Security Requirements Engineering; State of the Art and Research Challenges

M. A. Hadavi, V. S. Hamishagi, H. M. Sangchi

***Abstract*— In recent years software has faced a new challenge called security. The new idea in software security which has attracted the world's attention is to keep security in mind during development process. As requirements analysis plays an infrastructural role in this process, software security requirements would naturally be considered fundamental in secure software development. Stating peculiarities and deficiencies in security requirements engineering, this paper draws a picture from the current research situation by reviewing and classifying the efforts into four main categories; security requirements in the standard software development processes, security requirements engineering consist of eliciting and modeling security requirements, and threat modeling as a basis for security requirements engineering. Presenting challenges and open problems for each category, the paper will then set forth the research outlooks and future directions in security requirements.**

***Index Terms*—requirements engineering, security, software development process, threat modeling.**

I. Introduction

Today, the problems in developing secure software are exacerbated due to networks expansions which have led to connection of software to the internet, system extensibility to adapt with its environment in order to address diverse user requirements, and ongoing increase in system complexity and inflation. Such problems will waste so many resources, in terms of time and money, accepting the overhead of developing a secure application is not only optional, but also an essentially mandatory property [1].

The problem is that existing methods of developing secure software usually do not satisfy the requirements about security threats, risk assessment, security mechanisms and finally a systematic process for software security [11].

Determining and analyzing the requirements will result in a concrete base in software lifecycle. Outdated methods of software development divided the software requirements into two categories: functional and physical requirements. Functional requirements described system objectives whereas physical requirements stated the physical obligations like time and place. At present, this categorization has changed to functional and non-functional requirements [22]. Functional requirements are those that their fulfillment by the system is mandatory. They can be obtained through understanding the way system elements interact with their environment. It should be possible to evaluate their fulfillment through some test cases. Non-functional requirement focus on the way system functions are performed. In the other words, non-functional requirements impose some constraints to the way system functions are performed. Security is considered as a non-functional requirement. In recent years, tremendous growth in networks from one side and importance of information from the other side has contributed heavily to the importance of security requirements (SR). Adding security to software requirements indicates that security has been considered from the very first step of software development. SR objectives can be categorized as authentication, authorization, integrity, intrusion detection, non-repudiation, confidentiality and auditing [29].

This paper is organized into six sections. The next section discusses research issues in the context of SR. The third section points where SR stand in software development process. SR eliciting and modeling discuss in the forth section. Fifth section details the relationship between threat modeling and risk management. Concluding the discussions, the sixth section will describe the available gaps and research trends in this field.

II. SR Research Issues

There have been numerous research activities in different contexts of software SR. The most important of which is to develop methods of eliciting and modeling SR. As we all know, use cases are among the widely accepted methods of eliciting, documenting and analyzing functionality requirements of systems. Equivalently, there are Misuse Cases and Abuse cases which are accepted as systematic means used for SR. They have attracted a considerable amount of research activities which include specifying SR based on misuse cases [19], [33]–[37], applying abuse cases in developing SR [38], analyzing requirements based on preplanned scenarios [32] and methods of specifying user requirements based on business goals [17], [39]. Some other research activities have concentrated on application of existing security standards [22], [23]. In the forth section, we will elaborate the current status of research about eliciting and modeling SR.

Another part of research is dedicated to threat modeling and software risk assessment. Threat modeling is the basis for secure system development and helps in more accurate and complete determination of SR. While engineering system requirements, risk assessment will take decision about the methods of countering each threat. Threat modeling will be elaborately discussed in the fifth section.

Integration of security within software development especially during requirement engineering, is considered as another research challenge. One reason for these challenges, beside technical issues, is dependence of security on organizational policy [26]. So far, there have been some

efforts to integrate SR to standard procedures of obtaining requirements. Processes like SecureTROPOS [42]–[44] and CALSP [27] fall in this category. In the next section, we will elaborate the current status of research in this field.

Also, there has been some research on safety techniques among software security issues. Safety and security are counted as quality parameters as subsets of defensibility quality parameters [31], [41]. The main difference between security and safety is that safety addresses accidental loss whereas security deals with maliciously incurred loss [21]. It is possible that a system include vulnerabilities, like lack of input validation, that cause both accidental loss and successful attacks. In such situations, we can benefit from controls that address both kinds of vulnerabilities. In safety context, these controls are called safeguards and countermeasures in security context. Contrary to security engineering, safety engineering deals less with requirements as compared with architecture, design, implementation and test. Mentioning the similarities, Nancy Leveson and Mats Heimdahl have worked around this idea [40].

III. SR in Software Development Process

Common software engineering processes do not satisfy security requirements. There are at least two reason that system engineering does not support, or weakly support, SR; first, SR usually are not meant for simple analysis and modeling. The other reason is unwillingness or lack of expertise in developers to produce secure software. The developer's unwillingness stems from the new features that must be added to software. Also, considering security features will delay in the software release time [26].

However, some activities have been performed on the Tropos methodology. It is a formal methodology to produce secure software taking into consideration both the system and its operational environment. This methodology emphasizes on basic requirement analysis as well as the way system can satisfy organizational policy [2]. In order to produce secure software, the improved version of this methodology, called SecureTropos, was proposed [42]–[44].

CLASP, a plug-in to RUP, is another well defined and structured method to consider security in the very first step of software lifecycle. CLASP fully supports UML 2.0 in the entire software development lifecycle. This process was created by Secure Software Inc. in 2004 [27].

Some of the research activities in this field deal with formal methods in secure system engineering. Formal description of security protocols [4] and presenting a semi-formal method for validation of SR [16] are among these activities. Yet, improving in this field requires a lot of research efforts .On the other hand, these kinds of methods can be implemented only by security experts because security policies are usually determined by special security models that are not yet integrated into software engineering models [3]. Apart from these problems, from software engineering point of view, ideal software omits manual intervention and involves only formal methods in each phase of software development (including security analysis and modeling). Hence, integrating security within software development, especially in requirement analysis as the basis of software development, is considered as one of today's research challenges [26].

IV. SR Engineering

Requirements engineering is the branch of software engineering concerned with the real-world goals for, functions of, and constraints on software systems. It is also concerned with the relationship of these factors to precise specifications of software behavior, and to their evolution over time and across software families [45].

Although above definition concentrates on software engineering, it is not possible to evaluate the software separately from its operating environment. In that case, requirements engineering will be a multidisciplinary human centered process, though we can benefit from some tools and techniques beside the human expertise [12].

Requirement engineering process includes obtaining, modeling, analyzing and extending the requirements [12]. There are some inherent problems in the process. Requirements of different types of system users including customers, developers and system owners vary from one to another. They have different or even contradictory goals. Those goals, though unavoidable, might not be easily expressible. Perhaps, satisfaction of some requirements would result into other uncontrollable limitations [12]. These problems in specifying SR draw more attention due to lack of experience, expertise, techniques and tools.

Lack of an explicit, or even implicit, security policy is another problem in identifying SR. Though there are various kinds of security policies ranging from old models like Bell La Padula to newer ones like RBAC, many systems do not systematically apply them. This is because the requirement modeling methods for producing organizational procedures from above policies are not sufficient. Another point to note is that design mechanisms are not expressed in place of requirements. Normally those who are responsible for determining SR, impose a series of usually unnecessary limitations [29]. While determining system requirements (not only SR), determining design related mechanisms (like using user-id/ password for authentication) should be avoided. In simple words, SR does not state how security is to be implemented. It only expresses security needs [25].

One of the resources from which SR are obtained is attack patterns. In [14] there exist a method for determining and specifying SR using attack patterns. Also System Security Engineering-Capability Maturity Model (SSE-CMM) has given some guidelines for producing SR [30]. Following this part of the paper, we will describe some important research performed in the field of requirements elicitation.

In [18] a structured method for extracting SR based on system resources is presented within the following four steps:

1) Determining system roles and resources.
2) Resource classification in conceptual level.
3) Determining interaction between system resources.
4) Determining mechanisms appropriate for security services including authentication, authorization, availability, integrity and audit-ability (for all resources).

Security standards are other appropriate resource for developing SR. BS7799, published in 1995, concentrates on

security of information technology and is divided into two parts. Later, the first part was turned to ISO 17799 and the second part to ISO 27001 in 2005. ISO 17799/27001 manage organization security with the best practices. They can be used to capture SR using organizational policies [17]. Common Criteria (CC), which is known as ISO/IEC 15408, is used for security evaluation of software products and consequently evaluating their SR. The products are evaluated against a certain Evaluation Assurance level (ELA) ranging from ELA1 to ELA7 [47]. Reference [23] has given a solution for capturing SR of IT systems using CC methodologies.

A. Misuse and Abuse Case vs. Use Case

Misuse Case as negative scenarios or *hostile Use Cases* presents a new solution to obtaining SR [24], [25], [37]. *Abuse Case* is also derived from misuse case that describes a malicious or intentional use of the system. It demonstrates the interaction of the system elements that would cause system or resource damage [5]. Like every use case that presents a functional requirement of a system, each misuse and abuse case determine a security requirement and test the system against that requirement [10]. McDermott and Fox involved Use cases in the process of obtaining and analyzing system requirements and map them to abuse cases [48]. Sindre and Opdahl defined Misuse case contrary to use case as "The behavior system should not have". In their solution, SR are considered as analysis of system Misuse cases [36], [37].

In addition to use case and misuse case, *Mitigation Case* is also used for expressing SR [6]. Mitigation case states the requirements and countermeasures that must be taken against misuse case offensive scenarios. In this process, the main use cases and misuse case are first produced. Then each of these cases is split into a number of small cases. For each new misuse case, a mitigation case is introduced as a requirement. Firesmith believed that "a misuse case is an effective method for analyzing security threats, but might not be well used for determining and analyzing security requirements". So he introduced *Security Use Case* and integrated it with UML diagrams [24]. There are other researches around Use and Misuse cases in [33], [34], [39] that can be used to produce and model functional and SR.

Using attack patterns is another approach for specifying misuse cases. Exploiting Software [46] has described 48 attack patterns. In similar approach, Crook has introduced a*nti-requirement* for expressing malicious user or attacker's requirement [13]. An anti-requirement is met when a security threat is imposed by an attacker and puts the system assets into one or more of the variety of risks.

In [14] it is mentioned that the best practical solution to produce use case and misuse case is to brainstorm the subject by software security experts. This is because the theoretical solutions that require specifying, in its entirety, the system features, rely on formal methods and logical models that are very expensive as well as time consuming. In such a brainstorm, various issues are considered. Some of them include user interfaces, environmental factors and all the actions that developers do not assume users to take (like user cannot enter more than 50 characters, user will not recognize the contents of cached data, or user cannot change cached data). It is worth mentioning that attackers frequently exploit such wrong assumptions.

Another important issue is how misuse case can help in designing and implementing a more secure system. Basically, the process of designing a system, even while having definite requirements, is innovatory and therefore, it is possible for a system to have different acceptable designs. The idea of designing system architecture from use cases was first introduced in [12]. It almost filled the gap between system requirements and architecture. Reference [50] shows that requirements and architecture have a mutual influence and it's better to be developed parallel to each other. This point is clearly felt especially regarding SR [49]. In [7]–[9], the SR are discussed using misuse case. Then considering the type of the threat presented by a misuse case, system architecture is proposed in its totality, followed by a method for architecture analysis and evaluation. Also, [5] presents a method for analyzing and designing secure software system architecture which is based on the requirements obtained from use and misuse cases.

B. SR Modeling

The efforts in the field of SR modeling can be divided into two categories; activities that try to model security attributes within available frameworks, and activities that have improved existing frameworks that accept security attributes.

In the first category, frameworks like i*/Tropos, KAOS and UML are invoked and some SR are modeled. Then, analytical features of framework are used to obtain design and implementation reasoning and guidelines. In this method, SR analysis is a low cost process because neither a new language, nor a new framework for modeling and analyzing is required. Depending on the capabilities of the framework, one can benefit from formal features or available reasoning procedures for modeled SR of the framework. In these types of modeling, security concepts and other types of requirements are not separate from each other. Thus the system analyst and designer must find and consider the relation between SR and functional requirements. In this context, Liu et al. used i*/Tropos for dealing with security and privacy requirements [38]. Also, in [51] a goal driven framework was used for modeling confidentiality requirements in a role based engineering process. In this framework, the privacy requirements are modeled as contexts and constraints of permissions and roles.

In the second category of SR modeling activities, the existing modeling frameworks are improved with a new conceptual structure in order to address a wider range of SR. Therefore the rest of framework capabilities including analytical features or design and implementation guidelines should be revised or improved for compliance with new SR. Adding new structures to the model, though difficult, will convey more complete and accurate SR and, consequently, system analysis. Such activities have been performed especially around UML methodology. UMLsec[54], introduced by Jurjens, is a security improvement of UML. It models security features like confidentiality and access control. UMLsec includes all UML Analysis and Design artifacts like activity diagrams, deployment diagrams, sequence diagrams and statecharts. The design of the system

can thus be represented in UMLsec. The SR on the UMLsec models are then verified by model checking [4]. Research and case studies on UMLsec have focused on using UMLsec to model cryptographic systems and specify their properties such as secrecy, authentication, integrity, non-repudiation, fair exchange, electronic commerce systems and information flow. Generation of code or application infrastructure and of test cases, their adequacy, etc. are open questions and areas of prospective research [4]. Lodderstedt et al. improved UML and presented secureUML for modeling role based access control policies [52]. Also, Doan added Mandatory Access Control to UML [53].

We can conclude that modeling SR should either invoke existing frameworks, along with their analysis capabilities, or improve these frameworks to accept SR. Choosing between these two options depend on the trade off between usage simplicity and the ability to address SR.

V. Threat Modeling

A threat is defined as an attacker's target or what for which an attacker may act [55]. Threat modeling is an engineering approach used for specification of SR. It enables the application development team, armed with the knowledge of the attacker's approach, to protect the software from various attacks. In the other words, a threat model presents the attacker's view of the system and permits the system developers to design defense scenarios against that view. Also, threat modeling can be helpful while testing the system to evaluate its security and examining the mechanisms used in design and implementation. Howard and Leblanc introduced STRIDE[1] and DREAD[2] threat models. STRIDE is used in threat identification and DREAD for prioritizing threats to be protected against [1]. In [20], threat modeling based on system assets are performed in three steps:

1) Determining system specifications
2) Identifying assets and access points in the system
3) Identifying threats to each asset

First step is to model the via understanding the system elements and their interactions. DFD[3] can be used in this process. In the second step, assets are the abstract and inherent resource of the system that has to be protected against malicious acts. Processes, data and abstract concepts can all be regarded as system resources. Basically, the resources are the targets of attacks. Access point is the area where attackers use to access the resources. Open sockets, RPC interfaces, configuration files, hardware ports and system file read/write permissions are among the various types of access points. For the third step, one can use the list of threats and vulnerabilities identified in another similar system. Identifying system specific threats requires a more in-depth analysis of the modeled system.

In [6], that functional requirements are modeled with use case and SR with misuse case, STRIDE is used for threat modeling.

[1] Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, Elevation of privilege

[2] Damage potential, Reproducibility, Exploitability, Affected users, and Discoverability

[3] Data Flow Diagram

A. Risk Management

Risk is the probability of threat occurrence [28]. We've heard for many times that 100% security cannot be guaranteed. However, we can work toward 100% risk acceptance i.e., the risk resulting from unintentional events can be reduced to an acceptable level [15]. In simple words, there must be a balance established between what can be done and what can be accepted. This balance is achieved through the process of risk management. In order to assess the risk resulting from threats, they must be prioritized. Usually, damage severity and occurrence probability are the best priority factors, multiplication of which will indicate the risk of each threat [21]. After arranging the threats in ascending order, they are evaluated and decided upon. There are four options for risk management [20]:

1) Risk acceptance: in situations where the risk level is low but its cost of protection is relatively high
2) Risk transfer: Transfer the risk to somebody else via insurance, warnings etc.
3) Risk removal: the entity or attribute that has caused the risk is removed, if worth venturing.
4) Risk mitigation: reduce the risk level using countermeasures.

In the "threat modeling" book, written by swiderski and synder, the above mentioned solutions are described in detail [55]. However, risk identification is a manual process that uses a set of guidelines related to common vulnerabilities. Still there is no standard method of software risk assessment. In [28] some risk measurement techniques for software security are discussed.

VI. Conclusion and Future Directions

The SR, which are categorized as non-functional requirements, play a fundamental role in the process of developing secure software. The research activities performed in this area mainly focuses on the methods of eliciting, modeling and analyzing such requirements.

In this paper, we discussed the research activities performed in these areas. Fig. 1 illustrates those activities in five categories. In this figure, each category consists of the most important research and appropriate methods in its parent context of SR engineering.

The solutions described in this paper are the first steps in integrating security into the software development process. Each of these solutions, targets a specific section of SR engineering process. For example, Fox and McDermott solution [54] is used only in security analysis step and UMLsec concentrates only on access control policies and the way they are modeled in software development process. Thought such analyses are necessary, they cover only some part of the work which is restricted to the modeling process. Building integrated tools for capturing, modeling and analyzing SR through standard procedures could be one of the future directions.

Unfortunately the scope of activities performed in modeling and analysis, address only the system level requirements and neglect organizational policies. But the recent researches indicate that the main anxiety results not from the external but internal attacks [26]. Thus the need for

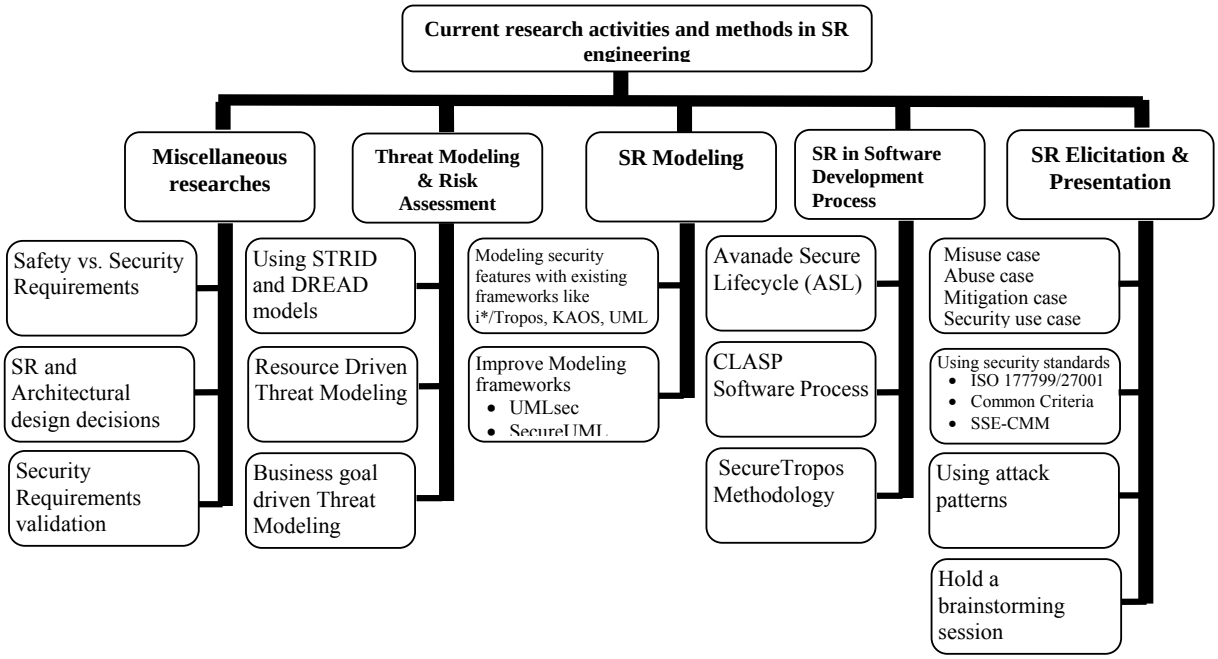

Fig.1- Categorization of research activities and current methods in SR engineering

model for analyzing and designing organizational structure, with security in mind, is obvious. Developing an anthology for assets, from organizational point of view, can be defined as a future research activity. This can help solving one of the important security challenges in organizational level.

Computer science also plays an important role in requirement engineering. Although the formal description techniques have attracted a lot of attention in security engineering research, there is still a long way to their final acceptance into the requirements engineering process [12]. The requirements engineering process must fill the gap between the real world of user requirements and the formal world of software behavior. In that case, we should answer a key question: when to use formal methods? Development of new techniques for formal modeling and analyzing environmental features, in regard with system behavior, can be a future perspectives of SR engineering [12]. As another problem, we could mention the improvement of analysis methods for establishing a balance between conflicting functional and SR which are seen in the real world.

Another research area is to develop a correct conception from the mutual influence between SR and system architecture. So far, there has been a major emphasis on software architecture with regard to providing attributes appropriate for system behavior. Nevertheless, it has to be considered that different options in software architecture can be influential in improvement and growth of requirements especially non-functional ones like security. From this point of view, architecture analysis and system design can be viewed as an open problem.

At the end, It should be mentioned that there are many questions present as to whether the security should be categorized into non-functional requirements. Isn't secure information transfer an application's function? Though these questions cannot be answered definitely but it seems that by considering security as a functional requirement, software security will face a drastic evolvement [4].

REFERENCES

[1] Nancy R. Mead, Gary McGraw, "From the Ground Up: The DIMACS Software Security Workshop", IEEE SECURITY & PRIVACY, PUBLISHED BY THE IEEE COMPUTER SOCIETY, 2003.

[2] J. Haralambos Mouratidis, Paolo Giorgini, Gordon Manson, and Eder, M. Missikiff (Eds.), "When security meets software engineering: A case of modeling", Advanced Information Systems Engineering, Springer LNCS 2681, 2003.

[3] Gerald Brose, Manuel Koch, Klaus-Peter Löhr, "Integrating Security Policy Design into the Software Development Process", Technical Report B–01–06, Institut für Informatik, Freie Universität Berlin, Germany, November 13, 2001.

[4] Jayaram K R and Aditya P. Mathur, "Software Engineering for Secure Software - State of the Art: A Survey", technical Report, Department of Computer Science, Purdue University, September 19, 2005.

[5] Joshua Pauli, Dianxiang Xu, "Misuse Case-Based Design and Analysis of Secure Software Architecture", International Symposium on Information Technology: Coding and Computing (ITCC 2005), Volume 2, 4-6 April 2005, Las Vegas, Nevada, USA. IEEE Computer Society 2005.

[6] Josh Pauli, Dianxiang Xu, "Integrating Functional and Security Requirements with Use Case Decomposition", 11th IEEE International Conference on Engineering of Complex Computer Systems (ICECCS 2006).

[7] Josh Pauli, Dianxiang Xu, "Trade-off Analysis of Misuse Case-based Secure Software Architectures: A Case Study", In Proceedings of the 3rd International Workshop on Modeling, Simulation, Verification and Validation of Enterprise Information Systems (MSVVEIS'05), 2005.

[8] Joshua Pauli, Dianxiang Xu, "THREAT-DRIVEN ARCHITECTURAL DESIGN OF SECURE INFORMATION SYSTEMS", proceedings of the 7th International Conference on Enterprise Information Systems (ICEIS 2005), Miami, USA, May 2005.

[9] Dianxiang Xu, Joshua J. Pauli1, "Threat-Driven Design and Analysis of Secure Software Architectures", Proc. of the 7th International Conference on Enterprise Information Systems (ICEIS 2005), May 2005, USA, to appear in Journal of Information Assurance and Security.

[10] R. KENNETH, VAN WYK, GARY MCGRAW, "Bridging the Gap between Software Development and Information Security", PUBLISHED BY THE IEEE COMPUTER SOCIETY, IEEE SECURITY & PRIVACY, 2005.

[11] Ivan Flechais, M. Angela Sasse, Stephen M. V. Hailes, "Bringing Security Home: A process for developing secure and usable systems", NSPW '03, Ascona, Switzerland, ACM, August 18-21, 2003.

[12] Bashar Nuseibeh, Steve Easterbrook, "Requirements Engineering: A Roadmap", Proceedings of International Conference on Software Engineering (ICSE-2000).
[13] Robert Crook, Darrel Ince, Luncheng Lin, Bashar Nuseibeh, “Security Requirements Engineering:When Anti-requirements Hit the Fan”, Proceedings of the IEEE Joint International Conference on Requirements Engineering (RE’02), 2002.
[14] PACO HOPE, GARY MCGRAW, ANNIE I. ANTO´N, "Misuse and Abuse Cases: Getting Past the Positive", IEEE SECURITY & PRIVACY, PUBLISHED BY THE IEEE COMPUTER SOCIETY, 2004.
[15] Howard Chivers, "Security and Systems Engineering", technical report, Department of Computer Science, University of York, Heslington, York, 2004.
[16] Charles B. Haley, Jonathan D. Moffett, Robin Laney, Bashar Nuseibeh, “Arguing Security: Validating Security Requirements Using Structured Argumentation”, Symposium on Requirements Engineering for Information Security(SREIS'05), In conjunction with RE 05 - 13th IEEE International Requirements Engineering Conference, Paris, France, August 29th, 2005.
[17] Xiaomeng Su, Damiano Bolzoni, Pascal van Eck, Roel Wieringa, “A Business Goal Driven Approach for Understanding and Specifying Information Security Requirements”, In Proceedings of the the Eleventh International Workshop on Exploring Modeling Methods in Systems Analysis and Design (EMMSAD), Luxembourg, June 2006.
[18] John Viega, “Building Security Requirements with CLASP”, International Conference on Software Engineering, Proceedings of the 2005 workshop on Software engineering for secure systems(SESS’05), ACM Press, USA, 2005.
[19] Ian Alexander, “Misuse Cases: Use Cases with Hostile Intent”, Journal of IEEE Software, Published by the IEEE Computer Society, 2003.
[20] Suvda Myagmar, Adam Lee, William Yurcik, “Threat Modeling as a basis for security requirements”, Symposium on Requirements Engineering for Information Security(SREIS'05), In conjunction with RE 05 - 13th IEEE International Requirements Engineering Conference, Paris, France, August 29th, 2005.
[21] Donald G. Firesmith, “Analyzing the Security Significance of System Requirements”, Symposium on Requirements Engineering for Information Security(SREIS'05), In conjunction with RE 05 - 13th IEEE International Requirements Engineering Conference, Paris, France, August 29th, 2005.
[22] John Wilander, Jens Gustavsson, “Security Requirements - A Field Study of Current Practice”, Symposium on Requirements Engineering for Information Security (SREIS'05), In conjunction with RE 05 - 13th IEEE International Requirements Engineering Conference, Paris, France, August 29th, 2005.
[23] Ware M.S, Bowles J.B, Eastman C.M, “Using the Common Criteria to Elicit Security Requirements with Use Cases”, SoutheastCon, Proceedings of the IEEE, 2006.
[24] Donald G. Firesmith, “Security Use Cases”, Journal of Object Technology, vol 2, no 3, May-June 2003, pp 53-64, also available at: http://www.jot.fm/issues/issue_2003_05/column6
[25] Charles B. Haley, Robin C. Laney, and Bashar Nuseibeh, "Deriving Security RequireJournalments from Crosscutting Threat Descriptions," in Proceedings of the Third International Conference on Aspect-Oriented Software Development (AOSD'04), ACM Press, 22-26 Mar 2004.
[26] Paolo Giorgini, Fabio Massacci, Nicola Zannone, “Security and Trust Requirements engineering”, Foundations for Security Analysis and Design, Lecture Notes in Computer Science, Volume 3655, Berlin: Springer, 2005.
[27] Viega J. “The CLASP Application Security Process”. Volume 1.1. Training Manual. Secure Software Inc. 2005.
[28] Denis Verdon, Gary McGraw, “Risk analysis in software design”, IEEE Security and Privacy, 2004.
[29] Firesmith D. G, “Engineering Security Requirements”, In Journal of Object Technology, vol 2, no 1, January-February, 2003, Also Available at: http://www.jot.fm/issues/issue_2003_01/column6
[30] M. Phillips, “Using a Capability Maturity Model to Derive Security Requirements”, SANS InfoSec Reading Room, GSEC Practical v1, March 13, 2003, Also available at: www.sans.org/rr/whitepapers/bestprac/1005.php
[31] D.G. Firesmith, “Common Concepts Underlying Safety, Security, and Survivability”, Technical Note, CMU/SEI-2003- TN-033, Software Engineering Institute, Pittsburgh, Pennsylvania, December 2003.
[32] K. Allenby, T. Kelly, “Deriving Requirements Using Scenarios”, In Proceedings of the 5th IEEE International Symposium on Requirements Engineering (RE'01), Aug 2001.
[33] I. Alexander, “Misuse Cases Help to Elicit Non Functional Requirements”, Computing & Control Engineering Journal, vol. 14, no. 1, pp. 40-45, Feb. 2003.
[34] I. Alexander, “Modeling the Interplay of Conflicting Goals with Use and Misuse Cases”, In Proceedings of 8th International Workshop on Requirements Engineering: Foundation for Software, September 2002.
[35] I. Alexander, “Initial Industrial Experience of Misuse Cases”, Proceedings of IEEE Joint International Requirements Engineering Conference, pp. 61-68, 2002.
[36] G. Sindre, A. L. Opdahl, “Eliciting security requirements with misuse cases”, Requirements Eng, 10(1):34–44, 2005.
[37] G. Sindre, A.L. Opdahl, “Templates for Misuse Case Description”, In Proceedings of the 7th International Workshop on Requirements Engineering, Foundation for Software Quality (REFSQ’2001), June 2001.
[38] L. Liu, E. Yu, J. Mylopoulos, “Security and Privacy Requirements Analysis within a Social Setting”, In Proceedings of the 11th IEEE International Requirements Engineering Conference (RE'03), pp. 151-161, 2003.
[39] I. Jacobson, M. Christerson, P. Jonsson, G. Overgaard, “Object-Oriented Software Engineering: A Use Case Driven Approach”, Addison Wesley, 1994.
[40] N. Leveson, M. Heimdahl, “New Approaches to Critical-Systems Survivability: Position Paper,” 1998 Information Survivability Workshop (ISW’98), IEEE, 28-30 October 1998.
[41] S. Brostoff, M. Sasse, “Safe and Sound: a Safety Critical Approach to Security”, New Security Paradigms Workshop’01, 11-13 September 2001.
[42] H. Mouratidis, P. Giorgini, G. Manson, “Modeling secure multi agent systems”, In Proceedings of AAMAS’03, pages 859–866. ACM Press, 2003.
[43] P. Giorgini, F. Massacci, J. Mylopoulos, N. Zannone, “Requirements Engineering meets Trust Management: Model,Methodology, and Reasoning”, In Proceedings of iTrust’04, LNCS 2995, pages 176–190. Springer-Verlag, 2004.
[44] P. Giorgini, F. Massacci, J. Mylopoulos, N. Zannone, 'Filling the gap between Requirements Engineering and Public Key/Trust Management Infrastructures”, In Proceedings of EuroPKI’ 04, LNCS 3093, pages 98–111, Springer-Verlag, 2004.
[45] P. Zave, “Classification of Research Efforts in Requirements Engineering”, ACM Computing Surveys, PP. 315-321, 1997.
[46] Greg Hoglund, Gary McGraw, “Exploiting Software: How to Break Code”, Published by Addison-Wesley Professional, ISBN: 0201786958, February 2004.Bowman, B., Debray, S. K., and Peterson, L. L. Reasoning about naming systems. ACM Trans. Program. Lang. Syst., 15, 5 (Nov. 1993), 795-825.
[47] National Institute of Standards and Technology, “Common criteria for information technology security evaluation (CC 2.1)”, also available at: http://csrc.nist.gov/cc/CC-v2.1.html
[48] McDermott J., Fox C., “Using Abuse Care Models for Security Requirements Analysis”, Proceedings of the 15th Annual Computer Security Applications Conference, December 1999.
[49] J. D. Moffett, B. Nuseibeh, “A Framework for Security Requirements Engineering”, Department of Computer Science, YCS368. University of York, UK, August 2003.
[50] B. Nuseibeh, “Weaving Together Requirements and Architectures”, IEEE Computer 34(3), March, 2001: pp. 115-117.
[51] Q. He, A. I. AntÂ´on, “A Framework for Modeling Privacy Requirements in Role Engineering”, In Proceedings of the 9th Int. Workshop on Requirements Eng. : Found. for Software Quality, pages 137–146, 2003.
[52] T. Lodderstedt, D. Basin, J. Doser, “SecureUML: A UML-Based Modeling Language for Model-Driven Security”, In Proceedings of UML’02, LNCS 2460, pages 426–441, Springer-Verlag, 2002.
[53] T. Doan, S. Demurjian, T. C. Ting, A. Ketterl, “MAC and UML for secure software design”, In Proceedings of FMSE’04, pages 75–85, ACM Press, 2004.
[54] J. Jurjens, “Towards Secure Systems Development with UMLsec”, Fundamental Approaches to Software Engineering (FASE/ETAPS) 2001, International Conference, Genoa 4-6 April 2001.
[55] F. Swiderski, W. Snyder, “Threat Modeling. Microsoft Press”, 2004.