International Journal of Computer Networks and Communications Security

VOL. 4, NO. 10, OCTOBER 2016, 294–303
Available online at: www.ijcncs.org
E-ISSN 2308-9830 (Online) / ISSN 2410-0595 (Print)

A Comprehensive Approach to Security Requirements Engineering

Ilham Maskani[1], Jaouad Boutahar[2] and Souhail Elghazi[3]

[1] LISER Laboratory, ENSEM, Hassan II University, Casablanca, Morocco

[2, 3] Systems, architectures and networks Team, EHTP, Casablanca, Morocco

[1]maskani.ilham@gmail.com, [2]jaouad.boutahar@gmail.com, [3]elghazis@gmail.com

ABSTRACT

Software's security depends greatly on how a system was designed, so it's very important to capture security requirements at the requirements engineering phase. Previous research proposes different approaches, but each is looking at the same problem from a different perspective such as the user, the threat, or the goal perspective. This creates huge gaps between them in terms of the used terminology and the steps followed to obtain security requirements. This research aims to define an approach as comprehensive as possible, incorporating the strengths and best practices found in existing approaches, and filling the gaps between them. To achieve that, relevant literature reviews were studied and primary approaches were compared to find their common and divergent traits. To guarantee comprehensiveness, a documented comparison process was followed. The outline of our approach was derived from this comparison. As a result, our approach reconciles different perspectives to security requirements engineering by including: the identification of stakeholders, asset and goals, and tracing them later to the elicited requirements, performing risk assessment in conformity with standards and performing requirements validation. It also includes the use of modeling artifacts to describe threats, risks or requirements, and defines a common terminology.

Keywords: *Security Requirements, Requirements Engineering, Security Standards, Comparison, Risk Analysis.*

1 INTRODUCTION

Security needs have evolved with the evolution of information systems (IS). IS are more and more open and interconnected, which makes securing these IS more necessary and more challenging. But, in the Software Development Life Cycle (SLDC), security issues are often addressed at the design phase at best, or at maintenance phase at worst by fixing detected vulnerabilities. As reported in this paper [1], finding and fixing a software problem after delivery is often 100 times more expensive than finding and fixing it during the requirements and design phase. A model developed by MIT, whose objective is to prove the return of investment on secure software development, showed that the earliest the security is addressed, the highest the benefit (21%) [2]. Thus, it is critical to address security issues at the earliest phase. This is the reason why OWASP recommends focusing a big part of security flaws detecting efforts on the requirements engineering phase and the design phase[3].

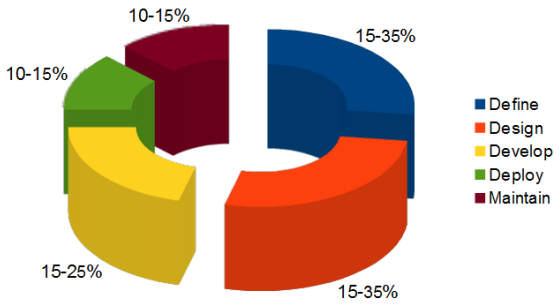

Fig. 1. Proportion of Test Effort in SDLC

Requirements engineering is the very first step to make any software. It is usually applied to functional requirements, and can be extended to quality and security requirements, traditionally considered non-functional. By integrating security requirements into requirements engineering, a big improvement can be made in term of security vulnerabilities, software maintenance efforts and development costs. Many initiatives propose different approaches to security requirements engineering (SRE), along with

literature reviews of these approaches. In the first section, we start by presenting these works. We will use the term "approach" to refer to any method, framework ... which sets out clear steps to obtain security requirements. In the second section, we present the selection and comparison process for approaches featured in our research. We then compare them according to the predefined criteria. In the final section, we will define a common terminology for the concepts used by the approaches and present the outline and the desired qualities for our comprehensive approach to engineering security requirements (COMPASRE).

.

2 RELATED WORK

To achieve our aim, relevant literature reviews were studied and primary approaches were compared to find their common and divergent traits. This section presents the reviews and approaches featured in our research. These approaches were selected by applying the selection & comparison process detailed in the next section.

2.1 Reviews

2.1.1 Survey and analysis on security requirements engineering

It is the most recent detailed analysis on the subject[4]. They discuss various types of security requirements with given examples, stretching the importance of considering security requirements as functional requirements. They compare approach activities to identify the weaknesses of each. The choice of an approach over another depends on covered activities and existing SW development methods in an organization.

2.1.2 A comparison of SRE methods

Proposed a conceptual framework against which approaches can be evaluated[5]. They made a commendable effort to categorize existing approaches: Multi-view approaches, Goal-based approaches …

2.1.3 A systematic review of security requirements engineering

[6]A systematic, thorough review which aims to supply researchers with a summary of all the existing information about security requirements in a thorough and unbiased manner, providing a background in which to appropriately position new research.

2.1.4 Security requirements for the Rest of us – a survey

[7]Highlights mainstream approaches, focuses on the importance of simplifying SRE methods, as a lightweight method is more likely to be adopted than a complex one. It also stretches the importance of scholar education of developers and SW engineers on SRE.

2.2 Overview of Approaches

2.2.1 SREF

Security Requirements Framework by Haley et al. [8] is a mix between engineering requirements and security requirements. It's iterative as it goes back and forth between modeling and requirements engineering. SREF follows 4 steps:

- Identify functional requirements
- Identify security goals
    - Identify assets
    - Generate threat description
    - Apply management principles (separation of duties, functions, ..)
- Identify security requirements: constraints on one or more security goal. The security requirements are denoted textually.
- Construct satisfaction arguments: show that the system can satisfy the security requirements.

2.2.2 KAOS anti-models

To elaborate security requirements, Van Lamsweerde[9] suggests using KAOS by constructing intentional anti-models. KAOS is a Goal Oriented Method for requirements engineering. A goal is a desired property of the IS to be, that has been expressed by a stakeholder. The satisfaction of this goal will depend on successful cooperation between all agents of the IS. KAOS documents requirements using a goal tree, with strategic goals as the root and IS requirements as leafs. Security requirements using anti-models are elaborated in 3 steps. First, model the security goals. Then, derive from the former model an anti-model based on threats. Finally, derive from both former models countermeasures and define the security requirements. A requirement is defined as a terminal goal under the responsibility of an agent in the software.

2.2.3 MOSRE

The aim of the Model Oriented Security Requirements Engineering approach[10][réf] is the use of models (App's use cases, misuse cases, …) to make the traceability and analysis of

requirements easier. It's tailored for web applications. There are two particularities to MOSRE. First, it encompasses identification of the whole IS goals and objectives, and then the elicitation and modeling of non-security requirements (functional or non-functional) before dealing with the security requirements. It is thus a method that could be applied to the whole requirements engineering phase, but has a special focus on security. MOSRE Steps are:

- Inception: Identify web app objectives, stakeholders and assets
- Elicitation
  - Elicit security and non-security goals and requirements
  - Identify threats and vulnerabilities
  - Risk assessment
  - Identify Security requirements
  - Generate Use case diagrams considering security requirements
- Elaboration : Generate structural analysis models (ex : data model, flow models) and develop UML diagrams to give a of the secure web app in general (ex: high level class diagram, sequence diagram)
- Negotiation and validation of requirements

2.2.4 MSRA

The focus of the MSRA (Multilateral security requirements analysis) approach is to identify and analyze security requirements from the multiple views of stakeholders[11]. Security requirements result from the reconciliation of multilateral security goals, which are selected from a rich taxonomy. Security goals, and later requirements, contain the attributes "stakeholders" who have an interest in the requirement, "counter-stakeholders" towards whom a requirement is stated, and other attributes such as "owner", "degree of agreement" between stakeholders, the "information" to be protected by the requirement, the security "goal" that the requirement achieves… A singularity of MSRA is that, when resolving conflicts between requirements, it takes into account both functional (assumed to be extracted prior to applying MSRA) and security goals. There is a variant of MSRA, the Confidentiality Requirements Elicitation and Engineering (CREE) approach, which focuses only on confidentiality requirements and how they can be formalized. The steps followed by the MSRA are:

1. Identify stakeholders

2. Identify episodes: Episodes are similar to scenarios, but are of a lower granularity, identifying sets of functionalities as would be meaningful to users. Episodes are used to partition the security goals and are later useful in identifying conflicts between multiple security goals.

3. Elaborate security goals: Identify and describe the security goals of the different security stakeholders for each of the episodes.

4. Identify facts and assumptions: These are the properties of the environment that are relevant for stating security goals.

5. Refine stakeholder views on episodes: Elaborate the stakeholder views taking facts, assumptions, and the relationships between episodes into account.

6. Reconcile security goals: Identify conflicts between security goals, find compromises between conflicting goals, and establish a consistent set of security system requirements.

7. Reconcile security and functional requirements: Trade functionality for security and vice versa in case of conflicting functional and security requirements.

2.2.5 Secure tropos

Tropos is a requirements-driven software development methodology. It's based on the i* framework, an agent-oriented modeling framework. While Tropos guides the development of agent-based systems through all phases of the SDLC, it is very focused on the requirements engineering phase. Secure Tropos [12] is based on the concepts of:

- actor, : have strategic goals and intentions within the system or the organization
- goal, soft goal : the strategic interests of an actor
- task: a particular course of action that produces a desired effect, and can be executed in order to satisfy a goal.
- resource : a physical or an informational entity
- social relationships for defining the obligations of actors to other actors : functional dependency, ownership, provisioning, trust, and delegation of permission.

Various activities contribute to the acquisition of a first requirement model, to its refinement into subsequent models: Actor modeling, Dependency modeling, Trust modeling, which consists of identifying actors which trust other actors for goal, plans, Delegation modeling and Goal refinement. Secure Tropos has been applied to the Italian data protection legislation compliance[13].

2.2.6 Holistic SRE

Holistic security requirements engineering [14] was conceived to overcome the shortcomings of other approaches to SRE that were, at that time, mostly based of risk analysis. This approach, aimed at electronic commerce systems, defines risks, business processes and stakeholder & environmental demands as sources of security requirements. This leads to holistic security requirements, defined as "A need or restriction from a user, a stakeholder or the environment related to the goal to improve the system security".
The approach is described by this biphasic process with the following activities:

- Phase I: Preparation, aims to gather requirements from each of the sources.
- Phase II: Compilation, aims to compile the different requirements and resolve conflicts between them.

An evolution of this approach, named SKYDD, was developed to better suit the needs of telecom providers.

2.2.7 SQUARE

Developed by Carnegie Mellon University, SQUARE (Security Quality Requirements Engineering)[15] is a 9-steps process whose goal is to get categorized and prioritized security requirements. Each step is described with inputs, outputs, participants and techniques:

- Agree on definitions
- Identify security goals
- Develop Artifacts to support security requirements definition
- Perform risk assessment
- Select elicitation techniques
- Elicit security requirements
- Categorize requirements
- Prioritize requirements
- Requirements inspection

This method had been extended to specifically treat privacy (P-SQUARE) and acquisition (A-SQUARE).

2.2.8 SREP

Security Requirements Engineering Process[16] is a Common Criteria centered and reuse-based process that deals with security requirements at the early stages of software development in a systematic and intuitive way, by providing a security resources repository as well as integrating the Common Criteria into the software lifecycle, so that it unifies the concepts of requirements engineering and security engineering. In order to support this method, many concepts and techniques are used: a security resources repository (with assets, threats, requirements, etc), misuse cases, threat/attack trees, and security uses cases. SREP has been developed by taking into account the standard ISO/IEC 27002[17]. SREP activities are:

- Agree on Definitions
- Identify Vulnerable &/or Critical Assets
- Identify Security Objectives & Dependencies
- Identify Threats & Develop Artifacts
- Risk Assessment
- Elicit Security Requirements
- Categorize & Prioritize Requirements
- Requirement Inspection
- Repository Improvement

SREPPLINE is a declination of SREP specific to Software Product Lines.

2.2.9 STS

Going from the statement that software operates within the context of larger socio-technical systems, STS is an approach for modeling and reasoning about security requirements [18]. Security requirements are specified, via the STS-ml requirements modeling language, as contracts that constrain the interactions among the actors. The requirements models of STS-ml have a formal semantics which enables automated reasoning for detecting possible conflicts among security requirements at design time. STS was applied to an e- Government system for tax collection. STS steps are:

- Modeling system components and interaction with STS-ml language
  - Social view for stakeholders
  - Information view
  - Authorizations view
- Use the models to specify security requirements as constraints on the interactions. Security requirements are specified in the STS-ml langage.
- Use the automated reasoning to detect conflicts.

3 COMPARISON OF SRE APPROACHES

This section presents the process followed to select and compare the approaches featured in our research and shows the results of the comparison.

3.1 Comparison Process

To guarantee the comprehensiveness of our approach, a documented selection and comparison process was followed (see figure1). This process is inspired by an evaluation method for engineering approaches in the secure SDLC named SecEval [19]. This distinguishes our work from the previous reviews as they compare only a certain set of approaches, without explaining the inclusion or exclusion criteria. Documenting our process makes this comparison reproducible for future research.

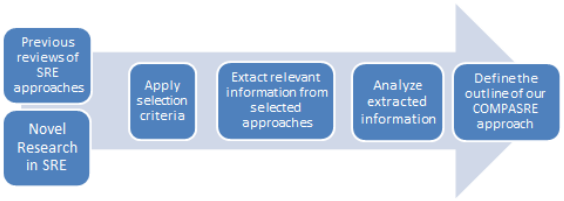

Fig. 2. Selection & Comparison process

3.1.1 Sources

The aforementioned reviews were a very rich source. To complete the information gathered, we queried different scientific databases to find novel research in the area. This way, we obtained other approaches that have not yet been featured in any of the previous reviews, such as MOSRE and STS. Other sources were: Sciencedirect, ResearchGate and GoogleScholar.

3.1.2 Selection criteria

Selection criteria were applied on the gathered research. The first criterion is if the proposed approach is focused on the early phase of the development lifecycle. Indeed, many approaches go straight to the design phase by proposing modeling approaches, without specifying how to extract those requirements in the first place. Others propose activities to enhance security through the whole Software Development Life Cycle such as CLASP[20] and Microsoft SDL [21]. To have a precise scope, we only kept the methods that focus on the requirements engineering phase. The second criterion is the novelty. Chosen approaches have been referenced in the years 2008 and up. The third criterion is that chosen approaches offer a clear process or clear steps about how to extract the security requirements, and not just general guidelines about security requirements, or their management.

3.1.3 Information extraction

Once the final approaches were selected, the following information was extracted to be used as comparison criteria.

- Steps: What are the clear steps followed to obtain security requirements
- Security Objectives: Whether the approach addresses all security objectives (Confidentiality, Integrity, Availability...) or focuses on a single one
- Tool / Notation support: Whether there is a tool or a notation developed to support the use of the approach
- Use / Application: Whether the approach had been applied to a case study or a real IS
- Includes modeling activities (design): Whether the approach includes high modeling activities, to support the obtained requirements
- Compliance with security standards: Whether the approach is compliant or inspired by any security standard
- Reusability of requirements: Whether the approach promotes the reuse of obtained requirements
- Use of ontology/taxonomy: Whether the approach uses an existing ontology ot taxonomy to define the approach steps and to define the security requirements
- Domain specific: Whether the approach is dedicated to a certain type of software (Web applications, Mobile, E-Gov,…)

3.2 Comparison Results

Table 1 summarizes the steps found in each approach, and gives a synthetic view about the most and the least common steps through all approaches. No single approach includes these steps all at once. First, we can see that "Identifying vulnerabilities/threats" and "Identifying security goals" are the most common steps since we can't derive requirements without establishing goals, and it's important to know a system's vulnerabilities and threats to be able to secure it. Then, other steps are also quite persistent such as "Identifying stakeholders", "Creating security artifacts" and "Validation of requirements". Identifying stakeholders is a way to make sure that all the systems goals have been mapped, since different stakeholders will have different views of the systems, and thus different goals. Creating security artifacts is important as it helps clarify the requirements by incorporating artifacts such as

attack trees, misuse cases. It will also help during later phases of the project for designers and implementers. As for Requirements validation, the goal of it is to make sure that all goals have been covered by the elicited requirements, with no conflicts between them. Finally, some steps are less common, while still being very important, such as "Risk assessment" and "Repository enhancement". Risk assessment builds on the identified threats and vulnerabilities to identify, analyze and evaluate them by choosing for example the risks to accept and those to mitigate. Assessing risks leads to thinking about security controls, which could lead to new requirements. Keeping and enhancing a repository is a way to promote de reuse of requirements, as such a repository can be used to identify other risks, and to validate the requirements.

As for the characteristics, we present in table 2 how each approach does regarding our comparison criteria. First thing we deduce is that there is no approach that fulfills all criteria. Apart from Secure Tropos, all approaches try to cover most security objectives, especially the CIA triad (Confidentiality, Integrity, and Availability). Some approaches are defined from the beginning to better suit certain systems such as Web Applications that are more and more used to replace custom applications. When applied, they are aimed at highly data sensitive systems such as e-gov, e-commerce and e-health. As for artifacts and notation, the most used are UML based (misuse cases, UMLSec [22]) and attack trees. Some approaches have developed their own notation system, or even a tool to create their artifacts and support their approach. The conformity to security

Table 1: Occurrences of steps per approach

STEPS	APPROCHES									
	SREF	KAOS anti-models	MOSRE WebApp	MSRA	Secure Tropos	Holistic SRE	SQUARE	SREP	STS	Number of occurrences
Agree on definitions							X			1/9
Identify assets	X	X	X					X		4/9
identify stakeholders		X	X	X	X	X			X	6/9
Identify security goals/objectives	X	X	X	X	X	X	X	X	X	9/9
identify business/ IS objectives	X		X			X				3/9
Identify threats	X	X	X		X		X	X	X	7/9
Develop Artifacts		X	X		X		X	X	X	6/9
Perform risk assessment			X			X	X	X		4/9
Select elicitation techniques			X				X			2/9
Elicit -non security requirements	X		X							2/9
Elicit security requirements	X	X	X	X	X	X	X	X	X	9/9
Categorize / Prioritize requirements			X				X	X	X	4/9
Requirements inspection/validation/Conflict resolution	X		X	X		X	X	X	X	7/9
Repository Improvement								X		1/9

standards is quite present, especially for the approaches that include risk assessment. Common security standards used are the ISO 27000 family of standards[23] and the SSE-CMM (Systems Security Engineering- Capability Maturity Model)[24]. For the purpose of better understanding of requirements, some approaches propose their own format in which requirements are documented. The rarest characteristics were the use of a taxonomy or ontology to build the approach, and the use of a repository of requirements for reuse.

4 OUTLINE OF OUR COMPASRE APPROACH

4.1 Common Terminology

From studying each approach, we can identify a set of concepts that are consistent through most approaches: Stakeholder, Asset, Risk, etc… These concepts are drawn from both the fields of security and requirements engineering. Table 3 below offers a definition of these concepts to establish a common terminology based on the ISO/IEC 27000:2016 vocabulary[25]. Some existing papers offer detailed taxonomies[26] and facilitate applying SRE approaches. This is the terminology that we will base our COMPASRE approach on.

Table 2: Characteristics (Comparison criteria)

COMPARISON CRITERIA	APPROACHES								
	Holistic SRE	KAOS anti-models	MOSRE WebApp	MSRA	Secure TROPOS	SREF	SREP	SQUARE	STS
Security Objectives Specific	confidentiality, integrity, non-repudiation	CIA + privacy, authentication, non-repudiation		CIA + accoutability, pseudonimity	Privacy, Trust				CIA + accountability, reliability, authenticity
Tool / Notation support	No	Temporal logic notations	No		Si*	No		P-square	STS-ml, STS Tool
Use / Application	e-Commerce, Telecom	e-Banking	e-Voting, e-Healh system	e-Health	Italian Legislation compliance				e-Government
Includes modeling activities of requirements	Yes	Yes	Security use cases, misuse cases, attack trees	UML	YES	No	Security use cases, misuse cases, attack trees	misuse cases, attack trees	Yes
Compliance with security standards	ISOO 27000, SSE-CMM	No		No	ISO/IEC 27002	No	Common Criteria, SSE-CMM, ISO/IEC 27002	ISO 27005	
Format / Reusability of requirements	Yes	No	Yes	Yes	No	No	Yes	No	Yes
Based on ontology or taxonomy	No	No	No	Yes	No	No		No	
Domain specific	e-Commerce, Telco	No	Web Apps	No	Agent based systems	No		No	Large socio-technical systems

Table 3: Common terminology

Concept	Definition	Alternate labels
Stakeholder	Person or organization that can affect, be affected by, or perceive themselves to be affected by a decision or activity. Some approaches include other systems that have an interest in the IS.	Actor, client, agent
Asset	Anything that has value to the organization, its business operations and their continuity, including Information resources that support the organization's mission (Data).	Information , Resource, Object
Goal	A Security objective that must be achieved by the system to be	Objective
Vulnerability	weakness of an asset or control that can be exploited by one or more threats	
Threat	potential cause of an unwanted incident, which may result in harm to a system or organization	
Risk	Potential that threats will exploit vulnerabilities of an information asset or group of information assets and thereby cause harm to an organization	
Risk Assessment	Overall process of risk identification, risk analysis and risk evaluation	Risk identification, risk analysis, risk evaluation
Requirement	Need or expectation that is stated, generally implied or obligatory. Requirements are low level details of goals.	Goal
Control	Measure that is modifying risk	Countermeasure
Attack	Attempt against the security of an asset	

4.2. Proposed activities

Based on the previous section, we can give guidelines about a new comprehensive approach that takes into account the strengths and weaknesses of studied approaches. We will try to avoid being too specific about a domain or any other specificity that might limit the use of our approach. Still, the new approach has to include important concepts and techniques such as: identification of stakeholders, identification of assets and threats, risk assessment and reuse of requirements. It also has to follow general guidelines of requirements engineering by documenting, tracing and validating requirements. These are the activities that we propose for our COMPASRE approach:

1. Identify stakeholders
2. Identify assets
3. Identify Security goals
4. Identify Threats/vulnerabilities
5. Create artifacts(Misuse cases, attack trees …)
6. Risk assessment (in conformity to 27005)
7. Elicit security requirements (use a specific elicitation technique)
8. Fomat security requirements
9. Categorize and Prioritize
10. Inspection/validation
11. IS Use case including security (ex : UML sec)
12. Repository Enhancement

If those activities are followed correctly, our approach would have the following qualities:

- Environment reconnaissance: The more complex the IS, the more important it is to identify the stakeholders and the assets. Elicited security requirements will have to be traced all the way back to the related assets and related stakeholders.
- Risk assessment: The finality of securing a system is to be prepared against all risks. Thus, it is important for our approach to identify all vulnerabilities and threats, to enable a thorough risk assessment.
- Favor re-usable requirements:
  - Propose a standard format to represent security requirements.
  - Keep a repository of sample and categorized requirements
- Follow the fundamentals of requirements engineering. Some of those fundamentals tend to be overlooked:
  - Traceability: It is important to be able to match each obtained requirement with the associated risk, the asset, the security goal it covers and the stakeholder who expressed it. This will help at the requirements inspection phase, and at later phases of the SDLC when managing requirements.
  - Inspection and validation: Obtained requirements should be inspected to resolve any conflicts, and to ensure complete coverage of all the initially stated security goals.

• Easy and faithful transition from requirements engineering phase to design phase -> use of modeling artifacts to describe threats, risks and requirements.

• Use of existing risk management standard and Bodies Of knowledge (ISO 27002, ISO 27005, EBIOS, BSI…) for threats, risk assessment and security goals.

• Ease of use: It should be detailed and documented enough to be applied easily. Complicated and time consuming steps (ex: modeling artifacts) should be simplified and kept to a minimum.

• Favor an iterative approach (Elicited requirement could lead to new assets, resulting in new security requirements).

5 CONCLUSION & PERSPECTIVES

Our aim was to define a comprehensive approach to security requirements engineering. The first contribution of our research is that it can be used by fellow researchers or practitioners to position themselves between heterogeneous approaches. Our comparison criteria and common terminology allows a better understanding of each approach, and can help choose the most appropriate approach for a certain need. The second contribution is our COMPASRE approach that conciliates between the different trends to security requirements engineering: goal oriented, risk analysis oriented and multilateral. As such, it distinguishes itself by being faithful to the fundamentals of requirements engineering, to security standards and by facilitating the use of security requirements in later phases of the SDLC through requirements formatting and security enhanced system artifacts. When eliciting requirements, regardless of the approach used, security requirements shouldn't be an afterthought, but an indivisible part of requirements engineering for the system as a whole. Security requirements should be confronted with other functional, quality or performance requirements for further validation and conflict resolution so they would be incorporated in the system's design.

Our plans for future work are to fully develop our COMPASRE approach following the described outline. We would document the inputs, activities and outputs of each step, describe the artifacts to be created, and develop a format for security requirements. We would also explain how our approach integrates with security in later phases of the SDLC. We plan to validate our approach by applying it to a concrete security sensitive system, and apply security metrics to improve its efficiency.

6 REFERENCES

[1] B. Boehm and V. R. Basili, "Top 10 list [software development]," Computer, vol. 34, no. 1, 2001, pp. 135–137.

[2] H. S. Venter and Information Security South Africa, Eds., Peer-reviewed proceedings of the ISSA 2004 enabling tomorrow conference. ISSA, 2004.

[3] "Testing Guide Introduction - OWASP." [Online]. Available: https://www.owasp.org/index.php/Testing_Guide_Introduction. [Accessed: 13-Oct-2016].

[4] P. Salini and S. Kanmani, "Survey and analysis on Security Requirements Engineering," Comput. Electr. Eng., vol. 38, no. 6, 2012, pp. 1785–1797.

[5] B. Fabian, S. Gürses, M. Heisel, T. Santen, and H. Schmidt, "A comparison of security requirements engineering methods," Requir. Eng., vol. 15, no. 1, 2010, pp. 7–40.

[6] D. Mellado, C. Blanco, L. E. Sánchez, and E. Fernández-Medina, "A systematic review of security requirements engineering," Comput. Stand. Interfaces, vol. 32, no. 4, 2010, pp. 153–165.

[7] I. A. Tondel, M. G. Jaatun, and P. H. Meland, "Security Requirements for the Rest of Us: A Survey," IEEE Softw., vol. 25, no. 1, 2008, pp. 20–27.

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

[9] A. van Lamsweerde, "Elaborating Security Requirements by Construction of Intentional Anti-Models," in Proceedings of the 26th International Conference on Software Engineering, Washington, DC, USA, 2004, pp. 148–157.

[10] P.Salini and S. Kanmani, "Security Requirements Engineering Process for Web Applications," Procedia Eng., vol. 38, 2012, pp. 2799–2807.

[11] S. F. Gürses and T. Santen, "Contextualizing Security Goals: A Method for Multilateral Security Requirements Elicitation.," in ResearchGate, 2006, pp. 42–53.

[12] P. Giorgini, F. Massacci, J. Mylopoulos, and N. Zannone, "Requirements engineering for trust management: model, methodology, and reasoning," Int. J. Inf. Secur., vol. 5, no. 4, 2006, pp. 257–274.

[13] F. Massacci, M. Prest, and N. Zannone, "Using a security requirements engineering

methodology in practice: The compliance with the Italian data protection legislation," Comput. Stand. Interfaces, vol. 27, no. 5, 2005, pp. 445–455.

[14] A. Zuccato, "Holistic security requirement engineering for electronic commerce," Comput. Secur., vol. 23, no. 1, 2004, pp. 63–76.

[15] Mead N, Hough E, Stehney T (2005) Security quality requirements engineering (SQUARE) methodology. Carnegie Mellon Software Engineering Institute, Technical report CMU/SEI-2005-TR-009.

[16] D. Mellado, E. Fernández-Medina, and M. Piattini, "A common criteria based security requirements engineering process for the development of secure information systems," Comput. Stand. Interfaces, vol. 29, no. 2, 2007, pp. 244–253.

[17] "ISO/IEC 27002:2013 - Information technology -- Security techniques -- Code of practice for information security controls," ISO. [Online]. Available: http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=54533. [Accessed: 20-Oct-2016].

[18] E. Paja, F. Dalpiaz, and P. Giorgini, "Modelling and reasoning about security requirements in socio-technical systems," Data Knowl. Eng., vol. 98, 2015, pp. 123–143.

[19] M. Heisel, W. Joosen, J. Lopez, and F. Martinelli, Eds., Engineering Secure Future Internet Services and Systems, vol. 8431. Cham: Springer International Publishing, 2014.

[20] "CLASP Concepts - OWASP." [Online]. Available: https://www.owasp.org/index.php/CLASP_Concepts. [Accessed: 20-Oct-2016].

[21] "Microsoft Security Development Lifecycle." [Online]. Available: https://www.microsoft.com/en-us/sdl/. [Accessed: 20-Oct-2016].

[22] J. Jrjens, Secure Systems Development with UML. Berlin, Heidelberg: Springer-Verlag, 2010.

[23] "ISO/IEC 27001 - Information security management," ISO. [Online]. Available: http://www.iso.org/iso/home/standards/management-standards/iso27001.htm. [Accessed: 20-Oct-2016].

[24] "ISO/IEC 21827:2008 - Information technology -- Security techniques -- Systems Security Engineering -- Capability Maturity Model® (SSE-CMM®)," ISO. [Online]. Available: http://www.iso.org/iso/catalogue_detail.htm?csnumber=44716. [Accessed: 20-Oct-2016].

[25] "ISO/IEC 27000:2016 - Information technology -- Security techniques -- Information security management systems -- Overview and vocabulary," ISO. [Online]. Available: http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=66435. [Accessed: 20-Oct-2016].

[26] N. Rjaibi and L. B. A. Rabai, "Developing a Novel Holistic Taxonomy of Security Requirements," Procedia Comput. Sci., vol. 62, 2015, pp. 213–220.