A Model for Structuring and Reusing Security Requirements Sources

Vom Fachbereich Informatik der

Technischen Universität Kaiserslautern

zur Verleihung des akademischen Grades

Doktor der Ingenieurwissenschaften (Dr.-Ing.)

genehmigte Dissertation von

Christian Schmitt

Datum der wissenschaftlichen Aussprache: 24.10.2017

Dekan:	Prof. Dr. Stefan Deßloch
Berichterstatter:	Prof. Dr. Peter Liggesmeyer
Berichterstatter:	Prof. Dr. Michael Waidner

D 386

Danksagung

An erster Stelle möchte ich meine generelle Dankbarkeit und Wertschätzung dafür aussprechen, dass ich in einem sozialen und politischen Umfeld aufwachsen durfte, welches mir die Rahmenbedingungen zum Studieren, wissenschaftlichen Arbeiten und zur Verfassung einer Doktorarbeit ermöglicht hat. Ich erachte das keinesfalls als Selbstverständlichkeit, sondern als besonderes Privileg, welches man nicht oder nur bedingt persönlich beeinflussen kann.

Prof. Dr.-Ing. Peter Liggesmeyer danke ich für die Betreuung der Promotion, insbesondere für seine Unterstützung und seinen fachlichen Rat. Caroline Frey, Dr. Max Steiner, Kai Bizik und Michael Roth danke ich für die organisatorischen und universitätsspezifischen Hilfestellungen, die mir das Leben als externer Doktorand deutlich erleichtert haben.

Uwe Blöcher danke ich für die inhaltliche, zeitliche und finanzielle Unterstützung der Promotion seitens der Siemens AG, ohne die eine Fertigstellung der Arbeit entweder gar nicht, oder nur sehr verzögert möglich gewesen wäre. Besonderer Dank bei der Siemens Corporate Technology gilt auch meinen Kollegen Dr. Martin Wimmer, Dr. Stefan Seltzsam, Dr. Jan Herrmann, Dr. Sebastian Ries und Steffen Fries, welche wertvolles Feedback im Rahmen von Diskussionen und Reviews von Ergebnissen gegeben haben. Überdies möchte ich Martin Rothfelder danken, über den ich damals den Kontakt zum Lehrstuhl und zu Professor Liggesmeyer erhalten habe, wodurch ich den ersten Schritt in Richtung Promotion gehen konnte.

Meiner Frau Christine Schmitt danke ich für ihre Unterstützung und ihre Liebe.

Kurzfassung

Der Forschungsschwerpunkt dieser Dissertation liegt auf den unterschiedlichen Quellen von Sicherheitsanforderungen (SRSs), insbesondere auf deren Kategorisierung und Strukturierung. Es wird ein Modell vorgeschlagen, welches die folgenden Eigenschaften erfüllt, um die aktuellen Herausforderungen aus dem Stand der Technik und dem Stand der Praxis zu adressieren.

- Umfang: Das Modell ist geeignet für Security-Anforderungen im Bereich Software, auf Systemebene sowie für technische, physikalische und organisatorische Aspekte.
- Flexibilität: Das Modell ist ausreichend flexibel, um (die meisten) Quellen für Security Anforderungen zu strukturieren.
- Wiederverwendung: Die Wiederverwendung von Informationen und Wissen wird unterstützt, um unnötigen Aufwand für die Identifizierung, das Verstehen, die Korrelation und die Anwendung von Elementen aus unterschiedlichen Quellen von Security-Anforderungen zu vermeiden.
- Beziehungen: Die Beziehungen zwischen den unterschiedlichen Arten von Security-Informationen und -Wissen (z.B. diagnostische vs. präskriptive Informationen) müssen verständlich sein.
- Qualität und Mindestsicherheit: Es ist möglich, die Qualitäts- und Vollständigkeitskontrolle zu unterstützen, um ein Mindestsicherheitsniveau zu erreichen und die häufigsten Security-Aspekte aus dem Problemraum zu betrachten.

Das Modell kann bei Organisationen für die Strukturierung und Bereitstellung von relevanten SRSs verwendet werden, und darauf basierend für die Erzeugung eines organisationsspezifischen, wiederverwendbaren SRS-Repositoriums zu dienen. Das Modell soll zu den konkreten Rahmenbedingungen einer Organisation passen, wie z.B. betroffener Umfang und Bereich sowie relevante SRSs, die einzubeziehen sind. Der sich daraus ergebende Vorteil für SRE-Praktiker, die das SRS-Repositorium verwenden, ist eine wiederverwendbare, umfassende und konsistente Struktur von SRS-Elementen, die direkt verwendet werden kann, ohne in einem Projekt zusätzlichen, unnötigen Aufwand

für die Identifizierung, den Abgleich und die Korrelation der verschiedenen SRSs zu erzeugen. Beziehungen zwischen den diversen SRS-Elementen werden bereitgestellt, die üblicherweise für Praktiker ohne Security-Expertise nicht ersichtlich wären. Überdies ermöglichen unterschiedliche Sichten auf das instanziierte Modell einen komfortablen Weg für die Nutzung der SRS-Elemente, deren Beziehungen und der zugehörigen erläuternden Informationen je Element.

Um dieses Ziel zu erreichen, werden im Rahmen dieser Dissertation folgende Beiträge erstellt:

- Klassifizierung und Beschreibung der wichtigsten wiederverwendbaren SRSs.
- Entwicklung eines generischen Modells, welches für die Strukturierung und Wiederverwendung von SRSs verwendet werden kann.
- Entwicklung eines strukturierten Ansatzes zur Instanziierung des generischen Modells für ein gegebenes Szenario zur Erstellung eines organisationsspezifischen SRS-Repositoriums.

Das Modell und der Instanziierungsansatz wurden anhand eines Beispiels innerhalb der Siemens AG evaluiert, unter Verwendung von SRSs aus drei verschiedenen Abteilungen, welche um weitere firmenfremde, gängige SRSs aus IT Security Online-Repositorien ergänzt wurden. In Summe wurden über 500 SRS-Elemente aus acht verschiedenen SRS-Kategorien klassifiziert, strukturiert und in einem wiederverwendbaren, Siemens-spezifischen SRS Repositorium bereitgestellt.

Abstract

The research focus of this PhD thesis is on the various types of existing Security Requirements Sources (SRSs), particularly on their categorization and structuring. A model is proposed that provides the following properties in order to address the current challenges in the state of the art and the state of the practice.

- Scope: The model can be used for software SRs and for the system level, as well as for technical, physical, and organizational aspects.
- Flexibility: The model is flexible enough to structure (most of) the relevant SRSs.
- Reuse: The reuse of information and knowledge is supported to avoid unnecessary effort for the identification, classification, comparison, correlation, and use of elements from different SRSs.
- Relationships: The relationships between different kinds of security information and knowledge (e.g., diagnostic vs. prescriptive) shall be understandable.
- Quality and Baseline Security: It is possible to support checks for quality and completeness of SRs in terms of 'baseline security', covering the most prevalent security aspects of the problem space.

The model can be used by organizations for the structuring and provision of relevant SRSs and, based on this, the creation of an organization-specific, reusable SRS repository that fits the particular circumstances of an organization, such as the affected scope and domain, and relevant SRSs to be incorporated. The benefit for practitioners using such an SRS repository is a reusable, comprehensive and consistent structure of relevant SRS elements, which can be used without requiring any additional, unnecessary effort for identifying, comparing, and consolidating different SRSs for each project. Relationships between the various SRS elements are provided, which normally would not be clear to practitioners without security expertise. Moreover, different views on the instantiated model provide a convenient way to make use of the SRS

elements, their relations and the associated explanatory information for each SRS element.

To reach this goal the following contributions are developed as part of this PhD thesis:

- Classification and description of the most important reusable SRSs.
- Development of a generic model that can be used for structuring and reusing relevant SRSs.
- Development of a structured approach for instantiating the generic model for a given scenario in order to create an organization-specific SRS repository.

The model and the instantiation approach were evaluated on the example of Siemens AG, using SRSs from three different corporate departments, in conjunction with other non-company SRSs from external state of the art IT security repositories. In total, over 500 SRS elements from eight different SRSs were classified, structured, and provided in a reusable, Siemens-specific SRS repository.

Contents

List of Figures

List of Tables

1

Introduction and Motivation

"Unfortunately, most of the work for traditional requirements engineering fails to explicitly consider security"
Unknown author, taken from [15]

In this chapter the general introduction and motivation for this PhD thesis will be described.

First, a categorization of the PhD topic will be given in section 1.1, followed by an overview of challenges and research gaps in Security Requirements Engineering (SRE) related to this work in section 1.2. The business demands to be solved and the research hypotheses to be verified will be presented in section 1.3. Moreover, an overview of contributions originating from this PhD thesis, the selected research approach, as well as the document structure will be presented in sections 1.4 to 1.6.

1.1 Categorization of PhD Topic

This PhD thesis is concerned with the structuring and reuse of Security Requirements Sources (SRSs) as a basis for the elicitation of Security Requirements (SRs). Therefore, it is an interdisciplinary research topic between the research fields Requirements Engineering (RE) and Security.

RE is the first of multiple phases in the System Development Life Cycle (SDLC). It is a very important phase for development projects since the overall project success significantly depends on properly developed requirements. Furthermore, errors made during the requirements engineering activities may cause significantly higher effort to rectify them in the later phases. SRs are developed just like all other functional and non-functional requirements in a Requirements Engineering process. However, the sources and techniques for Security Requirements Engineering differ strongly from 'traditional'

RE. Most 'traditional' requirements elicitation techniques are designed to discover requirements from human stakeholders. In contrast, it is the goal of SRE and security in general to identify, analyze, and prevent bad things that might happen. The aim to prevent harm from occurring to assets, to reduce the risk for an organization, and to ensure confidentiality, integrity, and availability. Therefore, SRE requires different types of information, knowledge, and techniques than 'traditional' RE.

1.2 Challenges and Research Gap in 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 that can be used immediately in projects [129]. In the literature on SRE, it is often stated that the application of SRE methods and techniques in projects and the maturity of Security Requirements (SRs) specifications is mostly poor (e.g., [96, 133]). Moreover, of the various methods that have been developed for SRE, just a small number have been used in practice to date [96].

Two aspects contribute to the sparse use of SRE methods and the low number and poor quality of SR specifications. The first aspect is the *lack of knowledge and skills for security and SRE* (e.g., [23]), which is considered one of the main challenges for the development of proper SRs. If one does not understand the mindset of an attacker, typical threats and weaknesses as well as available exploits, the results from the use of 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 had been involved. The second aspect relates to the various kinds of security information and knowledge sources that could potentially be used by the SRE community as sources to support the application of SRE methods and the elicitation of SRs. With proper selection and reuse of security information and knowledge, at least some of the skills- and knowledge-related problems could be addressed. However, Elahi et al. conclude in their survey on SRE [29] that *security information and knowledge sources are seldom used in practice.*

Besides these two aspects, the *increasing number of internal and external compliance obligations* is seen as an important source to be considered as input for SRE processes and methods. In practice, compliance obligations are very important for organizations since non-compliance can have a highly negative business impact due to delays or may even result in non-certification of a product or solution. However, compliance obligations are underrepresented in most of the published SRE processes and frameworks (e.g., [6, 69, 73, 115]). 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 SRs from compliance obligations. Mellado et al. [73] recommend the inclusion of legal, statutory, regulatory, and contractual requirements; however, they leave open how this should be done in practice.

Security information and knowledge sources and compliance obligations differ regarding scope, terminology, extent, and level of detail. Furthermore, the number and heterogeneity of Security Requirements Sources (SRSs) and compliance obligations is very high. They can provide information for different software-, system, organizational, or life-cycle security aspects. As an example, the code of practice for information security management [48] provides a mix of high-level software-, system-, and process-related control objectives for different predefined security categories such as access control, whilst the common weakness enumeration [127] provides an online repository with a focus on software related weaknesses. Sources for SRE are therefore often not directly comparable in terms of scope, structure, terminology, and level of detail and must therefore be understood and correlated by the practitioner on a project-wise basis, which in turn requires security knowledge and skills.

As a consequence, development teams and (S)RE practitioners without security skills and knowledge are caught in the dilemma that they require security and information sources to overcome their lack of security skills and knowledge, but at the same time require profound security knowledge and expertise to be able to identify, classify, compare, correlate and use security information and knowledge sources to benefit from them.

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 effort for identifying, understanding, and correlating applicable SRSs on a project-wise basis.

An approach and model is required to:

- classify and structure SRSs according to their scope and intended use;
- compare and correlate elements from different SRSs;
- provide and reuse relevant elements from different SRSs for the required purpose.

First approaches for the structuring and provision of reusable security information and knowledge propose the development, use, and improvement of a requirements repository or a knowledge base. In the SImple REuse of software requiremeNts (SIREN) approach [130], the requirements repository is filled with countermeasures taken from MAGERIT [74] that were translated into SRs. Mellado et al. [73] propose storing and reusing elements from Common Criteria (CC). Dikanski and Abeck [16] propose the creation of reusable Security Requirements Analysis Templates (SecRAT) in order to develop and use a knowledge base, offering various pieces of relevant information, such as security standards, technologies, security models, principles, and poli-

cies, which can be reused for SRE. Other approaches propose providing SRs in the form of templates [25] or pattern [11,108]. However, no model or framework exists that 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 above-mentioned challenges.

1.3 Business Demands and Research Hypotheses

The following desired capabilities for an SRS repository were derived from user stories elaborated by a team of security experts and representatives from software and system development teams.

The desired capabilities also serve as the research hypotheses, which form the basis for the validation of the model developed (cf. chapter 3.2) in the evaluation scenario (cf. chapter 4):

- **H1 - Scope:** The model can be used for software SRs and for the system level, as well as for technical, physical and organizational aspects.
- **H2 - Flexibility:** The model is flexible enough to structure (most of) the relevant SRSs.
- **H3 - Reuse:** The reuse of information and knowledge is supported to avoid unnecessary effort for the identification, classification, comparison, correlation, and use of elements from different SRSs.
- **H4 - Relations between SRSs:** The relationships between different kinds of security information and knowledge (e.g., diagnostic vs. prescriptive) shall be understandable.
- **H5 - Quality and Baseline Security:** It is possible to support checks for quality and completeness of SRs in terms of 'baseline security', covering the most prevalent security aspects of the problem space.

1.4 Contributions

The research focus of this thesis is on the various types of SRSs, particularly on their categorization and structuring. The aim is to address the above-mentioned problems by supporting organizations with the structuring and provision of relevant SRSs and, based on this, the creation of an organization-specific, reusable SRS repository that fits the particular circumstances of an organization, such as the affected scope and domain, and relevant SRSs to be incorporated. The benefit for practitioners using such an SRS repository is a reusable, comprehensive, and consistent structure of relevant SRS elements that can be used without requiring any additional, unnecessary effort for identifying, comparing, and consolidating different SRSs for each project. Relations between the various SRS elements are provided, which normally would not be clear to practitioners without security expertise. Moreover, different views on the instantiated model provide a convenient way to make use

of the SRS elements, their relations, and the associated explanatory information for each SRS element. To reach this goal, the following contributions are developed as part of this PhD thesis:

Contribution 1. Classification and description of the most important reusable SRSs.

Contribution 2. Development of a generic model that can be used for structuring and reusing relevant SRSs.

Contribution 3. Development of a structured approach for instantiating the generic model for a given scenario in order to create an organization-specific SRS repository.

1.5 Research Approach, Results and Information Validation

1.5.1 Engineering Method

The scientific approach used in this thesis is depicted in Figure 1.1. It is based on the scientific method of the experimental software engineering paradigm [4]. In this engineering method, the world is observed[1], a new or better model is developed[2], and the resulting model is validated with regard to explicitly stated hypotheses[3].

The engineering method used for this thesis is structured into the following four phases:

1. **State of the practice observations:** To identify the current state of the practice in SRE, relevant information gathered during the literature review is used in conjunction with the results from informal interviews (conducted by the author) and user stories developed in a project at Siemens. Thereby, practical shortcomings in current SRE practice as well as desired model capabilities and business needs for the development of a model for structuring and reusing SRSs are derived and documented.
2. **State of the art analysis:** To structure and analyze the state of the art in SRE and to identify potential research gaps, information is gathered through a literature review. Furthermore, the desired model capabilities resulting from the state of the practice observations are incorporated to identify potential solutions already existent in the state of the art. The results of the second phase are a state of the art overview incorporating different kinds of SRSs, methods, processes, and frameworks used for SRE. Furthermore, the identified research gap in the state of the art is described.

[1] cf. phase 1 and 2 of the research approach
[2] cf. phase 3 of the research approach
[3] cf. phase 4 of the research approach

3. **Design and improvement of generic model:** For the development of the generic model, the desired model capabilities and research hypotheses and the state of the art overview and research gap are used as input for the design of the generic model. The results of this phase are the classification of important SRSs to be considered in the model and the generic model itself. Moreover, an entity relationship diagram describing the generic model in greater detail is developed as a basis for the fourth phase.
4. **Instantiation and evaluation of the generic model and verification of research hypotheses:** In the last phase of the research approach, the generic model is instantiated together with the entity relationship diagram using a predefined evaluation scenario. The instantiated model is validated against the desired model capabilities and research hypotheses. In addition to the evaluation results, the limitations of the model and potential future work are described.

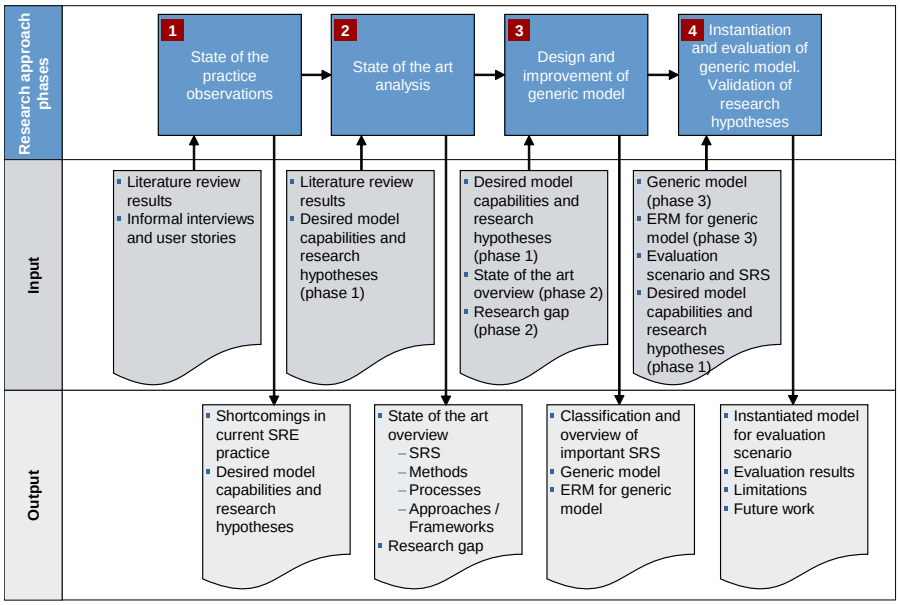


Fig. 1.1. Research approach overview

1.5.2 Additional Validation and Reviews

To receive reviews and feedback from the academic community and industrial practitioners, all contributions and related aspects shown in this PhD thesis were presented to the Requirements Engineering (RE) and Security

communities at the Siemens Software Excellence Conference [99], the 11th International Workshop on Security in Information Systems (WOSIS'14) [101], the 1st Workshop on Continuous Requirements Engineering (CRE'15) [102], the 2nd IEEE Workshop on Evolving Security and Privacy Requirements Engineering (ESPRE'15) [104], and in the journals Complex Systems Informatics and Modeling (CSIMQ) [105] and OBJEKTspektrum [103].

1.6 Outline

This thesis is structured as follows:

In chapter 2 the state of the art and the state of the practice in SRE are presented, providing the basis for the categorization of reusable sources for SRs to be used in the model.

All parts of the model for structuring and reusing SRSs are described in chapter 3. This includes the categorization of reusable SRSs (corresponding to contribution 1), the developed meta-model and entity relationship diagram (i.e., contribution 2), the approach for instantiating the generic model, and all relevant views and resulting benefits of the model.

The evaluation of the model is described in chapter 4 (i.e., contribution 3).

The thesis closes with a summary of the results and contributions along with the presentation of limitations of the model and the evaluation scenario, potential future work, and concluding remarks in chapter 5.

2

State of the Art and State of the Practice in SRE

> "It requires creativity, experience, and a different mindset to define the bad things that could happen." [134]

From a security perspective, the consequences of poor SRE motivate that security considerations should be made in the earlier phases of the development life-cycle (cf. section 2.2.2). In recent years the former paradigm of 'penetrate and patch' has been replaced by security software development at the design and coding stage [65]; however this has not yet reached the RE phase in many cases. A recent survey [29] indicates that SRs are seldom explicitly elicited and documented as part of the RE phase, but instead are mostly considered during the implementation phase.

In the literature on SRE, it is often stated that the application of SRE methods and techniques in projects and the maturity of SR specifications are mostly poor (e.g., [133]) or do not exist at all.

In this chapter, an overview of the current state of the art and state of the practice in SRE will be presented. Besides providing a general overview, the information serves as input for the derivation of reusable SRSs, as further used in chapter 3.1.

First, a general introduction to RE and SRE will be presented in sections 2.1 and 2.2. Thereafter, important aspects of SRE will be described, structured according to the SRE overview as shown in Figure 2.1. A distinction will be made between *customer and environment*, *security information and knowledge sources*, and *SRE methods*. Moreover, an overview of specific *security requirements engineering processes* will be provided.

2.1 Introduction to Requirements Engineering

In this section, the general role and importance of RE in software and systems engineering will be presented, and the particularities of requirements elicitation will be described.

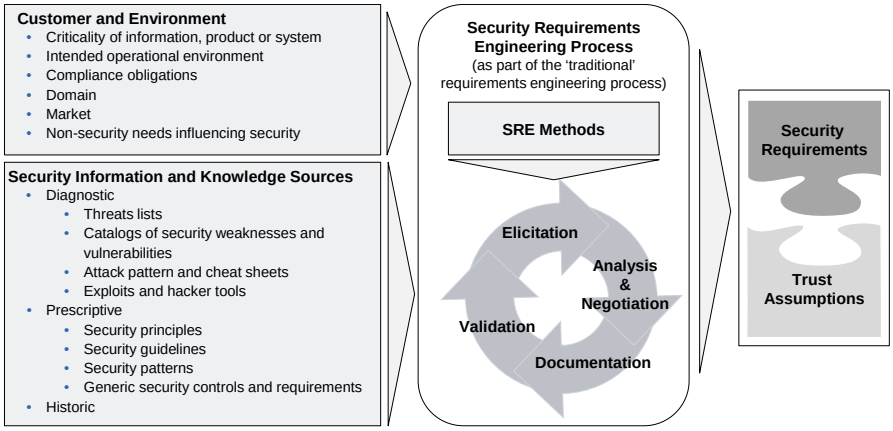


Fig. 2.1. SRE overview

The placement of RE in the development life-cycle will be illustrated in section 2.1.1, followed by an overview of example consequences of poor RE in section 2.1.2 in order to underline the importance of RE. In section 2.1.3, an overview of existing definitions and classifications of requirements will be given. Thereafter, the coarse-grained RE process model will be presented in section 2.1.4 and an introduction to requirements elicitation and elicitation techniques will be given in section 2.1.5. Finally, an overview of requirements sources to be considered according to the existing RE literature and standards will be provided in section 2.1.5.

2.1.1 Requirements Engineering in the Development Life-Cycle

The traditional SDLC is a commonly accepted model for the engineering of software and systems. Exemple representatives of software development life-cycle models are the sequential model (of which the waterfall model is a very popular form), the V-model, or the spiral model [2]. Although different terminologies regarding the traditional SDLC are used by various authors in the research community, all basically include similar process phases as shown in Figure 2.2. Within the SDLC, the first two process phases are concerned with software and system requirements engineering. As these requirements constitute the basis for all following development activities, all subsequent phases of the life-cycle benefit from properly developed and documented requirements. All subsequent phases may in turn suffer from poorly developed requirements, as presented in section 2.1.2.

2.1.2 Consequences of Poor Requirements Engineering

Although the number of 'scientifically accepted' investigations on the consequences of poor RE is very small, the following examples provide an overview

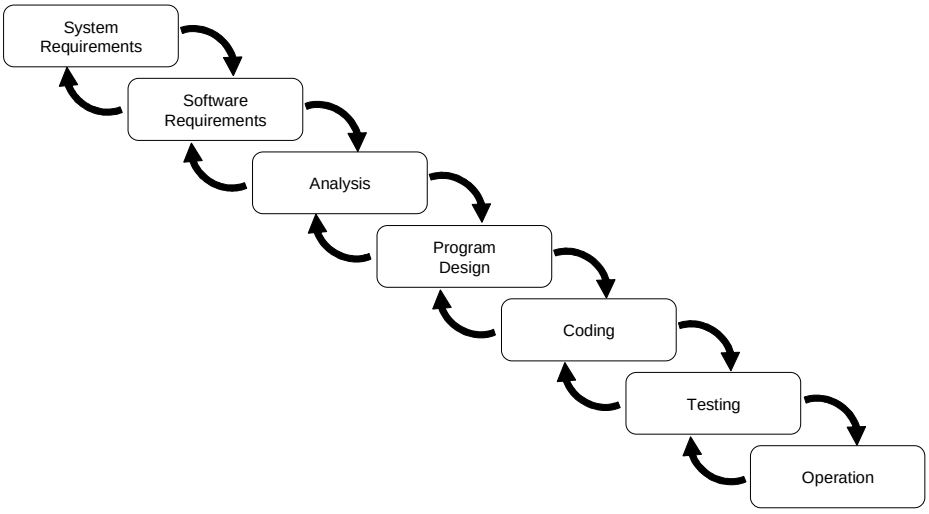


Fig. 2.2. Waterfall System Development Model according to W. W. Royce [95]

of the importance of RE. An empirical investigation of the reasons for software project cancellation [19] showed that in 2005, approximately 16 percent and in 2007, 12 percent of software projects (in the scope of the survey) were canceled before they delivered anything. In total, the rate of unsuccessful or canceled projects was approximately 34 percent in 2005 and 26 percent in 2007.

The following top three reasons for project cancellations are listed in the report:

- Too many requirements and scope changes (33 %)
- Senior management not sufficiently involved (33 %)
- Lack of necessary management skills (28 %)

The report shows that poor requirements engineering and management is one of the most relevant issues for software project cancellations. This perception is further substantiated by the theory, that the costs of correcting an error increase exponentially with each phase of the systems development life-cycle that the correction is delayed [5]. Therefore, errors made during the RE phase have a high impact on the overall success as well as on the schedule and budget of development projects.

The direct consequences of inadequate RE are [57]:

- The requirements do not reflect the real needs of the customer of the system.
- Requirements are inconsistent and / or incomplete.

- It is expensive to make changes to requirements after they have been agreed.
- There are misunderstandings between customers, those developing the system requirements, and software engineers developing or maintaining the system.

These consequences again may lead to the following negative impacts [9, 57]:

- Projects are past schedule and/or significantly over budget. The system may be delivered late and cost more than originally expected.
- Projects are significantly reduced in scope or are canceled.
- The customer and end users are not satisfied with the system.
- Products are not significantly used once delivered.
- Development teams deliver poor-quality applications.
- The system may be unreliable in use, with regular system errors and crashes disrupting normal operation.
- If the system continues in use, the costs of maintaining and evolving the system are usually very high.

These negative consequences show that requirements engineering is of high importance for the overall success of the software and system development. RE therefore plays a significant role in the software and systems engineering discipline.

2.1.3 Definition of Requirements Engineering

In the standards and in the literature related to computer science, several definitions for the terms *requirement* and *engineering*, as well as for the composed term *Requirements Engineering* exist. This section will provide an overview and brief examination of the most relevant definitions.

Definition of Requirement

Typically, there is an implicit understanding of the term *requirement* such as "a thing that is needed or wanted" [90]. Transferred to software or systems, the following definitions of the term *requirement* can be found: The ISO/IEC/IEEE standard for RE as a part of systems and software lifecycle processes [52] defines a requirement as a "statement which translates or expresses a need and its associated constraints and conditions". Another requirements definition given in the Project Management Body of Knowledge (PMBOK) [92] states that it is "a condition or capability that must be met or possessed by a system, product, service, result, or component to satisfy a contract, standard, specification, or other formally imposed document. Requirements include the quantified and documented needs, wants, and expectations of the sponsor, customer, and other stakeholders."

One software-specific definition of requirement is given in [57], which states: "Software requirements express the needs and constraints placed on a

software product that contribute to the solution of some real-world problem". Moreover, it is pointed out that "for many types of systems, it is impossible to separate the requirements for the software from broader requirements for the system as a whole. As well as software, the system may include computer hardware, other types of hardware device which are interfaced to the computer and the operational processes which are used when the system is installed in some working environment" [57]. Although definitions vary in scope and focus, they share similar aspects. The following common ground amongst the definitions can be stated:

- Requirements are conditions or capabilities that must be met or possessed by software, a system, or a system component.
- Requirements originate from various stakeholder needs, wants, and expectations as well as other constraints. Mentioned examples of stakeholders are users, sponsors, customers, and 'other' stakeholders. "The term stakeholder generalizes the traditional notion of customer or user in requirements engineering to all parties involved in a system's requirements" [26]. Examples of constraints are contracts, standards, specifications, or other formally imposed documents.

A comprehensive definition of *requirement* is given in the ISO/IEC/IEEE systems and software engineering vocabulary. This definition is also used in the context of this thesis:

Definition 1. ***Requirement:*** *"A requirement is*

1. *a condition or capability needed by a user to solve a problem or achieve an objective.*
2. *a condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed document*
3. *a documented representation of a condition or capability as in (1) or (2)" [51].*

Requirements are often distinguished by the RE community into functional and quality or non-functional requirements[1]. Generally speaking, functional properties determine *what* the software is able to do, whereas quality properties determine *how well* the system or software performs.

Functional Requirements:

A functional requirement specifies a function that a system or system component must be able to perform [2,46]. It is therefore a statement describing what a product must accomplish to produce required behavior and / or results [37]. "Software requirements by and large are requirements for functionality [...]

[1] Please note that quality requirements and non-functional requirements are not defined uniformly in the literature. In the context of this thesis, quality requirements and non-functional requirements are considered synonymously.

and tend to be expressed in positive terms, e.g., 'the system must. . . ' " [134]. A functional requirement must be testable, meaning that a test case result must be able to demonstrate whether a requirements is satisfied or not (pass or fail) [8].

Quality / Non-Functional Requirements:

Quality requirements are global qualities of software or a system, such as usability, reliability, security, and so forth. Quality requirements are also known as non-functional requirements (or quality attributes, qualities, or more colloquially '-ilities') that describe how well a system or software performs. As quality requirements do not only depend on software, non-functional requirements cannot be evaluated without looking at the system as a whole [8]. Traditional requirements elicitation techniques (as presented in section 2.1.5) and guidelines tend to be more focused on functional requirements than on quality requirements [134].

"[Quality] requirements are usually stated only informally, are often controversial (for example, management wants a secure system but staff desires user-friendliness), are difficult to enforce during design and implementation, and are difficult to validate. Not surprisingly, unmet quality requirements constitute an important failure factor for software development projects" [80]. While functional requirements can and should provide satisfaction or test criteria to check if a requirement has been satisfied, quality criteria typically do not have simple yes / no criteria [32, 79]. This makes it more difficult to validate non-functional requirements, or to determine if a quality requirement has been satisfied 'well enough' [79]. Non-functional requirements tend to be stated as crosscutting concerns that may impact much software functionality [8, 35]. It is argued that non-functional requirements are an abstract description of specific software or system functions and thus can be broken down into, and represented by a set of functional requirement which can be evaluated regarding satisfaction [8].

Non-functional requirements may not only concern the functional requirements but may also constrain the development process or reflect other non-functional requirement that are not directly related to the product or the development process, e.g., those resulting from external legal or normative constraints [57]. Therefore the need to separate functional and non-functional requirements is inevitable, simply due to the fact that there are non-functional security requirements that cannot be transferred into a functional requirement. An additional important aspect calling for a separation of functional and non-functional SRs originates from the different scopes of the requirements and the respective specifications (i.e., software, system, operational level). The more 'functional' a security requirement is specified in the very beginning (documented in terms of a design or implementation specification), the less flexible will be the remaining decision space for the resulting functional and non-functional requirements.

There are two widely accepted and commonly used standards for describing quality attributes, namely ISO/IEC 9126-1 [50] and its successor ISO/IEC 25010 [46]. The latter defines a quality model for software products and computer systems (in contrast to ISO/IEC 9126-1, whose scope is limited to software products) with the aim of providing consistent terminology for specifying, measuring, and evaluating system and software product quality. It complements the IEEE standard for software requirements specifications (IEEE Std. 830 [39]) and the IEEE guide for developing system requirements specifications (IEEE Std. 1233 [38]). A more comprehensive overview and discussion of definitions for the term *non-functional requirement* can be found in [17].

2.1.4 Requirements Engineering Process Model

Software and system requirements (functional and non-functional ones) are elicited, analyzed and negotiated, documented, and validated within a requirements engineering process. RE may require several iterations to result into a final software or system specification.

In the literature, process models and associated process steps for RE vary slightly. A very common (e.g., [1, 66]) and widely accepted generic process model for RE in general as well as for SRE is the coarse-grained process model, introduced by Kotonya and Sommerville [57]. The process consists of four RE process steps, namely requirements elicitation, requirements analysis and negotiation, requirements documentation and requirements validation, as depicted in Figure 2.3. The single activities or process steps are depicted as cloud icons, to indicate that each step is interleaved with the other process steps and thus there exist no distinct boundaries. The arrows indicate that, although the process is sequential in principle, there is a lot of interaction among the process steps.[2]

Since this thesis focuses on the elicitation of REs and relevant sources for it, details concerning the other RE process steps (i.e., requirements analysis and negotiation, requirements documentation and requirements validation) are omitted for the sake of readability and brevity.

2.1.5 Requirements Elicitation

"Requirements elicitation is the process of seeking, uncovering, acquiring, and elaborating requirements for computer based systems" [136]. It deals with diverse requirements sources and elicitation techniques [1]. In the literature requirements elicitation is sometimes also referred to as requirements acquisition or requirements discovery.

[2] Please note that other RE process models may also subsume requirements management under requirements engineering, add additional process step such as a 'feasibility study' prior to requirements elicitation, or combine process steps such as 'requirements elicitation and analysis', e.g., as presented in [119].

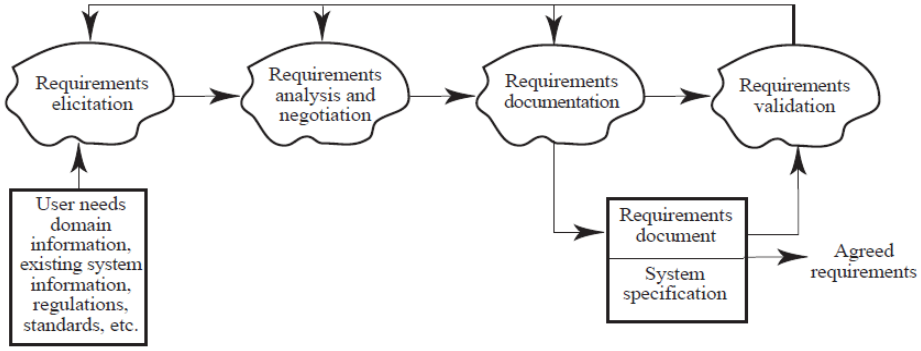


Fig. 2.3. Coarse-grained requirements engineering process [57]

Requirements Elicitation Process

As depicted in Figure 2.4, requirements elicitation consists of four main activities, namely objective setting, background knowledge acquisition, knowledge organization, and stakeholder requirements collection. The result of the requirements elicitation is a set of raw requirements[3], which is further analyzed and negotiated in the subsequent process step.

Objective setting, the first activity in the elicitation process, is intended to collect the business goals, describe the general problem to be solved, and identify other system constraints such as budget, time, or system interoperability constraints. By knowing the objectives that need to be fulfilled by the system, background information about the system can be gathered. Background information is any kind of information about the organization and its structure in which the system is to be installed, information about the application domain or other existing systems which may be replaced by the new system to be developed. This information provides the basis for the subsequent step. The third step is knowledge organization, the activity of identifying relevant stakeholders (such as customers, developers, users, administrators, etc.) and deriving stakeholder requirements as well as requirements coming from the application domain. The final activity in the requirements elicitation process is the collection of all stakeholder requirements, the application domain requirements, and other organizational requirements.

Requirements Elicitation Techniques

Requirements elicitation techniques are methods and skills required to carry out the elicitation of requirements involving the identified requirements sources.

[3] The term 'raw' requirement indicates that requirements are not further analyzed and formulated as so-called well-formed requirements.

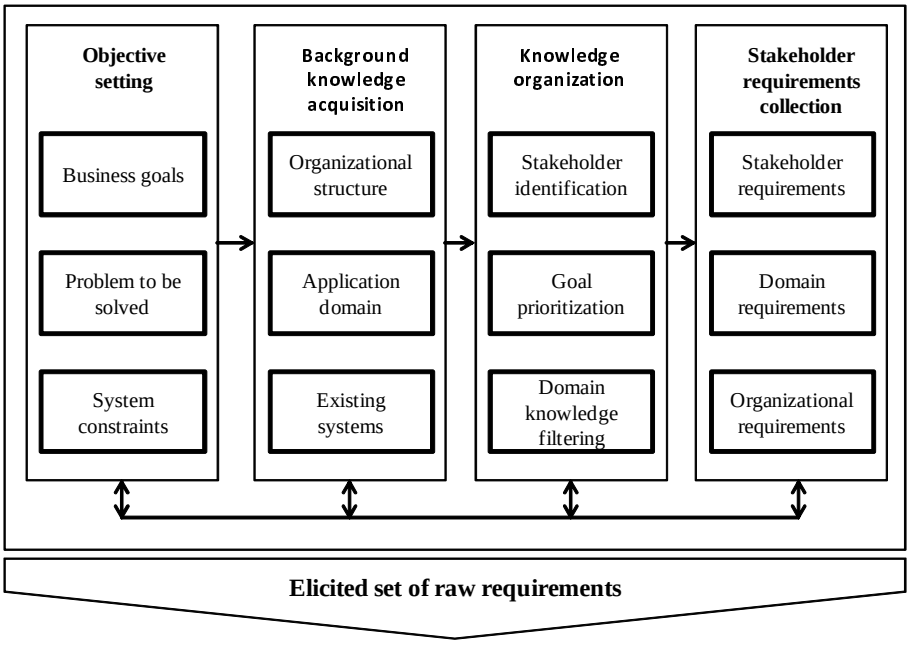


Fig. 2.4. Requirements elicitation process (based on [57])

Requirements elicitation focuses on techniques for the discovery of requirements from human stakeholders [1]. A huge number of requirement elicitation techniques exist, ranging from very intuitive techniques such as reading (i.e., visiting the library or surfing the web to locate information about experts' views of the problem or its potential solutions) [13] to complex and costly techniques such as prototyping.
The following techniques are examples of frequently referred and described ones[4]:

- **Interview:** Conducting interviews is a commonly used elicitation technique [57] and the traditional means of requirements elicitation [1]. Stakeholders are asked – preferably by an expert interviewer or requirements engineer – to name the problems to be solved and/or their needs; the aim is to build up an understanding of their requirements [13]. As interviews are seldom complete and adequate, they should be used in combination with other requirements elicitation techniques [57].

[4] Please note that the presented list of elicitation techniques is by far not complete. There exist plenty of other possible requirements elicitation techniques, such as facilitated meetings, brain-storming, research, problem domain story-boarding etc., which are not listed here. An overview of requirements elicitation methods such as Quality Function Deployment [93], Feature-oriented domain analysis [56] or Controlled Requirements Expression [124] is provided in [67]

- **Questionnaires:** A questionnaire is a document in paper or electronic form (email or web-based) containing a number of predetermined questions, which are distributed to a statistically significant and representative sample of stakeholders. Profound expertise in questionnaire formulation and statistical analysis is required [13]. The advantage of questionnaires is that a large number of stakeholders can be addressed elicitation process at relatively low costs.
- **Scenarios:** "A scenario can be defined as a description of a possible set of events that might reasonably take place. The main purpose of developing scenarios is to stimulate thinking about possible occurrences, assumptions relating these occurrences, possible opportunities and risks, and courses of action" [54]. Scenarios are real-life examples in a given context (instead of abstract descriptions of system functions) illustrating possible interactions between users and the system to be developed [57]. Use cases are the most common form of scenario-based elicitation [1, 121].
- **Prototypes:** Prototyping is a technique used to elicit and clarify unclear requirements during the early stages of the development process [13] by developing an initial version (the prototype) which represents or simulates the functionality or the final system. A prototype is a model of a system which is employed to help system designers build an information system that is intuitive and easy to manipulate for end users [98]. Prototyping can be very effective to obtain well-understood requirements if people can only express unclear or fuzzy requirements [13]. Furthermore it is used commonly to reduce the risk of developing a wrong system and to verify the usefulness and feasibility of the system before high development costs are incurred [57]. Various prototyping techniques exist, from paper mock-ups of screen designs to beta-test versions of software products [1]. Prototypes primarily used for the elicitation of requirements are 'throw-away' prototypes, in contrast to evolutionary prototypes [12, 63]. Quality assurance and non-functional aspects like performance, reliability, and security are often neglected [57].
- **Observation:** The idea behind this technique is to observe users as they carry out their normal work with the goal of understanding the users' needs from a computer-based system [57]. Observation is of particular value if user tasks and business processes are too subtle and complex for their actors to be described easily [1].
- **Requirements Reuse:** As the name of the technique already suggests, it is intended to reuse certain requirements already specified for another system with the aim of using them for the system to be developed. Although stakeholders, constraints, and conceptual aspects of the system to be developed may vary, a number of situations exist where requirements may be reused. Examples of such situations are requirements concerned with the application domain, requirements concerned with the style of presentation of the information, and requirements that reflect company policies [57]. The advantage of this technique is that it saves time and

effort while reducing the risk, as these requirements have already been analyzed and validated for other systems.

Requirements Sources

There are various potential sources mentioned in the literature on RE that may serve as inputs for the RE process; however, they are mostly very vague, which makes it hard for practitioners to specify them in a concrete, reusable way. Nevertheless, it is important to know about potential source categories where sources might originate in order to identify and evaluate them properly. A first example, incomplete set of generic requirements sources is show in Figure 2.3, in which user needs, domain information, existing system information, regulations and standards are mentioned. Furthermore, as part of the requirements elicitation process as shown in figure 2.4, an overview of potential sources for the elicitation of requirements is given. In order to develop raw requirements, the process indicates various sources such as general objectives, background knowledge, and stakeholder, domain, and organizational requirements.

The following inputs are reported in [57] as a set of generic inputs for the requirements engineering process:

- Existing system information (such as the functionality of the system(s) to be replaced or other relevant systems that interact with the specified system)
- Stakeholder needs (e.g. from users, operators, customers, market analysts, regulators and software engineers)
- Organizational standards (such as internal development practices, quality management, etc.)
- External regulations
- Domain information (knowledge about the application domain that might not be explicitly stated by stakeholders)

An alternative view on requirements sources is given in IEEE Std. 1233, the guide for developing system requirements specifications [38], as depicted in Figure 2.5.

In this standard, the two main input sources are raw requirements provided by the customer and constraints and influences coming from the system environment. A customer is defined as “the person, or persons, who pay for the product and usually (but not necessarily) decide the requirements” [38]. The environment is defined as “the circumstances, objects, and conditions that will influence the completed system; they include political, market, cultural, organizational, and physical influences as well as standards and policies that govern what the system must do or how it must do it” [38]. The output of the collected requirements, the system specification, is developed and provided

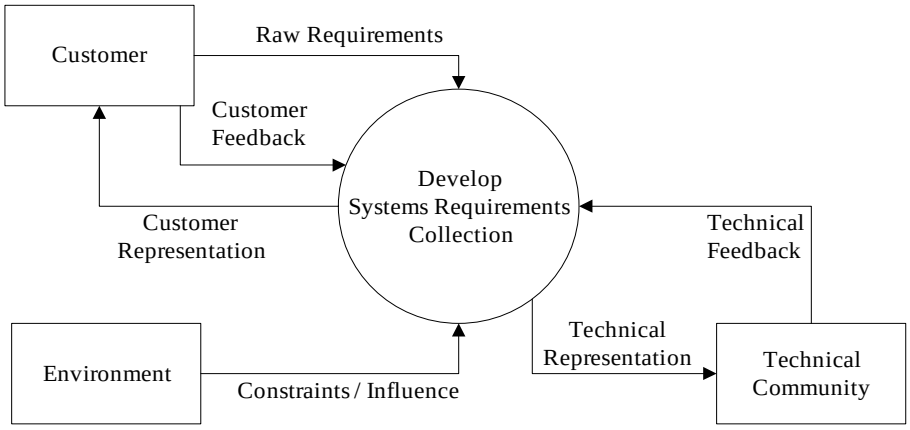


Fig. 2.5. Context for developing a requirements specification (based on [38])

to the customer as a customer representation, in order to clarify the requirements and comply to them. Additionally, a technical representation (such as a picture, drawing, or conceptual image) is provided to the technical community, which provides further technical feedback. The technical community includes various roles such as analysts, estimators, designers, quality assurance officers, certifiers, developers, engineers, integrators, testers, maintainers, and manufacturers.[5]

2.2 Introduction to Security Requirements Engineering

Security requirements (as part of the total set of requirements) are elicited, analyzed and negotiated, documented and validated within the RE process. Just like all other software and system requirements, SR elicitation and analysis may require several iterations to arrive at the final security-specific software and system specification. However, there are differences between 'traditional' and security-specific requirements engineering methods and techniques, which will be explained in the following.

2.2.1 Security and Security Requirements Engineering

Definition of Security

Security is the "condition that results from the establishment and maintenance of protective measures that enable an enterprise to perform its mission

[5] Please note that the customer- and technical representation is not part of requirements elicitation, but already part of requirements analysis and negotiation.

or critical functions despite risks posed by threats to its use of information systems" [82]. The aim is to protect information and information systems "from unauthorized access, use, disclosure, disruption, modification, or destruction" [82].

The security community defines three security core principles [7] (also referred to as "basic requirements" [128], "general security concerns" [32] or "security categories" [83]) that must be preserved: Confidentiality, integrity and availability. These core principles are sometimes enhanced by other properties such as accountability, authenticity, non-repudiation, and reliability.

Definition 2. ***Security:*** *Security is the preservation of confidentiality, integrity and availability of assets. (Similar to the definition given in [48])*

The security core principles are defined as follows (based on [42]):

Definition 3. ***Confidentiality:*** *The property that information is not made available or disclosed to unauthorized individuals, entities, or processes.*

Definition 4. ***Integrity:*** *The property of safeguarding the accuracy and completeness of assets.*

Definition 5. ***Availability:*** *The property of being accessible and usable upon demand by an authorized entity.*

Definition of Security Requirement (SR)

There is no common agreement on what a SR is [18]. Moreover, existing approaches do not agree on the extent to which the requirements should state concrete security measures [129]. This induces uncertainty among RE practitioners about good practice for SR specifications and also leads to heterogeneous sets of SRs ranging from a small set of high-level security goals to extensive lists of functional and non-functional security requirements or even concrete security measures. Several definitions are found in the literature that reflect different perspectives on security. Some definitions stem from the software or system (requirements) engineering community, while others originate from the security domain as briefly sketched in the following.

Security Requirements from a Requirements Engineering Perspective:

According to the Systems and software Quality Requirements and Evaluation (SQuaRE) process [46], security is a quality (or non-functional) requirement characteristic, containing the sub-characteristics confidentiality, integrity, non-repudiation, accountability, and authenticity. Correspondingly, Haley et al. define security requirements as constraints on the system's functional requirements (rather than being themselves functional requirements). "Security requirements express the system's security goals in operational

terms, precise enough to be given to a designer/architect. Security requirements, like functional requirements, are prescriptive, providing a specification (behavior in terms of phenomena) to achieve the desired effect" [32]. This definition also underpins the general requirements engineering principle, namely that requirements should not specify architectural constraints or implementation mechanisms (i.e., 'the how') but rather provide statements describing what the system must do (or, with regard to security also describing what it must not do). Wilander and Gustavsson state that SRs are both functional and non-functional. "In the functional case they represent abstract security features broken down into concrete functional requirements. In the non-functional case they are either restrictions on design and implementation, or requirements on assurance measures such as security testing" [133]. Kotonya and Sommerville argue similarly, stating that there is no clear distinction between functional and non-functional requirements, although "it is often helpful to distinguish requirements which specify system's functionality from non-functional requirements" [57].

Security Requirements from a Security Engineering Perspective:

Differentiation between functional and non-functional requirement is not common in current international security standards and publications such as the ISO/IEC 2700x series (e.g., [47–49]), Common Criteria [43–45], or the NIST Security and Privacy Controls for Federal Information Systems and Organizations [83]. A common definition for SRs in the security domain is: "Security requirements are those requirements levied on an information system that are derived from laws, Executive Orders, directives, policies, instructions, regulations, standards, guidelines, or organizational (mission) needs to ensure the confidentiality, integrity, and availability of the information being processed, stored, or transmitted" [82].

"Security is a chain; it's only as secure as the weakest link. Security is a process, not a product" [107]. A system consists of much more than just software. It is developed, commissioned, used, and maintained by various persons that depend on the system on the one hand, but on the other hand may also (deliberately or negligently) put the system at risk. Furthermore, there are various types of attackers that deliberately try to attack the system. A system can only be considered secure (and even this is hard or impossible to predict or claim) if a holistic security concept exists, providing a set of SRs that again serve as a basis for and adequate design and implementation of protective measures. The concept of threat- and risk-oriented security engineering is depicted in Figure 2.6.

Definition 6. ***Threat:*** *A threat is "any circumstance or event with the potential to adversely impact organizational operations [...], organizational assets [...] through an information system via unauthorized access, destruction, disclosure, modification of information, and/or denial of service" [83].*

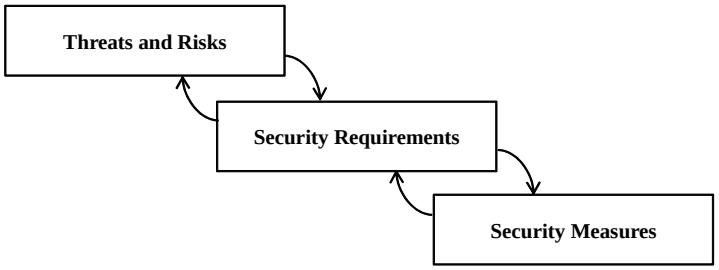


Fig. 2.6. Security Engineering Approach (based on [78])

Definition 7. ***Risk:*** *Information-system-specific risks are "those risks that arise from the loss of confidentiality, integrity, or availability of information or information systems and consider the adverse impacts to organizational operations (including mission, functions, image, or reputation), organizational assets, individuals, other organizations, and the Nation" [83].*

Definition 8. ***Security Measure:*** *Security measures*[6] *are protective measures prescribed to meet the security requirements and to fulfill the security core principles.*

Security measures are not only limited to software or technical measures, but also cover physical and organizational aspects, e.g., as stated in [48] or [83]. Some of the security measures have to be implemented in software, while others have to be ensured by the customer by means of adequate security measures in the operational environment. In the end, an adequate security level has to be ensured, meaning that an acceptable level of risk can be reached. As a simple example illustrating this circumstance, the best possible user name and password-based authentication mechanism (i.e., technical measure) may not be sufficient if users or administrators do not change their initial default passwords or pin their current credentials to the top of their office desktop screens. Without additional organizational measures to ensure sufficient awareness amongst users and administrators, the technical security measure would not provide sufficient (overall) security. Another example is a piece of software using (and relying) on the certificate store of the underlying server. Although the software itself might provide good security mechanisms, the server may be vulnerable, e.g. due to a lack of firewall protection, missing operating system hardening, insufficient malware protection, or use of default administrator passwords. The overall security may therefore be compromised by an attacker due to weaknesses and vulnerabilities in the environment of the software.

In addition, the following IT security terms and definitions are used throughout this thesis.

[6] Alternative terms for security measure are safeguard and security control.

Definition 9. ***Asset:*** *An asset is "anything that has value to the organization" [42].*

Examples of assets include "a major application, general support system, high impact program, physical plant, mission critical system, personnel, equipment, or a logically related group of systems" [82]. Typically, parts of an application or system, such as information/data, functions, as well as hardware or software are also considered assets.

Definition 10. ***Weakness:*** *"Weaknesses are flaws, faults, bugs, vulnerabilities, and other errors in (software) implementation, code, design, or architecture that if left unaddressed could result in systems and networks being vulnerable to attack" [127].*

Definition 11. ***Vulnerability:*** *A vulnerability is a weakness in an information system, system security procedures, internal controls, or implementation that can be directly exploited or triggered by an attacker (similar to definition provided in [83]).*

Based on the given generic definitions of the term *requirement* related to the REs and Security domain, for this thesis a *Security Requirement* is defined as follows:

Definition 12. ***Security Requirement:*** *"A security requirement is*

1. *a condition or capability needed to ensure the confidentiality, integrity and availability of a software or system in order to protect valuable assets.*
2. *a condition or capability imposed by security requirements sources that must be met or possessed by software or a system.*
3. *a documented representation of a condition or capability as in (1) or (2)".*

Please note that the definition does not prescribe the level of detail or the extent to which SRs have an impact on security measures. As far as this work is concerned, it is at the discretion of the development teams to decide which level of detail is adequate for a particular project. However, as discussed in this chapter, it is important that a sufficient level of detail concerning the problem space is reflected in the requirements, or that it is at least possible to trace a requirement to its originating source, to ensure that software and system architects do not only receive isolated, stand-alone generic requirements, but know about the originating source of a security requirement, regardless of whether it was identified via security information and knowledge sources, compliance obligations, or SRE methods.

Trust Assumptions

Some academic work in the area of SRE (cf. section 2.5.4) explicitly considers trust assumption in addition to SRs. A trust assumption is "the firm belief in the competence of an entity to act dependably, securely and reliably within

a specified context" [30]. "Trust assumptions can be used by a requirements engineer to help define and limit the scope of analysis and to document the decisions made during the process" [33]. "The level of trust has an approximate inverse relationship to the degree of risk" [30], which means that if everything and everyone is trusted, no security threats and risks need to be incorporated, and therefore no security requirements need to be specified to address the threats and risks (assuming that no other external constraints or influences demand security requirements). Unlike software-specific SRs, assumptions cannot be enforced by the software-to-be [58]; however, they should be documented to ensure that organizations deploying the software in a particular environment know them and can develop environment-specific security requirements and measures to fulfill the trust assumptions. Unfortunately, trust assumption or trust relations are seldom defined explicitly [132] since most related work focusing on SRs does not put much emphasis on trust [33]. Therefore, if they are specified separately, trust assumptions are counterparts of SRs and should be documented along with SRs. For exemplary SRE methods incorporating trust assumptions, please refer to section 2.5.4.

Please note that throughout this thesis, trust assumptions will not be mentioned along with SRs as possible output. Nevertheless, the proposed model and SRSs are also applicable for trust assumptions, of course, since the identification and structuring of SRSs and the development of an organization-specific SRS repository can be used for the specification of both SRs and trust assumptions. Moreover, trust assumptions can be considered a specific type of SR that must be fulfilled.

2.2.2 Consequences of Poor Security Requirements Engineering

Since SRs are part of the overall requirements specifications, the general negative impact of poorly developed requirements as described in section 2.1.2 also holds true for SRE. Therefore, shortcomings in the SRE phase can also have a highly negative impact on the overall project success as well as on the schedule, budget, and security of the developed software or system, just like any other type of requirement. In addition to these project-specific aspects, SRs also have an additional, more long-lasting influence, which makes proper SRE particularly important. According to [15], most vulnerabilities and weaknesses in software systems originate from inadequate or incomplete SRs. Although the actual financial damage is hard to quantify, security flaws cause a negative financial impact of several billion USD a year [31]. Thus, potential threats, weaknesses, and vulnerabilities that were not identified and addressed properly during the RE phase may lead to severe negative operational consequences during the future operation of a software or system.

2.2.3 Differences between 'Traditional' and Security Requirements Elicitation

'Traditional' requirements elicitation techniques (as presented in section 2.1.5) are primarily designed to seek, acquire, and elaborate requirements, mostly from human stakeholders. Therefore, this kind of RE is mostly done with the aim of identifying what the software or system *must do* in order to fulfill customer needs and solve practical problems.

In contrast, security is concerned with identifying and preventing the bad things that could happen, but that must not happen. The objective is to prevent harm from assets and to reduce the risk for an organization in order to ensure confidentiality, integrity, and availability (see section 2.2.1). Risks are introduced by threats, weaknesses, and vulnerabilities that might be exploited by an attacker. Therefore, the security community and the SRE community primarily require proper information and knowledge, both methods and processes, to be able to define what *must not* happen, which means in practical terms that potential threats, weaknesses and vulnerabilities need to be identified and analyzed. These are translated into security requirements, which in turn are the basis for defining proper security measures to counteract the identified threats, weaknesses, and vulnerabilities and thus to protect the valuable assets.

2.2.4 Security Requirements Sources

Unfortunately, only little information about the main sources for SRE is available and discussed in the literature. Although some examples of potential requirements sources in the area of IT security do exist [14, 15, 134], no structured list, big picture, or overview of security requirements sources (SRS) has been published to date.

According to ISO/IEC 27001, there are three main sources of security requirements. The first source results from threat and risk analysis, the second source are legal, statutory, regulatory, and contractual requirements, and the third source is a "set of principles, objectives and business requirements for information processing that an organization has developed to support its operations" [47].

2.3 Customer and Environment

The first source category that serves as input for the SRE process are raw requirements, constraints, and influences coming from the customer or the system environment, according to the 'IEEE Guide for Developing System Requirements Specifications' [38]. In the following, all internal and external raw requirements, influences, and constraints that are relevant for security are subsumed under the category Customer and Environment. They are derived

from analyzing the 'IEEE Recommended Practice for Software Requirements Specifications' [39] and the 'IEEE Guide for Developing System Requirements Specifications' [38] concerning their relevance for security and for reuse. Moreover, they are extended with additional security-specific raw requirements, influences, and constraints from the security community[7].

2.3.1 Criticality of Information, Product or System

It is common practice in the information and IT security community to classify information and information systems in terms of their "value, legal requirements, sensitivity and criticality to the organization" [47] before SRs, controls, or measures are selected or developed. This classification (sometimes also referred to as categorization) is typically derived from the application of a (business) impact classification method in which the classification results for the security objectives confidentiality, integrity, and availability are estimated. The classification result often serves as input for the selection of a set of recommended or obligatory requirements or controls that need to be fulfilled in order to protect information, information systems, or products. A popular example is the combination of the NIST standards 'Security Categorization of Federal Information and Information Systems' [21] with the associated 'Security and Privacy Controls for Federal Information Systems and Organizations' [83]. Similar approaches (consisting of a classification and a subsequent requirements or control selection step) can also be found in many companies for different scopes such as enterprise IT systems, products, and solutions.

2.3.2 Intended Operational Environment

The intended operational environment has an important influence on the development of a product or solution. As already explained in section 2.2.1, software is set up, operated, maintained, and used by various people who depend on the system on the one hand, but on the other hand may also put the system at risk. Furthermore, there are various types of attackers that deliberately want to attack the system. Security measures are not only limited to software or technical measures, but also cover physical and organizational aspects, e.g. as stated in [48] or [83]. Some of the security measures have to be implemented in the software, while others have to be ensured by the customer by means of adequate security measures in the operational environment. Schmitt and Liggesmeyer present in [101] a dependency model in which security aspects, constraints, and influences in the operational environment

[7] Please note that there are various other lists of customer, stakeholder and environmental requirements sources (e.g. [14, 57]), which partly use different terminologies. However, for the purpose of this thesis, the focus on the aspects of (security) requirements sources presented in the following is considered to be sufficient.

are grouped into three scope areas, namely software, system in its technical environment, and system in the organizational environment. All three scope areas mutually depend on and influence each other. They conclude that the less predictable the potential operational environment of a software product is, the more difficult it becomes to specify valid trust assumptions and proper security requirements, since constraints and influences are unclear or may vary significantly. This lack of concrete constraints and influences of the operational environment on software or systems calls for a greater degree of variability and quality of security mechanisms in order to fit different fields of application.

2.3.3 Compliance Obligations

For enterprise systems, as well as for products and solutions, various potentially applicable compliance obligations exist that 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, have to be incorporated into the requirements engineering process. For these kinds of requirements, typically advice from legal advisers or suitably qualified legal practitioners should be incorporated to cover and properly interpret the huge range of obligations. Examples range from regional and country-specific legislation (e.g., data privacy laws) to domain-specific obligations (such as HIPAA and FDA Part 11 in the healthcare area, or IEC 62443 [41] for industrial automation and control systems) or even export control related aspects that have an impact on SRs or even on security measures. Furthermore, there can also be accreditation, certification or auditing needs which need to be incorporated into the RE process. Besides external compliance obligations, often company- or organization-internal compliance obligations may also need to be followed, such as information and IT security policies, standards, and guidelines. In practice, compliance obligations are very important for an organization, since non-compliance and the resulting negative consequences may lead to a highly negative business impact, e.g. due to delays, or may even result in non-certification of a product or solution. Depending on the respective laws, policies, standards, etc., various requirements engineering aspects can vary strongly (e.g., the terminology used and the extent to which the 'raw requirements' are specified). For instance, raw requirements that need to be fulfilled may be named differently, e.g., 'security controls' [48, 83]; 'Security Functional Requirements (SFRs) and Security Assurance Requirements (SARs)' [43], or 'Foundational Requirements (FRs)' [36]. Some of them provide rather high-level goals and security principles, while others primarily address the solution space (which 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 the SRE process.

Practical experience shows that if there are any formal security compliance obligations that need to be fulfilled, and if these obligations are provided as list of relevant requirements or controls, the effort for SRE is primarily spent on the fulfillment of the compliance obligation, rather than on the application of SRE processes and methods in a 'green-field approach'. In such a scenario SRs are seldom developed 'from scratch'.

2.3.4 Domain Specifics

Domain specifics are characteristics or restrictions that need to be considered specifically for a certain domain (e.g., healthcare or process automation). For some domains, there are domain-specific standards which need to be incorporated. A popular example is ISA99/IEC62443 [41] as a series of standards, technical reports, and related information that define procedures for implementing electronically secure industrial automation and control systems. Moreover, there may be domain-specific restrictions, which are the reason why security cannot be dealt with in the same way in systems from different domains. As a practical example, a typical recommendation for enterprise IT systems is to frequently and timely patch a system with the latest security patches. For an industrial product or solution, this is typically only possible during predefined maintenance periods, or patching might not be foreseen at all since it might lead to a potential loss of accreditation, e.g., in the case of safety-critical systems.

2.3.5 Market Influence

The market influences how much a company is willing to invest into security for their products and solutions. If the market does not demand security, e.g. because of missing legal and regulatory compliance obligations or a lack of security awareness on the part of the stakeholders, security will not be seen as added value and thus not as a selling proposition. In such a case, typically only the minimum will be invested into security since customers are not willing to pay for a higher level of security, a certified product, or security features that are not needed. However, if customers are aware of or demand a certain level of security, a company may increase its reputation, gain a competitive edge, and consequently increase its market share with proper security. Therefore, it is important to investigate which level of security is demanded by the customers, which security functionality is required as a minimum, and what the state of the art is. In this regard, it is also important to know the level of security of competitors' products and solutions and which security features or functionality they offer.

2.3.6 Non-Security Needs Influencing Security Requirements

Besides the above-mentioned requirements sources and influencing factors, there are various other non-security constraints, influences, functional and

non-functional requirements that can have a direct or indirect influence on SRs. Various other product quality requirements such as performance efficiency, usability, and maintainability [46] may influence or even be in conflict with SRs. As an example, a SR concerning the protection of confidentiality and the integrity of transmitted data may be in conflict with requirements concerning the efficient transmission of data with as little latency as possible. These potential conflicts among requirements demand proper integration of SRE into the overall RE process and, in particular, mechanisms for the proper analysis and identification of trade-offs between requirements of different kinds.

2.3.7 Conclusions concerning Customer and Environment

Which aspects of the source category Customer and Environment need to be considered cannot be generalized but needs to be clarified and elaborated for each development project. Market influences and non-security needs influencing security should be addressed by the overall RE process, as they are not only specific to security. The criticality of the information, of a product or a system is typically incorporated into most SRE processes (e.g., [6,69,73,115]) and methods through the identification and documentation of critical / security-sensitive assets as well as the specification of security goals and objectives as a basis for later process or method steps. The incorporation of domain specifics and the operational environment is typically done through the development of artifacts such as system overview, data flow diagrams, or problem frames (cf. section 2.5). Compliance as one of the most important drivers for security is a crucial aspect for many organizations. Lots of internal and external compliance obligations exist that may need to be fulfilled. Therefore, SRE processes should incorporate raw requirements from compliance obligations in addition to eliciting SRs only via SRE methods.

In practice, compliance obligations are very important for an organizations as non-compliance may have a highly negative business impact due to delays or may even result in non-certification of a product or solution. However, compliance obligations are underrepresented in most of the published SRE processes and frameworks (e.g., [6, 69, 73, 115]). Although most of them propose the use of certain SRE methods for requirements elicitation, they do not explicitly foresee the incorporation of other requirements sources such as raw SRs from compliance obligations. Only Mellado et al. [73] recommend the inclusion of legal, statutory, regulatory, and contractual requirements. However, they leave open how this should be done in practice.

2.4 Security Information and Knowledge

The second source category for the development of SRs is Security Information and Knowledge. The lack of security knowledge and skills is considered one

of the main challenges in SRE. Traditional RE techniques are rather focused on functional requirements than on SRs, and "most requirements engineers are poorly trained to elicit, analyze, and specify security requirements" [23]. Various methods exist to identify and analyze the threat, attacker, and misuse perspective. However most of these methods require a certain level of knowledge regarding security and SRE. The quality of the results will improve if security requirements engineers and architects know about the capabilities of the attackers, about attacker tools available on the market, and about typical threats, weaknesses and vulnerabilities. If one does not understand the mindset of an attacker, typical weaknesses and available exploits, the results of a threat analysis are likely to be of limited value. Security information and knowledge is very multifaceted and made available in various ways for different purposes. In this thesis, it is subdivided into three sub-categories: diagnostic, prescriptive and historic, following the knowledge base structure proposed by Barnum and McGraw [3].

2.4.1 Diagnostic Security Knowledge and Information 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 SRs as a basis for the design of security measures. It is therefore a prerequisite for many methods such as threat and risk analysis since it forms the basis for the identification and analysis of threats as well as for the evaluation of the resulting risks. Hence, diagnostic security information and knowledge can directly provide threats, weaknesses, and vulnerabilities as input (e.g., in the form of structured lists) to the SRE process for the elicitation of SRs. Alternatively, threats, weaknesses, and vulnerabilities can also be derived via attack patterns or knowledge about exploits and hacker tools that might be used by an attacker to exploit potentially existing threats, weaknesses, or vulnerabilities.

Examples of diagnostic information and knowledge sources and their elements are:

- **Security threats:** Security threats are often provided as lists of (mostly generic) threats in risk assessment guides [81], risk analysis methods [40] and risk management standards [49].
  Example security threats are "obtain sensitive information through network sniffing of external networks" [81] and "abuse of rights" [49].
- **Security weaknesses and vulnerabilities:** Weaknesses and vulnerabilities are mostly provided and maintained in online catalogs or community developed dictionaries such as the Common Weakness Enumeration (CWE) [127] and the Common Vulnerabilities and Exposures (CVE) [126]. An example of a security weakness is "CWE-287: Improper Authen-

tication" [127] and an example of a well-known security vulnerability for 'OpenSSL Heartbleed' is CVE-2014-0160 [126].
- **Attack patterns:** e.g., [111, 125]. An example of an attack pattern is "CAPEC-242: Code Injection" [125].
- **Knowledge about exploits and hacker tools:** This refers to the knowledge about exploitable vulnerabilities and corresponding attacker tools, e.g., the metasploit framework.

2.4.2 Prescriptive Security Knowledge and Information 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) via guidelines for various security topics to rather concrete security controls (e.g., certificate-based user identification) and specific security design patterns. It can be assigned in many cases to the solution space, since it mostly prescribes *the how* in contrast to *the why* and *what*. Therefore, prescriptive security information and knowledge is typically input for the design phase in a development life-cycle and not necessarily a primary source for the SRE process. The reason why prescriptive security information and knowledge is nevertheless referenced here as input for SRE is due to the perception: Existing architectures may partly or fully influence the way problems are structured. It may be useful or necessary to reverse-engineer problems for which a known solution exists from existing architectural designs [53, 86]. Specification and implementation are often intertwined in practice [122] since limitations of available implementation technology may demand a specification change or implementation choices require the augmentation of the original specification [122]. This perception was adapted in the context of requirements and architecture in the twin peak model [86], a simplified version of the spiral model. The model foresees a concurrent, spiral development process, 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 RE and architectural design (cf. section 3.1.2). Therefore, prescriptive information and knowledge may provide a valuable input for the SRE process.

Examples of prescriptive information and knowledge sources are:

- **Security (design) principles** [88, 89]. Examples of security design principles are 'apply defense in depth', 'fail securely', or 'don't trust infrastructure'.
- **Security guidelines** [84]. Examples for guidelines are 'Guidelines on Security and Privacy in Public Cloud Computing' or company-specific guidelines for security architecture design.

- **Security (design) patterns** [109].
- **Security control lists** [83, 97]. An example control from [83] is "IA-3 Device Identification and Authentication - The information system uniquely identifies and authenticates [Assignment: organization-defined specific and/or types of devices] before establishing a [Selection (one or more): local; remote; network] connection."

Prescriptive security information and knowledge is seldom provided together with information about the problem space that it addresses or with diagnostic security information and knowledge, respectively. The reason or motivation for prescriptive information and knowledge sources is therefore often not transparent to the user community.

2.4.3 Historical Security Information and Knowledge Sources

Historical information and knowledge describes security incidents or vulnerabilities that were uncovered in real-world products or solutions. The goal of historical information is to learn from previous incidents or vulnerabilities in order to avoid making the same mistakes again. Therefore, they provide valuable input for aspects of diagnostic and prescriptive knowledge.

2.4.4 Dependency between Diagnostic and Prescriptive Security Information and Knowledge Sources

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

2.4.5 Conclusions Concerning Security Information and Knowledge Sources

The knowledge, skills and mindset required for SRE is different from 'traditional' requirements engineering. It is more difficult to define what a system should not do or to identify the threats to it than defining what it should do [134]. 'Traditional' RE techniques are usually focused on functional requirements rather than on SRs. Moreover, "most requirements engineers are poorly trained to elicit, analyze, and specify security requirements" [23]. Thus, for requirements engineering teams without security expertise, it is important that

security knowledge and information is provided in a structured, understandable, and reusable way, so that it can be incorporated when interpreting compliance obligations, applying SRE methods, and specifying security requirements. Thus, to do proper SRE, security knowledge and experience regarding the problem space (i.e., threats, weaknesses and vulnerabilities) is required in order to identify and analyze the bad things that should be prevented and to specify adequate SRs. Information about the problem space is mostly covered in diagnostic information and knowledge sources, making this category of particular interest for SRE. In addition to diagnostic sources, prescriptive security information and knowledge sources such as generic SRs, security principles, security guidelines, security control lists, and security patterns can also be very useful for SRE. Although prescriptive information mostly addresses the solution space, it can be helpful to support the alignment between the problem and the solution space, where requirements engineers and system architects work concurrently, and iteratively increase the level of detail of the requirements specification and the architecture design. Diagnostic and prescriptive information and knowledge sources have a strong correlation since the diagnostic sources provide the basis for understanding and motivating the prescriptive information. Therefore, if security information and knowledge sources are provided, it is highly desirable to combine diagnostic and prescriptive information and knowledge sources.[8]

Elahi et al. conclude in their survey on SRE [29] that security knowledge sources are seldom used in practice. Moreover, diagnostic and prescriptive information and knowledge sources are rarely provided together in a stringent and traceable fashion.

2.5 SRE Methods

Traditional RE techniques such as interviews, questionnaires, scenarios, prototypes, or observations are primarily designed for the discovery of ordinary functional requirements from human stakeholders [1] which are typically not oriented towards security requirements and therefore will not result in a complete and consistent set of security requirements [68]. The need for additional security-specific RE methods and techniques results primarily from the fact that one must first know the bad things that may happen to a system and what it must not do before being able to define which SRs and measures shall avoid or prevent these. To achieve this, particular attention is paid in the area of security to the identification and analysis of threats and resulting risks in order to protect the system assets by specifying adequate SRs and measures (cf. section 2.2.1).

[8] Since historical information serves as input to diagnostic and prescriptive security information and knowledge, it will not be treated as a separate source for the proposed model since it is assumed that it is already covered implicitly.

Various methods are mentioned and referenced in the literature in the context of SRE. A more comprehensive overview and comparison of SRE methods can be found in [20, 72].

In this thesis, SRE methods are grouped into *analysis*-oriented, *modeling*-oriented, and *reuse*-oriented methods. Furthermore, approaches incorporating *trust* aspects are presented.[9] Methods that are primarily designed to reveal threats, weaknesses, vulnerabilities and attacks are assigned to analysis-oriented methods. Methods that place a lot of emphasis on modeling and model-related aspects are assigned to modeling-oriented methods. Methods and approaches that provide techniques or utilize results for security requirements reuse are mentioned in reuse-oriented approaches. Finally, a short overview of methods incorporating trust aspects will be given.

2.5.1 Analysis-oriented Methods

Analysis-oriented methods are designed to analyze the problem space systematically in order to identify threats, weaknesses, vulnerabilities and attacks. The result is intended to be used as a basis for the specification of SRs.

Prominent methods and techniques are (among many others):

- Abuse cases [64]
- Misuse cases [116]
- Attack trees [106]
- Goal-oriented Requirements Engineering, Anti Goals and Anti Models [58–60]
- Anti Requirements and Abuse Frames [10, 61]
- Threat and risk analysis methods e.g., STRIDE [123] or company-specific threat and risk analysis frameworks [105].

2.5.2 Modeling-oriented Methods

Another research direction in SRE are methods for improving the capabilities for modeling SRs or security-specific information. TROPOS [76] and Secure TROPOS [75] as extensions of the i* modeling language, as well as STS-ml [91] are common examples. Furthermore, two extensions of the Unified Modeling Language (UML), namely UMLSec [55] and Secure UML [62] were introduced as possible ways to improve the integration of security-specific information into UML. Golnaz et al. present an approach on modeling and analyzing security trade-offs (e.g., [27, 28]).

[9] Please note that the structure is not always clear-cut since analysis-oriented methods, for instance, also consider modeling aspects, of course. The structure is therefore not disjoint but rather intended to provide a coarse-grained classification for the sake of readability.

2.5.3 Reuse-oriented and Pattern-based Approaches

The lack of skills and knowledge for security and security requirements engineering is a common problem and challenge for SRE in development teams (cf. also sections 1.2 and 2.4). One possible way to overcome skill- and knowledge-related shortcomings and furthermore decrease the effort for applying methods from scratch is to reuse the results or artifacts of SRE methods. As examples, attack trees [106] and misuse cases [117] can be used as methods for capturing and reusing security information and expertise. To standardize and improve the quality of reusable artifacts, several authors propose the use of templates for documenting the results of a method, e.g. for misuse cases [115] and security use cases [24, 115]. Consequently, misuse cases and security use cases are used in existing SRE processes and approaches (e.g., [73, 115]) to develop a reusable set of threats and security requirements. Firesmith claims that requirements can be specified as a limited set of quality factors (and related sub-factors) and presents examples of a reusable, parametrized template for SRs [25]. Similarly, SR patterns and boilerplates[10] also provide templates for specifying SRs, but additionally also describe how and when to reuse these requirements (templates) [94]. SR patterns therefore provide a reusable set of requirements to solve common security problems in combination with the associated contextual information. As an example, Withall [135] presents security-related patterns for access control (registration, authentication, authorization), audit, and privacy aspects among many other requirements patterns. Daramola et al. [11] present a pattern-based approach and related prototype tool that supports the specification of security requirements utilizing textual misuse cases (TMUC).
To increase the efficiency of pattern-based SR approaches, i.e., to review only relevant patterns and select the most appropriate ones for a specific situation, Slavin et al. [118] proposed an inquiry-cycle-based approach combined with a feature diagram notation.

Another way to reuse security information and knowledge, which was proposed by several authors, is to create a reusable requirements repository or knowledge base. In SIREN [130], the requirements repository is filled with countermeasures taken from Methodology for Information Systems Risk Analysis and Management (MAGERIT) that were translated into security requirements. Mellado et al. [73] propose storing and reusing elements from Common Criteria. Dikanski and Abeck [16] propose the creation of reusable SR analysis templates (SecRAT) in order to develop and use a knowledge base, offering various pieces of relevant information such as security standards, technologies, security models, principles and policies, which can be reused for SRE.

[10] i.e., a textual template for requirements specification that is based on predefined patterns aimed at reducing ambiguity and ensuring consistency regarding the expression of requirements [94]

2.5.4 Approaches incorporating Trust Assumptions

Missing consideration or documentation of (trust) assumptions is a challenge in SRE. Grandison and Sloman state that there is a need for a more formalized approach to trust establishment, evaluation, and analysis in order to support trust management [30]. "If developers overestimate or misjudge the trustworthiness of other components in the system, the deployment environment, or peering organizations, then the underlying security architecture may be inadequate" [131]. Haley and colleagues did a lot of research work on the relation between trust assumptions and SRs [33]. In [34], they present an approach to use trust assumptions for reasoning about security requirements by using problem frames [53] and threat descriptions [35].

2.5.5 State of the Practice concerning SRE Methods

To identify and analyze the threat, attacker, and misuse perspective, various methods exist on the market, such as Microsoft's STRIDE [123], Abuse Cases [64], and many others (for details, see section 2.5). However, most of these methods require security knowledge and method-specific know-how. In the literature on SRE, it is often stated that the application of SRE methods and techniques in projects and the maturity of SR specifications are mostly poor (e.g., [96, 133]). Of the various methods developed for SRE just a small number have been used in practice to date [96].

2.6 SRE Processes and Frameworks

The practical use of SRE processes, frameworks, and methods is considered to be poor in most cases. Security is often integrated into the development life-cycle too late, or not at all [18, 29, 77]. This typically leads to security being insufficiently considered or not being considered at all in software and system development projects. Another problem is that SRE can only very rarely be done completely up-front before the design phase starts. "Most of the work for traditional requirements engineering fails to explicitly consider security" [15]. Wilander and Gustavsson conclude in their field study on security requirements engineering [133] that "current practice in security requirements is poor". Salini et al. [96] also arrive at the same conclusion. Mead states in [68] that "when SRs are considered at all during the system life-cycle, they tend to be general lists of security features such as password protection, firewalls, virus detection tools, and the like". Although numerous publications state that SRE is important, only little concrete and specific advice is provided that can be used immediately in projects [129]. SRE approaches are mainly tested on a small scale, requiring broader external validation to demonstrate their effectiveness and usefulness for the development of security solutions [18]. There is a demand among security practitioners that SRE methods must be primarily

understandable and usable, besides being "formally pleasing or academically correct" [129].

To incorporate the particularities of SRE, the following process models have been elaborated:

2.6.1 Security Quality Requirements Engineering (SQUARE)

Probably the best-known process framework for security requirements engineering is the SQUARE methodology [70] developed by Nancy Mead et al. which is designed to support the elicitation and prioritization of SRs in software development projects.

(a) **Main concept / idea:**
- Proposal of a generic process to be followed when performing SRE
- Provision of additional details as well as information on input, helpful techniques, participants, and output for each process step

(b) **Structure elements / model:**
- n.a.

(c) **Reuse aspect:**
- n.a.

(d) **Considered standards / frameworks:**
- n.a.

(e) **Process model / steps:**
The proposed process consists of the following nine steps:
a) Agree on definitions
b) Identify security goals
c) Develop artifacts to support security requirements definition
d) Perform risk assessment
e) Select elicitation techniques
f) Elicit security requirements
g) Categorize requirements as to level (system, software, etc.) and whether they are requirements or other kinds of constraints
h) Prioritize requirements
i) Requirements inspection

2.6.2 Extending XP Practices to Support Security Requirements Engineering

Gustav Boström et al. developed a seven-step process for security requirements engineering that is primarily intended for agile development, particularly concerning eXtreme Programming (XP) practices [6]. The process model enhances XP user stories in order to include SRs.

- **Main concept / idea:**
  - Process for agile developments to be followed
  - Development of abuser stories / threat scenarios

  - Conduction of risk assessments
- **Structure elements / model:**
  - n.a.
- **Reuse aspect:**
  - n.a.
- **Considered standards / frameworks:**
  - n.a.
- **Process model / steps:**
  The proposed process consists of the following seven steps:
  1. Identification of security sensitive assets
  2. Formulation of abuser stories (threat scenarios)
  3. Abuser story risk assessment
  4. Abuser story and user story negotiation
  5. Definition of security-related user stories
  6. Definition of security-related coding standards
  7. Abuser story – countermeasure cross-checking

2.6.3 A Reuse-Based Approach to Determining Security Requirements

Sindre et al. propose a reuse-based approach for application development that involves identifying security threats with misuse cases and associated SRs with security use cases [115]. Threats and requirements are abstracted and incorporated into a repository of generic threats and requirements for later reuse.

- **Main concept / idea:**
  - Focus on reusing threats and security requirements described in terms of use cases
  - Identification of threats via misuse cases
  - Identification of security requirement bundles (specifications) using security use cases
  - Establishment of relationships between threats and requirements bundles
- **Structure elements / model:**
  - Meta-model for repository
- **Reuse aspect:**
  - Reuse of threats
  - Reuse of security requirements
  - Storing of all reusable elements as generic threats and requirements in a repository
- **Considered standards / frameworks:**

  - n.a.[11]
- **Process model / steps:**
  Process consisting of five steps:
  1. Identify critical and/or vulnerable assets
  2. Determine security goals for each asset
  3. Identify threats to each asset
  4. Analyze risk for each threat
  5. Determine requirements

2.6.4 Security Requirements Engineering Process (SREP)

Daniel Mellado et al. developed a CC-centered and reuse-based approach for security requirements engineering [73]. The approach builds on the process steps of SQUARE [70]. The components from CC (ISO/IEC 15408) [43] such as functional requirements, protection profiles, and evaluation assurance levels are utilized as reusable information to be provided in a Security Resources Repository (SRR). Based on a meta-model (which is an extended version of the meta-model described in [115]), elements are established using parameter-based mechanisms (e.g., reusable parametrized templates) and non-parametrized templates and checklists (e.g., asset checklists). SRs are elicited by identifying relevant threats in the repository and the related (one or more) associated cluster(s) of SRs.

- **Main concept / idea:**
  - Integration of CC into a reusable security resources repository
  - Reuse of security requirements with UMLSec, expressed as security use cases or as plaintext with formal specification
  - Reuse of threats expressed as misuse cases, threat/attack trees, UMLSec diagrams
  - Concept of assets, threats, requirements, and countermeasures
- **Structure elements / model:**
  Extended meta-model for repository based on [115]
- **Reuse aspect:**
  - Security Resources Repository (SRR)
  - Storing of all reusable elements (e.g., security requirements cluster using CC packages and profiles, threats)
- **Considered standards / frameworks:**
  - CC (ISO/IEC 15408) for security requirements
  - Secure Systems Engineering Capability Maturity Model (SSE-CMM) (ISO/IEC 21827) for security engineering process

  Other mentioned security standards with unclear incorporation in the model:

[11] No standards are directly foreseen in the approach since threats are derived from misuse cases and security requirements from security use cases. However, the authors mention that external sources could also be incorporated.

  - ISO/IEC 13335
  - ISO/IEC 17799
  - ISO/IEC 27001
- **Process model / steps:**
  Process steps based on SQUARE [70]
  1. Agree on definitions
  2. Identify vulnerable and/or critical assets
  3. Identify security objectives and dependencies
  4. Identify threats and develop artifacts
  5. Perfom risk assessment
  6. Elicit security requirements
  7. Categorize and prioritize requirements
  8. Perform requirements inspection
  9. Improve repository

2.6.5 SImple REuse of software requiremeNts (SIREN)

An approach proposed by Ambrosio Toval et al. is called SIREN [130], which is based on a spiral development process model [57] enriched with reuse aspects. Just like in SREP [73], it foresees the creation of a reusable requirements repository, but uses requirements specification standards and recommended practices as basis for the structuring of requirements (called 'requirements specification documents hierarchy') and combines it with MAGERIT [74], the Spanish public administration risk analysis and management method. A repository of SRs initially filled with content from MAGERIT is used, which can be structured according to domains and profiles and, is a concept comparable to that of SREP.

- **Main concept / idea:**
  - Structuring of security requirements according to standard-based documents hierarchies.
  - Reuse of security requirements taken from MAGERIT.
- **Structure elements / model:**
  - SIREN requirements documents hierarchy
  - MAGERIT information system layer shown as potential structure basis
- **Reuse aspect:**
  - Reusable requirements repository composed of profiles and domains
  - Reusable requirements lists taken from MAGERIT
- **Considered standards / frameworks:**
  Requirements specification standards:
  - IEEE Std. 830:1998 [39]
  - IEEE Std. 1233:1998 [38]

  Security standards / frameworks:
  - MAGERIT

- **Process model / steps:**
  Spiral model for RE adapted to SIREN:
  1. Elicitation incorporating reuse of requirements
  2. Analysis and negotiation
  3. Documentation
  4. Validation, incl. improvement of repository

3

Model for Structuring and Reusing Security Requirements Sources

In this chapter all parts of the model for structuring and reusing SRSs will be described. Initially, the categorization of reusable SRSs (corresponding to contribution 1) will be described in section 3.1. The meta-model and the entity relationship diagram (i.e., contribution 2) that were developed will be shown and explained in section 3.2. The approach for instantiating the generic model will be presented in section 3.3. Finally, relevant views and resulting benefits of the model will be described in section 3.4.

3.1 Security Requirements Sources

Ideas, like large rivers,
never have just one source.
Willy Ley

To derive a categorization of reusable, security-specific SRSs to be considered in the model, first the structure and information provided in chapter 2 will be taken and placed into the context of the common security engineering approach (cf. Figure 2.6). The resulting extended security engineering approach including relevant SRS categories will be described in section 3.1.1. Furthermore, the perceptions gained from chapter 2 will be combined with the Twin Peaks model to examine potential SRSs in the context of an iterative development process, as described in section 3.1.2.

Based on the developed extended security engineering approach and the SRS view on the Twin Peaks model, the SRSs to be incorporated in the model will be presented in section 3.1.3, and a definition of SRSs will be given in section 3.1.4.

3.1.1 Extended Security Engineering Approach

From a security engineering perspective, SRs are derived from threats and risks as depicted in the security engineering approach (cf. Figure 2.6). How-

ever, as presented in section 2.4, SRs may not only originate from threats and risks, but can also be derived from any diagnostic security information and knowledge and SRE methods that support the analysis of the problem space. Moreover, raw requirements can also originate from customers and from the environment, which is referred to as compliance obligations in this thesis. These compliance obligations impose raw requirements, which need to be analyzed together with diagnostic security information and knowledge sources and SRE method results before being transferred into well-formed SRs.

The resulting extended security engineering approach is depicted in Figure 3.1, which shows the reusable categories of SRSs as orange boxes.

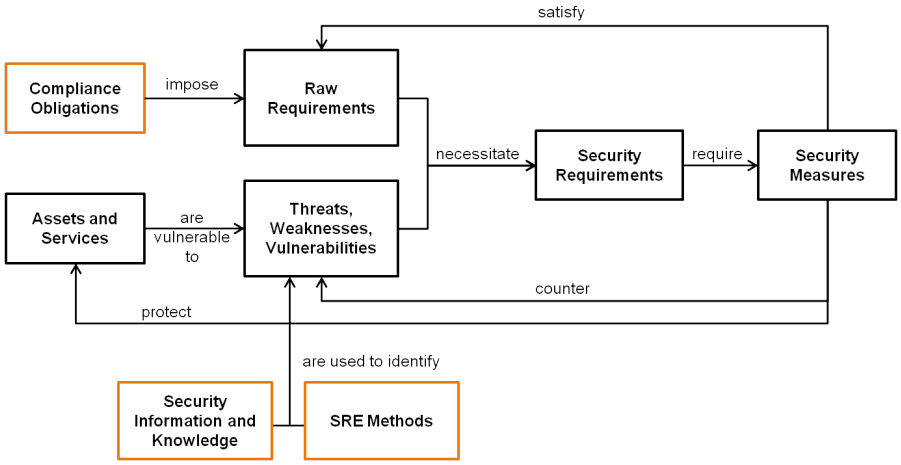


Fig. 3.1. Extended security engineering approach with SRSs

The approach shows that SRs are not only necessitated by threats and risks, but also result from weaknesses and vulnerabilities, as well as other raw requirements.

For the identification of threats, weaknesses, and vulnerabilities, a differentiation is made between the two SRS categories *security information and knowledge* (cf. section 2.4) and results and artifacts from *SRE methods* (cf. section 2.5). *Compliance obligations*, as the third SRS category, impose raw requirements that need to be incorporated and analyzed together with the identified threats, weaknesses, and vulnerabilities when specifying SRs. There are several other potential SRSs from the environment or from customers; however, they can only be provided in a reusable form in exceptional cases (cf. section 2.3).

3.1.2 SRS View on the Twin Peaks Model

Another view on the development of SRs is related to the iterative development of requirements specifications and architectures.

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 [71]), since typically not all constraints and influences can be clarified in advance or they may change during the system's development. Furthermore, design decisions made on the basis of the initial set of SRs and trust assumptions may introduce new influences and constraints such as new or changed threats and weaknesses. This might in turn result in a demand for additional security analysis and additional or revised requirements. Thus, SRE activities should not be performed sequentially in a single phase within the development life-cycle. Instead, (security) RE and architecture design should be interwoven in an incremental approach. To visualize and explain such an incremental approach for security, the Twin Peaks model [87] was extended for this work, as shown in Figure 3.2.

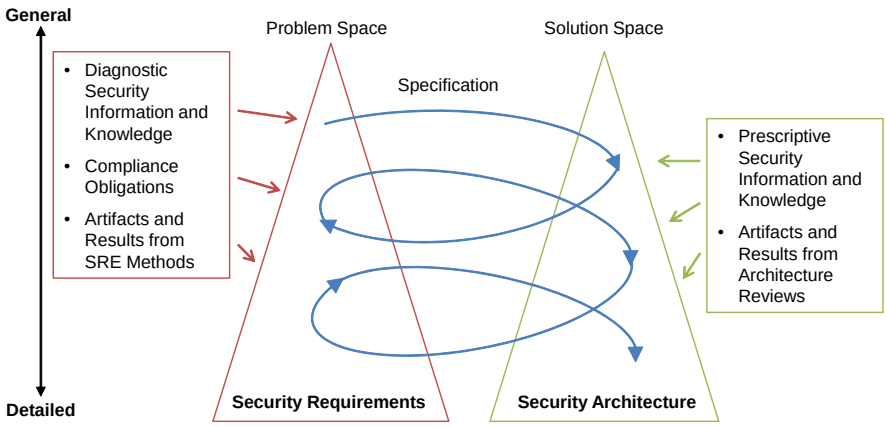


Fig. 3.2. SRS view on the twin peaks model

In this approach, SRs and architectural specifications are developed concurrently after the initial set of SRs has been identified.

As shown in Figure 3.2, the primary SRSs to be considered for the analysis of the problem space are *diagnostic security information and knowledge*, *compliance obligations*, and artifacts and results from *SRE methods*, which are assigned to the SR peak. As primary sources for the solution space, and thus for the architecture and design of security measures, *prescriptive security information and knowledge* and *artifacts and results from architecture reviews* serve as input. As described, both peaks and the respective SRSs are

intertwined in practice, which demands their incorporation in both peaks or development phases, and consequently also the identification and documentation of their relationships. This mutual influence between the problem space and the solutions space indicates that it is highly desirable to combine diagnostic and prescriptive information and knowledge resources for the elicitation of SRs.

3.1.3 SRS Categories to be Incorporated into the Model

Based on the extended security engineering approach and the SRS view on the Twin Peaks model, the following four main SRS categories are incorporated into the model for structuring and reusing SRSs.

- Diagnostic security information and knowledge (cf. section 2.4.1)
- Prescriptive security information and knowledge (cf. section 2.4.2)
- Compliance obligations (cf. section 2.3.3)
- Results and artifacts from SRE methods (cf. section 2.5)

3.1.4 Definition of Security Requirements Sources

In online dictionaries[1], the term *source* is defined as anything from which something comes into being or is derived or obtained (also referred to as origin), or something such as a book, statement, person or document, that supplies information.

In standards for requirements specifications (Institute of Electrical and Electronics Engineers (IEEE) Std. 830 [39], IEEE Std. 1233 [38]), the term requirements source is not explicitly defined. However, the term *requirements source* is used in standards to describe the traceability from a requirement to the originating source. "Each requirement should be further classified by a label that indicates the originator. There may be multiple sources that can all be considered creators of the requirement. It is useful to identify the creator(s) of each requirement so that if requirements are unclear, conflict, or need to be modified or deleted, it will be possible to identify the individual(s) or organization(s) to be consulted" [38]. A requirements specification is traceable "if the origin of each of its requirements is clear and if it facilitates the referencing of each requirement" [39].

For this thesis with its special focus on the reuse of sources, we define the term *Security Requirements Source* considering the perceptions gained from chapter 2, the extended security engineering approach (cf. section 3.1.1), and the SRS view on the Twin Peaks model (cf. section 3.1.2):

Definition 13. ***Security Requirements Source:*** *Security Requirements Sources (SRSs) are the origin from which security requirements come into being or from which they are derived or obtained. The main categories of*

[1] http://dictionary.reference.com/browse/source, http://www.thefreedictionary.com/source

reusable SRSs are diagnostic and prescriptive security information and knowledge sources, compliance obligations, and results and artifacts from SRE methods. SRSs can be provided in different forms such as documents, books, online repositories, persons, or documented method results.

3.2 Meta-Model and Entity Relationship Diagram

"You never change things
by fighting the existing reality.
To change something, build a new model
that makes the existing model obsolete."
R. Buckminster Fuller

In this section, the generic model including its components will be presented, which serves as basis for the instantiation of an organization-specific SRS repository. An overview and explanation of the model will be provided in section 3.2.1, followed by the associated entity relationship diagram for the model in section 3.2.2.

3.2.1 Meta-Model

For structuring and reusing SRSs, a generic model was developed, which serves as a blueprint for the instantiation of an organization-specific SRS repository, as shown in Figure 3.3. The intention was to develop a structure that is as stable as possible and in which SRS elements are structured and provided to development teams for the specification of SRs.
SR *scope areas* and *security topics* are the two main structure elements of the model.

Definition 14. ***Scope Area:*** *A scope area determines the scope of consideration, the work to be accomplished, the problem space to be analyzed, and the security topics to be covered when developing SRs.*

Examples of scope areas are 'software and interfaces', according to the 'IEEE Recommended Practice for Software Requirements Specifications' [39], and 'system', according to the 'IEEE Guide for Developing System Requirements Specifications' [38]. If desired, additional scope areas can be added such as the organizational environment (i.e., operational, personnel and process-related aspects) or the development environment (i.e., aspects related to the development process and environment of a software or system). Alternatively, a different structure or terminology of the scope areas can also be chosen if the typical separation into software and system does not fit, e.g., due to specific domains or use cases.

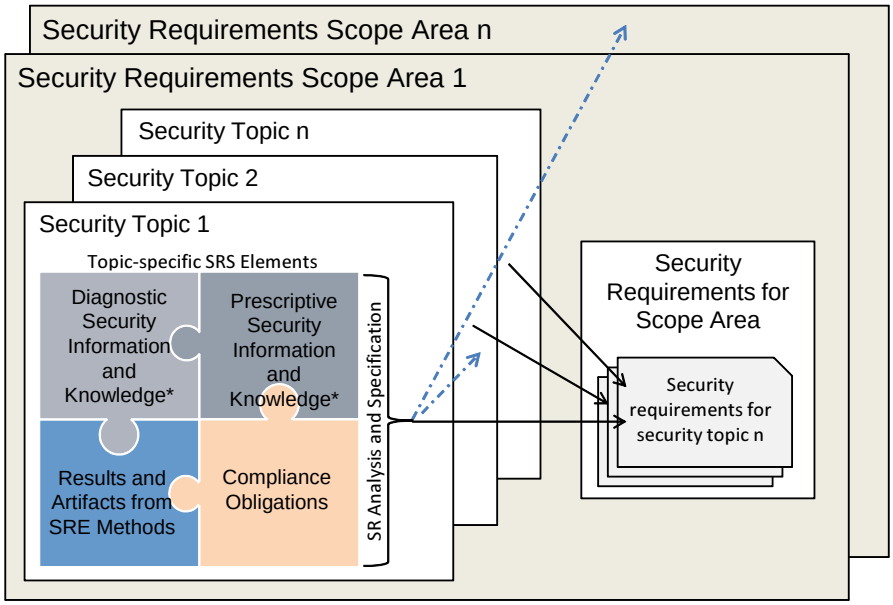


Fig. 3.3. Model overview for structuring and reusing SRSs

Definition 15. ***Security Topic:*** *A security topic aggregates relevant SRS elements that support the analysis of the topic-related problem space and the specification of SRs.*

An example of a security topic in the scope area 'software' is 'user authentication'.

Definition 16. ***SRS element:*** *An SRS element is a single element from an SRS in scope. An SRS element can be assigned to one or multiple security topics in different scope areas.*

Examples of SRS elements assigned to the security topic 'user authentication' are:

- Diagnostic information and knowledge sources: 'CWE-287: Improper Authentication'.
- Prescriptive information and knowledge sources: 'Do authenticate the identity of a user before performing any operation that depends on the verified identity of the subject'.
- Compliance obligations: 'Unambiguous proof of identity by a strong authentication scheme shall be enforced'.

Security topics inherit topic-specific SRS elements from all four SRS categories as defined in the scope for the model (cf. section 3.1.3), namely diagnostic and prescriptive information and knowledge sources, results and artifacts from

SRE methods, as well as raw requirements from compliance obligations. SRS elements are often in a mutual relationship, as indicated by the puzzle pieces. More information on the relationships between the SRS elements and SRS categories in general will be provided in section 3.2.2.

For each security topic, the assigned SRS elements are analyzed and specified into a set of security topic-specific SRs as part of the overall SR specification. In addition, the analyzed SRS elements can also motivate SRs in other security topics and scope areas, as indicated by the blue dotted arrow.

As an example, 'shoulder surfing'[2] is a threat (i.e., an SRS element in the model) that is assigned to SRS the category *diagnostic information and knowledge* in the security topic 'user authentication'. Possible requirements could either specify that a strong authentication scheme shall be enforced (to avoid the problem of shoulder surfing), that the user-name- and password-based authentication mechanism shall mask the actual characters while they are being typed (to reduce the risk of shoulder surfing), or that users should be trained to be aware of the threat and act accordingly (to reduce the risk of shoulder surfing). The first two requirements candidates would be assigned to SRs for user authentication. The latter, however, would need to be assigned to the operational environment, e.g., to an SR for a security topic such as 'training and awareness'.

Figure 3.4 presents a model overview with example scope areas, security topics, and SRS elements from different SRSs assigned to the SRS categories.

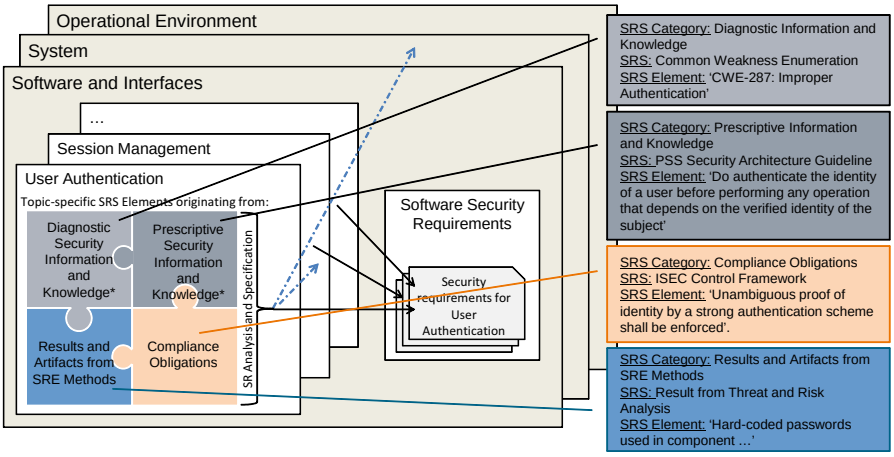


Fig. 3.4. Model example for structuring and reusing SRSs

[2] Shoulder surfing is an observation techniques where an attacker is looking over someone's shoulder to get sensitive information such as user name and password.

In addition, the following example gives an impression of potential SRSs that provide SRS elements to the security topic 'user authentication' in the scope area 'software':

- 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)
- Prescriptive security information and knowledge (provided as reusable information in the security requirements repository):
  - Design pattern for user-name- and password-based authentication
  - Design pattern for certificate-based user authentication
  - Design principles for user authentication (e.g., user authentication should be implemented on the 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

3.2.2 Entity Relationship Diagram

Figure 3.5 shows the entity relationship diagram of the proposed model. For the sake of brevity, most of the entity attributes are omitted here since they need to be customized anyway during the instantiation of the model.

The relationships between the model elements scope area, security topic, and SRS, as described in section 3.2.1, are depicted in the upper part of the Entity Relationship (ER) diagram. It shows that one scope area consists of one to many security topics.

A security topic consists of one to many SRS elements. Each SRS (element) has a unique SRS category assigned by an attribute (e.g., compliance obligation). SRs are needed by one to many SRS elements, or conversely, one to many SRS elements are used for the specification of an SR. This is how the traceability from the resulting SRs to the SRS(s) can be ensured.

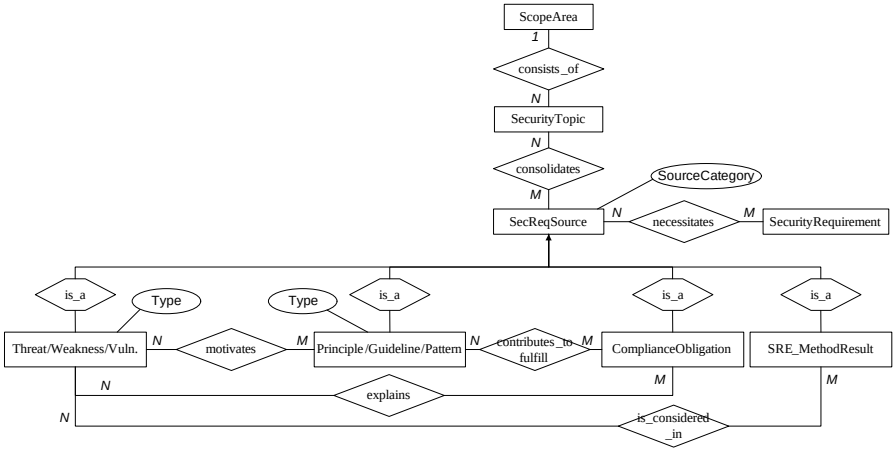


Fig. 3.5. SRS model - Entity relationship diagram

The inherited subordinate objects in the lower part of the ER diagram reflect the different SRS categories and their relationships. The subordinate objects correspond to the SRS categories, as depicted in Figure 3.3:

- 'Threat/Weakness/Vuln.' corresponds to diagnostic information and knowledge sources.
- 'Principles/Guidelines/Pattern' corresponds to prescriptive information and knowledge sources.
- 'ComplianceObligations' corresponds to compliance obligations.
- 'SRE_MethodResult' corresponds to results and artifacts from SRE methods.

The relationships between the SRS elements from the different SRS categories can be summarized as follows:

To do proper SRE, understanding of the problem space (i.e., threats, weaknesses, and vulnerabilities) is required to identify and analyze the bad things that should be prevented in order to specify adequate SRs. This information is included in diagnostic information and knowledge sources, or can be identified with SRE methods.

Diagnostic information and knowledge sources have a strong relationship with prescriptive information and knowledge sources and compliance obligations since they motivate and provide the problem to be solved by prescriptive information (cf. section 2.4.4). Moreover, they explain the motivation why certain controls or raw requirements in security standards exist and allow examining whether a selected security measure is suitable for satisfying the raw requirement from the security standard and for mitigating the underlying security problem. Furthermore, they can provide valuable input to the application of SRE methods such as threat and risk analysis methods in order to

ensure that the most common threats, weaknesses, and vulnerabilities have been considered.

Another important relationship exists between prescriptive information and knowledge sources and compliance obligations. Although mainly addressing the solution space, prescriptive information and knowledge can be helpful to support the alignment between the problem and the solution space in which requirements engineers and system architects work concurrently and iteratively increase the level of detail of the requirements specification and the architecture design. In doing so, they provide valuable information such as security principles, design pattern, or control lists, which contribute to the fulfillment of raw requirements from compliance obligations.

Besides the above-mentioned relationships reflected in the ER diagram, the following additional relationships exist (note that this list not exhaustive), which could also be considered in the model, if required.

- There can exist relationships between SRS (elements) of one SRS category. For instance, helpful relationships do exist among elements from different SRSs in the SRS category *diagnostic information and knowledge sources*, such as the relationships between threats, weaknesses, and vulnerabilities. Another example from the SRS category *prescriptive information and knowledge* is the relationship between security principles and related security patterns or controls.
- There can exist relationships between SRE methods and prescriptive information and knowledge sources, e.g., if checking the integration of certain security principles for a product or system is desired.

The relationships between the SRS (elements) as depicted in Figure 3.2.2 can be changed or refined if required, e.g., if more fine-grained relations between different SRSs need to be modeled, or to ensure that the terminology is in line with the security ontology used in an organization[3].

3.3 Model Instantiation Approach

"Out of clutter, find simplicity.
From discord, find harmony.
In the middle of difficulty lies opportunity."
Albert Einstein

In this section, the prerequisites and the approach for instantiating the model described in section 3.2 will be presented. Using the generic model presented in section 3.2, the following steps, inputs, and outputs will be proposed to instantiate the model and to create the SRS repository, as shown in Figure 3.6.

[3] A literature survey of different security requirements ontologies is provided in [120].

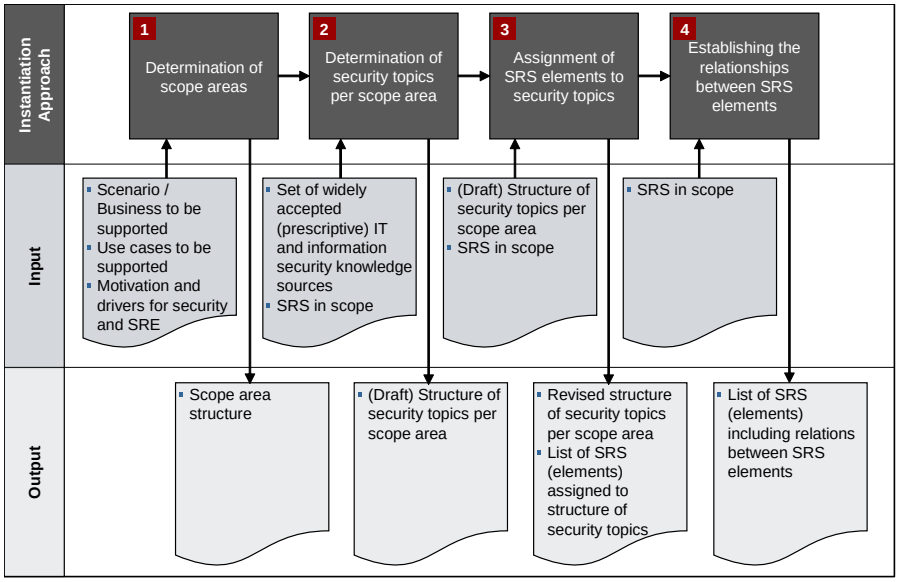


Fig. 3.6. SRS model instantiation approach

3.3.1 Prerequisites for Model Instantiation

To be able to successfully instantiate the model for a scenario or organization, the following prerequisites should be fulfilled.

Required Business- and Scenario-specific Decisions

Before the model for structuring and reusing SRSs can be instantiated, it is important to decide on the fundamental scenario(s) and business as well as the use cases to be supported by the SRS repository. Moreover, the motivation and the drivers for security and SRE must be identified and specified before the instantiation of the model is started. This information is of particular importance since it has significant influence on the structure of the scope areas and security topics, the terminology used, the SRSs to be incorporated, and the complexity of and the effort for applying the resulting repository in the organization.

Important considerations to be regarded are (but are not limited to):

- **Intended user groups, organizational units, and application domain(s) to be supported:** It is important to define the intended user groups (e.g., software development projects, system integration projects, entire software and system scope), organizational units (e.g., corporate information security organization and enterprise IT units, departments developing products or solutions, entire organizations) and the application

domains (e.g., enterprise and office IT, automotive, industry, healthcare) in which the developed SRS repository will be used. This makes it possible to clearly specify the intended focus and pursue it during the instantiation of the model. For instance, the required scope areas, security topics, and SRSs to be incorporated in software development (e.g., a web application) differ from development projects where system integration, deployment, and configuration aspects are the primary focus.

- **Obligatory SRSs and related terminology to be incorporated:** It makes a difference for the instantiation if mandatory SRSs such as internal or external compliance obligations need to be incorporated. Where applicable, it must be decided which of them is the most relevant or important source to be regarded when determining scope areas and security topics. An example: if for a company or organizational unit in a particular domain the primary driver for their security are compliance obligations, they would be well advised to follow the structure and terminology of the security standard instead of developing their own proprietary structure of scope areas and security topics. A prominent example for industrial automation control systems is IEC 62443, which already provides a structure and controls for different scopes, e.g., for the system level (i.e., IEC 62443 3-3). If no obligatory SRS(s) exist, one must decide if one of the SRSs in scope should be the leading one, or if a completely new organization-specific structure of scope areas and security topics can and should be developed.
- **Security strategy and use cases of a company or organizational unit:** Depending on the desired coverage of security aspects and SRSs to be incorporated, the security level as well as the protection against potential attacker types can be influenced. The higher the level of broadness and detail of the incorporated SRSs becomes, the higher the chance of specifying sufficient and adequate security requirements and, consequently, the higher the chance of implementing proper security mechanisms. Therefore, before the start of the instantiation of the SRS repository, tradeoff decisions need to be made concerning the desired broadness and depth of the security investigations as well as regarding the resulting effort for using the SRS repository in projects.

Organizational and Skill-related Prerequisites

Development teams and (S)RE practitioners without security skills and knowledge are caught in the dilemma that they would require security and information sources to overcome their lack of security skills and knowledge, but simultaneously require a certain level of security knowledge and expertise to identify, classify, compare, correlate, and use security information and knowledge sources to gain benefit from them, as outlined in section 1.2. Since the instantiated model is intended to overcome this dilemma by providing an instantiated SRS repository, profound knowledge about relevant SRS and se-

curity in general is required to successfully create an SRS repository. Besides security skills and knowledge, further knowledge and capabilities are also required. Before instantiation the model, it is recommended identifying the following required knowledge and capabilities and the associated example roles to be involved during the respective steps.

1. **Knowledge about relevant compliance obligations, their scope and content:**
   a) External compliance obligations such as laws, statutory, regulatory, or contractual obligations: legal advisor or suitably qualified legal practitioner, compliance officers, quality manager, auditor.
   b) Internal compliance obligations such as policies, security control frameworks: (corporate) information security officer, product and solution security officer, compliance officers, quality manager, assessor.
2. **Business decisions and security strategy:** Chief executive officer, product manager, (corporate) information security officer, product and solution security officer.
3. **Security knowledge and skills:** Security expert, information security adviser.

3.3.2 Instantiation Approach Steps

Four process steps are proposed to be followed for instantiating an organization-specific SRS repository.

Step 1) Determination of Scope Areas

The first step is to determine which scope areas are in scope for the resulting SRS repository. To identify the relevant scope areas, people with knowledge and skills in the area of (internal and external) compliance obligations as well as representatives from business and security should be involved.

If a standard-compliant selection of scope areas is desired, the typical partitioning of scope areas would be *software and interfaces* and *system*, following the practice recommended by the IEEE software and system requirements specification standards [38, 39]. Additionally, if operational process requirements or development-specific requirements are also to be specified, additional scope areas such as *organizational environment* (i.e., operational personnel and process-related aspects) and *development environment* (i.e., aspects related to the development process and environment of a software or system) should be added.

Schmitt and Liggesmeyer propose in [101] the differentiation of scope areas into *software*, *physical and technical environment*, and *organizational environment*. Moreover, if required, a fourth scope area *development environment* can

also be added, which addresses all aspects required for developing a product or solution securely[4].

To properly determine the relevant scope areas as the first structuring layer, following inputs regarding business- and scenario-specific decisions should be considered, which are the prerequisites for step 1 (cf. section 3.3.1).

- Scenario / Business to be supported
- Use cases to be supported
- Motivation and drivers for security and SRE

The output of step 1 is the resulting scope area structure, which determines the scope of consideration, the work to be accomplished, the problem space to be analyzed, and the security topics to be covered when developing SRs.

Step 2) Determination of Security Topics per Scope Area

To determine a suitable structure of security topics for scope areas in the scenario, the following approach is recommended. Knowledgeable persons regarding internal and external compliance obligations should be involved, cf. section 3.3.1.

(a) **Development of a generic structure blueprint:** First, a set of widely accepted IT and information security knowledge sources is chosen that is relevant for the organization for which the SRS repository is to be instantiated. These standards do not necessarily need to be added as SRSs; however, they serve as good practice regarding terminology and structure in the respective domain. It is recommended focusing primarily on prescriptive information and knowledge sources, as well as compliance obligations since they already provide a consistent and 'security-topic-like' structure. Example standards for enterprise/information IT security are [48, 83] and for product and solution security [36].
The identified standards are analyzed concerning their structure and the incorporated security aspects, and are mapped to the determined scope areas (as specified in step 1). This allows identification of common and recurring terminologies for security topics that fit the particular organization and application domain. The resulting structure serves as a first draft structure of security topics for each scope area, which can be used for the mapping of the SRSs in the scope.

(b) **Development of a scenario-specific structure:** In the second sub-step, the elements of the relevant SRSs in scope are coarsely mapped to the draft structure of the security topics as developed in sub-step (a). If an element does not fit into the existing structure of the security topics, it is noted down separately for later analysis. Elements that cannot be assigned uniquely to a single security topic, e.g. because of their generic

[4] In contrast to the development of secure products and solutions.

nature, are assigned multiple times. In this way, a structure of security topics can be derived for each scope area.

(c) **Troubleshooting:** In the third sub-step, the SRS elements that did not fit into the existing structure (from sub-step (b)) are analyzed once again, which results either in a reorganized structure or the creation of additional security topics.

(d) **Quality assurance:** In order to create a consistent and acceptable (draft) structure of security topics for each scope area, several iterations of revisions should be performed.

The output of step 2 is a relatively stable organization-specific SRS repository structure consisting of scope areas and security topics, as depicted in Figure 3.7.

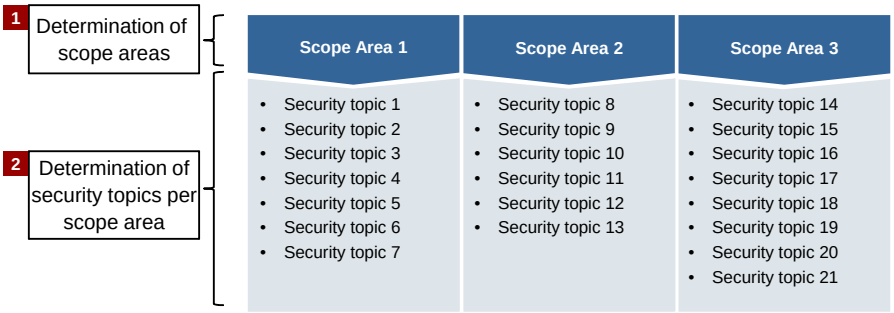


Fig. 3.7. Structure of scope areas and security topics

Step 3) Assignment of SRS Elements to Security Topics

In the third step of the instantiation approach, all elements from all SRSs in scope are mapped and assigned to the SRS repository structure developed in step 2. Please note that in practice, steps 2b-c and step 3 may be parallelized to a certain degree, particularly if the number of SRS elements in scope is relatively low.

For the mapping and assignment, a structured walk-through is performed for each SRS, where each single SRS element is assigned to the security topics in the respective scope areas. Multiple assignments of SRS elements to security topics are possible. In general, the more generic an SRS element, the higher the likelihood of multiple assignments. SRS elements that are overly generic and thus would need to be mapped to almost every security topic can either be left out or should at least be marked. Figure 3.8 shows the basic idea of the assignment of SRS elements to security topics, represented by the black dotted arrows.

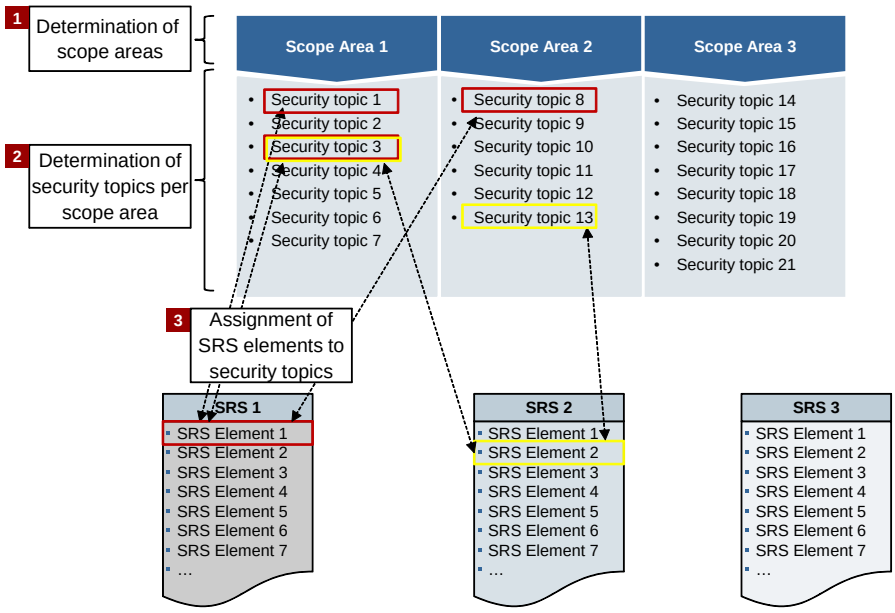


Fig. 3.8. Assignment of SRS elements to security topics

In the following an example of a mapping of SRS elements to security topics will be given to make this process step easier to understand.[5]

Example - Mapping of an SRS element from a diagnostic security information and knowledge source. Figure 3.9 shows an example of the assignment of an SRS element from the SRS *Common Weaknesses Enumeration* to an example structure of scope areas and security topics.

SRS Content			Assigned Scope Areas and Security Topics		
Weakness ID	**Weakness Topic**	**Description**	**Software**	**System in Technical Environment**	**System in Operational Environment**
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	

Fig. 3.9. Assignment of CWE element to scope areas and security topics

[5] For the example we assume that the scope areas and security topics mentioned in the example were identified in steps 1 and 2. However, please note that these scope areas and security topics do not necessarily have to be identical in other organization-specific SRS repositories. The evaluation result of a larger set of SRSs at Siemens AG is presented in chapter 4

For the example we use the SRS element 'CWE-759: Use of One-Way Hash without salt' from SRS [127].

This SRS element would be assigned to the three scope areas as proposed in [101] and the first example security topics due to the following rationale:

- 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 the security topic *SW cryptography and key management*.
- System in technical environment: Security topics *access control* and *system data at rest*.
  Rationale: Hashed passwords are stored somewhere at the system-level, e.g., in files of the operating system or somewhere in the database. Therefore, this weakness is also related to the security topics *access control* and *system data at rest* since passwords need to be secured against unauthorized access and security of data at rest (in this case passwords) must be ensured.
- System in operational environment: In this scope area, no relevant security topics are assigned for the example since the implementation of secure password storage via salted hashes can only be implemented in the scope areas *software* and *system in its technical environment*. Please note that if the example weakness would have been different, e.g., 'creation of weak passwords', then in this scope area a security topic such as *awareness and training* would also be of interest.

As the example indicates, each SRS element needs to be read, understood, and interpreted before it can be assigned to the structure of the security topics in the various scope areas. Therefore, it is inevitable that for the third process step, people with profound security knowledge and skills must be involved.

Step 4) Establishing the Relationships between SRS Elements

Once a suitable structure of scope areas and security topics has been created and the respective SRS elements have been assigned to security topics, the relationships between the various SRS elements must be identified and documented, following the generic relationship model as depicted in Figure 3.5.

In practice, this means that for each element from the SRS it needs to be analyzed and documented to which other SRS element a relationship exists. In Figure 3.10 some example relationships between SRS elements are illustrated by green arrows.

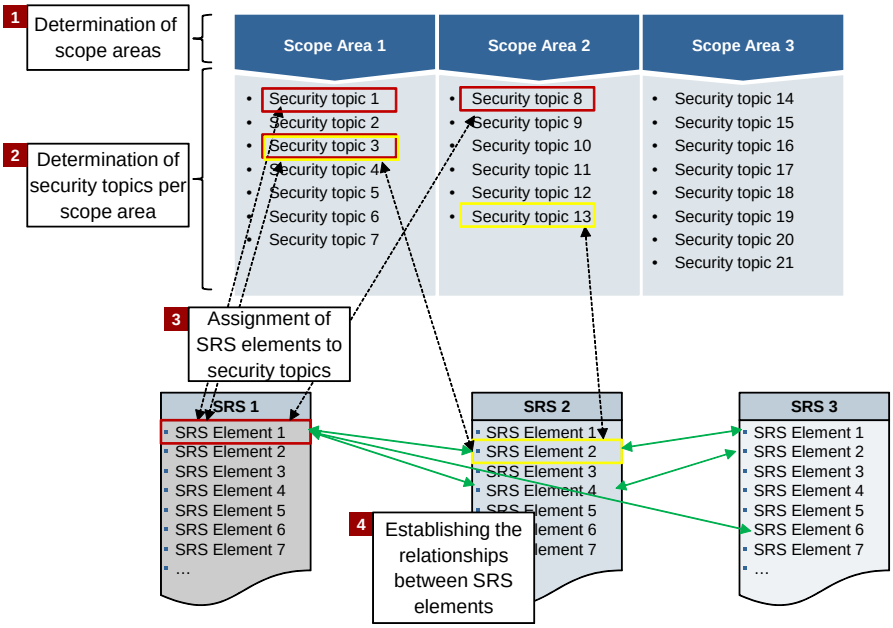


Fig. 3.10. Establishing relationships between SRS elements

Example - Relationships between an element from diagnostic information and knowledge and other SRS elements. For the threat 'Eavesdropping', all relationships to other relevant SRS elements in scope are documented, such as 'Encryption of data during transport or logical and/or physical separation in a dedicated network' (which is an element from a prescriptive information and knowledge source, supporting the mitigation of the threat) and 'Personally Identifiable Information (PII) data transferred over networks shall be protected against unauthorized disclosure and access' (which is an affected raw requirement from a compliance obligation). Furthermore, helpful relationships between different diagnostic SRS elements can also be documented, e.g., to concretize the generic threat with weaknesses from CWE such as 'CWE-327: Use of a Broken or Risky Cryptographic Algorithm'. This allows providing all related weaknesses and vulnerabilities related to a particular threat in order to enhance the understanding of the problems to be solved.

Just as for step 3, people with profound security knowledge and skills must be involved to provide the understanding of the various SRS elements and their relationships.

3.4 Model Views and Benefits

"I'm constantly asking for alternative views
on most things that come to me."
Paul Wolfowitz

An instantiated SRS repository provides all relevant information to understand and analyze the security problem space and to specify security requirements. Scope areas and security topics, and the assigned relationships from SRS elements to security topics and between the various SRS elements, provide the necessary basis; however, their usability and efficiency highly depend on their automated analyzability. The usability and efficiency of an instantiated SRS repository can be highly increased by providing SRE practitioners with useful automated views, just like in any database system. The following four views are foreseen for the efficient use of an instantiated SRS repository. The views are ordered from generic to concrete.[6]

3.4.1 Scope Area View

The first view shows the structure of the scope areas and the assigned security topics. Although this view is the most generic and trivial one, it is mentioned here since it provides the problem space to be analyzed and the security topics to be covered when developing security requirements in a concrete project. For instance, if a distinction between software and system is made for the scope areas, the relevant scope and the relevant security aspects to consider for a software project are provided via the scope area *software* and all security topics assigned to this scope area. Therefore, this view is not only important for SRE practitioners but also for product or project managers and quality officers since all relevant security topics to be covered in a project are shown.

3.4.2 Security Topic View

The security topic view shows all SRS elements that are assigned to a particular security topic. This view therefore queries for all SRS elements assigned during the instantiation approach in step 3 (cf. section 3.3.2).

For instance, the security topic view on *user authentication* in the scope area *software* provides all SRS elements from all SRSs in scope assigned to this topic. The result of the query can be filtered and sorted according to different criteria.

As filter criteria, a user of the SRS repository can only display elements from particular SRSs or SRS categories. As an example, only SRS elements from the SRS within a specific SRS category can be shown for a security topic,

[6] In an operational tool, these queries should be automated, e.g., by using views in a database system.

e.g., to display only relevant SRS elements related to the security topic *user authentication* that originate from the SRS category *compliance obligations*. Moreover, a filter could also be set on one specific SRS, or specific SRSs from different SRS categories.

The sorting of the query result can provide all relevant SRS elements, structured first according to their SRS category and second according to their SRS.

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

- Understand the problem space for a security topic and 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 it with software or system (security) architects' viewpoints.
- Identify identical or similar raw requirements from different compliance obligations to increase the efficiency for fulfilling these requirements.
- Incorporate results from previous applications of SRE methods to avoid the recurrence of security problems.

3.4.3 SRS Element View

With the third view, all SRS elements with relationships to a single SRS element are shown. For instance, for a raw requirement from a compliance obligation, all related diagnostic and prescriptive information and knowledge sources as well as results from SRS methods are provided, which help to understand and satisfy the compliance obligation.

This leads to the following benefits:

- Possibility to view all related diagnostic information and knowledge sources as well as 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 that 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 into an SRE method result (e.g., an identified weakness from a threat and risk analysis).

3.4.4 SRS Coverage Views

As presented in section 3.2.2, a relationship between an SR and the originating SRS elements exists in order to ensure traceability. In practice, this means that every time an SR is specified, the incorporated SRS elements that led to the specification of the SR need to be documented.[7] For instance, it must be documented which raw requirements stem from compliance obligations or which threats are addressed by the specified requirements.

Through this creation of relationships between SRs and incorporated SRS elements, a view showing the coverage of SRSs can be created. For instance, for a threat list (as an example of a diagnostic information and knowledge source) it can be checked if all threats in the list (i.e., all SRS elements) are incorporated into at least one of the specified SRs. Another example would be an external control list (as an example of a compliance obligation), for which it can be checked if at least all raw requirements have been considered in the resulting set of requirements and, if not, which are still open. Please note that for all SRSs and their elements, it can only be checked if they were considered in the final set of requirements. However, if one also wants to check if the SRS elements are addressed completely (e.g., if a threat is addressed completely or if a raw requirement is covered entirely by the resulting set of SRs), an additional metric for measuring the completeness or maturity would have to be incorporated.

The following benefits result from this view:

- Possibility to view which elements of SRS in scope are already incorporated into the specified set of SRs.
- Possibility to view for each specified SR which SRS elements are incorporated (e.g., which threats are addressed, which design pattern is related to the requirement, which compliance obligation is satisfied by the requirement).

[7] This documentation of incorporated SRS elements per SR should be supported with a proper tool support.

4

Evaluation

"Don't fear failure so much that you refuse to try new things.
The saddest summary of a life contains three descriptions:
could have, might have, and should have."
Louis E. Boone

In this section, the evaluation of the theoretic parts of this PhD thesis will be described (cf. sections 3.1-3.4).

After presenting the evaluation scenario and the SRSs in scope, the instantiated SRS repository and the resulting views and benefits will be presented. Representative examples demonstrating the applicability of the model and the instantiation approach in a real-world scenario at Siemens AG will be shown and empirical perceptions and challenges will be described.[1]

4.1 Tool Support for the Evaluation

The evaluation was done completely using Microsoft Excel® and its built-in functionality, particularly the customizable filter mechanisms. All SRSs were transferred from their original formats into spreadsheets in order to harmonize their layout and formatting.[2] All examples and views shown in this thesis were composed using custom filters for each spreadsheet.

For commercialized or professional use of the model and an instantiated SRS repository during the course of daily business, a more automated and user-friendly tool, preferably based on a database and convenient Graphical User Interface (GUI) is recommended, offering features such as interfaces for

[1] Due to company-internal information handling reasons and binding non-disclosure agreements, not all SRSs and the resulting instantiated SRS repository will be shown in detail.

[2] The original formats of the SRSs are .pfd [49, 100, 110, 114], html/website [127], .xls [113], .doc [112].

data upload and predefined queries and views for using the repository. However, to show the applicability and the results of the proposed model and instantiation approach, Microsoft Excel® is the tool preferred by the author, particularly due to many assignments, enhancements, and adaptations done during the evaluation.

4.2 Evaluation Scope and Scenario

The following industrial scenario was used to evaluate the applicability of the proposed model and the instantiation approach.

4.2.1 Business- and Scenario-specific Decisions for the SRS Repository to be Instantiated

The evaluation focuses on combining various SRSs from three different Siemens corporate departments, namely Product and Solution Security (PSS), Corporate Information Security (ISEC), and Healthcare Data Protection (HDP). The SRSs of these departments are enhanced with additional external diagnostic and prescriptive information and knowledge sources. Prior to the evaluation, none of the SRSs had been aligned or correlated with each other.

The goal of the instantiation was to develop a preferably universal structure of scope areas and security topics, resulting in a Siemens-specific SRS repository that can be used for the elicitation of SRs by PSS and ISEC practitioners.

The aim was to make it possible to identify relevant SRS elements for one or several of the incorporated SRSs (particularly for the compliance obligations) together with additional helpful information for understanding the problem space underlying each SRS. For this reason, not only the SRSs for a single company department, but also the SRSs from all different company departments and the additional security SRSs were to be correlated with each other. The resulting structure and terminology needed to be universal enough to fit the different SRSs and purposes of the different organizational departments. This means that the instantiated model is not structured according to a particular compliance obligation nor is it specifically customized for a certain target group such as software or system architects.

Differentiation between typical aspects related to software, systems, and additional non-technical and organizational topics was to be ensured. From an organizational perspective, at least the following use cases were to be supported, which are an integral part of IT security, product management, and product life-cycle management processes:

- Provision of relevant SRSs to be considered and proof of compliance, when developing ISEC security concepts for an application or a complete IT system (consisting of an application and infrastructure). At least compliance

with SRSs from the ISEC department needs to be ensured. Optionally, additional SRSs shall be extendable.
- Provision of relevant SRSs to be regarded when developing products or solutions (i.e., in the PSS department). The SRSs are intended to be used during the design of security architectures and the conduction of threat and risk analysis workshops.
- Provision of SRSs for which HDP is responsible, enhanced with additional guidance from other SRSs. The goal was not to ensure compliance with data privacy laws. However, the provided SRSs are intended to support the development of a security concept that can be used as part of a procedure index (according to BDSG).

SRS elements that are related to the security life-cycle (e.g., what the development environment should look like, or which kind of security tests should be conducted at which point in time in the life-cycle) were left out, since they are not relevant for the presented scenarios.

4.2.2 Considered Security Requirements Sources

To achieve a high significance and broadness of the evaluation, the following SRSs from the three Siemens AG departments ISEC, PSS, and HDP were selected and extended with additional, non-company SRSs from external repositories. In total, over 500 elements from eight different SRSs had to be structured, covering at least one representative per SRS category, as depicted in Table 4.1.

SRS Category	**Security Requirements Source**
Diagnostic Information and Knowledge Sources	PSS threat and risk analysis questionnaire [110] Selected weaknesses from CWE [127] Deliberate threats from ISO/IEC 27005 [49]
Prescriptive Information and Knowledge Sources	PSS secure architecture guidelines [100] ISEC secure software development policy [114]
Compliance Obligations	ISEC information security control framework [113] HDP data privacy controls [112]
SRE Method Results	Results from selected threat and risk analysis workshops

Table 4.1. SRS sources in the evaluation scenario

Due to this broad basis of SRSs, different technical, physical, and organizational security aspects in different scope areas had to be structured and their relationships had to be established. The following examples give an impression of SRS elements in order to illustrate the content of the various SRSs. To facilitate comparability, all examples were selected from the security topic 'authentication'.

Example Diagnostic Information and Knowledge Sources

Tables 4.2 - 4.4 show some example SRS elements from the incorporated diagnostic information and knowledge sources.

The purpose of the PSS threat and risk analysis questionnaire is to support threat and risk analysis workshop teams in the identification of common IT security weaknesses and corresponding threats.

PSS threat and risk analysis questionnaire			
ID	**Threat Title**	**Threat Question (to be asked during Threat and Risk Analysis)**	**Threat Explanation**
SW-1	Missing User Authentication	Is it ensured that a successful authentication (not only identification) against the application is required / mandatory before – any non-public action is executed on behalf of a user, – any non-public data is presented to a user or to the user's client, or – access to any non-public subcomponent is granted?	An adversary might be able to bypass user authentication using access paths that are not covered by user authentication mechanisms. The adversary might learn about data they are not authorized to see. Thus, an adversary might have unauthorized access to data or might be able to perform unauthorized actions. Depending on the concrete scenario, the impact might be disastrous.
SW-2	Client-Side User Authentication	Does your software provide robust and secure server-side (not client-side) authentication mechanisms?	Unauthorized Access to Resources: An adversary can usually very easily manipulate the client code to circumvent client-side checks. This might be done by e.g. reverse engineering or modifying the code at runtime using a debugger. A client, as a matter of fact, has to be deemed insecure as it is very hard to prevent manipulation of code an adversary has access to. Thus, final authentication decisions must never be taken on the client-side, but always on the server-side.
...	...	...	...

Table 4.2. SRS 1 – PSS threat and risk analysis questionnaire [110]

The selected weaknesses from CWE contain example entries from the community-developed CWE online dictionary. The intention of the dictionary is to provide "a unified, measurable set of software weaknesses" and to enable a "better understanding and management of software weaknesses related to architecture and design" [127].

Selected weaknesses from CWE		
ID	**Weakness Title**	**Description Summary**
CWE-759	Use of a One-Way Hash without a Salt	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 part of the input.
CWE-798	Use of Hard-coded Credentials	Hard-coded credentials typically create a significant hole that allows an attacker to bypass the authentication that has been configured by the software administrator. This hole might be difficult for the system administrator to detect. Even if detected, it can be difficult to fix, so the administrator may be forced into disabling the product entirely.
...	...	...

Table 4.3. SRS 2 - Selected weaknesses from CWE [127]

A list of deliberate threats was taken from the international standard for information security risk management ISO/IEC 27005:2011. The incorporated threats used as SRS elements during the evaluation are deliberate threats from the (informative) threat list provided in annex c.[3]

[3] Accidental and environmental (i.e. natural) threats were left out since they are primarily relevant for information security, but not for IT security, which is the primary focus of this PhD thesis.

Deliberate threats from ISO/IEC 27005		
ID (self-defined, not from SRS)	**Threat Type**	**Threat**
ISO-37	Unauthorised actions	Corruption of data
ISO-38	Unauthorised actions	Illegal processing of data
...	...	...

Table 4.4. SRS 3 - Deliberate threats from ISO/IEC 27005 [49]

Example Prescriptive Information and Knowledge Sources

Tables 4.5 and 4.6 show some example SRS elements from the incorporated prescriptive information and knowledge sources.

The purpose of the PSS secure architecture guideline is to "provide the reader with guidance on architectural security design considerations, required for the secure design of software/components, systems and solutions" [100].

PSS secure architecture guidelines		
ID	**Guideline Category**	**Guideline Topic**
A-UA-002-D001	Default User Authentication Functionalities	Do provide mechanism to set the initial user passwords for each user to a random value, fulfilling the defined password quality criteria.
A-UA-002-D002	Default User Authentication Functionalities	Do enforce a mechanism to randomly generate initial authentication secrets, if the initial authentication secrets are created before a user can do so. Never use default passwords.
A-UA-002-D003	Default User Authentication Functionalities	Do only use well-proven and secure algorithms for automatic password generation to avoid predictable sequences.
...	...	...

Table 4.5. SRS 4 - PSS secure architecture guidelines [100]

The ISEC secure software development policy provides rules and principles with a special focus on the development of secure software.

ISEC secure software development policy	
ID	**Guideline Topic**
SSD-15	Every access to non-public interfaces by users or systems must be appropriately authenticated. (Note: Non-public interfaces can be web applications, web services, backend system connections, database connections and administrative interfaces.)
SSD-16	Use dedicated framework/library functions for password fields in user interfaces, for example password fields that mask the actual characters and that prevent content caching and auto-completion.
SSD-18	After a failed authentication attempt, do not report the exact cause of the failure: whether the user was unknown or whether the user existed but the password was incorrect.
...	...

Table 4.6. SRS 5 - ISEC secure software development policy [114]

Example Compliance Obligations

Tables 4.7 and 4.8 show some example SRS elements from the incorporated compliance obligations.

The ISEC information security control framework prescribes a mandatory list of protection requirements to be followed when developing and implementing applications or infrastructures for Siemens AG. It is a list of raw requirements derived on the basis of the classification of the processed and stored information and the resulting security level.

ISEC information security control framework			
ID	**Security Level 1 (Low)**	**Security Level 2 (Medium)**	**Security Level 3 (High)**
402002-02	Single-factor authentication (i.e. something you know, something you have or something you are) following good security practices.	Unambiguous proof of identity by a strong authentication scheme (higher level of assurance than single-factor authentication; according to current state of technology)	Non-repudiable proof of identity by a strong authentication scheme through two-/ multi-factor authentication (according to current state of technology)
...	...	...	...

Table 4.7. SRS 6 - ISEC information security control framework [113]

The HDP data privacy controls are Siemens-specific data privacy requirements that need to be followed in order to achieve compliance with data privacy laws.

HDP data privacy controls		
ID	**Control**	**General Guidance**
P8	Every access to network and user interfaces shall be authenticated.	Authentication is the prerequisite for preventing unauthorized access or disclosure of PII. This applies to interactive access by users as well as to access by other systems. After successful authentication the authorization of the user or system has to be checked to determine if access is allowed.
P9	Password-based authentication shall be state-of-the-art.	Today, state-of-the-art password based user authentication comprises the following aspects. Each aspect shall be supported. a) The user shall be able to change his password. b) A configurable password policy, allowing to enforce - password changes at regular intervals, - complex passwords with sufficient length, - policy changeable by administration role only c) Mechanisms to prevent password guessing, e.g. enforcing delays after failed authentication requests d) The authentication protocol itself shall be secured, i.e. passwords shall not be transferred in plaintext and secured against replay. Use of SSL/TLS with server-authentication is considered as state-of-the-art for protection of passwords during transmission. e) Passwords shall be stored securely within the system, i.e. restrictive access rights, storage as salted hash where possible.
...	...	...

Table 4.8. SRS 7 - HDP data privacy controls [112]

Results from selected PSS threat and risk analysis workshops

'Results from selected threat and risk analysis workshops' are real findings from Threat and Risk Analysis (TRA) workshops performed at Siemens AG. Due to the strictly confidential content of threat and risk analysis workshop results, this particular SRS is not available for anybody except for the TRA team and the product owner. Therefore, this SRS is not handled like the other SRSs or a normal reference in this thesis. Since TRA workshop results reveal security problems to avoid, such as threats and weaknesses (cf. section 3.1.3), they can be treated similar to diagnostic information and knowledge sources and therefore not shown explicitly in the following evaluation. A neutralized example of a TRA workshop is presented in Table 4.9, based on the Siemens TRA method [105][4].

Results from selected threat and risk analysis workshops		
TRA Workshop Reference	**Threat**	**Risk**
TRA workshop for SW product <anonymized>, <date>.	Use of hard-coded password to encrypt data in transit between component <anonymized> and <anonymized>. If revealed, this enables an attacker to gain unauthorized access to information.	• Exposure rating (likelihood whether an attack may be attempted): The software can be reached via the internet. -> *Exposure rating: High* • Exploitability rating (likelihood whether an attempted attack is likely to succeed): Hard-coded passwords might easily be found out using e.g., available documentation, reverse engineering, or debugging. Hard-coded passwords can't be changed easily and therefore exploited over a long period of time by adversaries.to grain unauthorized access to information and services. -> *Exploitability rating: High* • Impact: If hard-coded passwords are revealed and available on the internet, customers don't buy the product any more which would lead to critical loss of reputation, customers and market share. -> *Impact Rating: Critical* • Overall risk rating: *Major risk* (risk has to be treated with highest priority in terms of definition and implementation of countermeasures or accepted by senior management).
...	...	

Table 4.9. Results from selected threat and risk analysis workshops

4.3 Instantiated Model

To apply the instantiation approach presented in section 3.3.2 on the SRSs as defined in the evaluation scenario and scope (cf. section 4.2), the following inputs were used and outputs were created using the defined instantiation approach steps. Further information on perceptions and challenges encountered are not explained here for each instantiation step, but will rather be summarized in section 4.5.

[4] The detailed analysis result by means of likelihood rating (based on the exploitability and exposure rating of the weakness) and the resulting impact (based on several impact scenarios) for the concrete threat are left out since the ratings are not relevant for understanding the example and require the entire method to be fully understood.

4.3.1 Step 1) Determine Scope Areas

As a prerequisite for the development of an organization-specific SRS repository, the evaluation scope and scenario described in section 4.2 were used to determine the relevant scope areas as the first structuring layer (cf. section 3.3.1).

The output of step 1 is the scope area structure, which determines the scope of consideration, the work to be accomplished, the problem space to be analyzed, and the security topics to be covered when analyzing the problem space and developing SRs. To ensure the differentiation between typical aspects related to software, systems, and additional non-technical and organizational topics, the structure of scope areas proposed by Schmitt and Liggesmeyer in [101] was used.

For the proposed model, a distinction was made between the three scope areas software, system in technical environment, and system in organizational environment, as depicted in Figure 4.1[5]. These three scope areas ensure that the model can be used for the software, the system level, as well as aspects in the operational environment. Thus, it is ensured that scope areas can be mapped to typical security aspects and responsibilities in classical enterprise IT and also to typical product and solution business 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.

- **Software:** The heart of the model is the software itself. Typical security topics (sometimes also referred to as security quality factors [22]) are authentication, authorization, session management, data at rest security, or data in transit security. The typical outcome of the security requirements engineering phase for this scope area is a set of security requirements and trust assumptions, which are typically part of a software requirements specification document. Software is deployed on a system in its technical environment[6].
- **System in Technical Environment:** A system in its technical environment consists of relevant physical and technical conditions and objects that are relevant for the software. Examples of objects within the physical and technical environment are servers, the areas, buildings, and rooms where relevant system components are located, client devices, network(s), other neighboring systems or services relevant for the software. Furthermore, other security mechanisms and services protecting the infrastructure

[5] The potential fourth scope area 'development environment', which addresses all SDLC aspects required for the secure development of products, systems, and solutions are left out since SDLC aspects are beyond the scope of this evaluation (cf. section 4.2.2).

[6] Environment (similar to the definition of the term 'environment' in [85]) refers to procedures, conditions, and objects affecting the development, operation, and maintenance of a system.

are in scope. Typical security topics related to the system in its technical environment are, e.g., physical security, network security, secure software configuration, secure system configuration and hardening, malware protection.

- **System in Organizational Environment:** An organizational environment includes relevant organizational aspects as well as processes and procedures required to securely set up, operate, maintain, and use the software and the system in its technical environment. Examples of organizational aspects are, e.g., the issuance of mandatory security policies and guidelines, the set-up of a security organization with clearly defined roles and responsibilities, and the conduction of security training and awareness activities for employees and third-party personnel. Examples of operational, security-related processes and procedures include user management, privilege management, key management, vulnerability and patch management, incident management, and many more.

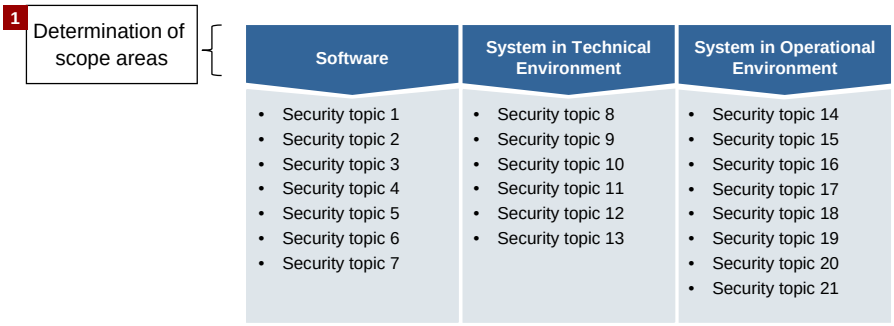


Fig. 4.1. Evaluation - Step 1: Determination of scope areas

4.3.2 Step 2) Determine Security Topics per Scope Area

To figure out a suitable structure of security topics for the determined scope areas in the evaluation scenario, the approach described in section 3.3.2 was applied.

(a) **Development of a generic structure blueprint:** First, a set of widely accepted IT and information security knowledge sources [47, 48, 83] was chosen, together with the structure (particularly the table of contents per SRS) of the incorporated SRSs in the evaluation scope. Their structure and incorporated SRS elements were analyzed and mapped to the three above-mentioned scope areas. This enabled the identification of common and recurring terminologies for security topics. The resulting structure of

security elements for each scope area served as a draft structure, which formed the basis for the detailed assignment of the SRS elements.

(b) **Development of a scenario-specific structure:** In the second sub-step, the SRS elements of the SRSs in the evaluation scenario were parsed. The primary focus was on prescriptive information and knowledge sources as well as on compliance obligations, since they mostly already provide a consistent and 'security-topic-like' structure. The SRSs were mapped and assigned to the structure as developed in sub-step (a). Elements that were not in the scope of the evaluation were discarded prior to the mapping. Most of the SRS elements could be assigned directly to the structure of the security elements. SRS elements that did not fit the existing structure of the security topics were noted down separately for later analysis. This is how the draft structure of the security topics was developed for each scope area that fits the SRSs within the scope of the evaluation.

(c) **Troubleshooting:** In the third sub-step, the SRS elements (from sub-step (b)) that did not fit into the existing structure were analyzed. The structure was either reorganized or additional security topic were created. During the evaluation, only some specific SRS elements primarily from the SRS 'ISEC information security control framework' [113], which did not fit directly into the existing structure, led to additional security topics, mostly because the originating SRS covered additional aspects in the scope area 'System in Technical Environment', which the other SRSs did not address. An example of such an SRS element is "Proper management and controlling of service providers must be in place", which led to the additional security topic 'provider and service level management'.

(d) **Quality assurance:** To arrive at a consistent and acceptable structure, several iterations of revisions were conducted, resulting in the structure of the scope areas and security topics shown in Figure 4.2.

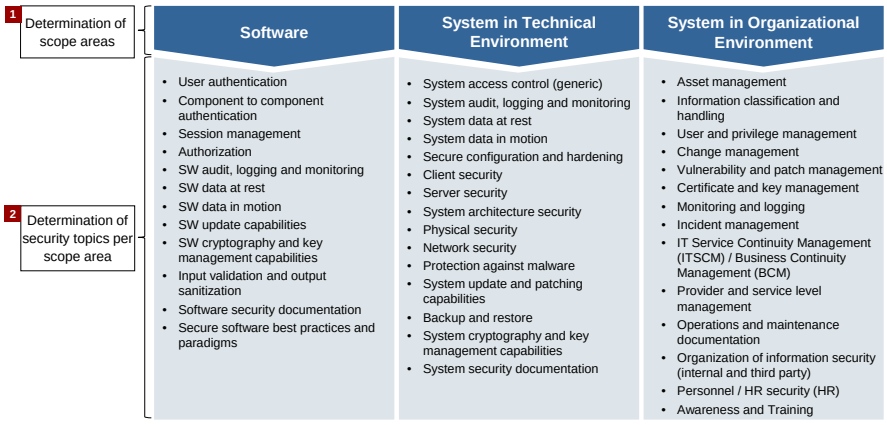


Fig. 4.2. Evaluation - Step 2: Determination of security topics per scope area

The more than 500 SRS elements from the 8 SRSs within the scope of the evaluation could be assigned to 12 security elements in tbe scope area 'Software', 15 security elements in the scope area 'System in Technical Environment', and 14 security elements in the scope area 'System in Organizational Environment'.

4.3.3 Step 3) Assign SRS Elements to Security Topics

In the third step, the elements of all SRSs in our scenario were analyzed and assigned to the security topics in the scope areas, as derived in step 2 (multiple assignments possible).[7]

Table 4.10 shows an example of the assignment of the SRS element 'CWE-759 - Use of a One-Way Hash without Salt' from the SRS 'Selected weaknesses from CWE' [127] to the respective security topics in their scope areas.

SRS - Selected weaknesses from CWE				
SRS Element		Assigned Scope Areas		
CWE ID and Weakness Title	Description Summary	Software	System in Technical Environment	System in Operational Environment
CWE-759 - Use of a One-Way Hash without a Salt	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 part of the input.	• User authentication • SW cryptography and key management	• Access control • System data at rest	

Table 4.10. Evaluation - Step 3: Assignment of SRS elements to security topics

The SRS element 'CWE-759: Use of One-Way Hash without Salt'[8] was assigned to the referenced scope areas and security topics because of the following rationale:

1. **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 security issue in the hashing algorithm, it is furthermore assigned to the security topic 'SW cryptography and key management'. Being assigned to both topics, the weakness

[7] Please note that the major difference from step 2b is that in this step, the complete mapping of all SRS elements to all relevant security topics is done, whereas in step 2b, the objective is only to create the draft structure of the security topics.

[8] Background information on password storage and hashing: Software providing a user-name- and password-based authentication scheme typically does not store user passwords in plaintext, but does so 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 is 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.

will be provided to SRE or security practitioners when querying the SRS repository for typical weaknesses with regard to user authentication as well as SW cryptography and key management.

2. **System in Technical Environment:** Security topics 'access control' and 'system data at rest'. Rationale: Hashed passwords are stored somewhere at the system level, e.g., in files of the operating system or somewhere in the database. Therefore, this weakness also relates to the security topics 'access control' and 'system data at rest' since passwords need to be secured against unauthorized access and security of data at rest (in this case passwords) must be ensured. When querying the SRS repository for typical weaknesses related to access control and system data at rest as well, this weakness will be shown to SRE or security practitioners. This enables them to check on the one hand, whether passwords at the system level are stored securely, and on the other hand, whether the access to the password hashes is adequately protected by proper access control mechanisms.
3. **System in Operational Environment:** Since the secure storage of passwords can only be designed and implemented at the software level and at the system level, no security element from this scope area was assigned.

4.3.4 Step 4) Establish Relationships between SRS Elements

Once a suitable structure of scope areas and security topics has been created and the respective SRS elements have been assigned to the security topics, the relationships between the various SRS elements must be established, following the generic relationship model depicted in Figure 3.5. In practice, this means that for each element of the SRS it needs to be analyzed and documented to which other SRS element a relationship exists. The mutual relationships between the various SRSs as a basis for the assignments are shown in sections 3.2.2 and 3.1.

For instance, for the threat 'Eavesdropping' all relationships to other relevant SRS elements in scope are documented, such as 'encryption of data during transport or logical and/or physical separation in a dedicated network' (which is an element of a prescriptive information and knowledge source, supporting the mitigation of the threat) and 'PII data transferred over networks shall be protected against unauthorized disclosure and access' (which is an affected raw requirement from a compliance obligation). Furthermore, helpful relationships between different diagnostic SRS elements can also be documented, e.g., to concretize the generic threat with weaknesses from CWE such as 'CWE-327: Use of a Broken or Risky Cryptographic Algorithm'. This makes it possible to provide all related weaknesses and vulnerabilities related to a particular threat in order to enhance the understanding of the problems to be solved.

Table 4.11 shows the documented relationships from the SRS element CWE-759 to all other relevant SRS elements with a relationship to this weakness in the instantiated repository.

SRS - Selected weaknesses from CWE							
SRS Element		SRS References					
CWE ID & Weakness Title	Description Summary	PSS threat and risk analysis questionnaire	Deliberate threats from ISO/IEC 27005	PSS secure architecture guidelines	ISEC secure software development policy	ISEC information security control framework	HDP data privacy controls
CWE-759 - Use of a One-Way Hash without a Salt	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 part of the input.	• SW-9: Insecure Password Storage	• ISO-37: Corruption of data • ISO-38: Illegal processing of data	• 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): Fail-secure Access control must be in place to restrict access to defined authorized / trained users • 402002-02-Authentication: - Single-factor authentication (i.e. something you know, something you have or something you are) following good security practices.	• P9: Password-based authentication shall be state-of-the-art.

Table 4.11. Evaluation - Step 4: Establishment of relationships between SRS elements

The rationale for the relationships established from SRS element CWE-759 to the other SRSs and their elements in scope is as follows:

Common Weakness Enumeration (CWE) is an SRS of the SRS category 'diagnostic information and knowledge sources'. The example shows that the weakness motivates several elements from the SRSs 'PSS threat and risk analysis questionnaire' and 'ISEC secure software development policy'. Furthermore, the weakness can be used to explain the reason for the implementation of proper access control and authentication mechanisms as part of the SRS 'ISEC information security control framework' and the implementation of a state-of-the-art password-based authentication mechanism as part of the SRS 'HDP data privacy controls'. Also, the SRS elements of the two other representatives from the SRS category 'diagnostic information and knowledge sources' are correlated to show that relationships between SRS elements of SRSs in the same SRS category can also be modeled, although this is not directly foreseen in the ER diagram, as shown in Figure 3.5.

Practical advice: Since the relationships between SRS elements are in most cases mutual, one would be well advised to directly document existing backwards relationships for each SRS element. As an example, the SRSs 'SW-9: Insecure Password Storage' and 'CWE-759 - Use of a One-Way Hash without a Salt' should have a mutual reference to each other since they are on a similar level of granularity and complement each other very nicely. From practical

experience, few exceptions exist where a backwards reference might not make sense[9], in particular if the relationship from an overly generic SRS element to other very specific SRS elements would not be understandable anymore for the practitioners using the SRS repository. As an example, for a practitioner is is well understandable, that if passwords are not stored securely (according to example CWE-759 'Use of a One-Way Hash without a Salt'), that 'corruption of data' (ISO-37 from SRS [49]) might result, if an adversary can get illegitimate access to data due to the leaked passwords. However, the other way round is more difficult.

4.4 Views and Benefits of the Instantiated SRS Repository

The following views, as already described in theory in section 3.4, can be invoked on the instantiated SRS repository in order to obtain helpful information from various SRSs and to support the elicitation of SRs.

4.4.1 Scope Area View

The scope area view shows the structure of determined security topics for the respective scope area. As explained in section 3.4, this view is the most generic and trivial one since it simply shows the structure of security elements for the queried scope area(s), as determined in step 2 of the instantiation approach (cf. section 3.3.2).

However, the scope area view provides a starting point and a good overview on all security topics to consider for the respective scope area when carrying out a project and developing security requirements. Thus, this view is to a certain degree comparable to the table of contents of a single security policy or standard, and similar to the quality factors proposed by Firesmith [22].

As an example, if a new software product is to be developed, the scope area view for scope area 'software' provides 12 security elements in our evaluation scenario, which include all software-related elements of all SRSs to be investigated, as depicted in Figure 4.3.

4.4.2 Security Topic View

The security topic view shows all SRS elements that are related to a specific security topic. Through this view, all elements of the SRSs within the scope of the evaluation can be queried that were assigned to this specific security topic.

[9] although this still is at the discretion of the people instantiating the SRS repository

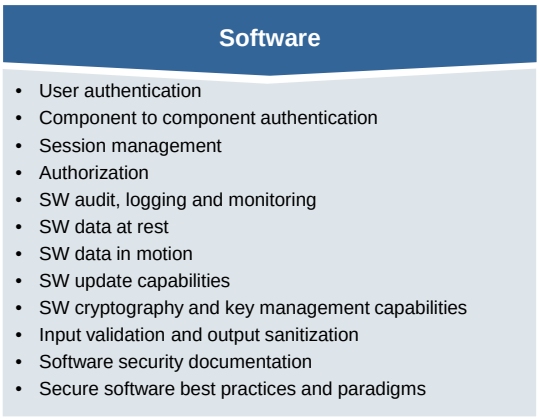


Fig. 4.3. Example: Scope area view for the scope area 'Software'

As an example of the security topic view 'user authentication' in the scope area 'Software', the following SRS elements are provided, as also depicted in Figure 4.4:

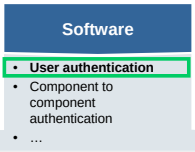
- 9 threats from the PSS threat and risk analysis questionnaire
- 5 weaknesses from the selected weaknesses from CWE
- 5 deliberate threats from ISO/IEC 27005
- 38 recommendations from the PSS secure architecture guidelines
- 7 recommendations from the ISEC secure software development policy
- 3 compliance obligations from the ISEC information security control framework
- 2 HDP data privacy controls.

The next example, depicted in Figure 4.5, shows the security topic view for 'system audit, logging and monitoring' in the scope area 'System in Technical Environment'.

While the SRS elements from the incorporated SRSs for the security element 'user authentication' show a quite homogeneous picture with regard to quantity and level of detail, the example for the security topic view 'system audit, logging and monitoring' shows that the number, scope and level of detail of the SRS elements from different SRSs assigned to the security topic can also vary.

The last example of an SRS element view presents a security topic view for 'provider and service level management' in the scope area 'System in Organizational Environment', where only the SRS 'ISEC information security control framework' contributes eight SRS elements to the view, as depicted in Figure 4.6.

Considering the defined SRS categories (cf. section 3.1) and the evaluation results shown above, several benefits arise from this view for practitioners, as it helps to:



Assigned SRS elements to security topic user authentication						
Diagnostic Information and Knowledge Sources			Prescriptive Information and Knowledge Sources		Compliance Obligations	
PSS threat and risk analysis questionnaire	**Selected weaknesses from CWE**	**Deliberate threats from ISO/IEC 27005**	**PSS secure architecture guidelines**	**ISEC secure software development policy**	**ISEC information security control framework**	**HDP data privacy controls**
• SW-1: Missing User Authentication • SW-2: Client-Side User Authentication • SW-3: Weak User Authentication • SW-4: Insecure Initial Passwords • SW-5: Hard-coded Passwords • SW-6: Insecure Password Change Functionality • SW-7: Kiosk Mode or Operating System Group Accounts • SW-8: Clear-Text Transmission of Passwords • SW-9: Insecure Password Storage	• CWE-304: Missing Critical Step in Authentication • CWE-306: Missing Authentication for Critical Function • CWE-602: Client-Side Enforcement of Server-Side Security • CWE-603: Use of Client-Side Authentication • CWE-759: Use of a One-Way Hash without a Salt • CWE-798: Use of Hard-coded Credentials	• ISO-19: Compromise of information - Remote spying • ISO-25: Compromise of information - Data from untrustworthy sources • ISO-34: Unauthorized actions - Unauthorised use of equipment • ISO-37: Unauthorised actions - Corruption of data • ISO-38: Unauthorised actions - Illegal processing of data	• A-UA-001-C001: Consider that users should be identified and authenticated with personal accounts. • 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 • A-UA-001-N001: Don't report the exact cause of a failed log on attempt. • ... • A-UA-002-D003: Do only use well-proven and secure algorithms for automatic password generation to avoid predictable sequences. • ...	• SSD-16: Use dedicated framework/library functions for password fields in user interfaces, for example password fields that mask the actual characters and that prevent content caching and auto-completion. • SSD-18: After a failed authentication attempt, do not report the exact cause of the failure: whether the user was unknown or whether the user existed but the password was incorrect. • SSD-19: If passwords are transmitted in an insecure way, for example by unencrypted email, the lifetime of those passwords must be limited appropriately and a password change must be enforced after first use. • ...	• 402001 – Access Control (general): Fail-secure Access control must be in place to restrict access to defined authorized / trained users • 402002-01- Explicit proof of identity required if the access is restricted to defined users, i.e. the digital identity is assignable to a person, department, etc. • 402002-02- Authentication: - Single-factor authentication (i.e. something you know, something you have or something you are) following good security practices. (Note: The example applies for "security Level 1")	• P8: Every access to network and user interfaces shall be authenticated. • P9: Password-based authentication shall be state-of-the-art. state-of-the-art password based user authentication comprises the following aspects. Each aspect shall be supported. a) The user shall be able to change his password. b) A configurable password policy, allowing to enforce - password changes at regular intervals, - complex passwords with sufficient length, - policy changeable by administration role only. c) Mechanisms to prevent password guessing, e.g. enforcing delays after failed authentication requests d) …

Fig. 4.4. Example: Security topic view for 'user authentication'

- Understand the problem space for each security topic and 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 it with software or system (security) architects' viewpoints.
- Identify identical or similar raw requirements from different compliance obligations to increase the efficiency for fulfilling these requirements.
- Incorporate results from previous applications of SRE methods to avoid the recurrence of security problems.
- Review the scope, quality, and completeness of incorporated SRSs by comparing assigned SRS elements for each security topic. The analysis result can reveal:

System in Technical Environment

- System access control (generic)
- **System audit, logging and monitoring**
- ...

Assigned SRS elements to security topic system audit, logging and monitoring						
Diagnostic Information and Knowledge Sources			Prescriptive Information and Knowledge Sources		Compliance Obligations	
PSS threat and risk analysis questionnaire	Selected weaknesses from CWE	Deliberate threats from ISO/IEC 27005	PSS secure architecture guidelines	ISEC secure software development policy	ISEC information security control framework	HDP data privacy controls
• SW-22: Insufficient Logging of Security-Relevant Events • SW-23: Sensitive Data in Log Files • SW-24: Missing Log Server Integration • CD-20: Lack of monitoring of security events	• N/A	• ISO-34: Unauthorised actions - Unauthorised use of equipment • ISO-37: Unauthorised actions - Corruption of data • ISO-38: Unauthorised actions - Illegal processing of data • ISO-40: Compromise of functions - Abuse of rights	• A-UA-001-D005: Do log successful and failed log on attempts centrally. • A-MWP-001-D001: Do (where permitted and possible) employ automated tools to continuously monitor workstations, servers ... • A-MWP-001-D008: Monitor for use and attempted use of external devices. • A-MWP-001-C004: Consider using automated monitoring tools for behavior-based anomaly detection to complement and enhance traditional signature-based detection. • A-NWS-003-D001: Do log at appropriate levels • ...	• N/A	• 402006-03: Unauthorized modifications are detectable, e.g. by a simple information versioning • 402007-01: Audit record with access control decisions must be in place (including access by privileged users); audit records must be protected	• P18: Logging of user management actions shall be provided • P19: Logging of PHI processing shall be provided • P20: Log files shall be protected against unauthorized access

Fig. 4.5. Example: Security topic view for 'system audit, logging and monitoring'

- Missing SRS elements in an SRS (i.e., SRS elements covered by one SRS but not by other SRSs).
- SRS elements of an SRS that might exceed the intended scope of an SRS (e.g., system-specific weaknesses, although the intended scope of an SRS is only on software)

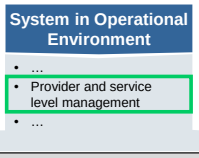
4.4.3 SRS Element View

The SRS element view shows for a single selected SRS element all other SRS elements with a relationship to it. 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 that help to understand and address it.

An SRS element view example for the SRS element "CWE-759 - Use of a One-Way Hash without a Salt" was already shown in Table 4.11.[10]

The following benefits arise from this view:

[10] Please note that step 4 of the instantiation approach forms the basis for the SRS element view, i.e., the view of relationships established between SRS elements.



Assigned SRS elements to security topic provider and service level management						
Diagnostic Information and Knowledge Sources			Prescriptive Information and Knowledge Sources		Compliance Obligations	
PSS threat and risk analysis questionnaire	Selected weaknesses from CWE	Deliberate threats from ISO/IEC 27005	PSS secure architecture guidelines	ISEC secure software development policy	ISEC information security control framework	HDP data privacy controls
• N/A	• N/A	• N/A	• N/A	• N/A	• 407001-01: Before selecting a service provider the risks from exposing the information to the service provider must be assessed • 407001-02: The provider develops/implements and/or operates security controls having the required security capabilities for level 1 • 407001-03: No obvious product/ service errors exist; flaws are addressed in a timely manner • 407001-04: A reasonable management and controlling of the service provider must be established. • 407001-05: Service levels, security, legal requirements and confidentiality agreements are part of the contracts • 407001-06: Regular monitoring and audits should be performed • 407002-02 Operations and development of Information processing systems (e.g. ITIL processes) • 407002-04: Standards for Provider's HR management / employment processes are applied	• N/A

Fig. 4.6. Example: Security topic view for 'provider and service level management'

- Possibility to view all related diagnostic information and knowledge sources, as well as 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 that 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 into an SRE method result (e.g., a weakness identified from a threat and risk analysis).

4.4.4 SRS Coverage View

The SRS coverage view is an additional view, which cannot be requested solely on the instantiated SRS repository, but requires additional information regarding the coverage (or consideration) of a particular SRS element in the specified set of SRs within a particular project. As a prerequisite, for each specified SR, the relationships to the SRS elements must be documented, to enable retrieving those SRSs that are already addressed or incorporated by the developed set of SRs. In other words, it must be documented which threats and weaknesses or raw requirements from compliance obligations are addressed or considered when an SR is specified.

The SRS coverage view allows checking the degree of coverage and completeness of SRS elements. Checks can be helpful:

- For an SRS: e.g. to reveal all threats or weaknesses, for which no SRs have been specified so far, and to check if for all raw requirements of a compliance obligation a SR does exist in the project-specific set of SRSs [11].
- For a security topic, e.g., to check whether all SRS element of a security topic are already incorporated.
- For an entire scope area: It can be analyzed if all relevant SRSs are addressed in the requirements specification, before it is handed over to the software or system architects.

4.5 Perceptions and Challenges Encountered during Evaluation

The proposed model for structuring and reusing SRSs as well as the instantiation approach could be applied successfully in practice within the defined evaluation scenario. Both the model and the instantiation approach were successfully applied to the incorporated SRSs, resulting in a Siemens-specific SRS repository based on the defined evaluation scenario. The various views on the instantiated SRS repository underline the practical applicability and usefulness of the contributions developed in this PhD thesis.

In this section, perceptions and challenges with a potential impact on the resulting instantiated SRS repository will be described that appeared during the instantiation of the Siemens-specific SRS repository. The perceptions and challenges reveal potential pitfalls to be avoided when instantiating the model and summarize potential adjustments or variants of the described model and the instantiation approach. For each challenge or problem, potential solutions are proposed.

4.5.1 General Influences on the Creation of an SRS Repository

Two general influences on the creation of an SRS repository were identified. First, the human factor and the subjectivity of the person(s) performing the instantiation, and second, the types and levels of detail of the incorporated SRSs in the resulting SRS repository.

[11] Please note that this view does not necessarily allow a statement on the fulfillment of a compliance obligation since the extent of coverage for each SRS element as well as the appropriateness and effectiveness of the measures designed to fulfill the requirement cannot be incorporated into this view.

Human Factor and Subjectivity

Challenge and perception: The instantiation approach is performed by humans. Therefore, subjectivity and personal experience will influence the resulting SRS repository, particularly if no preferred SRS (e.g., an internal information security policy or an external international IT security standard) exists as the primary structure for the SRS repository, but a completely new structure of scope areas and security topics is to be developed.

Depending on the person(s) and their knowledge, experience, and background, the resulting SRS repository will look different in terms of structure and terminology.

As an example of how the human factor and subjectivity can influence a resulting SRS repository, particularly the structure and terminology of the security topics, 'access control' is used. Access control as used in ISO/IEC 27001 [47] incorporates all kinds of organizational and technical aspects related to controlling access to information, including authentication, authorization, user and privilege management. In contrast, other more technical standards and glossaries consider access control synonymous with authorization, which is a significantly smaller scope than that used in ISO/IEC 27001.

Another terminology-related example concerns the denomination of the 'bad guy', the person who tries to compromise the confidentiality, integrity, and availability of information, software, or systems. Terminologies used in SRSs range from attacker via adversary to threat agent.

It can be summarized that the fewer constraints and preconditions are specified for the instantiation of the SRS repository, the higher the likelihood for an increased level of personal preferences and subjectivity in the instantiated repository.

The human factor and subjectivity therefore has influence on

- the number and granularity of the scope areas and security elements being determined, and
- the terminology used for scope areas, security elements, and other terms.

Proposed solution: Although the described instantiation approach recommends as a prerequisite that decisions regarding the structure of scope areas and security topics as well as the preferred terminology should be derived from many prerequisites to be clarified for the instantiation and the SRS to be incorporated, the evaluation revealed that there still remains a certain degree of freedom for the instantiating person(s) with regard to structure and terminology. It is therefore highly recommended to thoroughly clarify all preconditions (cf. section 3.3.1) and to be as specific as possible. Furthermore, early reviews with relevant stakeholders should be done for each instantiation approach step in order to align the chosen structure and terminology at an early point in time.

Security Requirements Sources

Challenge and perception: Most of the incorporated SRSs in the evaluation scenario were used straightforward during the instantiation. However, the entire SRS 'deliberate threats from ISO/IEC 27005' [49]) and a limited number of SRS elements from the other SRSs in scope caused problems due to increased instantiation effort and reduced the usability and benefits of the views on the instantiated repository.

Problems with SRS elements occurred during the evaluation if:

- SRS elements were specified in an overly generic manner;
- The level of detail of a single SRS element or an entire SRS differed a lot in comparison to the other incorporated SRSs.

An example of an exceptional, overly generic SRS element from [113] is "SSD-3: The security of the Siemens IT infrastructure must not be jeopardized by the software or one of its components." While most of the other SRS elements from this SRS could reasonably be used to determine and assign scope areas and security topics and to establish beneficial and manageable relationships to other SRS elements, such overly generic SRS elements were troublesome during the evaluation.

The same problem regarding overly generic SRS elements applies to the entire SRS 'deliberate threats from ISO/IEC 27005' [49]. As an example, threats like "corruption of data" or "unauthorized access to information" are almost relevant for all security topics. Furthermore, they have relationships to almost all other SRS elements within the evaluation scope. Therefore, besides the challenge that overly generic SRS elements do not support the determination of scope areas and security topics, they result in hugely nonsensical assignments of SRS elements to security topics and lead to a massively increased number of relationships between the SRS elements. If from a very generic threat like 'corruption of data' hundreds of SRS elements from other more specific SRSs are referenced, the rationale for a relationship might not be clear anymore for the practitioners.

Proposed solution: Preferably, only a homogeneous set of SRS elements should be selected as a basis for the instantiation of an SRS repository. If overly generic SRS elements are to be incorporated anyway though (e.g., due to business decisions), the following approach turned out to overcome the abovementioned challenges to a certain degree:

- Overly generic SRS elements should be left out for the determination of the scope areas (step 1) and security topics (step 2) in the instantiation approach since they do not provide any added value for the determination of the SRS repository structure.
- The assignment of SRS elements to security topics can be done according to step 3 of the instantiation approach.

- To limit the number of relationships between SRS elements with very different levels of detail, an indirection from a generic SRS element to other SRS elements with a medium level of detail can be done, which in turn references the more detailed SRS elements.

The following example shows an indirection via SRS elements with an intermediate level of detail (here 402001 – Access Control) to more detailed SRS elements (here CWE-759, CWE-798, ...):

- Corruption of data
  –> References SRS element with medium level of detail
  - 402001 – Access Control (general): Fail-secure access control must be in place to restrict access to defined authorized / trained users
    –> References SRS elements with high level of detail
    - CWE-759 - Use of a One-Way Hash without a Salt
    - CWE-798 - Use of Hard-Coded Credentials
    - ...
    - SW-1 - Missing User Authentication
    - SW-2 - Client-Side User Authentication
    - ...
    - SSD-37: Do not rely on the secrecy of key material or passwords in the client.
    - SSD-43: Require server authentication, if possible.
    - ..

4.5.2 Instantiation Approach

The instantiation approach worked properly as described in the evaluation. However, some particularities and challenges should be regarded related to the assignment of SRS elements to security topics (i.e. step 3 of the instantiation approach) and the rationale used to assign SRS elements to security topics and to establish the relationships between the SRS elements.

Assignment of SRS Elements to Security Topics

Challenge and perception: A single SRS element can, and often should, be assigned to multiple security topics in the determined scope areas. Examples of multiple assignments of an SRS element to security topics are presented in sections 4.3 and 4.4.3 (cf. Table 4.10). During the evaluation the vast majority of SRS elements could be assigned to security topics without any problems.

For security experts, this assignment step is straightforward. However, in cases where SRS elements primarily address technical aspects that nevertheless implicitly also require operations and management aspects, also security topics in the scope area ‘System in Organizational Environment’ might also be of interest for security and SRE practitioners.

As an example of the evaluation scenario, SRS elements related to the security topic 'secure configuration and hardening' also implicitly incorporate related organizational aspects, meaning that processes or procedures must be in place that ensure that secure configuration and hardening is actually done in practice, since secure configuration and hardening is not only a stable condition that needs to be achieved once, but requires regular activities. Although there is no concrete organizational or procedural SRS element for secure configuration and hardening within the SRS elements within the scope of the evaluation, it nevertheless might be of interest to create a security topic in the scope area 'System in Organizational Environment' and assign technical SRS elements related to secure configuration and hardening. Alternatively, new SRS elements could also be created in order to complement the incorporated SRS elements.

Another challenge concerning the assignment of SRS elements to security topics arises if SRS elements are not 'atomic', meaning that a single SRS element addresses multiple aspects of the problem space, which would normally have to be separated into several SRS elements. Such SRS elements would have to be assigned to different security topics, potentially also in different scope areas, although they would also cover additional aspects that do not fit to a particular security topic.

The following example shows this circumstance: In the good case, SRS elements are clearly separated, as the logging example shows. The SRS element "A-MWP-001-D001: Do (where permitted and possible) employ automated tools to continuously monitor workstations, servers and mobile devices for active, up-to-date anti-malware protection with anti-virus, anti-spyware, personal firewalls and host-based IPS functionality..." can be clearly assigned to the security topic 'system audit, logging and monitoring' in the scope area 'System in Technical Environment', since it requires a technical system capability. The SRS element "A-NWS-001-D004: Regularly check the log events: "If you're going to log it, read it"." can be clearly assigned to the security topic 'monitoring and logging' in the scope area 'System in Organizational Environment', since it clearly refers to a procedure and the organization that regularly has to check the log files. However, there are also mixed forms of SRS elements that have both a technical and an organizational aspect and therefore normally should be treated as separate SRS elements. The SRS element "A-MWP-001-D008: Do limit the use of external devices to those that are really necessary for the operation and maintenance of the solution/product. Monitor for use and attempted use of external devices." is a good example of such a mixed SRS element.

Proposed solution: A general solution for the first challenge concerning the potential multiple assignments of a single SRS element cannot be provided, since the decision about which SRS element to assign where is a very individual one that depends on various factors such as the quality and level of detail of the incorporated SRS element, the use cases to be supported, as well as the

structure of scope areas and security topics created. Therefore, it needs to be decided and aligned on a case-wise basis how to handle each SRS element.

Regarding the latter challenge concerning SRS elements that incorporate multiple security aspects in different scope areas, either a separation of the SRS elements into multiple SRS elements can be done prior to assigning it to security topics and establishing the relationships to the other SRS elements (which is the recommended option), or multiple assignments of SRS elements to security topics are an option that can be selected. In the latter case, suitable comments should be added to the concerned SRS element to indicate the circumstance.

Rationale for Assignments of SRS Elements to Security Topics

To successfully create an SRS repository, several organizational and skill-related prerequisites are required, as explained in section 3.3.1. Profound security knowledge and expertise is required to identify, classify, compare, correlate, and use SRS elements. Only security experts should assign the SRS elements to security topics in order to ensure that the assignments are correct and complete. During the evaluation, a challenge related to this process step was detected when discussing the resulting assignments of SRS elements to security topics with SRE practitioners. The problem in some cases was that the rationale for the assigment of a security element to a security topic is in some (rather exceptional) cases not clear to the practitioners using the SRS repository. Therefore, if in such cases the rationale is not properly documented in the SRS repository, the SRE practitioners cannot draw their conclusions regarding the mapping and might wonder about the reason for the assignments and possibly omit certain aspects due to this lack of understanding.

Therefore, assignments of SRS elements to security topics must either be understandable without requiring security expert knowledge, or the description for each SRS element needs to provide this information. If this is not the case, the rationale for the mapping should be documented somewhere for the practitioners.

Proposed solution: In step 3 of the instantiation approach, an additional review of the assignments between SRS elements and security topics should be conducted, ideally performed by non-security experts, to check if the assignments are understandable. If not, the assignment and the description text of the concerned SRS elements must be revised and explanatory text should be added to the repository, e.g., as part of the description text for each SRS element.

4.5.3 Structural Challenges

Challenge and perception: During the evaluation, instantiation steps 1 and 2 (cf. section 3.3.2) worked as intended and turned out to be pragmatic and

useful for creating a structure of scope areas and security topics, as depicted in Figure 4.2.

In step 3 of the instantiation approach, it was also straightforward to assign SRS elements to the determined structure of security topics, except for the challenges explained in section 4.5.2. However, the other way around, i.e., the security topic views, can be more problematic in some cases, especially if the security topic selected (during step 2 of the instantiation approach) was too coarse-grained. When performing a security topic view, in some cases (mostly for security topics with many assigned SRS elements), the variety of different security aspects and the level of details can be very high, particularly if the scope and level of detail of the incorporated SRS varies (cf. section 4.5.1). As an example, for the security topic 'user authentication', various security aspects ranging from good practices for password storage to certificate-based authentication aspects are covered (cf. Figure 4.4). In such a case, SRE practitioners would have to go through all assigned SRS elements and could not avoid unnecessary SRS elements, e.g., SRS elements on secure password storage (salted hashes, etc.) if a strong authentication mechanism is prescribed anyway.

Proposed solution: To solve such cases, three alternatives are proposed.

- Revise the determined (flat) structure of the SRS elements and split the affected single security topics into multiple security topics (on the same hierarchy level). Note: This has the disadvantage that the number of security topics increases and that potential preferred security element structures, e.g. aimed at compliance with a specific standard, cannot be kept.
- Adapt the model by adding an additional hierarchy level below the security topics, e.g., by subdividing the security topic 'user authentication' into weak / single factor and strong / multi-factor authentication. Note: This has the drawback that the (ER) model as described in section 3.2 needs to be extended or redesigned; also, it becomes more complicated for practitioners to use.
- Tag each SRS element using additional meta-information. By doing this, additional selection criteria could be added for each SRS element to only show relevant SRS elements, using more complex database queries. Note: The downside of this solution alternative is that additional tags and meta-information have to be determined and maintained together with the general structure of the security topics.

Relationships between SRS elements

Challenge and perception: The vast majority of the overall effort of the instantiation of an organization-specific SRS repository lies in step 4, the establishment of relationships between SRS elements, which took approximately 70% of the overall time. In theory, each element from a SRS has to be compared with all SRS elements from the other SRSs in scope, creating massive effort, particularly if a high number of SRS elements is to be incorporated.

Proposed solution: The high effort for establishing the relationships between the SRS elements can be compensated to a certain degree:

In general, detailed knowledge about the incorporated SRSs (which topics are covered where in which SRS) should be available present in the team instantiating the model.

In addition, also the search space for SRS elements to be compared can be decreased in most cases. Based on step 3, instead of sequentially parsing through all possible SRS elements in scope, they can be filtered only for SRS elements from the other SRSs which are assigned to the identical security topics. In most cases, particularly with regard to the more technical SRS elements, this filter helps a lot to decrease the complexity and the number of SRS elements to be compared, and nearly all possible SRS elements with actual relationships can be properly assigned.

However, for very generic SRS elements e.g., 'deliberate threats from ISO/IEC 27005' [49] where a clear mapping to user authentication or logging is not obvious, huge effort for the mapping needs to be done. In such cases, where the relationships are not obvious and no searches for keywords (logging, log file, monitoring etc.) can be used, again detailed security knowledge is required.

5

Summary and Future Work

"Most creative work is a process of people passing ideas and inspirations from the past into the future and adding their own creativity along the way."
Joichi Ito

5.1 Summary

Several challenges with regard to SRE are stated in the literature that can also be observed in practice. Often, the maturity of SR specifications is poor (e.g., [96,133]). This situation is aggravated by the lack of knowledge and skills regarding security and SRE (e.g., [23]), the fact that SRE methods and techniques are rarely applied for the development of SRs (e.g., [96]), and the fact that security information and knowledge sources are seldom used in practice (e.g., [29]). Besides these shortcomings, there is an increasing number of internal and external compliance obligations as important SRSs to be considered when developing SRs.

Therefore, support must be provided to SRE practitioners in order to incorporate information and knowledge sources as well as compliance obligations in a well-structured and reusable way to mitigate the lack of security skills and knowledge and to avoid unnecessary effort for identifying, understanding, and correlating SRS elements on a project-wise basis.

The research activities described in this PhD thesis focus on the various types of SRSs, particularly on their categorization and structuring in order to provide them to practitioners in a reusable form. The goal is to address the above-mentioned challenges by supporting organizations with the identification and structuring of relevant SRSs and, based on this, the creation of an organization-specific SRS repository that fulfills the particular needs of an organization. The repository is intended to aid requirements engineers and development teams during the elicitation and analysis of SRs by providing SRSs in a reusable form and making the relationships between the various SRSs understandable.

To reach this goal, the research effort during the course of this PhD thesis was spent on the following aspects:

- Identification, structuring, and specification of a consistent classification for the most important SRSs.
- Development of a generic model that can be used for structuring and reusing relevant SRSs.
- Specification of criteria and a structured approach for instantiating the generic model for a given scenario in order to create an organization-specific SRS repository.

The resulting SRS repository is intended to be used by development teams during the elicitation and analysis of security requirements with the aim of understanding the security problem space, incorporating all relevant requirements sources, and avoiding unnecessary effort for identifying, understanding, and correlating applicable security requirements sources on a project-wise basis.

The following contributions were elaborated and presented as part of this PhD thesis:

In section 3.1, a categorization of reusable, security-specific SRSs to be considered in the model was introduced, which was derived from information and perceptions obtained from the state of the art and the state of the practice reviews, presented in chapter 2. The SRS categories were used for the development of an extended security engineering approach (cf. section 3.1.1) and discussed in the context of the Twin Teaks model (cf. section 3.1.2).

Using the SRS categories, a meta-model and a related ER diagram for structuring and reusing SRSs including its components were presented in section 3.2. The model serves as a basis for the instantiation of an organization-specific SRS repository. An overview and explanation of the model were provided in section 3.2.1, followed by the associated ER diagram for the model in section 3.2.2.

To instantiate the model for a concrete scenario, general prerequisites and the approach for instantiating the model along with all theoretical process steps, inputs, and outputs for creating the SRS repository were described in section 3.3.

To use and benefit from an instantiated SRS repository, four different model views were introduced and explained in section 3.4, which provide all relevant information for understanding and analyzing the security problem space, and for specifying security requirements. These views enable automated analyzability of an instantiated SRS repository view by providing SRE practitioners with useful automated queries.

All developed contributions, i.e., the SRS categories, the model, and the instantiation approach were evaluated in a comprehensive real-world industrial scenario at Siemens. The development of the three scope areas was published and discussed in [101]. The theoretical model and the instantiation approach

were published in [102]. The model, the instantiation approach, and the practical evaluation approach were presented in [99, 103, 104]. Related aspects in the area of SRE methods were presented in [105], which were also used in part for the evaluation of the model. The evaluation incorporated over 500 elements from eight different SRSs, covering at least one representative per SRS category. Using several examples from the instantiated repository, it was shown that the proposed model can be used for a variety of different SRSs (cf. chapter 4). The instantiation in the described scenario showed that the model is capable of fulfilling the desired business demands and research hypotheses specified in theory in section 1.3. Perceptions and challenges encountered during the evaluation were explained in section 4.5 to highlight potential pitfalls and support future instantiations of the model.

The desired model capabilities (i.e., business demands and research hypotheses) are fulfilled by the following contributions and evaluation results:

H1 - Scope: The model can be used for software SRs and for the system level, as well as for technical, physical, and organizational aspects.

In general, the model and its described components (i.e., scope areas, security topics, and SRSs, as described in section 3.2) are flexible enough to develop arbitrary structures as needed by a particular organization.

Based on the business demand to support software, the system level, and related technical, physical, and organizational aspects, a structure of the following three scope areas was developed during the evaluation (cf. section 4.1).

- Software
- System in Technical Environment
- System in Organizational Environment

The structure was determined in step 1 of the instantiation approach and was successfully used as the basis for the subsequent steps of the instantiation approach. For each scope area, a limited number of relevant security topics were determined, and for each security topic, all SRS elements in the evaluation scope could be assigned (i.e., step 2, cf. section 3.3.2).

As the SRS elements within the scope of the evaluation contained a variety of technical, physical, and organizational aspects, even from different SRS categories, the first business demand and research hypothesis (H1) is fulfilled.

H2 - Flexibility: The model is flexible enough to structure (most of) the relevant SRSs.

Using step 2 of the proposed instantiation approach (cf. section 3.3.2), a suitable structure of security topics for the three determined scope areas could be created (cf. section 4.3). The developed structure was flexible enough to assign the over 500 SRS elements in the evaluation scenario to the determined security topics. Example assignments were demonstrated using security topics views, cf. section 4.4.2 and Table 4.10.

Moreover, the relationships between most of the over 500 SRS elements in the evaluation scenario could be identified and documented. Relationships between SRS elements were demonstrated applying the SRS element view on three example SRS elements (cf. section 4.4.3 and Table 4.11).

While the model in general can be considered flexible enough to structure any kind of SRS element, the evaluation revealed increased effort for the instantiation, particularly for the assignment of SRS elements to security topics and the determination of relationships between SRS elements, if overly generic SRS elements were used and the level of detail of the incorporated SRS elements differed. This decreases the usability and the benefits of the views on the instantiated repository since overly generic SRS elements are assigned to almost any security topic and, moreover, relationships to almost all SRS elements exist (cf. section 4.5.1). Business demand and research hypothesis H2 can therefore also be considered fulfilled. However, the evaluation showed that certain constraints regarding SRSs exist since the effort for the instantiation and the usability of the resulting SRS repository depend on a homogeneous level of detail of the incorporated SRS elements (cf. section 4.5.1).

H3 - Reuse: The reuse of information and knowledge is supported to avoid unnecessary effort for the identification, classification, comparison, correlation, and use of elements from different SRSs.

The evaluation showed that the instantiation approach (cf. section 4.3) works well for the identification, classification, comparison, and correlation of elements from different SRSs. In combination with the presented views on the instantiated SRS repository (cf. section 4.4), information can be requested in a consistent, complete, and correlated manner by SRE practitioners.

Reusability implies completeness and correctness of the instantiated SRS repository and efficient and easy use of the resulting repository for practitioners. Completeness and correctness of the assignments and relationships of the SRS elements must be achieved through proper instantiation of the model and the resulting SRS repository. Only if the organizational and skill-related prerequisites for the instantiation of the model are fulfilled (cf. section 3.3.1), sufficient security knowledge and expertise is available to know in detail and understand the incorporated SRSs, and to identify, classify, compare, correlate, and use security information and knowledge sources. This allows providing practitioners with a comprehensive and consistent set of SRS elements with properly assigned security topics and reasonable relationships to other SRS elements, without requiring additional significant effort for identifying, comparing, and consolidating different SRSs in order to support the elicitation of SRs.

Regarding completeness and correctness difficulties with regard to overly generic SRS elements or SRS elements with very different levels of detail may exist and need to be resolved (cf. section 4.5.1). Moreover, completeness and correctness depend to a certain degree on the person(s) performing the

instantiation approach due to the human factor and subjectivity (cf. section 4.5.1).

Efficiency and usability of the instantiated repository are achieved through the provision of four different views for a variety of different purposes (cf. section 3.4). These views enable practitioners to query for the desired information from the instantiated model and support the elicitation of SRs, as shown in various examples in section 4.4.

H4 - Relationships between SRSs: The relationships between different kinds of security information and knowledge (e.g., diagnostic vs. prescriptive) shall be understandable.

Based on step 4 in the instantiation approach (cf. sections 3.3.2), relevant relationships between the SRS elements were identified and documented during the evaluation (cf. sections 4.3.4). Relationships between diagnostic and prescriptive security information and knowledge sources could be established, along with SRSs from the other defined SRS categories (cg. section 3.1.3), namely compliance obligations as well as results and artifacts from SRE methods. SRE practitioners can make use of the established relationships using the SRS elements view (cf. sections 3.4.3 and 4.4.3).

To ensure that the established relationships can actually be understood by SRE practitioners, three aspects should be ensured, as emphasized during the evaluation:

- The level of detail of the incorporated SRS elements must not differ too much in order to have a homogeneous set of relationships between the SRS elements (cf. sections 4.5.1 and 4.5.3).
- A consistent and aligned terminology should be used throughout the instantiated repository (cf. section 4.5.1).
- Regular and early reviews of the assignments between SRS elements and security topics should be conducted to check if the assignments and relationships are understandable; this check should ideally be performed by staff without security expertise (cf. sections 4.5.1 and 4.5.2). If relationships are not intuitively understandable, the assignment and the description text of the concerned SRS elements must be revised and explanatory text should be added to the repository.

H5 - Quality and baseline security: It is possible to support the check for quality and completeness of SRs in terms of 'baseline security', covering the most prevalent security aspects of the problem space.

The desired model capability regarding baseline security is supported with the model through the use of a set of mandatory SRSs (e.g., the most relevant threats, weaknesses, vulnerabilities, and attacks) as input for the instantiation of the model. Thereby, mandatory SRSs become part of the resulting SRS repository, which ensures that the most prevalent aspects of the problem space

are considered when eliciting security requirements. Using the SRS coverage views (see sections 3.4.4 and 4.4.4), it can be verified which of the defined minimum SRSs are already incorporated in the specified set of SRs.

Although the model by itself is capable of providing and checking that the defined mandatory SRS elements were used during the development of SRs, the quality of the resulting SRs and the resulting security measures aimed at fulfilling the SRs and addressing the related threats, weaknesses, and vulnerabilities cannot be ensured solely by the model but must be supported by additional semantic reviews conducted by human professionals. Nevertheless, the basis for developing SRs can be improved through the following two aspects. First, the higher the quality of the SRS repository, the better the basis for the development of SRs. Since the instantiation approach is performed by a skilled security expert and the developed repository is reviewed and improved several times, quality assurance is done to ensure that the terminology used is consistent and that the assignments and relationships are understandable, complete, and correct. Second, the additional content of the assigned and correlated SRSs extends the basis for the elicitation of SRs. The number of SRS elements that can be used for the development of SRs is not only limited to an isolated SRS, but includes all information and knowledge from the incorporated SRSs. The respective relationships and views can be used for the elicitation of SRs. Obviously, the added value for SRE practitioners depends on the quantity and quality of information included in the SRSs and therefore cannot be claimed for any case. However, as the example for user authentication in the evaluation scenario showed (cf. section 4.4.2), multiple additional information and knowledge sources are provided that were not considered by or provided at such a level of detail by the other SRSs. As an example, the two compliance obligations for user authentication from the SRS 'HDP data privacy controls' are enhanced by nine threats from the PSS threat and risk analysis questionnaire, five weaknesses from the selected weaknesses from CWE, five deliberate threats from ISO/IEC 27005, 38 recommendations from the PSS secure architecture guidelines, and seven recommendations from the ISEC secure software development policy. As can be easily recognized in Figure 4.4, the additionally provided SRS elements offer much more information and knowledge that can be used for the elicitation of the security requirements than if only the two raw requirements from the HDP data privacy controls had been used.

In addition to the use of the instantiated repository and the SRS coverage views, additional reviews of the developed SRs and the resulting SR specification should be performed as part of the SR(E) process in a project.

The contributions presented in this PhD thesis help to address the shortcomings presented in section 1.2 in the following ways:

- The lack of knowledge and skills regarding security and SRE can be mitigated through the use of an instantiated organization-specific, understandable, and correct SRS repository. The repository supports the reuse

of information and knowledge of the incorporated SRS elements without requiring IT security expert knowledge to identify, classify, compare, correlate, and use elements from different SRSs.

- The problem that information and knowledge sources are seldom used in practice cannot be solved solely by the model or by an instantiated SRS repository, since the identification and use of SRSs must be anchored in an organization, e.g., in development processes and quality gates. Nevertheless, the instantiated SRS repository provides all SRSs that were identified and considered to be relevant for the organization at the time of model instantiation. Therefore, it provides all the SRSs incorporated in the resulting SRS repository that address the hurdle faced by SRE practitioners when identifying SRSs on a project-wise basis.
- The fact that SRE methods and techniques are rarely applied for the development of SRs cannot be improved by the presented model and approach. However, diagnostic information and knowledge sources (cf. section 2.4.1) can be used as valuable input, particularly for the application of analysis-oriented SRE methods (cf. section 2.5) such as attack trees or abuse cases.
- The increasing number of internal and external compliance obligations as an important SRS to be considered when developing SRs is addressed by taking into account a dedicated SRS category (cf. section 3.1.3) for compliance obligations.

5.2 Comparison with Related Work

The following relations exist between the results of this PhD thesis and the SRE methods, processes, and frameworks presented in the state of the art and the state of the practice (cf. chapter 2).

In general, the proposed model and the instantiated repository fit to any SRE process (e.g., SQUARE [70]) and method (cf. section 2.5.1) that follows a security-engineering-based approach as presented in section 2.2.1. In other words, the contributions of this PhD thesis support and complement any SRE process and method in which SRs are derived from threats, weaknesses, and vulnerabilities since the model is able to incorporate and provide SRS elements from diagnostic information and knowledge sources as well as from previous results from analysis-oriented SRE methods. As an example, a threat and risk analysis (as a representative of an analysis-oriented SRE method) can be supported using the 'diagnostic information and knowledge sources' of an instantiated SRS repository. Relevant SRS elements can be obtained and applied simply in workshops using the available views without requiring lengthy identification of potential threats and weaknesses during a workshop.

Moreover, since not only diagnostic information and knowledge sources are used in the model, it is also possible to obtain related prescriptive information

and knowledge sources and compliance obligations from the instantiated SRS repository for a given SRS element using the SRS element view.

The idea of developing a reusable repository or knowledge base was already pursued in SIREN [130], SREP [73], and the work by Dikanski and Abeck [16]. However, there are several differences concerning the incorporated sources to be used to fill the repository, and the general approach and concept applied. The model and instantiation approach pursued in this thesis focus on structuring and reusing multiple existing SRSs with the aim of creating an organization-specific structure of scope areas, security topics, and relationships between the incorporated SRS elements. The model and the instantiation approach can therefore be used flexibly for any kind of business, for arbitrary SRSs, and for different kinds of SRE methods. Therefore, the approach is not limited or specifically designed for a single method or security standard, but is capable of incorporating different kinds of SRSs relevant for an organization.

In contrast, existing reuse-oriented approaches either focus on the reuse of dedicated security standards, or they utilize predefined SRE methods for the identification of relevant threats or the creation of reusable security requirements for the repository. SIREN is oriented towards the structure and information provided by MAGERIT and SREP primarily focuses on packages and profiles from Common Criteria.

Only Dikanski and Abeck [16] and Mellado et al. [73] foresee the incorporation of a broader focus of different SRSs. However, the model and examples provided by Dikanski and Abeck are in an early state. Although Mellado et al. recommend the inclusion of legal, statutory, regulatory, and contractual requirements, they leave open how this should be done in practice. Therefore, no approach existed to date that is based upon a categorization of possible reusable SRSs, nor a model and approach that is capable of flexibly structuring, correlating and providing SRS elements in the form of an organization-specific SRS repository.

As an example, in the model described in this thesis, the focus is not only on threats (as a representative of diagnostic information and knowledge) but weaknesses and vulnerabilities are also incorporated. Moreover, prescriptive information and knowledge sources, compliance obligations, and results from SRE methods are possible sources to be integrated.

Another related research direction are security requirements templates [25] and security requirements patterns [11, 135]. The major difference to them is that the focus of the work presented in this thesis is on providing a suitable SRS repository (i.e., an organization-specific structure of scope areas and security topics combined with the relationships between the elements and views on the instantiated model) to understand the security problem space. An instantiated SRS repository intentionally does not prescribe any textual patterns, templates, or other means of pre-formulated reusable SRs since the intention is to leave the desired level of detail, the wording, and the number of SRs

to the discretion of the SRE practitioners. This work is intended to enhance the understanding of the security problem space for different scope areas and topics and to provide the required dependencies between the elements from different SRSs of different categories in order to allow the practitioners to understand what needs to be avoided and addressed in the problem space (i.e., diagnostic information and knowledge), what kind of principles and controls help to address the problem space (i.e., prescriptive information and knowledge), and which compliance obligations need to be satisfied and for which reason. Nevertheless, the instantiated model is a potential source for the efficient development of security requirements templates or security requirements patterns as it offers relevant information and knowledge in a well-structured form.

5.3 Limitations

The following limitations related to the presented model and evaluation scenario exist:

- **Support for security requirements elicitation:** The proposed model is intended to complement existing SRE and security engineering processes by structuring and reusing SRSs to support the elicitation of SRs.
  Nevertheless, there are further undoubted standard activities in SRE processes (cf. section 2.1.4), such as the specification of security objectives and the identification and description of valuable assets to be protected, which should of course be done in addition since they are also required for later SRE phases. It is therefore highly recommended embedding the organization-specific SRS repository into the organization-specific (S)RE process and aligning it with existing practices and ontologies used in the organization.
- **Placement of the model in the 'SRE world':** There is no common agreement on what a security requirement is [18]. As a consequence, there are various, partly contradictory, definitions of SRs and thus also different SR specifications. Existing approaches do not agree on whether the requirements should be limited to high-level security goals or whether they should be detailed into security measures. This induces uncertainty among requirements engineering practitioners about what constitutes good practice in security requirements engineering. Although the proposed model of scope areas and security topics is generic and thus could be used for different levels of granularity, the aim was to design a security-engineering-focused SRE model in which the elicitation of SRs is necessitated rather by concrete SRSs than by high-level security objectives.
  Moreover, we see the strengths of the model particularly in the context of incremental development approaches (cf. section 3.1.2). It has not been investigated so far, whether or not the model can be used in settings where

only high-level security goals or security core principles (e.g., confidentiality, integrity, and availability) are used as model structure elements. However, during the evaluation, several challenges arose concerning some overly generic SRS elements that did not really fit into the instantiated repository. Therefore, the model as well as the presented evaluation scenario must be placed more on the concrete, functional side of SRE.

- **The level of detail of scenario-specific SRSs must not deviate too much:** During the evaluation of the model, the experience was made that the assignment of SRS elements to security topics and the establishment of relationships between SRS elements that differ significantly in their level of granularity was very challenging and caused many unhelpful references. In particular, the mapping related to diagnostic information and knowledge sources might not always make sense or may ultimately result in extra effort for revising either the incorporated SRSs or the developed structure. Thus, usability and the final output depend on the incorporated SRSs, particularly on their level of detail and the scope they address. It is therefore highly recommended to carefully select SRSs that are not mandatory and do not provide an obvious added value in order to avoid huge discrepancy among SRSs. For instance, in the presented evaluation, the list of high-level deliberate threats from [49] were of limited added value for the instantiated SRS repository.
- **Evaluation of the instantiated SRS repository against further quality indicators:** No evaluation was performed on the instantiated SRS repository against additional quality indicators such as improved efficiency or enhanced quality. It was discussed and agreed that for this thesis, the focus of the evaluation would be on applicability and on challenges with regard to the model and the instantiation approach. Further aspects that highly depend on the quality and completeness of the incorporated SRSs or on the individual knowledge and experience of the test persons were left out on purpose.
  Consequently, other practical evaluations such as the comparison of SR specifications developed by test persons without security skills using the instantiated repository and SR specifications created by security experts without using the instantiated repository were not conducted, since the informative value of such results would have been very limited with regard to the contributions of this PhD thesis.

5.4 Potential Future Work

- **Classification scheme for SRSs:** To ensure that a homogeneous set of SRSs is available and that SRSs elements fit together prior to the instantiation of an SRSs repository, a classification scheme for the scope and level of detail of the incorporated SRSs could be developed.

- **Further instantiations and refinement of the model and of the instantiation approach:** Currently, the model and the instantiation approach were evaluated in a single comprehensive industrial scenario with a broad scope. To get more evaluation results and to identify potential additional challenges or hints to consider when instantiating the model, further evaluations could be conducted. Topics of interest that could be evaluated are:
  - Instantiation of the model by persons who did not use the model and instantiation approach before.
  - Exclusive use of very generic, high-level SRSs for the model instantiation.
  - Elaboration of very specific scenarios and use cases such as embedded devices or web applications.
- **Derivation of common, recurring scenarios, use cases, and structures that can be applied by multiple organizations:** Based on further instantiations, common scenarios, use cases, and recurring structures could be derived. This would allow deriving typical scope areas and security topics for specific scenarios and use cases (e.g., web application development) and proposed assignments and relationships of typical external SRSs for a particular scenario, and providing them to the user community.

Abbreviations

BDSG	Bundesdatenschutzgesetz, Federal Data Protection Act
CC	Common Criteria
cf.	compare, see also
CVE	Common Vulnerabilities and Exposures
CWE	Common Weakness Enumeration
ER	Entity Relationship
e.g.	for example (from latin 'exempli gratia')
FDA	Food and Drug Administration
GUI	Graphical User Interface
HDP	Healthcare Data Protection
HIPAA	Health Insurance Portability and Accountability Act
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IT	Information Technology
ISA	International Society of Automation
ISEC	Corporate Information Security
ISO	International Organization for Standardization
i.e.	that is (from latin 'id est')
MAGERIT	Methodology for Information Systems Risk Analysis and Management
NIST	National Institute of Standards and Technology
N/A	Not Available
PhD	Philosophiae Doctor (Doctor of Philosophy)
PHI	Protected Health Information
PII	Personally Identifiable Information
PMBOK	Project Management Body of Knowledge
PSS	Product and Solution Security
RE	Requirements Engineering
SR	Security Requirement
SecRAT	Security Requirements Analysis Templates
SDLC	System Development Life Cycle

SIREN	SImple REuse of software requiremeNts
SQuaRE	Systems and software Quality Requirements and Evaluation
SQUARE	Security Quality Requirements Engineering
SRE	Security Requirements Engineering
SREP	Security Requirements Engineering Process
SRR	Security Resources Repository
SRS	Security Requirements Source
SSE-CMM	Secure Systems Engineering Capability Maturity Model
Std.	Standard
STRIDE	Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, Elevation of privilege
STS-ml	Socio-Technical System modeling language
TRA	Threat and Risk Analysis
UML	Unified Modeling Language
UMLSec	Unified Modeling Language extended to express Security-relevant information
XP	eXtreme Programming

References

[1] Abran, A., Moore, J.W.: Guide to the Software Engineering Body of Knowledge (SWEBOK(R)). IEEE Computer Society Press (2005)

[2] Balzert, H.: Lehrbuch der Softwaretechnik. Spektrum Akademischer Verlag, Heidelberg, Germany, 3 edn. (2009)

[3] Barnum, S., McGraw, G.: Knowledge for Software Security. IEEE Security and Privacy 3(2), 74–78 (2005)

[4] Basili, V.R.: The Experimental Paradigm in Software Engineering. In: Proceedings of the International Workshop on Experimental Software Engineering Issues: Critical Assessment and Future Directions. pp. 3–12. Springer-Verlag, London, UK (1993)

[5] Boehm, B.W.: Software engineering economics. Prentice Hall, Englewood Cliffs and N.J (1981)

[6] Boström, G., Wäyrynen, J., Bodén, M., Beznosov, K., Kruchten, P.: Extending XP Practices to Support Security Requirements Engineering. In: Proceedings of the 2006 International Workshop on Software Engineering for Secure Systems (SESS'06). pp. 11–18. ACM, New York, USA (2006)

[7] BSI: Informationssicherheit und IT-Grundschutz: BSI-Standards 100-1, 100-2 und 100-3. Bundesanzeiger-Verlag, Cologne, Germany, 2 edn. (2008)

[8] Burge, J.E., Brown, D.C.: NFRs: Fact or Fiction? (2002), retrieved from: http://digitalcommons.wpi.edu/computerscience-pubs/121, visited on 2016-06-08

[9] Charette, R.N.: Why Software Fails. IEEE Spectrum 42(9), 42–49 (Sep 2005)

[10] Crook, R., Ince, D., Lin, L., Nuseibeh, B.: Security Requirements Engineering: When Anti-Requirements Hit the Fan. In: Proceedings of IEEE Joint International Conference on Requirements Engineering (RE'02). pp. 203–205. IEEE Computer Society, Washington, DC, USA (2002)

[11] Daramola, O., Sindre, G., Stalhane, T.: Pattern-based Security Requirements Specification using Ontologies and Boilerplates. In: Proceed-

ings of IEEE Second International Workshop on Requirements Patterns (RePa). pp. 54–59 (2012)
[12] Davis, A.M.: Operational Prototyping: A New Development Approach. IEEE Software 9(5), 70–78 (1992)
[13] Davis, A.M.: Achieving Quality in Software Requirements. Software Quality Professional 1(3), 37–44 (1999)
[14] Department of Homeland Security: Secure Software Assurance: A Curriculum Guide to the Common Body of Knowledge to Produce, Acquire, and Sustain Secure Software (2007), https://buildsecurityin.us-cert.gov/sites/default/files/CurriculumGuideToTheCBK-1.pdf, visited on 2016-06-07
[15] Department of Homeland Security: Requirements Analysis for Secure Software (2012), https://buildsecurityin.us-cert.gov/swa/, visited on 2016-06-07
[16] Dikanski, A., Abeck, S.: Towards a Reuse-oriented Security Engineering for Web-based Applications and Services. In: Proceedings of the 7th International Conference on Internet and Web Applications and Services (ICIW 2012) (2012)
[17] Dörr, J.: Elicitation of a Complete Set of Non-Functional Requirements. Ph.D. thesis, University of Kaiserslautern (2011)
[18] Dubois, E., Mouratidis, H.: Guest Editorial: Security Requirements Engineering: Past, Present and Future. Requirements Engineering 15(1), 1–5 (2010)
[19] Emam, K.E., Koru, A.G.: A Replicated Survey of IT Software Project Failures. IEEE Software 25(5), 84–90 (2008)
[20] Fabian, B., Gürses, S., Heisel, M., Santen, T., Schmidt, H.: A Comparison of Security Requirements Engineering Methods. Requirements Engineering 15(1), 7–40 (2010)
[21] FIPS PUB 199, Standards for Security Categorization of Federal Information and Information Systems. National Institute of Standards & Technology (NIST), Gaithersburg, MD, United States (2004)
[22] Firesmith, D.: Analyzing and Specifying Reusable Security Requirements. In: Proceedings of the 2nd Workshop on Requirements Engineering for High-Availability Systems (RHAS'03). pp. 7–11 (2003)
[23] Firesmith, D.: Engineering Security Requirements. Journal of Object Technology 2(1), 53–68 (2003)
[24] Firesmith, D.: Security Use Cases. Journal of Object Technology 2(3), 53–64 (2003)
[25] Firesmith, D.: Specifying Reusable Security Requirements. Journal of Object Technology 3(1), 61–75 (2004)
[26] Glinz, M., Wieringa, R.J.: Guest Editors' Introduction: Stakeholders in Requirements Engineering. IEEE Software 24(2), 18–20 (2007)
[27] Golnaz, E.: A Goal Oriented Approach for Modeling and Analyzing Security Trade-Offs with Knowledge Support. Master's thesis, University of Toronto (2007)

[28] Golnaz, E.: Making Trade-Offs among Security and Other Requirements during System Design. Ph.D. thesis, University of Toronto (2012)
[29] Golnaz, Elahi, Yu, Eric, Tong Li, Lin Liu: Security Requirements Engineering in the Wild: A Survey of Common Practices. In: Proceedings of the 35th Annual IEEE International Computer Software and Applications Conference (COMPSAC'11). pp. 314–319 (2011)
[30] Grandison, T., Sloman, M.: A Survey of Trust in Internet Applications. IEEE Communications Surveys & Tutorials 3(4), 2–16 (2000)
[31] Greenberg, A.: A Tax On Buggy Software (2008), http://www.forbes.com/2008/06/26/rice-cyber-security-tech-security-cx_ag_0626rice.html, visited on 2016-06-07
[32] Haley, C., Laney, R., Moffett, J., Nuseibeh, B.: Security Requirements Engineering: A Framework for Representation and Analysis. IEEE Transactions on Software Engineering 34(1), 133–153 (2008)
[33] Haley, C.B., Laney, R.C., Moffett, J.D., Nuseibeh, B.: The Effect of Trust Assumptions on the Elaboration of Security Requirements. In: Proceedings of the 12th IEEE International Requirements Engineering Conference (RE'04), pp. 102–111 (2004)
[34] Haley, C.B., Laney, R.C., Moffett, J.D., Nuseibeh, B.: Using Trust Assumptions with Security Requirements. Requirements Engineering 11(2), 138–151 (2006)
[35] Haley, C.B., Laney, R.C., Nuseibeh, B.: Deriving Security Requirements from Crosscutting Threat Descriptions. In: Proceedings of the 3rd International Conference on Aspect-oriented Software Development (AOSD'04). pp. 112–121 (2004)
[36] IEC 62443-3-3:2013, Industrial communication networks - Network and system security - Part 3-3: System security requirements and security levels (2013)
[37] IEEE Std 1220:2005 (Revision of IEEE Std 1220:1998), IEEE Standard for Application and Management of the Systems Engineering Process. Standard, IEEE (2005)
[38] IEEE Std 1233:1998, IEEE Guide for Developing System Requirements Specifications. Standard, IEEE (1998)
[39] IEEE Std 830:1998, IEEE Recommended Practice for Software Requirements Specifications. Standard, IEEE (1998)
[40] Information Security Forum: Information Risk Analysis Methodology 2 (IRAM2), https://www.securityforum.org/tool/information-risk-assessment-methodology-iram2/, visited on 2016-06-07
[41] ISA99 committee website, http://isa99.isa.org/ISA99%20Wiki/Home.aspx, visited on 2016-06-07
[42] ISO/IEC 13335-1:2004, Information technology – Security techniques – Management of information and communications technology security – Part 1: Concepts and models for information and communications

technology security management. International Organization for Standardization (ISO), Geneva, Switzerland (2004)
[43] ISO/IEC 15408-1:2009, Information technology – Security techniques – Evaluation criteria for IT security – Part 1: Introduction and general model. International Organization for Standardization (ISO), Geneva, Switzerland (2009)
[44] ISO/IEC 15408-2:2008, Information technology – Security techniques – Evaluation criteria for IT security – Part 2: Security functional components. International Organization for Standardization (ISO), Geneva, Switzerland (2008)
[45] ISO/IEC 15408-3:2008, Information technology – Security techniques – Evaluation criteria for IT security – Part 3: Security assurance components. International Organization for Standardization (ISO), Geneva, Switzerland (2008)
[46] ISO/IEC 25030:2007, Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) – System and software quality models. International Organization for Standardization (ISO), Geneva, Switzerland (2011)
[47] ISO/IEC 27001:2013, Information technology – Security techniques – Information security management systems – Requirements. International Organization for Standardization (ISO), Geneva, Switzerland (2013)
[48] ISO/IEC 27002:2013, Information technology – Security techniques – Code of practice for information security controls. International Organization for Standardization (ISO), Geneva, Switzerland (2013)
[49] ISO/IEC 27005:2011, Information technology – Security techniques – Information security risk management. International Organization for Standardization (ISO), Geneva, Switzerland (2011)
[50] ISO/IEC 9126-1:2001, Software engineering – Product quality – Part 1: Quality model. International Organization for Standardization (ISO), Geneva, Switzerland (2001)
[51] ISO/IEC/IEEE 24765:2010, Systems and software engineering – Vocabulary. International Organization for Standardization (ISO), Geneva, Switzerland (2010)
[52] ISO/IEC/IEEE 29148:2011, Systems and software engineering – Life cycle processes – Requirements engineering. International Organization for Standardization (ISO), Geneva, Switzerland (2011)
[53] Jackson, M.J.: Problem Frames: Analysing and Structuring Software Development Problems. Addison-Wesley/ACM Press, Harlow (2001)
[54] Jarke, M., Bui, X.T., Carroll, J.M.: Scenario Management: An Interdisciplinary Approach. Requirements Engineering Journal 3(3), 155–173 (1999)
[55] Jürjens, J.: Towards Development of Secure Systems Using UMLsec. In: Proceedings of the 4th International Conference on Fundamental Approaches to Software Engineering (FASE '01). pp. 187–200. Springer-Verlag, London, UK (2001)

[56] Kang, K.C., Cohen, S.G., Hess, J.A., Novak, W.E., Peterson, A.S.: Feature-Oriented Domain Analysis (FODA) feasibility study. Carnegie Mellon University, Software Engineering Institute, Pittsburgh (1990)
[57] Kotonya, G., Sommerville, I.: Requirements Engineering: Processes and Techniques. J. Wiley, Chichester and New York (1998)
[58] van Lamsweerde, A.: Goal-oriented Requirements Engineering: A Guided Tour. In: Proceedings of the 5th IEEE International Symposium on Requirements Engineering (RE'01). pp. 249–262 (2001)
[59] van Lamsweerde, A.: Goal-Driven Requirements Engineering: the KAOS Approach, `https://www.info.ucl.ac.be/~avl/gore.php`, visited on 2016-06-07
[60] van Lamsweerde, A.: Elaborating Security Requirements by Construction of Intentional Anti-Models. In: Proceedings of the 26th International Conference on Software Engineering (ICSE'04). pp. 148–157. IEEE Computer Society, Washington, USA (2004)
[61] Lin, L., Nuseibeh, B., Ince, D., Jackson, M., Moffett, J.: Introducing Abuse Frames for Analysing Security Requirements. In: Proceedings of the 11th IEEE International Requirements Engineering Conference (RE'03). pp. 371–372 (2003)
[62] Lodderstedt, T., Basin, D.A., Doser, J.: SecureUML: A UML-Based Modeling Language for Model-Driven Security. In: Proceedings of the 5th International Conference on The Unified Modeling Language (UML'02). pp. 426–441. London, UK (2002)
[63] Mannio, M., Nikula, U.: Requirements Elicitation Using a Combination of Prototypes and Scenarios. Research Reports / Telecom Business Research Center Lappeenranta, Lappeenranta University of Technology, Lappeenranta (2001)
[64] McDermott, J., Fox, C.: Using Abuse Case Models for Security Requirements Analysis. In: Proceedings of the 15th Annual Computer Security Applications Conference (ACSAC'99). pp. 55–64 (1999)
[65] McGraw, G.: Testing for security during development: Why we should scrap penetrate-and-patch. Aerospace and Electronic Systems Magazine, IEEE 13(4), 13–15 (1998)
[66] Mead, N.R.: Requirements Engineering for Survivable Systems. Tech. Rep. CMU/SEI-2003-TN-013, Carnegie Mellon University, Pittsburgh (2003)
[67] Mead, N.R.: Requirements Elicitation Introduction (2006), `https://buildsecurityin.us-cert.gov/bsi/articles/best-practices/requirements/533-BSI.html`, visited on 2016-06-07
[68] Mead, N.R.: Security Requirements Engineering (2006), `https://buildsecurityin.us-cert.gov/bsi/articles/best-practices/requirements/243-BSI.html`, visited on 2016-06-07
[69] Mead, N.R., Hough, E.D., Stehney, T.R.: Security Quality Requirements (SQUARE) Methodology. Tech. Rep. CMU/SEI-2005-TR-009, Carnegie Mellon University, Pittsburgh (2005)

[70] Mead, N.R., Viswanathan, V., Padmanabhan, D., Raveendran, A.: Incorporating Security Quality Requirements Engineering (SQUARE) into Standard Life-Cycle Models. Tech. Rep. CMU/SEI-2008-TN-006, Carnegie Mellon University, Pittsburgh (2008)

[71] Meier, J.: Web Application Security Engineering. IEEE Security & Privacy Magazine 4(4), 16–24 (2006)

[72] Mellado, D., Blanco, C., Sánchez, L.E., Fernández-Medina, E.: A Systematic Review of Security Requirements Engineering. Computer Standards & Interfaces 4(Volume 32), 153–165 (2010)

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

[74] Ministerio de Administrationes Públicas: MAGERIT – Version 2: Methodology for Information Systems Risk Analysis and Management: II - Catalogue of Elements (2006)

[75] Mouratidis, H., Giorgini, P., Manso, G., Philp, I.: A Natural Extension of Tropos Methodology for Modelling Security. In: Proceedings of the Workshop on Agent-oriented Methodologies, pp. 91–103 (2002)

[76] Mouratidis, H., Giorgini, P., Manson, G.: An Ontology for Modelling Security: The Tropos Approach. In: Proceedings of the 7th International Conference on Knowledge-Based Intelligent Information and Engineering Systems (KES'03). Lecture Notes in Computer Science, vol. 2773, pp. 1387–1394. Springer Berlin Heidelberg, Oxford, UK (2003)

[77] Mouratidis, H., Giorgini, P., Manson, G.: When Security Meets Software Engineering: A Case of Modelling Secure Information Systems. Information Systems 30(8), 609–629 (2005)

[78] Myagmar, S., Lee, A., Yurcik, W.: Threat Modeling as a Basis for Security Requirements. In: Proceedings of the 2005 ACM Workshop on Storage Security and Survivability (StorageSS'05). pp. 94–102. ACM Press, New York, USA (2005)

[79] Mylopoulos, J., Chung, L., Nixon, B.: Representing and Using Nonfunctional Requirements: A Process-oriented Approach. IEEE Transactions on Software Engineering 18(6), 488–497 (1992)

[80] Mylopoulos, J., Chung, L., Yu, E.: From Object-oriented to Goal-oriented Requirements Analysis. Commun. ACM 42(1), 31–37 (1999)

[81] NIST Special Publication 800-30, Guide for Conducting Risk Assessments. National Institute of Standards & Technology (NIST), Gaithersburg, MD, United States, revision 1 edn. (2012)

[82] NIST Special Publication 800-39, Managing Information Security Risk: Organization, Mission, and Information System View. National Institute of Standards & Technology (NIST), Gaithersburg, MD, United States (2011)

[83] NIST Special Publication 800-53, Security and Privacy Controls for Federal Information Systems and Organizations. National Institute of Stan-

dards & Technology (NIST), Gaithersburg, MD, United States, revision 4 edn. (2013)
[84] NIST Special Publications (SP) 800s - Computer Security, http://csrc.nist.gov/publications/PubsSPs.html, visited on 2016-06-07
[85] NISTIR 7298, Glossary of Key Information Security Terms. National Institute of Standards & Technology (NIST), Gaithersburg, MD, United States, 2 edn. (2013)
[86] Nuseibeh, B.: Weaving the Software Development Process Between Requirements and Architectures. In: Proceedings of the 1st International Workshop From Requirements to Architectures. pp. 12–19. Toronto, Ontario, Canada (2001)
[87] Nuseibeh, B.: Weaving Together Requirements and Architectures. Computer 34(3), 115–117 (2001)
[88] OWASP: CLASP Security Principles, https://www.owasp.org/index.php/CLASP_Security_Principles, visited on 2016-06-08
[89] OWASP: Security Principle, https://www.owasp.org/index.php/Category:Principle, visited on 2016-06-08
[90] Oxford Dictionaries: "requirement", http://oxforddictionaries.com/definition/english/requirement?q=requirement, visited on 2016-06-08
[91] Paja, E., Dalpiaz, F., Poggianella, M., Roberti, P., Giorgini, P.: Modelling Security Requirements in Socio-Technical Systems with STS-Tool. In: Forum of the 24th International Conference on Advanced Information Systems Engineering (CAiSE'12 Forum), pp. 155–162 (2012)
[92] Project Management Institute: A Guide to the Project Management Body of Knowledge: (PMBOK Guide). Newton Square and Pa, 3 edn. (2004)
[93] QFD Institute: QFD: Quality Function Deployment, http://www.qfdi.org/, visited on 2016-06-08
[94] Riaz, M., Williams, L.: Security Requirements Patterns: Understanding the Science Behind the Art of Pattern Writing. In: Proceedings of the 2nd IEEE International Workshop on Requirements Patterns (RePa'12. pp. 29–34 (2012)
[95] Royce, W.W.: Managing the Development of Large Software Systems: Concepts and Techniques. In: Proceedings of the 9th International Conference on Software Engineering (ICSE'87). pp. 328–338. IEEE Computer Society Press, Los Alamitos, CA, USA (1987)
[96] Salini, P., Kanmani, S.: Survey and Analysis on Security Requirements Engineering. Computers & Electrical Engineering 38(6), 1785–1797 (2012)
[97] SANS: Critical Security Controls, http://www.sans.org/critical-security-controls/, visited on 2016-06-08
[98] Sauter, V.L.: What is Prototyping? (2012), http://www.umsl.edu/~sauterv/analysis/prototyping/proto.html, visited on 2016-06-07

[99] Schmitt, C.: An Instantiated Model for Structuring and Reusing Security Requirements Sources at Siemens. In: Siemens Software Excellence Conference (2015)
[100] Schmitt, C., Bloecher, U., Barthold, J.: Secure Architecture Guidelines (2013), restricted, for Siemens-internal use only
[101] Schmitt, C., Liggesmeyer, P.: Implications of the Operational Environmental on Software Security Requirements Engineering. In: Proceedings of the 11th International Workshop on Security in Information Systems (WOSIS'14), pp. 63–74. SCITEPRESS, Lisbon, Portugal (2014)
[102] Schmitt, C., Liggesmeyer, P.: A Model for Structuring and Reusing Security Requirements Sources and Security Requirements. In: In Proceedings of the 1st Workshop on Continuous Requirements Engineering (CRE'15), pp. 28–37. Essen, Germany (2015)
[103] Schmitt, C., Liggesmeyer, P.: Getting grip on security requirements elicitation by structuring and reusing security requirements sources. In: Complex Systems Informatics and Modeling (CSIMQ), vol. 3, pp. 15–34 (2015)
[104] Schmitt, C., Liggesmeyer, P.: Instantiating a Model for Structuring and Reusing Security Requirements Sources. In: Proceedings of the 2nd IEEE Workshop on Evolving Security and Privacy Requirements Engineering (ESPRE'15). pp. 25–30. Ottawa, Canada (2015)
[105] Schmitt, C., Maidl, M., Seltzsam, S., Wirsz, N.: Effektive Analyse von IT-Security-Risiken in einem heterogenen Produkt- und Lösungsportfolio. In: OBJEKTspektrum, vol. 4, pp. 14–18. SIGS Datacom (2015)
[106] Schneier, B.: Attack Trees. In: Dr. Dobb's Journal of Software Tools, vol. 24, pp. 21–29 (1999)
[107] Schneier, B.: Secrets and Lies: Digital Security in a Networked World. John Wiley, New York (2000)
[108] Schumacher, M.: Security Patterns: Integrating Security and Systems Engineering. Wiley Series in Software Design Patterns, John Wiley & Sons, Chichester and England and Hoboken and NJ (2006)
[109] Schumacher, M.: Security patterns: Integrating security and systems engineering. Wiley series in software design patterns, John Wiley & Sons, Chichester and England and Hoboken and NJ (2006)
[110] Seltzsam, S., Maidl, M., Schmitt, C.: PSS Threat and Risk Analysis Questionnaire (2014), restricted, for Siemens-internal use only
[111] Sethi, A., Barnum, S.: Introduction to Attack Patterns (May 14, 2013), `https://buildsecurityin.us-cert.gov/articles/knowledge/attack-patterns/introduction-to-attack-patterns`, visited on 2016-06-07
[112] Siemens Healthcare Division - Data Privacy Department: Data Privacy Requirements for PHI (2013), restricted, for Siemens-internal use only
[113] Siemens Information Security (ISEC) Department: IT Asset Classification and Protection (ACP) Framework (2013), restricted, for Siemens-internal use only

[114] Siemens Information Security (ISEC) Department: Secure Software Development Policy (2013), restricted, for Siemens-internal use only
[115] Sindre, G., Firesmith, D., Opdahl, A.L.: A Reuse-Based Approach to Determining Security Requirements. In: Proceedings of the 9th International Workshop on Requirements Engineering: Foundation for Software Quality (REFSQ'03) (2003)
[116] Sindre, G., Opdahl, A.L.: Eliciting Security Requirements by Misuse Cases. In: Proceedings of the 37th Conference on Techniques of Object-Oriented Languages and Systems, TOOLS Pacific 2000, pp. 120–131 (2000)
[117] Sindre, G., Opdahl, A.L.: Eliciting Security Requirements with Misuse Cases. Requirements Engineering 10(1), 34–44 (2005)
[118] Slavin, R., Lehker, J.M., Niu, J., Breaux, T.D.: Managing Security Requirements Patterns Using Feature Diagram Hierarchies. In: Tony Gorschek, Robyn R. Lutz (eds.) Proceedings of the 22nd International Requirements Engineering Conference (RE'14). pp. 193–202. IEEE, Karlskrona, Sweden (2014)
[119] Sommerville, I.: Software Engineering. Addison-Wesley, Munich, 8 edn. (2007)
[120] Souag, A., Salinesi, C., Comyn-Wattiau, I.: Ontologies for security requirements: A literature survey and classification. In: Advanced Information Systems Engineering Workshops, pp. 61–69. Springer Berlin Heidelberg (2012)
[121] Sutcliffe, A.: Scenario-Based Requirements Engineering. In: Proceedings of the 11th IEEE International Conference on Requirements Engineering (RE'03). pp. 320–330. IEEE Computer Society, Washington, USA (2003)
[122] Swartout, W., Balzer, R.: On the Inevitable Intertwining of Specification and Implementation. Communications of the ACM 25(7), 438–440 (1982)
[123] Swiderski, F., Snyder, W.: Threat Modeling. Microsoft Press, Redmond and Wash (2004)
[124] Systems Designers Scientific: CORE—The Method: User Manual. SD-Scicon, Camberley, England (1986)
[125] The MITRE Corporation: Common Attack Pattern Enumeration and Classification (CAPEC) Library (2012), `http://capec.mitre.org/`, visited on 2016-06-07
[126] The MITRE Corporation: Common Vulnerabilities and Exposures (CVE) (December 09, 2013), `http://cve.mitre.org/`, visited on 2016-06-08
[127] The MITRE Corporation: Common Weakness Enumeration (CWE) (December 11, 2013), `http://cwe.mitre.org/`, visited on 2016-06-07
[128] Tipton, H.F.: Official (ISC)2 guide to the CISSP CBK. CRC Press, Boca Raton and FL, 2 edn. (2010)

[129] Tondel, I., Jaatun, M., Meland, P.: Security Requirements for the Rest of Us: A Survey. IEEE Software 25, 20–27 (2008)

[130] Toval, A., Nicolás, J., Moros, B., García, F.: Requirements Reuse for Improving Information Systems Security: A Practitioner's Approach: SIREN. Requirements Engineering 6, 205–219 (2001)

[131] Viega, J., Kohno, T., Potter, B.: Trust (and Mistrust) in Secure Applications. Communications of the ACM 44(2), 31–36 (2001)

[132] Viega, J., McGraw, G.: Building Secure Software: How to Avoid Security Problems the Right Way. Addison-Wesley Professional Computing Series, Addison-Wesley, Boston (2002)

[133] Wilander, J., Gustavsson, J.: Security Requirements – A Field Study of Current Practice. In: E-Proceedings of the Symposium on Requirements Engineering for Information Security, in conjunction with the 13th IEEE International Requirements Engineering Conference (RE'05) (2005)

[134] Winograd, T., McKinley, H.L., Oh, L., Colon, M., McGibbon, T., Fedchak, E., Vienneau, R.: Software Security Assurance: A State-of-the Art Report (SOAR). Information Assurance Technology Analysis Center, Herndon and Virginia (2007)

[135] Withall, S.: Software Requirement Patterns. Safari Books Online, Microsoft Press, Redmond, USA (2007)

[136] Zowghi, D., Coulin, C.: Requirements Elicitation: A Survey of Techniques, Approaches, and Tools. In: Aurum, A., Wohlin, C. (eds.) Engineering and Managing Software Requirements, pp. 19–46. Springer (2005)

Curriculum Vitae

Zur Person

Name:	Christian Erich Schmitt
Adresse:	Saherrstr. 56, 80689 München
Email:	schmitt.chr@gmx.net
Nationalität:	Deutsch

Aus- und Weiterbildung

2012 - heute:	Doktorand, TU Kaiserslautern, Lehrstuhl Software Engineering: Dependability.
2007 - 2011:	Master of Applied IT Security, Ruhr-Universität Bochum, Abschlussnote: Sehr gut (86%).
2002 - 2005:	Diplomingenieur für Informationstechnik (BA), Berufsakademie Mannheim, Abschlussnote: Sehr gut (1,4).
2001 - 2002:	Zivildienst.
1992 - 2001:	Max-Born-Gymnasium, Germering, Abschluss: Hochschulreife.

Berufserfahrung

Seit April 2017:	Leiter IT-Sicherheitsmanagement, BMW AG.
2015 - 2017:	Referent IT Security, BMW AG.
2014 - 2015:	Senior Key Expert Consultant, Siemens Corporate Technology.
2011 - 2014:	Consultant, Siemens Corporate Technology.
2009 - 2011:	Senior Management Consultant, Siemens IT Solutions and Services Consulting.
2007 - 2009:	Management Consultant, Siemens IT Solutions and Services Consulting.
2005 - 2007:	Consultant, Siemens IT Solutions and Services Consulting.