineering

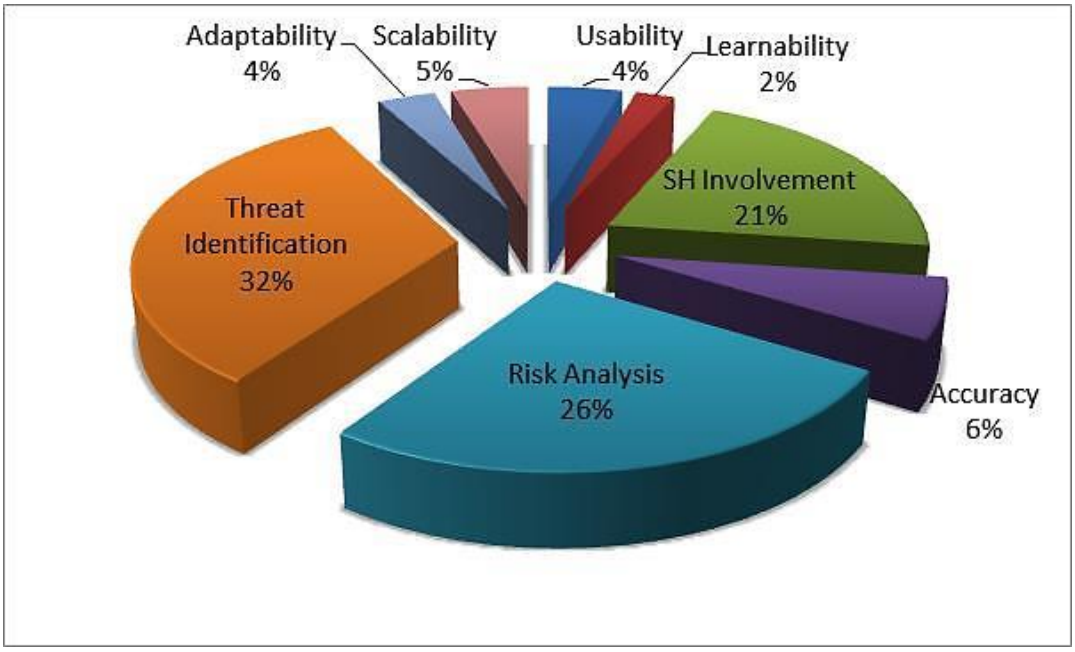

Figure 3.4 Pie chart representation of the result obtained

3.5 Discussion

In this study, it is clear that 'Threat Identification' feature of security requirements engineering approach were most important criteria in security experts' selections. This undoubtedly shows that the recognition of threats is very much significant and every security requirements engineering approach should consider this criterion. In many existing security requirements engineering methods, the threat identification has played a significant role in eliciting effective security requirements for a software project. Prior to eliciting the effective security requirements of a software project, it is significant to identify all the possible threats to the software system. Assessing the threats to a software system assists software developer to build complete and accurate security requirements [30, 74].

Further risk analysis and stakeholders involvement in security requirements engineering process is also important. Although considering stakeholders view in security requirement engineering is an important concern, but only some security requirements engineering approaches address this concern. This doesn't mean that it is impossible to consider the views of different stakeholders using other methods. However, most of the security requirements do not capture this issue in their various activities. Security requirements engineering approach should propose steps to establish cooperation

between different security concerns accepted by different stakeholders. All potential stakeholders must be incorporated during the security requirements engineering process. There are comprehensive security threat models but stakeholder's identification is not emphasized. There is a need to provide steps for considering security interests of all the stakeholders of the software system and also involves the effective mechanism for threat agent identification and risk analysis for easy, complete and well-organized security requirements engineering.

After the criteria were prioritized accordingly, the model used in this study enabled us to analyses the main concept of the consistency of preferences made by the security experts. Furthermore, the main purpose of this study is to provide a model enabling security experts to make a more consistent decision. After the features of the security requirements engineering method became clear, this model can be used to predict effective security requirements engineering approach selection in real world. This study determined priority of different criteria which are considered in selecting or developing an effective security requirements engineering method irrespective of alternatives. These criteria also highlight the features to which a software developer should pay attention.

3.6 Conclusions

The study results suggest that the criteria Threat identification (32.4% of the weight of influencing factors) is more important than Risk analysis (26.1%), SH Involvement (20.5%), Accuracy (6.4%), Scalability (4.6%), Usability (4.4%), Adaptability (3.4%) or Learnability (2.3%) in manipulating performance towards security requirements engineering approaches. Determining weights of essential motivation, purpose, and consciousness focus areas can help security decision-making and compliance with policy, and support design of effective security requirements engineering. However, these weights may in turn be affected by local organisational and educational factors. The presented AHP results in this chapter can be used to select or design an effective security requirements engineering approach which may assist the software developers in developing trustworthy quality software products.

Chapter 4 Proposed Security Requirements Engineering Methodology

"There were 5 Exabytes of information created between the dawns of civilization through 2003, but that much information is now created every 2 days."

- ***Eric Schmidt***

4.1 Background

Software security requirements engineering, which is a process and preparation to identify software security problems in a organized way, is recognized to be a very essential part of the software development process for the accomplishment of secure and quality software product. Security requirements engineering delivers different methodologies, frameworks, techniques, methods and standards for undertaking this task during the early stages i.e. requirement phase of the software development life cycle, subsequently the construction of security into the initial phases of the software development process is economical and also transports approximately extra vigorous software designs [86]. Security requirements engineering should include the use of repeatable and organized processes in a determination to guarantee that the set of security requirements identified is comprehensive, reliable, understandable and analysable by the security requirements engineer and other participants involved in the development of the software product [7]. The elicitation of effective and efficient security requirements is an important and challenging task. It is an important task

because there are many problems associated with the consideration of security issues during the software development phases that must be overcome. Security requirement engineering is challenging because there are requirements that such a security-oriented approach must satisfy. Generally, software security is not considered by the developers during the early phases of the software development life cycle [18].

A complete requirements specification document should comprise both functional requirements and non-functional requirements. In this information technology era, different types of business, bank, education, and many other sectors heavily depending of information technology applications as it provides automation to their systems through different software and applications which play a serious role, it is unconditionally important to guarantee that the software products are safe right from the very beginning [58].

Most security Requirements engineering approach normally does not comprise all significant stakeholders and does not use the well-organized techniques for stakeholder identification and prioritization [41]. Normally the security requirements specification is incomplete, confusing, conflicting, not cohesive, disorganized, infeasible, obsolete, unable to be validated, and not usable by their anticipated persons [8]. Several authors have been proposed methodology for analyzing security requirements engineering. All effective and efficient security requirements should be well organized in a systematic manner otherwise the software system cannot be evaluated for accomplishment. Therefore, there is a need to develop a framework which is capable of eliciting more effective and efficient security requirements by considering all the issues which are neglected by the previous approaches.

The main objective of this research is to design and develop a novel security threat oriented security requirements engineering methodology that is capable of eliciting security requirement which is effective, efficient, complete, clear, consistent, organized, feasible, up to date and easy to be validated. This methodology is especially suitable for any type of software product that requires security from the beginning of the development process. This chapter presents a security threat oriented requirement engineering (STORE) methodology which is a ten step systematic process. Here the researcher describes every step and also the participants involved with every step. Our main aim in the proposed methodology is to get as many effective and efficient security requirements as possible for the secure and quality software product development.

4.2 Security Threat Oriented Requirements Engineering (STORE) Methodology

The proposed security requirements engineering methodology is a ten-step sequential process which provides an effective, efficient and systematic way of eliciting and documenting security requirements for the software as well as web-based applications from the early phases of software development. In this methodology, security requirements are often discussed in the context of threats. Threat helps the security requirement engineer to calculate the risk associated with it and also represents the adversary's abilities. Stakeholder plays an important role in the STORE methodology. Figure 4.1 shows the security requirement engineering concept model for the proposed STORE methodology.

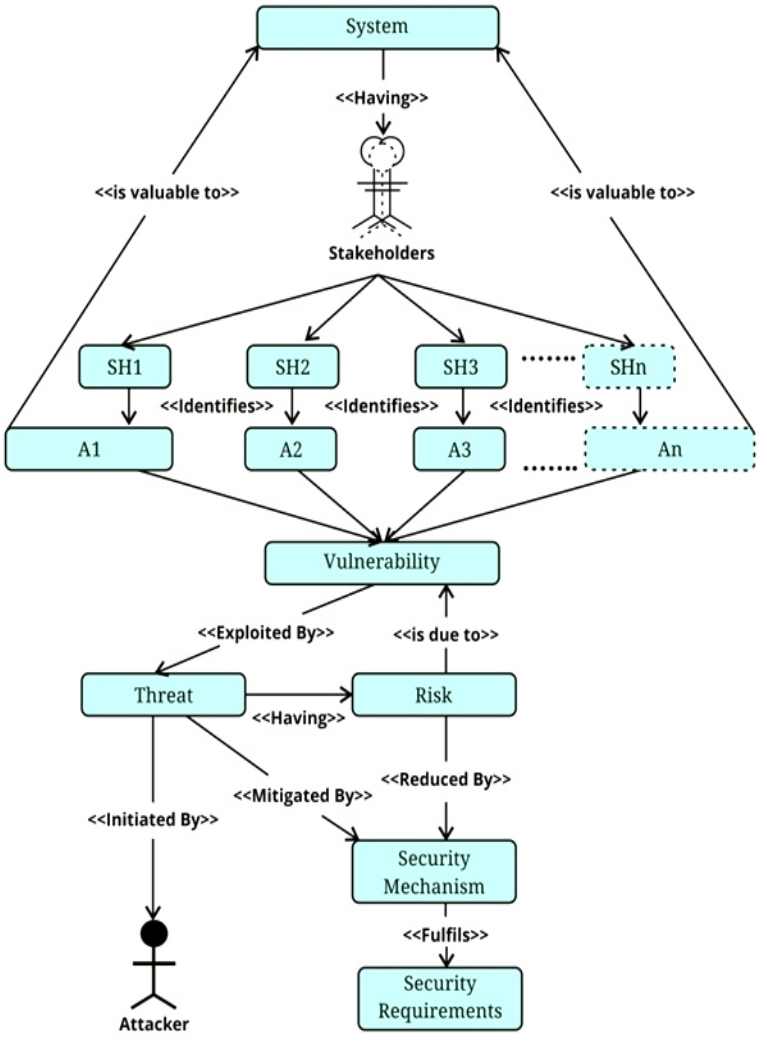

Figure 4.1 Security requirement engineering concept model for STORE methodology

A software system which is to be developed can have several stakeholders, only a few stakeholders are associated with the security of software products. Those stakeholders, who have security concern of the software system, have knowledge about the related assets of the systems which are to be protected from the threats. In STORE methodology we identify and prioritize all such stakeholders based on their importance. It is important to consider every significant stakeholder from the beginning of software development. The proposed methodology considers security threats for identifying security requirements with the help of potential stakeholders. These stakeholders help the requirement engineer in asset identification of the software product. The STORE methodology starts with identifying system goals.

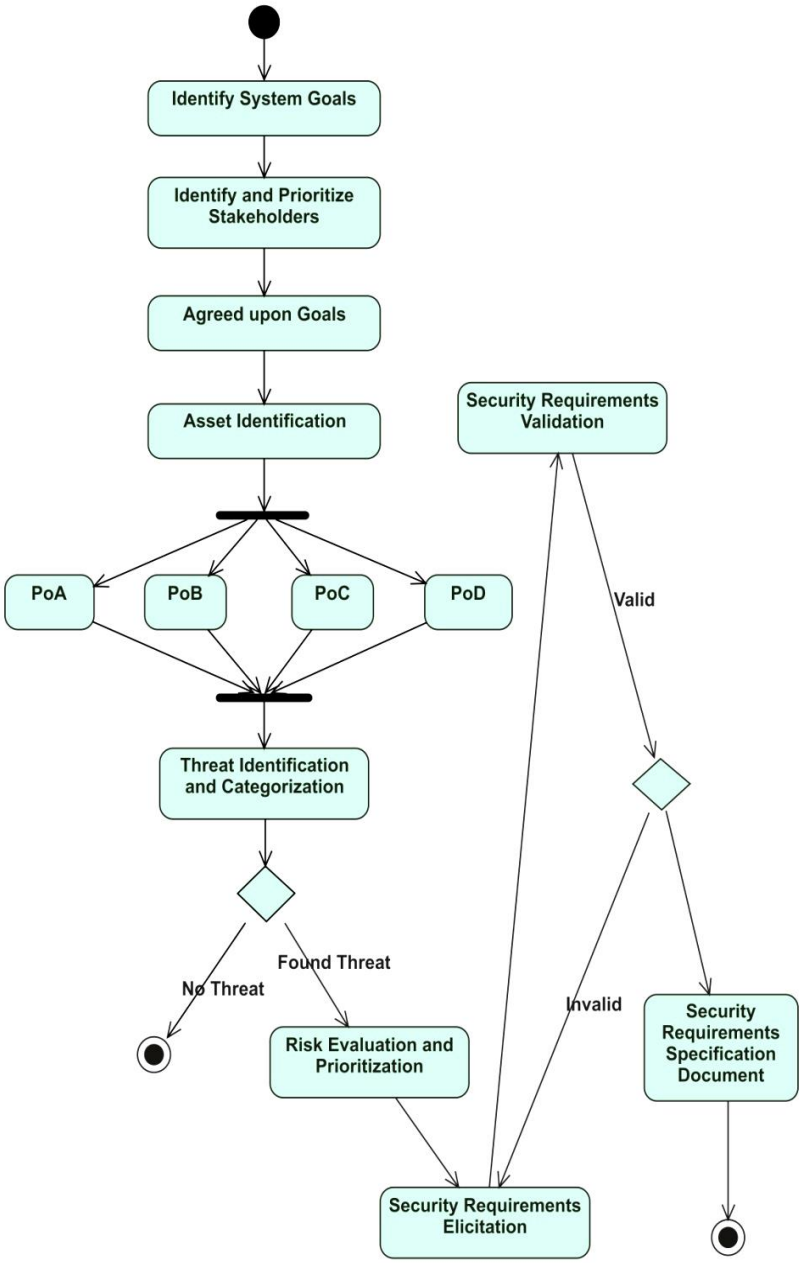

Figure 4.2 Activity diagram of STORE Methodology

Each step of STORE methodology is equally important. The requirement engineer can't skip or jump to any other step because it is a systematic approach for eliciting and documenting security requirement. The STORE methodology is completely based on threats because after identifying potential threats the requirement engineer is capable of eliciting security requirements with the help of threat dictionary. Figure 4.2 shows the activity diagram of STORE methodology.

4.3 Details of STORE Methodology Steps

The STORE Methodology is a ten-step security requirements engineering methodology. This section is to clarify the purpose of each step, the expectations of the participants, what will be the possible taking in and taking out for each step and also the appropriate approach for each step. The following section lists all the steps of STORE methodology as well as describing the functionality of each step.

Step 1. Identify System Goals

Step 2. Identify and Prioritize Stakeholders

Step 3. Agreed upon Goals

Step 4. Asset Identification

Step 5. Security Attack Analysis

A. Point of Attack (PoA)
B. Point of Belief (PoB)
C. Point of Conjecture (PoC)
D. Point of Dependency (PoD)

Step 6. Threat Identification and Categorization

Step 7. Risk Evaluation and Prioritization

Step 8. Security Requirements Elicitation

Step 9. Security Requirements Validation

Step 10. Security Requirements Specification Document

The following Table 4.1 shows the different steps of STORE methodology and also discuss the taking in, approach, different participants and taking out at each step.

Table 4.1 Steps in the proposed STORE Methodology

Step No.	Step	Taking in	Approach	Participants	Taking out
1.	Identify System Goals	Objectives of the proposed system, Policy and Procedure	Interview, Brainstorming	Requirement Engineer, Client	List of System Goals
2.	Identify and Prioritize Stakeholders	System Goals	Review, Analysis	Requirement Engineer	Stakeholders

3.	Agreed upon Goals	System Goals	Meeting	Requirement Engineer, Stakeholders	Agreed upon Goals
4.	Asset Identification	Stakeholder's Valuable Asset	Interview, questionnaire, brainstorming	Requirement Engineer, Stakeholders	Valuable Asset
5.	Security Attack Analysis	Valuable Asset	Security Attack Analysis	Requirement Engineer, Security Expert	PoA, PoB, PoC, PoD
6.	Threat Identification and Categorization	PoA, PoB, PoC, PoD	STRIDE Analysis	Requirement Engineer	Categorized & Prioritized Threats
7.	Risk Evaluation and Prioritization	Categorized & Prioritized Threats	DREAD Risk Assessment Methods	Requirement Engineer, Risk Manager	Risk Assessment Report
8.	Security Requirements Elicitation	Potential Threats	Threat Dictionary	Requirement Engineer	Security requirements
9.	Security Requirements Validation	Security requirements	Review, Walk-through	Requirement Engineer, Security Expert	Valid Security Requirements
10.	Security Requirements Specification Document	Valid Security Requirement	Documentation	Requirement Engineer	Security Requirements Specification Document

4.3.1 Identify System Goals

Identify system goals is the first step of the proposed STORE methodology. Goals are written or verbal statements that illustrate the desired outcomes in a software product

and the development team is aiming to accomplish it through several activities. The goals of a software system have to be identified by the requirement engineer from the client who wants software product. According to Yue [87] and Van Lamsweerde [88] goals are important components to identify in the requirements engineering process. Achieving complete requirements is a major concern for any requirement engineer. Goals provide an accurate condition for getting satisfactory completion of a requirements specification. The outcome of requirement specification is complete with respect to a set of goals if all the goals can be proved to be achieved from the specification and the properties known about the domain considered. For better goal identification the preliminary analysis of the existing system is a vital starting place. The requirement engineer can use several techniques to identify system goals like interview, brainstorming etc. The output of the first step of STORE methodology is to produce a list of system goals.

4.3.2 Identify & Prioritize Stakeholders

The second step of STORE methodology is to identify and prioritize stakeholders who are associated with software system development. Several literature studies are available in the strategic management field which considers the importance of stakeholder by discussing organizations in provisions of a stakeholder model. Sharp et al. [89] described that identifying and prioritizing stakeholder is a vital process and this can be used to evaluate an organization's performance and control its future strategic direction. Several types of stakeholders are associated with any software development process. A stakeholder can express security concerns at different levels of detail [75]. According to Glinz & Wieringa [90] "A stakeholder is a person or organization who influences a system's requirements or who are impacted by that system." A stakeholder can be a customer, end user, developer, requirement engineer, project manager and other persons who are associated with software development. All stakeholders are not equally important, therefore we must prioritize the identified stakeholders. The identified stakeholders may be prioritized into critical, major and minor. Each stakeholder has different security constraints to enforce the same service. Each stakeholder has their own security needs that may conflict with the other stakeholder's needs [91]. All stakeholders in a software development do not have equal importance,

so Glinz & Wieringa [90] prioritize the identified stakeholder roles with their importance.

- Critical: The stakeholder's role is critical if ignoring the stakeholder may destroy the project or render the system useless.
- Major: The stakeholder's role is major if ignoring the stakeholder would have a major unenthusiastic impact.
- Minor: The stakeholder's role is minor if ignoring the stakeholder would have a minor impact on the system.

4.3.3 Agreed upon Goals

The requirement engineers and stakeholders both must be agreed on the identified goals of the software product for the development. This will create a perfect environment for the development of a software product in which every stakeholder can communicate with each other clearly.

4.3.4 Asset Identification

Identifying valuable assets is one of the most important and complex steps of STORE methodology. The outcome of this step should be complete and correct otherwise this methodology will not elicit an effective and efficient security requirement. It is vital to identify all the valuable assets, in order to be able to elicit complete security requirements. Assets may be digital cash, data, password, information, commodities, people, computers etc. The assets should be identified with the help of all the potential stakeholders who are involved in the software development process. The requirement engineer can use different techniques like questionnaires, brainstorming, interview, analysis, discussion for asset identification. The objective of security requirement engineering is to protect these valuable assets from the attacker so assets should be viewed not only in stakeholder perspective but also an attacker's point of view. Each stakeholder identifies assets from his point of view. Later the identified assets have to be categorized and prioritized. Further, the identified assets categorized under Confidentiality, Integrity, and Availability and prioritized as low, medium and high level of preference.

4.3.5 Security Attack Analysis

Once the all valuable assets are identified and prioritized, the next step of STORE methodology is to analyze all the sources of security attack. Analyzing potential security attacks on a software system under development is an important step in engineering secure software systems, as the identified security attacks would elicit necessary security requirements. The following Figure 4.3 shows the four points of security attack analysis that is essential for effective security requirements elicitation.

A. Point of Attack (PoA)

Point of attack for any software or web-based application is the point from where the adversary can enter into the system. The adversary always tries to identify PoA to attack the system. Point of attack represents like a loophole through which the adversary tries to interact with the system. A software has many points of attack like login page, sign up page, data entry page etc. Adversary tries to identify these points so he can potentially harm the system's confidentiality, integrity, and availability.

B. Point of Belief (PoB)

Point of belief shows the belief on external entities of the software system. The software system provides an access right to the external entities. These entities are known as the point of belief. Several entities can be the point of belief for a software product like the anonymous user who visit the web-based application, authorized users who have valid credentials, admin of software product, database server administrator etc. The security requirement engineer must identify all the point of belief while analyzing a security attack.

C. Point of Conjecture (PoC)

Point of conjecture shows the hypothesis about security issues which may be faced by the software product in the future after the development. The Point of Conjecture should be identified before the implementation phase has started and these points are very much significant in eliciting security requirements so this should not be despoiled. Point of Conjecture should be validated after the development of the software product.

D. Point of Dependency (PoD)

Point of dependency shows the software dependency on external entities. For a software system, Point of Dependency could be the reason for potential harm. The PoD should be valid and authorized because inconsistency can show the way to a security attack. A software product can have several points of dependency like the Web server on which the web-based application depends, the database server, network connection etc.

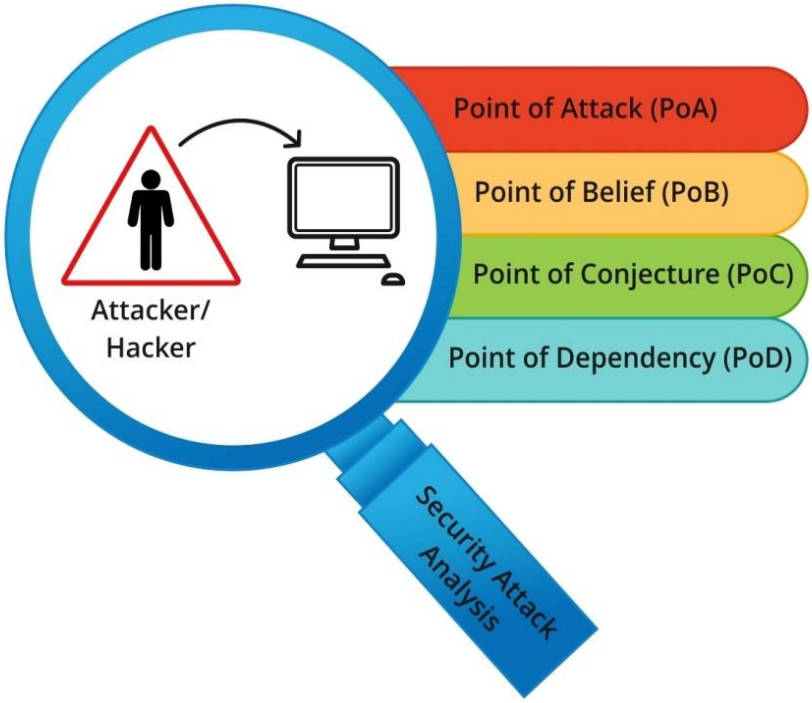

Figure 4.3 Four Points of Security Attack Analysis

4.3.6 Threat Identification and Categorization

After identifying the four points PoA, PoB, PoC, PoD of security attack analysis in the previous step, it will be easy to identify the threats. This step identifies all potential threats to the software system which is to be developed. The threat is considered as a source of possible attacks to a software development and vulnerabilities are a weakness in the design of software. There are numerous potential threats that may harm the software application. Although it is not practical to identify all potential threats to a software development, all probable threats should be considered. There are a number of

methods of identifying threats such as reviews of attack histories, reviews of current headlines involving data breaches, reviews of internet sites, Threat modeling. Threat modeling is an effective and best way to identify threats and vulnerabilities [92]. Threat modeling is a technique used to look at the potential attacks that can be applied to a given software development by breaking down the software into its most basic components. The researcher has constructed a threat dictionary that contains a collection of previously identified threats with corresponding most appropriate security requirements. The main intention is to help the security requirement engineer to elicit security requirements by accessing the threat dictionary knowledge. This step of STORE methodology also categorizes each threat through Microsoft's STRIDE model. STRIDE [93] is a classification methodology for potential threats based on the threat categories like Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, and Elevation of privilege. The following Table 4.2 discusses the Microsoft's STRIDE Threat classification.

Table 4.2 Microsoft's STRIDE Threat classification system

Acronym	Full form	Description
S	Spoofing	Spoofing threat allows an unauthorized user to become like an authorized user. In this category of threat, the attacker gets the log-in credentials of the authorized user through hacking techniques such as shoulder surfing, keystroke logging etc.
T	Tampering	Tampering threat allows the attacker to modify the data within the software system to get the intentional malicious goal.
R	Repudiation	This type of threat allows the attacker to claim that they didn't perform the malicious activity because the software system does not have sufficient evidence to prove otherwise.
I	Information disclosure	In this type of threat, the attacker revealed the protected data, such as login credentials, credit/debit card number etc. to unauthorized users.

D	Denial of service	This type of threat occurs when an attacker can prevent legitimate users from using the general functionality of the software system.
E	Elevation of privilege	This type of threat occurs when the attackers are able to gain additional rights and privileges for that they are not eligible.

4.3.7 Risk Evaluation and Prioritization

The next step of STORE methodology is risk evaluation for identifying potential threats. This step evaluated the impact of threats on the software application. There are many risk assessment approach for assessing threats during the software development process. Mitigation of all the identified threats may not be necessary and economically feasible. The identified threats are ranked with their potential from the highest to lowest risk. The lowest rank threats may be ignored because of their low potential to harm the software system. A typical risk probability formula given below used in industry shows the risk and consequence of a particular vulnerability as equal to the probability of a threat occurring multiplied by the damage potential.

Risk= Probability x Damage Potential

This formula measures the probability and damage potential on a 10 scale, where 1 represents the least likely to occur and 10 represents the most likely to occur. According to this formula if the probability of a threat occurring is 5 and damage potential of threat is 10 then the risk will be 50%. Risk evaluation can be done by Microsoft's DREAD risk assessment model (Swiderski & Snyder, 2004).

DREAD_Risk_Value = (DAMAGE + REPRODUCIBILITY + EXPLOITABILITY + AFFECTED USERS + DISCOVERABILITY) / 5

Another risk assessment methodology is OCTAVE (Operationally Critical Threat, Asset, and Vulnerability Evaluation) is a very complex risk assessment approach developed by the SEI of CMU. Common Vulnerability Scoring System (CVSS) can also be used to rate the risk by the vulnerability. The Common Vulnerability Scoring System provides an open framework for communicating the characteristics and impacts of IT vulnerabilities [94]. The Risk Evaluation and Prioritization step of STORE

methodology calculate the impact of threats to assets and prioritize the threats with an inclusive evaluation about the degree of risk to an asset.

4.3.8 Security Requirements Elicitation

The next step of STORE methodology is to elicit effective and efficient security requirements from the previous output of STORE methodology. The STORE methodology first considers the threats with higher risk, then the average risk potential threats and in last the low potential threats. The researcher has proposed a threat dictionary that contains a collection of previously identified threats with corresponding most appropriate security requirements. The main intention is to help the security requirement engineer to elicit security requirements by accessing the threat dictionary knowledge. This step of STORE methodology elicited appropriate security requirements for each identified threat.

4.3.9 Security Requirements Validation

After the successful elicitation of security requirements in the previous step of STORE methodology, we get efficient and effective security requirement for the mitigation of each potential threat. Threat mitigation is the security feature or assurance techniques for mitigating specific security threats. Unmitigated security threats are vulnerabilities that can be exploited by the attackers to harm the system [95]. Every threat is not potentially harmful to the software system. Therefore, mitigation of every threat is not necessary.

4.3.10 Security Requirements Specification Document

The last step of STORE methodology is to create the SRS document. The security requirements specifications are made and they are validated with the stakeholders. This security requirement specification report will be submitted by the requirement engineer to the software development team for early integration of security in software development. Finkelstein and Fuks [96] advocate that the construction of the specification document by taking several stakeholder viewpoints who have different

views may be collected through a categorical dialogue that records responses in the form of statements, queries, rejections, challenges, etc.

4.4 Conclusion

In this chapter, we have described our proposed Security Threat Oriented Requirement Engineering (STORE) methodology for complete and well-organized elicitation of security requirements. The STORE methodology overcomes the several limitations of other existing approaches like they don't categorize or prioritize the potential threats, lack of coding standards, process planning and unorganized documentation of security requirements etc. of several other security requirements elicitation approaches. This may lead to complexity of adopting the other framework. The existing methodology follows the iterative process which may add advantage to find new security requirements but this will lead to more complexity to the web applications. The simplicity of the STORE methodology makes it usable by technical persons and developers who are not expert in the software security field. STORE methodology integrates security with the requirements engineering process based on threats. STORE methodology identifies and priorities the potential threats and eliciting the most appropriate security requirements in a systematic way. The identification and analysis of four points of security attack PoA, PoB, PoC, PoD makes the identification of threats easier. We have constructed a threat dictionary that contains a collection of previously identified threats with corresponding most appropriate security requirements. The main intention is to help the security requirement engineer to elicit security requirements by accessing the threat dictionary knowledge.

Chapter 5 Case Study Evaluation

"It takes 20 years to build a reputation and few minutes of cyber-incident to ruin it."

- ***Stephane Nappo***

5.1 Background

In this chapter, the researcher has implemented the STORE methodology on college ERP system for case study evaluation. An Enterprise Resource Planning (ERP) system is a useful tool that educational institutions, business organizations are turning to, in order to build strong capabilities, improve performance, undertake better decision making, and achieve a competitive advantage. The ERP system aims to integrate all key business activities through improved relationships at all levels to achieve a competitive advantage [97]. This system is an effort to build an combined product that manages the majority of processes in a educational or business organization. ERP system is defined by Scott [70] as: "A suite of integrated corporate wide software applications that drives manufacturing, financial, distribution, HR, and other business functions in a real time environment".

With the increasing need of ERP system for processing large amounts of data stored in different places and structured in many different formats as the order of the day, the necessity for seamless integration of all IT functions becomes inevitable. ERP system is an integrated flow of information to support businesses becomes critical not only for gaining competitive advantage but also for ensuring the survival of already existing systems. With enterprises becoming bigger and bigger, they become more and more dependent on automation in the form of IT. While in the recent past the database system

was the core of the IT, with hundreds of application programs situated around these databases, nowadays enterprises use ERP systems for their IT problems. Security is critical for ERP systems, as they are used in numerous industries including defence, intelligence, medical, educational and financial [111]. First, we need to elicit all possible security requirements in the development process of the ERP systems.
The researcher has implemented the STORE methodology on a case study of college ERP system to elicit complete, accurate and well-structured set of security requirements for the development of trustworthy and quality software.

5.2 Case Study of ERP System

The proposed STORE methodology applied to the Enterprise resource planning (ERP) web-based application software. This college ERP system is capable of managing student records, department, faculties, library, and other information. The college ERP system has all the information about the students, faculties, staff, library, departments and other confidential information which requires security. The proposed methodology was applied to the college ERP system before the development of this software product so that the requirement engineer can easily elicit all the security requirements needed for this software product. Eliciting the entire security requirement before the development of a software product is not an easy task, but the proposed methodology makes it easy for the requirement engineer. Several literature studies have suggested several significant success factors in ERP implementation [98-103]. For the successful implementation of an ERP system requirement must be created and the system requirement should be documented in a proper manner [104]. According to Nah and Lau [98], an ERP system has the ability to automate and integrate an organization's business processes, share common data and practices across the entire enterprise and produce and access information in a real-time environment. An ERP system can be highly beneficial to an organization if the organization is able to overcome the implementation hurdles. ERP systems cannot remain inside organizational boundaries. Technically, ERP developers are preferably considering the browser/Web server architecture over traditional client/server in order to deliver e-business capabilities. Hence, for their first-generation e-business solutions almost all big ERP vendors are using a mixed Java/XML strategy [104].

5.3 Execution of the Case Study

A case study of an ERP system presented here shows the support of the STORE methodology for security requirements engineering. Execution of the case study for the validation of a new methodology is a very common and complex task in the software engineering field. All the sequential steps of STORE methodology have been followed in the following section for the effective and efficient security requirements engineering.

5.3.1 Identify System Goals

Identify system goals is the first step of the proposed STORE methodology. In this step, the requirement engineer collects all the system goals for the software system which is to be developed. Requirement engineer can use several techniques to identify and collect system goals like interviews, brainstorming, etc. For the college ERP system following can be system goals. The following Table 5.1 shows the identified system goals for the development of a college ERP system.

Table 5.1 Identified system goals for college ERP system

Goal ID	Description
G1	The college ERP system will be installed on a web server that has been secured to protect confidential information. All security patches for the web server must be enabled.
G2	The college ERP system will also be installed on a database server that has been secured. All security patches for the database server must be enabled.
G3	The database server must be protected from the direct access from the internet by a firewall.
G4	The Web server should be protected from direct access from the internet by a firewall.
G5	Only HTTP and HTTPS ports allowed direct access from the internet.
G6	Communication between the web server and database server should

	be conducted over a private network.
G7	The college ERP system should be deployed over HTTPS.

5.3.2 Identify and Prioritize Stakeholders

This step of STORE methodology, identify and prioritize the internal and external stakeholders concerned at different levels of ERP system development. The following Table 5.2 contains the list of several potential stakeholders for the development of college ERP system.

Table 5.2 List of possible stakeholders with their significance for college ERP system

No.	Name	Significance	Type
1	President	Critical	Managerial
2	Director	Critical	
3	Senior Executives	Major	
4	Internal Auditor	Critical	Marketing
5	Purchasing Manager	Critical	
6	Key users	Major	
7	End users	Critical	
8	ERP Project manager	Critical	Information System
9	Database administrator	Critical	
10	Developer	Critical	
11	Networking team	Major	
12	Assistant	Minor	User
13	Students	Minor	
14	Faculty	Minor	

5.3.3 Agreed upon Goals

After identifying all the stakeholders related to the ERP system project the next step of STORE methodology is to make all the stakeholders agreed upon the proposed system goals. The requirement engineers and stakeholder both must be agreed on the identified

goals of the software product for the development. This will create a perfect environment for the development of a software product in which every stakeholder can communicate with each other clearly.

5.3.4 Asset Identification

The STORE methodology is used to protect the assets from threats. This step is an important step of the STORE methodology as this step identifies the assets related to a particular software system. The following Table 5.3 shows the list of identified valuable assets for college ERP system.

Table 5.3 Identified assets of the college ERP system

Asset ID	Name	Description
A1	Student, staff, and admin	An asset that relates to a student, staff or admin.
A2	Student's login data	The student's credentials: username and password.
A3	Staff login data	The staff's credentials: username and password.
A4	Admin login data	The admin's credentials: username and password.
A5	Student's personal data	The personal data that the student enters, such as student record and assets.
A6	Staff's personal data	The personal data that the staff enters, such as staff record and assets
A7	System	Assets that relate to the essential system.
A8	Availability of ERP System	If the college ERP system goes down, student /and staff cannot request or receive quotes.
A9	Process	Assets that relate to the process of running the web application
A10	Application	Assets that relate to the web application
A11	Login Session	The web session associated with a logged in student, staff or admin.

A12	Backend database access	The ability to interact with the database that stores, student's data, staff data, and login credentials.
A13	Student fee details	The student's fee record must be secure. Tampering with this data could cause the loss of college.
A14	Staff salary details	The staff salary record must be secure. Tampering with this data could cause the loss of college.
A15	Message Notification	The message notification contains the information for students and staff.
A16	Audit data	Attackers might try to attack the system without being logged or audited.
A17	Access to the record	Only authorized people should be able to view his/her record.

5.3.5 Security Attack Analysis

After identifying all the valuable assets, the next step of STORE methodology is to analyze all the sources of security attacks to the college ERP system. Analyzing potential security attacks on ERP software system under development is an important step in engineering secure software systems, as the identified security attacks will help in identifying important security threats. The following four points are essential for security attack analysis.

A. Point of Attack (PoA)

The following Table 5.4 shows the identified Point of Attack (PoA) for the college ERP system in a security attack analysis process of STORE methodology.

Table 5.4 List of identified Point of Attack (PoA)

ID	Point of Attack	Description
PA1	Web Server Listening Port (HTTPS)	The port (HTTPS) that the web server listens on.

PA2	Login Page	Page for students or staff to create a login and perform a login to the site to begin requesting or reviewing records.
PA3	CreateLogin function	Creates a new student or staff login (Admin login must be created directly through the database stored procedures.)
PA4	LoginToSite function	Compares authorized person credentials to those in the database and if credentials match, create a new session.
PA5 **PA6**	Data entry page RetrieveData function	Page used to enter student or staff personal data into the database so that the admin can review it. Allow the authorized person to view his/her records from the database.
PA7	SubmitData function	Submits student or staff data to be reviewed by the admin
PA8	Admin Review page	This page used by the admin to review the student or staff request.
PA9	RetrieveData function	Retrieves student or staff data.
PA10	SubmitData function	Submits any information for the student or staff.
PA11	ListRequests function	Lists requests ready for review.
PA12	Database Listening Port	Enables the database to be used remotely by the authorized persons.
PA13 **PA14**	Database stored procedures CreateLogin procedure	Store and retrieve records in the database. Create a login for the authorized person.
PA15	RemoveLogin procedure	Logout from the college ERP system.

PA16	StoreUserData procedure	Used to store user data from the data entry page of the ERP system.
PA17	RetrieveUserData procedure	Retrieves the user's data and request.

B. Point of Belief (PoB)

The following Table 5.5 shows the identified Point of Belief (PoB) for the college ERP system in a security attack analysis process of STORE methodology.

Table 5.5 List of Identified Point of Belief (PoB)

ID	Point of Belief	Description
PB1	Unauthorized remote user	A user who has connected to the ERP system, but has not provided valid credentials yet.
PB2	Authorized remote user	An authorized user who has created an account and has valid login credentials.
PB3	Admin	Admin uses login credentials to access and modify the database.
PB4	HTTP user	A remote user that accesses a page via HTTP.
PB5	HTTPS user	A remote user that accesses a page via HTTPS.
PB6	Web server process identity	Used to authenticate the web server to the database when storing or retrieving information.
PB7	Database server process identity	The account that the database server process runs as, represented by its process token.

C. Point of Conjecture (PoC)

The following Table 5.6 shows the identified Point of Conjecture (PoC) for the college ERP system in a security attack analysis process of STORE methodology.

Table 5.6 List of identified Point of Conjecture (PoC)

ID	Point of Conjecture	Description
PC1	Online payment	The online payment system can be another function for this ERP system. If this functionality added, this function should not provide a way for attackers to attack existing security features.
PC2	Payment Gateway	If added payment gateway in future, ERP system must comply with PCI DSS or other security standards.
PC3	Encrypted Communication	If encrypted communication functionality is added to the ERP system in the future, message exchange should be completed according to standards.

D. Point of Dependency (PoD)

The following Table 5.7 shows the identified Point of Dependency (PoD) for the college ERP system in a security attack analysis process of STORE methodology.

Table 5.7 List of identified Point of Dependency (PoD)

ID	Point of Dependency	Description
PD1	Database Server	The ERP system depends on the security of the database server.
PD2	Web Server	The ERP system depends on the security of the web server.
PD3	Network	The ERP system depends on the security of the network between the web server and database server.
PD4	External SMTP	The ERP system depends on an

		external SMTP server to deliver any message.
PD5	Session Management	The ERP system depends on the session management of the web server being secure.

5.3.6 Threat Identification

After identifying the assets and analyzing the security attack with the help of STORE methodology, it is necessary to identify the potential threat for each asset. The following Table 5.8 consists of recognized threats to a college ERP system.

Table 5.8 Identified threats for the college ERP system

ID	Threat	Description	STRIDE						Mitigated	Assets
			S	T	R	I	D	E		
T1	Malicious SQL data in user input	The attacker might try to inject SQL commands into the application via Login.		✓				✓	No	A12
T2	Login Information Disclosure	The attacker gets the login credentials of the authorized user.				✓		✓	No	A2, A3, A4
T3	Session Id Theft	The attacker gets the session ID of another authorized user.						✓	No	A11
T4	User Data Disclosure	Disclosing another authorized user data raises privacy	✓			✓			No	A5, A6

		issues.								
T5	Access to the Database	The Attacker attacks to the database of the ERP system.		✓	✓	✓		✓	Yes	A1-A6
T6	Attack on Admin Login	The attacker performs as an admin of the ERP system.						✓	Yes	A4
T7	Blocking Message Notification	The attacker prevents an authorized user from receiving any notification.					✓		Yes	A15
T8	User Data Tampering	The attacker modifies the authorized person's data.	✓	✓				✓	No	A5, A6
T9	User Account Deletion	The attacker deletes an authorized user account.					✓	✓	Yes	A2, A3, A4
T10	Crashing the ERP system	The attacker crashes the ERP web application.					✓		Yes	A8
T11	Unauthorized access	The attacker access the ERP system without valid credentials.						✓	Yes	A5, A6
T12	Access without Login	The attacker access the information of authorized person without being logged.			✓				No	A16

5.3.7 Risk Evaluation and Prioritization

The Risk assessment and prioritization step of STORE methodology calculate the impact of threats to assets, and shows an inclusive evaluation of the degree of risk to an asset. The following Table 5.9 consists of the identified threats with their risk value.

Table 5.9 Prioritize threats to their DREAD risk value

Threat ID	Threat	DREAD Value	Mitigated
T1	Malicious SQL data in user input	10	No
T5	Access to the Database	10	Yes
T10	Crashing the ERP system	10	Yes
T4	User Data Disclosure	9.2	No
T8	User Data Tampering	9.2	No
T6	Attack on Admin Login	7.6	Yes
T9	User Account Deletion	7.6	Yes
T12	Access without Login	7.6	No
T2	Login Information Disclosure	6.6	No
T7	Blocking Notification	6.4	Yes
T11	Unauthorized access	5.2	Yes
T3	Session Id Theft	3.8	No

5.3.8 Security Requirement Elicitation

This step of STORE methodology elicits the appropriate security requirement for each identified potential threats. The following Table 5.10 consists of security requirements for each identified threat of the college ERP system.

Table 5.10 Elicited security requirements for each threat

Threat ID	Security Requirement ID	Security Requirement
T1	SR1	Use of prepared statements with parameterized queries.
T5	SR2	Use of Access control, Auditing, Authentication, Encryption, Integrity controls, Backups techniques
T10	SR3	Upgrade to the new version by fixing all identified flaws.
T4	SR4	Use of complex encryption methods that limits the risks of user data disclosure of ERP system.
T8	SR5	Use a firewall and proper authorization technique for granting the access right to use of the software system.
T6	SR6	Implement account lockout procedure, captcha and enforce the user of the ERP system to use strong passwords.
T9	SR7	Complex security password and account lockout should be used which locked the account after some failed login attempts.
T12	SR8	Use firewalls, VPN and SSL techniques.
T2	SR9	The database server of ERP system should be protected from the direct internet access by a firewall.
T7	SR10	Ensure the proper security of SMTP server.

T11	SR11	Implement two-factor authentication, i.e. strong password and one-time passcode.
T3	SR12	Use SSL/HTTPS encryption for the ERP system.

5.3.9 Security Requirements Validation

This step is related to the validation of recognized security requirements for college ERP system obtained in the previous step of the STORE methodology. All the identified security requirements must be capable to enhance the quality of the software system by enforcing appropriate security concern. In this step, the security requirements are prioritized based on the potential risk of the associated threat. The validation of the security requirements process consists of security requirement engineers and concern stakeholders as a participant. They use the review or walk-through approaches to validate the security requirements for the college ERP system.

5.3.10 Security Requirements Specification Document

Generating security requirement specification document is the last and crucial step of STORE methodology. All recognized and validated security requirement is documented in an organized manner for the college ERP system which is considered as a web-based application software. This security requirements specification document assists the developer to develop a more secure and quality ERP system which ensures security from the beginning of the software product.

5.4 Result and Discussion

This section describes the comparative analysis of the proposed STORE methodology with two different security requirements elicitation approaches that are similar to the proposed methodology. The researcher has compared STORE methodology with SQUARE [105] and MOSRE framework [106]. The researcher has chosen to compare with these security requirements engineering techniques because the proposed STORE methodology is motivated by such methodologies in the way of eliciting security

requirements for the software products. The researcher showed that the security requirements are more effective and efficient when following the sequential steps of the proposed STORE methodology**.**

5.4.1 Comparison with the SQUARE methodology

In this section, the researcher has applied the proposed STORE methodology in a case study presented by Gordon et al. [105]. They have applied the SQUARE methodology for eliciting security requirements on a case study of the Asset management system. They elicited total nine security requirements and after requirements prioritization, they selected five security requirements R01, R02, R06, R07, R08 out of nine as essential security requirements. The following Table 5.11 shows the comparison between SQUARE methodology and STORE methodology. The researcher has compared the results with the essential security requirements obtained by SQUARE methodology. The result shows that the STORE methodology covers more threats than SQUARE methodology and also elicits more effective and efficient security requirements. Therefore, STORE methodology elicits more complete security requirements for the asset management system.

Table 5.11 STORE methodology comparison results with the SQUARE methodology

SQUARE Methodology results	
Essential security requirements	
R01	The system is required to have strong authentication measures in place at all system gateways/entrance points.
R02	The system is required to have sufficient process-centric and logical means to govern which system elements (data, functionality, etc.) users can view, modify and/or interact with.
R06	It is required that the system's network communications be protected from unauthorized information gathering and/or eavesdropping by encryption and other reasonable techniques.
R07	It is a requirement that both process-centric and logical means be in place to prevent the installation of any software or device without prior authorization.

R08	It is required that the AMS's physical devices be protected against destruction, damage, theft, tampering or surreptitious replacement (including but not limited to damage due to vandalism, sabotage, terrorism or acts of God/Nature).
Results from the proposed STORE methodology	
Elicited Security Requirements	
SR1	Use of Access control, Auditing, Authentication, Encryption, Integrity controls, Backups techniques
SR2	Implement account lockout procedure, captcha and enforce the user of the ERP system to use strong passwords.
SR3	Use of complex encryption methods that limits the risks of user data disclosure of E-Health system.
SR4	Use a firewall and proper authorization technique for granting the access right to use of the software system.
SR5	Use HIPAA security standards and policy to ensure proper external security.

5.4.2 Comparison with the MOSRE Framework

The researcher has also applied the proposed STORE methodology in the case study presented by the Salini and Kanmani [106]. They have applied their proposed MOSRE framework for eliciting security requirements on a case study of the E-Health system. They applied MOSRE in the early stages of E-Health system development, to identify assets, threats, and vulnerabilities. The following Table 5.12 shows the comparison between MOSRE framework and STORE methodology. The researcher has compared the results with the system asset-based security requirements obtained by MOSRE framework. The result shows that the STORE methodology covers more threats than MOSRE framework and also elicits more effective and efficient security requirements for the E-Health system. Therefore, STORE methodology elicits more complete security requirements.

Table 5.12 STORE methodology comparison results with MOSRE framework

MOSRE framework results	
Results from System asset-based security requirements	
SR1	Use secure authentication, which does not send passwords over the network
SR2	Use secure communication channels
SR3	Use remote procedure call encryption
SR4	Firewall policies that block all traffic except expected communication ports
Results from the proposed STORE methodology	
Elicited Security Requirements	
SR1	Use of Access control, Auditing, Authentication, Encryption, Integrity controls, Backups techniques
SR2	Implement account lockout procedure, captcha and enforce the user of the E-Health system to use strong passwords.
SR3	Use of complex encryption methods that limits the risks of user data disclosure of E-Health system.
SR4	Use a firewall and proper authorization technique for granting the access right to use of the software system.

5.5 Conclusion

In this chapter the researcher has successfully executed the proposed STORE methodology by applying this approach for a case study of ERP system. The motivation for this is that security requirements engineering processes are often not performed by considering the view point of different stakeholders who have deep knowledge about the valuable assets to the system, and often without much contact between the stakeholders which results in failure of the system. It is therefore important to develop both the ability of the several stakeholders involved in the software development process to recognize potential security aspects, and the experiences of the development team to resolve these needs in practice through secure design. Hence, the researcher has designed and developed a security requirements engineering methodology that is easy to adopt, extra comprehensive and assist the requirement engineers to elicit effective and

efficient security requirements in a more organized manner. STORE methodology can be used for the web applications which consider information as treasure or assets and evaluated the strength of the methodology. In the early phases of STORE methodology, the researcher begins with the context of the system, the functional requirements, and the primary security goals and requirements. The researcher has successfully executed each phase of STORE methodology in order to get a complete set of security requirements for a quality and trustworthy development of college ERP software system.

The researcher has also compared the STORE methodology with two other appropriate methodologies to demonstrate the benefits of using the STORE methodology presented in this chapter. The researcher has compared STORE methodology with SQUARE [105] and MOSRE framework [106]. The researcher has chosen to compare with these security requirements engineering techniques because the proposed STORE methodology is motivated by such methodologies in the way of eliciting security requirements for the software products. The researcher has showed that the security requirements are more effective and efficient when following the sequential steps of the proposed STORE methodology. The STORE methodology approach presented here has not yet been validated on a big project. The next step is to deploy it on a real time project. The researcher want to ensure that our approach is practical, and a suitably accurate assessment requires a real project. In the future, the researcher has planned to apply our STORE methodology for many other software development projects. The other future work is to develop a tool for STORE methodology to elicit security requirements.

Chapter 6 Validation of Proposed Methodology

"When flimsy cyber defense fails, Format Preserving Encryption prevails"

- ***James Scott***

6.1 Background

This chapter validated the proposed STORE methodology. The validation of STORE methodology presented here was done by the security expert evaluation. This form of validation requires one to construct a systematic evaluation process where security expert's plays a role of assessing the methodology based on some pre-defined criteria. It is necessary for each security expert to understand and experience the operational mechanism of the proposed methodology before participating in the systematic evaluation process. They have to experience the functionality of proposed methodology through the different example problem to show how the contributions presented in this thesis help identify effective and efficient security requirements. The only precondition is that the example shows how the contributions presented in this thesis are used. The researcher uses this validation method in this thesis. Using a constructed example has one significant disadvantage: the way the example is constructed may mask problems with the contributions that real examples would make evident. To help minimize this risk and to provide us with a sanity check on our work, the researcher has published the contributions in several peer-reviewed. The criticisms received have helped enormously with filling in gaps in the contributions.

6.2 Evaluation and Validation of STORE Methodology

The evaluation of the STORE methodology is a vital process to validate and improve the applicability of the STORE methodology for the real software industry. The evaluation of STORE methodology was conducted in two phases. First, a self-review of methodology during design time and later post design with help of a systematic evaluation process. The following sub-section describes the evaluation of the STORE methodology using review during design time and by the help of systematic evaluation process.

6.2.1 Review during design time

The STORE methodology was developed to help the security requirements engineers and other security professionals of any academic and software industry to elicit security requirements in a more systematized way. Initially, the researcher reviews the STORE methodology and different steps of the methodology during design. The STORE methodology was finalized using security requirements from the software projects. In this process, the security requirements elicited for the development of software product were aligned to the categories of security requirements categorization. The main aim was to identify any missing, irrelevant or overlapping categories. This was done by analyzing if there was any security requirement which does not fit to the existing categories or if there are certain security requirements which stay fit in more than one category. The researcher also tested the methodology steps by executing it on case studies of several software products. The systems were selected from different application domain. To enhance the usability and usefulness of the methodology, the researcher conducted different analysis and brainstorming sessions during the design time of STORE methodology followed by different subsequent investigation rounds.

6.2.2 Systematic Evaluation Process

The systematic evaluation process of the STORE methodology was conducted in a systematic manner. The systematic evaluation procedure starts by the designing the structure of experiment. In this phase the structure of evaluation criteria was created. In second phase which is Pre-tryout the feedback questionnaires was presented to the ten

security experts. The security experts of the evaluation were selected from the educational institutions and software industries. The questionnaire form shown in Appendix B was given to the security experts, when they were considered to have the proficiency and acquaintance to answer questions on the STORE methodology performance. In the evaluation procedure, it is also specified for privacy and business contemplations that their associated organizations will not be discussed. The security experts then participated and completed the evaluation tasks in the Pre-tryout phase. After taking feedback from participants through Appendix B in Pre-tryout phase, the security experts were also requested to complete the questionnaires in Appendix C to evaluate any suggestion, correction or modification to the STORE methodology.

6.3 Systematic Evaluation Process

Another way of STORE methodology evaluation is systematic evaluation process which performed after the development of STORE methodology. The systematic evaluation procedure involves the participation of different security experts and security engineers. This process starts with the design of experiment phase. In this phase the structure of evaluation criteria was created. In second phase which is expert Pre-tryout the feedback questionnaires was presented to the ten security experts. The security experts of the evaluation were selected from the educational institutions and software industries. The questionnaire form shown in Appendix B was given to the security experts, when they were considered to have the proficiency and acquaintance to answer questions on the STORE methodology performance.

In the evaluation procedure, it is also specified for privacy and business contemplations that their associated officialdoms will not be discussed in this paper. The security experts then participated and completed the evaluation tasks in the expert Pre-tryout phase. After taking feedback from participants through Appendix B in Pre-tryout phase, the security experts were also requested to complete the questionnaires in Appendix B to evaluate any suggestion, correction or modification to the STORE methodology. The evaluation criteria were explained in the evaluation criteria section. The outcomes of the expert Pre-tryout were analyzed for identify weaknesses and revise the STORE methodology in the third phase which is review and revision. . Based on the security experts suggest some changes to aid the future improvement of the methodology. The

STORE methodology then modifies according to the received feedback from the security experts. The updated STORE methodology again presented to the security experts to analyze the changes made to the previous version in Tryout phase. Finally, the systematic evaluation process finalize on the basis of feedback received in the Tryout phase. The following Figure 6.1 shows the structure of systematic evaluation process.

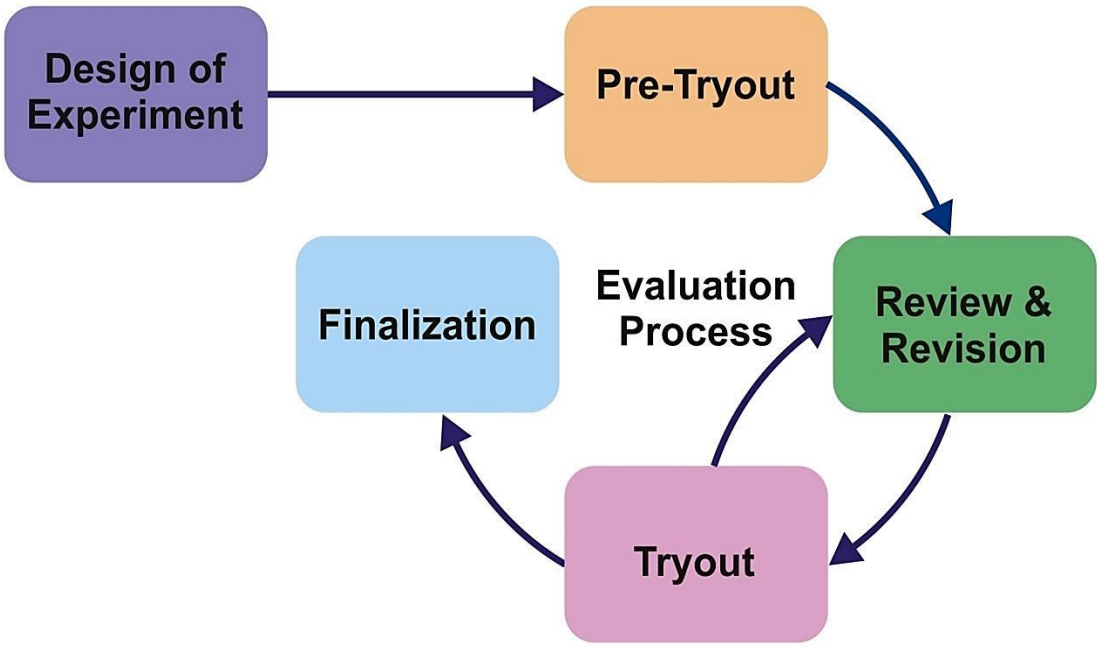

Figure 6.1 The systematic evaluation process diagram for STORE methodology

6.3.1 Design of Experiment:

The STORE methodology was discussed and properly analyzed by the security experts. The evaluation process was opened with the presentation of what security requirements mean, followed by the presentation describing the role of stakeholder and the impact of security threats on software application. Flaws and strengths of threat modeling were also discussed. The main aim of the STORE methodology is to elicit the effective and efficient security requirements early during the software development process. Owing to the informal nature of the evaluation process, there were discussions about the proposed solution. The salient aspects of the discussion were noted down and are documented

here as major points. The researcher has designed the experiment for evaluation based on some evaluation criteria. These evaluation criteria are the parameters on which several feedback questionnaires are asked by the security experts.

A. STORE Methodology Evaluation Criteria: The researcher has designed the questionnaire form presented in Appendix B for security expert feedback based on some criteria which are as follows:

- **Ease of Use:** The ease of use criterion evaluates the usability of the STORE methodology. This criterion involves the STORE methodology to have flexibility and be unmistakable because difficult methodology will require training and extra efforts.
- **Ease of Learning:** The ease of learning criterion evaluates user's learnability conferring to the results of the STORE methodology. The security requirements engineer should be able to learn the STORE methodology without any misinterpretation or complications to accomplish the effective and efficient security requirements.
- **Satisfaction:** The satisfaction criterion evaluates users' satisfaction conferring to the results of the STORE methodology. They should be able to satisfy with the functionality and performance of STORE methodology without any difficulties.

6.3.2 Pre-Tryout

The evaluation of STORE methodology was performed by taking feedback by ten security experts from different software organizations and institutions. All ten security experts have completed the feedback questionnaire forms to assess several features of the STORE methodology. The evaluation criteria were ease of use, ease of learning and satisfaction with STORE methodology. These were evaluated using quantitative measurement. In addition, descriptive questions were also provided to collect the security experts' analyses and recommendations for improving the STORE methodology. The evaluation results of the STORE methodology in Pre-Tryout phase are shown in Table 6.1, Table 6.2, Table 6.3 and Table 6.4.

Table 6.1 Ease of Use evaluation result of STORE Methodology in Pre-Tryout phase

Ease of Use

		Q1	Q2	Q3	Q4	Q5	Q6	Q7
N	Valid	10	10	10	10	10	10	10
	Missing	0	0	0	0	0	0	0
Mean		3.8000	3.8000	3.8000	3.9000	3.9000	3.8000	3.7000
Std. Error of Mean		.13333	.13333	.13333	.10000	.10000	.13333	.15275
Median		4.0000	4.0000	4.0000	4.0000	4.0000	4.0000	4.0000
Mode		4.00	4.00	4.00	4.00	4.00	4.00	4.00
Std. Deviation		.42164	.42164	.42164	.31623	.31623	.42164	.48305
Variance		.178	.178	.178	.100	.100	.178	.233
Skewness		-1.779	-1.779	-1.779	-3.162	-3.162	-1.779	-1.035
Std. Error of Skewness		.687	.687	.687	.687	.687	.687	.687
Kurtosis		1.406	1.406	1.406	10.000	10.000	1.406	-1.224
Std. Error of Kurtosis		1.334	1.334	1.334	1.334	1.334	1.334	1.334
Range		1.00	1.00	1.00	1.00	1.00	1.00	1.00
Minimum		3.00	3.00	3.00	3.00	3.00	3.00	3.00
Maximum		4.00	4.00	4.00	4.00	4.00	4.00	4.00
Sum		38.00	38.00	38.00	39.00	39.00	38.00	37.00
Percentiles	25	3.7500	3.7500	3.7500	4.0000	4.0000	3.0000	3.0000
	50	4.0000	4.0000	4.0000	4.0000	4.0000	4.0000	4.0000
	75	4.0000	4.0000	4.0000	4.0000	4.0000	4.0000	4.0000

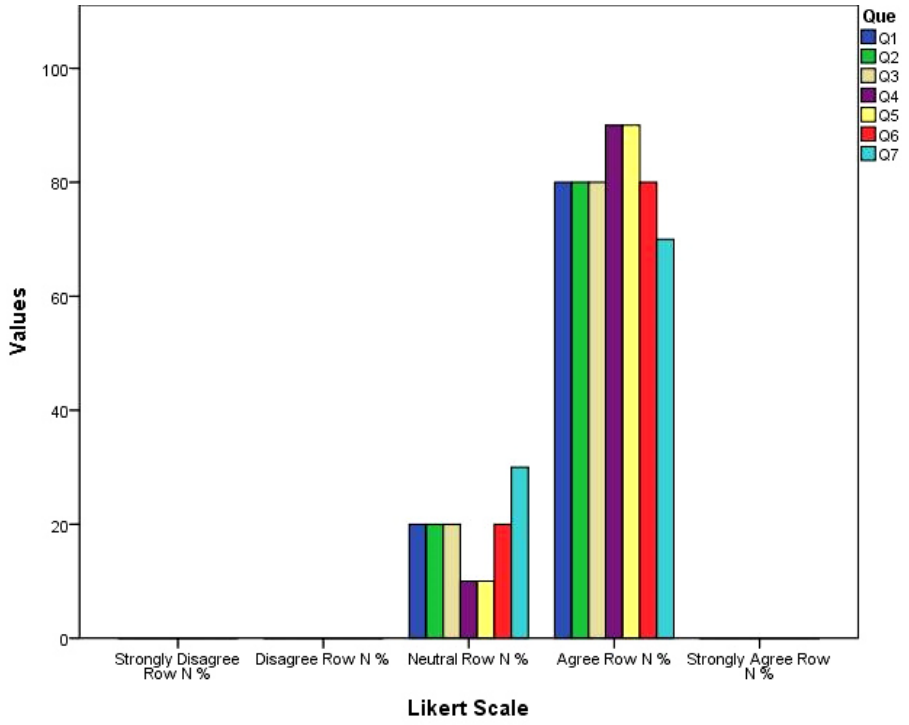

Figure 6.2 Bar graph of Ease of Use feedback in Pre-tryout phase

Table 6.2 Ease of Learning evaluation result of STORE Methodology in Pre-Tryout phase

Ease of Learning

		Q8	Q9	Q10	Q11	Q12	Q13
N	Valid	10	10	10	10	10	10
	Missing	0	0	0	0	0	0
Mean		3.9000	3.9000	3.9000	3.8000	3.8000	3.9000
Std. Error of Mean		.10000	.10000	.10000	.13333	.13333	.10000
Median		4.0000	4.0000	4.0000	4.0000	4.0000	4.0000
Mode		4.00	4.00	4.00	4.00	4.00	4.00
Std. Deviation		.31623	.31623	.31623	.42164	.42164	.31623
Variance		.100	.100	.100	.178	.178	.100

Skewness		-3.162	-3.162	-3.162	-1.779	-1.779	-3.162
Std. Error of Skewness		.687	.687	.687	.687	.687	.687
Kurtosis		10.000	10.000	10.000	1.406	1.406	10.000
Std. Error of Kurtosis		1.334	1.334	1.334	1.334	1.334	1.334
Range		1.00	1.00	1.00	1.00	1.00	1.00
Minimum		3.00	3.00	3.00	3.00	3.00	3.00
Maximum		4.00	4.00	4.00	4.00	4.00	4.00
Sum		39.00	39.00	39.00	38.00	38.00	39.00
Percentiles	25	4.0000	4.0000	4.0000	3.7500	3.7500	4.0000
	50	4.0000	4.0000	4.0000	4.0000	4.0000	4.0000
	75	4.0000	4.0000	4.0000	4.0000	4.0000	4.0000

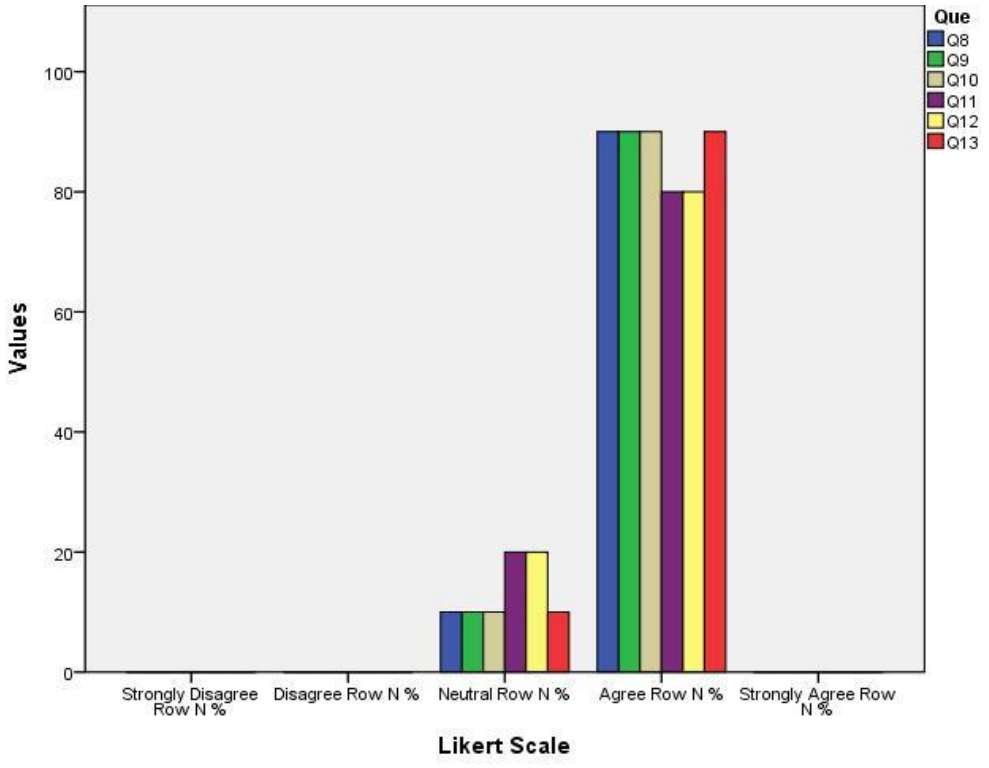

Figure 6.3 Bar graph of Ease of Learning feedback in Pre-tryout phase

Table 6.3 User satisfaction evaluation result of STORE Methodology in Pre-Tryout phase

User Satisfaction

		Q14	Q15	Q16	Q17	Q18
N	Valid	10	10	10	10	10
	Missing	0	0	0	0	0
Mean		3.8000	3.9000	3.9000	3.9000	3.9000
Std. Error of Mean		.13333	.10000	.10000	.10000	.10000
Median		4.0000	4.0000	4.0000	4.0000	4.0000
Mode		4.00	4.00	4.00	4.00	4.00
Std. Deviation		.42164	.31623	.31623	.31623	.31623
Variance		.178	.100	.100	.100	.100
Skewness		-1.779	-3.162	-3.162	-3.162	-3.162
Std. Error of Skewness		.687	.687	.687	.687	.687
Kurtosis		1.406	10.000	10.000	10.000	10.000
Std. Error of Kurtosis		1.334	1.334	1.334	1.334	1.334
Range		1.00	1.00	1.00	1.00	1.00
Minimum		3.00	3.00	3.00	3.00	3.00
Maximum		4.00	4.00	4.00	4.00	4.00
Sum		38.00	39.00	39.00	39.00	39.00
Percentiles	25	3.7500	4.0000	4.0000	4.0000	4.0000
	50	4.0000	4.0000	4.0000	4.0000	4.0000
	75	4.0000	4.0000	4.0000	4.0000	4.0000

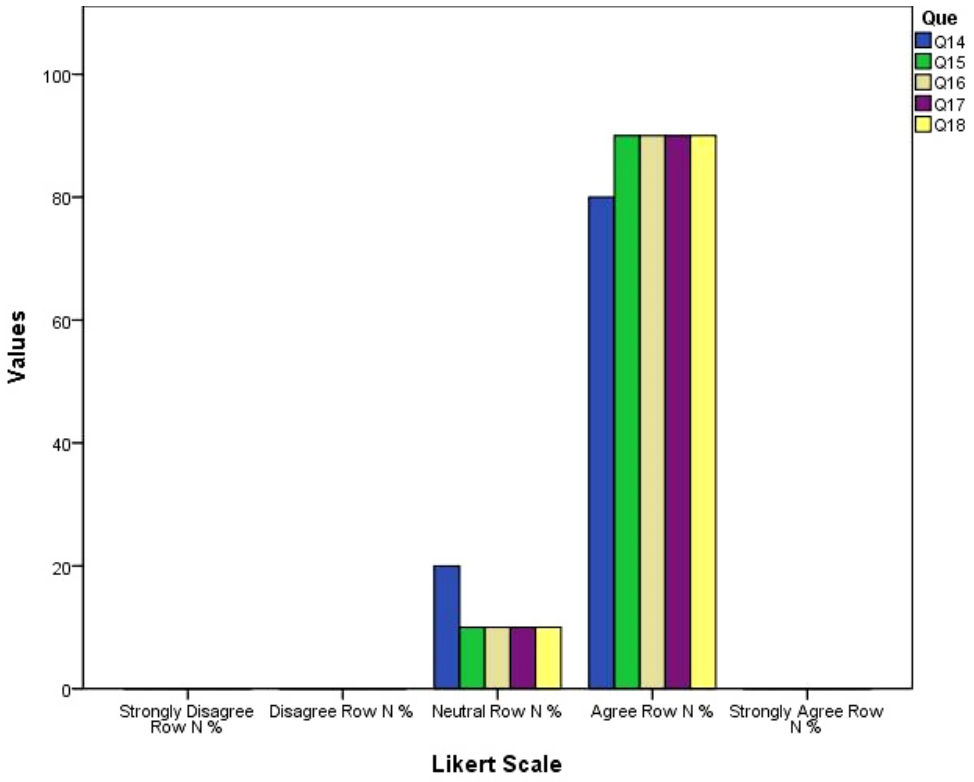

Figure 6.4 Bar graph of User satisfaction feedback in Pre-tryout phase

Table 6.4 Descriptive feedback result from the security experts

Questions	**Response**										**Result**
	SE 1	**SE 2**	**SE 3**	**SE 4**	**SE 5**	**SE 6**	**SE 7**	**SE 8**	**SE 9**	**SE 10**	
Do you suggest any correction or enhancement to the STORE Methodology?	No	No	No	Yes	No	No	Yes	No	No	No	Positive
Do you suggest any new step for the STORE methodology in the future and if so, provide the reason?	No	No	No	No	No	No	No	No	No	No	Very Positive
What is your opinion on the security attack analysis step?	No	No	No	No	No	No	No	No	No	No	Very Positive

Any other comment	No	No	No	No	No	No	No	No	No	No	Very Positive

6.3.3 Review and Revision

In this section the researcher reviewed the outcome taken from the previous section Pre-tryout and made some modification to the STORE methodology. In previous section, the security experts were asked to evaluate the ease of use, ease of learning and satisfaction aspect. After analysing the Table 6.1, all ten security experts positively agreed that the STORE methodology is clear and easy to use. However, a small training is still essential to recognize how to use STORE methodology accurately. Second, ease of learning was evaluated. This criterion evaluates the security requirements engineers' learnability conferring to the results of the STORE methodology. As Table 6.2 shows, all ten security experts agreed that the STORE methodology is easy to learn. Third, satisfaction with STORE methodology was evaluated by all ten security experts, as shown in Table 6.3 the security experts were satisfied with the ability of the STORE methodology to identify the effective and efficient security requirements, which no needs further improvement. They agreed that the functioning of STORE methodology is suitable and clear. They were also interested in utilizing the STORE methodology in their software organization.

The various levels and the distribution of the security requirements engineering steps cause no confusion or ambiguity. Based on their evaluation outcomes, STORE methodology can be utilized to elicit effective and efficient security requirements for software products which is to be developed in software development organizations. The researcher also received suggestions feedback response in Appendix C from all ten security experts in a description form. As Table 6.4 shows security expert 4 and 7 suggested some modification for STORE methodology. They want to expand the knowledge of threat dictionary as several new threats are increasingly attack our systems. They also suggest implementing the STORE methodology on big software development projects.

Finally, a group of ten senior researchers and security experts Yevseiev et. al. [107] from different security research organization in Ukraine have also evaluated STORE

methodology in their research paper. They find out that the STORE methodology provides procedural principles for security in corporate networks using standard system platforms and operating systems in the context of increasing potential risk for the security of critical infrastructure. They also pointed out that the STORE methodology practical solutions do not take into account then complexion of modern threats, not only among themselves, but also with the use of social engineering methods. This also pointed out that STORE methodology should allow timely response to the modification of threats.

After analysing the outcome of critical feedback from several respondents the researcher then made some modifications to the STORE methodology. The modification to the STORE methodology were planned to guarantee higher usability of this security requirements engineering approach in the software organizations. The changes were related to the identification and modification of modern complex threats. The researcher has constructed a modified threat dictionary that contains a collection of previously identified threats with corresponding most appropriate security requirements. This threat dictionary is based on enigma software threat database which has currently 18,012 threats listed on the threat database. The main intention is to help the security requirement engineer to elicit more effective security requirements by accessing the recent complex threat dictionary knowledge. This was motivated by the outcome of Ukrainian software security researchers that showed high interest in securing the software system from newly complex threats. The threat dictionary knowledge is updated on a continual basis to provide the security requirements engineers with information specific to threats affecting their software systems.

6.3.4 Tryout

After modifying STORE methodology according to the pre-tryout feedback result in review and revision phase of evaluation process, the researcher conducted again evaluation process with security experts to evaluate the modified STORE methodology. In Tryout phase we have taken feedback from 20 security experts in which 10 participant from the pre-tryout phase and 10 new participants. They develop several education systems such as a ticket booking system, college ERP system, student registration system, and library management system. All participants are security expert

from academic and software industries. They also have more than 8 years' experience in security engineering research or software development fields.

They were able to understand the purpose and the importance of the STORE methodology. Similar to the Pre-tryout phase, the researcher introduced the STORE methodology in the evaluation process with the security experts. The questionnaire similar to Pre-tryout was conducted with security experts and the respondent's answers were collected. The questionnaire required an assessment of STORE methodology in terms of ease of use, ease of learning, user satisfaction and a feedback form. Once the respondent had completed the questionnaire, their answers were extracted and analyzed based on the defined evaluations.

The following information describes the outcomes of the Tryout phase. In Tryout phase the researcher have accomplished a high score in the entire questionnaire criterion for the evaluation of STORE methodology. The evaluation results of the STORE methodology are shown in Table 6.5, Table 6.6, and Table 6.7. The entire respondent from different software development organization strongly agreed that the STORE methodology is clear and easy to use, easy to learn, and satisfactory for the elicitation of security requirements. They agreed with the procedures of the STORE methodology, including the different levels of analysis and that the elicitation of the effective security requirements was clear or explicit. STORE methodology can be utilized to elicit effectively security-critical requirements in the software development organizations. Finally they left no recommendation or modification to the STORE methodology.

Table 6.5 Ease of Use evaluation result of STORE Methodology in Tryout phase

Ease of Use

		Q1	Q2	Q3	Q4	Q5	Q6	Q7
N	Valid	20	20	20	20	20	20	20
	Missing	0	0	0	0	0	0	0
Mean		4.6500	4.6000	4.5500	4.4500	4.4000	4.3500	4.6500
Std. Error of Mean		.10942	.11239	.11413	.11413	.11239	.10942	.10942
Median		5.0000	5.0000	5.0000	4.0000	4.0000	4.0000	5.0000

Mode		5.00	5.00	5.00	4.00	4.00	4.00	5.00
Std. Deviation		.48936	.50262	.51042	.51042	.50262	.48936	.48936
Variance		.239	.253	.261	.261	.253	.239	.239
Skewness		-.681	-.442	-.218	.218	.442	.681	-.681
Std. Error of Skewness		.512	.512	.512	.512	.512	.512	.512
Kurtosis		-1.719	-2.018	-2.183	-2.183	-2.018	-1.719	-1.719
Std. Error of Kurtosis		.992	.992	.992	.992	.992	.992	.992
Range		1.00	1.00	1.00	1.00	1.00	1.00	1.00
Minimum		4.00	4.00	4.00	4.00	4.00	4.00	4.00
Maximum		5.00	5.00	5.00	5.00	5.00	5.00	5.00
Sum		93.00	92.00	91.00	89.00	88.00	87.00	93.00
Percentiles	25	4.0000	4.0000	4.0000	4.0000	4.0000	4.0000	4.0000
	50	5.0000	5.0000	5.0000	4.0000	4.0000	5.0000	5.0000
	75	5.0000	5.0000	5.0000	5.0000	5.0000	5.0000	5.0000

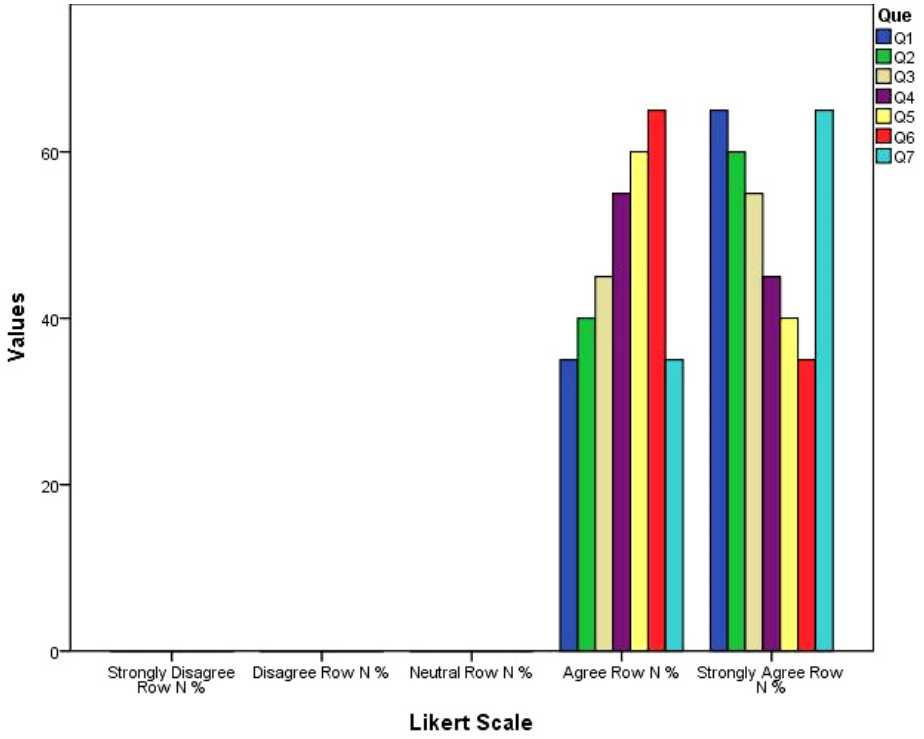

Figure 6.5 Bar graph of Ease of Use feedback in Tryout phase

Table 6.6 Ease of Learning evaluation result of STORE Methodology in Tryout phase

Ease of Learning

		Q8	Q9	Q10	Q11	Q12	Q13
N	Valid	20	20	20	20	20	20
	Missing	0	0	0	0	0	0
Mean		4.6000	4.4500	4.5500	4.4000	4.4000	4.6500
Std. Error of Mean		.11239	.11413	.11413	.11239	.11239	.10942
Median		5.0000	4.0000	5.0000	4.0000	4.0000	5.0000
Mode		5.00	4.00	5.00	4.00	4.00	5.00
Std. Deviation		.50262	.51042	.51042	.50262	.50262	.48936
Variance		.253	.261	.261	.253	.253	.239
Skewness		-.442	.218	-.218	.442	.442	-.681

Std. Error of Skewness		.512	.512	.512	.512	.512	.512
Kurtosis		-2.018	-2.183	-2.183	-2.018	-2.018	-1.719
Std. Error of Kurtosis		.992	.992	.992	.992	.992	.992
Range		1.00	1.00	1.00	1.00	1.00	1.00
Minimum		4.00	4.00	4.00	4.00	4.00	4.00
Maximum		5.00	5.00	5.00	5.00	5.00	5.00
Sum		92.00	89.00	91.00	88.00	88.00	93.00
Percentiles	25	4.0000	4.0000	4.0000	4.0000	4.0000	4.0000
	50	5.0000	4.0000	5.0000	4.0000	4.0000	5.0000
	75	5.0000	5.0000	5.0000	5.0000	5.0000	5.0000

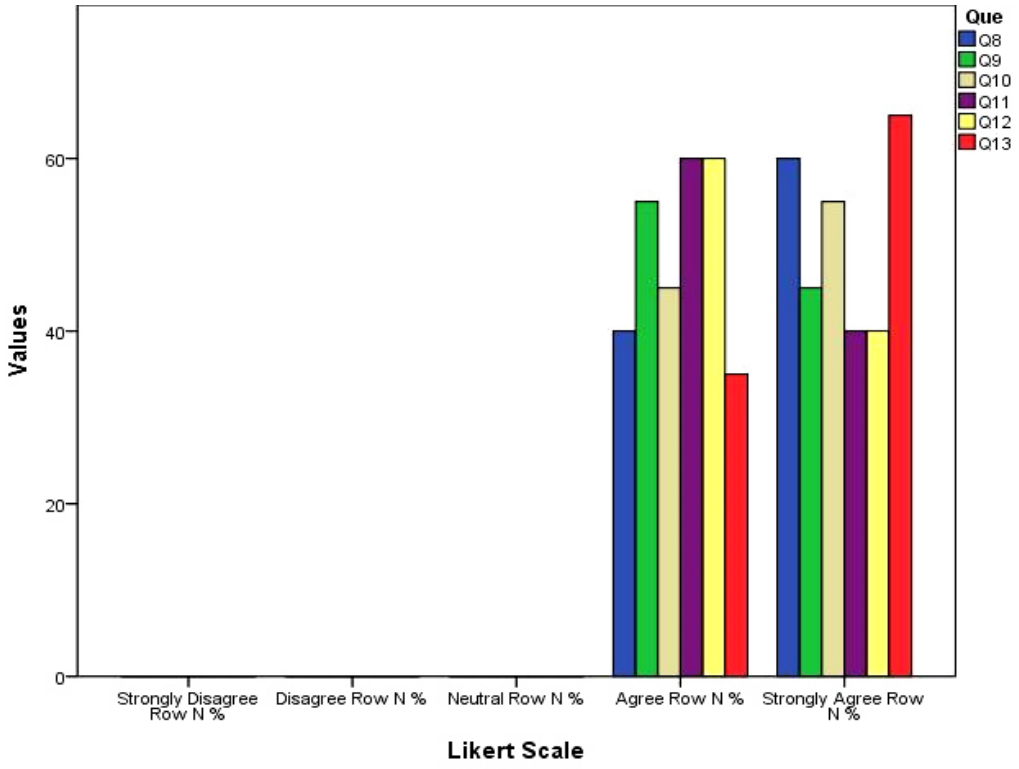

Figure 6.6 Bar graph of Ease of learning Feedback in Tryout phase

Table 6.7 User Satisfaction evaluation result of STORE Methodology in Tryout phase

User Satisfaction

		Q14	Q15	Q16	Q17	Q18
N	Valid	20	20	20	20	20
	Missing	0	0	0	0	0
Mean		4.5500	4.6000	4.3500	4.4500	4.7500
Std. Error of Mean		.11413	.11239	.10942	.11413	.09934
Median		5.0000	5.0000	4.0000	4.0000	5.0000
Mode		5.00	5.00	4.00	4.00	5.00
Std. Deviation		.51042	.50262	.48936	.51042	.44426
Variance		.261	.253	.239	.261	.197
Skewness		-.218	-.442	.681	.218	-1.251
Std. Error of Skewness		.512	.512	.512	.512	.512
Kurtosis		-2.183	-2.018	-1.719	-2.183	-.497
Std. Error of Kurtosis		.992	.992	.992	.992	.992
Range		1.00	1.00	1.00	1.00	1.00
Minimum		4.00	4.00	4.00	4.00	4.00
Maximum		5.00	5.00	5.00	5.00	5.00
Sum		91.00	92.00	87.00	89.00	95.00
Percentiles	25	4.0000	4.0000	4.0000	4.0000	4.2500
	50	5.0000	5.0000	4.0000	4.0000	5.0000
	75	5.0000	5.0000	5.0000	5.0000	5.0000

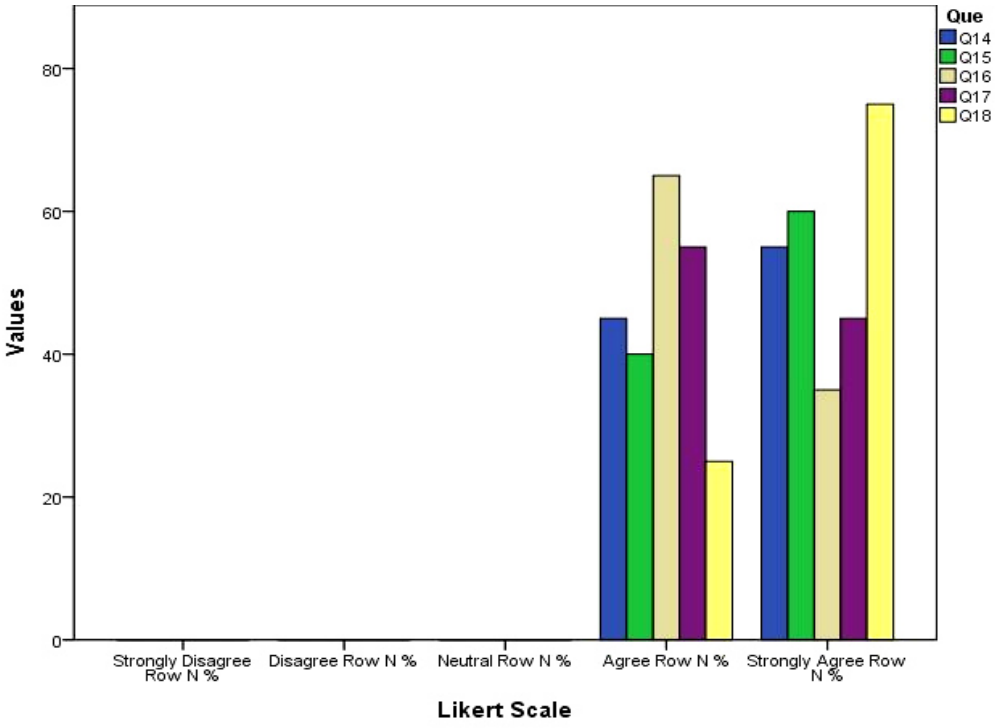

Figure 6.7 Bar graph of Satisfaction Feedback in Tryout phase

6.3.5 Finalization

The outcome of the Tryout phase shown in the Table 6.5, Table 6.6, Table 6.7 and also the graphical representation of the outcome can be seen in Figure 6.5, Figure 6.6 and Figure 6.7. After comparing these results with the outcomes of the previous step in Pre-tryout phase the researcher observed more improved outcome in each criterion. For example, the mean value for ease of use increased from 3.9 to 4.65. The similar improved result noticed in each outcome of the criterion. In addition, the modified STORE methodology is more powerful in recognizing the modern complex threats with the help of updated threat dictionary that contains a collection of previously identified threats with corresponding most appropriate security requirements. This threat dictionary is based on enigma software threat database. This modification assists the security requirement engineer to elicit more effective security requirements by accessing the new complex threat dictionary knowledge. In the Appendix C feedback section, the security experts did not add any comments or suggestions regarding the modified STORE methodology. Though, they agreed that the modified STORE

methodology fulfilled the ease of use evaluation, ease of learning evaluation, and satisfaction evaluation.

6.4 Results and Discussion

In this study, the researcher designed an evaluation process for the validation of STORE methodology. The purpose of the STORE methodology is to assist the security requirements engineer by eliciting complete and well-organized security requirement for the software which is to be developed in software development organizations. STORE methodology is proficient of recognize and priorities the possible threats and identify the most suitable security requirements in an organized way. The identification and analysis of four points of security attack PoA, PoB, PoC, PoD makes the identification of threats easier. Today's most of the software development organizations are anticipated to be able to build quality software products. In context of software security in order to evaluate and validate the STORE methodology, the researcher has designed a systematic evaluation process.

In this process the researcher first created an experiment design. In the second phase which is Pre-tryout, the feedbacks from ten security experts were taken and analyzed in order to assess the usability, learnability and user satisfaction of the STORE methodology. The outcomes of the Pre-tryout phase in the form of feedback result encouraged some modifications to the STORE methodology. The suggested modifications included the improved capability of the threat dictionary to cover the modern complex threats by updating the threat dictionary knowledge. Thus, the researcher constructed a modified threat dictionary that encompasses a group of earlier recognized threats with corresponding most suitable security requirements. This threat dictionary is based on enigma software threat database which has currently huge number of updated threat database. The security requirement elicitation process becomes more effective and flexible with the help of STORE methodology. These modifications improved the functionality and appropriateness of the STORE methodology to elicit the security requirements in the software organizations. The researcher used the SPSS software to assess the feedback received from the security experts in Pre-tryout and Tryout phase. The experimental result for each parameter describes the rate of acceptance of the STORE methodology. Undoubtedly STORE

methodology is capable of eliciting more powerful security requirements to develop quality software products.

6.5 Conclusion

The functionality of STORE methodology is easy to use, easy to learn and support the security requirement engineers to elicit complete and well-organized security requirements in a more systematic manner. STORE methodology can be applied for the web applications and security-critical software system which consider data as valuable assets and assessed the strength of the methodology. The researcher begins the software development with the context of the system, the functional requirements, and the primary security requirements. STORE methodology is a systematic and practical approach, and an appropriately accurate assessment requires a real project. In the future, the researcher intended to implement the STORE methodology for many other big software development projects. The additional future work is to build a tool for STORE methodology to identify security requirements for security-critical software products. To further improve this work, STORE methodology requires generating complete outcomes in arrears to different security mechanisms and facilities in various organizations. It requires more participation and association with several software development organizations. Furthermore, we also plan to extend the STORE methodology with detailed features appropriate to several modern technologies, for example Internet of Things (IoT), edge computing, cloud computing etc.

Chapter 7 Conclusion and Future Work

"A conclusion is the place where you get tired of thinking."

- Arthur Bloch

7.1 Background

This chapter summarizes the main contributions of the research and also presents the future work in the area of security requirements engineering. The main problems that have been taken into deliberation and those remaining for the area have been discussed in addition to limitations of the research work. Security requirements engineering is used for the elicitation of security requirements which ensures quality software product. Security requirements are the non-functional requirements and it must be consider with the functional requirements of the software product. Security requirements engineering is broadly used in research and software industries. The security requirements engineering concept begins with the requirements engineering phase of software development life cycle. In this phase identification of complete and well-organized security requirement is very difficult and challenging task. Further, it is not only about identification of security requirements. Security requirements identified from various sources and integrated with other sources to analyses the whole security requirements for generating a list of complete and well-organized security requirements.

The big problem with most of the security requirements engineering approaches that they are not capable of eliciting complete and well-organized set of security

requirements. If they are capable of doing so then they are more complex and don't assist the requirements engineer properly. An adversary can attack the software and access the private data with the help of some basic information of that particular user available in multiple sources. It is the responsibility of security requirements approach to identify every possible way through the adversary can enter into the system. This is possible with only a good and well-structured security requirements engineering approach. It is obvious from the literature available in this area that most of the security requirements engineering approaches unable to elicit a complete and efficient security requirements specification. Most security Requirements engineering approach normally does not encompass all important stakeholders and does not use the well-organized techniques for stakeholder identification and prioritization [41]. Generally the security requirements specification is incomplete, unclear, contradictory, not cohesive, unsystematic, infeasible, out-dated, unable to be validated, and not usable by their anticipated persons [8]. Several authors have been proposed methodology for analysing security requirements engineering. A good security requirement should be well organized in a systematic manner otherwise the software system cannot be assessed for achievement. Therefore, there is a need to develop an approach or methodology which is proficient of eliciting more complete and efficient security requirements by considering all the issues which are neglected by the previous approaches.

Hence, an effective and well-organized security requirements engineering approach for quality and secure software development has been proposed by the researcher. This is the complete security requirements engineering approach. The proposed STORE methodology is a ten-step sequential process which provides an effective, efficient and systematic way of eliciting and documenting security requirements for the software as well as web-based applications from the early phases of software development. In this methodology, security requirements are repeatedly conferred in the context of threats. Threat helps the security requirement engineer to calculate the risk associated with it and also characterizes the adversary's abilities. Stakeholder plays a significant character in the STORE methodology.

In STORE methodology the researcher identify and prioritize all such stakeholders based on their importance. It is important to consider every significant stakeholder from the beginning of software development. The proposed methodology considers security threats for identifying security requirements with the help of potential stakeholders.

These stakeholders help the requirement engineer in asset identification of the software product. The STORE methodology starts with identifying system goals. Each step of STORE methodology is equally important. The requirement engineer can't skip or jump to any other step because it is a systematic approach for eliciting and documenting security requirement. Further, the researcher has provided strategies to elicit complete set of security requirements. The main aim is to develop a complete and well-organized security requirements engineering approach which is easy to use and overcomes the issues with existing security requirements engineering approaches. The STORE methodology has been implemented on several case studies of software development. Validation of the STORE methodology has been also performed with the help of several experts and academics in the area of security requirements engineering. Statistical interpretation represents that the proposed STORE Methodology approach is acceptable as well as useful to supports the security requirement engineer to elicit security requirements in a more organized manner.

7.2 Major Research Contributions

In depth study of the existing literature on security requirements engineering and a challenging research process done by the researcher has made the following major contributions:

- **Literature Survey on Security Requirements Engineering**

A significant review has been done security requirements engineering during the preliminary stages of research work. It has been perceived that from the past few years, research in security requirement engineering has been going on worldwide. Software security is currently one of the most important concerns to the world as the researcher are depending on information technology rapidly. Continuous growth and heavily dependency on information technology and IT based resources has produced a new problem: software attack cannot be controlled by out-dated security requirements engineering approaches because attackers created different new attack mechanism to break the confidentiality, integrity, and authenticity of software product. So, security is the elementary requirement of the software system. The development of secure and

quality software product is the demand of time. It is the first step towards providing the secure environment. The detailed description is presented in chapter 2. This study has been published in [112].

- **Criteria Decision Making for Effective Security Requirements Engineering**

After reviewing the several available literatures in the area of security requirements engineering, the researcher has observed that most significant criteria which assess the operational strength of any security requirements engineering approach. The researcher prioritizes these significant criteria with the help of security expert's pairwise criteria weight evaluation through AHP model. The main purpose of prioritizing security requirements engineering criterion is to help in selection and development of effective and efficient security requirements engineering approach. In order to improve decision making and to achieve this, it is necessary to establish a process adapted to requirements which takes into account these relationships, to help provide consistency to the prioritization done. For that, before assigning a final value of priority to each criterion it is important to consider the operational significance of those criteria in context of effective security requirements engineering with which it is in interdependency. In this chapter the researcher uses the principle of pair-wise comparisons of AHP the method that is deliberated as the most helpful method, to help to accomplish the best decisions conceivable and to clearly present the rationality of the decision made about prioritization.

The study results suggest that the criteria Threat identification (32.4% of the weight of influencing factors) is more important than Risk analysis (26.1%), SH Involvement (20.5%), Accuracy (6.4%), Scalability (4.6%), Usability (4.4%), Adaptability (3.4%) or Learnability (2.3%) in manipulating performance towards security requirements engineering approaches. Determining weights of essential motivation, purpose, and consciousness focus areas can help security decision-making and compliance with policy, and support design of effective security requirements engineering. However, these weights may in turn be affected by local organisational and educational factors. The presented AHP results in this chapter can be used to select or design an effective security requirements engineering approach which may assist the software developers in developing trustworthy quality software products.

The researcher also discuss about the stakeholders identification and prioritization based on its role in the elicitation and specification of security requirements. The detailed description is shown in chapter 3. The research work highlighting the same has been submitted for publication in [113].

- **A Novel Approach for Security Requirements Engineering**

A novel Security Threat Oriented Requirements Engineering (STORE) methodology has been developed to elicit the most appropriate, complete, clear, and well-organized security requirements. The objective of the proposed STORE methodology is to develop secure and quality software products by eliciting the complete security requirements. This methodology can be executed with software, web application and other security-critical applications. The proposed STORE methodology accomplishes better security requirements in comparison with other existing methods. Software system is very sensitive to any individual or business organization. The aim is to design a systematic approach to prevent the software attack and unauthorised access. With the help of this STORE methodology, software security can ensure from the beginning of the software development product. Conflicting from the earlier works in this field, the researcher has absorbed on the importance of stakeholders, threats and security attack analysis. It is a novel security threat oriented security requirements engineering methodology. The proposed approach significantly improves the quality and security of software system to prevent it from old as well as modern threats. The STORE methodology is implemented on different current case studies of software systems like ERP system, Online Examination System (OES) and many other software projects to validate the STORE methodology. The researcher has also presented a comparative performance analysis of the proposed approach STORE methodology with two existing techniques, namely, SQUARE and MOSRE. The researcher has shown that more effective and efficient security requirements can be elicited by the STORE methodology and that it helps the security requirement engineer to elicit security requirements in a more organized manner. The detailed description is presented in chapter 4 and chapter 5. This study has been published in [74].

- **Validation of STORE methodology through a Systematic Evaluation Process:**

Furthermore, the researcher has also evaluated and validated the STORE methodology with the help of a systematic evaluation process. The evaluation of the STORE methodology is a vital process to validate and improve the applicability of the STORE methodology for the real software industry. The evaluation of STORE methodology was conducted in two phases. First, a self-review of STORE methodology conducted during design time and second with the help of a systematic evaluation process. The detailed description is presented in chapter 6. The study has been submitted for publication [114].

7.3 Significance of the Work

Nowadays, software security is one of the big issues that required building reliable and quality software systems. In recent years, the researcher has witnessed several organizations are becoming comprehensively dependent on information technology and its applications to get more help rapidly [74]. This revolutionary technological dependency has created a requirement for the development of defensive software systems from potential threats. Therefore, considering software security at the early phases of software development is becoming an essential issue [2]. In this situation, software security has become a challenging problem, because there are no universally acceptable security requirements engineering approach to deliver secure software. It is very difficult to find an appropriate way to elicit complete and well-organized security requirements and most of the characteristics related to it. In addition, study of security requirements has become an important need for the software industry, security developers and the users. The study carried on developing a systematic approach which is capable of eliciting effective and efficient security requirements through considering all stakeholders involved in the software development process and by performing security attack analysis. With its significant adding in the area of knowledge; are significant directly or indirectly in context of the followings:

- The study will help to minimize the software security issues.

- It may help to provide complete and well-organized security requirements for the security-critical software and web applications.
- With the help of proposed STORE Methodology, adversary's attack can be prevented successfully.
- Proposed STORE methodology algorithm is easy to implement and it is very difficult for attacker to attack the software product.
- STORE methodology will assists the security requirements engineer in developing quality and secure software product.
- The proposed work may inspire and allow the researchers to come out with more competent methodologies for refining the security requirements engineering approaches.
- In proposed approach, modern well-organized security requirements are identified easily.

7.4 Future Directions

Research is an on-going activity. Accomplishment of one milestone encourages the way to the next. As a future research plan, there may be the following tasks to be performed:

- In future, the researcher has intended to implement the STORE methodology for many other big software development projects.
- Analysing a variety of other case studies and comparative experiments using other existing configurations.
- The researcher will develop an automatic tool for STORE methodology to elicit security requirements more efficiently.
- The researcher will plan to conduct more experiments with the help of different software industry in their software projects to draw more concrete conclusions.
- The researcher also plans to extend the STORE methodology with detailed features appropriate to several modern technologies, for example Internet of Things (IoT), edge computing, cloud computing etc.

7.5 Research Findings

During the course's study, the research objective has been to find out the answers to research questions posed in chapter-1. In this line, the researcher has also tried to solve the security challenges acknowledged during survey of literature. The present section answers the research questions raised in chapter-1.

- **Research Question:** What are the major challenges with respect to secure and trustworthy software products?

  Research Finding: A major challenge toward secure software development is the lack of security knowledge and expertise among ordinary software developers.
- **Research Question:** What is the current status of security requirements engineering research?

  Research Finding: There are many security requirements engineering approaches available today. Although considering stakeholders view in security requirement engineering is an important concern, but only some SRE approaches address this concern. This doesn't mean that it is impossible to consider the views of different stakeholders using other methods. There is need for an effective and efficient security requirements engineering approach which consider every potential stakeholder's knowledge about system security and having effective mechanism for security threat identification.
- **Research Question:** Does the stakeholder's involvement in security requirements engineering process have been addressed previously?

  Research Finding:

  Though this issue has been addressed by some researchers previously but the work still needs improvement. There are some security requirements engineering approaches available like MSRA, KAOS, Secure Tropos, SQUARE, STS, DIGS, and SRERM which considers stakeholders concerns during security requirements engineering process.
- **Research Question:** Is there any security requirements engineering approach available which provide effective security attack analysis?

  Research Finding: No, there are no security requirements engineering approach available which provide security attack analysis during the security requirements

engineering process. The proposed STORE methodology is the only security requirements engineering approach which analyses the security attack for a software system in order to elicit complete and well-structured security requirements.

7.6 Conclusion

The STORE methodology is easy to use, easy to learn and support the security requirement engineers to elicit effective and efficient security requirements in a more organized manner. STORE methodology can be applied for the web applications and security-critical software system which consider data as valuable assets and assessed the strength of the methodology. The researcher begins the software development with the context of the system, the functional requirements, and the primary security requirements. STORE methodology approach is practical, and an appropriately accurate assessment requires a real project. In the future, the researcher has intended to implement the STORE methodology for many other big software development projects. The additional future work is to develop a tool for STORE methodology to elicit security requirements.

To further improve this work, STORE methodology needs to generate complete outcomes due to different security mechanisms and facilities in various organizations. It needs more collaboration with several software development organizations. Some organizations that have security third parties or a large number of security experts will have a different priority for implementing SRE to growing organizations. Moreover, the researcher also plans to extend the STORE methodology with detailed features appropriate to several modern technologies, for example Internet of Things (IoT), edge computing, cloud computing etc. STORE Methodology will confidently bring significant improvement in security requirements engineering field. Successful validation of the proposed methodology has also reflected its acceptability.

References

[1] Ari, A. A. A., Ngangmo, O. K., Titouna, C., Thiare, O., Mohamadou, A., & Gueroui, A. M. (2019). Enabling Privacy and Security in Cloud of Things: architecture, applications, security & privacy challenges. *Applied Computing and Informatics*.

[2] Zulkernine, M., & Ahamed, S. I. (2006). Software security engineering: toward unifying software engineering and security engineering. In *Enterprise Information Systems Assurance and System Security: Managerial and Technical Issues* (pp. 215-233). IGI Global.

[3] Türpe, S. (2017, September). The trouble with security requirements. In *Requirements Engineering Conference (RE), 2017 IEEE 25th International* (pp. 122-133). IEEE.

[4] Gartner,2018, Security Risks Drive Growth in Overall Security Spending, Cited 16 April 2018. Retrieved from: https://www.gartner.com/newsroom/id/3836563

[5] Rob Joyce, "Improving and Making the Vulnerability Equities Process Transparent is the Right Thing to Do" In National Security & Defence, Cited 16 April 2018. Retrieved from: https://www.whitehouse.gov/articles/improving-making-vulnerability-equities-process-transparent-right-thing/

[6] Amoroso, E. (2018). Recent Progress in Software Security. *IEEE Software*, *35*(2), 11-13.

[7] Kotonya, G., & Sommerville, I. (1998). *Requirements engineering: processes and techniques*. Wiley Publishing.

[8] Mead, N. R. (2008). Security requirements engineering. Build Security In 2006-08, 10.

[9] Yu, E. S. (1997, January). Towards modelling and reasoning support for early-phase requirements engineering. In *Requirements Engineering, 1997., Proceedings of the Third IEEE International Symposium on* (pp. 226-235). IEEE.

[10] Moffett, J. D., & Nuseibeh, B. A. (2003). A framework for security requirements engineering. *Report-University of York Department of Computer Science YCS*.

[11] Viega, J. (2005, May). Building security requirements with CLASP. In *ACM SIGSOFT Software Engineering Notes* (Vol. 30, No. 4, pp. 1-7). ACM.

[12] Tsoumas, B., & Gritzalis, D. (2006, April). Towards an ontology-based security management. In *Advanced Information Networking and Applications, 2006. AINA 2006. 20th International Conference on* (Vol. 1, pp. 985-992). IEEE.

[13] Lee, S. W., Gandhi, R., Muthurajan, D., Yavagal, D., & Ahn, G. J. (2006, May). Building problem domain ontology from security requirements in regulatory documents. In *Proceedings of the 2006 international workshop on Software engineering for secure systems* (pp. 43-50). ACM.

[14] Hussein, M., & Zulkernine, M. (2007). Intrusion detection aware component-based systems: A specification-based framework. *Journal of Systems and Software*, *80*(5), 700-710.

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

[16] Salini, P., & Kanmani, S. (2013). Model oriented security requirements engineering (MOSRE) framework for web applications. In *Advances in Computing and Information Technology* (pp. 341-353). Springer, Berlin, Heidelberg.

[17] Riaz, M., Stallings, J., Singh, M. P., Slankas, J., & Williams, L. (2016, September). DIGS: A framework for discovering goals for security requirements engineering. In *Proceedings of the 10th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement* (p. 35). ACM.

[18] Ansari, M. T. J., & Pandey, D. (2017). An Integration of Threat Modeling with Attack Pattern and Misuse Case for Effective Security Requirement Elicitation. *International Journal of Advanced Research in Computer Science*, *8*(3).

[19] Rehman, S., & Gruhn, V. (2018). An Effective Security Requirements Engineering Framework for Cyber-Physical Systems. *Technologies*, *6*(3), 65.

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

[21] Popp, G., Jurjens, J., Wimmel, G., & Breu, R. (2003, December). Security-critical system development with extended use cases. In *Software Engineering Conference, 2003. Tenth Asia-Pacific* (pp. 478-487). IEEE.

[22] Peeters, J. (2005, August). Agile security requirements engineering. In *Symposium on Requirements Engineering for Information Security*.

[23] Firesmith, D. G. (2005, May). Engineering safety-related requirements for software-intensive systems. In *Proceedings of the 27th international conference on Software engineering*(pp. 720-721). ACM.

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

[25] Basin, D., Doser, J., & Lodderstedt, T. (2006). Model driven security: From UML models to access control infrastructures. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, *15*(1), 39-91.

[26] Paja, E., Dalpiaz, F., & Giorgini, P. (2015). Modelling and reasoning about security requirements in socio-technical systems. *Data & Knowledge Engineering*, *98*, 123-143.

[27] Toval, A., Nicolás, J., Moros, B., & García, F. (2002). Requirements reuse for improving information systems security: a practitioner's approach. *Requirements Engineering*, *6*(4), 205-219.

[28] Lee, J., Lee, J., Lee, S., & Choi, B. (2003, November). A CC-based security engineering process evaluation model. In *Computer Software and Applications Conference, 2003. COMPSAC 2003. Proceedings. 27th Annual International* (pp. 130-135). IEEE.

[29] Zuccato, A. (2004). Holistic security requirement engineering for electronic commerce. *Computers & Security*, *23*(1), 63-76.

[30] Myagmar, S., Lee, A. J., & Yurcik, W. (2005, August). Threat modeling as a basis for security requirements. In *Symposium on requirements engineering for information security (SREIS)*(Vol. 2005, pp. 1-8).

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

[32] Van Lamsweerde, A. (2007). Engineering requirements for system reliability and security. *NATO Security Through Science Series D-Information and Communication Security*, *9*, 196.

[33] Shin, M. E., & Gomaa, H. (2007). Software requirements and architecture modeling for evolving non-secure applications into secure applications. *Science of Computer Programming*, *66*(1), 60-70.

[34] Hassan, R., Eltoweissy, M., Bohner, S., & El-Kassas, S. (2010). Formal analysis and design for engineering security automated derivation of formal software security specifications from goal-oriented security requirements. *IET software*, *4*(2), 149-160.

[35] Mufti, Y., Niazi, M., Alshayeb, M., & Mahmood, S. (2018). A Readiness Model for Security Requirements Engineering. *IEEE Access*.

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

[37] Jennex, M. E. (2005). Modeling security requirements for information systems development. In *Proc Symposium on Requirements Engineering for Information Security* (pp. 3-3).

[38] Mead, N. R., & Stehney, T. (2005). *Security quality requirements engineering (SQUARE) methodology* (Vol. 30, No. 4, pp. 1-7). ACM.

[39] Gürses, S. F., & Santen, T. (2006). Contextualizing Security Goals: A Method for Multilateral Security Requirements Elicitation. In *Sicherheit* (Vol. 6, pp. 42-53).

[40] El-Hadary, H., & El-Kassas, S. (2014). Capturing security requirements for software systems. *Journal of advanced research*, *5*(4), 463-472.

[41] Faily, S. (2015). Engaging stakeholders during late stage security design with assumption personas. *Information & Computer Security*, *23*(4), 435-446.

[42] McGraw, G. (2006). *Software security: building security in* (Vol. 1). Addison-Wesley Professional.

[43] Ramachandran, M. (2015, September). Software security requirements engineering: State of the art. In *International Conference on Global Security, Safety, and Sustainability* (pp. 313-322). Springer, Cham.

[44] Chung, L., Nixon, B., Yu, E., & Mylopoulos, J. (2000). Non-functional requirements. *Software Engineering*.

[45] Devanbu, P. T., & Stubblebine, S. (2000, May). Software engineering for security: a roadmap. In *Proceedings of the Conference on the Future of Software Engineering* (pp. 227-239). ACM.

[46] Firesmith, D. (2004). Engineering safety requirements, safety constraints, and safety-critical requirements. *Journal of object technology*, *3*(3), 27-42.

[47] Glinz, M. (2005, September). Rethinking the notion of non-functional requirements. In *Proc. Third World Congress for Software Quality* (Vol. 2, pp. 55-64).

[48] Ansari, M. T. J., & Pandey, D. (2018). Risks, Security, and Privacy for HIV/AIDS Data: Big Data Perspective. In *Big Data Analytics in HIV/AIDS Research* (pp. 117-139). IGI Global.

[49] Massacci, F., Prest, M., & Zannone, N. (2005). Using a security requirements engineering methodology in practice: The compliance with the Italian data protection legislation. *Computer Standards & Interfaces*, *27*(5), 445-455.

[50] Van Lamsweerde, A. (2000, June). Requirements engineering in the year 00: a research perspective. In *Proceedings of the 22nd international conference on Software engineering* (pp. 5-19). ACM.

[51] Herrmann, P., & Herrmann, G. (2006). Security requirement analysis of business processes. *Electronic Commerce Research*, *6*(3-4), 305-335.

[52] Olivier, M. S. (2002). Database privacy: balancing confidentiality, integrity and availability. *ACM SIGKDD Explorations Newsletter*, *4*(2), 20-27.

[53] Firesmith, D. (2004). Specifying reusable security requirements. *Journal of Object Technology*, *3*(1), 61-75.

[54] Haley, C. B., Moffett, J. D., Laney, R., & Nuseibeh, B. (2006, May). A framework for security requirements engineering. In *Proceedings of the 2006 international workshop on Software engineering for secure systems* (pp. 35-42). ACM.

[55] Firesmith, D. (2003). Engineering security requirements. *Journal of object technology*, *2*(1), 53-68.

[56] Rushby, J. (2001, March). Security requirements specifications: How and what. In *Symposium on Requirements Engineering for Information Security (SREIS)* (Vol. 441).

[57] Anton, A. I. (1996, April). Goal-based requirements analysis. In *Proceedings of the second international conference on requirements engineering* (pp. 136-144). IEEE.

[58] McDermott, J., & Fox, C. (1999, December). Using abuse case models for security requirements analysis. In *Proceedings 15th Annual Computer Security Applications Conference (ACSAC'99)* (pp. 55-64). IEEE.

[59] Yu, E., & Liu, L. (2001). Modelling trust for system design using the i* strategic actors framework. In *Trust in Cyber-societies* (pp. 175-194). Springer, Berlin, Heidelberg.

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

[61] den Braber, F., Dimitrakos, T., Gran, B. A., Lund, M. S., Stolen, K., & Aagedal, J. O. (2003). The CORAS methodology: model-based risk assessment using UML and UP. In *UML and the Unified Process* (pp. 332-357). IGI Global.

[62] Firesmith, D. G. (2003). Security use cases. *Journal of object technology*, *2*(3).

[63] Lin, L., Nuseibeh, B., Ince, D., & Jackson, M. (2004, September). Using abuse frames to bound the scope of security problems. In *Proceedings. 12th IEEE International Requirements Engineering Conference, 2004.* (pp. 354-355). IEEE.

[64] Mayer, N., Rifaut, A., & Dubois, E. (2005, June). Towards a risk-based security requirements engineering framework. In *Proc. of REFSQ* (Vol. 5).

[65] Asnar, Y., & Giorgini, P. (2006, August). Modelling risk and identifying countermeasure in organizations. In *International Workshop on Critical Information Infrastructures Security* (pp. 55-66). Springer, Berlin, Heidelberg.

[66] Mellado, D., Fernández-Medina, E., & Piattini, M. (2006, September). Applying a security requirements engineering process. In *European Symposium on Research in Computer Security* (pp. 192-206). Springer, Berlin, Heidelberg.

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

[68] Hatebur, D., Heisel, M., & Schmidt, H. (2007, September). A security engineering process based on patterns. In *18th International Workshop on Database and Expert Systems Applications (DEXA 2007)* (pp. 734-738). IEEE.

[69] Basin, D., Doser, J., & Lodderstedt, T. (2003, June). Model driven security for process-oriented systems. In *Proceedings of the eighth ACM symposium on Access control models and technologies* (pp. 100-109). ACM.

[70] Giorgini, P., Massacci, F., Mylopoulos, J., & Zannone, N. (2006). Requirements engineering for trust management: model, methodology, and reasoning. *International Journal of Information Security*, *5*(4), 257-274.

[71] Ali, R., Dalpiaz, F., & Giorgini, P. (2009). A goal modeling framework for self-contextualizable software. In *Enterprise, Business-Process and Information Systems Modeling* (pp. 326-338). Springer, Berlin, Heidelberg.

[72] Dalpiaz, F., Giorgini, P., & Mylopoulos, J. (2009, June). An architecture for requirements-driven self-reconfiguration. In *International Conference on Advanced Information Systems Engineering* (pp. 246-260). Springer, Berlin, Heidelberg.

[73] Hassan, R., & Mansour, A. M. (2009). Formal analysis and design for engineering security (FADES).

[74] Ansari, M. T. J., Pandey, D., & Alenezi, M. (2018). STORE: Security Threat Oriented Requirements Engineering Methodology. *Journal of King Saud University-Computer and Information Sciences*.

[75] Fabian, B., Gürses, S., Heisel, M., Santen, T., & Schmidt, H. (2010). A comparison of security requirements engineering methods. *Requirements engineering*, *15*(1), 7-40.

[76] Saaty, T. L. (1990). How to make a decision: the analytic hierarchy process. *European journal of operational research*, *48*(1), 9-26.

[77] Ho, W., Dey, P. K., & Higson, H. E. (2006). Multiple criteria decision-making techniques in higher education. *International journal of educational management*, *20*(5), 319-337.

[78] Kuruoglu, E., Guldal, D., Mevsim, V., & Gunvar, T. (2015). Which family physician should I choose? The analytic hierarchy process approach for ranking of criteria in the selection of a family physician. *BMC medical informatics and decision making*, *15*(1), 63.

[79] Deshmukh, M. (2009, July). Security Requirements Engineering Process. In *Seminar in Information System, Security Engineering*.

[80] Gutgarts, P. B., & Temin, A. (2010, November). Security-critical versus safety-critical software. In *2010 IEEE International Conference on Technologies for Homeland Security (HST)* (pp. 507-511). IEEE.

[81] Bulusu, S. T., Laborde, R., Wazan, A. S., Barrere, F., & Benzekri, A. (2018, April). Applying a requirement engineering based approach to evaluate the security requirements engineering methodologies. In *Proceedings of the 33rd Annual ACM Symposium on Applied Computing* (pp. 1316-1318). ACM.

[82] Ramachandran, M. (2016). Software security requirements management as an emerging cloud computing service. *International Journal of Information Management*, *36*(4), 580-590.

[83] Pandey, D., Suman, U., & Ramani, A. K. (2011). Security requirement engineering issues in risk management. *International Journal of Computer Applications*, *975*, 8887.

[84] Ullah, S., Iqbal, M., & Khan, A. M. (2011, July). A survey on issues in non-functional requirements elicitation. In *International Conference on Computer Networks and Information Technology* (pp. 333-340). IEEE.

[85] Luburić, N., Sladić, G., & Milosavljević, B. (2018, October). Applicability Issues in Security Requirements Engineering for Agile Development. In *Proceedings/8 th International conference on applied internet and information technologies* (Vol. 8,

No. 1, pp. II-VII). “St Kliment Ohridski” University-Bitola, Faculty of Information and Communication Technologies-Bitola, Republic of Macedonia.

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

[87] Yue, K. (1987, April). What does it mean to say that a specification is complete?. In *Proc. IWSSD-4, Fourth International Workshop on Software Specification and Design*.

[88] Van Lamsweerde, A. (2001). Goal-oriented requirements engineering: A guided tour. In *Requirements Engineering, 2001. Proceedings. Fifth IEEE International Symposium on*(pp. 249-262). IEEE.

[89] Sharp, H., Finkelstein, A., & Galal, G. (1999). Stakeholder identification in the requirements engineering process. In *Database and expert systems applications, 1999. proceedings. tenth international workshop on* (pp. 387-391). Ieee.

[90] Glinz, M., & Wieringa, R. J. (2007). Guest editors' introduction: Stakeholders in requirements engineering. *IEEE software*, *24*(2), 18-20.

[91] Almorsy, M., Grundy, J., & Ibrahim, A. S. (2011, July). Collaboration-based cloud computing security management framework. In *2011 IEEE 4th International Conference on Cloud Computing* (pp. 364-371). IEEE.

[92] Ashbaugh, D. A. (2008). *Security Software Development: Assessing and Managing Security Risks*. Auerbach Publications.

[93] Swiderski, F., & Snyder, W. (2004). *Threat Modeling (Microsoft Professional)* (Vol. 7). Microsoft Press.

[94] Mell, P., Scarfone, K., & Romanosky, S. (2007, June). A complete guide to the common vulnerability scoring system version 2.0. In *Published by FIRST-Forum of Incident Response and Security Teams* (Vol. 1, p. 23).

[95] Xu, D., & Nygard, K. E. (2006). Threat-driven modeling and verification of secure software using aspect-oriented Petri nets. *IEEE Transactions on Software Engineering*, *32*(4), 265-278.

[96] Finkelstein, A., & Fuks, H. (1989). Multi-party specification. In *Proceedings of the Fifth International Workshop on Software Specification and Design*.

[97] Al-Mudimigh, A., Zairi, M., & Al-Mashari, M. (2001). ERP software implementation: an integrative framework. *European Journal of Information Systems*, *10*(4), 216-226.

[98] Nah, F. and Lau, J. (2001). Critical Success Factors for Successful Implementation of Enterprise Systems. Business Process Management Journa,. 7(3), 285 - 296.

[99] Hong, K. and Kim, Y. (2002). The Critical Success Factors for ERP Implementation: An Organizational Fit Perspective. Information and Management, 40(1), 25-40

[100] Umble, E., Haft, R. and Umble, M. (2003). Enterprise Resource Planning: Implementation Procedures and Critical Success Factors. European Journal of Operational Research, 146(2), 24 1-257.

[101] Al-Mashari M. and Al-Mudimigh A. (2003). ERP Implementation: Lessons from a Case Study. Information Technology and People, 16(1), 2 1-33.

[102] Al-Mashari, M., Al-Mudimigh, A. & Zairi, M. (2003). Enterprise Resource Planning: A Taxonomy of Critical Factors. European Journal of Operational Research, 146(2), 352 - 364.

[103] Okunoye, A., Frolick, M., & Crable, E. (2008). Stakeholder influence and ERP implementation in higher education. Journal of Information Technology Case and Application Research, 10(3), 9-38.

[104] Scheer, A. W., & Habermann, F. (2000). Enterprise resource planning: making ERP a success. Communications of the ACM, 43(4), 57-61.

[105] Gordon, D., Stehney, T., Wattas, N., & Yu, E. (2005). *System quality requirements engineering (square): Case study on asset management system, phase ii* (No. CMU/SEI-2005-SR-005). CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST.

[106] Salini, P., & Kanmani, S. (2012, October). Elicitation of security requirements for e-health system by applying Model Oriented Security Requirements Engineering (MOSRE) framework. In *Proceedings of the Second International Conference on Computational Science, Engineering and Information Technology* (pp. 126-131). ACM.

[107] Yevseiev, S., Aleksiyev, V., Balakireva, S., Peleshok, Y., Milov, O., Petrov, O., ... & Shmatko, O. (2019). Development of a methodology for building an information security system in the corporate research and education system in the context of university autonomy. *Восточно-Европейский журнал передовых технологий*, (3 (9)), 49-63.

[108] Kostlan, E. (1991). Statistical complexity of dominant eigenvector calculation. *Journal of Complexity*, *7*(4), 371-379.

[109] Dong, Y., Zhang, G., Hong, W. C., & Xu, Y. (2010). Consensus models for AHP group decision making under row geometric mean prioritization method. *Decision Support Systems*, *49*(3), 281-289.

[110] Scott, J. E. (2005). Post-implementation usability of ERP training manuals: the user's perspective. *Information systems management*, *22*(2), 67-77.

[111] She, W., & Thuraisingham, B. (2007). Security for enterprise resource planning systems. *Information Systems Security*, *16*(3), 152-163.

[112] Ansari, M. T. J., Pandey, D., & Khan, N. A. (2019). Comparative Literature Analysis on Security Requirements Engineering. *International Journal of Engineering Sciences & Research Technology*, *8*(12), 113–124. doi: 10.5281/zenodo.3596327

[113] **Ansari, M. T. J., Khan, A. I., Abushark, Y. B., Alam, M. M., & Khan, R. A. (n.d.). Analytic Hierarchy Process (AHP) based analysis toward selection of effective Security Requirements Engineering approach for trustworthy healthcare software development.** ***BMC Medical Informatics and Decision Making*****.**

[114] **Ansari, M. T. J., Pandey, D., Khan, A. I., & Khan, R. A. (n.d.). Evaluation and Validation of STORE Methodology: A Case Study on OES Software Project.** ***Journal of King Saud University Computer and Information Sciences*****.**

Appendix A: AHP Based Security Experts Response Form

Date…….

Participant's Profile

Name:_____Organization:_____

Designation:_____

Professional Experience (In years):_____

Please compare the importance of the following criteria in relation to the effective security requirements engineering approach and fill in the table: Which criterion of each pair is more important while selecting an effective security requirements engineering approach, A or B and how much more on a scale 1-9 as given below.

Intensity	Definition	Explanation
1	Equal importance	Two criteria contribute to the objective with equal relevance.
3	Moderate importance	A criteria is slightly more important than one other
5	Strong importance	Judgment strongly favors one element over another
7	Very strong importance	An element is strongly important than another
9	Extreme importance	The compared element is favored over another
2,4,6,8	Intermediate values	Used to represent a compromise between the above-mentioned preferences

You can use the above Saaty's 1–9 scale table to rate each criterion.

				Important?	**Scale**
i	**j**	**A**	**B**	**A or B**	**(1-9)**
1	2	**Usability**	Learnability		
1	3		SH Involvement		
1	4		Accuracy		
1	5		Risk Analysis		
1	6		Threat Identification		
1	7		Adaptability		
1	8		Scalability		
2	3	**Learnability**	SH Involvement		
2	4		Accuracy		

2	5		Risk Analysis		
2	6		Threat Identification		
2	7		Adaptability		
2	8		Scalability		
3	4	**SH Involvement**	Accuracy		
3	5		Risk Analysis		
3	6		Threat Identification		
3	7		Adaptability		
3	8		Scalability		
4	5	**Accuracy**	Risk Analysis		
4	6		Threat Identification		
4	7		Adaptability		
4	8		Scalability		
5	6	**Risk Analysis**	Threat Identification		
5	7		Adaptability		
5	8		Scalability		
6	7	**Threat Identification**	Adaptability		
6	8		Scalability		
7	8	**Adaptability**	Scalability		

Appendix B: Questionnaire Form

Assessment form for the evaluation of STORE Methodology

Date…….

Participant's Profile

Name:_____Organization:_____

Designation:_____

Professional Experience (In years):_____

Please rate how you strongly agree or disagree for each statement by placing a check mark in the appropriate box.

A. Ease of Use

Q.1 The STORE Methodology is easy to use

Strongly Agree □ Agree □ Neutral □ Disagree □ Strongly Disagree □

Q.2 A little knowledge of security requirements engineering is required to use STORE methodology

Strongly Agree □ Agree □ Neutral □ Disagree □ Strongly Disagree □

Q.3 Use of Security Attack Analysis by identifying four points PoA, PoB, PoC, PoD made easy to identification of Threats

Strongly Agree □ Agree □ Neutral □ Disagree □ Strongly Disagree □

Q.4 STORE Methodology made the security elicitation easier than any other existing SRE approach

Strongly Agree □ Agree □ Neutral □ Disagree □ Strongly Disagree □

Q.5 It is necessary to follow all the sequential steps of STORE methodology

Strongly Agree □ Agree □ Neutral □ Disagree □ Strongly Disagree □

Q.6 Some training shoud be provided to assist the use of STORE methodology

Strongly Agree □ Agree □ Neutral □ Disagree □ Strongly Disagree □

Q.7 If I need to identify security requirements in a future project, I will use the STORE Methodology.

Strongly Agree □ Agree □ Neutral □ Disagree □ Strongly Disagree □

B. Ease of Learning

Q.8 I learned to use STORE Methodology quickly

Strongly Agree □ Agree □ Neutral □ Disagree □ Strongly Disagree □

Q.9 I easily remember how to use STORE Methodology

Strongly Agree □ Agree □ Neutral □ Disagree □ Strongly Disagree □

Q.10 It is easy to learn to use STORE Methodology

Strongly Agree □ Agree □ Neutral □ Disagree □ Strongly Disagree □

Q.11 I quickly became skillful with STORE Methodology

Strongly Agree □ Agree □ Neutral □ Disagree □ Strongly Disagree □

Q.12 I easily learned the Security Attack Analysis of STORE methodology

Strongly Agree □ Agree □ Neutral □ Disagree □ Strongly Disagree □

Q.13 Every step of STORE Methodology is easy to learn and utilize for security requirements engineering

Strongly Agree □ Agree □ Neutral □ Disagree □ Strongly Disagree □

C. Satisfaction

Q.14 I am satisfied with STORE Methodology

Strongly Agree □ Agree □ Neutral □ Disagree □ Strongly Disagree □

Q.15 If a company I am employed in discusses what technique to introduce for early security requirements engineering I will suggest STORE Methodology.

Strongly Agree □ Agree □ Neutral □ Disagree □ Strongly Disagree □

Q.16 The STORE Methodology produces of all types of security requirements

Strongly Agree □ Agree □ Neutral □ Disagree □ Strongly Disagree □

Q.17 It works the way I want it to elicit the security requirements

Strongly Agree □ Agree □ Neutral □ Disagree □ Strongly Disagree □

Q.18 I feel I need to have it for my software organization

Strongly Agree □ Agree □ Neutral □ Disagree □ Strongly Disagree □

Appendix C: Questionnaire Form

Security Experts suggestions for the STORE Methdology

Participant's Profile Date…….

Name:_____Organization:_____

Designation:_____

_____Professional Experience (In years):_____

Please answer the following questions

Questions

1. Do you suggest any correction or enhancement to the STORE Methodology?

Response:_____

_____

_____

2. Do you suggest any new step for the STORE methodology in the future and if so, provide the reason?

Response:_____

_____

_____

3. What is your opinion on the security attack analysis step?

Response:_____

_____

_____

4. Any other comment

Response:_____

_____

_____

Appendix D: Plagiarism Report

URKUND

Urkund Analysis Result

Analysed Document:	My Thesis_tjtjansari@gmail.com.pdf (D62334958)
Submitted:	1/14/2020 11:21:00 AM
Submitted By:	gbl.bbau@gmail.com
Significance:	5 %

Sources included in the report:

For Plag Check.docx (D39937678)
Tarique SM for plag.docx (D55668113)
OES Software.docx (D58849065)
Tarique Ansari.docx (D41444090)
https://www.researchgate.net/publication/329726569_STORE_Security_Threat_Oriented_Requirements_Engineering_Methodology
https://arxiv.org/pdf/1901.01500
https://iajit.org/PDF/May%202018,%20No.%203/9978.pdf
https://cyberleninka.org/article/n/7495
https://www.researchgate.net/publication/326356262_An_Effective_Security_Requirements_Engineering_Framework_for_Cyber-Physical_Systems
https://cdn.ttgtmedia.com/searchSoftwareQuality/downloads/Software_Security_Engineering_Project_Managers_Requirements_CH3.pdf

Instances where selected sources appear:

53