Complex Systems Informatics and Modeling Quarterly
CSIMQ, Issue 3, July, 2015, Pages 15-34
Published online by RTU Press, https://csimq-journals.rtu.lv
http://dx.doi.org/10.7250/csimq.2015-3.02
ISSN: 2255-9922 online

Getting Grip on Security Requirements Elicitation by Structuring and Reusing Security Requirements Sources

Christian Schmitt[1] and Peter Liggesmeyer[2]

[1] Siemens AG
[2] University of Kaiserslautern

ch.schmitt@siemens.com, liggesmeyer@cs.uni-kl.de

Abstract. This paper presents a model for structuring and reusing security requirements sources. The model serves as blueprint for the development of an organization-specific repository, which provides relevant security requirements sources, such as security information and knowledge sources and relevant compliance obligations, in a structured and reusable form. The resulting repository is intended to be used by development teams during the elicitation and analysis of security requirements with the goal to understand the security problem space, incorporate all relevant requirements sources, and to avoid unnecessary effort for identifying, understanding, and correlating applicable security requirements sources on a project-wise basis. We start with an overview and categorization of important security requirements sources, followed by the description of the generic model. To demonstrate the applicability and benefits of the model, the instantiation approach and details of the resulting repository of security requirements sources are presented.
Keywords: Security requirements engineering, security engineering, security requirements sources, compliance.

1 Introduction

Starting with the motivation for the topic, this section presents the contributions, related work and the structure of this paper.

1.1 Motivation

Security requirements engineering (SRE) is a challenging task, requiring profound security and SRE method knowledge. Although numerous publications state that SRE is important, only little concrete and specific advice is provided, which can immediately be used in projects [1]. In literature on SRE it is often stated that the application of SRE methods and techniques in projects and the maturity of security requirements specifications is mostly poor (e.g., [2], [3]). Moreover, from the various methods, which have been developed for SRE, just a small number has been used in practice so far [3]. Two aspects contribute to the sparse use of SRE methods and the little number and poor quality of security requirements (SR) specifications. The first aspect is the lack of knowledge and skills for security and SRE (e.g., [4]), which is considered as one of the main challenges in security requirements engineering. If one does not understand the mindset of an attacker, typical threats and weaknesses, as well as available exploits, the results from SRE methods particularly from analysis-oriented methods, such as threat modeling or attack tree

development, will likely not produce the same results and quality as if a security professional would have been involved. The second aspect relates to the various kinds of security information and knowledge sources, which potentially could be used by the SRE community as Security Requirements Sources (SRS[1]) to support the application of SRE methods and the elicitation of security requirements. Elahi et al. conclude in their survey on SRE [5] that security knowledge sources are seldom used in practice.

Besides these two aspects, we see the increasing number of internal and external compliance obligations as an important SRS to be considered as input into SRE processes and methods. In practice, compliance obligations are very important for organizations, since non-conformities can have a high negative business impact due to delays or even the refusal of the admission of a product or solution. However, compliance obligations are underrepresented in most of the published SRE processes and frameworks (e.g., [6]–[9]). Although most of them propose the use of certain SRE methods for requirements elicitation, they do not explicitly foresee the incorporation of other requirement sources, such as raw security requirements from compliance obligations. Only Mellado et al. [8] recommend to include legal, statutory, regulatory, and contractual requirements, however, they leave it open how it should be done in practice. Therefore, support must be provided to organizations and SRE practitioners to incorporate security information and knowledge as well as compliance obligations in a well structured and reusable way in order to mitigate the lack of security skills and knowledge and to avoid unnecessary efforts for identifying, understanding, and correlating applicable SRS on a project-wise basis.

1.2 Related Work

First approaches for the structuring and provisioning of reusable security information and knowledge propose the development, use, and improvement of a requirements repository or a knowledge base. In SIREN [10], the requirements repository is filled with countermeasures taken from MAGERIT [11], which were translated into security requirements. Mellado et al. [8] propose to store and reuse elements from Common Criteria. Dikanski and Abeck [12] propose to create reusable security requirements analysis templates (SecRAT) in order to develop and use a knowledge base, offering various relevant information, such as security standards, technologies, security models, principles, and policies, which can be reused for security requirements engineering. To the best of our knowledge, no model or framework exists, which combines compliance obligations, security information and knowledge sources, as well as results and artifacts from SRE methods in order to support the SRE activities and overcome the challenges as mentioned in the motivation section.

1.3 Contributions

With our research activities, we focus on the various types of security requirements sources, particularly on their categorization and structuring in order to provide them in a reusable form to practitioners. It is our goal to address the abovementioned problems by supporting organizations with the identification and structuring of relevant security requirements sources and, based on this, the creation of an organization-specific SRS repository, which fulfills the particular needs of an organization. The repository is intended to aid requirements engineers and development teams during the elicitation and analysis of security requirements by providing SRS in a reusable form and making the relation between the various SRS understandable and traceable.

To reach this goal, we put our research effort on the following aspects:

- Identification, structuring, and specification of a consistent classification for the most important SRS.

[1] We use the acronym SRS for singular as well as for plural throughout the paper.

- Development of a generic model, which can be used for structuring and reusing relevant SRS.
- Specification of criteria and a structured approach how to instantiate the generic model for a given scenario in order to create an organization-specific SRS repository.

1.4 Outline

The paper is structured as follows: Section 2 gives an overview and categorization of reusable security requirements sources, followed by the introduction of our generic model in Section 3. An overview about the industrial evaluation scenario and instantiation approach is presented in Section 4. A description of helpful views and benefits of the instantiated SRS repository is given in Section 5. Finally, we present a short summary and sketch limitation of our model and the evaluation scenario in Section 6.

2 Security Requirements Sources

To come to a categorization of reusable, security-specific security requirements sources to be considered in our model, we reviewed SRE literature and guides for requirements specification (e.g., [13]–[15]) regarding requirements sources with relevance for security and combined them with the common security engineering approach[2] as depicted in Figure 1.

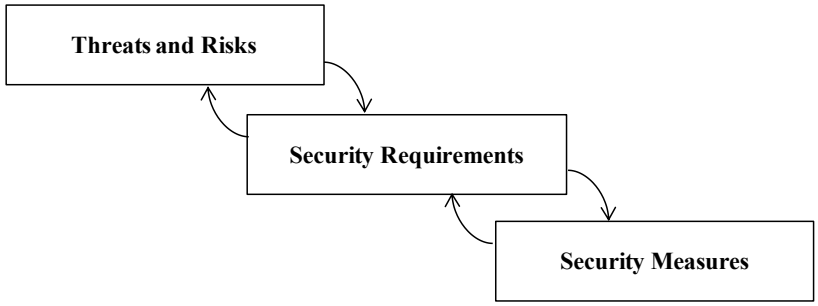

Figure 1. Security Engineering (based on [16])

The resulting extended view on the security engineering principle is depicted in Figure 2, showing the reusable categories of SRS as orange boxes.[3] The SRS categories *security information and knowledge* and *SRE methods* are used to identify what must not happen (i.e., threats, weaknesses, and vulnerabilities) in order to protect the confidentiality, integrity and availability of valuable assets and services. *Compliance obligations*[4]*,* as the third SRS category, impose raw requirements, which may need to be incorporated and analyzed together with the identified threats, weaknesses, and vulnerabilities when specifying security requirements.

[2] In this approach, security requirements are derived from threats and risks. The resulting requirements are used to design suitable security measures in order to fulfill security requirements and to counter the identified threats.

[3] Please note that from a security perspective there are many more potential security requirements sources (e.g., criticality of information, product or system, intended operational environment, domain aspects, market influences etc.). Unfortunately, they cannot be provided in a reusable fashion. For instance, market influences might have an influence also on the security (e.g., as an unique selling proposition), but they are very project-specific and must be addressed by the overall requirements engineering process since they are not only specific to or primarily motivated by security. More detailed information about source categories e.g., structure, examples, and interrelation between SRS can be found in [17].

[4] In literature, compliance obligations are also referred to as organizational standards and external regulations [13] or constraints and influences from the environment (e.g., political influence, market influence, or standards and technical policies) [15].

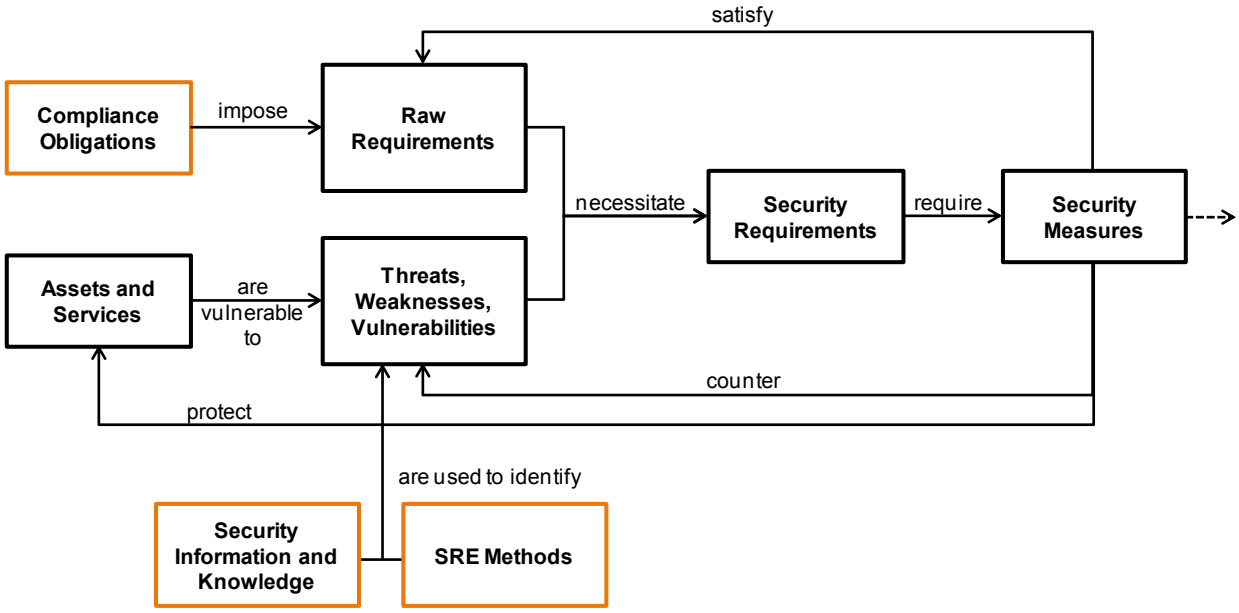

Figure 2. Extended Security Engineering

In the following, we briefly introduce all three SRS categories as the basis for description of our SRS model in Section 3.

2.1 Security Information and Knowledge Sources

Security information and knowledge is very multifaceted. We differentiate this source category in diagnostic and prescriptive information and knowledge, inspired by the knowledge base structure proposed by Barnum and McGraw [18].

2.1.1 Diagnostic Security Information and Knowledge Sources

Diagnostic security information and knowledge addresses the problem space by means of 'the bad things that might happen'. In other words, diagnostic security information and knowledge describes what needs to be avoided and should be addressed by security requirements as the basis for the design of security measures.

Examples of diagnostic information and knowledge sources are:

- Security threats: e.g., provided as lists of (mostly generic) threats in risk assessment guides [19], risk analysis methods [20] and risk management standards [21]
- Security weaknesses and vulnerabilities: e.g., provided in online catalogues or community developed dictionaries such as the Common Weakness Enumeration [22] and the Common Vulnerabilities and Exposures [23]
- Attack patterns: e.g., [24] and [25]
- Knowledge about exploits and hacker tools, meaning the knowledge about exploitable vulnerabilities, and corresponding exploitation tools

Hence, diagnostic security information and knowledge can either directly provide threats, weaknesses, and vulnerabilities as input (e.g., in the form of structured lists) to the SRE process, or provide attack patterns or knowledge about exploits and hacker tools, which might be used by an attacker to exploit potentially existing threats, weaknesses, or vulnerabilities.

2.1.2 Prescriptive Security Information and Knowledge Sources

Prescriptive security information and knowledge sources provide statements of practice about what to do, when building secure products and solutions. Prescriptive security information and knowledge ranges from high-level security principles (e.g., least privilege principle), over to guidelines for various security topics, up to rather concrete security controls and specific security design patterns. It can be assigned in many cases to the solution space, since it mostly prescribes *the what* (and sometimes also *the how*) in contrast to *the why* (as it is the case with diagnostic information and knowledge sources). Therefore, prescriptive security information and knowledge is typically also input for the design phase in a development lifecycle and not necessarily a primary source for the SRE process. The reason why we nevertheless reference the prescriptive security information and knowledge as input for SRE is due to the perception. Existing architectures may partially or fully influence the way how problems are structured. It may be useful or necessary to reverse engineer problems, for which a known solution exists from existing architectural designs [26], [27]. Specification and implementation are often intertwined in practice, since limitations of implementation technology may demand a specification change or implementation choices require the augmentation of the original specification [28]. This perception was adapted in the context of requirements and architecture in the twin peaks model [27], a simplified version of the spiral model. As further explained in Section 3.4, the model foresees a concurrent, spiral development processes in which requirements engineers and system architects work concurrently and iteratively increase the level of detail of both, the requirements specification and the architecture design. It is a distinct, but yet intertwined, activity of requirements engineering and architectural design. Therefore prescriptive information and knowledge may put a valuable input for the security requirements engineering process.

Here are the examples of prescriptive information and knowledge sources:

- Security principles [29], [30]
- Security guidelines [31]
- Security (design) patterns [32]
- Security control lists [33], [34]

2.1.3 Dependency between Diagnostic and Prescriptive Security Information and Knowledge Sources

Diagnostic and prescriptive information and knowledge sources have a strong relation, since diagnostic information and knowledge provides the basis for understanding and motivating application of prescriptive information and knowledge. Unfortunately, they are seldom provided together in a stringent and comprehensible fashion. Therefore the reason or motivation for application of prescriptive information and knowledge sources (i.e., the underlying threats, weaknesses, or vulnerabilities) is often not understandable for the user community.

2.2 Compliance Obligations

For enterprise systems as well as products and solutions, there are various potentially applicable compliance obligations, which need to be fulfilled. External compliance obligations represent legally binding or contractually agreed requirements (i.e., any law, statutory, regulatory, or contractual obligations), which must be identified and, if relevant, incorporated into the requirements engineering process. Examples range from regional and country-specific legislations (e.g., data privacy laws) to domain-specific obligations (such as HIPAA[5] and FDA[6] Part 11 for the healthcare sector). Besides external compliance obligations, often also company-

[5] Health Insurance Portability and Accountability Act

[6] Food and Drug Administration

or organization-internal compliance obligations have to be followed, such as information and IT security policies, standards, and guidelines. Depending on the respective laws, policies, standards, etc., various requirements engineering aspects can vary strongly (e.g., used terminology and extent and how the 'raw requirements' are specified). For instance, raw requirements, which need to be fulfilled, are named differently: 'security controls' [35], [36]; 'Security Functional Requirements (SFRs) and Security Assurance Requirements (SARs)' [37]; or 'Foundational Requirements' (FRs) [38]. Some of them provide rather high-level goals and security principles, while others primarily address the solution space (as it is typically the case with security controls). Furthermore, they address and partially also mix different scope areas, such as software, a system in its technical environment, or even organizational issues. Nevertheless, the raw requirements need to be initially identified, analyzed, and verified as part of SRE process.

2.3 SRE Methods

Various methods and approaches are mentioned and referenced in literature in the context of SRE. An overview and comparison of SRE methods can be found in [39] and [40]. Many of them are designed to analyze the problem space in systematic ways in order to identify threats, weaknesses, vulnerabilities, and attacks. The results are intended to be used as the basis for the specification of security requirements. Prominent methods and techniques (amongst others) are abuse cases [41], misuse cases [42], and attack trees [43]. Other frequently mentioned methods in the context of SRE are intended to improve the capabilities for modeling security requirements or security-specific information. Secure TROPOS [44] as extension of the i* modeling language is one example of such methods. Furthermore, two extensions of UML, namely UMLSec [45] and Secure UML [46] were introduced as a possible way to improve the integration of security-specific information into UML.[7] Some of the analysis-oriented methods utilized for SRE can also be applied for security architecture reviews, since the goal to reveal threats, weaknesses, and vulnerabilities remains the same for both. However, there is a difference regarding the level of detail, in which the methods are applied and the software or systems are analyzed. The methods analyzing abuse and misuse cases can be applied in early stages of the development project, since they are developed on the basis of use cases and therefore do not require detailed knowledge about the architecture and design. In contrast, methods used for security architecture and design reviews require and benefit from a certain level of detail about the architecture and design as well as knowledge about the intended operational environment. Examples of such methods are attack trees and methods for threat modeling such as STRIDE [47].

2.4 SRS View on Twin Peaks Model

Typically, SRE can only be completed in very exceptional small cases before the architecture and design phase starts. A 'big-bang' approach, in which all security constraints and influences can be considered and specified beforehand will seldom work in practice (see also [48]), since typically not all constraints and influences can be clarified in advance or they may change during the systems development. Furthermore, design decisions made on the basis of the initial set of security requirements and trust assumptions may introduce new influences and constraints such as threats and weaknesses. This might, in turn, result in the demand for additional security analysis and new requirements. Thus, SRE activities should not be performed sequentially in a single phase within the development lifecycle. Instead (security) requirements engineering and

[7] Please note that we limit our examples here on analysis and modeling-oriented approaches for the sake of readability and compactness. Besides these methods, also reuse-oriented approaches and methods emphasizing trust assumptions are available.

architecture design should be woven together in an incremental approach. To visualize and explain such an incremental approach for security, we use and extend the twin peaks model [49], which is shown in Figure 3. In this approach, security requirements and architectural specifications are developed concurrently after the initial set of security requirements has been identified. We furthermore use the model to assign different types of SRS to the respective peaks.

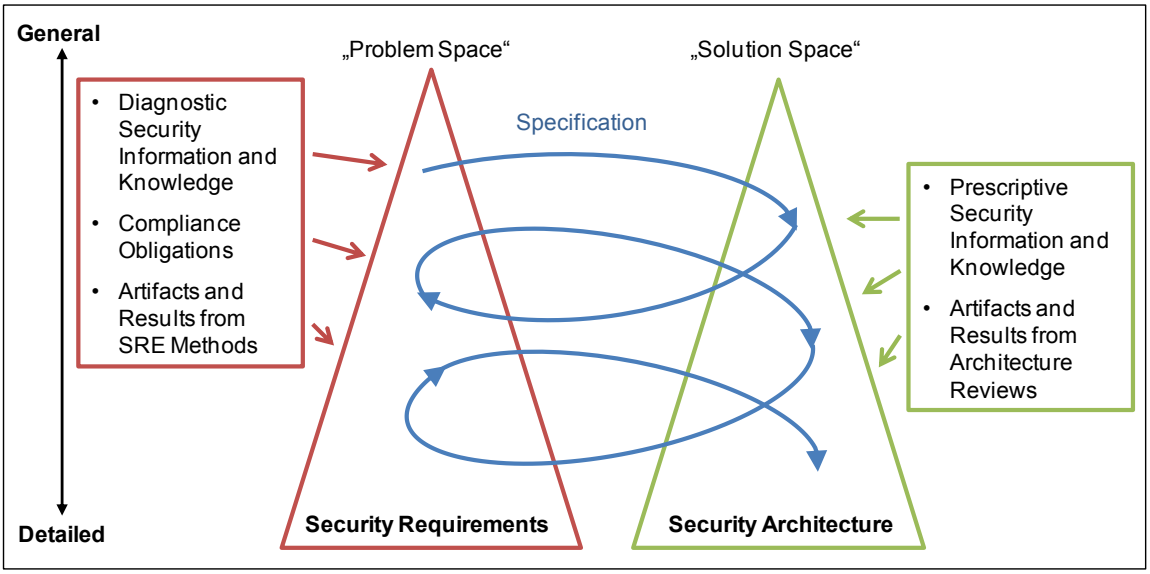

Figure 3. SRS in the twin peaks model

3 Model for Structuring and Reusing Security Requirements Sources

For structuring and reusing SRS, we developed a generic model, which can be used as a blueprint for the instantiation of an organization-specific SRS repository.

3.1 Desired Model Capabilities

The following desired capabilities for an SRS repository were derived from user stories, which were elaborated in the team of security experts and representatives from software and system development teams:

- **Scope.** The model shall be useable for software security requirements and also incorporate the system level as well as technical, physical, and organizational aspects.
- **Flexibility.** The model shall be flexible enough to structure (most of) the relevant SRS.
- **Relations between SRS.** The relationships between different kinds of security information and knowledge sources (e.g., diagnostic vs. prescriptive) shall be understandable.
- **Reuse.** The reuse of information and knowledge shall be incorporated to increase the efficiency of SRE and ensure the quality of security requirements.
- **Quality and Baseline Security.** It shall be possible to verify the quality and completeness of security requirements by means of ‘baseline security’[8] covering the most prevalent aspects of the problem space.

[8] With ‘baseline security’ we mean covering at least a predetermined set of typical threats, weaknesses, and vulnerabilities that should be addressed through security requirements.

3.2 SRS Model

Figure 4 shows the proposed model. It was our intention to develop the structure, which is as stable as possible and in which the relevant SRS can be structured and provided to development teams for specification of security requirements.

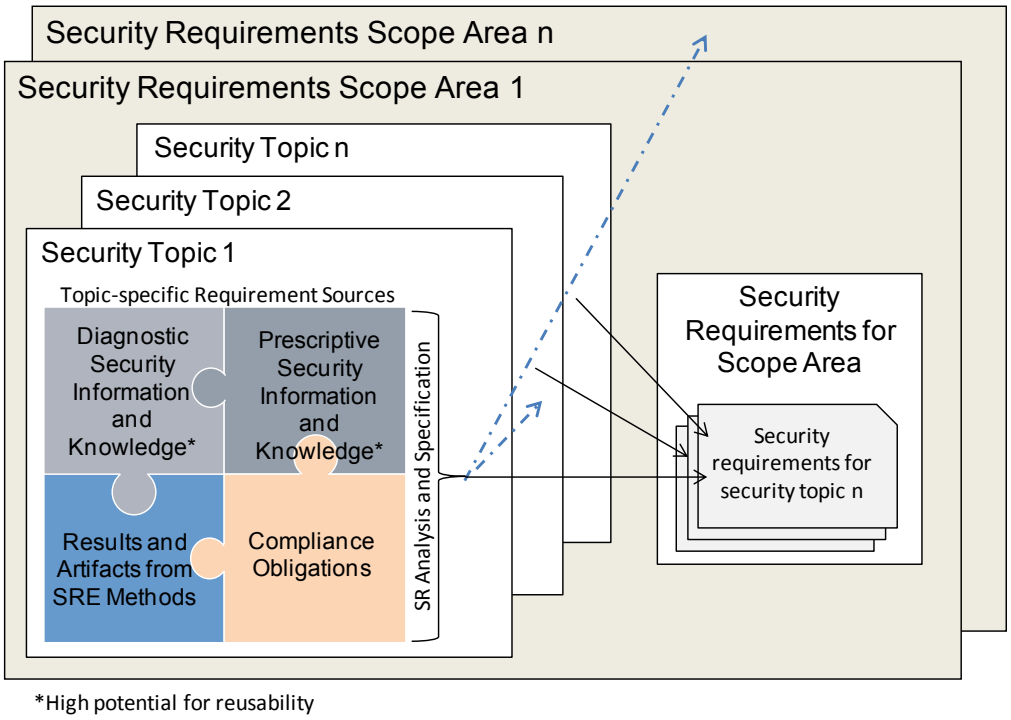

Figure 4. SRS Model

Security requirements scope areas and security topics are the two main structure elements of the model:

- The scope area determines the work to be accomplished, the problem space to be analyzed, and the topics to be covered when developing security requirements. The scope area consists of several security topics, for which topic-specific requirement sources are provided and analyzed. The examples of scope areas are 'software' or 'system'.
- The security topic consolidates relevant elements from SRS that support the analysis of the problem space and necessitate the specification of security requirements. It therefore inherits topic-specific SRS elements from the source categories diagnostic and prescriptive security information and knowledge, results and artifacts from SRE methods, as well as raw requirements from compliance obligations.[9,10]

To give an example for the proposed generic model, we use the security topic 'User Authentication' in the scope area 'Software' to which the following exemplary SRS are assigned:

- **Scope area:** Software
- **Security topic:** User Authentication
- **Diagnostic security information and knowledge** (provided as reusable information in the security requirements repository):
  - Threats (from threat lists) related to user authentication
  - Common authentication weaknesses (e.g., provided in CWE [22])

[9] The current model does only foresee one hierarchy level of security topics per scope area. Nevertheless, also parent-child relations between security topics are possible, e.g., for security topic "password security" as a child of "user authentication". For brevity and simplicity we omit the hierarchical structure of security topics in this paper.

[10] A more detailed overview of mutual implications between the scope areas and security topics is provided in [50].

- **Prescriptive security information and knowledge** (provided as reusable information in the security requirements repository):
  - Design pattern for username and password-based authentication
  - Design pattern for certificate-based user authentication
  - Design principles for user authentication (e.g., user authentication should be implemented on server-side and not on the client-side)
- **Compliance obligations.** Raw requirements relevant for User Authentication derived from:
  - Company Information Security policy on 'password security'
  - Company Information Security policy on 'secure access to IT systems'
  - Data privacy law / data privacy controls
  - Customer requirements on user authentication
- **Results and artifacts from SRE methods:**
  - Generic system overview of the software to be developed (including description of assets and provided functionality)
  - Misuse case(s) (e.g., developed along with use cases incorporating user authentication)
  - Relevant user authentication threats and risks identified via a threat and risk analysis

Figure 5 shows an entity relationship diagram of the proposed model.[11]

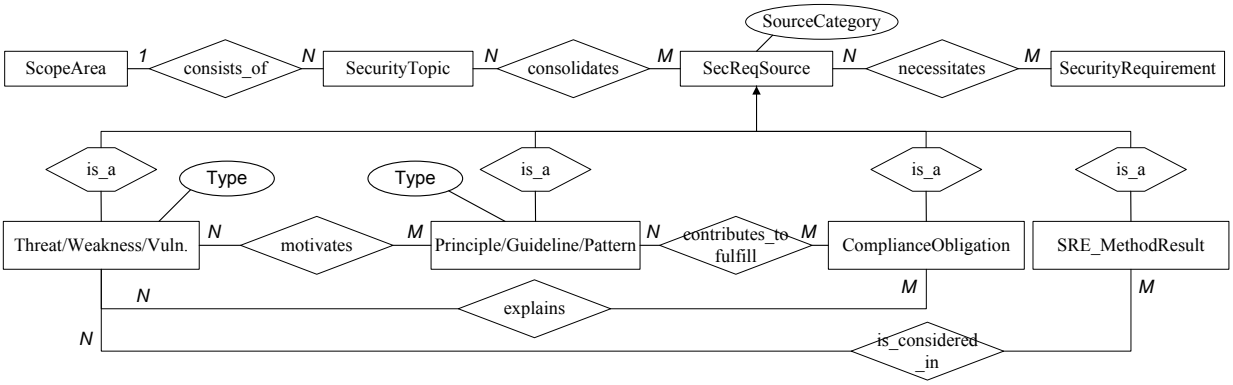

Figure 5. SRS Entity Relationship Model (ERM)

The general relationships between the SRS (i.e., SecReqSource subclasses) can be changed or refined if required, e.g., if more fine-grained relationships between different SRS need to be modeled or to ensure that the terminology is in line with the security ontology used in an organization.[12] For our purpose, the defined SRS subclasses and their relations are fine grained enough, since they are capable to reflect the general dependencies between the defined SRS categories.[13]

4 Evaluation

In this section we present the evaluation scenario, as well as the proposed instantiation approach to create an organization-specific SRS repository.

[11] For brevity reasons we omit most of the entity attributes.

[12] A literature survey of different security requirements ontologies is given in [51].

[13] Please note that additionally also relationships between SRS in a SRS category may be possible, e.g., if a threat can be assigned to a related weakness, or if raw requirements are similar and relate to other raw requirements derived from compliance obligations.

4.1 Evaluation Scenario and Scope

An organization-specific SRS repository must primarily fit to the needs of a particular organization. A one-fits-all structure of scope areas and security topics is not feasible due to varying SRS to be considered, differing businesses and use cases to be supported, as well as varying motivations for SRE. We chose the following industrial evaluation scenario to validate the applicability of the model by instantiating a relatively universal structure of scope areas and security topics, which we hope may also serve as a good starting point for other organizations:

- **Instantiation criteria and SRS to be considered.** The evaluation scenario is based on the instantiation criteria and SRS from a big engineering company. To gain a high significance and broadness of our evaluation, we selected SRS from three different corporate departments, namely, information security, product and solution security, and data privacy departments. Moreover, we extended the obtained list of SRS with additional SRS from external repositories. In total, we had to structure over 500 elements [14] from eight different SRS, covering at least one representative per SRS category, as depicted in Table 1. Due to this broad basis, different technical, physical, and organizational security aspects in different scope areas had to be structured and their relationships established.

Table 1. Scenario-specific SRS

SRS Category	SR Source
Diagnostic security information and knowledge sources	Corporate threat lists / questionnaires
	Selected weaknesses from CWE [22]
	Deliberate threats from ISO/IEC 27005 [21]
Prescriptive security information and knowledge sources	Corporate product security guidelines
	Corporate guideline for secure software development
Compliance obligations	Corporate information security control framework
	Corporate data privacy controls
Results from SRE methods	Results from three selected threat and risk analysis workshops

- **Business and use cases to be supported.** Based on the incorporated SRS, in our scenario, we support the classical enterprise information security business (i.e., corporate information, applications, and infrastructure), as well as departments developing products and solutions.
- **Motivation and drivers for security and SRE.** In our scenario the primary motivation for the instantiation of the model is to create a repository, which can be used for information security as well as product and solution security aspects. Thus, the resulting structure needs to be universal enough to fit the different SRS and the purposes of various organizational units.[15]

4.2 Instantiation Approach

Using the generic model (as presented in Section 3.2) and the SRS from the evaluation scenario (see Section 4.1), we applied the following steps to instantiate the model and to create the SRS repository (Figure 6).

[14] With SRS element we mean one element of a security requirement source, e.g., a single threat from a threat list or a raw requirement from a compliance obligation.

[15] Conversely, this means that the instantiated model is not structured according to a particular compliance standard (e.g., the company-wide information security control framework) or is specifically customized for a certain target group such as software or system architects.

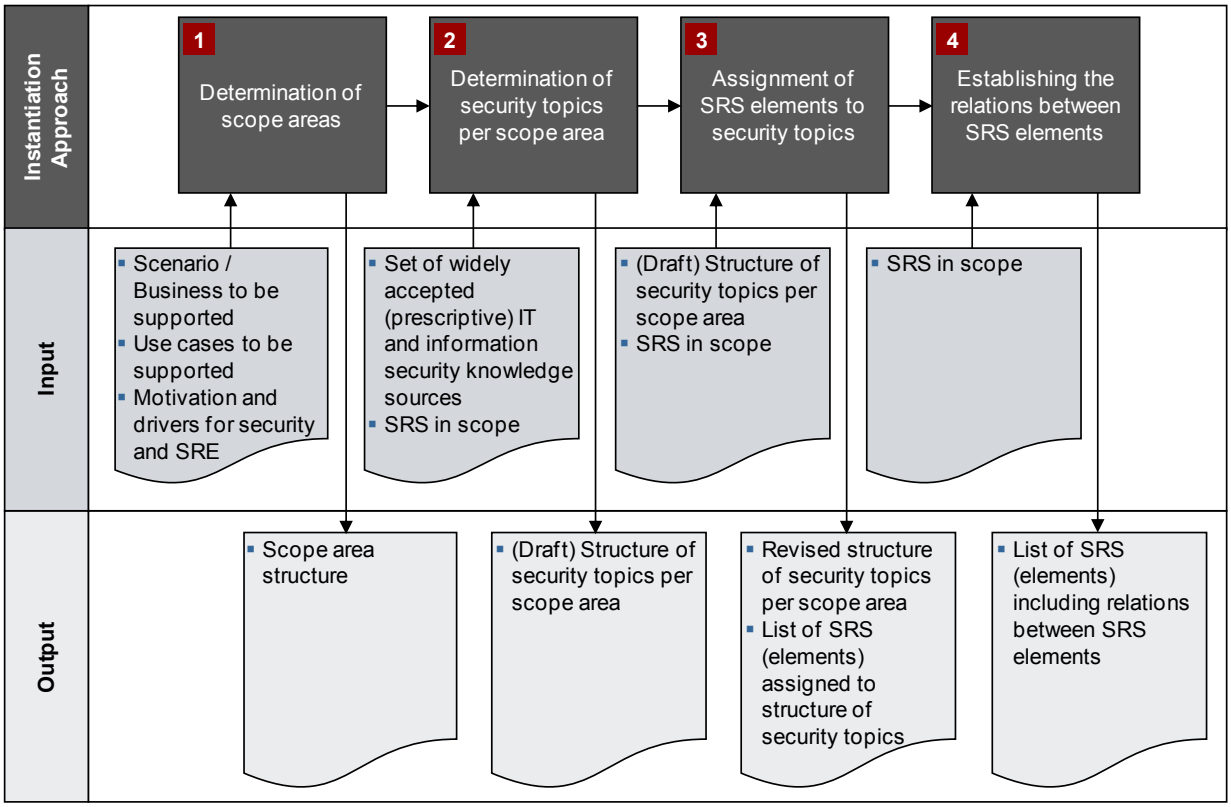

Figure 6. Instantiation approach

Step 1: Determination of scope areas

Concerning the evaluations scenario presented in Section 4.1, we distinguish between three scope areas *software*, *system in technical environment*, and *system in organizational environment*.[16] These three scope areas ensure that the model is useable for the software and system level as well as it refers to the aspects in the organizational environment. Thus, it is ensured that scope areas can be mapped to typical security aspects and responsibilities in the classical enterprise IT and also to typical product and solution businesses in organizations. Another advantage is that, if desired, organizational and process-related aspects can be treated separately from technical and physical aspects, which are often mixed up in security standards (e.g., [52]).

Step 2: Determination of security topics per scope area

To determine a suitable structure of security topics for three scope areas in the scenario, we use the following approach.[17]

a. **Development of a generic structure blueprint.** First, we chose a set of widely accepted IT and information security knowledge sources [22], [31], [34], [52], analyzed their structure and covered security aspects, and mapped it to the three above-mentioned scope areas. Thereby we identified common and recurring terminologies for

[16] A potential fourth scope area is development environment, which addresses all aspects required for securely developing a product or solution (e.g., system planning and acceptance, compliance). However, in this paper we focus on the development of secure software and systems and therefore omit aspects related to this scope area on purpose. More details on the characteristics and implications between them can be found in [50].

[17] Please note that this approach might need to be adapted for other scenarios and instantiation criteria, e.g., if compliance to a particular standard is the primary driver for SRE or structures of established and accepted SRS should be retained.

security topics. The resulting structure served as a draft structure of security topics per scope area, which can be used for the mapping of the SRS in the scope.

b. **Development of a scenario-specific structure.** As the second sub-step, we parsed through the elements of the SRS in the scenario. We primarily focused on prescriptive information and knowledge sources as well as compliance obligations, since they mostly already provide a consistent and 'security-topic-alike' structure. We mapped and assigned them to the structure developed in sub-step a. If an element did not fit at all into the existing structure, we noted it down separately for later analysis. Elements, which could not be uniquely assigned to a single security topic, e.g., because of their generic nature, were assigned multiple times. Thereby, we were able to derive the structure of security topics per scope areas and to assign most of the SRS elements.
c. **Troubleshooting.** As the third sub-step, we analyzed the SRS elements (from sub-step b), which did not fit into the existing structure and either reorganized the structure or created an additional security topic.
d. **Quality assurance.** To come to a consistent and acceptable structure, we did several iterations of revisions, resulting in the structure of scope areas and security topics shown in Figure 7.

Software	System in Technical Environment	System in Organizational Environment
• User authentication • Component to component authentication • Session management • Authorization • SW audit, logging and monitoring • SW data at rest • SW data in motion • SW update capabilities • SW cryptography and key management capabilities • Input validation and output sanitization • Software security documentation • Secure software best practices and paradigms	• System access control (generic) • System audit, logging and monitoring • System data at rest • System data in motion • Secure configuration and hardening • Client security • Server security • System architecture security • Physical security • Network security • Protection against malware • System update and patching capabilities • Backup and restore • System cryptography and key management capabilities • System security documentation	• Asset management • Information classification and handling • User and privilege management • Change management • Vulnerability and patch management • Certificate and key management • Monitoring and logging • Incident management • IT Service Continuity Management (ITSCM) / Business Continuity Management (BCM) • Provider and service level management • Operations and maintenance documentation • Organization of information security (internal and third party) • Personnel / HR security (HR) • Awareness and Training

Figure 7. Scope areas and security topics overview

Step 3: Assignment of SRS elements to security topics

As the next step we parsed through the elements of all SRS in our scenario and assigned them to the determined security topics per scope area, as derived in Step 2 (multiple assignments possible). Figure 8 shows one example for the assignment of an element from SRS 'Selected weaknesses from CWE [22]' to the concerned scope areas and security topics.

SRS Content			Assigned Scope Areas		
Weakness ID (#35)	Assigned Guideline Topic	Description Summary	Software	System in TE	System in OE
CWE-759: Use of a One-Way Hash without a Salt	Insecure Password Storage	The software uses a one-way cryptographic hash against an input that should not be reversible such as a password, but the software does not also use a salt as a part of the input.	# User authentication # SW cryptography and key management	# Access control # System data at rest	

Figure 8. Example: assignment of CWE element to scope areas and security topics

The scope areas and security topics were assigned to SRS element 'CWE-759: Use of One-Way Hash without salt'[18] because of the following rationales:

- **Software.** Security topics *'user authentication'* and *'SW cryptography and key management'*.
  Rationale: The use of a one-way hash without salt is a weakness, which primarily belongs to the security topic *'User authentication'*. Since the weakness is introduced by a weakness in a hashing algorithm, it is furthermore assigned to security topic *'SW cryptography and key management'*.
- **System in Technical Environment.** *'Access control'* and *'System data at rest'*.
  Rationale: Hashed passwords are stored somewhere on the system-level, e.g., in files of the operating system or somewhere in the database. Therefore, this weakness also relates to the security topic *'access control'* and *'system data at rest'*, since password has to be secured against unauthorized access and security of data at rest (in this case – passwords) must be ensured.

Step 4: Establishing the relationships between SRS elements

Once a suitable structure of scope areas and security topics is created and the respective SRS elements are assigned to security topics, the relationships between the various SRS elements must be established. Thereby, the relationships between the different kinds of SRS become understandable and traceable. In practice this means that for each element from the SRS it needs to be analyzed and documented to which other SRS elements it is related. Thus all relevant relationships between the security requirements source elements are identified and documented as marked in orange color in Figure 9.

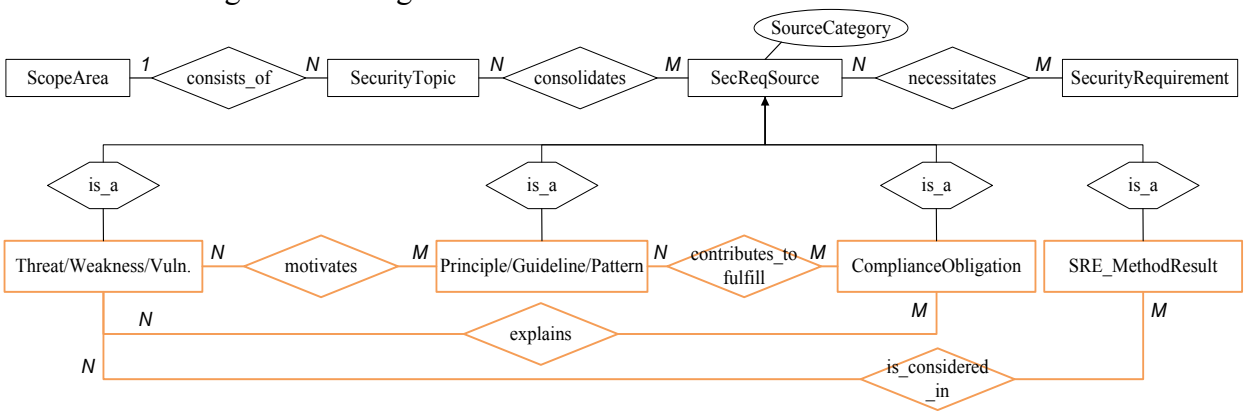

Figure 9. SRS relationships in ERM

[18] Background information on password storage and hashing: Software providing a username and a password-based authentication scheme typically does not store user passwords in plaintext, but do it in the form of a cryptographic hash value of the password. This is done to ensure that passwords are not available in plaintext and cannot be reversed if the password table was disclosed to an attacker. To enhance the security of hashed passwords, an additional random value (i.e., salt) should be added to the computation of a hash-value, in order to complicate so-called dictionary attack techniques.

Figure 10 shows the documented relationships from SRS element CWE-759 to all other relevant SRS elements with a relationship to this weakness in the instantiated repository. The example shows that the weakness motivates several elements from the SRS 'Corporate product security guidelines' and 'Corporate guideline for secure software development'. Furthermore, the weakness can be used to explain the reason for the implementation of proper access control and authentication mechanisms as a part of SRS 'Corporate information security control framework' and the implementation of a state-of-the-art password-based authentication mechanism as part of SRS 'Data privacy controls'.

SRS Content			SRS References			
Weakness ID (#35)	Assigned Guideline Topic	Description Summary	Corporate product security guidelines	Corporate guideline for secure software development	Corporate information security control framework	Data privacy controls
CWE-759: Use of a One-Way Hash without a Salt	Insecure Password Storage	The software uses a one-way cryptographic hash against an input that should not be reversible such as a password, but the software does not also use a salt as a part of the input.	# A-UA-001-C002: Consider using strong authentication mechanisms (two factor / multifactor) instead of username- / password-based mechanisms… # A-UA-007-D001: Do store passwords only on the server, but not on the client. # A-UA-007-D002: Do keep the passwords persisted only in a secure way, e.g., …	# SSD-37: Do not rely on the secrecy of key material or passwords in the client. # SSD-43: Require server authentication, if possible.	# 402001 – Access Control (general) # 402002-02-Authentication	# P9: Password-based authentication shall be state-of-the-art.

Figure 10. Example: Relationships between SRS Elements

5 Views and Benefits of the SRS Repository

Different views on the created SRS repository support the elicitation of security requirements.

5.1 Security Topic View

The security topic view shows all SRS elements, which are related to a security topic. Through this view all SRS elements from all incorporated SRS, which were related to this certain security topic, can be queried. For instance, the security topic view on user authentication in the scenario provides 9 threats from the corporate threat lists / questionnaires, 5 elements from the selected CWE [22], 4 deliberate threats from ISO/IEC 27005 [21], 38 recommendations from the product security guidelines, 7 recommendations from the corporate guideline for secure software development, 3 compliance obligations from the corporate information security control framework, and 6 data privacy controls grouped according to their security requirements source categories and SRS.

Several benefits arise from this view for practitioners as it helps to:

- Understand the problem space for a security topic and thus the motivation for utilizing related prescriptive information and knowledge as well as compliance obligations.
- Use the information about typical threats and weaknesses during the application of SRE methods.
- Consider existing recommendations and best practices when analyzing a security topic and aligning with software or system (security) architects viewpoints.
- Identify identical or similar raw requirements from different compliance obligations to increase the efficiency of fulfilling these requirements.

- Incorporate results from former applications of SRE methods to avoid recurring security problems.

5.2 SRS Element View

The SRS view shows all SRS elements, which have a relationship to a single SRS element. For instance, for an element from a compliance obligation all related diagnostic and prescriptive information and knowledge sources as well as results from SRE methods are provided, which help to understand and address the control. Thereby the following benefits arise:

- Possibility to view all relevant prescriptive information and knowledge sources as well as compliance obligations, for a diagnostic SRS element. This describes the motivation for application of prescriptive information and knowledge and helps to derive raw requirements from compliance obligations.
- Possibility to view all related diagnostic information and knowledge sources, as well as or a compliance obligations, for a prescriptive SRS element. This supports the mitigation of threats, weaknesses, and vulnerabilities and, moreover, helps to properly address raw requirements stemming from compliance obligations.
- Possibility to view all related diagnostic and prescriptive information and knowledge sources, which help to analyze and address a raw requirement or control from a compliance obligation.
- Possibility to view, which diagnostic information and knowledge sources are incorporated in SRE method result (e.g., an identified weakness from a threat and risk analysis).

5.3 SRS Coverage Views

Through the creation of relations between security requirements and incorporated SRS, a view showing the coverage of SRS can be created. As a prerequisite, for each specified security requirement, the relationships to the incorporated SRS elements, which are addressed by the requirement, must be documented. For instance, it must be documented, which raw requirements stem from compliance obligations or which threats are addressed by specified requirements. Inversely, these relationships enable to verify, which SRS are already addressed, e.g., by creating a query that shows all threats or weaknesses, for which there are no specified security requirements.

6 Summary and Limitations

6.1 Summary

We described the model for structuring and reusing security requirement sources, evaluated it in a comprehensive real-world industrial scenario, and showed that proposed model can be used for a variety of different SRS. The instantiation of the model in the described scenario showed that it is capable to fulfill the desired model capabilities. In particular, the following aspects contribute to the fulfillment of the desired model capabilities:

- **Scope:** We were able to define and successfully instantiate the structure of different scope areas incorporating relevant security topics in software, the system in its technical environment, and the system in its organizational environment.

- **Flexibility:** Using the proposed instantiation approach, we developed the structure of security topics for three defined scope areas, which was capable and therefore flexible enough to structure the SRS elements in the evaluation scenario.[19]
- **Relations between SRS:** Based on Step 4 in the instantiation approach, we identified and documented not only the relationships between different kinds of security information and knowledge sources, but also established the relationships between all elements from all SRS in the scope.
- **Reuse:** In general, the reuse of information and knowledge is reached through a proper instantiation of the model and the resulting SRS repository. Furthermore the development and provisioning of views on the instantiated model enables to query for the desired information for a variety of different purposes. Thereby, a comprehensive and consistent set of SRS is provided to practitioners, without requiring any additional effort for identifying, comparing and consolidating different SRS, which might support the elicitation of security requirements.
- **Quality and Baseline Security:** The desired model capability concerning the minimum quality and baseline security is supported by using the set of mandatory SRS (e.g., the most relevant threats, weaknesses, vulnerabilities, and attacks) when instantiating the model. Thereby, these SRS become part of the resulting SRS repository, which ensures that the most prevalent aspects of the problem space can be considered when eliciting security requirements. Using the SRS coverage views (see section 5.3), it can be verified, which of the SRS are already incorporated in the specified set of security requirements.[20]

6.2 Limitations

We see the following limitations related to the presented model and evaluation scenario:

- **Support for security requirements elicitation:** The proposed model is intended to complement existing SRE and security engineering processes and approaches by structuring and reusing SRS to support the elicitation of security requirements. Nevertheless, the undoubted standard activities in SRE processes (e.g., [6]), such as the specification of security objectives and identification and description of valuable assets to be protected should, of course, be done in addition, since they are also required for later SRE phases such as specification of system-specific security requirements. We therefore recommend to embed the organization-specific SRS repository into the organization-specific (S)RE process and align it with existing practices and ontologies.
- **Orientation of the model in the 'SRE world':** There is no common agreement, what a security requirement is [53] that results in various, partially contradicting, definitions for security requirements and thus also different security requirement specifications. Existing approaches do not agree whether the requirements should be limited to high-level security goals or have to be detailed to security measures [1]. This induces uncertainty among requirements engineering practitioners about the good practice in security requirements engineering. Although the proposed model of scope areas and security topics is generic and thus might be useable for different levels of granularity, we intended to design a security engineering focused SRE model in which the elicitation of security requirements is necessitated rather by concrete SRS than by high-level objectives. Therefore, the model and also presented evaluation scenario can be placed at the concrete, more functional, side of SRE. Moreover, we see the strengths of the model particularly in the context of an incremental development approaches (as depicted in

[19] The only SRS elements, which we did not consider in the instantiation, were overly generic and therefore left out. See also section 6.2 for limitations.

[20] Please note, that this property cannot be reached by the model alone, but also requires a predefined set of minimum SRS elements to be considered.

Section 3.4). It was not investigated so far, whether or not the model is usable in the setting, where only high-level security goals or security core principles (e.g., confidentiality, integrity, and availability) are used as model structure elements.

- **The level of detail of scenario-specific SRS must not deviate too much:** During the evaluation of the model, we experienced that the mapping of SRS, which differed significantly in their level of granularity, was very challenging and caused many unhelpful references. Particularly, the mapping related to diagnostic information and knowledge sources, may not always make sense or may finally result in extra effort for customizing either the SRS or the developed structure. Thus, the usability and final output depend on the incorporated SRS, particularly on their level of detail and the scope they address. We therefore recommend to carefully select SRS from external sources, which are not mandatory, in order to avoid a huge discrepancy of sources. For instance, the list of high-level threats from the information security risks management framework is not helpful SRS for the SRS repository to be primarily used for software security requirements elicitation.

References

[1] I. Tondel, M. Jaatun, and P. Meland, "Security requirements for the rest of us: A survey," IEEE Software Journal, 25(1), pp. 20–27, 2008. http://dx.doi.org/10.1109/ms.2008.19

[2] J. Wilander and J. Gustavsson. "Security requirements - a field study of current practice," in E-Proceedings of the Symposium on Requirements Engineering for Information Security, 2005, pp. 1-8.

[3] P. Salini and S. Kanmani. "Survey and analysis on security requirements engineering," Computers & Electrical Engineering, 38(6), pp. 1785–1797, 2012. http://dx.doi.org/10.1016/j.compeleceng.2012.08.008

[4] D. G. Firesmith, "Engineering security requirements," Journal of Object Technology, 2(1), pp. 53–68, 2003.

[5] G. Elahi, E. Yu, L. Tong, L. Lin. "Security requirements engineering in the wild: A survey of common practices," in Proc. 35th Annual IEEE International Computer Software and Applications Conference: COMPSAC 2011, 2011, pp. 314–319. http://dx.doi.org/10.1109/compsac.2011.48

[6] N. R. Mead, E. D. Hough, and T. R. Stehney. "Security quality requirements (SQUARE) methodology," in Proc. SESS'05 2005 workshop on Software engineering for secure systems—building trustworthy applications, New York: ACM, 2005, pp. 1-7. http://dx.doi.org/10.1145/1082983.1083214

[7] G. Sindre, D. G. Firesmith, and A. L. Opdahl. "A Reuse-based approach to determining security requirements," in Proc. 9th International Workshop on Requirements Engineering: Foundation for Software Quality (REFSQ'03), Klagenfurt, AT, 2003, p. 10.

[8] D. Mellado, E. Fernández-Medina, and M. Piattini, "A common criteria based security requirements engineering process for the development of secure information systems," Computer Standards & Interfaces, 29(2), pp. 244–253, 2007. http://dx.doi.org/10.1016/j.csi.2006.04.002

[9] G. Boström, J. Wäyrynen, M. Bodén, K. Beznosov, and P. Kruchten. "Extending XP practices to support security requirements engineering," in Proc. 2006 international workshop on Software engineering for secure systems (SESS '06), 2006, pp. 11–18. http://dx.doi.org/10.1145/1137627.1137631

[10] A. Toval, J. Nicolás, B. Moros, and F. García, "Requirements reuse for improving information systems security: a practitioner's approach: SIREN," Requirements Engineering Journal, vol. 6, pp. 205–219, 2001. http://dx.doi.org/10.1007/pl00010360

[11] F. Crespo, M. Gomez, J. Candau, J. A. Manas, MAGERIT – version 2: Methodology for Information Systems Risk Analysis and Management. II - Catalogue of Elements, Ministerio de Administrationes Públicas, Madrid, 2006.

[12] A. Dikanski and S. Abeck. "Towards a reuse-oriented security engineering for web-based applications and services," 7th International Conference on Internet and Web Applications and Services (ICIW 2012), 2012, pp. 282–285.

[13] G. Kotonya and I. Sommerville, "Requirements engineering: Processes and techniques," Chichester, NY: J. Wiley, 1998.

[14] IEEE Recommended Practice for Software Requirements Specifications: IEEE Std. 830, Institute of Electrical and Electronics Engineers, NY, 1998.

[15] IEEE Guide for Developing System Requirements Specifications: IEEE Std. 1233, Institute of Electrical and Electronics Engineers, NY, 1998.

[16] S. Myagmar, A. Lee, and W. Yurcik. "Threat modeling as a basis for security requirements, storage SS '05," in Proc. 2005 ACM workshop on Storage security and survivability, NY: ACM Press, 2005, pp. 94-102.

[17] C. Schmitt and P. Liggesmeyer. "A model for structuring and reusing security requirements sources and security requirements," in 1st Workshop on Continuous Requirements Engineering - CRE'15, 2015, pp. 28–37.

[18] S. Barnum and G. McGraw, "Knowledge for software security," IEEE Security and Privacy, 3(2), 2005, pp. 74–78. http://dx.doi.org/10.1109/msp.2005.45

[19] Guide for Conducting Risk Assessments: NIST Special Publication 800-30, 1st ed. Gaithersburg, Dept. of Commerce, National Institute of Standards and Technology, 2012.

[20] Information Security Forum. Information Risk Analysis Methodology (IRAM) [Online]. Available: https://www.securityforum.org/tools/isf-risk-manager/ (accessed 17th, June 2015)

[21] S. Klipper, "ISO/IEC 27005," in Information technology - Security techniques - Information security risk management, Génève, CH: ISO, 2011, pp. 63-97. http://dx.doi.org/10.1007/978-3-8348-9870-8_3

[22] The MITRE Corporation. Common Weakness Enumeration (CWE) [Online]. Available: http://cwe.mitre.org/ (accessed 17th, June 2015).

[23] The MITRE Corporation. Common Vulnerabilities and Exposures (CVE) [Online]. Available: http://cve.mitre.org/ (accessed 17th, June 2015).

[24] The MITRE Corporation. Common Attack Pattern Enumeration and Classification (CAPEC) Library [Online]. Available: http://capec.mitre.org/.

[25] A. Sethi and S. Barnum. Introduction to Attack Patterns [Online]. Available: https://buildsecurityin.us-cert.gov/articles/knowledge/attack-patterns/introduction-to-attack-patterns (accessed 17th, June 2015).

[26] M. J. Jackson, "Problem frames: analysing and structuring software development problems," Harlow: Addison-Wesley/ACM Press, 2001.

[27] B. Nuseibeh, "Weaving the software development process between requirements and architectures," 2001.

[28] W. Swartout and R. Balzer, "On the inevitable intertwining of specification and implementation," Magazine Communications of the ACM, 25(7), pp. 438–440, 1982. http://dx.doi.org/10.1145/358557.358572

[29] OWASP. CLASP Security Principles [Online]. Available: https://www.owasp.org/index.php/CLASP_Security_Principles (accessed 17th, June 2015)

[30] OWASP. Security principle [Online]. Available: https://www.owasp.org/index.php/Category:Principle (accessed 17th, June 2015)

[31] NIST. Special Publications (800 series) [Online]. Available: http://csrc.nist.gov/publications/PubsSPs.html (accessed 17th, June 2015)

[32] M. Schumacher, E. Fernandez-Buglioni, D. Hybertson, F. Buschmann, P. Sommerlad, "Security Patterns: Integrating Security and Systems Engineering," Chichester, GB: John Wiley & Sons, Ltd, 2006.

[33] Recommended security controls for federal information systems and organizations: SP 800-53, 3rd ed. [Gaithersburg, MD]: U.S. Dept. of Commerce, National Institute of Standards and Technology, 2009. http://dx.doi.org/10.6028/nist.sp.800-53

[34] SANS [Online]. Available: http://www.sans.org/critical-security-controls/ (accessed 17th, June 2015)

[35] Information Technology - Security Techniques - Management of information and communications technology security, ISO/IEC 27001, 1st ed., Geneva: ISO/IEC, 2004.

[36] NIST, "Security and privacy controls for federal information systems and organizations," in Information Systems and Organizations," 4th ed., Gaithersburg, MD: U.S. Dept. of Commerce, National Institute of Standards and Technology, 2013. http://dx.doi.org/10.6028/nist.sp.800-53r4

[37] Information Technology - Security Techniques - Evaluation Criteria for IT Security. Part 1: Introduction and General Model, ISO/IEC 15408-1, 2009. http://dx.doi.org/10.3403/01947025

[38] ISA99 Committee Website [Online]. Available: http://isa99.isa.org/ISA99%20Wiki/Home.aspx (accessed 17th, June 2015)

[39] B. Fabian, S. Gürses, M. Heisel, T. Santen, and H. Schmidt, "A comparison of security requirements engineering methods," Springer-Verlag New York, Inc, 2010. http://dx.doi.org/10.1007/s00766-009-0092-x

[40] D. Mellado, C. Blanco, L. E. Sánchez, and E. Fernández-Medina, "A systematic review of security requirements engineering," Computer Standards & Interfaces, 32(4), pp. 153–165, 2010. http://dx.doi.org/10.1016/j.csi.2010.01.006

[41] J. McDermott and C. Fox. "Using abuse case models for security requirements analysis," in Computer Security Applications Conference, 1999 (ACSAC '99), 1999, pp. 55–64. http://dx.doi.org/10.1109/csac.1999.816013

[42] G. Sindre and A. L. Opdahl. "Eliciting security requirements by misuse cases," 37th Conf. Techniques of Object-Oriented Languages and Systems, TOOLS Pacific 2000, pp. 120–131. http://dx.doi.org/10.1007/s00766-004-0194-4

[43] B. Schneier, "Attack trees," in Dr. Dobb's Journal of Software Tools, pp. 21–29, 1999.

[44] H. Mouratidis, P. Giorgini, G. Manso, and I. Philp. "A natural extension of tropos methodology for modelling security," in Workshop on Agent-Oriented Methodologies, 2002, pp. 91–103.

[45] J. Jürjens, "Towards development of secure systems using UMLsec," in Lecture Notes in Computer Science, Fundamental Approaches to Software Engineering, Springer Berlin Heidelberg, 2001, pp. 187–200. http://dx.doi.org/10.1007/3-540-45314-8_14

[46] T. Lodderstedt, D. Basin, and J. Doser, "SecureUML: A UML-based modeling language for model-driven security," in Lecture Notes in Computer Science, ≪UML≫ 2002 — The Unified Modeling Language, J.-M. Jézéquel, H. Hussmann, and S. Cook, Eds., Springer Berlin Heidelberg, 2002, pp. 426–441. http://dx.doi.org/10.1007/3-540-45800-x_33

[47] F. Swiderski and W. Snyder, "Threat Modeling," Redmond, WA: Microsoft Press, 2004.

[48] J. Meier, "Web application security engineering," IEEE Security and Privacy Magazine, 4(4), pp. 16–24, 2006. http://dx.doi.org/10.1109/msp.2006.109

[49] B. Nuseibeh, "Weaving together requirements and architectures," Computer, 34(3), pp. 115-117, 2001. http://dx.doi.org/10.1109/2.910904

[50] C. Schmitt and P. Liggesmeyer. "Implications of the operational environmental on software security requirements engineering," in WOSIS 2014 - 11th International Workshop on Security in Information Systems: SCITEPRESS, 2014, pp. 63–74.

[51] A. Souag, C. Salinesi, and I. Comyn-Wattiau. "Ontologies for security requirements: A literature survey and classification," in Lecture Notes in Business Information Processing, Advanced Information Systems Engineering Workshops, M. Bajec and J. Eder, Eds., Springer Berlin Heidelberg, 2012, pp. 61–69. http://dx.doi.org/10.1007/978-3-642-31069-0_5

[52] Information technology - Security techniques - Code of practice for information security management: Norme internationale ISO/IEC 27002:2005, Geneva: ISO, 2005. http://dx.doi.org/10.3403/30062176

[53] E. Dubois and H. Mouratidis, "Guest editorial: security requirements engineering: past, present and future," Requirements Engineering, 15(1), pp. 1–5, 2010. http://dx.doi.org/10.1007/s00766-009-0094-8