International Journal of Reviews in Computing
31st December 2011. Vol. 8
IJRIC

ISSN: 2076-3328 www.ijric.org E-ISSN: 2076-3336

A SURVEY ON SECURITY REQUIREMENTS ENGINEERING

[1]P.SALINI, [2]S.KANMANI

[1]Asstt Prof., Department of Computer Science and Engineering, Pondicherry Engineering College, Puducherry, India

[2] Prof., Department of Information Technology, Pondicherry Engineering College, Puducherry, India

E-mail: [1]salini@pec.edu, [2]kanmani@pec.edu

ABSTRACT

Security engineering is a new research area in software engineering that covers the definition of processes, plans and designs for security. The researchers are working in this area and however there is a lack in security requirements treatment in this field. The security requirements is one of the non functional requirements which acts as constrains on the functions of the system. An increasing part of the communication and sharing of information in our society utilizes electronic media. Many organizations, especially distributed and Net-centric are entirely dependent on well functioning information systems. Thus IT security is becoming central to the ability to fulfill business goals, build trustworthy systems, and protect assets. In order to develop systems with adequate security features, it is essential to capture the corresponding security needs and requirements. Security requirements engineering is emerging as a branch of software engineering, spurred by the realization that security must be dealt with early during requirements phase. A number of researchers' proposals have major limitations as they treat security in system oriented terms. In this paper we present a view on Security Requirements, Security Requirements issues, and types, Security Requirements Engineering and methods. Comparison on different methods and trends of Security Requirements Engineering is given. With this short view information security requirements for banks and the approach that can be adopted for security requirements engineering can be easily identified by the developers.

Keywords: *Security Engineering, Security Requirements, Security Requirements Engineering, Security Requirements Engineering Methods*

1. INTRODUCTION

Applications In recent years, reports of software security failures have become commonplace. Many proposals that incorporate security in the software engineering process. At one end of the spectrum, such proposals ensure good coding practices [2]. At the other extreme, the emphasis is on securing the organization within which a software system functions [1]. In either case, modeling and analysis of security requirements has become a key challenge for Software Engineering [3,4].When building secure systems, it is instrumental to take security concerns into account right from the beginning of the development process. It is well recognized in the software industry that requirements engineering is critical to the success of any major development project. The elaboration, specification, analysis and documentation of application-specific security requirements is an area that has been left virtually unexplored by requirements engineering research to date. A requirements engineering (RE) technique for security critical systems should ideally meet the following meta-requirements.

Early deployment: In view of the criticality of security requirements, the technique should be applicable as early as possible in the RE process, that is, to declarative assertions as they arise from stakeholder interviews and documents.

Incrementality: The technique should support the intertwining of model building and analysis and therefore allow for reasoning about partial models.

Reasoning about alternatives: The technique should make it possible to represent and assess alternative options so that a "best" route to security can be selected.

High assurance: The technique should allow for formal analysis when and where needed so that compelling evidence of security assurance can be provided.

ISSN: 2076-3328 www.ijric.org E-ISSN: 2076-3336

Security-by-construction: To avoid the endless cycle of defect fixes generating new defects, the RE process should be guided so that a satisfactory level of security is guaranteed by construction.

Separation of concerns: the technique should keep security requirements separate from other types of requirements so as to allow for interaction analysis [5, 6].

The Source of Security Problem are not considering Security requirements of complete system and not considering Security in the application itself. The Security Goals [7] such as confidentiality, integrity, accessibility, and accountability and combining management control principles and application business goals forms the Security requirements.

This paper is structured as follows: Section 2 reviews Security Requirements, Security Requirements issues and types; Section 3 presents Security Requirements Engineering and methods and also gives Comparison on different methods and trends of Security Requirements Engineering. Section 4 concludes with future works.

2. SECURITY REQUIREMENTS

The input Security Requirements is defined as constraints on the functions of the system, and these constraints operationalize one or more security goals. But most requirements engineers are poorly trained to elicit, analyze, and specify security requirements, often confusing them with the architectural security mechanisms that are traditionally used to fulfill them. They thus end up specifying architecture and design constraints rather than true security requirements. This paper presents the different types of security requirements and provides associated examples with the intent of enabling requirements engineers to adequately specify security requirements without unnecessarily constraining the security and architecture teams from using the most appropriate security mechanisms for the job.

The engineering of the requirements for a business, system or software application, component, or (contact, data, or reuse) center involves far more than merely engineering its functional requirements. One must also engineer its quality, data, and interface requirements as well as its architectural, design, implementation, and testing constraints.

Whereas some requirements engineers might remember to elicit, analyze, specify, and manage such quality requirements as interoperability, operational availability, performance, portability, reliability, and usability, many are at a loss when it comes to security requirements. Most requirements engineers are not trained at all in security, and the few that have been trained have only been given an overview of security architectural mechanisms such as passwords and encryption rather than in actual security requirements. Thus, the most common problem with security requirements, when they are specified at all, is that they tend to be accidentally replaced with security-specific architectural constraints that may unnecessarily constrain the security team from using the most appropriate security mechanisms for meeting the true underlying security requirements. This paper will help to distinguish between security requirements and the mechanisms for achieving them, and will provide you with good examples of each type of security requirement. In today's world of daily virus alerts, malicious crackers, and the threats of cyber terrorism, would remember the following objectives of security requirements.

2.1 Security Requirements Issues

In reviewing requirements documents, we typically find that security requirements, when they exist, are in a section by themselves and have been copied from a generic set of security requirements. They tend to be general mechanisms such as password protection, firewalls, virus detection tools, and the like. The requirements elicitation and analysis that is needed to get a better set of security requirements seldom takes place. Even when it does, the security requirements are often developed independently of the rest of the requirements engineering activity and hence are not integrated into the mainstream of the requirements activities. As a result, security requirements that are specific to the system and that provide for protection of essential services and assets are often neglected.

The Problem of Negative Requirements

Many requirements engineering research and practice have addressed the capabilities that the system will provide. So a lot of attention is given to the functionality of the system, from the user's perspective, but little attention is given to what the system should not do. In one discussion on requirements prioritization for a specific large system, ease of use was assigned a higher priority than security requirements. Security requirements were in the lower half of the prioritized requirements. This occurred in part because the only security requirements that were considered had to do with access control. Current research

ISSN: 2076-3328 www.ijric.org E-ISSN: 2076-3336

recognizes that security requirements are negative requirements. General security requirements, such as "the system shall not allow successful attacks," are therefore generally not feasible, because there is no agreement on ways to validate them other than to apply formal methods to the entire system, including COTS components. We can, however, identify the essential services and assets that must be protected. We are able to validate that mechanisms such as access control, levels of security, backups, replication, and policy are implemented and enforced.

We can also validate that the system will properly handle specific threats identified by a threat model and correctly respond to intrusion scenarios. [8]

2.2 Security Requirements Types

[1] Identification Requirements

An identification requirement is any security requirement that specifies the extent to which a business, application, component, or center shall identify its externals (e.g., human actors and external applications) before interacting with them.

Examples

• "The application shall identify all of its client applications before allowing them to use its capabilities."

• "The application shall identify all of its human users before allowing them to use its capabilities."

• "The data center shall identify all personnel before allowing them to enter."

• "The application shall not require an individual user to identify himself or herself multiple times during a single session."

• "The application shall ensure that the name of the employee in the official human resource and payroll databases exactly matches the name printed on the employee's social security card."

[2] Authentication Requirements

An authentication requirement is any security requirement that specifies the extent to which a business, application, component, or center shall verify the identity of its externals (e.g., human actors and external applications) before interacting with them.

The typical objectives of an authentication requirement are to ensure that externals are actually who or what they claim to be and thereby to avoid compromising security to an impostor.

Examples

• "The application shall verify the identity of all of its users before allowing them to use its capabilities."

• "The application shall verify the identity of all of its users before allowing them to update their user information."

• "The application shall verify the identity of its user before accepting a credit card payment from that user."

• "The application shall verify the identity of all of its client applications before allowing them to use its capabilities."

• "The data center shall verify the identity of all personnel before permitting them to enter."

[3] Authorization Requirements

An authorization requirement is any security requirement that specifies the access and usage privileges of authenticated users and client applications.

The typical objectives of an authorization requirement are to:

• Ensure that one or more persons (who have been properly appointed on behalf of the organization that owns and controls the application or component) are able to authorize specific authenticated users and client applications to access specific application or component capabilities or information.

• Ensure that specific authenticated externals can access specific application or component capabilities or information if and only if they have been explicitly authorized to do so by a properly appointed person(s).

• Thereby prevent unauthorized users from:

1. Obtaining access to inappropriate or confidential data.

2. Requesting the performance of inappropriate or restricted services.

Examples

• "The application shall allow each customer to obtain access to all of his or her own personal account information."

ISSN: 2076-3328 www.ijric.org E-ISSN: 2076-3336

• "The application shall not allow any customer to access any account information of any other customer."

• "The application shall not allow customer service agents to access the credit card information of customers."

• "The application shall allow customer service agents to automatically email a new customer password to that customer's email address."

• "The application shall not allow customer service agents to access either the original or new customer password when emailing the new customer password to the customer's email address."

• "The application shall not allow one or more users to successfully use a denial of service (DoS) attack to flood it with legitimate requests of service."

[4] Immunity Requirements

An immunity requirement is any security requirement that specifies the extent to which an application or component shall protect itself from infection by unauthorized undesirable programs (e.g., computer viruses, worms, and Trojan horses).

The typical objectives of an immunity requirement are to prevent any undesirable programs from destroying or damaging data and applications.

Examples

• "The application shall protect itself from infection by scanning all entered or downloaded data and software for known computer viruses, worms, Trojan horses, and other similar harmful programs."

• "The application shall disinfect any file found to contain a harmful program if disinfection is possible."

• "The application shall notify the security administrator and the associated user (if any), if it detects a harmful program during a scan."

• "To protect itself from infection by new infectious programs as they are identified and published, the application shall daily update its definitions of known computer viruses, worms, Trojan horses, and other similar harmful programs."

[5] Integrity Requirements

An integrity requirement is any security requirement that specifies the extent to which an application or component shall ensure that its data and communications are not intentionally corrupted via unauthorized creation, modification, or deletion.

The typical objectives of an integrity requirement are to ensure that communications and data can be trusted.

Examples

• "The application shall prevent the unauthorized corruption of emails (and their attachments, if any) that it sends to customers and other external users."

• "The application shall prevent the unauthorized corruption of data collected from customers and other external users."

• "The application shall prevent the unauthorized corruption of all communications passing through networks that are external to any protected data centers."

[6] Intrusion Detection Requirements

An intrusion detection requirement is any security requirement that specifies the extent to which an application or component shall detect and record attempted access or modification by unauthorized individuals.

The typical objectives of an intrusion detection requirement are to:

• Detect unauthorized individuals and programs that are attempting to access the application or component.

• Record information about the unauthorized access attempts.

• Notify security personnel so that they can properly handle them.

Examples

• "The application shall detect and record all attempted accesses that fail identification, authentication, or authorization requirements."

• "The application shall daily notify the data center security officer of all failed attempted accesses during the previous 24 hours."

• "The application shall notify the data center security officer within 5 minutes of any repeated failed attempt to access the employee and corporate financials databases."

[7] Nonrepudiation Requirements

A nonrepudiation requirement is any security requirement that specifies the extent to which a business, application, or component shall prevent a party to one of its interactions (e.g., message, transaction) from denying having participated in all

ISSN: 2076-3328 www.ijric.org E-ISSN: 2076-3336

or part of the interaction. The typical objectives of a nonrepudiation requirement are to:

• Ensure that adequate tamper-proof records are kept to prevent parties to interactions from denying that they have taken place.

• Minimize any potential future legal and liability problems that might result from someone disputing one of their interactions.

Examples

• "The application shall make and store tamper-proof records of the following information about each order received from a customer and each invoice sent to a customer:

1. The contents of the order or invoice.

2. The date and time that the order or invoice was sent.

3. The date and time that the order or invoice was received.

4. The identity of the customer. "

[8] Privacy Requirements

A privacy requirement is any security requirement that specifies the extent to which a business, application, component, or center shall keep its sensitive data and communications private from unauthorized individuals and programs.

The typical objectives of a privacy requirement are to:

• Ensure that unauthorized individuals and programs do not gain access to sensitive data and communications.

• Provide access to data and communications on a "need to know" basis.

• Minimize potential bad press, loss of user confidence, and legal liabilities.

Examples

• Anonymity. - "The application shall not store any personal information about the users."

• Communications Privacy. - "The application shall not allow unauthorized individuals or programs access to any communications."

• Data Storage Privacy. - "The application shall not allow unauthorized individuals or programs access to any stored data."

[9] Security Auditing Requirements

A security auditing requirement is any security requirement that specifies the extent to which a business, application, component, or center shall enable security personnel to audit the status and use of its security mechanisms.

The typical objectives of a security auditing requirement are to ensure that the application or component collects, analyzes, and reports information about the:

• Status (e.g., enabled vs. disabled, updated versions) of its security mechanisms.

• Use of its security mechanisms (e.g., access and modification by security personnel).

Examples

• "The application shall collect, organize, summarize, and regularly report the status of its security mechanisms including: Identification, Authentication, Authorization, Immunity, Privacy and Intrusion Detection."

[10] Survivability Requirements

A survivability requirement is any security requirement that specifies the extent to which an application or center shall survive the intentional loss or destruction of a component.

The typical objective of a survivability requirement is to ensure that an application or center either fails gracefully or else continues to function (possibly in a degraded mode), even though certain components have been intentionally damaged or destroyed.

Examples

• "The application shall not have a single point of failure."

• "The application shall continue to function (possibly in degraded mode) even if a data center is destroyed."

[11] Physical Protection Requirements

A physical protection requirement is any security requirement that specifies the extent to which an application or center shall protect itself from physical assault.

The typical objectives of physical protection requirements are to ensure that an application or center are protected against the physical damage, destruction, theft, or replacement of hardware, software, or personnel components due to vandalism, sabotage, or terrorism.

ISSN: 2076-3328 www.ijric.org E-ISSN: 2076-3336

Examples

• "The data center shall protect its hardware components from physical damage, destruction, theft, or surreptitious replacement."

• "The data center shall protect its personnel from death, injury, and kidnapping."

[12] System Maintenance Security Requirements

A system maintenance security requirement is any security requirement that specifies the extent to which an application, component, or center shall prevent authorized modifications (e.g., defect fixes, enhancements, updates) from accidentally defeating its security mechanisms.

The typical objective of a system maintenance security requirement is to maintain the levels of security specified in the security requirements during the usage phase.

Examples

• "The application shall not violate its security requirements as a result of the upgrading of a data, hardware, or software component."

• "The application shall not violate its security requirements as a result of the replacement of a data, hardware, or software component."

3. SECURITY REQUIREMENTS ENGINEERING

Software systems become more and more critical in every domain of the human society. Transportation, telecommunications, entertainment, health care, military, and education; the list is almost endless. These systems are used not only by major corporations and governments but also across networks of organizations and by individual users. Such wide use has resulted in these systems containing a large amount of critical information and processes which inevitably need to remain secure. Therefore, although it is important to ensure that software systems are developed according to the user needs, it is equally important to ensure that these systems are secure.

However, the common approach towards the inclusion of security within a software system is to identify security requirements after the definition of a system. This typically means that security enforcement mechanisms have to be fitted into a pre-existing design, leading to serious design challenges that usually translate into the emergence of computer systems afflicted with security vulnerabilities. Recent research has argued that from the viewpoint of the traditional security paradigm, it should be possible to eliminate such problems through better integration of security and requirements engineering. Security should be considered from the early stages of the development process and security requirements should be defined alongside with the system's requirements specification.

Taking security into account alongside the functional requirements helps to limit the cases of security/functional requirements conflict by avoiding them from the very beginning or by isolating them very early in the software system process.

The Security Requirements Engineering is the process of eliciting, specifying, and analyzing the security requirements for system fundamental ideas like "what" of security requirements is, it is concerned with the prevention of harm in the real world and considering them as constraints upon functional requirements.

Many methods have been developed that facilitate this kind of requirements analysis and the development of security requirements. The objective of this paper is to provide an overview of various security requirements engineering methods and to compare them.

3.1 Security Requirements Engineering Methods

Many requirements engineering research projects undertaken in recent years have resulted in the development of methods and processes that can be used in identifying security requirements. These are some of them:

A The Software Engineering Institute's Square (Secure Quality Requirements Engineering

Security Quality Requirements Engineering (SQUARE) is a process aimed specifically at security requirements engineering.
Square is based on interaction between requirements engineers and an IT project's stakeholders, where facilitation by a requirements engineering team is of major importance. [9].

1. Agree on definitions.
2. Identify security goals.
3. Develop artifacts.
4. Perform risk assessment.

ISSN: 2076-3328 www.ijric.org E-ISSN: 2076-3336

5. Select an elicitation technique.
6. Elicit security requirements.
7. Categorize requirements.
8. Prioritize requirements.
9. Inspect requirements

B Haley and his colleagues' framework.

A framework for Security Requirements Engineering has the following 4 activities done in every iteration. [7].
Iteration between requirement and design activities is an important part of the framework. Fulfilling a security requirement might lead to new assets, resulting in new security requirements.

1. Identify functional requirements.
2. Identify security goals—including assets, threats, management principles, and business goals.
3. Identify security requirements.
4. Verify the system.

C Gustav Boström and his colleagues consider security requirements engineering in a different context, agile development with a focus on extreme programming (XP) practices.

This process extends XP user stories to include security requirements. The main ideas should also be relevant for other types of development processes**.** [10]

1. Identify critical assets.
2. Formulate abuser stories.
3. Assess abuser story risk.
4. Negotiate abuser and user stories.
5. Define security-related user stories.
6. Define security-related coding standards.
7. Cross-check abuser stories and countermeasures.

D Clasp is a major initiative for securing software development life cycles.

The Comprehensive, Lightweight Application Security Process (CLASP) approach to security requirements engineering is a life-cycle process that suggests a number of different activities across the development life cycle to improve security. This is a specific approach for security requirements. [11]

1. Document security-relevant requirements (for example, identify business requirements and functional security requirements and dependencies) for determining risk mitigations and resolving deficiencies and conflicts.
2. Identify resources and trust boundaries, such as network-level design and data resources.

E Steve Lipner and Michael Howard describe the Microsoft Trustworthy Computing Security Development Lifecycle.

It is focusing on planning security activities [12, 13, and 14] and they also state that we should identify key security objectives together with security feature requirements that are based on customer demand and compliance with standards. It has the following four steps.

1. Identify use scenarios.
2. Identify assets.
3. Identify threats.
4. Identify dependencies

F Axelle Apvrille and Makan Pourzandi suggest four steps for the security requirements and analysis phase

1. Identify the security environment and objectives.
2. Determine the threat model.
3. Choose a security policy, which includes prioritizing according to the information's sensitivity.
4. Evaluate risk.

They intend that the developers themselves perform these steps, but they don't explain the steps in much detail. [15]

G Security Requirements Engineering Process

The Security Requirements Engineering Process (SREP) [Mellado 2007] is a nine-step process that is based partially on SQUARE but incorporates consideration of the Common Criteria and notions of reuse. [16]

SREP is quite similar to SQUARE.SREP activities:
1. Agree on definitions
2. Identify vulnerable and/or critical assets
3. Identify security objectives and dependencies
4. Identify threats and develop artifacts.
5. Risk assessment
6. Elicit security requirements
7. Categorize and prioritize requirements
8. Requirements inspection
9. Repository improvement

Eduardo Fernandez -use cases are helpful for determining the rights each actor needs and for considering possible attacks. [17]. **Gunnar**

ISSN: 2076-3328 www.ijric.org E-ISSN: 2076-3336

Peterson -use and misuse cases as a basis for security requirements, with additional nonfunctional requirements. [18]. **Kenneth van Wyk and Gary McGraw** suggests using abuse cases. [19] **Core security requirements artifacts** [20] takes an artifact view and starts with the artifacts that are needed to achieve better security requirements. **Security patterns** are useful in going from requirements to architectures and then designs [21, 22, 23].**Tropos** is a self-contained life-cycle approach [24]. It is very specific in terms of how to go about requirements specification. **Formal specification** approaches to security requirements, such as Software Cost Reduction (SCR) [25] have also been useful. The higher levels of the Common Criteria [26] provide similar results.

3.2 Comparing the Methods of Security Requirements Engineering[SRE]

Some of the methods apply to the entire life cycle, not just requirements engineering. Some of them are processes specifically aimed at security requirements engineering, in a similar fashion to SQUARE. A third category encompasses specific methods that could be applied within a variety of processes, including SQUARE. The Tropos material by Giorgini et al. is a self-contained life-cycle approach. If he or she were using Tropos, he or she would use it throughout. CLASP is a life-cycle process that suggests a number of different activities across the development life cycle in order to improve security.

The Core Artifacts approach is not inconsistent with SQUARE, in that the goals and some of the process steps are similar, but it is a different process for arriving at security requirements. Fernandez's use of misuse cases and attack patterns is consistent with SQUARE, as used misuse cases and attack trees as part of the process. However, there is less detail on how to use these specifically in the requirements area than the SQUARE process provides. Weiss's material on security patterns is consistent with SQUARE and could be used as part of the SQUARE process. Specifically, security patterns could be used to help identify and document security requirements. The security patterns described by Rosado fall into the architecture domain and would be most useful once requirements are in place. SREP is quite similar to SQUARE.

Formal methods can be used in the specification and verification of requirements for secure systems. From a life-cycle viewpoint, the specification typically represents either formal requirements or a formal step between informal requirements and design. The elicitation approaches used in SQUARE do not lead directly to formal specifications, but approaches such as Software Cost Reduction or the higher levels of the Common Criteria could be used as follow-on to the SQUARE process when formal specifications are called for.

TABLE I
APPROACHES TO SRE
REQUIREMENTS PHASE TASKS

Approach	Definitions	Objectives	Misuse/ threats	Assets	Coding standards	Categorize & prioritize	Inspect & validate	Process planning
SQUARE	*	*	*			*	*	
Charles Haley and colleagues		*	*	*			*	
Gustav Boström and colleagues			*	*	*	*		
Clasp		*		*			*	
Microsoft		*	*+	*+				*
Axelle Apvrille and Makan Pourzandi		*	*				*	
Eduardo Fernandez			*					
Kenneth van Wyk and Gary McGraw			*					
Gunnar Peterson			*					
SREP	*		*	*				

From the TABLE I we find some limitations with each method. SQUARE requires design work as background for elicitation. Square covers most of the tasks in Table 1. The main SQUARE methodology doesn't include asset identification. Haley and his colleagues suggest artifacts that are probably too complex for regular developers.[27] CLASP does not include detail misuse cases and considers determining risk mitigations to be a requirements activity, while others consider this part of design. Microsoft considers some requirement activities as part of design phase. CLASP and Axelle Apvrille and Makan Pourzandi are more general and they are less concrete. Gustav Boström and colleagues who focus on XP development, the approaches require different

ISSN: 2076-3328 www.ijric.org E-ISSN: 2076-3336

levels of expert knowledge. Microsoft and CLASP are more lightweight.

3.3 Trends in Security Requirements Engineering

Many organizations are realizing that security requirements need to be addressed early in the lifecycle process. It is a very active research area, with a wide variety of methods and tools under development. Some organizations, such as Microsoft, already have security requirements engineering methods incorporated into their life-cycle processes. At present, there is no consensus on a single best approach to security requirements engineering. However, many organizations intuitively feel that attention to this area will pay off in supporting their business goals. There are a number of conferences and workshops that have been held over the last few years on the subject of security requirements. This is a trend that is likely to continue, as additional workshops of this type are already showing up on the calendar. Another trend, which is somewhat unfortunate, is that many industrial organizations feel that their internal processes give them a competitive edge, so they are unwilling to publish or discuss the details. It was surprising to find that many organizations seem to have established processes for engineering security requirements when so few have published their methods.

4. CONCLUSION

In this paper we discussed about, Security Requirements and its types, Security Requirements Engineering and its need is given. All the approaches of Security Requirements Engineering and its limitations were identified. In short, there is no single right answer when it comes to security requirements engineering. A lot depends on the processes that are already in place in a particular organization. Some organizations may prefer a detailed, specific method, whereas other organizations may prefer an approach that allows them to select methods to incorporate into existing processes. Another factor is the extent to which the project or organization is mission critical. This can dictate the level of formality used in requirements engineering and the need for assurance levels of security. In future with these approaches we have planned to apply to analyze the security requirements for banks and web applications. The other future work is to develop a framework that implements all the requirements phase tasks for information security. We have also planned to develop a tool to generate test cases from the security requirements.

REFRENCES:

[1] R. Anderson. Security Engineering: A Guide to Building Dependable Distributed Systems.Wiley Computer Publishing, 2001.

[2] J. Viega and G. McGraw. Building Secure Software. Addison-Wesley, 2001.

[3] R. Crook, D. Ince, L. Lin, and B. Nuseibeh. Security Requirements Engineering: When Antirequirements Hit the Fan. In Proc. of RE'02, pages 203–205. IEEE Press, 2002.

[4] P. T. Devanbu and S. G. Stubblebine. Software engineering for security: a roadmap. In Proc. of ICSE'00, pages 227–239, 2000.

[5] A. van Lamsweerde, R. Darimont, E. Letier, "Managing Conflicts in Goal-Driven Requirements Engineering", IEEE Transactions on Software Engineering, Nov. 1998, 908-926.

[6] W. N. Robinson, "Requirements Interaction Management",*ACM Computing Surveys*, June 2003.

[7] C.B. Haley, R. Laney, J.D. Moffett, and B. Nuseibeh, "Security Requirements engineering: A Framework for Representation and Analysis," IEEE Transaction on Software Eng. Vol 34, no. 1, pp. 133-152, Jan/Feb 2008.

[8] Donald Firesmith: "Engineering Security Requirements", in Journal of Object Technology, vol. 2, no. 1, January-February 2003, pages 53-68. http://www.jot.fm/issues/issue_2003_01/column6

[9] N.R. Mead, E.D. Houg, and T.R. Stehney, Security Quality Requirements Engineering (Square) Methodology,tech. report CMU/SEI-2005-TR-009, Software Eng. Inst., Carnegie Mellon Univ., 2005.

[10] G. Boström et al., "Extending XP Practices to Support Security Requirements Engineering," Proc. 2006 Int'l Workshop Software Eng. for Secure Systems (SESS), ACM Press, 2006, pp. 11–18.

[11] Graham, Dan. "Introduction to the CLASP Process." *Build Security In*, 2006. https://buildsecurityin.us-cert.gov/daisy/bsi/ articles/ best-practices/requirements/548.html.

[12] P. Torr, "Demystifying the Threat Modeling Process," IEEE Security & Privacy, vol. 3, no. 5, 2005, pp.66–70.

ISSN: 2076-3328 www.ijric.org E-ISSN: 2076-3336

[13] S. Lipner and M. Howard, "The Trustworthy Computing Security Development Lifecycle," Microsoft Corp.,2005; http://msdn2.microsoft.com/en-us/library/ms995349.aspx.

[14] J.D. Meier, "Web Application Security Engineering," IEEE Security & Privacy, vol. 4, no. 4, 2006, pp.16–24.

[15] A. Apvrille and M. Pourzandi, "Secure Software Development by Example," IEEE Security & Privacy, vol. 3, no. 4, 2005, pp. 10–17.

[16] Mellado, D.; Fernandez-Medina, E.; & Piattini, M. "A Common Criteria Based Security Requirements Engineering Process for the Development of Secure Information Systems." *Computer Standards & Interfaces 29*, 2 (February 2007): 244-253.

[17] E.B. Fernandez, "A Methodology for Secure Software Design," paper presented at the Int'l Symp. Web Services and Applications (ISWS), 2004; www.cse.fau.edu/~ed/EFLVSecSysDes1.pdf.

[18] G. Peterson, "Collaboration in a Secure Development Process Part 1," Information Security Bull., June 2004,pp. 165–172.

[19] K.R. van Wyk and G. McGraw, "Bridging the Gap between Software Development and Information Security," IEEE Security & Privacy, vol. 3, no. 5, 2005, pp. 75–79.

[20] Moffett, J.D.; Haley, C.B.; & Nuseibeh, B. *Core Security Requirements Artefacts* (Technical Report 2004/23, ISSN 1744-1986). Open University, 2004.

[21] Haley, C.; Laney, R.; Moffett, J.; & Nuseibeh, B. "Arguing Satisfaction of Security Requirements," 16-43. *Integrating Security and Software Engineering*. Edited by H. Mouratidis and P.Giorgini. Hershey, PA: Idea Group Publishing, 2007 (ISBN 1-599-04147-2).

[22] Rosado, David G.; Gutiérrez, Carlos; Fernández-Medina, Eduardo; Piattini, Mario. "Security Patterns and Requirements for Internet-Based Applications." Internet Research 16, 5 (2006): 519-536.

[23] Weiss, M. "Modelling Security Patterns Using NFR Analysis," 127-141. Integrating Security and Software Engineering. Edited by H. Mouratidis and P. Giorgini. Hershey, PA: Idea Group Publishing, 2007 (ISBN 1-599-04147-2).

[24] Giorgini, P.; Mouratidis, H.; & Zannone, N. "Modelling Security and Trust with Secure Tropos," 160-189. *Integrating Security and Software Engineering*. Edited by H. Mouratidis and P. Giorgini. Hershey, PA: Idea Group Publishing, 2007 (ISBN 1-599-04147-2).

[25] Heitmeyer, C. "Software Cost Reduction." *Encyclopedia of Software Engineering*, 2nd ed. Edited by John J. Marciniak. New York, NY: John Wiley and Sons, 2002 (ISBN 978-0-471-37737-6).

[26] The Common Criteria Evaluation and Validation Scheme. http://www.niap-ccevs.org/cc-scheme/ (2007).

[27] Inger Anne Tøndel, Martin Gilje Jaatun, and Per Håkon Meland, "Security Requirements for the Rest of Us:A Survey " IEEE Software Published by the IEEE Computer Society, 0740 - 7459 / 08 / 2008 IEEE

AUTHOR PROFILES:

P.Salini is from Puducherry, India born in 1981.She received her B.Tech degree in Information Technology and M.Tech in Computer Science and Engineering from Pondicherry University and now she is doing her Ph.D in Computer Science and Engineering, from Pondicherry Engineering College affiliated to Pondicherry University.

In 2005 she joined as a Lecturer in Department of Information Technology in a Private Engineering College. Now she is working as a Assistant Professor in Department of Computer Science and Engineering, Pondicherry Engineering College. Her research interests are in Software Engineering, Security Engineering and Requirements Engineering. She is a member of ISTE.

Dr. S.Kanmani received her B.E and M.E in Computer Science and Engineering from Bharathiar University and Ph.D from Anna University, Chennai.

She has been the faculty of Department of Computer Science and Engineering, Pondicherry Engineering College from 1992. Presently she is a Professor in the Department of Information Technology. Her research interests are in software Engineering, Software Testing and Object Oriented Systems. She is a member of computer society of India, ISTE and Institute of Engineers, India. She has published about 50 papers in international conferences and journals**.**