Analysis of Security Requirements Engineering: Towards a Comprehensive Approach

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

Jaouad Boutahar[2], Souhaïl El Ghazi El Houssaïni[3]
Systems, architectures and networks Team
EHTP
Casablanca, Morocco

***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, it reconciles different perspectives to security requirements engineering by including: the identification of stakeholders, assets 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 assessment

I. 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 (SDLC), 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 as shown in fig. 1[3]. 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, these works will be presented. The term "approach" will be used to refer to any method, framework, etc. which sets out clear steps to obtain security requirements. In the second section, the selection and comparison process followed for featured SRE approached will be explained. Then, approaches will be compared according to the predefined criteria. In the final section, a common terminology will be defined for the concepts used by the approaches. Then the outline of our comprehensive approach to SRE will be presented, along with its desired qualities.

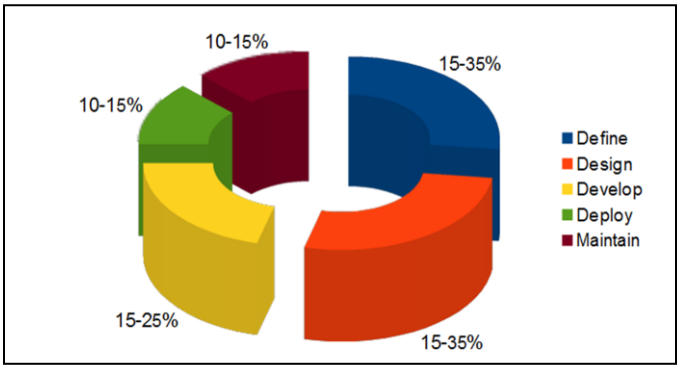

Fig. 1. Recommended proportions of Test Effort in SDLC

II. 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.

A. Reviews

1) Survey and analysis on Security Requirements Engineering: [4] It is the most recent detailed analysis on the subject. 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) A Comparison of SRE methods: [5] proposed a conceptual framework against which approaches can be evaluated. They made a commendable effort to categorize existing approaches: Multi-view approaches, Goal-based approaches …

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.

4) Security Requirements for the Rest of Us - A Survey : [7]This survey highlights mainstream approaches. It focuses on the importance of simplifying SRE methods since a lightweight method is more likely to be adopted than a complex one. It also stretches the importance of scholar education of developers and software engineers on the SRE discipline.

B. Overview of Approaches

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) KAOS anti-models: To elaborate security requirements, Van Lamsweerde 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.

3) MOSRE: The aim of the Model Oriented Security Requirements Engineering approach [9] 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. The particularity to MOSRE is that it encompasses identification of goals for the whole IS, the elicitation and the modeling of non-security requirements (functional or non-functional) before dealing with the security requirements. It is thus a method that can be applied to the whole requirements engineering phase, with 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 view of the secure web application in general (ex: high level class diagram, sequence diagram)
- Negotiation and validation of requirements

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[10]. 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:

- Identify stakeholders
- Identify episodes: Episodes are similar to scenarios, but are of a lower granularity. They are used to partition the security goals and are later useful in identifying conflicts between multiple security goals.

- Elaborate security goals: Identify and describe the security goals of the different stakeholders for each of the episodes.
- Identify facts and assumptions: These are the properties of the environment that are relevant for stating security goals.
- Refine stakeholder views on episodes: Elaborate the stakeholder views taking facts, assumptions, and the relationships between episodes into account.
- Reconcile security goals: Identify conflicts between security goals, find compromises between conflicting goals, and establish a consistent set of security system requirements.
- Reconcile security and functional requirements: Trade functionality for security and vice versa in case of conflicting functional and security requirements.

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[11] is based on the concepts of social relationships for defining the obligations of actors to other actors : functional dependency, ownership, provisioning, trust, and delegation of permission.
Secure Tropos steps are:

- Early requirements phase: studies the organizational setting of the future system
    - Actor diagram : identifying stakeholders and trust relationships between them (Trust modeling, Functional Modeling and Trust Management implementation)
    - Goal diagrams for each actor
- Late requirements phase: describes the future system within its operational environment, along with relevant functions and qualities, using further actor and goal diagrams.
- Requirements analysis :
    - Expressing system requirements in form of actors' properties and relations
    - validation of both functional and security requirements

Secure Tropos has been applied to the Italian data protection legislation compliance[12].

6) Holistic security requirements engineering: Holistic security requirements engineering [13] was conceived to overcome the shortcomings of other approaches to SRE. 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.
    - Definition of goals
    - Security enhanced business modeling: Modeling business information exchange, considering security as business functionality.
    - Requirement transformation: transforming security considerations from the business model into security requirements
    - Internal requirement elicitation: Detailing the transformed requirements
    - Stakeholder definition
    - Requirement elicitation: From stakeholders' points of view
    - Risk assessment : through a baseline investigation of risks using checklists
- Phase II: Compilation, aims to compile the different requirements and resolve conflicts between them.
    - Compilation
    - Formal security requirements specification
    - Prototyping
    - Validation

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

7) SQUARE: Developed by Carnegie Mellon University, SQUARE (Security Quality Requirements Engineering)[14] 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 approach had been extended to specifically treat privacy (P-SQUARE) and acquisition (A-SQUARE).

8) SREP: Security Requirements Engineering Process[15] is a process centered on the security evaluation standard Common Criteria[16] and based on the notion of reuse. It deals with security requirements in a systematic and intuitive way. It provides a security resources repository and integrates the Common Criteria into the software lifecycle, so that it unifies the concepts of requirements engineering and security engineering. In order to support this approach, 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.

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 in such systems [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. STS was applied to an e- Government system for tax collection.

STS steps are:

- Model 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 language.
- Use the automated reasoning to detect conflicts

III. 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.

A. Comparison Process

To guarantee the comprehensiveness of our approach, a documented selection and comparison process was followed (see fig. 2). 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.

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.

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, only the methods that focus on the requirements engineering phase were kept. 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) 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.

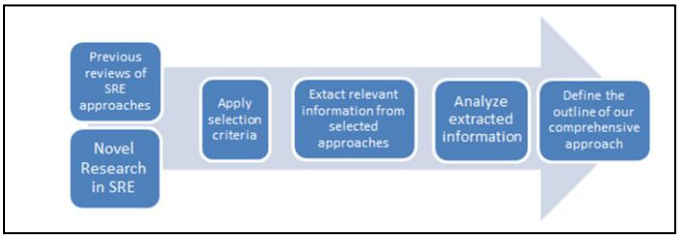

Fig. 2. Selection & Comparison process

- Includes modeling activities (design): Whether the approach includes high level design activities, taking into account the obtained security 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, etc.)

B. Comparison Results:

Tab. 1 summarizes the steps found in each approach, and gives a synthetic view about the most and the least common steps. 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 and misuse cases. It will also help designers and developers during later phases of the project. 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 often neglected even if they're very important, such as "Risk assessment" and "Repository enhancement". Risk assessment builds on the identified threats and vulnerabilities to identify analyze and evaluate risks 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. Such a repository can be used to validate the obtained requirements and identify new ones.

As for the characteristics comparison, we present in tab. 2 the results for each approach regarding to the aforementioned 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 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 existence of a tool supporting the approach.

IV. Outline of Our Comprehensive Approach

A. 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. Tab. 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 approach on.

B. 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 approach:

1) Identify stakeholders
2) Identify assets
3) Identify Security goals
4) Identify Threats/vulnerabilities
5) Create artifacts: Misuse cases, attack trees, etc.
6) Risk assessment (in conformity to ISO/IEC 27005)
7) Elicit security requirements Format security requirements
8) Categorize and Prioritize
9) Inspection/validation
*10)*Enhance IS Use case by including security (ex : UML sec)
*11)*Repository Enhancement

TABLE I. OCCURENCES OF STEPS

Steps	Approaches									
	SREF	**KAOS anti-models**	**MOSRE WebApp**	**MSRA**	**Secure Tropos**	**Holistic SRE**	**SQUARE**	**SREP**	**STS**	**Number of occurrences**
Agree on definitions							X	X		2/9
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

TABLE II. CHARACTERISTICS OF APPROACHES (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	No	Si*, ST-tool	No		P-square	STS-ml, STS Tool
Use / Application	e-Commerce, Telecom	e-Banking	e-Voting, e-Healh system	e-Health	Italian Legislation compliance	No	Software Product Lines	Asset Management System	e-Government
Includes modeling activities of requirements	Yes	Yes	Security use cases, misuse cases, attack trees		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	NIST SP 800-30	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, Telecom	No	Web Apps	No	Agent based systems	No		No	Large socio-technical systems

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 resolveany conflicts, and to ensure complete coverage of all the initially stated security goals.

TABLE III. 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, objective
Control	Measure that is modifying risk	Countermeasure
Attack	Attempt against the security of an asset	

- 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, etc.) 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.

V. CONCLUSION & PERSPECTIVES

Our aim was to define the outline of a comprehensive approach to security requirements engineering. To achieve that, a thorough analysis of existing SRE approaches was conducted. The outline, along with a common terminology, was drawn from this analysis. 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 comprehensive 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 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 measure security metrics to improve its efficiency.

REFERENCES

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

[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, pp. 1785–1797, Nov. 2012.

[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, pp. 7–40, Mar. 2010.

[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, pp. 153–165, Jun. 2010.

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

[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, pp. 133–153, Jan. 2008.

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

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

[11] 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, pp. 257–274, 2006.

[12] 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, pp. 445–455, Jun. 2005.

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

[14] 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.

[15] 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, pp. 244–253, Feb. 2007.

[16] "ISO/IEC 15408-1:2009 - Information technology -- Security techniques -- Evaluation criteria for IT security -- Part 1: Introduction and general model," ISO. [Online]. Available: http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=50341. [Accessed: 25-Oct-2016].

[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, pp. 123–143, Jul. 2015.

[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, pp. 213–220, 2015.